Cisco Security Manager 4.4 API 仕様

（バージョン 1.1）

このマニュアルでは、Cisco Security Manager 4.4 North Bound API での CSM のメッセージ交換、XML スキーマ、クライアント/サーバ動作の仕様について説明します。

バージョン 1.0 発行日：2012 年 6 月 14 日
バージョン 1.0 改訂日：2012 年 7 月 10 日（セクション 8 にサンプル プログラムを追加）

バージョン 1.1 発行日：2013 年 3 月 22 日（バージョン 1.0 に Unified および Trustsec サポートを追加）

目次

図のリスト

表のリスト

1　概要

このマニュアルでは、Cisco Security Manager（CSM）API のプロトコルの説明と仕様、CSM サーバ上の API および CSM サーバ（インフラストラクチャ デバイス）自体を使用する CSM クライアント製品の動作要件を示します。また、CSM NorthBound（NB）API によって伝送されるメッセージ コンテンツの基盤となる XML スキーマを示します。

CSM NB API は、ネットワーク セキュリティの設定情報、さらにこうした設定に対する今後公開する変更のイベントを参照するクライアント製品で使用するように設計されています。

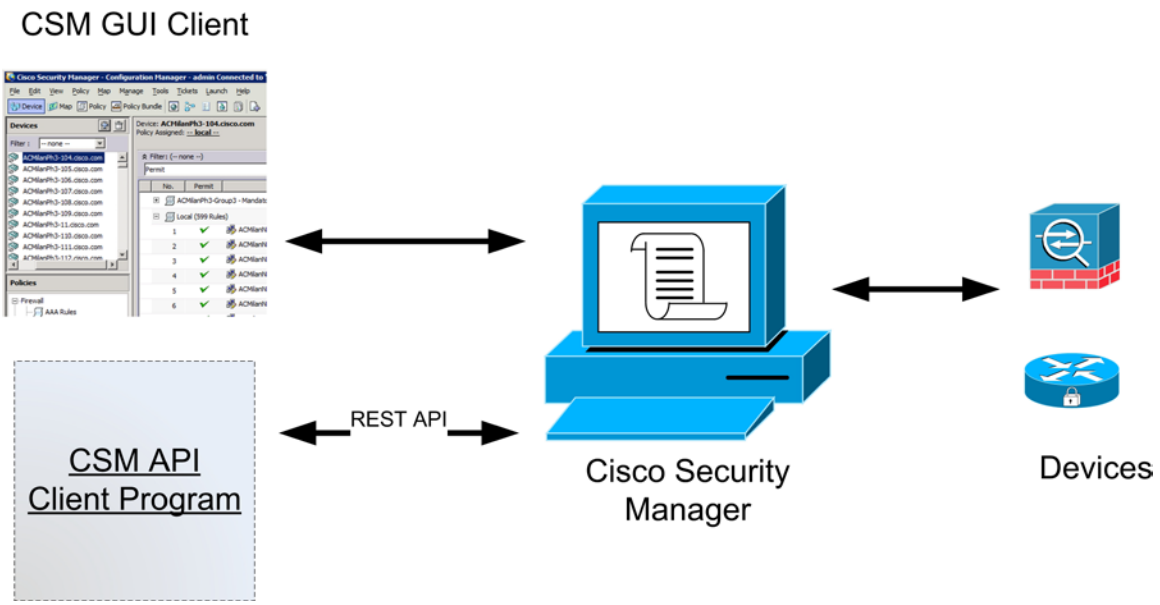
CSM NB API は、Cisco ASA および IPS デバイスのグローバルおよびデバイス特有のネットワーク設定ポリシーへのアクセスを提供するさまざまな機能を実現するサービスに分けられます。

クライアントがポリシー情報を受信すると、セキュリティ分析やセキュリティ イベントへのリンクなどを実行するために情報を使用できます。

1.1 スコープ

このマニュアルには、CSM NB API 1.1 仕様、CSM API ペイロードで実行されるメッセージ コンテンツの基盤を提供する XML スキーマの仕様、および CSM クライアント製品、CSM NB API を実装するインフラストラクチャ製品の動作仕様が記載されています。

注：特に明記されていない限り、このマニュアル全体を通して、CSM クライアントまたは CSM「API」クライアントへのすべての参照は、CSM サーバと通信する REST インターフェイスを使用するサードパーティの「CSM API クライアント プログラム」を意味します。このドキュメントで説明されている CSM クライアントと、サーバにアクセスするためにユーザのデスクトップにインストールされている、事前に組み込まれている CSM GUI クライアント アプリケーションを混同しないでください。混乱を避けるため、このドキュメント内で必要な場合には、事前に組み込まれている CSM GUI クライアント アプリケーションへの参照に、関連する画面のスクリーン ショットが含まれています。次の図を参照してください。CSM API クライアント プログラムとの対話を明確に表示しています。



1.2 以前のバージョンからの変更点

NB API 1.1 は NB API 1.0 の後継です。API のバージョン 1.1 に次の変更が追加されました。

1.2.1 ユニファイド アクセス ルール

これは、このバージョンでサポートされる新しいポリシーです。

1.2.2 セキュリティ ポリシー オブジェクト

これは、このバージョンでサポートされている新しいポリシー オブジェクトです。

1.2.3 ネットワーク オブジェクト

このオブジェクトは、API がユニファイド アクセス ルールで使用される場合、<ipv4Data> の代わりに <ipData> としてデータ要素を表すように変更されました。

1.2.4 設定ルールを最終変更したユーザ/チケットを戻す

API の使用例の 1 つは、準拠性チェックのために CSM から設定データを取得することです。コンフィギュレーション サービス API は、設定ルールを変更したユーザとチケットに関する情報も返すようになりました。

1.2.5 デバイス状態の追加：イベント サービスの一部としてのアップ/ダウン

Security Manager の一部ユーザには、毎日実行する作業スレッドおよび Security Manager で管理するネットワーク デバイスから変更された設定をフェッチする作業スレッドがあります。つまり、ユーザはデバイスから設定を取得する際に execDeviceReadOnlyCLICmds API 呼び出しを使用します。

execDeviceReadOnlyCLICmds API コールはデバイス ステータスがダウンすると停止します。したがって、この分野に関心があるユーザは execDeviceReadOnlyCLICmds API コールを実行する前にデバイス ステータスのアップ/ダウンを確認する必要があります。

1.2.6 EXEC コマンド API コールはカスタム タイムアウトをサポートします。

API の以前のバージョン（バージョン 1.0）では、execDeviceReadOnlyCliCmds にタイムアウト値がないため、デバイスが応答不能の場合は無限期間実行され、API サービスが停止する可能性があります。この問題を回避するため、API の現在のバージョン（バージョン 1.1）では、メソッド execDeviceReadOnlyCliCmds が要求のオプション属性を取得できます。このオプションの属性では API クライアントが execDeviceReadOnlyCliCmds メソッド呼び出しのタイムアウトを設定できます。

1.2.7 CSM で定義されたすべての共有ポリシーのリストを返す API 機能拡張

新しい API がバージョン 1.1 で追加され、特定のポリシー タイプのシステムのすべての共有ポリシーのリストを返します。

1.2.8 デバイス オブジェクトのデバイスの SysObjectID を戻す

API バージョン 1.0 には CSM に存在するすべてのデバイスの一覧を表示する多くの API があります。さらに、API バージョン 1.1 では、応答に、SysObjectID を含めます。

1.2.9 CSM 監査ログは API および CSM クライアントによるログインを区別する必要がある

CSM 4.3 では、NB API または CSM クライアントから CSM へのログイン時に、ユーザ ログインを示す監査ログ メッセージを作成します。しかし監査ログ メッセージには、NB API を使用したユーザ ログインの区別/説明はありませんでした。

CSM 4.4 リリースの一部として、監査ログは API および CSM クライアントによるログインを区別するよう拡張されました。

1.2.10　新しいファイアウォール ポリシー

次の新しいファイアウォール ポリシーは CSM 4.4 および API バージョン 1.1 以降で使用可能です。

- InterfaceNAT64ManualFirewallPolicy。InterfaceNAT64ManualFirewallPolicy は Unified（IPv6/IPv4）手動 NAT ルールを表します。
- InterfaceNAT64ObjectFirewallPolicy。InterfaceNAT64ObjectFirewallPolicy は Unified（IPv6/IPv4）オブジェクト NAT ルールを表します。

1.3 対象読者

このマニュアルは本質的に技術文書で、技術者、開発エンジニア、テクニカル マーケティング エンジニアを対象としています。

1.4 参照

以下は、このドキュメント内に表示され、ここで指定された範囲でこの仕様の標準部分を構成します。本書と次の参照された仕様で矛盾がある場合、本書の内容が優先されます。

- RFC-3986、Uniform Resource Identifier (URI)：Generic Syntax、Berners Lee、Fielding、Masinter、2005 年 1 月
- RFC-3579、RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)：Aboba、Calhoun、2003 年 9 月
- RFC-2387、MIME Multipart/Related Content-type：Levinson、1998 年 8 月

- XML-binary Optimized Packaging：Gudgin、Mendelsohn、Nottingham and Ruellan、http://www.w3.org/TR/xop10/、2005 年 1 月 25 日
- Describing Media Content of Binary Data in XML：Karmarkar、Yalçınalp、http://www.w3.org/TR/xml-media-types/、2005 年 5 月 4 日
- RFC-2616、Hypertext Transfer Protocol -- HTTP/1.1：Fielding、Gettys、Mogul、Frystyk、Masinter、Leach、Berners-Lee、1999 年 6 月
- XML Path Language：http://www.w3.org/TR/xpath/、1999 年 11 月

1.5 用語集

次に、このマニュアルで使用されている用語の略語と定義を示します。

省略形	定義
AAA	認証、許可、アカウンティング
API	アプリケーション プログラミング インターフェイス
AS	認証サーバ（例：AAA）
AUS	Auto Update Server
CA	認証局。raw デバイス設定データを保管する CSM の「Configuration Archive」モジュールも参照。
CNS	Cisco Networking Service
CR	証明書リポジトリ
CSM	Cisco Security Manager
DNS	ドメイン ネーム サービス
FQDN	完全修飾ドメイン名
HPM	CSM のヘルスとパフォーマンス モニタリング アプリケーション
HTTP	ハイパーテキスト転送プロトコル
IP	インターネット プロトコル
LAN	ローカル エリア ネットワーク
NAS	ネットワーク認証サーバ
NB	North Bound
OOB	アウトオブバンド（CSM の外部でデバイスで直接実行する未管理の変更）
PKI	公開キー インフラストラクチャ
PKC	公開キー暗号化
SDK	ソフトウェア開発キット
UI	ユーザ インターフェイス
URI	ユニバーサル リソース識別子
XML	拡張マークアップ言語
XPath	XML パスの言語
REST	REpresentational State Transfer

表 1：用語一覧

1.6 表記法

次のテキストの表記法がこのマニュアルで使用されます。

- すべてのタイプの定義は大文字で始まります。
  - 例：**xs:simpleType name="ObjectIdentifier”**
- すべての要素名は小文字で始まります。
  - 例：**<xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>**

1.7 CSM メッセージ フローの概要

CSM クライアントが CSM サーバを使用する場合の、主要なメッセージ フロー概要の一部を図 1 に示します。CSM サーバは、ネットワーク上のデバイスを認識するように設定され、ファイアウォール、IPS などに対する設定を読み取ります。CSM クライアントは開始後に、CSM サーバに認証され、API で提供するメソッドでアクセスすることができます。CSM のインターフェイスは共通（セクション 2）、設定（セクション 3）、イベント（セクション4）、ユーティリティ（セクション 5）などの複数のサービスとして定義されています。

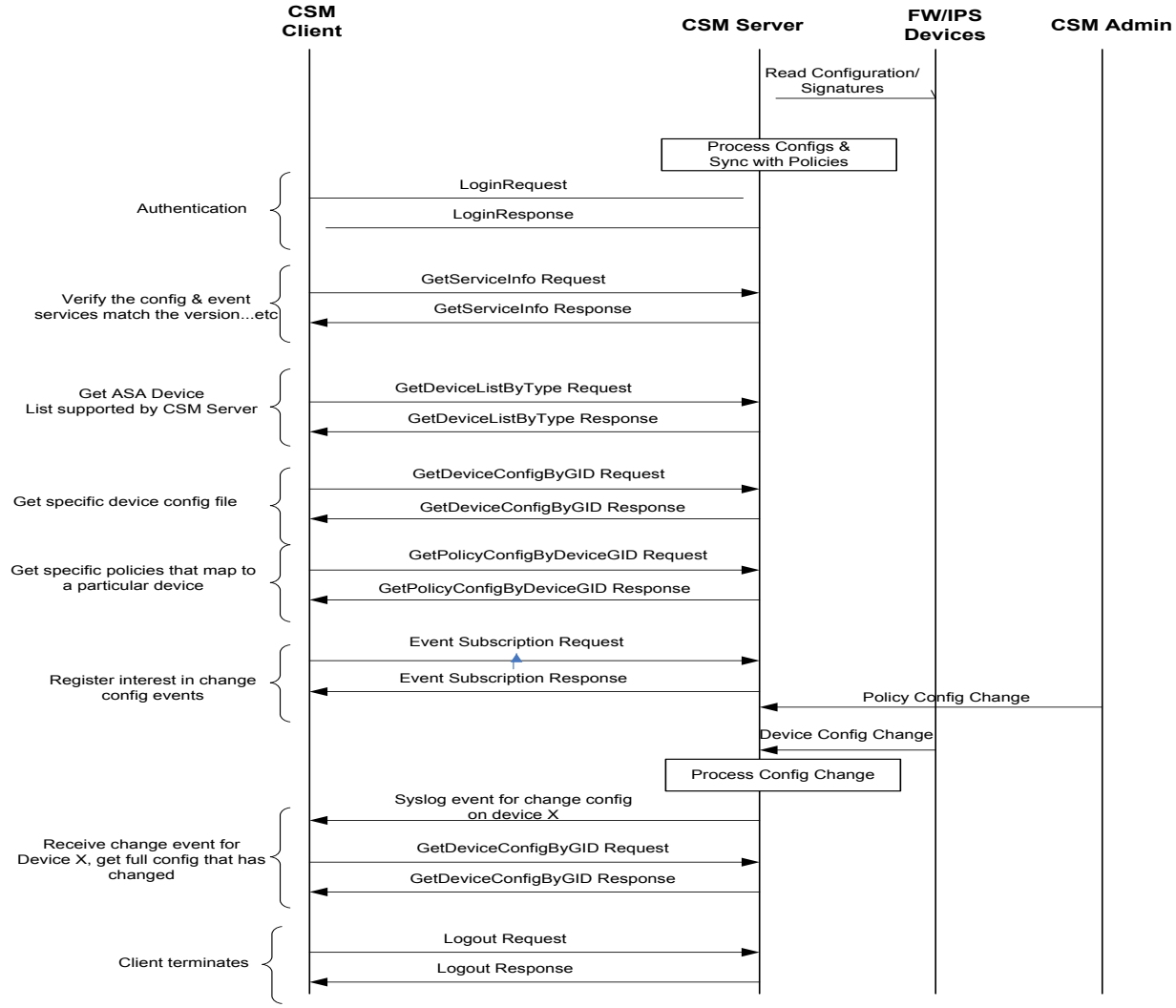


図 1：CSM のメッセージ交換の概要

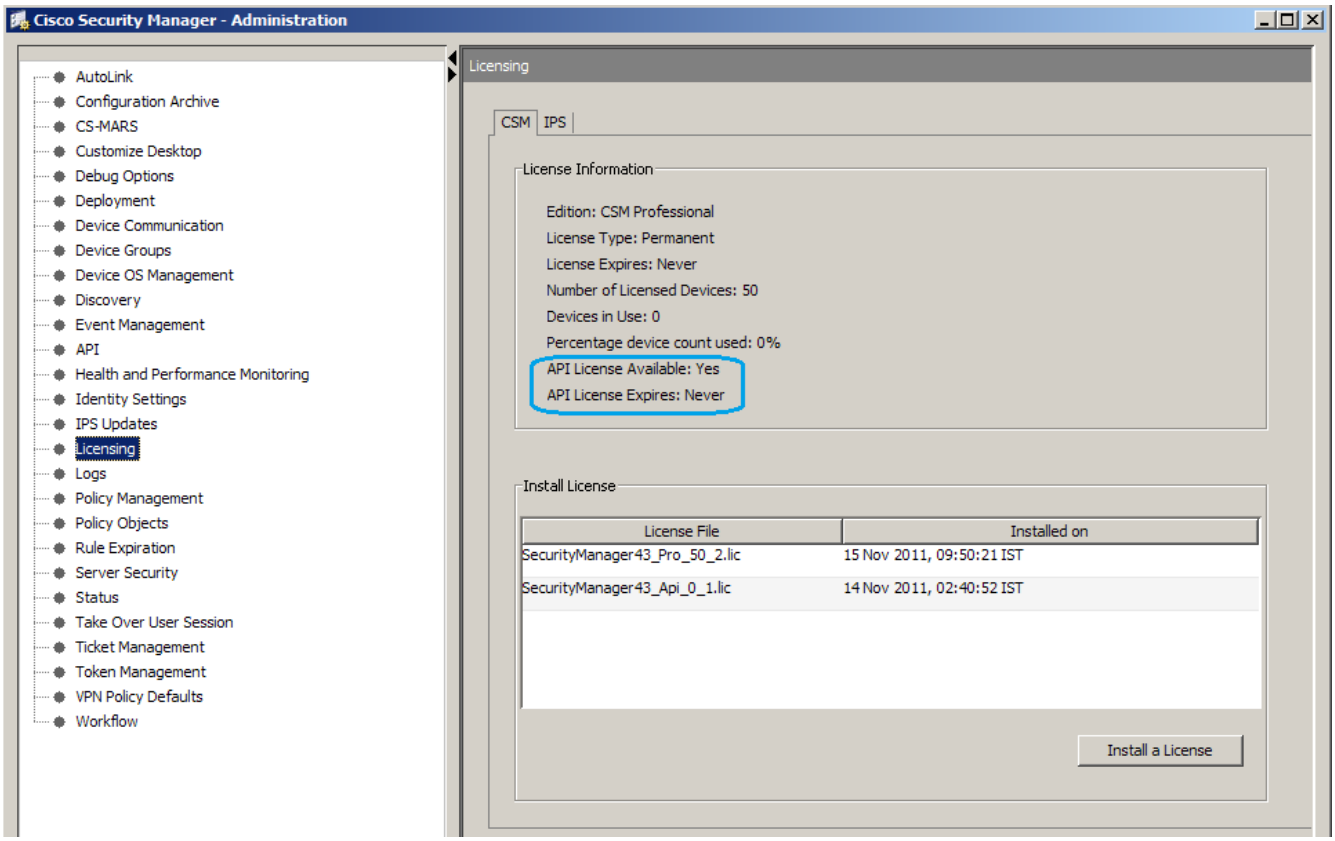
REST は CSM メッセージの伝送に使用されます。つまり、CSM のメッセージは CSM クライアントがアクセスした URL です。CSM メッセージ用の XML スキーマは、このマニュアルの各サービス セクション、またセクション 10 の完全な XSD ファイルで提供されます。

CSM API は、4 つの主なサブカテゴリにわたって構造化されます。

- **共有サービス API**：共通のメソッド定義およびスキーマの詳細を含む（ログイン、ログアウト、ping/hearbeat などに関連する）。詳細についてはタイトル「共有サービス API」のセクションを参照してください。
- **設定サービス API**：デバイス設定データを取得する（読み取り専用）メソッドとスキーマを含む。詳細についてはタイトル「CSM コンフィギュレーション サービス API」のセクションを参照してください。
- **イベント サービス API**：クライアントが特定の場合（配置、アウトオブバンド変更イベントなど）にイベント通知をサブスクライブできるメソッドとスキーマを含む。詳細についてはタイトル「CSM Events Service API」のセクションを参照してください。
- **Utility Service API**：ユーザが CSM API インターフェイスを使用して、デバイスの特定の CLI コマンドを実行できるスキーマとメソッドが含まれます。詳細についてはタイトル「CSM Utility Service API」のセクションを参照してください。

1.8 ライセンス

この機能はライセンスされています。特定の CSM API ライセンスは CSM [Tools] → [Security Manager Administration] → [Licensing] ページで適用する必要があります。API ライセンスは、CSM Professional エディションのライセンスが付与されている サーバにのみ適用できます。ライセンスが標準エディションで実行されている CSM、または CSM が評価モードの場合は、ライセンスは CSM に適用できません。



有効なライセンスのないすべての API 要求はエラー応答の項目を返します。

エラー コード：26

エラーの説明：API ライセンスはイネーブルではありません。有効な API ライセンスを追加し、この操作を再試行します。

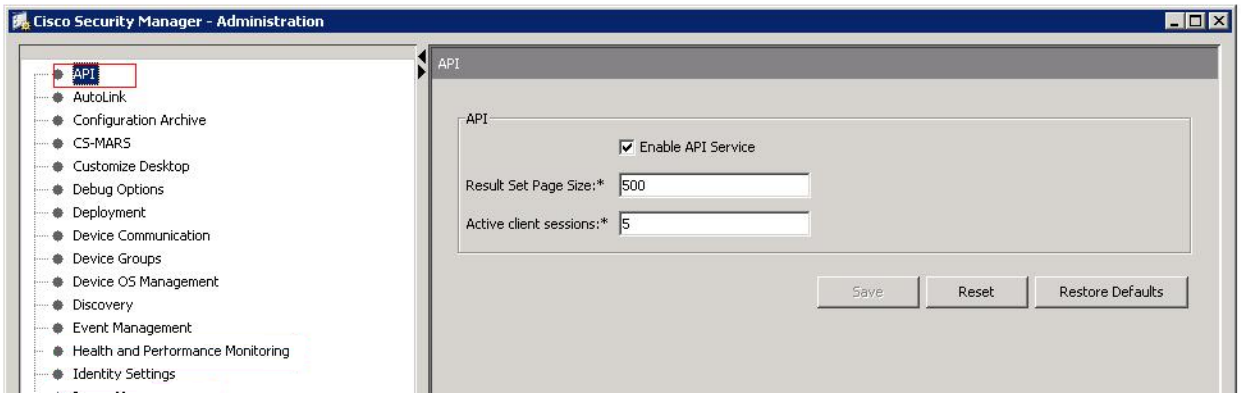
1.9 前提条件

以下は、API を使用および処理するための前提条件です。

- API サポートを含む、CSM サーバの最新バージョンをインストールする必要があります。詳細については、『CSM Install Guide』を参照してください。
- サーバをインストールしたら、対応するポリシー データを API で照会する前に、管理対象デバイス（ASA/IPS）を CSM に追加する必要があります。
- API は、ポリシー データベースに「コミットされた」データを返すだけです。そのため、該当するアクティビティを送信しないと、データは API を介して表示できません。
- Health and Performance Monitoring（HPM）モジュールは、アウトオブバンド（OOB）に対してイネーブルにする必要があり、デバイス ステータスのアップ/ダウン変更通知が動作するようにする必要があります。
- API には、特定のプログラミング言語の SDK を含みません。API には REST（Representational State Transfer）ベースのインターフェイスがあるため、XML メッセージ プロトコルに準拠している限り、API クライアントはどの言語でも実装できます。
- すべての API サービスは HTTPS を使用してのみアクセスできます。HTTP アクセスは許可されません。REST ベースの API アクセスの「root」URL は *https://<server-ip or host>/nbi/* です。

1.10 API 管理の設定

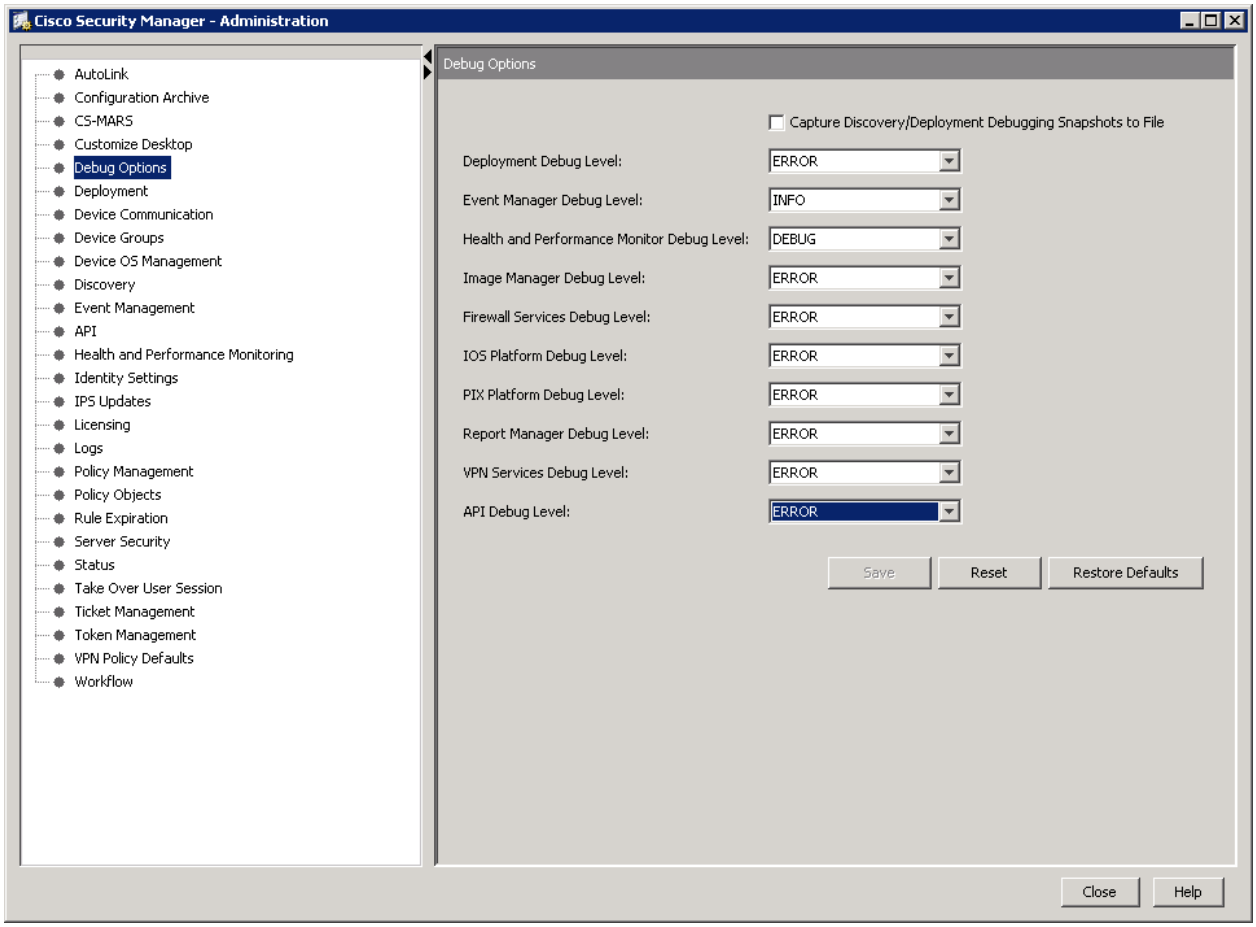
次のグローバル管理の設定は API 機能に固有です。（詳細情報については、バージョン固有 CSM のユーザ ガイド、またはインストール ガイドを参照してください）



- **Enable API Service**：API 機能を完全にイネーブルまたはディセーブルにすることができます。
- **Result Set Page Size**：単一の応答で返される結果セットのサイズを制御します。値は 100（最小）、500（デフォルト）、1000（最大）です。この設定は **GetPolicyConfigurationByName**（セクション 3.2.8）および **GetPolicyConfigurationByDeviceGID**（セクション 3.2.9）のメソッドにだけ適用されます。さらに応答がページを付ける方法の詳細については、2.2.1.1 を参照してください。
- **Active Client Sessions**：許可される同時アクティブ API クライアント セッション（ログイン）の合計数を管理します。値は 1（最小）、5（デフォルト）、10（最大）です。

1.11 デバッグの設定

CSM サーバ上の API 要求のデバッグをイネーブルにする場合は、API Debug Level フィールドの値を DEBUG に設定します。（このフィールドのデフォルト値は ERROR です）



2　共有サービス API

この項では、CSM API のすべてのサービスに共通のメソッドと共通のオブジェクト モデルについて説明します。

2.1 オブジェクト モデル

次のオブジェクト クラスは API 仕様で使用されます。

2.1.1 オブジェクト ID

オブジェクト ID は、オブジェクトのグローバルで固有な識別子です。RFC4122 に基づいてオブジェクト ID は 128 ビット値です。

```
<xs:simpleType name="ObjectIdentifier">
      <xs:restriction base="xs:string">
         <xs:pattern value="[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}"
         > </xs:pattern>
      </xs:restriction>
   </xs:simpleType>

   <xs:complexType name="ObjectIdentifierList">
            <xs:sequence>
                  <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="unbounded"/>
            </xs:sequence>
   </xs:complexType>
```

図 2：ObjectIdentifier および ObjectIdentifierList XML スキーマ

2.1.2 ベース オブジェクト

システム内のすべてのオブジェクトの共通クラスおよびすべてのオブジェクト クラスはこのクラスから継承します。次の属性があります。

属性	タイプ	コメント
gid	オブジェクト ID	オブジェクトに対する一義的なオブジェクト ID はオブジェクトのライフタイム中は変更されません。
lastUpdateTime	TimeStamp	作成または更新されたオブジェクトの更新時間を示します。
name	文字列	オブジェクトのオプションの表示名。
parentGID	オブジェクト ID	オブジェクトの親インスタンスを識別するオプションの親オブジェクト ID。
updatedByUser	文字列	オブジェクトを更新したユーザのユーザ名。
lastCommitTime	dateTime	オブジェクトの最終更新時間

ticketId	文字列	オブジェクトが更新された一環としてチケットのチケット ID。 これは、チケット生成がイネーブルになっていない場合は使用できません。
activityName	文字列	オブジェクトが更新された一環としてアクティビティのアクティビティ名。これは、CSM が WorkFlow モードの場合は適用されます。

表 2：BaseObject クラス属性

```
<xs:complexType name="BaseObject" >
      <xs:sequence>
            <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
            <xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="lastUpdateTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
            <xs:element name="parentGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
            <xs:element name="updatedByUser" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="lastCommitTime" type="xs:dateTime" minOccurs="0" maxOccurs="1"/>
            <xs:element name="ticketId" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="activityName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
</xs:complexType>
```

図 3：BaseObject XML スキーマ

2.1.3 Device

デバイス オブジェクトはシステム内の単一のデバイスを表すために使用されます。仮想コンテキストはシステムの個々のデバイスとしてモデル化されます。デバイス オブジェクトは親（admin/system context）および関連の子コンテキスト間の関係も表します。

Device クラスは、すべての属性を含む BaseObject から継承します。

要素と属性	タイプ	コメント
osType	文字列	列挙リスト {IOS、FWSM、ASA、PIX、IPS} からのデバイス OS タイプ。
osVersion	文字列	デバイスで実行されている OS のソフトウェア バージョン。バージョン文字列は、PIX プラットフォームでは 6.1、6.2 など、IOS プラットフォームでは 12.1、12.2S などが考えられます。
imagename	文字列	OS イメージ名。
mgmtInterface	Interface	デバイスの管理インターフェイスへの参照。CSM のデバイスの管理に使用されるインターフェイスは、管理インターフェイスとして使用されます。

InterfaceList	インターフェイスのシーケンス	デバイス内のインターフェイスのリスト。（*デバイスの管理インターフェイス以外のインターフェイスのリストを含みます*）。
fullConfig	文字列	デバイスの完全な設定を含む要素。デバイスの完全な設定は、show running config CLI コマンドの ASCII 出力で表されます。これは、*getDeviceConfigByGid* または *getDeviceConfigByName* API がデバイスの完全な設定を取得する場合にのみ表示されます。その他の場合は、この要素の値は設定されません。
virtualContextList	デバイスのリスト	デバイスに属する仮想コンテキスト オブジェクトのリスト。
configState	ConfigurationSTate	デバイスの設定状態およびコミットされていない変更の有無。{ committed, deployed } から取得した値。
sysObjectID	文字列	デバイスのシステム オブジェクト ID。

表 3：デバイス クラスの属性

```
<xs:complexType name="Device">
    <xs:complexContent>
        <xs:extension base="BaseObject">
            <xs:sequence>
                <xs:element name="osType" type="OSType" minOccurs="1" maxOccurs="1"/>
                <xs:element name="osVersion" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="imageName" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="sysObjectID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="fullConfig" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="mgmtInterface" type="Interface" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="interfaceList" type="InterfaceList" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="virtualContextList" type="Device" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="configState" type="ConfigurationState" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 4：デバイス XML スキーマ

2.1.3.1 Interface

Interface クラスはネットワーク デバイスのネットワーク インターフェイスを定義します。
InterfaceList クラスはインターフェイス インスタンスのシーケンスを定義します。

属性	タイプ	コメント
type	文字列	ネットワーク インタフェースのタイプ。以下はサポートされているインターフェイス タイプの値です（*Null*、*Management*、*Analysis-module*、*Async*、*ATM*、*BRI*、*BVI*、*Content-engine*、*Dialer*、*Dot11Radio*、*Ethernet*、*FastEthernet*、*GigabitEthernet*、*TenGigabitEthernet*、*HundredGigabitEthernet*、*FDDI*、*Group-Async*、*HSSI*、*IDS-Sensor*、*Loopback*、*Multilink*、*Port-channel*、*POS*、*PRI*、*Serial*、*Switch*、*Tokenring*、*Tunnel*、*VG-anylan*、*Virtual-Template*、*Virtual-TokenRing*、*VLAN*、*Redundant*）。
identifier	文字列	ネットワーク インターフェイスの識別子。名前がデバイスの任意のインターフェイスに設定されている場合、インターフェイス ID 値が名前として使用され、それ以外はインターフェイス タイプ/ポート/スロットが identifier として表示されます。たとえば GigabitEthernet インターフェイスは *outside* として設定されているため、API は識別子を *outside* として、*GigabitEthernet* をタイプとして示します。
ipInterface.domainName	文字列	オプションの DNS ドメイン名。
ipInterface.ipAddress	文字列	インターフェイスに設定されている IP アドレス（マスクを含む）。マスクは、IPS デバイスでは保存されません。
ipInterface.isNatAddress	文字列	管理 IP アドレスが NATed アドレスの場合は True。（このプレースホルダの要素は API バージョン 1.1 では使用できません）
ipInterface.realIpAddress	文字列	管理 IP アドレスが NATed アドレスである場合、realIpAddress は管理インターフェイスの NATed の非 IP アドレスです。（このプレースホルダの要素は API バージョン 1.1 では使用できません）
macInterface.macAddress	文字列	（任意）インターフェイスの MAC アドレス。ASA/PIX デバイスの MAC アドレスが単独で保存されます。他のデバイスのタイプについては、CSM は MAC アドレスを保持しません。また、CSM は MAC アドレスの詳細の書き込みを管理しません。CSM はユーザが明示的に変更する MAC アドレスのみを管理するので、応答にこのデータのみを保持します。

表 4：インターフェイス クラス属性

```
<xs:complexType name="InterfaceList">
        <xs:sequence>
            <xs:element name="interface" type="Interface" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="Interface">
        <xs:sequence>
            <xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="identifier" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="ipInterface" type="IPInterfaceAttrs" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="macInterface" type="MACInterfaceAttrs" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="MACInterfaceAttrs">
        <xs:sequence>
            <xs:element name="macAddress" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="IPInterfaceAttrs">
        <xs:sequence>
            <xs:element name="domainName" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="ipAddress" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="isNatAddress" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
            <xs:element name="realIpAddress" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
```

図 5：インターフェイス XML スキーマ

2.1.3.2 ファイアウォール機能

属性	タイプ	コメント
fwOsMode	Int	FWSM/ASA が Transparent（1）モードで実行中か、Routed（2）モードで実行中かを示します。
fwOsMultiplicity	Int	FWSM/ASA が単一の（1）コンテキストで実行中か、マルチ（2）コンテキストで実行中かを返します。
contextName	文字列	コンテキストの名前を返します。現在のデバイスがコンテキストを表す場合。
isComposite	ブール	デバイスが「複合」かそうでないかを示します。FWSM を含む Catalyst 6500 などのブレードは複合とマークされます。

表 5：FirewallCapabilities クラス属性

2.1.4 DeviceGroup

デバイス グループ オブジェクトがシステムでデバイスのコンテナを表すために使用されます。デバイス グループはゼロ以上のデバイスおよびゼロ以上の子デバイス グループを含みます。

DeviceGroup クラスは、すべての属性を含む BaseObject から継承します。

要素と属性	タイプ	コメント
要素：パス	文字列	グループを示す階層パス文字列
要素：デバイス	Device	このデバイス グループ内のゼロ以上のデバイス
要素：deviceGroup	DeviceGroup	このデバイス グループ内のゼロ以上のデバイス グループ

表 6：DeviceGroup クラス属性

```
<xs:complexType name="DeviceGroup">
        <xs:complexContent>
            <xs:extension base="BaseObject">
                <xs:sequence>
                    <xs:element name="path" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="device" type="Device" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="deviceGroup" type="DeviceGroup" minOccurs="0"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="DeviceGroupPath">
        <xs:sequence>
            <xs:element name="pathItem" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
```

図 6：DeviceGroup および DeviceGroupPath XML スキーマ

2.1.5 ポート ID

PortIdentifier クラスはネットワーク デバイスの物理または仮想ポート ID を定義します。

属性	タイプ	コメント
slotNum	unsignedInt	モジュラ シャーシの場合はスロットの識別子。非モジュラ シャーシでは、この属性は空です。
moduleNum	unsignedInt	スロット内のサブモジュールの場合はモジュールの ID。非モジュラ シャーシでは、この属性は空です。
portnum	unsignedInt	ポート番号

表 7：PortIdentifier クラス属性

```
<xs:complexType name="PortIdentifier">
        <xs:sequence>
            <!-- for non-modular chassis or chassis with a continuous port numbering scheme slot/module
are not included -->
            <xs:element name="slotNum" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
            <xs:element name="moduleNum" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
            <xs:element name="portNum" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
```

図 7：PortIdentifier XML スキーマ

2.1.6 BaseError

すべての要求エラーの共通クラスは BaseError クラスで定義されます。次の属性があります。

属性	タイプ	コメント
Code	Unsigned Long	要求で発生したエラーのタイプを識別する固有のエラーコード。
Description	文字列	発生したエラーの説明。

表 8：BaseError クラス属性

```
<xs:complexType name="BaseError">
    <xs:sequence>
        <xs:element name="code" type="xs:unsignedLong" maxOccurs="1"></xs:element>
        <xs:element name="description" type="xs:string" maxOccurs="1"></xs:element>
    </xs:sequence>
</xs:complexType>
```

図 8：BaseError XML スキーマ

次の一般エラー コードが現在定義されています。

コード	説明
0	Reserved
1	一般的なエラー
2	リソース不足
3	オブジェクト作成の失敗
4	認証に失敗しました：セッションが見つかりません
5	認証に失敗しました：無効または期限切れのセッション
6	内部通信の失敗
13	XML 要求のペイロードにデータは含まれません

15	XML 要求が無効です。（*このエラーは、XML 要求が公開 XML スキーマに準拠していないか、XML が不適格または無効です。これはシステム全体のエラーですが、このエラーは通常、アプリケーションが呼び出されている要求メソッドを解析できる場合、特定の応答オブジェクト内に設定されます*）
17	protVersion はオプションで、サポートされていないバージョンを指定すると、このエラーが返されます。サポートされているバージョンは、このリリースでは 1.0 です。
21	内部エラー
25	API サービスはディセーブルです
26	API ライセンスは適用されません

表 9：システム エラー コード

追加メソッドの特定のエラー コードは、個々のセクションで定義されています。すべての応答オブジェクトは基本エラー オブジェクトからの拡張です。エラーの内容は CSM API がメソッドを実行するときにエラーが発生した場合に設定されます。

返されたエラー メッセージは 2 種類の一般的なタイプです。

1. **一般/システム全体のエラー**：上記の表で定義されています。これらのエラーは通常、アプリケーションで発生した回復不能なエラーが原因です。これらのエラーは要求を実行する結果として発生することがあります。
2. **メソッド特有のエラー**：処理されるメソッドに特有のアプリケーション エラーです。

システム全体のエラーは唯一の応答として基本エラーの内容を返します。

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:baseError xmlns:ns1="csm">
      <code>1</code>
      <description>General Failure</description>
</ns1:baseError>
```

メソッド特有のエラーは応答オブジェクト内の基本エラーの内容を符号化します。以下は、ログイン要求を実行する際に発生するエラーの例です。

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:loginResponse xmlns:ns1="csm">
      <protVersion>1.0</protVersion>
      <error>
            <code>7</code>
            <description> Authentication Failure: Invalid username and/or password specified.</description>
      </error>
      <serviceVersion>1.0</serviceVersion>
      <sessionTimeoutInMins>15</sessionTimeoutInMins>
</ns1:loginResponse>
```

2.2 メソッド

2.2.1 共通の要求と応答

この API のすべてのメソッドは、メソッド要求と応答の引数として、XML オブジェクトをとります。要求の一部として渡される（また応答でエコーされる）XML オブジェクトは、次の属性を含む次クラスから生成されます。

属性	タイプ	コメント
protVersion	double	送信された特定の要求/応答関連プロトコルのバージョンを指定します。
reqId	文字列	サーバが関連づける応答でエコーする要求でクライアントによって送信された一義的なトークンを指定します。
startIndex	Unsigned Long	「ページを付けられたクライアント」要求によって指定されたオプションの開始インデックス。これは、複数行のデータを返す可能性があるファイアウォール ルールのようなポリシーに適用されます。クライアントは、データの次のページを取得するために、前の応答の終了インデックスと同じ startIndex を設定する必要があります。
endIndex	Unsigned Long	「一部のデータ」を返さなかった場合に「サーバ」で指定したオプションの終了インデックス。
totalCount	Unsigned Long	「サーバ」からのオプションの総数で、このポリシーでの合計行数を示します。
Error	BaseError	サーバに要求が送信される時に発生する可能性があるエラーを識別します。

表 10：BaseReqResp クラス属性

```
<xs:complexType name="BaseReqResp" >
    <xs:sequence>
        <xs:element name="protVersion" type="xs:double" minOccurs="0" maxOccurs="1"/>
        <xs:element name="reqId" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="startIndex" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
        <xs:element name="endIndex" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
        <xs:element name="totalCount" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
        <xs:element name="error" type="BaseError" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
</xs:complexType>
```

図 9：BaseReqResp XML スキーマ

2.2.1.1 ページ付け

サービス コールの一部（アクセス ルールのリストのフェッチ）は、大規模データ セットを返す可能性があります。このような大規模なフェッチがサーバおよびクライアントのパフォーマンスに関する問題を引き起こすことを防ぐには、これらの結果に「ページを付ける」ようにします。ページ付けは、*GetPolicyConfigByName* メソッドおよび *GetPolicyConfigByDeviceGID* メソッドだけに適用されます。ページ付けスキーマは、次のとおりに動作します。

1) クライアントが送信する最初の要求で、パラメータ startIndex、endIndex のいずれも設定されません。これは新しい要求であることをサーバに示しています。

2) サーバはこの要求にページを付加する必要があることを判別すると（結果セットの総数がページサイズより大きいため）、ページ付けされた結果を返し、次の 2 個の要素を設定します。

a. endIndex：現在の結果セットの endIndex に設定されます。たとえばこれが最初の要求で 1000 行が返される場合、endIndex は 1000 になります。

b. totalCount：クエリー自体の総数が含まれます。たとえば合計結果が 10,000 行になると、要素の totalCount は 10,000 に設定されます。

3) 後続の要求では、クライアントは BaseReqResp オブジェクトの startIndex を設定し、同じクエリー要求を再送信する必要があります。この場合は、クライアント要求の startIndex は受信した最後の応答の endIndex と同じです。

4) 最終応答でフェッチするデータがこれ以上ない場合は、サーバは endIndex および totalCount データを**設定しません**。

クライアントが 3600 のルールを持つデバイス「A」のファイアウォール ルールを照会する例を考えてみます。システムに設定されたページ サイズは 1000 です。以下は呼び出しのシーケンスです。

- クライアント → デバイス「A」のファイアウォール ルールを取得します。
- サーバ → 1000 ルールと endIndex=1000 および totalCount=3600 の応答を返します。
- クライアント → デバイス「A」のファイアウォール ルールと startIndex=1000 を取得します。
- サーバ → 1000 ルールと endIndex=2000 および totalCount=3600 の応答を返します。
- クライアント → デバイス「A」のファイアウォール ルールと startIndex=2000 を取得します。
- サーバ → 1000 ルールと endIndex=3000 および totalCount=3600 の応答を返します。
- クライアント → デバイス「A」のファイアウォール ルールと startIndex=3000 を取得します。
- サーバ → 600 ルールと endIndex=<not-set> および totalCount=<not-set> の応答を返します。

一般的なケースとして、クライアントが endIndex または totalCount が設定されていないことを認識すると、すべての行が返されているものと想定します。

2.2.2 ログイン メソッド

ログイン メソッドは CSM サーバによって提供されるサービスにアクセスしようとする CSM クライアントを認証します。このメソッドは、他のサービスで呼び出す他のメソッドよりも前に呼び出す必要があります。

2.2.2.1 要求

ログイン要求メソッドの例を図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/login

XML Argument:

<?xml version 1.0 encoding="UTF-8"?>
<loginRequest>
     <protVersion>1.0</protVersion>
     <reqId>123</reqId>
     <username>allan</username>
     <password>myPassword123</password>
     <heartbeatRequested>true</heartbeatRequested>
</loginRequest>
```

図 10：ログイン要求の例

表 11：ログイン要求の要素と属性の説明

XML 引数	定義
loginRequest	ログイン要求はサーバに対してクライアントを認証し、以降の要求で使用される Cookie を返します。
username	セッションに関連付けられた CSM クライアントのユーザ名
password	セッションに関連付けられた CSM クライアントのパスワード
heartbeatRequested	この属性は任意に定義することができます。属性を true に設定すると、CSM クライアントは CSM サーバからハート ビートのコールバックを受信します。サーバは、*（非アクティブ タイムアウト）/ 2* 分に近い頻度で、クライアントの ping を再試行します。クライアントがハート ビートに応答しないと、API は次のインターバルの間にハートビートを再試行します。ハート ビートが成功するとセッションの非アクティブ タイムアウトがリセットされます。
callbackUrl	CSM サーバがコールバックを発信する URL。これは heartbeatRequested が true の場合に指定する必要があります。HTTPS ベースのコールバック URL だけが許可されます
HTTP メソッド	POST
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="loginRequest" type="LoginRequest"/>
<xs:complexType name="LoginRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp">
                  <xs:sequence>
                        <xs:element name="username" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="password" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="heartbeatRequested" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="callbackUrl" type="xs:string" minOccurs="0" maxOccurs="1"/>
                  </xs:sequence>
            </xs:extension>
      </xs:complexContent>
```

図 11：LoginRequest XML スキーマ

2.2.2.2 応答

Login API では、ユーザ クレデンシャルを認可し、安全な Cookie としてセッション トークンを返します。セッション値は「asCookie」キーに格納されます。このセッションには、15 分のデフォルトのセッション非アクティブ タイムアウトがあります。HTTP ヘッダーおよび XML コンテンツの応答の例を、以下の図に示します。これらのメッセージのフィールドを次の表に示します。

```
HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Content:

<?xml version=”1.0” encoding=”UTF-8”?>
<loginResponse>
      <protVersion>1.0</protVersion>
      <serviceVersion>1.0.1</serviceVersion>
      <sessionTimeoutInMins>15<sessionTimeoutInMins>
</loginResponse>
```

serviceVersion は、次のサービスの場合は 1.1 となります。

- 構成サービス
- イベント サービス
- ユーティリティ サービス

図 12：ログイン応答の例

表 12：ログイン応答の要素と属性の説明

HTTP/XML 応答	定義
HTTP ヘッダー：**asCookie**	後続のメソッド呼び出しで渡す必要のある認証文字列、有効期限、パス、ドメインを定義する Cookie。 例）Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/; domain=.hostdomain.com
XML 要素：loginResponse	セッション情報または失敗を返す XML コンテンツ
serviceVersion	実行中のコンフィギュレーション サービスのサービス バージョン。この属性は、ユーザが正常に認証された場合にのみ含まれます。
sessionTimeoutInMins	セッション タイムアウト（分）は、CSM サーバがセッションをアクティブではなくなったとして破棄する前に、クライアントによってアクティビティなしを渡す必要がある分数です。セッションのタイムアウト後の CSM クライアントによるアクセスはすべて拒否され、CSM クライアントは CSM サーバに対するアクセスを再認証する必要があります。この属性は、ユーザが正常に認証された場合にのみ含まれます。デフォルトは 15 分です。

```
<xs:element name="loginResponse" type="LoginResponse"/>
<xs:complexType name="LoginResponse">
      <xs:complexContent>
            <xs:extension base="BaseReqResp">
                  <xs:sequence>
                        <xs:element name="serviceVersion" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="sessionTimeoutInMins" type="xs:positiveInteger" minOccurs="1"
maxOccurs="1"/>
                  </xs:sequence>
            </xs:extension>
      </xs:complexContent>
</xs:complexType>
```

図 13：LoginResponse XML スキーマ メソッドの heartbeatCallback

メソッド heartbeatCallback の例を次に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://csm-clienthost/heartbeatCallback

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<heartbeatCallbackRequest>
   <protVersion>1.0</protVersion>
</heartbeatCallbackRequest>
```

図 14：メソッドの heartbeatCallback の例

表 13：メソッド heartbeatCallback の要素および属性の説明

HTTP Header/XML 引数	定義
heartbeatCallbackRequest	CSM クライアントで heatbeatCallback メソッドが CSM サーバによって呼び出され、CSM クライアントが引き続きアクティブであることが確認されます。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK
	401 Unauthorized

メソッド特有のエラー：

コード	説明
7	認証に失敗しました：指定されたユーザ名/パスワードが無効です。
8	認証に失敗しました：ログイン セッションのエラー。要求に含まれる asCookie が不正です。このエラーは、セッションが失効している（無効/タイムアウト）場合に発生する可能性があります。セッションは ping またはハートビートを使用してアライブ保持できます。
9	到達した最大アクティブ セッション数。これ以上のセッションは許可されません。（API 管理オプションのセクション 1.10 を参照してください）
10	サーバ リソースに接続中にエラーが発生しました。（またはログアウト メソッドに適用）
11 および 12	ユーザ入力の認証エラー。次の条件のいずれかで返されます。 • エラーは、ユーザ入力を解析中に発生しました。 • ハートビート タグの欠落 • 指定されたコールバック URL が無効、またはコールバック URL が

	欠落しています。 • 要求されたハートビートが false であると場合に指定されるコールバック URL
14	セッションの作成に失敗しました
16	コールバック URL に許可された HTTPS プロトコルのみ

表 14：ログイン メソッドのエラー コード

2.2.3 ログアウト メソッド

ログアウト メソッドは、以前認証された CSM クライアントが CSM サーバへのセッション アクセスを要求していないことを CSM サーバに通知します。CSM クライアントは、セッション アクセス タイムアウト期間内に CSM サーバでのアクセス メソッドを意図しない場合は、CSM サーバからログアウトする必要があります。

2.2.3.1 要求

ログアウト要求メソッドの例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/logout

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<logoutRequest>
      <protVersion>1.0</protVersion>
      <reqId>123</reqId>
</logoutRequest>
```

図 15：ログアウト要求の例

表 15：ログアウト要求の要素と属性の説明

HTTP/XML 引数	定義
logoutRequest	CSM サーバから CSM クライアントをログアウトする XML 引数
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。

HTTP/XML 引数	定義
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="logoutRequest" type="LogoutRequest"/>
<xs:complexType name="LogoutRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp"/>
      </xs:complexContent>
</xs:complexType>
```

図 16：LogoutRequest XML スキーマ

2.2.3.2 応答

ログアウト応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
      <logoutResponse>
         <protVersion>1.0</protVersion>
         <reqId>123</reqId>
      </logoutResponse>
```

図 17：ログアウト応答の例

表 16：ログアウト応答の要素と属性の説明

XML 応答	定義
logoutResponse	セッション情報または失敗を返します

```
<xs:element name="logoutResponse" type="LogoutResponse"/>
<xs:complexType name="logoutResponse">
      <xs:complexContent>
            <xs:extension base="BaseReqResp"/>
      </xs:complexContent>
</xs:complexType>
```

図 18：LogoutResponse XML スキーマ

2.2.4 ping メソッド

ping メソッドは、タイムアウトまたはサーバによる廃棄からアクティブな認証セッションを維持します。CSM クライアントは、sessionTimeoutInMins の前に ping メソッドを呼び出して、CSM サーバが認証されたセッションを廃棄しないようにする必要があります。*注：サーバにコールを作成するため*

に認証セッションが使用されている場合、認証されたセッションの非アクティブ タイムアウトは暗黙的にリセットされます。

2.2.4.1 要求

ping 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/ping

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
   <pingRequest>
          <protVersion>1.0</protVersion>
          <reqId>123</reqId>
   </pingRequest>
```

図 19：ping 要求の例

表 17：ping 要求の要素と属性の説明

HTTP/XML 引数	定義
pingRequest	CSM サーバを ping する XML 引数
HTTP メソッド	PUT
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="pingRequest" type="PingRequest"/>
<xs:complexType name="PingRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp"/>
      </xs:complexContent>
</xs:complexType>
```

図 20：PingRequest XML スキーマ

2.2.4.2 応答

ping 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version=”1.0” encoding=”UTF-8”?>
   <pingResponse>
          <protVersion>1.0</protVersion>
          <reqId>123</reqId>
   </pingResponse>
```

図 21：ping 応答の例

表 18：ping 応答の要素と属性の説明

XML 応答	定義
pingResponse	ping の情報を返します。

```
<xs:element name="pingResponse" type="PingResponse"/>
<xs:complexType name="PingResponse">
      <xs:complexContent>
            <xs:extension base="BaseReqResp"/>
      </xs:complexContent>
</xs:complexType>
```

図 22：PingResponse XML スキーマ

3　CSM コンフィギュレーション サービス API

コンフィギュレーション サービスは、ネットワークと CSM ポリシー オブジェクトの設定を読み込むためのアクセスを提供します。ポリシーの状態は、コミット済みまたは展開済みの場合があります。コミットされたポリシーは、デバイスにまだ展開されていないポリシーである可能性があります。コンフィギュレーション サービス API ポリシー メソッドは**コミットされたポリシーだけ**を返します。デバイスに対するポリシーのコミットおよび展開には、2 種類の操作があります。すべてのポリシー設定の変更が正常に展開されるまで、現在実行中のデバイス設定に変更が反映されていない可能性があります。設定の状態を把握する方法の詳細については、BasePolicy および BasePolicyObject クラス定義を参照してください。

BasePolicy および BasePolicyObject 要素（セクション 3.1.1 および 3.1.2）の **configState** 属性は、この状態を示します。設定状態が「コミット済み」の場合、この設定がフェッチされたデバイスには、**未展開**で未処理のコミット済み変更があります。設定状態が「展開済み」の場合は、**コミットされたすべての変更が展開済みであること、つまり CSM ポリシーとデバイスが同期していること**を意味します。

選択的なポリシー管理は、CSM の管理者が選択的に CSM ポリシーを管理する CSM 機能です。*管理に選ばれなかった*ポリシーに対応するデータは、CSM ポリシー データベースに保持されません。またこのようなポリシー データはこの API では返されません。このような事例において、Utility Service API を使用することを考慮してください。

3.1 オブジェクト モデル

ここでは、CSM コンフィギュレーション サービスで使用されるオブジェクト モデルについて説明します。

3.1.1 基本ポリシー

CSM オブジェクト モデルは、2 つのプライマリ クラスから構成されます。1 つは特定のポリシーを示す「ポリシー」クラスで、AAA ポリシー、インターフェイス ポリシー、ファイアウォール アクセス ルールなどがあります。ポリシー クラスはさらに、ネットワーク アドレス、サービス、ポート リストなどの再利用可能なオブジェクトを示す際に使用する「ポリシー オブジェクト」（ビルディング ブロックとも呼ばれます）を参照することができます。例として、「ファイアウォール ポリシー」は「ネットワーク ポリシー オブジェクト」としてソースおよび宛先アドレスを定義することができます。定義済みポリシー オブジェクトは再利用可能で、複数のポリシーで使用できます。

すべてのポリシー クラスは「BasePolicy」クラスから拡張され、すべての「ポリシー オブジェクト」は「BasePolicyObject」と呼ばれる基本クラスから拡張されます。次のクラス図はこの関係を示しています。

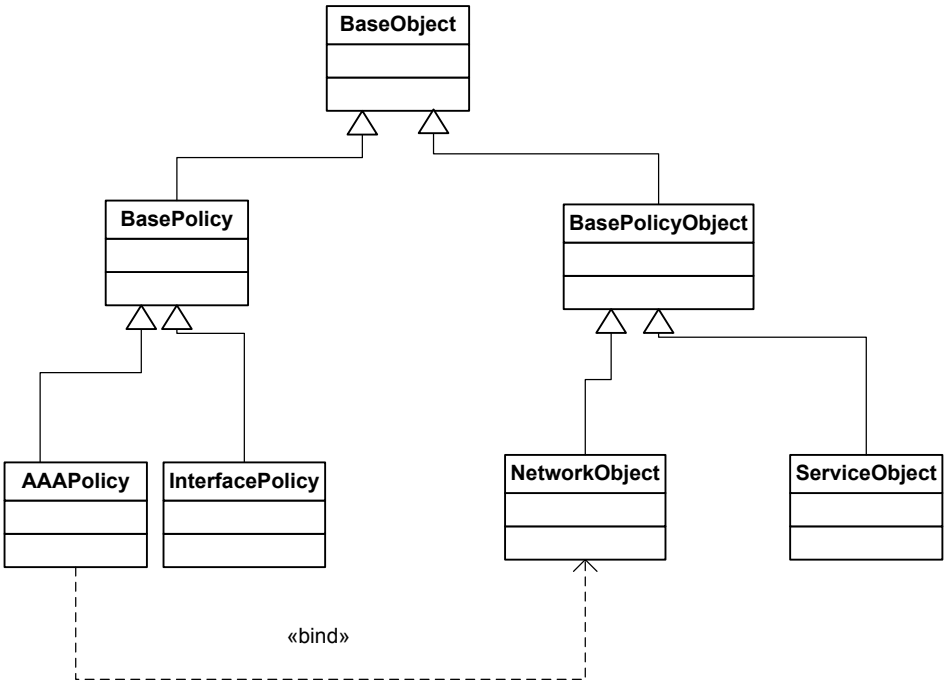


図 23：BasePolicy クラス継承

BasePolicy クラスは、すべての属性を含む BaseObject クラスから継承します。

属性	タイプ	コメント
type	文字列	policyExample のコンテンツ「AAAPolicy」を記述するポリシーの必須属性です。
isMandatoryAggregation	boolean	onion 集約用にだけ関連します。ポリシーが必須かどうかを返します。必須ポリシーは親の先頭に付加され、デフォルト ポリシーが追加されます。
orderId	Int	O ベースの注文 ID ポリシー レコードにあるポリシーのインデックスを返します。
description	文字列	ポリシーの説明（任意）
configState	Enumeration	{ committed, deployed } から取得したポリシーの現状。

表 19：BasePolicy クラス属性

```
<xs:complexType name="BasePolicy">
    <xs:complexContent>
        <xs:extension base="BaseObject">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="orderId" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="isMandatoryAggregation" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="description" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="configState" type="ConfigurationState" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 24：BasePolicy XML スキーマ

ここでは、重要なポリシーおよびポリシー オブジェクト タイプのデータ内容を表示します。

3.1.2 BasePolicyObject

これは再利用可能なオブジェクト定義であるすべてのポリシー オブジェクトの基本クラスです。AAA ポリシー、ファイアウォール ポリシーなどのポリシー データは、ポリシー オブジェクトへの参照を維持します。ポリシー オブジェクトの複数「タイプ」があります。また、ポリシーのオブジェクトは「グローバル」または「ローカル」を指定できます。グローバル ポリシー オブジェクトはオブジェクトが任意のデバイスのポリシーによって参照されているグローバル定義であることを示します。ポリシー オブジェクト「オーバーライド」は、特定のデバイス用に「オーバーライドされた」グローバル ポリシー オブジェクトを示します。

一連のポリシー オブジェクトは、同一「タイプ」の 1 つのポリシー オブジェクトに「グループ化」される場合があります。場合によっては、ポリシー オブジェクトはまったく異なるタイプのポリシー オブジェクトを参照する場合もあります（これは同じタイプのポリシー オブジェクトをグループ化する「グループ化された」ポリシー オブジェクトとは異なります）。

BasePolicyObject クラスは、すべての属性を含む BaseObject から継承します。

属性	タイプ	コメント
type	文字列	ポリシー オブジェクトのタイプを示すポリシー オブジェクトの必須属性です。例：「ネットワーク」または「サービス」。
comment	文字列	このポリシー オブジェクト（オプション）関連のコメントまたは説明。
nodeGID	オブジェクト ID	これが「オーバーライド」ポリシー オブジェクトの場合はデバイス ID。グローバルの場合は -1 に設定します。
isProperty	ブール	値 True はグローバル ポリシー オブジェクトの「オーバーライド」が許可されていることを示します。False の場合、このグローバルの「オーバーライド」が許可されていないことを示します。
subType	文字列	「ネットワーク」や「サービス」ポリシー オブジェクに適切なサブタイプ。たとえば、「ホスト」、「アドレス範囲」は、「ネットワーク」ポリシー オブジェクトに対するサブタイプです。
isGroup	ブール	True の場合、このポリシー オブジェクトが同じタイプの別のポリシー オブジェクトの「グループ」かどうかを示します。
refGIDs	ObjectIdentifierList	「isGroup」が true の場合にだけ適用されます。リストには、参照される ポリシー オブジェクト ID があります。
configState	Enumeration	{ committed, deployed } から取得したポリシー オブジェクトの現状。

表 20：BasePolicyObject クラス定義

名前のオーバーライドの動作（つまり、オブジェクトに関連付けられた名前です。名前が空（“”）である場合は、内部ポリシー オブジェクトを示します。すべてのユーザ定義のポリシー オブジェクトは、名前がなければなりません。内部ポリシー オブジェクトは場合によってはシステムによって自動的に作成されます。たとえば、ユーザがルールのリテラル IP アドレス（ポリシー オブジェクトの

代わりに）を提供する場合、「名前のない」ポリシー オブジェクトは、ルールに自動的に作成されます。

parentId = このポリシー オブジェクトが「オーバーライド」である親グローバル ポリシー オブジェクト ID。オーバーライド以外の場合、この値は -1 に設定されます。

```
<xs:complexType name="BasePolicyObject" >
    <xs:complexContent>
        <xs:extension base="BaseObject">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="comment" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="nodeGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="isProperty" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="subType" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="isGroup" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                <xs:element name="refGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="configState" type="ConfigurationState" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 25：BasePolicyObject XML スキーマ

3.1.3 ポリシーのユーティリティ クラス

次のユーティリティ クラスは、複数のポリシーで使用するために定義されています。

-NetworkInterfaceObjectsRefs

- ネットワーク、interfaceRoles および ipv4 データ ストリングのリストを参照します。

-NetworkObjectsRefs

- ネットワークおよび ipv4 データ ストリングのリストを参照します。

-IdentityUserGrpObjectsRefs

- ユーザ グループ オブジェクト GID、ユーザ名、およびユーザ グループのリストを参照します。

-SecurityGrpObjectsRef

- セキュリティ名、セキュリティ タグまたはセキュリティ オブジェクト GID を参照します。

-SecurityGrpObjectsRefs

- SecurityGrpObjectsRef のリストを参照します。

-NetworkObjectRefs

- ネットワーク、ipv4 データ文字列をおよびインターフェイス キーワードを参照します。

-NetworkOrIPRef

- ホスト、ネットワークまたは ipv4 データ ストリングを参照します。

```
    <xs:complexType name="NetworkInterfaceObjectsRefs">
        <xs:sequence>
            <xs:element name="networkObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="interfaceRoleObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="NetworkObjectsRefs">
        <xs:sequence>
            <xs:element name="networkObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SecurityGrpObjectsRef">
      <xs:sequence>
        <xs:choice>
          <xs:element name="securityGrpObjectGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
          <xs:element name="secName" type="xs:string" minOccurs="0" maxOccurs="1"/>
          <xs:element name="secTag" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SecurityGrpObjectsRefs">
      <xs:sequence>
        <xs:element name="securityTag" type="SecurityGrpObjectsRef" minOccurs="1"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="IdentityUserGrpObjectsRefs">
        <xs:sequence>
            <xs:element name="identityUserGrpObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="userNameData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="userGroupData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="NetworkObjectRefs">
        <xs:sequence>
            <xs:element name="networkObjectGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="interfaceKeyword" type="xs:string" fixed="interface" minOccurs="0"
maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="NetworkOrIPRef">
        <xs:choice>
            <xs:element name="hostOrNetworkObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
            <xs:element name="ipv4Data" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:choice>
    </xs:complexType>
```

図 26：ポリシーのユーティリティ クラス XML スキーマ

3.1.4 PolicyObject 派生クラス

このセクションおよびサブセクションは、この API でサポートされる ポリシー オブジェクト クラスを定義します。

3.1.4.1 NetworkPolicyObject

NetworkPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。NetworkPolicyObject は IPv4 アドレス、ネットワークまたは範囲を定義します。

ポリシー定義は gid 値で NetworkPolicyObject を参照します。継承された「**サブタイプ**」属性に含める IPv4 データのタイプを定義します。NetworkPolicyObject サブタイプに対する有効な値は「ホスト」、「ネットワーク」、「アドレス範囲」、「FQDN」および「グループ」です。NetworkPolicyObject の内容は、継承された **isGroup** 属性が true に設定される場合（およびサブタイプは「グループ」）、「空」になることがあります。このような場合、NetworkPolicyObject 自体は、「その他のネットワーク ポリシー オブジェクト」への参照を含みます。

このような PolicyObject の GID 値のリストは、**refs** に継承される属性から取得されます。「グループ」NetworkPolicyObject も、リテラル IPv4 アドレス、ネットワークまたは範囲を表示する複数の IPv4Data の要素を含めることができます。refs 属性参照および IPv4Data の要素からのデータの組み合わせは、ポリシー オブジェクトが参照するアドレスの完全なグループを示します。

要素	タイプ	コメント
ipv4Data	文字列	アドレス、範囲、またはネットワークなどの特定の IPv4 データを定義します。
ipData	文字列	アドレス、範囲、またはネットワークなどの特定の IP データを定義します。このオプションは、IPv4 と IPv6 の両方を指定できます。
fqdnData	Complex	これが Fully Qualified Domain Name（FQDN）タイプの NetworkPolicyObjects の場合、FQDN が含まれます。
fqdnData.value	文字列	FQDN 文字列
fqdnData.isIPv4Only	boolean	true の場合、生成され、デバイスに送信されるコマンドは「v4」パラメータが含まれます。

表 21：NetworkPolicyObject クラス定義

注：API バージョン 1.1 以降、<ipData> と呼ばれる新しいタグがネットワーク オブジェクト定義に追加されました。DeviceAccessRuleFirewallPolicy などのレガシー ポリシーで参照されるネットワーク オブジェクトは、これらのポリシーが IPv4 アドレスだけを参照するときに <ipv4Data> を引き続き使用します。ただし、DeviceAccessRuleUnifiedFirewallPolicy などの新しいポリシーはポリシーで <ipData> タグを使用します。これは <ipData> タグが IPv4 アドレスと IPv6 アドレスの両方を含む場合があるためです。

```
<xs:complexType name="NetworkPolicyObject">
    <xs:complexContent>
      <xs:extension base="BasePolicyObject">
        <xs:sequence>
          <xs:choice>
            <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="ipData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
          </xs:choice>
          <xs:element name="fqdnData" minOccurs="0" maxOccurs="1">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="value" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="isIPv4Only" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

図 27：NetworkPolicyObject XML スキーマ

3.1.4.2 IdentityUserGroupPolicyObject

IdentityUserGroupPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。IdentityUserGroupPolicyObject はユーザとユーザ グループを定義します。ポリシー定義は gid 値で IdentityUserGroupPolicyObject を参照します。

要素	タイプ	コメント
userNameData	文字列	グループ オブジェクトのユーザのリスト。
userGroupData	文字列	このグループ オブジェクトのユーザ グループ部分のリスト。

表 22：IdentityUserGroupPolicyObject クラス定義

```
<xs:complexType name=”IdentityUserGroupPolicyObject">
    <xs:complexContent>
        <xs:extension base="BasePolicyObject">
            <xs:sequence>
                <xs:element name="userNameData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="userGroupData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 28：IdentityUserGroupPolicyObject XML スキーマ

3.1.4.3 PortListPolicyObject

PortListPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。PortListPolicyObject は個々のポートまたはポートのグループを定義します。ポリシー定義は gid 値で PortListPolicyObject を参照します。PortListPolicyObject の内容は、継承された **isGroup** 属性が true に設定される場合、「空」になることがあります。このような場合、PortListPolicyObject 自体は、「その他のポート リスト ポリシー オブジェクト」への参照を含みます。このような PolicyObject の GID 値のリストは、**refs** に継承される属性から取得されます。

場合によっては PortListPolicyObject には **refs** 属性の GID 値のグループがあり、さらに単一のポート リスト オブジェクトの定義を含めることができます。このような場合、参照と PortListPolicyObject 要素内の値を組み合わせて、定義で指定されたポートのグループを構成します。

次の表に、ServicePolicyObject の内容を定義します。

要素	タイプ	コメント
startPort	PortIdentifier	開始ポートを定義します。
endPort	PortIdentifier	終了ポートを定義します。開始ポートから終了ポートはこの PolicyObject で指定されたポートの範囲を指定します。開始ポートと終了ポートが等しい場合は、定義が「単一」ポートになります。

表 23：PortListPolicyObject クラス定義

```
<xs:complexType name="PortListPolicyObject">
    <xs:complexContent>
        <xs:extension base="BasePolicyObject">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element name="startPort" type="PortIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="endPort" type="PortIdentifier" minOccurs="1"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 29：PortListPolicyObject XML スキーマ

3.1.4.4 ServicePolicyObject

ServicePolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。ServicePolicyObject は tcp、udp などのサービスを定義します。ポリシー定義は gid 値で ServicePolicyObject を参照します。ServicePolicyObject の内容は、継承された **isGroup** 属性が true に設定される場合、「空」になることがあります。このような場合 ServicePolicyObject 自体は、「その他のサービス ポリシー オブジェクト」への参照を含みます。このような PolicyObject の GID 値のリストは、**refs** に継承される属性から取得されます。

場合によっては ServicePolicyObject には **refs** 属性の GID 値のグループがあり、さらに単一のサービスの定義を含めることができます。このような場合、参照と ServicePolicyObject 要素内の値を組み合わせて、定義で指定されたポートのグループを構成します。

次の表に、ServicePolicyObject の内容を定義します。

要素	タイプ	コメント
protocol	文字列	tcp、udp などのプロトコルを定義する文字列
sourcePort	Complex	送信元ポートのデータを保持する要素のコンテナ。これは、選択タイプの複合要素であるため、サブ要素のうち 1 つのみ定義されます。
sourcePort.port	Unsigned Int	実際のポート値。
sourcePort.portRef	ObjectIdentifier	PortListPolicyObject への Gid の参照。
DestinationPort	Complex	宛先ポートのデータを保持する要素のコンテナ。これは、選択タイプの複合要素であるため、サブ要素のうち 1 つのみ定義されます。
destinationPort.port	Unsigned Int	実際のポート値。
destinationPort.portRef	ObjectIdentifier	PortListPolicyObject への Gid の参照。
icmpMessage	文字列	ICMP メッセージの内容。

表 24：ServicePolicyObject クラス定義

```
    <xs:complexType name="ServiceParameters">
        <xs:sequence minOccurs="1" maxOccurs="1">
            <xs:element name="protocol" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="sourcePort" minOccurs="0" maxOccurs="1">
                <xs:complexType>
                    <xs:choice>
                        <xs:element name="port" type="xs:unsignedInt"/>
                        <xs:element name="portRefGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    </xs:choice>
                </xs:complexType>
            </xs:element>
            <xs:element name="destinationPort" minOccurs="0" maxOccurs="1">
                <xs:complexType>
                    <xs:choice>
                        <xs:element name="port" type="xs:unsignedInt"/>
                        <xs:element name="portRefGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    </xs:choice>
                </xs:complexType>
            </xs:element>
            <xs:element name="icmpMessage" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ServicePolicyObject">
        <xs:complexContent>
            <xs:extension base="BasePolicyObject">
                <xs:sequence minOccurs="0" maxOccurs="1">
                    <xs:element name="serviceParameters" type="ServiceParameters" minOccurs="1"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 30：ServicePolicyObject XML スキーマ

3.1.4.5 InterfaceRolePolicyObject

InterfaceRolePolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。InterfaceRolePolicyObject は、FastEthernet0、内部、外部などのインターフェイスまたはインターフェイス グループを表示するパターンを定義します。継承された gid 属性は InterfaceRole PolicyObject の「インスタンス」に固有 ID を指定します。ポリシー定義は gid 値で InterfaceRolePolicyObject を参照します。

次の表に、InterfaceRolePolicyObject の内容を定義します。

要素	タイプ	コメント
pattern	文字列	1 つ以上のインターフェイス ロールの regex パターンを定義します。「*」は、すべてのインターフェイスに一致します。

表 25：InterfaceRolePolicyObject クラス定義

```
<xs:complexType name="InterfaceRolePolicyObject">
        <xs:complexContent>
            <xs:extension base="BasePolicyObject">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="pattern" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 31：InterfaceRolePolicyObject XML スキーマ

3.1.4.6 TimeRangePolicyObject

TimeRangePolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。TimeRangePolicyObject は時間範囲と関連の繰り返しを定義します。ポリシー定義は gid 値で TimeRangePolicyObject を参照します。TimeRangePolicyObject の内容は空にすることはできません。また、TimeRangePolicyObject の「グループ化」がサポートされていません

次の表に、TimeRangePolicyObject の内容を定義します。

要素	タイプ	コメント
startTime	dateTime	開始時刻を指定します。開始時刻が指定されていない場合、この範囲によって指定された時間範囲は「すでに開始」されていることを意味します。
endTime	dateTime	終了時刻を指定します。終了時刻が指定されていない場合、この範囲によって指定された時間範囲は「終了しない」ことを意味します。
recurrence	複合タイプ	時間範囲は、時間範囲に対して繰り返しを指定するゼロ以上の繰り返し範囲とすることができます。これは選択タイプの複合要素であり、「曜日」ベース繰り返しパターンまたは「週」ベース繰り返しパターンを含む可能性があります。
recurrence.dayOfWeekInterval	複合タイプ	「曜日」ベース繰り返しを定義します。
recurrence.dayOfWeekInterval.dayOfWeek	文字列	dayOfWeek 属性は、パターンを適用すべき曜日を定義します。また、次のストリング値のうち「1 つ」をとることができます。 • **Everyday**：毎日繰り返す（または） • **Weekday**：平日に繰り返す（または） • **Weekends**：週末に繰り返す（または） • 特定の曜日のカンマ区切りリスト（月曜日、火曜日など） その他
recurrence.dayOfWeekInterval.startTime	Time	開始時刻を指定します。
recurrence.dayOfWeekInterval.endTime	Time	終了時刻を指定します。
recurrence.weeklyInterval	複合タイプ	「毎週」ベースの繰り返しを定義します。
recurrence.weekInterval.startDay	文字列	開始日を定義します（「月曜日」、「火曜日」など）。
recurrence.weeklyInterval.startTime	Time	開始時刻を指定します。

recurrence.weeklyInterval.endDay	文字列	終了日を定義します（「月曜日」、「火曜日」など）。
recurrence.weeklyInterval.endTime	Time	終了時刻を指定します。

表 26：TimeRangePolicyObject クラス定義

```
<xs:complexType name="TimeRangePolicyObject">
    <xs:complexContent>
        <xs:extension base="BasePolicyObject">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="startTime" type="xs:dateTime" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="endTime" type="xs:dateTime" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="recurrence" minOccurs="0" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="dayOfWeekInterval">
                                <xs:complexType>
                                    <xs:sequence minOccurs="1" maxOccurs="1">
                                        <xs:element name="dayOfWeek" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="startTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="endTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="weeklyInterval">
                                <xs:complexType>
                                    <xs:sequence minOccurs="1" maxOccurs="1">
                                        <xs:element name="startDay" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="startTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="endDay" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="endTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                        </xs:choice>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 32：TimeRangePolicyObject XML スキーマ

3.1.4.7 SLA モニタ ポリシー オブジェクト

SLAMonitorPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。SLAMonitorPolicyObject は SLA の監視ポリシーを定義します。ポリシー定義は gid 値で SLAMonitorPolicyObject を参照します。

次の表に、SLAMonitorPolicyObject の内容を定義します。

要素.サブ要素	タイプ	コメント
slaId	int	SLA 操作の ID 番号。
interface	ObjectIdentifier	可用性をテストするためにモニタリング対象のアドレスに対して送信される、すべての ICMP エコー要求の送信元インターフェイス。インターフェイスやインターフェイス ロールの名前を入力するか、または [**Select**] をクリックしてリストから名前を選択するか新しいインターフェイス ロールを作成します。
monitoredAddress	string	SLA 操作によって可用性をモニタリングされる IP アドレス。
dataSizeInBytes	int	ICMP 要求パケット ペイロードのサイズ（バイト単位）。
thresholdInMilliSeconds	int	上昇しきい値が宣言されるまでに、ICMP エコー要求のあとに経過する必要のある時間（ミリ秒単位）。
timeoutInMilliSeconds	int	SLA 操作が ICMP エコー要求への応答を待機する時間（ミリ秒単位）。
frequencyInSeconds	int	ICMP エコー要求の送信頻度（秒単位）。
toS	int	ICMP 要求パケットの IP ヘッダー内に定義されたタイプ オブ サービス（ToS）。値の範囲は 0 ～ 255 です。デフォルトは 0 です。
numberOfPackets	int	送信されたパケットの数。値の範囲は 1 ～ 100 です。デフォルトは 1 パケットです。

表 27：SLAMonitorPolicyObject クラス定義

```
<xs:complexType name="SLAMonitorPolicyObject">
        <xs:complexContent>
            <xs:extension base="BasePolicyObject">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="slaId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="monitoredAddress" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="dataSizeInBytes" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                    <xs:element name="thresholdInMilliSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                    <xs:element name="timeoutInMilliSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                    <xs:element name="frequencyInSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                    <xs:element name="toS" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="numberOfPackets" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 33：SLAMonitorPolicyObject XML スキーマ

3.1.4.8 標準 ACE ポリシー オブジェクト

StandardACEPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。StandardACEPolicyObject は標準の IP アクセス コントロール エントリを定義します。ポリシー定義は gid 値で StandardACEPolicyObject を参照します。

要素.サブ要素	タイプ	コメント
networkGID	ObjectIdentifier	トラフィックの送信元または宛先。
doLogging	boolean	トラフィックがエントリ基準を満たしたときにログ エントリを作成するかどうか。
permit	boolean	一致が存在する場合に実行するアクションを示します。

表 28：StandardACEPolicyObject クラス定義

```
<xs:complexType name="StandardACEPolicyObject">
    <xs:complexContent>
        <xs:extension base="BasePolicyObject">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="networkGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="doLogging" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="permit" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 34：StandardACEPolicyObject XML スキーマ

3.1.4.9 拡張 ACE ポリシー オブジェクト

ExtendedACEPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。ExtendedACEPolicyObject は拡張アクセス コントロール エントリを定義します。ポリシー定義は gid 値で ExtendedACEPolicyObject を参照します。

要素.サブ要素	タイプ	コメント
sourceGID	ObjectIdentifier	トラフィックの送信元。
destinationGID	ObjectIdentifier	トラフィック宛先。
serviceGID	ObjectIdentifier	対象とするトラフィックのタイプを定義するサービス

doLogging	文字列	ロギングが PIX、ASA、FWSM デバイスに対してイネーブルの場合は「true」値を含み、そうでない場合は「false」を含みます。logInterval および logLevel 要素が指定されていない場合は、「デフォルトのロギング」がイネーブルであることを意味します。
logInterval	文字列	ロギング間隔を秒単位で指定します。これが指定されていると、「ACE のロギングごとにイネーブル」を意味します。
logLevel	文字列	ログレベルを指定します。「緊急」、「アラート」、「重要」、「エラー」、「警告」、「通知」、「情報」または「デバッグ」のいずれかです。これが指定されていると、「ACE のロギングごとにイネーブル」を意味します。
logOption	文字列	IOS ログインを指定するために使用されます。IOS ロギングがイネーブルの場合「log」が含まれます。IOS ロギングがイネーブルかつ入力ログが IOS デバイスに対してイネーブルの場合は、「log-input」が含まれます。
permit	boolean	これが許可 ACE の場合は true、拒否の場合は false となります。

表 29：ExtendedACEPolicyObject クラス定義

```
    <xs:complexType name="ExtendedACEPolicyObject">
        <xs:complexContent>
            <xs:extension base="BasePolicyObject">
                <xs:sequence>
                    <xs:element name="sourceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="destinationGID" type=" ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="serviceGID" type=" ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="doLogging" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="logInterval" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="logLevel" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="logOption" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="permit" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 35：ExtendedACEPolicyObject XML スキーマ

3.1.4.10 ACL ポリシー オブジェクト

要素.サブ要素	タイプ	コメント
references	複合タイプ	ACE への参照のリスト
references .sequenceNumber	Unsigned Int	このエントリのシーケンス番号
references .aclObjectReferenceGID	ObjectIdentifier	ACE ポリシー オブジェクトへの参照
References.aceReferenceGID	ObjectIdentifier	標準/拡張 ACE ポリシー オブジェクトへの参照

表 30：ACLPolicyObject クラス定義

```
<xs:complexType name="ACLPolicyObject">
    <xs:complexContent>
        <xs:extension base="BasePolicyObject">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="references" minOccurs="1" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="sequenceNumber" type="xs:unsignedInt"
minOccurs="1" maxOccurs="1"/>
                            <xs:choice>
                                <xs:element name="aclObjectReferenceGID"
type="ObjectIdentifier"/>
                                <xs:element name="aceReferenceGID"
type="ObjectIdentifier"/>
                            </xs:choice>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 36：ACLPolicyObject XML スキーマ

3.1.4.11 SecurityGroupPolicyObject

SecurityGroupPolicyObject は **BasePolicyObject** クラスから拡張され、属性をすべて継承します。SecurityGroupPolicyObject はセキュリティ タグまたは名前を定義します。

ポリシー定義は gid 値で SecurityGroupPolicyObject を参照します。SecurityGroupPolicyObject の内容は、継承された **isGroup** 属性が true に設定される場合に「空」になることがあります。そのような場合、SecurityGroupPolicyObject 自体に「他のセキュリティ グループ ポリシー オブジェクト」への参照が含まれます。

このような PolicyObject の GID 値のリストは **refs** に継承される属性から取得できます。「グループ」SecurityGroupPolicyObject も、リテラル セキュリティ タグ/名前または含まれているポリシー オブジェクトを表示する複数の secuTag 要素を含めることができます。セキュリティ ポリシー オブジェクトの完全な内容は、securityTag 要素から取得できます。

このオブジェクトは API バージョン 1.1 以降で使用可能です。

要素	タイプ	コメント
securityTag	複合 （SecurityGrpObjectsRef）	参照先のポリシー オブジェクトのセキュリティ タグ/名前または GID を定義します。
securityTag .securityGrpObjectGID	ObjectIdentifier	参照先のセキュリティ グループ ポリシー オブジェクトの GID を定義します。
securityTag .secName	文字列	セキュリティ タグを定義します。
securityTag .secTag	文字列	セキュリティ タグまたは名前を定義します。

表 31：SecurityGroupPolicyObject（）クラス定義

```
<xs:complexType name="SecurityGroupPolicyObject">
  <xs:complexContent>
    <xs:extension base="BasePolicyObject">
      <xs:sequence>
        <xs:element name="securityTag" type="SecurityGrpObjectsRef" minOccurs="1"
maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

図 37：SecurityGroupObject XML スキーマ

```
<xs:complexType name="SecurityGrpObjectsRef">
  <xs:sequence>
    <xs:choice>
      <xs:element name="securityGrpObjectGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
      <xs:element name="secName" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="secTag" type="xs:string" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

図 38：SecurityGrpObjectsRef XML スキーマ

3.1.5 ポリシー派生クラス

このセクションおよび次のサブ セクションでは、この API でサポートされるポリシー クラスを定義します。

3.1.5.1 DeviceAccessRuleFirewallPolicy

DeviceAccessRuleFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。DeviceAccessRuleFirewallPolicy インスタンスでは、単一のアクセス コントロール エントリを表示します。BasePolicy クラスからの orderId 属性は、これらのルールの順序を定義します。

DeviceAccessRuleFirewallPolicy は NetworkPolicyObject、ServicePolicyObject、IdentityUserGroupPolicyObject または InterfaceRolePolicyObject オブジェクトを参照することがあります。

ソースおよび**宛先**要素は次のいずれかの**組み合わせ**を含めることができます。

- **networkObjectGIDs**：これには、ネットワーク ポリシー オブジェクトへの 1 つ以上の GID の参照が含まれます。
- **interfaceRoleObjectGIDs**：これには、インターフェイス ロール ポリシー オブジェクトへの 1 つ以上の GID の参照が含まれます。
- **ipv4Data**：1 つ以上のリテラル IPv4 アドレス

2 つのネットワーク ポリシー オブジェクトを参照し 1.1.1.1/32 などのリテラル アドレスを含める宛先エレメントを指定することが可能です。

ソースおよび宛先サブ要素の*外部*で指定された **interfaceRoleObjectID** は、ACL が適用されているインターフェイスを指定します。

次の表に、DeviceAccessRuleFirewallPolicy の内容を定義します。

要素	タイプ	コメント
isEnabled	ブール	ルールがイネーブルの場合は true、そうでない場合は false です。
direction	文字列	In または Out。
permit	boolean	true は許可、fale は拒否を示します。
interfaceRoleObjectIDs	ObjectIdentifierList	一連の InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID のリストを送信します。対応する InterfaceRole オブジェクトの GID 属性の ID のリンクを送信します。
users	複合タイプ	ルールが適用されるユーザ グループおよびユーザを含みます（ASA デバイスのバージョンが 8.4（2））以降の場合のみ適用可能）。
users. identityUserGrpObject GIDs	ObjectIdentifierList	IdentityUserGroupPolicyObject オブジェクト GID のリストを参照します。
users. userNameData	文字列	ユーザのリスト。
users. userGroupData	文字列	ユーザ グループのリスト。

要素	タイプ	コメント
sources	複合タイプ	送信元ネットワークとインターフェイス ロールのコンテナ。
sources.networkObjectGIDs	ObjectIdentifierList	一連のネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID のリスト。ID は、対応するネットワーク オブジェクトの GID 属性にリンクします。
sources.interfaceRoleObjectGIDs	ObjectIdentifierList	一連の InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID のリストを送信します。対応する InterfaceRole オブジェクトの GID 属性の ID のリンクを送信します。
sources.ipv4Data	文字列	IPv4 ホスト、ネットワーク、または範囲を含む複数の IPv4Data 要素。
destination	複合タイプ	宛先ネットワークとインターフェイス ロールのコンテナ。
destination.networkObjectGIDs	ObjectIdentifierList	一連のネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID のリスト。ID は、対応するネットワーク オブジェクトの GID 属性にリンクします。
destination.interfaceObjectGIDs	ObjectIdentifierList	一連の InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID のリストを送信します。対応する InterfaceRole オブジェクトの GID 属性の ID のリンク。
destination.ipv4Data	文字列	IPv4 ホスト、ネットワーク、または範囲を含む複数の IPv4Data 要素。
serviceObjectIDs	ObjectIdentifierList	一連のサービス ポリシー オブジェクトを参照する ObjectIdentifier ID のリスト。対応するサービス オブジェクトの GID 属性への ID リンク。
serviceParameters	ServiceParameters	プロトコルおよびポートの詳細を含むサービス データを保存できる複数のサービス データ要素。
serviceParameters.protocol	文字列	tcp、udp などのプロトコルを定義する文字列。
serviceParameters.sourcePort	複合タイプ	送信元ポートのデータを保持する要素のコンテナ。これは、選択タイプの複合要素であるため、サブ要素のうち 1 つのみ定義されます。
serviceParameters.sourcePort.port	文字列	実際のポート値。<開始ポート>-<終了ポート> の形式で指定される範囲（例：1-65535）にもできます。
serviceParameters.sourcePort.portRef	ObjectIdentifier	PortListPolicyObject への Gid の参照。
serviceParameters.destinationPort	複合タイプ	宛先ポートのデータを保持する要素のコンテナ。これは、選択タイプの複合要素であるため、サブ要素のうち 1 つのみ定義されます。
serviceParameters.destinationPort.port	文字列	実際のポート値。<開始ポート>-<終了ポート> の形式で指定される範囲（例：1-65535）にもできます。

要素	タイプ	コメント
serviceParameters.destinationPort.portRef	ObjectIdentifier	PortListPolicyObject への Gid の参照。
serviceParameters.icmpMessage	文字列	ICMP メッセージの内容。
logOptions	複合タイプ	ルールに関連するロギング オプションを保存するルート エレメント。
logOptions.isFirewallLoggingEnabled	ブール	ロギングが PIX、ASA、FWSM ルールでイネーブルかどうかを示します。
logOptions.isDefaultLogging	ブール	デフォルトのロギングがイネーブルかどうか。これは isFirewallLoggingEnabled が true の場合のみ、PIX、ASA、および FWSM デバイスに適用可能です。
logOptions.loggingInterval	Int	ロギング間隔を秒単位で指定します。これは isDefaultLogging が false または指定されていない場合に、PIX、FWSM、および ASA に適用可能です。（ACE のロギングごとに適用）
logOptions.loggingLevel	文字列	ログ レベル（「緊急」、「アラート」、「重要」、「エラー」、「警告」、「通知」、「情報」、「デバッグ」）を指定します。これは isDefaultLogging が false または指定されていない場合に、PIX、FWSM、および ASA に適用可能です。（ACE のロギングごとに適用）
logOptions isIOSLoggingEnabled	ブール	IOS デバイスのロギングがイネーブルの場合に true に設定します。IOS のみに適用可能
logOptions.isLogInput	ブール	ログの入力がイネーブルの場合は true。これは isIOSLoggingEnabled が true の場合のみ、IOS デバイスに適用可能です。
iosOptions	文字列	IOS オプションに許可されている値は、「none」、「Fragment」、「Established」です。
timeRangeObjectId	ObjectIdentifier	時間範囲がこのルールに指定されている場合、TimeRangePolicyObject を参照します。

表 32：DeviceAccessRuleFirewallPolicy クラス定義

このクラスの XML スキーマについては、このマニュアルの最後の XML スキーマを参照してください。

3.1.5.1.1 ポリシー設定デバイスの応答例

```
<?xml version="1.0" encoding="utf-8"?>
<n:policyConfigDeviceResponse xmlns:n="csm" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" >
    <policy>
        <deviceAccessRuleFirewallPolicy>
            <gid>00000000-0000-0000-0000-000000000889</gid>
            <name>string</name>
            <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
            <updatedByUser>admin</updatedByUser>
            <lastCommitTime>2012-12-05T09:18:39.371Z </lastCommitTime>
            <activityName>admin_05.Dec.2012_01.17.10</activityName>
            <type>Access Rule</type>
            <orderId>0</orderId>
            <isMandatoryAggregation>false</isMandatoryAggregation>
            <description/>
            <isEnabled>true</isEnabled>
            <direction>In</direction>
            <permit>true</permit>
            <interfaceRoleObjectGIDs>00000000-0000-0000-0000-
000000000123</interfaceRoleObjectGIDs>
            <sources>
                <networkObjectGIDs>00000000-0000-0000-0000-
000000000124</networkObjectGIDs>
                <interfaceRoleObjectGIDs>00000000-0000-0000-0000-
000000000123</interfaceRoleObjectGIDs>
                <ipv4Data>1.1.1.1/32</ipv4Data>
            </sources>
            <destinations>
                <networkObjectGIDs>00000000-0000-0000-0000-
000000000125</networkObjectGIDs>
                <interfaceRoleObjectGIDs>00000000-0000-0000-0000-
000000000123</interfaceRoleObjectGIDs>
                <ipv4Data>2.2.2.2</ipv4Data>
            </destinations>
            <services>
                <serviceObjectGIDs>00000000-0000-0000-0000-
000000000126</serviceObjectGIDs>
                <serviceParameters>
                    <protocol>tcp</protocol>
                    <sourcePort>
                        <port>80</port>
                    </sourcePort>
                    <destinationPort>
                        <port>80</port>
                    </destinationPort>
                </serviceParameters>
            </services>
            <logOptions>
                <isFirewallLoggingEnabled>false</isFirewallLoggingEnabled>
            </logOptions>
            <iosOptions>None</iosOptions>
            <timeRangeObjectGID>00000000-0000-0000-0000-
000000000127</timeRangeObjectGID>
        </deviceAccessRuleFirewallPolicy>
```

```
</policy>
<policyObject>
    <networkPolicyObject>
        <gid>00000000-0000-0000-0000-000000000124</gid>
        <name>mySource</name>
        <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
        <type>Network</type>
        <comment/>
        <isProperty>false</isProperty>
        <subType>Host</subType>
        <isGroup>false</isGroup>
        <refGIDs/>
        <ipv4Data>1.1.2.2</ipv4Data>
    </networkPolicyObject>
    <networkPolicyObject>
        <gid>00000000-0000-0000-0000-000000000125</gid>
        <name>myDest</name>
        <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
        <type>Network</type>
        <comment/>
        <isProperty>false</isProperty>
        <subType>Host</subType>
        <isGroup>false</isGroup>
        <refGIDs/>
        <ipv4Data>1.1.3.3</ipv4Data>
    </networkPolicyObject>
    <portListPolicyObject>
        <gid>00000000-0000-0000-0000-000000002782</gid>
        <name>MyPortList</name>
        <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
        <type>PortList</type>
        <comment/>
        <isProperty>false</isProperty>
        <subType>Host</subType>
        <isGroup>false</isGroup>
        <refGIDs/>
        <startPort>514</startPort>
        <endPort>514</endPort>
    </portListPolicyObject>
    <servicePolicyObject>
        <gid>00000000-0000-0000-0000-000000000126</gid>
        <name>string</name>
        <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
        <type>Service</type>
        <comment/>
        <isProperty>false</isProperty>
        <subType/>
        <isGroup>false</isGroup>
        <refGIDs/>
        <serviceParameters>
            <protocol>udp</protocol>
            <sourcePort>
            <portRefGID>00000000-0000-0000-0000-000000002782</portRefGID>
        </sourcePort>
        <destinationPort>
            <port>514</port>
```

```
                </destinationPort>
                <icmpMessage>string</icmpMessage>
            </serviceParameters>
                </servicePolicyObject>
            <interfaceRolePolicyObject>
                <gid>00000000-0000-0000-0000-000000000123</gid>
                <name>FA1</name>
                <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
                <type>InterfaceRole</type>
                <comment/>
                <isProperty>false</isProperty>
                <subType/>
                <isGroup>false</isGroup>
                <refGIDs/>
                <pattern>FastEthernet0/0</pattern>
            </interfaceRolePolicyObject>
            <timeRangePolicyObject>
                <gid>00000000-0000-0000-0000-000000000127</gid>
                <name>string</name>
                <lastUpdateTime>1697-02-01T00:00:00Z</lastUpdateTime>
                <type>TimeRange</type>
                <comment/>
                <isProperty>false</isProperty>
                <subType/>
                <isGroup>false</isGroup>
                <refGIDs/>
                <startTime>1697-02-01T00:00:00Z</startTime>
                <endTime>1697-02-01T00:00:00Z</endTime>
                <recurrence>
                    <weeklyInterval>
                        <startDay>Monday</startDay>
                        <startTime>13:20:00-05:00</startTime>
                        <endDay>Friday</endDay>
                        <endTime>13:20:00-05:00</endTime>
                    </weeklyInterval>
                </recurrence>
            </timeRangePolicyObject>
        </policyObject>
</n:policyConfigDeviceResponse>
```

3.1.5.2 DeviceAccessRuleUnifiedFirewallPolicy

DeviceAccessRuleUnifiedFirewallPolicy は基本 DeviceAccessRuleFirewallPolicy クラスから拡張され、属性をすべて継承します。DeviceAccessRuleUnifiedFirewallPolicy のインスタンスが単一の統合型アクセス コントロール エントリを表示します。BasePolicy クラスからの orderId 属性は、これらのルールの順序を定義します。

DeviceAccessRuleUnifiedFirewallPolicy はさらに SecurityGroupPolicyObject も参照することがあります。

SecurityGrpObjectsRef のリストである SecurityGrpObjectsRefs の **sourceSG** および **destinationSG** 要素。SecurityGrpObjectsRef のセクション 3.1.4.11 を参照してください。

XML コンテンツには、ロギング、フラグメント、確立などの IOS オプションを除き、基本のすべての属性が含まれます。

このポリシーは、API バージョン 1.1 以降で使用できます。

要素	タイプ	コメント
sourceSG	複合 （SecurityGrpObjectsRefs）	着信パケットの送信元のセキュリティ タグまたはオブジェクトを定義します。
destinationSG	複合 （SecurityGrpObjectsRefs）	着信パケットの宛先としてセキュリティ タグまたはオブジェクトを定義します。

表 33：DeviceAccessRuleUnifiedFirewallPolicy クラス定義

```
<xs:complexType name="DeviceAccessRuleUnifiedFirewallPolicy">
  <xs:complexContent>
    <xs:extension base="DeviceAccessRuleFirewallPolicy">
      <xs:sequence minOccurs="1" maxOccurs="1">
        <xs:element name="sourceSG" type="SecurityGrpObjectsRefs" minOccurs="0" maxOccurs="1"/>
        <xs:element name="destinationSG" type="SecurityGrpObjectsRefs" minOccurs="0"
maxOccurs="1"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
  </xs:complexType>
```

図 37：DeviceAccessRuleUnifiedFirewallPolicy XML スキーマ

3.1.5.3 DeviceStaticRoutingFirewallPolicy

要素.サブ要素	タイプ	コメント
FwStaticRoutePolicy	複合タイプ	
interfaceGID	ObjectIdentifier	このスタティック ルートが適用されるインターフェイス。
networks	複合タイプ	宛先ネットワーク。1 つ以上の IP アドレス/ネットマスク エントリ、1 つ以上のネットワーク/ホスト オブジェクト、または両方の組み合わせを指定できます。エントリはカンマで区切ります。 デフォルト ルートを指定するには、0.0.0.0 を使用します。0.0.0.0 の IP アドレスは、0 と省略できます。
networks.networkObjectIdentifierGIDs	ObjectIdentifierList	ポリシー オブジェクトへの参照としての値
networks.ipv4Data	string	raw 文字列として値
gateway	複合タイプ	このルートのネクスト ホップ アドレスであるゲートウェイ ルータ。
gateway.hostObjectIdentifierGID	ObjectIdentifier	ポリシー オブジェクトへの参照
gateway.ipv4Data	string	ゲートウェイ IP アドレスの raw 文字列値
metric	unsignedInt	宛先ネットワークへのホップ数。有効値の範囲は 1 ～ 255 で、デフォルト値は 1 です。
tunnelled	boolean	これがトンネルのルートかどうかを示します。
slaMonitorGID	ObjectIdentifier	モニタリング ポリシーを定義する Service Level Agreement（SLA）オブジェクトの名前。

表 34：DeviceStaticRoutingFirewallPolicy

```
<xs:complexType name="DeviceStaticRoutingFirewallPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="networks" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence minOccurs="1" maxOccurs="1">
                                <xs:element name="networkObjectIdentifierGIDs"
type="ObjectIdentifierList" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="ipv4Data" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="gateway" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence minOccurs="1" maxOccurs="1">
                                <xs:element name="hostObjectIdentifierGID"
type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="ipv4Data" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="metric" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="tunnelled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="slaMonitorGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>

                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 38：DeviceStaticRoutingFirewallPolicy

3.1.5.4 DeviceStaticRoutingRouterPolicy

要素.サブ要素	タイプ	コメント
destinationNetwork	複合タイプ	宛先ネットワーク設定の要素のコンテナ。
destinationNetwork.useAsDefaultRoute	boolean	スタティック ルートがこのルータによって転送される不明なパケットのデフォルト ルートかどうかを示します。
destinationNetwork.prefix	複合タイプ	スタティック ルートの宛先 IP アドレス。
destinationNetwork.prefix.networkObjectReferenceGID	ObjectIdentifier	オブジェクト参照としてのスタティック ルートの宛先 IP アドレス。
destinationNetwork.prefix.ipV4Data	string	文字列形式のスタティック ルートの宛先 IP アドレス。
fowarding	複合タイプ	
fowarding.forwardingInterfaceGID	ObjectIdentifier	このルータのネクスト ホップ アドレスであるゲートウェイ ルータに関連付けられるインターフェイス名。
fowarding.forwardingIPAddress	複合タイプ	このルータのネクスト ホップ アドレスであるゲートウェイ ルータに関連付けられる IP アドレス。
fowarding.forwardingIPAddress.hostObjectReferenceGID	ObjectIdentifier	オブジェクト参照としての IP アドレス。
fowarding.forwardingIPAddress.ipv4HostAddress	string	文字列値としての IP アドレス。
distanceMetric	unsignedInt	ゲートウェイ IP から宛先へのホップ数。メトリックによって、このルートのプライオリティが決まります。ホップが少ないほど、コストが低くなるため、ルートに割り当てられるプライオリティは高くなります。 2 つのルーティング エントリで同じネットワークが指定されている場合、メトリックの小さい（つまり、プライオリティが高い）エントリが選択されます。
isPermanentRoute	boolean	スタティック ルートが永続的なルートとして定義されるかどうかを示します。永続的なルートとは、インターフェイスがシャットダウンされるか、ルータが次のルータと通信できない場合でも、削除されないことを意味します。

表 35：DeviceStaticRoutingRouterPolicy

```
<xs:complexType name="DeviceStaticRoutingRouterPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="destinationNetwork" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="useAsDefaultRoute" type="xs:boolean"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="prefix" minOccurs="1" maxOccurs="1">
                                    <xs:complexType>
                                        <xs:choice>
                                            <xs:element name="networkObjectReferenceGID"
type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                            <xs:element name="ipV4Data" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                        </xs:choice>
                                    </xs:complexType>
                                </xs:element>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="fowarding" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="forwardingInterfaceGID"
type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="forwardingIPAddress" minOccurs="1"
maxOccurs="1">
                                    <xs:complexType>
                                        <xs:choice>

        <xs:element name="hostObjectReferenceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="ipv4HostAddress"
type="xs:string" minOccurs="1" maxOccurs="1"/>
                                        </xs:choice>
                                    </xs:complexType>
                                </xs:element>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="distanceMetric" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                    <xs:element name="isPermanentRoute" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 39：DeviceStaticRoutingRouterPolicy

3.1.5.5 DeviceBGPRouterPolicy

要素.サブ要素	タイプ	コメント
asNumber	unsignedLong	ルータが配置されている自律システムの数。有効な値の範囲は 1 ～ 65535 です。この番号によって、BGP ルーティング プロセスがイネーブルになります。
networks	複合タイプ	BGP ルートに関連付けられるネットワーク。
networks.networkObjectIdentifierGIDs	ObjectIdentifierList	ポリシー オブジェクトへの参照としての値
networks.ipv4Data	string	文字列としての値
neighbors	複合タイプ	ルータの*内部*ネイバー（ルータと同じ AS 内に配置）および*外部*ネイバー（異なる AS 内に配置）。
neighbors.ipAddress	複合タイプ	ネイバー アドレス
neighbors.ipAddress.networkObjectGIDs	ObjectIdentifierList	ポリシー オブジェクト リストへの参照としてのアドレス
neighbors.ipAddress.ipv4Data	string	文字列値としてのアドレス
neighbors.asNumber	unsignedLong	ネイバーの AS 番号
autoSummary	boolean	自動集約をイネーブルにします。サブネットが IGP（RIP、OSPF、EIGRP など）から BGP に再配布されるときに、この BGP バージョン 3 機能によってネットワーク ルートだけが BGP テーブルに注入されます。自動サマライズによって、ルータで保持する必要があるルーティング テーブルのサイズと複雑さが低減します。
synchronization	boolean	同期をイネーブルにします。選択すると、同期がイネーブルになります。この機能を使用して、アドバタイズするルートに関してネットワーク内のすべてのルータの一貫性が確保されるようにします。同期によって、BGP は、IGP がルーティング情報を AS 全体に伝播するまで待機します。 選択を解除すると、同期がディセーブルになります。このルータが別の AS からのトラフィックを第三の AS に渡さない場合、または AS 内のすべてのルータが BGP を実行している場合、同期をディセーブルにすることができます。この機能をディセーブルにすると、IGP が伝送する必要があるルートの数が減少し、コンバージェンス時間が向上するという利点があります。これはデフォルトです。

logNeighbor	boolean	選択すると、BGP ネイバーのリセット、ネットワークへの接続、または切断時に生成されるメッセージのロギングがイネーブルになります。
redistributionEntry	複合タイプ	再配布を BGP Autonomous System（AS）に実行するときの再配布設定。複数にできます。
redistributionEntry.protocol	複合タイプ	再配布されているプロトコル。
redistributionEntry.protocol.static	文字列	プロトコルの選択。指定できる値は「IP」または「OSI」です。
redistributionEntry.protocol.connected	文字列	プロトコルの選択。
redistributionEntry.protocol.rip	文字列	プロトコルの選択。
redistributionEntry.protocol.eigrp	複合タイプ	プロトコルの選択。
redistributionEntry.protocol.eigrp.asNumber	unsignedInt	EIGRP が選択されている場合は AS 番号。
redistributionEntry.protocol.ospf	複合タイプ	プロトコルの選択
redistributionEntry.protocol.ospf.processId	UnsignedInt	EIGRP が選択されている場合はプロセス番号。
redistributionEntry.protocol.ospf.match	文字列	複数一致基準。各一致基準の許容値のうち 1 つを取ります。 • **Internal**：自律システム（AS）の内部へのルートが再配布されます。 • **External1**：AS の外部にルートするタイプ 1 が再配布されます。 • **External2**：AS の外部にルートするタイプ 2 が再配布されます。 • **NSSAExternal1**：Not-So-Stubby Area（NSSA）の外部にルートするタイプ 1 が再配布されます。 • **NSSAExternal2**：NSSA の外部にルートするタイプ 2 が再配布されます。
redistributionEntry.metric	unsignedLong	再配布されるルートのプライオリティを決定する値。

表 36：DeviceBGPRouterPolicy クラス定義

このクラスの XML スキーマについては、このマニュアルの最後の XML スキーマを参照してください。

3.1.5.6 InterfaceNATRouterPolicy

InterfaceNATRouterPolicy は BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATRouterPolicy のインスタンスは、インターフェイスの内部および外部に 1 つ NAT を指定します。

InterfaceNATRouterPolicy は InterfaceRole PolicyObject を参照できます。

次の表に、InterfaceNATRouterPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
interfaceGID	ObjectIdentifier	内部インターフェイスまたは外部インターフェイスを示す InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID。この ID は、対応する InterfaceRole オブジェクトの GID 属性にリンクします。
isNatInside	boolean	これが NAT の内部インターフェイス（True）か外部インターフェイス（false）かを示すブール値

表 37：InterfaceNATRouterPolicy クラス定義

```
<xs:complexType name="InterfaceNATRouterPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="isNatInside" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 39：InterfaceNATRouterPolicy XML スキーマ

3.1.5.7 InterfaceNATStaticRulesRouterPolicy

InterfaceNATStaticRulesRouterPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATStaticRulesRouterPolicy インスタンスでは、スタティック NAT ルールを表示します。

InterfaceNATStaticRulesRouterPolicy は Network ポリシー オブジェクトおよび InterfaceRole ポリシー オブジェクトを参照できます。基本 orderId 属性はスタティック ルールの順序を定義します。

次の表に、InterfaceNATStaticRulesRouterPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
staticRuleType	Enumeration	このスタティック ルールで変換されるローカル アドレスのタイプ。 • [Static Host]：単一ホストでスタティック アドレス変換が必要な場合。 • [Static Network]：サブネットでスタティック アドレス変換が必要な場合。 • [Static Port]：単一ポートでスタティック アドレス変換が必要な場合。このオプションを選択した場合は、[Port Redirection] パラメータを定義する必要があります。
original	複合タイプ	IP アドレスを識別する複合タイプ要素、または変換するアドレスを表すネットワーク/ホスト オブジェクト。
original.ipv4Data	文字列	リテラル IP アドレス。
original.networkObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated	複合タイプ	元のアドレスが変換されるアドレスを含む複合タイプ要素。これは、特定の IP アドレスとネットワーク オブジェクトを含める、またはインターフェイスの指定に使用できます。インターフェイスを指定した場合は、インターフェイスに割り当てられた IP アドレスは、変換済みアドレスとして使用されます。
translated.originalIP	複合タイプ	IP データやネットワーク ポリシー オブジェクトを指定する複合タイプ。
translated.originalIP.ipv4Data	文字列	リテラル IP アドレス。
translated.originalIP networkObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated.interfaceGID	オブジェクト ID	InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID。
portRedirection	複合タイプ	アドレス変換のポート情報を指定する複合タイプ。これらのポートは、ルールのタイプとして [Static Port] を選択している場合にだけ使用できます。

要素.サブ要素	タイプ	コメント
portRedirection.protocol	文字列	これらのポートで使用する通信プロトコル（TCP または UDP）。
portRedirection.localPort	unsignedInt	送信元ネットワーク上のポート番号。有効な値の範囲は 1 ～ 65535 です。
portRedirection.globalPort.	unsignedInt	ルータがこの変換に使用されることを示す、宛先ネットワーク上のポート番号。有効な値の範囲は 1 ～ 65535 です。
settings	複合タイプ	詳細オプションを含むオプションの複合タイプ要素。
settings.noAlias	boolean	true の場合は、グローバル IP アドレス変換の自動エイリアスをディセーブルにします。
settings.noPayload	boolean	true の場合は、ペイロードの埋め込みアドレスまたはポートの変換が禁止されます。
settings.createExtTransEntry	boolean	true の場合は、拡張変換エントリ（アドレスおよびポート）が変換テーブルに作成されます。

表 38：InterfaceNATStaticRulesRouterPolicy クラス定義

```
<xs:complexType name="InterfaceNATStaticRulesRouterPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="staticRuleType" minOccurs="1" maxOccurs="1">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="Static Host"/>
                                <xs:enumeration value="Static Network"/>
                                <xs:enumeration value="Static Port"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="original" type="NetworkOrIPRef" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="translated" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="originalIP" type="NetworkOrIPRef"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="interfaceGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="portRedirection" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="protocol" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                <xs:element name="localPort" type="xs:unsignedInt"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="globalPort" type="xs:unsignedInt"
minOccurs="1" maxOccurs="1"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="settings" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="noAlias" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                <xs:element name="noPayload" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                <xs:element name="createExtTransEntry" type="xs:boolean"
minOccurs="1" maxOccurs="1"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 40：InterfaceNATStaticRulesRouterPolicy XML 定義

3.1.5.8 InterfaceNATDynamicRulesRouterPolicy

InterfaceNATDynamicRulesRouterPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATDynamicRulesRouterPolicy インスタンスが NAT のダイナミックなルールを指定します。

InterfaceNATDynamicRulesRouterPolicy は ACL PolicyObject および InterfaceRole ポリシー オブジェクトを参照できます。基本 orderId 属性はダイナミック ルールの順序を定義します。

次の表に、InterfaceNATDynamicRulesRouterPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
trafficFlowAclObjectGID	ObjectIdentifier	エントリがダイナミック変換を必要とするアドレスを定義する、Access Control Lists（ACL）ポリシー オブジェクト GID を参照します。
translatedAddress	複合タイプ	ダイナミック変換に使用するメソッドおよびアドレスを指定する複合タイプ要素。インターフェイス ロール オブジェクトまたはアドレス プールが含まれます。インターフェイス ロール ポリシー オブジェクトが参照されると、特定のインターフェイスに割り当てられた登録済み IP アドレスは、変換済みアドレスとして使用されます。
translatedAddress.interfaceGID	オブジェクト ID	InterfaceRole ポリシー オブジェクトを参照する ObjectIdentifier ID。
translatedAddress.addressPool	文字列	プレフィックスを含めた 1 つまたは複数のアドレス範囲。書式は min1-max1/prefix（CIDR 表記）を使用し、「prefix」は有効なネットマスクを示します。たとえば、`172.16.0.0-172.31.0.223/12` などです。
settings	複合タイプ	オプション設定を含む複合タイプ要素。
settings.enablePortTrans	ブール	true の場合、アドレス プールのグローバル アドレスの提供が枯渇していると、ルータはポート アドレッシング（PAT）を使用します。false の場合は PAT は使用されません。
settings.noTransVPN	ブール	True の場合、アドレス変換は VPN トラフィックに対して実行されません。false の場合、ルータは、NAT ACL とクリプト ACL 間でアドレスが重複している場合に、VPN トラフィックに対してアドレス変換を実行します。

表 39：InterfaceNATDynamicRulesRouterPolicy クラス定義

```
<xs:complexType name="InterfaceNATDynamicRulesRouterPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="trafficFlowAclObjectGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
                    <xs:element name="translated" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="interfaceGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="addressPool" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="settings" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="enablePortTrans" type="xs:boolean"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="noTransVPN" type="xs:boolean"
minOccurs="1" maxOccurs="1"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 41：InterfaceNATDynamicRulesRouterPolicy XML スキーマ

3.1.5.9 DeviceNATTimeoutsRouterPolicy

DeviceNATTimeoutsRouterPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。DeviceNATTimeoutsRouterPolicy インスタンスは、ポート アドレス（オーバーロード）変換用に NAT タイムアウト値を指定します。

次の表に、DeviceNATTimeoutsRouterPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
maxEntries	unsignedLong	ダイナミック NAT テーブルに格納できるエントリの最大数。1～2147483647 の値に対応します。指定しない場合、テーブル内のエントリの数が無制限であることを示します。
timeoutInSecs	unsignedLong	ダイナミック変換が期限切れになるまでの秒数。PAT（オーバーロード）変換には適用されません。デフォルトは 86400 秒（24 時間）です。
udpTimeoutInSecs	unsignedLong	ユーザ データグラム プロトコル（UDP）ポートに適用されるタイムアウト値。デフォルトは 300 秒（5 分）です。
dnsTimeoutInSecs	unsignedLong	Domain Naming System（DNS; ドメイン ネーミング システム）サーバ接続に適用されるタイムアウト値。デフォルトは 60 秒です。
tcpTimeoutInSecs	unsignedLong	伝送制御プロトコル（TCP）ポートに適用されるタイムアウト値。デフォルトは 86400 秒（24 時間）です。
finRstTimeoutInSecs	unsignedLong	終了（FIN）パケットまたはリセット（RST）パケット（どちらも接続を終了させる）が TCP ストリーム内で検出された場合に適用されるタイムアウト値。デフォルトは 60 秒です。
icmpTimeoutInSecs	unsignedLong	インターネット制御メッセージ プロトコル（ICMP）フローに適用されるタイムアウト値。デフォルトは 60 秒です。
pptpTimeoutInSecs	unsignedLong	NAT Point-to-Point Tunneling Protocol（PPTP; ポイントツーポイント トンネリング プロトコル）フローに適用されるタイムアウト値。デフォルトは 86400 秒（24 時間）です。
synTimeoutInSecs	unsignedLong	同期伝送（SYN）メッセージ（正確なクロッキングに使用）が検出されたあと、TCP フローに適用されるタイムアウト値。デフォルトは 60 秒です。

表 40：DeviceNATTimeoutsRouterPolicy クラス定義

```
<xs:complexType name="DeviceNATTimeoutsRouterPolicy">
    <xs:complexContent>
        <xs:extension base="BasePolicy">
            <xs:sequence>
                <xs:element name="maxEntriesInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="timeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="udpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="dnsTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="tcpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="finRstTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="icmpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="pptpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="synTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
```

図 42：DeviceNATTimeoutsRouterPolicy

3.1.5.10 InterfaceNATAddressPoolFirewallPolicy

InterfaceNATAddressPoolFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATAddressPoolFirewallPolicy インスタンスはダイナミックな NAT ルールで使用されるグローバル アドレス プールを管理します。このポリシーは、PIX、FWSM、および ASA 8.3 以前に適用されます。

次の表に、InterfaceNATAddressPoolFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
interfaceGID	ObjectIdentifier	マッピングされた IP アドレスが使用される InterfaceRole ポリシー オブジェクト GID のインターフェイスを参照します。
poolId	unisignedInt	このアドレス プールの一意の識別番号（1 ～ 2147483647 の整数）。ダイナミックな NAT ルールを設定する場合は、変換に使用されるアドレス プールを指定するには、プール ID が使用されます。
ipAddressRange	複合タイプ	このアドレス プールに割り当てられるアドレスを含む複合タイプ要素。アドレスは、ネットワーク ポリシー オブジェクトへのリテラル IPv4 アドレスまたは参照の組み合わせを含めることができます。
ipAddressRange.ipv4Data	文字列	リテラル IP アドレス。
ipAddressRange networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
interfaceKeyword	文字列	「インターフェイス」キーワードを指定した場合、ポート アドレス変換が指定されたインターフェイスでイネーブルであることを示します。

表 41：InterfaceNATAddressPoolFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATAddressPoolFirewallPolicy">
    <xs:complexContent>
        <xs:extension base="BasePolicy">
            <xs:sequence>
                <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="ipAddressRange" type="NetworkObjectsRefs" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="interfaceKeyword" type="xs:string" fixed="interface"
minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 43：InterfaceNATAddressPoolFirewallPolicy XML スキーマ

3.1.5.11 DeviceNATTransOptionsFirewallPolicy

DeviceNATTransOptionsFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。DeviceNATTransOptionsFirewallPolicy インスタンスで、選択したセキュリティ アプライアンスのネットワーク アドレス変換に影響するオプションを管理します。これらの設定は、デバイス上のすべてのインターフェイスに適用されます。

このポリシーは、PIX、FWSM、および ASA に適用されます。

次の表に、DeviceNATTransOptionsFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isEnableTrafficWithoutTrans	boolean	true の場合は、アドレス変換なしでトラフィックがセキュリティ アプライアンスを通過できます。このオプションが false の場合、変換ルールに一致しないトラフィックはすべてドロップされます。 （注） このオプションは、PIX 7.x、FWSM 3.x、および ASA デバイスでのみ使用可能です。
isXlateByPass	boolean	true の場合は、imtranslated トラフィックの NAT セッションはディセーブルです。 （注） このオプションは、FWSM 3.2 以降でだけ使用可能です。

表 42：DeviceNATTransOptionsFirewallPolicy クラス定義

```
<xs:complexType name="DeviceNATTransOptionsFirewallPolicy">
    <xs:complexContent>
        <xs:extension base="BasePolicy">
            <xs:sequence>
                <xs:element name="isEnableTrafficWithoutTrans" type="xs:boolean"
minOccurs="0" maxOccurs="1"/>
                <xs:element name="isXlateByPass" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
```

図 44：DeviceNATTransOptionsFirewallPolicy XML スキーマ

3.1.5.12 InterfaceNATTransExemptionsFirewallPolicy

InterfaceNATTransExemptionsFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATTransExemptionsFirewallPolicy インスタンスは、アドレス変換からトラフィックを免除するルールを指定します。ルールは、リスト内の順序に従って評価されます。

このポリシーは、PIX、FWSM、および ASA 8.3 以前に適用されます。

次の表に、InterfaceNATTransExemptionsFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isRuleEnabled	boolean	true の場合ルールはイネーブルに、false の場合ルールはディセーブルになることを示します。
isExempt	boolean	true の場合、ルールは NAT から免除されるトラフィックを識別します。false の場合、ルールは、NAT から免除されないトラフィックを識別します。
realInterfaceGID	ObjectIdentifier	ルールが適用されるのデバイス インターフェイス。
original	複合タイプ	ルールが適用される発信元ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスやネットワーク ポリシー オブジェクトへの参照を含むことが可能です。
original.ipv4Data	文字列	リテラル IP アドレス。
original.interfaceRoleObjectGIDs	オブジェクト ID	インターフェイス ロールの ポリシー オブジェクト GID のリスト。
original.networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
outsideNAT	boolean	true はルール外部のキーワードが NAT ルールで定義されることを示します。
destinations	複合タイプ	ルールが適用される宛先ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスやネットワーク ポリシー オブジェクトへの参照を含むことが可能です。
destinations.ipv4Data	文字列	リテラル IP アドレス。
destinations.interfaceRoleObjectGIDs	オブジェクト ID	インターフェイス ロールの ポリシー オブジェクト GID のリスト。
destinations.networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
fwsmAdvancedOptions	複合タイプ	*FWSM* の場合のみ詳細オプション
fwsmAdvancedOptions.isTransDNSReplies	boolean	true にすると、外部クライアントが内部 DNS サーバを使用して内部ホストの名前を解決できるように、またその逆ができるように、セキュリティ アプライアンスは DNS 応答を書き換えます。

要素.サブ要素	タイプ	コメント
fwsmAdvancedOptions.maxTCPConnPerRule	UnsignedInt	許容される TCP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
fwsmAdvancedOptions.maxUDPConnPerRule	UnsignedInt	許容される UDP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
fwsmAdvancedOptions.maxEmbConnections	UnsignedInt	セキュリティ アプライアンスが初期接続を拒否し始めるまでに確立が許可される初期接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
fwsmAdvancedOptions.randomizeSeqNum	boolean	true にすると、セキュリティ アプライアンスによって TCP パケットのシーケンス番号がランダム化されます。

表 43：InterfaceNATTransExemptionsFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATTransExemptionsFirewallPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="isExempt" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="original" type="NetworkObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="destinations" type="NetworkObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="fwsmAdvancedOptions" type="FirewallNATAdvancedOptions"
minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
```

図 45：InterfaceNATTransExemptionsFirewallPolicy XML スキーマ

3.1.5.13 InterfaceNATDynamicRulesFirewallPolicy

InterfaceNATDynamicRulesFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATDynamicRulesFirewallPolicyspecifies のダイナミックな NAT および PAT ルールのインスタンス。ルールは、リスト内の順序に従って評価されます。

このポリシーは、PIX、FWSM、および ASA 8.3 以前に適用されます。

次の表に、InterfaceNATDynamicRulesFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isRuleEnabled	boolean	true の場合ルールはイネーブルに、false の場合ルールはディセーブルになることを示します。
realInterfaceGID	ObjectIdentifier	ルールを適用するデバイス インターフェイス ロール ポリシー オブジェクトにマップします。
poolId	Unsigned Int	変換に使用されるアドレス プールの ID 番号。これをアイデンティティ NAT ルールとして指定するには、値 0 を入力します。
original	複合タイプ	ルールが適用される発信元ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスやネットワーク ポリシー オブジェクトへの参照を含むことが可能です。
original.ipv4Data	文字列	リテラル IP アドレス。
original. networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
outsideNAT	ブール	true の場合、「外部」キーワードがこの NAT ルールに存在することを示します。
advancedOptions	複合タイプ	詳細オプションです。
advancedOptions.isTrans DNSReplies	boolean	true にすると、外部クライアントが内部 DNS サーバを使用して内部ホストの名前を解決できるように、またその逆ができるように、セキュリティ アプライアンスは DNS 応答を書き換えます。
advancedOptions.maxTCP ConnPerRule	UnsignedInt	許容される TCP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxUD PConnPerRule	UnsignedInt	許容される UDP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxEm bConnections	UnsignedInt	セキュリティ アプライアンスが初期接続を拒否し始めるまでに確立が許可される初期接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.randomi zeSeqNum	boolean	true にすると、セキュリティ アプライアンスによって TCP パケットのシーケンス番号がランダム化されます。

表 44：InterfaceNATDynamicRulesFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATDynamicRulesFirewallPolicy">
    <xs:complexContent>
        <xs:extension base="BasePolicy">
            <xs:sequence>
                <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="original" type="NetworkObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions"
minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 46：InterfaceNATDynamicRulesFirewallPolicy XML スキーマ

3.1.5.14 InterfaceNATPolicyDynamicRulesFirewallPolicy

InterfaceNATPolicyDynamicRulesFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATPolicyDynamicRulesFirewallPolicy インスタンスは送信元アドレスと宛先アドレスおよびサービスに基づいたダイナミック変換ルールを指定します。ルールは、リスト内の順序に従って評価されます。

このポリシーは、PIX、FWSM、および ASA 8.3 以前に適用されます。

次の表に、InterfaceNATPolicyDynamicRulesFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isRuleEnabled	boolean	true の場合ルールはイネーブルに、false の場合ルールはディセーブルになることを示します。
realInterfaceGID	ObjectIdentifier	ルールを適用するデバイス インターフェイス ロール ポリシー オブジェクトにマップします。
poolId	Unsigned Int	変換に使用されるアドレス プールの ID 番号。これをアイデンティティ NAT ルールとして指定するには、値 0 を入力します。
original	複合タイプ	ルールが適用される発信元ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスおよび/またはネットワーク/インターフェイス ロール ポリシー オブジェクトへの参照を含むことが可能です。
original.ipv4Data	文字列	リテラル IP アドレス。
original.interfaceRoleObjectGIDs	ObjectIdentifierList	インターフェイス ロール ポリシー オブジェクトのリスト。
original.networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
outsideNAT	ブール	true の場合、「外部」キーワードがこの NAT ルールに存在することを示します。
destinations	複合タイプ	ルールが適用される宛先ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスおよび/またはネットワーク/インターフェイス ロール ポリシー オブジェクトへの参照を含むことが可能です。
destinations.ipv4Data	文字列	リテラル IP アドレス。
destinations.interfaceRoleObjectGIDs	ObjectIdentifierList	インターフェイス ロール ポリシー オブジェクトのリスト。
destinations.networkObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
services	複合タイプ	ルールが適用されるサービスを含む複合タイプ。これは、プロトコル/送信元ポート/宛先ポート/サービス ポリシー オブジェクトへの参照、の形式で、サービス情報の組み合わせにすることができます。

要素.サブ要素	タイプ	コメント
services.serviceData	文字列	サービス固有の構文は次のとおりです。 **{tcp | udp | tcp&udp}/{source_port_number | port_list_object}/ {destination_port_number | port_list_object}** ポート パラメータが 1 つだけの場合は、宛先ポートを参照します（送信元ポートは任意）。たとえば、**tcp/4443** は tcp、送信元ポート any、宛先ポート 4443 を意味し、**tcp/4443/Default Range** は tcp、送信元ポート 4443、宛先ポート Default Range（通常、1 ～ 65535）を意味します。
services.serviceObjectGID	オブジェクト ID	サービス ポリシー オブジェクトを参照する ObjectIdentifier ID。
advancedOptions	複合タイプ	詳細オプションです。
advancedOptions.isTransDNSReplies	boolean	true にすると、外部クライアントが内部 DNS サーバを使用して内部ホストの名前を解決できるように、またその逆ができるように、セキュリティ アプライアンスは DNS 応答を書き換えます。
advancedOptions.maxTCPConnPerRule	UnsignedInt	許容される TCP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxUDPConnPerRule	UnsignedInt	許容される UDP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxEmbConnections	UnsignedInt	セキュリティ アプライアンスが初期接続を拒否し始めるまでに確立が許可される初期接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.randomizeSeqNum	boolean	true にすると、セキュリティ アプライアンスによって TCP パケットのシーケンス番号がランダム化されます。

表 45：InterfaceNATPolicyDynamicRulesFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATPolicyDynamicRulesFirewallPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="original" type="NetworkInterfaceObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="destinations" type="NetworkInterfaceObjectsRefs"
minOccurs="1" maxOccurs="1"/>
                    <xs:element name="services" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="serviceData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                                <xs:element name="serviceObjectGID" type="ObjectIdentifier"
minOccurs="0" maxOccurs="unbounded"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions"
minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 47：InterfaceNATPolicyDynamicRulesFirewallPolicy XML スキーマ

3.1.5.15 InterfaceNATStaticRulesFirewallPolicy

InterfaceNATStaticRulesFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATStaticRulesFirewallPolicy インスタンスは、セキュリティ アプライアンスに対してスタティック変換ルールを指定します。ルールは、リスト内の順序に従って評価されます。

このポリシーは、PIX、FWSM、および ASA 8.3 以前に適用されます。

次の表に、InterfaceNATStaticRulesFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isRuleEnabled	boolean	true の場合ルールはイネーブルに、false の場合ルールはディセーブルになることを示します。
translationType	Enumeration	ルールの変換タイプ（「NAT」または「PAT」）。
realInterfaceGID	ObjectIdentifier	ルールを適用するデバイス インターフェイス ロール ポリシー オブジェクトにマップします。
mappedInterfaceGID	オブジェクト ID	変換後のアドレスが使用されるインターフェイス ロール ポリシー オブジェクト インターフェイスへのマッピング。
original	複合タイプ	ルールが適用される発信元ホストおよびネットワーク オブジェクトの IP アドレスを含む複合タイプ。複数のリテラル IP アドレスやネットワーク ポリシー オブジェクトへの参照を含むことが可能です。
original.ipv4Data	文字列	リテラル IP アドレス。
original.networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated	複合タイプ	変換後のアドレスを含む複合タイプ要素。
translated.ipv4Data	文字列	リテラル IP アドレス。
translated.networkObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated.interfaceKeyword	文字列	「interface」の値は、このキーワードが NAT ルールに存在することを指定します。
policyNAT	複合タイプ	ポリシー NAT がこのルールでイネーブルになっている場合に限り、ポリシー NAT の詳細を含む複合タイプ。
policyNAT.destAddress	複合タイプ	宛先アドレスを含む複合タイプ。
policyNAT.destAddress.ipv4Data	文字列	リテラル IP アドレス。
policyNAT.destAddress.networkObjectGIDs	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。

要素.サブ要素	タイプ	コメント
policyNAT.services	複合タイプ	ルールが適用されるサービスを指定する複合タイプ。これは、プロトコル/送信元ポート/宛先ポート/サービス ポリシー オブジェクトへの参照、の形式で、サービス情報の組み合わせにすることができます。
policyNAT.services.serviceData	文字列	サービス固有の構文は次のとおりです。 **{tcp | udp | tcp&udp}/{source_port_number | port_list_object}/ {destination_port_number | port_list_object}** ポート パラメータが 1 つだけの場合は、宛先ポートを参照します（送信元ポートは任意）。たとえば、**tcp/4443** は tcp、送信元ポート any、宛先ポート 4443 を意味し、**tcp/4443/Default Range** は tcp、送信元ポート 4443、宛先ポート Default Range（通常、1 ～ 65535）を意味します。
policyNAT.services.serviceObjectGID	オブジェクト ID	サービス ポリシー オブジェクトを参照する ObjectIdentifier ID。
protocol	Enumeration	ルールの適用対象となるプロトコル「UDP」、「TCP」、「IP」。
originalPort	Unsigned Int	[Translation Type] で [PAT] を選択した場合は、これは、変換されるポート番号を指定します。
translatedPort	Unsigned Int	[Translation Type] で [PAT] を選択した場合は、元のポート番号が変換されるポート番号を指定します。
advancedOptions	複合タイプ	詳細オプションです。
advancedOptions.isTransDNSReplies	boolean	true にすると、外部クライアントが内部 DNS サーバを使用して内部ホストの名前を解決できるように、またその逆ができるように、セキュリティ アプライアンスは DNS 応答を書き換えます。
advancedOptions.maxTCPConnPerRule	UnsignedInt	許容される TCP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxUDPConnPerRule	UnsignedInt	許容される UDP 接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.maxEmbConnections	UnsignedInt	セキュリティ アプライアンスが初期接続を拒否し始めるまでに確立が許可される初期接続の最大数。有効な値は 0 ～ 65,535 です。この値を 0 に設定すると、接続数は無制限になります。
advancedOptions.randomizeSeqNum	boolean	true にすると、セキュリティ アプライアンスによって TCP パケットのシーケンス番号がランダム化されます。

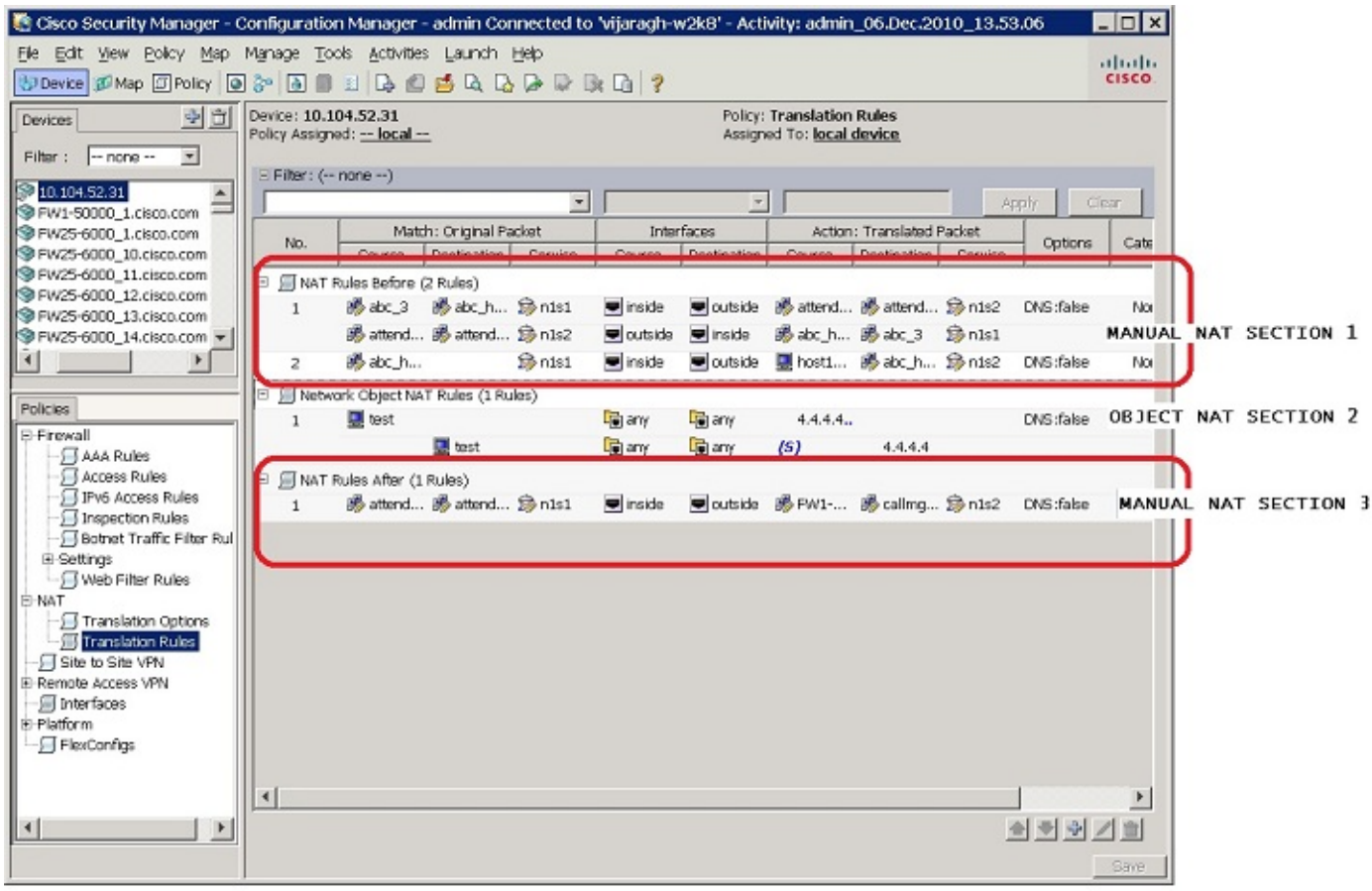
表 46：InterfaceNATStaticRulesFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATStaticRulesFirewallPolicy">
    <xs:complexContent>
        <xs:extension base="BasePolicy">
            <xs:sequence>
                <xs:element name="isRuleEnabled" type=xs:boolean" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="translationType" minOccurs="1" maxOccurs="1">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:enumeration value="NAT"/>
                            <xs:enumeration value="PAT"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="mappedInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="original" type="NetworkOrIPRef" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="translated" type="NetworkObjectRefs" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="policyNAT" minOccurs="0" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="destAddress" type="NetworkObjectsRefs"
minOccurs="1" maxOccurs="1"/>
                            <xs:element name="services" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="serviceData" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="serviceObjectGID"
type="ObjectIdentifier" minOccurs="0" maxOccurs="unbounded"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="protocol" type="IPTransportProtocol" minOccurs="1"
maxOccurs="1"/>
                <xs:element name="originalPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="translatedPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions"
minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
```

図 48：InterfaceNATStaticRulesFirewallPolicy XML スキーマ

3.1.5.16 InterfaceNATManualFirewallPolicy

このポリシーは、デバイス **ASA バージョン 8.3 以降**にのみ適用されます。ASA 8.3 以降は、完全な NAT の表示は、2 種類の主要ポリシー タイプの組み合わせです。以下に示すように、これは手動 NAT 事前ルールと事後ルール（セクション 1 と 3）およびオブジェクト NAT ルール（セクション 2）から構成されます。



この項では、手動 NAT（InterfaceNATManualFirewallPolicy）を説明します。また次のセクションでは、オブジェクト NAT（InterfaceNATObjectFirewallPolicy）を説明します。

InterfaceNATManualFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATManualFirewallPolicy インスタンスはデバイス上に「手動で」定義された NAT ルールを指定します。基本ポリシーから継承した基本 **orderId** 属性は、ポリシー内でのルールの順序を指定します。ポリシー データの **section** 要素は、これが NAT の事前ルールか事後ルールかを指定します。

次の表に、InterfaceNATManualFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
isRuleEnabled	boolean	true の場合ルールはイネーブルに、false の場合ルールはディセーブルになることを示します。
section	Enumeration	ルール セクションを指定します。有効な値は「1」、

要素.サブ要素	タイプ	コメント
		「2」、「3」です。以下は解釈です。 「1」→ 事前 NAT ルールつまりルール前 NAT を指します。 「2」→ オブジェクト NAT ルールを指します。 「3」→ 事後 NAT ルールつまりルール後 NAT を指します。 このポリシー タイプは「1」と「3」のみ可能です。BasePolicy の orderId 要素は、このセクション内のルールの 「順序」を指定します。
realInterface	複合タイプ	インターフェイス ロール ポリシー オブジェクトまたはインターフェイス名への参照
realInterface.realInterfaceGID	オブジェクト ID	インターフェイス ロール ポリシー オブジェクトへの参照
realInterface.realInterfaceName	文字列	実際のインターフェイス名
mappedInterface	複合タイプ	インターフェイス ロール ポリシー オブジェクトまたは名前への参照 (注：実際のインターフェイスおよびマッピング インターフェイスがいずれも指定されていない場合、デフォルトは「任意」とみなされます。)
mappedInterface.mappedInterfaceGID	オブジェクト ID	インターフェイス ロール ポリシー オブジェクトへの参照
mappedInterface.mappedInterfaceName	文字列	マッピングされたインターフェイス名。
source	複合タイプ	オリジナルと変換された送信元を含む複合タイプ
source.natType	Enumeration	変換ルールのタイプを「スタティック」または「ダイナミック」と指定します。
source.originalObjectGID	オブジェクト ID	NAT ルールで変換される送信元アドレスを指定します。アドレス範囲またはネットワークの場合は、範囲またはネットワーク内のすべてのアドレスが変換されます。この要素には、ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID が含まれます。
source.translated	複合タイプ	ネットワーク アドレスのプールまたは変換された送信元を表すインターフェイス オブジェクトへの参照の**いずれか**を含む複合タイプ。
source.translated.ObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクト GID への参照。**リテラル IPv4 アドレスは、手動 NAT の設定には使用できません。**

要素.サブ要素	タイプ	コメント
source.translated.interfaceKeyword	文字列	「interface」の値は、インターフェイスのキーワードがNAT ルールに適用済みであることを指定します。
source.translated.patPool	複合タイプ	PAT のオプションを含む複合タイプ
source.translated.patPool.patAddressPool	複合タイプ	PAT アドレスのプールを含む複合タイプ
source.translated.patPool.patAddressPool.patPoolAddressGID	オブジェクト ID	ネットワーク ポリシー オブジェクトへの参照
source.translated.patPool.patAddressPool.interfaceKeyword	文字列	固定文字列「interface」がこの要素の値として使用される場合は、Fallthrough Interface を提供するインターフェイスは宛先インターフェイスと同じになります。
source.translated.patPool.isPatAllocatedInRoundRobin	boolean	True の場合、使用可能な IP アドレスとポート番号を使用した「ラウンド ロビン」サイクルを意味します。このメソッドでは、プールでそれぞれ連続するアドレスを使用して、アドレス/ポートの組み合わせを割り当てます。その後、最初のアドレスを異なるポートで再度使用し、次に 2 番目のアドレスを使用し、以後、同様に動作します。
destination	複合タイプ	宛先アドレスのスタティック変換を含む複合タイプ
destination.natType	Enumeration	変換ルールのタイプを「スタティック」または「ダイナミック」と指定します。
destination.originalObject	複合タイプ	元のオブジェクトへの参照
destination.originalObject.networkObjectGIDs	Object IdentifierList	ネットワーク ポリシー オブジェクト GID のリストへの参照。
destination.originalObject.ipV4Data	文字列	宛先 IPv4 アドレス。
destination.translatedObjectGID	ObjectIdentifier	変換されたネットワーク GID オブジェクトへの参照。
service	複合タイプ	ポート アドレス変換を指定する複合タイプ。
service.originalObjectGID	オブジェクト ID	変換されるサービスを定義するサービス ポリシー オブジェクトへの参照。
service.transObjectGID	オブジェクト ID	変換に使用されるサービスを定義するサービス ポリシー オブジェクトへの参照。
isTransDNSReplies	boolean	true の場合、このルールに一致する DNS 応答に埋め込まれたアドレスが書き換えられます。
Direction	Enumeration	単一方向だけのスタティック NAT ルール（「単方向」）、または両方向（順方向と逆方向）に 1 つずつのデュアル ルール（「双方向」）を指定します。

要素.サブ要素	タイプ	コメント
isNoProxyARP	boolean	True の場合、これは指定した宛先インターフェイスでプロキシ ARP をディセーブルにします。 デフォルトでは、すべての NAT ルールは、出力インターフェイスでプロキシ ARP が含まれます。NAT 免除ルールは出力インターフェイスを検出するときにルートの概要に依存する入力トラフィックと出力トラフィックの両方に対して NAT をバイパスするために使用されます。したがって、プロキシ ARP は、NAT 免除ルールを無効にする必要があります。（NAT 免除ルールが常に優先し、[Translation Rules] テーブルの他のすべての NAT ルールの上に表示されます。）
isRouteLookUp	boolean	このオプションが true の場合、出力インターフェイスは、指定した宛先インターフェイスを使用する代わりにルート ルックアップを使用して決定します。これは、NAT 免除ルールの場合は true にする必要があります。

表 47：InterfaceNATManualFirewallPolicy クラス定義

```
<xs:complexType name="PatOptions">
        <xs:sequence>
            <xs:element name="patAddressPool" minOccurs="0" maxOccurs="1">
                <xs:complexType>
                    <xs:choice>
                        <xs:element name="patPoolAddressGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="interfaceKeyword" type="xs:string" fixed="interface" minOccurs="0"
maxOccurs="1"/>
                    </xs:choice>
                </xs:complexType>
            </xs:element>
            <xs:element name="isPatAllocatedInRoundRobin" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="InterfaceNATManualFirewallPolicy">
        <xs:complexContent>
            <xs:extension base="BasePolicy">
                <xs:sequence>
                    <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="section" minOccurs="1" maxOccurs="1">
                        <xs:simpleType>
                            <xs:restriction base="xs:unsignedInt">
                                <xs:enumeration value="1"/>
                                <xs:enumeration value="2"/>
                                <xs:enumeration value="3"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="realInterface" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="realInterfaceGID" type="ObjectIdentifier"
minOccurs="0" maxOccurs="1"/>
                                <xs:element name="realInterfaceName" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="mappedInterface" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="mappedInterfaceGID" type="ObjectIdentifier"
minOccurs="0" maxOccurs="1"/>
                                <xs:element name="mappedInterfaceName" type="xs:string"
minOccurs="0" maxOccurs="1"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="source" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="natType" type="NATType" minOccurs="1"
maxOccurs="1"/>
                                <xs:element name="originalObjectGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
                                <xs:element name="translated" minOccurs="0" maxOccurs="1">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="objectGID"
type="NetworkObjectRefs" minOccurs="0" maxOccurs="1"/>
```

図 49：InterfaceNATManualFirewallPolicy

3.1.5.17 InterfaceNAT64ManualFirewallPolicy

InterfaceNAT64ManualFirewallPolicy は Unified（IPv6/IPv4）手動 NAT ルールを表します。ASA 9.0 から Unified NAT ルールはサポートされます。以下はサポートされるネットワーク変換です。

- IPv4 -> IPv6
- IPv6 -> IPv4
- IPv6 -> IPv6
- IPv4 -> IPv4

InterfaceNAT64ManualFirewallPolicy は基本 InterfaceNATManualFirewallPolicy クラスから拡張され、InterfaceNAT64ManualFirewallPolicy インスタンスは 1 つの Unified NAT ルールを示します。

このポリシーは、API バージョン 1.1 以降で使用できます。

要素	タイプ	コメント
isInterfaceIPv6	boolean	インターフェイスの IPv6 アドレスを使用します。
isNetToNet	boolean	IPv4 サーバへの単一 IPv6 アドレスの 1 対 1 のマッピングのオプション

表 48：InterfaceNAT64ManualFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNAT64ManualFirewallPolicy">
   <xs:complexContent>
      <xs:extension base=" InterfaceNATManualFirewallPolicy ">
         <xs:sequence minOccurs="1" maxOccurs="1">
            <xs:element name="isInterfaceIPv6" type="boolean" minOccurs="0" maxOccurs="1"/>
            <xs:element name="isNetToNet" type="boolean" minOccurs="0" maxOccurs="1"/>
          </xs:sequence>
      </xs:extension>
</xs:complexContent>
</xs:complexType>
```

図 50：InterfaceNAT64ManualFirewallPolicy XML スキーマ

3.1.5.18 InterfaceNATObjectFirewallPolicy

InterfaceNATObjectFirewallPolicy は基本 BasePolicy クラスから拡張され、属性をすべて継承します。InterfaceNATObjectFirewallPolicy のインスタンスは、デバイスでオブジェクトの NAT ルールを指定します。基本ポリシーから継承した基本**注文 ID** 属性は、ポリシー内でこれらのルールの順序を指定します。

このポリシーは、デバイス **ASA バージョン 8.3 以降**にのみ適用されます。

次の表に、InterfaceNATObjectFirewallPolicy の内容を定義します。

要素.サブ要素	タイプ	コメント
section	Enumeration	ルール セクションを指定します。有効な値は「1」、「2」、「3」です。以下は解釈です。 「1」→ 事前 NAT ルールつまりルール前 NAT を指します。 「2」→ オブジェクト NAT ルールを指します。 「3」→ 事後 NAT ルールつまりルール後 NAT を指します。 このポリシー タイプには、「2」のみが適用されます。BasePolicy の orderId 要素は、このセクション内のルールの「順序」を指定します。
realInterface	文字列	インターフェイス文字列
mappedInterface	文字列	インターフェイス文字列
natType	Enumeration	変換ルールのタイプを「スタティック」または「ダイナミック」と指定します。
originalObjectGID	文字列	NAT ルールで変換される送信元アドレス。
translated	複合タイプ	変換がアドレスまたはインターフェイスの**いずれに**基づいているかを指定する複合タイプ。
translated.objectGID	複合タイプ	アドレス定義を含む複合タイプ
translated.objectGID.ipv4Data	文字列	リテラル IP アドレス。
translated.objectGID.networkObjectGID	オブジェクト ID	ネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated.objectGID.interfaceKeyword	文字列	「interface」の値は、インターフェイス キーワードがこの NAT ルールに定義されていることを示します。
translated.patPool	複合タイプ	ASA バージョン 8.4.2 以降では、ダイナミックな NAT の個別の PAT プールと PAT ルールを定義できます。PAT プール アドレスは、この [PAT Pool Address Translation] フィールドを使用して指定されます。これには、PAT プール オプションが含まれます。
translated.patPool.patAddressPool	複合タイプ	アドレス情報が記載されています。

要素.サブ要素	タイプ	コメント
translated.patPool.patAddressPool.patPoolAddressGID	オブジェクト ID	PAT プールのネットワーク ポリシー オブジェクトを参照する ObjectIdentifier ID。
translated.patPool.patAddressPool.interfaceKeyword	文字列	「interface」の値は、インターフェイス キーワードがこの NAT ルールに定義されていることを示します。
translated.patPool . isPatAllocatedInRoundRobin	boolean	これが true の場合、ASA デバイスは PAT プールにラウンド ロビン割り当てを使用します。
isTransDNSReplies	boolean	true の場合、このルールに一致する DNS 応答に埋め込まれたアドレスが書き換えられます。
isNoProxyARP	boolean	true の場合、宛先インターフェイスで ARP はプロキシしないでください。
isRouteLookUp	boolean	true の場合、宛先インターフェイスのルート検索を行います。
service	複合タイプ	設定のスタティック ポート アドレス変換を定義する複合タイプ。スタティック ルールのみに対するアプリケーション
service.protocol	Enumeration	ルールの適用対象となるプロトコル「UDP」、「TCP」、または「IP」。
service.originalPort	Unsigned Int	トラフィックがデバイスに着信するポート。
service.transPort	Unsigned Int	元のポート番号を交換するポート番号。

表 49：InterfaceNATObjectFirewallPolicy クラス定義

```
<xs:complexType name="InterfaceNATObjectFirewallPolicy">
	<xs:complexContent>
		<xs:extension base="BasePolicy">
			<xs:sequence>
				<xs:element name="section" fixed="2" minOccurs="1" maxOccurs="1">
					<xs:simpleType>
						<xs:restriction base="xs:unsignedInt">
							<xs:enumeration value="1"/>
							<xs:enumeration value="2"/>
							<xs:enumeration value="3"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="realInterface" type="xs:string" minOccurs="1" maxOccurs="1"/>
				<xs:element name="mappedInterface" type="xs:string" minOccurs="1" maxOccurs="1"/>
				<xs:element name="natType" type="NATType" minOccurs="1" maxOccurs="1"/>
				<xs:element name="originalObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
				 <xs:element name="translated" minOccurs="0" maxOccurs="1">
					<xs:complexType>
						<xs:sequence>
							<xs:element name="objectGID" type="NetworkObjectRefs" minOccurs="0"
maxOccurs="1"/>
							<xs:element name="patPool" type="PatOptions" minOccurs="0"
maxOccurs="1"/>
						</xs:sequence>
					</xs:complexType>
				</xs:element>
				<xs:element name="isTransDNSReplies" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
				<xs:element name="isNoProxyARP" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
				<xs:element name="isRouteLookUp" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
				<xs:element name="service" minOccurs="0" maxOccurs="1">
					<xs:complexType>
						<xs:sequence>
							<xs:element name="protocol" type="IPTransportProtocol" minOccurs="1"
maxOccurs="1"/>
							<xs:element name="originalPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
							<xs:element name="transPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
						</xs:sequence>
					</xs:complexType>
				</xs:element>
			</xs:sequence>
```

図 51：InterfaceNATObjectFirewallPolicy XML スキーマ

3.1.5.19 InterfaceNAT64ObjectFirewallPolicy

InterfaceNAT64ObjectFirewallPolicy は Unified（IPv6/IPv4）オブジェクト NAT ルールを表します。ASA 9.0 から Unified NAT ルールはサポートされます。以下はサポートされるネットワーク変換です。

- IPv4 -> IPv6
- IPv6 -> IPv4
- IPv6 -> IPv6
- IPv4 -> IPv4

InterfaceNAT64ObjectFirewallPolicy は基本 InterfaceNATObjectFirewallPolicy クラスから拡張されます。InterfaceNAT64ObjectFirewallPolicy インスタンスが単一の Unified オブジェクトの NAT ルールを表示します。

このポリシーは、API バージョン 1.1 以降で使用できます。

要素	タイプ	コメント
isInterfaceIPv6	boolean	インターフェイスの IPv6 アドレスを使用します。
isNetToNet	boolean	IPv4 サーバへの単一 IPv6 アドレスの 1 対 1 のマッピングのオプション

表 50：InterfaceNAT64ObjectFirewallPolicy クラス定義

```
<xs:complexType name=" InterfaceNAT64ObjectFirewallPolicy ">
   <xs:complexContent>
      <xs:extension base=" InterfaceNATObjectFirewallPolicy ">
         <xs:sequence minOccurs="1" maxOccurs="1">
            <xs:element name="isInterfaceIPv6" type="boolean" minOccurs="0" maxOccurs="1"/>
            <xs:element name="isNetToNet" type="boolean" minOccurs="0" maxOccurs="1"/>
          </xs:sequence>
      </xs:extension>
</xs:complexContent>
</xs:complexType>
```

図 52：InterfaceNAT64ObjectFirewallPolicy XML スキーマ

3.2 メソッド

設定サービスは次のメソッドを定義します。

1. GetServiceInfo
    a. サービス名、バージョン、日付などを含むサービス固有の情報を返します。
    b. このメソッドは、認証が有効なセッション Cookie を返した後、他のメソッドよりも前に呼び出す必要があり、クライアントが現在実行中のサービス バージョンと互換性があることを確認する必要があります。
2. GetGroupList
    a. システムで定義されたデバイス グループのリストを返します。
3. GetDeviceListByCapability
    a. システムの機能またはワイルドカードの 1 つ以上に一致するすべてのデバイスのリストを返します。
4. GetDeviceListByGroup
    a. システムの指定されたグループ内のすべてのデバイスのリストを返します。
5. GetDeviceConfigByGID
    a. メソッド引数で指定したオブジェクト ID で識別されるデバイス設定を返します。
6. GetDeviceConfigByName
    a. メソッド引数で指定したデバイス名で識別されるデバイス設定を返します。
7. GetPolicyConfigByName
    a. 名前が割り当てられた複数のデバイスに割り当てられたポリシー設定を返します。
8. GetPolicyConfigByDeviceGID
    a. メソッド引数に指定された特定のタイプに一致するデバイスのオブジェクト ID に属するポリシー設定を返します。
9. GetPolicyListByDeviceGID
    a. メソッド引数に指定されたデバイスのオブジェクト ID に属するポリシー名とタイプのリストを返します。
10. GetSharedPolicyListByType
    a. 特定のポリシー タイプの CSM で定義されたすべての共有ポリシーのリストを戻します。

3.2.1 メソッド GetServiceInfo

GetServiceInfo メソッドは、サービスに関連するサービスの説明、バージョン情報、および該当する属性を返します。

3.2.1.1 要求

メソッド GetServiceInfo 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/GetServiceInfo

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
   <getServiceInfoRequest>
          <protVersion>1.0</protVersion>
          <reqId>123</reqId>
   </getServiceInfoRequest>
```

図 53：メソッド GetServiceInfo 要求の例

表 51：メソッド GetServiceInfo 要求の URL 引数の説明

HTTP/XML の内容	定義
getServiceInfoRequest	要求の引数を含むオブジェクト。
HTTP メソッド	PUT
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="getServiceInfoRequest" type="GetServiceInfoRequest"/>
<xs:complexType name="GetServiceInfoRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp"/>
      </xs:complexContent>
</xs:complexType>
```

図 54：GetServiceRequest XML スキーマ

3.2.1.2 応答

GetServiceInfo 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version=”1.0” encoding=”UTF-8”?>
<ns1:getServiceInfoResponse xmlns:ns1="csm">
   <protVersion>1.0</protVersion>
   <reqId>123</reqId>
   <serviceVersion>1.0</serviceVersion>
   <serviceName>CSM Configuration Service</serviceName>
   <serviceDesc>A configuration service that enables network services configuration to be
   retrieved</serviceDesc>
</ns1:getServiceInfoResponse>
```

図 55：GetServiceInfo 応答の例

表 52：GetServiceInfo 応答の要素と属性の説明

XML 要素と属性	定義
getServiceInfoResponse	コンフィギュレーション サービスのサービス情報を返します
serviceVersion	コンフィギュレーション サービス実行のサービス バージョン
serviceName	サービス名
serviceDescr	サービスの機能の詳細を提供するサービスの説明

```
    <xs:element name="getServiceInfoResponse" type="GetServiceInfoResponse"/>
    <xs:complexType name="GetServiceInfoResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence>
                    <xs:element name="serviceVersion" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="serviceName" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="serviceDesc" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 56：GetServiceInfoResponse XML スキーマ

3.2.2 メソッド GetGroupList

ワイルドカードの引数を選択した場合、GetGroupList メソッドは、特定のタイプのポリシーに一致するデバイスまたはすべてのデバイスを返します。

3.2.2.1 要求

メソッド GetGroupList 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getGroupList

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<groupListRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <includeEmptyGroups>false</includeEmptyGroups>
</groupListRequest>
```

図 57：メソッド GetGroupList 要求の例

表 53：メソッド GetGroupList 要求の URL 引数の説明

HTTP/XML の内容	定義
GroupListRequest	要求の引数を含むグループ リストの要求
要素：includeEmptyGroups	内部にデバイスが存在しないグループを応答に含むべきかどうかを指定する要素。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="groupListRequest" type="GroupListRequest"/>
<xs:complexType name="GroupListRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp">
                  <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="includeEmptyGroups" type="xs:boolean"/>
                  </xs:sequence>
            </xs:extension>
      </xs:complexContent>
</xs:complexType>
```

図 58：GroupListRequest XML スキーマ

3.2.2.2 応答

GroupListRequest 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。GetGroupListResponse では、デバイスの完全な設定は*実行されません*。これは GetDeviceConfigByName または GetDeviceConfigByGID メソッドが呼び出されたときにのみ設定されます。

```
<?xml version=”1.0” encoding=”UTF-8”?>
<groupListResponse>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <deviceGroup>
          <gid>00000000-0000-0000-0000-000000000001</gid>
          <name>San Jose</name>
          <lastUpdateTime>2011-05-22T07:07:29.129Z</lastUpdateTime>
          <path>/VmsVirtualRoot/San Jose</path>
          <deviceGroup>
            <gid>00000000-0000-0000-0000-000000000002</gid>
            <name>Building 13</name>
            <lastUpdateTime>2011-05-22T07:07:29.129Z</lastUpdateTime>
            <path>/VmsVirtualRoot/San Jose/Building 13</path>
              <device>
                 <gid>00000000-0000-0000-0000-214748364932</gid>
                 <name>10.77.208.138</name>
                 <lastUpdateTime>2011-05-26T00:11:53Z</lastUpdateTime>
                 <osType>asa</osType>
                 <osVersion>8.4(1)</osVersion>
                 <imageName>disk0:/asa831-k8.bin</imageName>
                 <sysObjectID>1.3.6.1.4.1.9.1.670</sysObjectID>
                 <fullConfig/>
                 <mgmtInterface>
                    <type>Management</type>
                    <identifier>mgmt</identifier>
                    <ipInterface>
                       <ipAddress>10.77.208.138/255.255.255.0</ipAddress>
                    </ipInterface>
                 </mgmtInterface>
                 <interfaceList>
                    <interface>
                       <type>GigabitEthernet</type>
                       <identifier>outside</identifier>
                       <ipInterface>
                            <ipAddress>20.10.30.42/255.255.255.0</ipAddress>
                       </ipInterface>
                    </interface>
                    <interface>
                       <type>FastEthernet</type>
                       <identifier>FastEthernet5/1</identifier>
                       <ipInterface>
                            <ipAddress>20.10.30.45/255.255.255.0</ipAddress>
                       </ipInterface>
                    </interface>
                 </interfaceList>
              <configState>committed</configState>
            </device>
          </deviceGroup>
        </deviceGroup>
```

図 59：GetGroupList 応答の例

表 54：GetGroupList 応答の要素と属性の説明

XML 要素と属性	定義
groupListResponse	0 個以上のデバイス グループのリストを返します。
要素のリスト：DeviceGroup	グループ要素のリスト

```
<xs:element name="groupListResponse" type="GroupListResponse"/>
<xs:complexType name="GroupListResponse">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="deviceGroup" type="DeviceGroup"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 60：GetGroupList 応答 XML スキーマ

メソッド特有のエラー：

コード	説明
2000	このエラーは、API がログイン ユーザのライセンス済みデバイスのリストを取得できない場合に返されます。
2001	このエラーは、API が以降の処理のために CSM サーバから使用可能なグループを受け取れない場合に返されます。
2002	このエラーは、API が CSM サーバからデバイスを取得できない場合に返されます。
2003	このエラーは、API がデバイスの設定状態を検出できない場合に返されます。
2004	このエラーは、API がデバイスに関するインターフェイスの詳細を取得できない場合に返されます。
2005、2006	これらのエラーは、デバイス固有のデータの処理時に API が内部エラーを検出した場合に返されます。

表 55：GetGroupList メソッドのエラー コード

3.2.3 メソッド GetDeviceListByCapability

ワイルドカードの引数を選択した場合、GetDeviceListByCapability メソッドは、1 つ以上のカテゴリと一致するデバイスまたはすべてのデバイスのリストを返します。

3.2.3.1 要求

メソッド GetDeviceListByType 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getDeviceListByType

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<deviceListByCapabilityRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <deviceCapability>firewall</deviceCapability>
</deviceListByCapabilityRequest>
```

図 61：メソッド GetDeviceListByCapability 要求の例

表 56：メソッド GetDeviceListByCapability 要求の URL 引数の説明

HTTP/XML の内容	定義
getDeviceListByCapabilityRequest	要求の引数を含むデバイス リストの要求
要素：deviceCapability	要求されているデバイスの機能を識別する要素の 1 つ以上のインスタンス。許容値は次のとおりです: • **firewall**：すべての ASA、PIX、および FWSM デバイスを返します。 • **ids**：すべての IPS デバイスを返します。 • **router**：ルータに戻ります。 • **switch**：スイッチを返します。 ワイルドカードは <csm:deviceCapability>*</csm:deviceCapability> と指定することができます。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="deviceListByCapabilityRequest" type="DeviceListByCapabilityRequest"/>
      <xs:complexType name="DeviceListByCapabilityRequest">
            <xs:complexContent>
                  <xs:extension base="BaseReqResp">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                              <xs:element name="deviceCapability" type="DeviceCapability" minOccurs="1"
maxOccurs="unbounded"/>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
```

図 62：DeviceListByCapabilityRequest XML スキーマ

3.2.3.2 応答

GetDeviceListByCapability 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<deviceListResponse>
         <protVersion>1.0</protVersion>
         <reqId>123</reqId>
         <deviceId>
           <gid>00000000-0000-0000-0000-12211312321</gid>
           <deviceCapability>firewall</deviceCapability>
           <ipv4Address>12.1.1.1</ipv4Address>
         </deviceId>
</deviceListResponse>
```

図 63：GetDeviceListByCapability 応答の例

表 57：GetDeviceListByCapability 応答の要素と属性の説明

XML 要素と属性	定義
deviceListResponse	このメソッドで渡されるフィルタ パラメータに一致する 0 個以上のデバイスのリストを返します。
要素のリスト：DeviceId	デバイス ID の要素のリスト
属性：Gid	デバイスの GID 属性
要素：deviceCapability	デバイスの 1 つ以上の機能（必須）
要素：deviceName	デバイスの名前（必須）
要素：ipv4Address	（任意）デバイスの IPv4 アドレス

```
<xs:element name="deviceListResponse" type="DeviceListResponse"/>
      <xs:complexType name="DeviceListResponse">
            <xs:complexContent>
                  <xs:extension base="BaseReqResp">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                              <xs:element name="deviceId" minOccurs="0" maxOccurs="unbounded">
                                    <xs:complexType>
                                          <xs:sequence minOccurs="1" maxOccurs="1">
                                                <xs:element name="deviceCapability" type="DeviceCapability"
minOccurs="1" maxOccurs="1"/>
                                                <xs:element name="deviceName" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                                <xs:element name="ipv4Address" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                                <xs:element name="gid" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                          </xs:sequence>
                                    </xs:complexType>
                              </xs:element>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
```

図 64：DeviceListResponse XML スキーマ

GetGroupList メソッドに記載されたエラー コードがこのメソッドにも適用されます。

3.2.4 メソッド GetDeviceListByGroup

ワイルドカードの引数が選択されている場合、GetDeviceListByGroup メソッドは、特定のグループに含まれているデバイスまたはすべてのデバイスのリストを返します。グループ名のパスはリクエストボディに定めるパス全体の項目の組み合わせで作成されます。デバイス リストはパス項目の組み合わせからなります。たとえば、パス項目が *San Jose* 単独として提供されていて CSM サーバが San Jose の下にサブグループを持つ場合、API は *San Jose* グループおよび San Jose の下にあるその他のサブグループに一致するすべてのデバイスを返します。

たとえばパス項目が *San Jose* と *Building14* の場合、グループ パス *「/San Jose/Building 14」*とそのサブグループ（該当する場合）に一致するデバイスが返されます。

3.2.4.1 要求

メソッド GetDeviceListByGroup 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getDeviceListByGroup

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<deviceListByGroupRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <deviceGroupPath>
          <pathItem>building10</pathItem>
          <pathItem>4th floor</pathItem>
        </deviceGroupPath>
</deviceListByGroupRequest>
```

図 65：メソッド GetDeviceListByGroup 要求の例

表 58：メソッド GetDeviceListByGroup 要求の URL 引数の説明

HTTP/XML の内容	定義
getDeviceListByGroupRequest	グループに含まれるデバイスのリストを返すデバイス リスト要求。

HTTP/XML の内容	定義
要素：deviceGroupPath	要求されているデバイス グループを識別する要素。1 つ以上の *pathItem* エントリを含みます。結合されたすべての pathItems は（スラッシュ「/」で区切る）、1 つのグループ名として処理されます。GetGroupList メソッドの応答からのパス情報がここで使用できます。GetGroupList の各グループに完全なパス名を提供するパス属性があります。パス内の各要素（「/」よって区切る）はパス項目として符号化されます。たとえばグループのデバイス詳細が「/VmsVirtualRoot/San Jose/Building 13」などのパス（getGroupList API 応答から取得）を含むと、パス項目は「VmsVirtualRoot」、「San Jose」、「Building 13」となります。「VmsVirtualRoot」は、全グループの仮想「ルート ノード」です。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="deviceListByGroupRequest" type="DeviceListByGroupRequest"/>
<xs:complexType name="DeviceListByGroupRequest">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="deviceGroupPath" type="DeviceGroupPath"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 66：DeviceListByGroupRequest XML スキーマ

3.2.4.2 応答

GetDeviceListByGroup 応答とエラー コードは、メソッド GetDeviceListByCapability 応答と同じです。

3.2.5 メソッド GetDeviceConfigByGID

GetDeviceConfigByGID メソッドは、メソッドに渡されるデバイス ID に基づいて、特定のデバイス オブジェクトと関連する設定を返します。デバイス設定を要求しているユーザは、設定を要求しているデバイスの「view_device」および「view_cli」権限の両方がなければなりません。このメソッドが返す「fullConfig」要素のデータは、デバイスで実行される一般的なコマンド「show running-config」からのデータに等しくなります。このすべての設定は、CSM の Configuration Archive（CA）のコンポーネントから返されます。CA は、デバイス検出または展開時に更新されます。注：*この fullConfig は、CSM を使用せずにデバイスで直接実行されるアウトオブバンド（OOB）設定アップデートを含みません。*

セクション4およびセクション5も参照してください。

3.2.5.1 要求

メソッド GetDeviceConfigByGID 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getDeviceConfigByGID

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<deviceConfigByGIDRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <gid>00000000-0000-0000-0000-051539607555</gid>
</deviceConfigByGIDRequest>
```

図 67：メソッド GetDeviceConfigByGID 要求の例

表 59：メソッド GetDeviceConfigByGID 要求 URL 属性の説明

URL 属性名	定義
deviceConfigByGIDRequest	グローバル ID の要求
gid	返すことを要求されているデバイスのグローバル オブジェクト ID。 注：デバイス GID は *getDeviceListByCapability*、*getDeviceListByGroup*、*GetGroupList* のいずれかの API を使用して取得することができます。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。

URL 属性名	定義
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="deviceConfigByGIDRequest" type="DeviceConfigByGIDRequest"/>
<xs:complexType name="DeviceConfigByGIDRequest">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="gid" type="ObjectIdentifier"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 68：DeviceConfigByGIDRequest XML スキーマ

3.2.5.2 応答

GetDeviceConfigByGID 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<deviceConfigResponse>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
              <device>
                 <gid>00000000-0000-0000-0000-214748364932</gid>
                 <name>10.77.208.138</name>
                 <lastUpdateTime>2011-05-26T00:11:53Z</lastUpdateTime>
                 <osType>asa</osType>
                 <osVersion>8.4(1)</osVersion>
                 <imageName>disk0:/asa831-k8.bin</imageName>
                 <fullConfig>
                    <!– will contain the full config of the device -->
                 </fullConfig>
                 <mgmtInterface>
                    <type>Management</type>
                    <identifier>mgmt</identifier>
                    <ipInterface>
                       <ipAddress>10.77.208.138/255.255.255.0</ipAddress>
                    </ipInterface>
                 </mgmtInterface>
                 <interfaceList>
                    <interface>
                       <type>GigabitEthernet</type>
                       <identifier>outside</identifier>
                       <ipInterface>
                            <ipAddress>20.10.30.42/255.255.255.0</ipAddress>
                       </ipInterface>
                    </interface>
                    <interface>
                       <type>FastEthernet</type>
                       <identifier>FastEthernet5/1</identifier>
                       <ipInterface>
                            <ipAddress>20.10.30.45/255.255.255.0</ipAddress>
                       </ipInterface>
                    </interface>
                 </interfaceList>
              <configState>committed</configState>
           </device>
</deviceConfigResponse>
```

図 69：GetDeviceConfigByGID 応答の例

表 60：GetDeviceConfigById 応答の要素と属性の説明

要素.属性名	定義
deviceConfigResponse	オブジェクト ID に一致するデバイス設定を返します。
デバイス	2.1.3で説明されているデバイス クラス。

```
<xs:element name="deviceConfigResponse" type="DeviceConfigResponse"/>
<xs:complexType name="DeviceConfigResponse">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="device" type="Device"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 70：DeviceConfigResponse XML スキーマ

メソッド特有のエラー（また、*GetGroupList* メソッドで定義されているエラー コードも該当します）。

コード	説明
2007	このエラーは、要求されたデバイス GID が CSM サーバにない場合に返されます。
2008	このエラーは、API が Configuration Archive モジュールと通信できない場合に返されます。
2009	このエラーは、API が Configuration Archive モジュールから設定を取出せない場合に返されます。
2011	このエラーは、ユーザが設定の表示を許可されていない場合に返されます。

表 61：GetDeviceConfigByGID メソッドのエラー コード

3.2.6 メソッド GetDeviceConfigByName

GetDeviceConfigByName メソッドは、メソッドに渡されるデバイス名に基づいて、特定のデバイス オブジェクトと関連する設定を返します。デバイス設定を要求しているユーザは、設定を要求しているデバイスの「view_device」および「view_cli」権限の両方がなければなりません。このメソッドが返す「fullConfig」要素のデータは、デバイスで実行される一般的なコマンド「show running-config」からのデータに等しくなります。このすべての設定は、CSM の Configuration Archive（CA）のコンポーネントから返されます。CA は、デバイス検出または展開時に更新されます。注：*この fullConfig は、CSM を使用せずにデバイスで直接実行されるアウトオブバンド（OOB）設定アップデートを含みません。*

セクション 4 およびセクション 5 も参照してください。

3.2.6.1 要求

メソッド GetDeviceConfigByName 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getDeviceConfigByName

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<deviceConfigByNameRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <name>rtr-1.cisco.com</name>
</deviceConfigByNameRequest>
```

図 71：メソッド GetDeviceConfigByName 要求の例

表 62：メソッド GetDeviceConfigByName 要求 URL 属性の説明

URL 属性名	定義
deviceConfigByNameRequest	デバイス名要求
name	返すことを要求されているデバイス名
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="deviceConfigByNameRequest" type="DeviceConfigByNameRequest"/>
<xs:complexType name="DeviceConfigByNameRequest">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="name" type="xs:string"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 72：DeviceConfigByNameRequest XML スキーマ

3.2.6.2 応答

応答は GetDeviceConfigByGID メソッドと同じです。

メソッド特有のエラー（また、*GetGroupList* メソッドと *GetDeviceConfigByGID* メソッドで定義されているエラー コードも該当します）。

コード	説明
2010	このエラーは、要求されたデバイス名が CSM サーバにない場合に返されます。
2012	このエラーは、デバイス名が要求でヌルまたはブランクの場合に返されます。

表 63：GetDeviceConfigByName メソッドのエラー コード

3.2.7 メソッド GetPolicyListByDeviceGID

GetPolicyListByDeviceGID メソッドは、特定のデバイス GID のポリシー名とタイプのリストを返します。返されるリストには、API の現在のバージョンでサポートされるポリシー タイプのみを含めます。ポリシー タイプは、対応するポリシーが設定されていない場合にも返されません。この API を使用するには、ユーザは *view_device* RBAC 権限を持っている必要があります。

以下はこのバージョン 1.0 でサポートされているポリシーのリストです。API *GetPolicyConfigByName* および *GetPolicyConfigyByDeviceGID* メソッドの要求で使用される必要があるポリシー タイプの列の正確な値。

ポリシー タイプ	説明
DeviceAccessRuleFirewallPolicy	ACL の設定に使用されます。
DeviceAccessRuleUnifiedFirewallPolicy	Unified ACL の設定に使用されます。
DeviceBGPRouterPolicy	ルーティング プロトコル。
DeviceNATTimeoutsRouterPolicy	ルータ NAT を設定するために使用されます。
DeviceNATTransOptionsFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
DeviceStaticRoutingFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
DeviceStaticRoutingRouterPolicy	ルータのスタティック ルートの設定に使用されます。
InterfaceNAT64ManualFirewallPolicy	NAT64 ポリシーを設定するために使用されます。
InterfaceNATAddressPoolFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATDynamicRulesFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATDynamicRulesRouterPolicy	ルータ NAT を設定するために使用されます。
InterfaceNATManualFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATObjectFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATPolicyDynamicRulesFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATRouterPolicy	ルータ NAT を設定するために使用されます。
InterfaceNATStaticRulesFirewallPolicy	ファイアウォール NAT を設定するために使用されます。
InterfaceNATStaticRulesRouterPolicy	ルータ NAT を設定するために使用されます。
InterfaceNATTransExemptionsFirewallPolicy	ファイアウォール NAT を設定するために使用されます。

3.2.7.1 要求

メソッド GetPolicyListByDeviceGID 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getPolicyListByDeviceGID

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version=”1.0” encoding=”UTF-8”?>
<policyListByDeviceGIDRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <gid>00000000-0000-0000-0000-000023456781</gid>
</policyListByDeviceGIDRequest>
```

図 73：メソッド GetPolicyListByDeviceGID 要求の例

表 64：メソッド GetPolicyListByDeviceGID 要求の URL 引数の説明

HTTP/XML の内容	定義
getPolicyListByDeviceGID	要求の引数を含むポリシー リストの要求
要素：GID	ポリシーが設定されるデバイスを識別する要素。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="policyListByDeviceGIDRequest" type="PolicyListByDeviceGIDRequest"/>
<xs:complexType name="PolicyListByDeviceGIDRequest">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence minOccurs="1" maxOccurs="1">
                <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 74：PolicyListByDeviceGIDRequest XML スキーマ

3.2.7.2 応答

GetPolicyListByDeviceGID 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<policyListDeviceResponse>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <policyList>
          <policyDesc>
           <name>– local --</name>
           <type>DeviceOSPFPolicy</type>
          </policyDesc>
          <policyDesc>
           <name>MySharedPolicy</name>
           <type> DeviceAccessRuleFirewallPolicy </type>
          </policyDesc>
        </policyList>
</policyListDeviceResponse>
```

図 75：GetPolicyListByDeviceGID 応答の例

表 65：GetPolicyListByDeviceGID 応答の要素と属性の説明

XML 要素と属性	定義
policyListResponse	特定されたデバイスで設定された 0 個以上のポリシーの記述子リストを返します。
要素のリスト：policyList	ポリシー記述子要素のリスト
要素：policyDesc	名前とタイプを含むポリシー記述子。タイプは、通常スキーマのそのポリシーに定義されたオブジェクト名と同じ名前です。たとえばファイアウォール ルールの場合、タイプは「DeviceAccessRuleFirewallPolicy」です。ポリシーがこのデバイスでローカル/プライベートの場合、ポリシー名は*ローカル*です。また、既存のポリシーが「*共有*」ポリシー（たとえば「SharedAccessRule」）の場合、名前はユーザ設定ポリシー名（ローカル以外）を含みます。 共有ポリシーが複数のデバイスで共有されているポリシーです。共有ポリシーの名前はシステム全体にわたって特定のポリシー タイプに一義的です。たとえば、*DeviceAccessRuleFirewallPolicy* のように *SharedAccessRule* として命名されるポリシーは 1 つだけです。またこの *SharedAccessRule* は、このポリシーをサポートする複数のデバイスに適用できます。

```
        <xs:element name="policyListDeviceResponse" type="PolicyListDeviceResponse"/>
    <xs:complexType name="PolicyListDeviceResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="policyList" minOccurs="1" maxOccurs="1">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="policyDesc" type="EntityDescriptor"
minOccurs="1" maxOccurs="unbounded"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 76：PolicyListDeviceResponse XML スキーマ

メソッド特有のエラー：

コード	説明
1001	このエラーは、要求されたデバイスがサポートされるポリシーのいずれも使用せずに設定された場合に返されます。
2013	このエラーは、API が CSM サーバから内部ポリシーのリストを取出せない場合に返されます。

表 66：GetPolicyListByDeviceGID メソッドのエラー コード

3.2.8 メソッド GetPolicyConfigByName

GetPolicyConfigByName メソッドは、メソッドに渡される共有ポリシー名に基づいて、特定のポリシー オブジェクトと関連する設定を返します。この API へのアクセスには、照会されているポリシーの **view_policy** 権限が必要です。このメソッドは、共有ポリシー（*ローカル*以外のポリシー）のデータを取り出すためだけに使用する必要があります。デバイスの適切な共有ポリシーは GetPolicyListByDeviceGID メソッドで入手してください。

3.2.8.1 要求

メソッド GetPolicyConfigByName 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getPolicyConfigByName

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<policyConfigByNameRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <name>global FW policy-1</name>
        <policyType>DeviceAccessRuleFirewallPolicy</policyType>
</policyConfigByNameRequest>
```

図 77：メソッド GetPolicyConfigByName 要求の例

表 67：メソッド GetPolicyConfigByName 要求 URL 属性の説明

メソッドの詳細	定義
policyConfigByNameRequest	名前の要求、および GetPolicyListByDeviceGID のコールによって以前に定義され返されたものと同じ属性。
要素：policyType	GetPolicyListByDeviceGID コールで返されるタイプと同じポリシー タイプ。有効なタイプのリストは **GetPolicyListByDeviceGID** メソッドのセクションで説明されています。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="policyConfigByNameRequest" type="PolicyConfigByNameRequest"/>
<xs:complexType name="PolicyConfigByNameRequest">
      <xs:complexContent>
            <xs:extension base="BaseReqResp">
                  <xs:sequence>
                        <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="policyType" type="xs:string" minOccurs="1" maxOccurs="1"/>
                  </xs:sequence>
            </xs:extension>
      </xs:complexContent>
</xs:complexType>
```

図 78：PolicyConfigByName 要求 XML スキーマ

3.2.8.2 応答

GetPolicyConfigByName 要求に対する応答の例を、次の図に示します。これらのメッセージのフィールドを次の表に示します。応答にはポリシー、およびポリシーに含まれる参照先のポリシー オブジェクト データが含まれます。注：このメソッドの応答にはページを付けられます。その他の詳細については 2.2.1.1 を参照してください。

```
<?xml version=”1.0” encoding=”UTF-8”?>
<ns1:policyConfigDeviceResponse xmlns:ns1="csm">
  <protVersion>1.0</protVersion>
  <policy>
    <deviceAccessRuleFirewallPolicy>
      <gid>00000000-0000-0000-0000-004294967927</gid>
      <name/>
      <lastUpdateTime>2011-08-05T05:04:49.926Z</lastUpdateTime>
      <type>FirewallRule</type>
      <orderId>1000</orderId>
      <isMandatoryAggregation>true</isMandatoryAggregation>
      <configState>deployed</configState>
      <isEnabled>true</isEnabled>
      <direction>In</direction>
      <permit>true</permit>
      <interfaceRoleObjectGIDs>
        <gid>00000000-0000-0000-0000-004294967559</gid>
      </interfaceRoleObjectGIDs>
      <sources>
        <networkObjectGIDs>
          <gid>00000000-0000-0000-0000-000000000100</gid>
        </networkObjectGIDs>
        <interfaceRoleObjectGIDs/>
      </sources>
      <destinations>
        <networkObjectGIDs>
          <gid>00000000-0000-0000-0000-000000000100</gid>
        </networkObjectGIDs>
        <interfaceRoleObjectGIDs/>
      </destinations>
      <services>
        <serviceObjectGIDs>
          <gid>00000000-0000-0000-0000-000000001041</gid>
        </serviceObjectGIDs>
      </services>
      <logOptions>
      </logOptions>
      <iosOptions>None</iosOptions>
    </deviceAccessRuleFirewallPolicy>
  </policy>
  <policyObject>
    <networkPolicyObject>
      <gid>00000000-0000-0000-0000-000000000100</gid>
      <name>any</name>
      <lastUpdateTime>2011-08-04T18:58:22.816Z</lastUpdateTime>
      <parentGID>00000000-0000-0000-0000-000000000000</parentGID>
      <type>Network</type>
      <comment>Predefined any network</comment>
      <nodeGID>00000000-0000-0000-0000-000000000001</nodeGID>
      <isProperty>false</isProperty>
      <subType/>
      <isGroup>false</isGroup>
      <ipv4Data>0.0.0.0/0.0.0.0</ipv4Data>
    </networkPolicyObject>
          ……..
  </policyObject>
</ns1:policyConfigDeviceResponse>
```

図 79：GetPolicyConfigByName 応答の例

要素.属性名	定義
policyConfigResponse	オブジェクト ID に一致するポリシー設定を返します。
ポリシー（Policy）	3.1.1 に説明されているポリシー クラス。

表 68：GetPolicyConfigByName 応答の要素と属性の説明

```
<xs:element name="policyConfigResponse" type="PolicyConfigResponse"/>
<xs:complexType name="PolicyConfigResponse">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="policy" type="BasePolicy"/>
                <xs:element name="policyObject" type="BasePolicyObject"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
```

図 80：PolicyConfigResponse XML スキーマ

次は、このメソッドに特有のエラー コードです。エラーの場合にこのメソッドが返す追加の汎用エラーが存在することがあります。

コード	説明
18	設定を取得中に内部エラーが発生している場合に返さます。
19	必要な入力パラメータが要求から欠落している場合に返されます。
20	要求されたポリシー タイプが正しくない場合に返されます。
22	ユーザがこのポリシー データの表示を許可されていない場合に返されます。
23	設定データが要求された入力パラメータとして使用できない場合に返されます。
24	サーバによって内部エラーが検出された場合に返されます（サポートされないポリシー タイプを処理する場合）。

表 69：GetPolicyConfigByName メソッドのエラー コード

3.2.9 メソッド GetPolicyConfigByDeviceGID

GetPolicyConfigByDeviceGID メソッドは、メソッドに渡されるデバイス ID およびポリシー タイプに基づいて、特定のポリシーおよび関連するポリシー オブジェクトを返します。この API へのアクセスには、照会されているポリシーの **view_policy** 権限、およびこのポリシーがフェッチされているデバイスの **view_device** 権限が必要です。

このメソッドはデバイスに適切なポリシー（*ローカル*、*共有に関係なく*）を返します。そのため、多くの場合、これはデバイスのサポートされるポリシー設定を読み込む最も便利なメソッドです。

3.2.9.1 要求

メソッド GetPolicyConfigByDeviceGID 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/configservice/getPolicyConfigById

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<policyConfigByDeviceGIDRequest>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <gid>00000000-0000-0000-0000-004294967927</gid>
        <policyType>DeviceAccessRuleFirewallPolicy</policyType>
</policyConfigByDeviceGIDRequest>
```

図 81：メソッド GetPolicyConfigByDeviceGID 要求の例

表 70：メソッド GetPolicyConfigByDeviceGID 要求 URL 属性の説明

メソッドの詳細	定義
policyConfigByDeviceByGIDRequest	グローバル ID の要求およびすでに定義されているものと同じ属性
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
<xs:element name="policyConfigByDeviceGIDRequest" type="PolicyConfigByDeviceGIDRequest"/>
<xs:complexType name="PolicyConfigByDeviceGIDRequest">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="gid" type="ObjectIdentifier"/>
                <xs:element name="policyType" type="string"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 82：PolicyConfigByDeviceGIDRequest XML スキーマ

3.2.9.2 応答

スキーマとエラー コードを含むこのメソッドの応答は、以前に記述された *GetPolicyConfigByName* メソッドと同じです。

3.2.10　メソッド GetSharedPolicyNamesByType

このメソッドは、特定のポリシー タイプについて CSM ポリシー ビューに存在するすべての共有ポリシーのリストを返します。

3.2.10.1 REST 要求：

```
URL:
http://hostname/configservice/getSharedPolicyListByType

HTTP Header:
Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT;
path=/; domain=.hostdomain.com

XML Argument:
<?xml version=”1.0” encoding=”UTF-8”?>
< policyNamesByTypeRequest>
                <protVersion>1.0</protVersion>
                <reqId>123</reqId>
                <policyType>DeviceAccessRuleFirewallPolicy</policyType>
</ policyNamesByTypeRequest >
```

図 83：getSharedPolicyNamesByType 要求の例

表 71：メソッド getSharedPolicyNamesByType 要求 URL 属性の説明

メソッドの詳細	定義
policyNamesByTypeRequest	特定のポリシー タイプについて CSM で定義されたすべての共有ポリシーのリストを取得する要求。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

3.2.10.2 応答オブジェクト：

```
<?xml version="1.0" encoding="UTF-8"?>
<policyNamesResponse>
        <protVersion>1.0</protVersion>
        <reqId>123</reqId>
        <policyType>DeviceAccessRuleFirewallPolicy</policyType>
        <policy>
          <name>policy-1</name>
          <deviceAssignments>
              <device>
                 <deviceGID>00000000-0000-0000-0000-004294967308</deviceGID>
                 <deviceName>1.1.1.1</deviceName>
              </device>
        </deviceAssignments>
        </policy>
        <policy>
          <name>policy-2</name>
          <deviceAssignments>
              <device>
                 <deviceGID>00000000-0000-0000-0000-004294967309</deviceGID>
                 <deviceName>1.1.1.2</deviceName>
              </device>
          </deviceAssignments>
        </policy>
```

図 84：GetSharedPolicyNamesByType 応答の例

要素.属性名	定義
policyNamesResponse	要求で渡されたポリシー タイプのポリシー名とデバイス割り当てを返します。
policyNamesResponse.policyType	要求で渡されたポリシー タイプ。
policyNamesResponse.policy	要求されたポリシー タイプごとに、CSM で定義されたポリシー。
policyNamesResponse.policy.name	共有ポリシーの名前。
policyNamesResponse.policy.deviceAssignments	共有ポリシーが割り当てられるデバイスのリスト。
policyNamesResponse.policy.deviceAssignments.deviceGID	共有ポリシーが割り当てられるデバイスのデバイス GID。
policyNamesResponse.policy.deviceAssignments.deviceName	共有ポリシーが割り当てられるデバイスのデバイス名。

表 72：GetSharedPolicyNamesByType 応答の要素と属性の説明

```
<xs:element name="policyNamesResponse" type="PolicyNamesResponse"/>
<xs:complexType name="PolicyNamesResponse">
      <xs:complexContent>
            <xs:extension base="BaseReqResp">
                  <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="policyTYpe" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="policy" minOccurs="1" maxOccurs="unbounded">
                              <xs:complexType>
                                    <xs:sequence>
                                          <xs:element name="policyName" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                          <xs:element name="deviceAssignments" minOccurs="0"
maxOccurs="1">
                                                <xs:complexType>
                                                      <xs:sequence>
                                                            <xs:element name="device" type="xs:string"
minOccurs="1" maxOccurs="unbounded”>
                                                                  <xs:complexType>
                                                                        <xs:sequence>
                                                                              <xs:element name="deviceGID"
type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                                                              <xs:element name="deviceName"
type="xs:string" minOccurs="1" maxOccurs="1"/>
                                                                        </xs:sequence>
                                                                  </xs:complexType>
                                                            </xs:element>
                                                      </xs:sequence>
                                                </xs:complexType>
                                          </xs:element>
                                    </xs:sequence>
                              </xs:complexType>
```

図 85：PolicyNamesResponse XML スキーマ

4 CSM Events Service API

イベント サービス API は、CSM で実行された操作に基づいて クライアントが CSM サーバからイベント通知を受信できるようにする機能を提供します。

イベント通知は、管理対象デバイスの CSM によって開始された設定の更新について説明します。管理対象デバイスのアウトオブバンド（OOB）設定変更は、CSM のヘルスおよびパフォーマンス モニタリング（HPM）機能がイネーブルになっている場合だけ、ASA（バージョン 7.2 以降）でサポートされます。

イベント通知は、デバイス ステータスの変更イベントについて説明します。デバイス ステータスの変更イベント通知を動作させるためには、デバイスが CSM と HPM それぞれによって管理およびモニタリングされる必要があります。

4.1 メソッド

イベント サービスは次のメソッドを定義します。

1. GetServiceInfo
    a. サービス名、バージョン、日付などを含むサービス固有の情報を返します。
2. EventSubscription
    a. クライアントがイベント通知を登録できます。

4.1.1 メソッド GetServiceInfo

GetServiceInfo メソッドは、サービスに関連するサービスの説明、バージョン情報、および該当する属性を返します。要求、応答、およびオブジェクト モデルは、セクション 3.2.1 で説明したものと同じです。

4.1.2 メソッド EventSubcription

eventSubscription メソッドにより、CSM クライントは、要求で CSM クライアントによって指定された条件に基づいてフィルタリングするイベント通知をサブスクライブできます。イベント サブスクリプション API は設定変更通知の Syslog イベントおよびデバイス ステータス変更通知の Syslog イベントのみをサポートします。

4.1.2.1 要求

メソッド eventSubscription 要求の例を次の図に示します。これらのメッセージのフィールドについて表 73 で説明します。

```
URL:

https://hostname/nbi/eventservice/eventSubscription

XML Argument:

<?xml version="1.0" encoding="UTF-8"?>
<eventSubRequest>
      <op>add</op>
      <subscriptionId>123454</subscriptionId>
      <eventFilterItem>
          <filterEventType>syslog</filterEventType>
          <filterEventFormat>xml</filterEventFormat>
          <filterEventCategory>configChange</filterEventCategory>
      </eventFilterItem>
      <syslogServer>
           <port>514</port>
           <destAddress>12.1.1.1</destAddress>
      </syslogServer>
</eventSubRequest>
```

図 86：eventSubscription ConfigChange XML の例

上記の要求は、イベントを登録するためにクライアントによって送信されます。API は、syslog プロトコルを使用したイベント通知のみサポートします（syslog でのイベント データの形式は、表および以降のセクションで説明する *XML* または syslog *プレーン テキスト* となります）。

すべてのイベント サブスクリプションがアクティブなログイン セッション にリンクされます。（サブスクリプション要求を開始する）セッションが**ログアウトまたは期限切れ**になると、セッションに対応するすべてのサブスクリプションは削除されます。そのため、イベント通知を受信する必要がある場合、クライアントがセッションを**有効な状態に**維持することが重要です。

次のイベントがサポートされます。

1. 設定変更イベント
    a. イベントは、設定が変更されてイベントで識別されたネットワーク デバイスに配布されるたびに送信されます（CSM またはアウトオブバンド経由）。
2. デバイス ステータス変化イベント
    a. このイベントは、CSM がデバイス接続状態の変更（接続または未接続）を検出するたびに送信されます。

表 73：eventSubscription 要求の要素と属性の説明

要素.属性名	定義
eventSubRequest	eventSubscription 要求により、クライアントは 1 つまたは複数のクラスにイベント ハンドラを登録できます。

要素.属性名	定義
eventSubRequest.op	列挙リストの 1 つである実行中の処理 {追加、削除}。 追加処理はイベント ハンドラの新規サブスクリプションを登録します。 削除処理はサブスクリプションを削除し、CSM クライアントはそのサブスクリプション ID のイベントを受信しなくなります。
eventSubRequest.subscriptionId	CSM クライアントによって登録されるこのサブスクリプションを一意的に識別するサブスクリプション ID。要求で指定されたフィルタに一致する各イベント通知でサブスクリプション ID が返されます。
eventSubRequest.eventFilterItem	オブジェクト EventFilterItem は、CSM クライアントが通知を要求するイベント タイプを識別するフィルタを指定します。
eventSubRequest.eventFilterItem.filterEventType	1.0 では、「syslog」として定義されます。将来のタイプは 1.0 以降でサポートされます。
eventSubRequest .eventFilterItem.filterEventCategory	クライアントの対象となるイベント カテゴリを指定します。複数の filterEvent は複数のタイプのイベントをリッスンするように指定できます。configChange および deviceStatus がサポートされます。
eventSubRequest.eventFilterItem.filterEventFormat	通知メッセージで使用されるイベント形式。これは形式の列挙リストです。リストは{xml、プレーン テキスト}になります。
eventSubRequest.SyslogServer	クライアントが特定の宛先およびポート番号で syslog プロトコルによってイベントを受信する場合、サブスクリプションに含まれます。本来 syslog 形式ではないイベントがカプセル化されます。
eventSubRequest.syslogServer.port	syslog リレーがイベントを転送する UDP ポート番号
eventSubRequest.syslogServer.destAddress	転送された syslog イベントを受信する IPv4 アドレス
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
        <xs:simpleType name="EventType">
                <xs:restriction base="xs:token">
                        <xs:enumeration value="syslog"/>
                </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="EventFormat">
                <xs:restriction base="xs:token">
                        <xs:enumeration value="xml"/>
                        <xs:enumeration value="plainText"/>
                </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="EventCategory">
                <xs:restriction base="xs:token">
                        <xs:enumeration value="configChange"/>
                        <xs:enumeration value="deviceStatus"/>
                </xs:restriction>
        </xs:simpleType>
        <xs:complexType name="EventFilterItem">
                <xs:sequence>
                        <xs:element name="filterEventType" type="EventType" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="filterEventFormat" type="EventFormat" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="filterEventCategory" type="EventCategory" minOccurs="1"
maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>
        <xs:element name="eventSubRequest" type="EventSubRequest"/>
        <xs:complexType name="EventSubRequest">
                <xs:complexContent>
                        <xs:extension base="BaseReqResp">
                                <xs:sequence>
                                        <xs:element name="op" type="SubscriptionOperation" minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="subscriptionId" type="xs:string" minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="eventFilterItem" type="EventFilterItem" minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="syslogServer" type="SyslogServer" minOccurs="0" maxOccurs="1"/>
                                </xs:sequence>
                        </xs:extension>
                </xs:complexContent>
        </xs:complexType>
```

図 87：EventSubRequest XML スキーマ

4.1.2.2 応答

eventSubscription 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version=”1.0” encoding=”UTF-8”?>
      <eventSubResponse>
         <protVersion>1.0</protVersion>
         <reqId>123</reqId>
         <subscriptionId>12345</subscriptionId>
      </eventSubResponse>
```

図 88：eventSubscription 応答の例

表 74：eventSubscription 応答の要素と属性の説明

要素.属性名	定義
eventSubResponse	サブスクリプションが成功したかどうかを返します。成功した場合、クライアントから送信されたサブスクリプション ID を含みます。

```
<xs:element name="eventSubResponse" type="EventSubResponse"/>
<xs:complexType name="EventSubResponse">
    <xs:complexContent>
        <xs:extension base="BaseReqResp">
            <xs:sequence>
                <xs:element name="subscriptionId" type="xs:string" minOccurs="0"
maxOccurs="1"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

図 89：EventSubResponse XML スキーマ

以下は syslog 通知に関する警告です。

- syslog 通知は、CSM によって現在管理されているデバイスだけに送信されます。つまり、デバイスはデバイス インベントリに追加され、CSM によって「管理」する必要があります。
- CSM はデバイスの CSM での管理方法に基づいて、オプションで Auto Update Server (AUS) および Cisco Networking Service (CNS) Configuration Engine への変更を展開できます。AUS/CNS 方式は設定の更新が実際のデバイスに展開される前に、中間の「ステージング システム」を提供します。詳細については、http://www.cisco.com/en/US/docs/security/security_management/cisco_security_manager/security_manager/4.4/user/guide/dpman.html#wp938164 を参照してください。CNS または AUS に正常に展開された場合、イベントの通知方式は**正常な設定変更**の更新を送信します（そのためイベントの送信時に、設定変更が実際のエンド デバイスでアクティブではない可能性があります）。
- 以下はアウトオブバンド（OOB）通知に関する固有の要点です。
    - アウトオブバンド変更検出は、現在 ASA デバイスでのみイネーブルです（バージョン 7.2.x 以降）。また、ASA デバイスの OOB 検出は ASA デバイスが CSM の Health and Performance Monitoring（HPM）機能によって監視されている場合だけイネーブルになります。
    - OOB の通知は、少なくとも「1 つの正常なイベント サブスクリプションの実行後」にのみ、OOB 変更の検出を開始します。この登録以前の OOB イベントは不明で、モニタされません。
    - HPM コンポーネントは、デバイスがアクティブに監視されているか非アクティブに監視されているかに基づいて、デバイスを 5 または 10 分ごとに監視します。したがって、OOB イベントは HPM のモニタリング サイクル/ポーリングが完了した場合だけ作成されます。つまり、OOB 変更がデバイスで実行された後すぐには OOB イベントは生成されないことになります。
    - イベント サービスは、CSM によって管理されていない可能性がある変更された CLI/config ラインを含むすべての OOB 設定変更を検出して通知します。検出には変更されたり復元された可能性のある CLI も含まれます。つまり、システムはデバイスの設定で変更されたイベントをすべて検出します。

- まれなケースとして、CSM による展開が OOB 変更の直後（HPM モジュールが変更を検出する前）に実行されると、OOB のイベントが生成されない場合があります。次の例を考慮してください。
  - 午前 10:00 にユーザは 10 分間隔でモニタされている ASA1 に OOB 変更を加えます（次のモニタリングは午前 10:09）。
  - 午前 10:03 にユーザは ASA1 の変更を上書きする CSM 展開を実行し、展開は正常に完了します。
  - 午前 10:03 に実行された CSM 展開によって上書きされるため、HPM ポールサイクルは午前 10:09 に OOB 変更を認識しません。

- 以下は deviceStatus 通知に特有の要点です。
  - 通知は、CSM の Health and Performance Monitoring（HPM）機能の記述「Device Polling」を含むすべてのアラートに対して送信されます。すべての通知が CSM サーバが再起動されるたびに送信されます。デバイス ステータスの変更通知だけその後送信されます。
  - 警告の記述が「Device Polling: Connection Timedout」の場合は、DEVICE_DOWN の通知が送信されます。
  - キ警告の記述が「Device Polling: Connected」の場合は、DEVICE_UP 通知が送信されます。
  - HPM コンポーネントは、デバイスがアクティブに監視されているか非アクティブに監視されているかに基づいて、デバイスを 5 または 10 分ごとに監視します。したがって、デバイス ステータス イベントは HPM のモニタリング サイクル/ポーリングが完了した場合だけ作成されます。つまりデバイスがダウンして起動する可能性があり、HPM がこれを検出しないため、この場合デバイス ステータスの変更通知は送信されません。

次は、このメソッドに特有のエラー コードです。エラーの場合にこのメソッドが返す追加の汎用エラーが存在することがあります。

コード	説明
4001	これは重複するサブスクリプション要求がイベント サブスクリプションの間に作成されたことを示します。2 つのサブスクリプションが同じサブスクリプション ID を使用して（同じユーザ セッション）作成されると、このエラーが返されます。
4002	このエラーは、指定された Syslog の宛先 IP アドレスが無効であることを示します。IP アドレスは、マスク アドレスを指定せずに a.b.c.d 形式にする必要があります（例：192.168.10.10）。
4003	このエラーは、ユーザが存在しないサブスクリプションを削除しようとすると返されます。
4004	このエラーは、指定された syslog ポートが無効であったことを示します。有効なポートの範囲は 1～65535 です。
4005	このエラーは、無効なサブスクライバ ID が指定されたことを示します。空白だけ含めるサブスクライバ ID は許可されません。
4006	このエラーは、「Syslog Server」要素または「Event Filter」要素が指定されていないことを示します。これらの要素は、追加要求でのみ指定する必要があります。

4007	このエラーは、「Syslog Server」要素または「Event Filter」要素のいずれかが削除処理に指定されたことを示します。これらの要素は両方とも削除要求に指定しないでください。

表 75：EventSubscription メソッドのエラー コード

次の 2 つのセクションは、サブスクリプション時にプレーン テキスト形式に基づいた XML が選択されたかに基づいて、syslog 通知の内容を詳しく説明します。

4.1.2.3 Syslog XML イベント通知

クライアントが Syslog サーバ オプションを使用して Syslog イベント転送に登録すると、すべてのイベントが、登録された宛先 IP とポート番号に転送されます。syslog メッセージの内容は、クライアントに指定された登録パラメータによって、XML またはプレーン テキストで送信される可能性があります。XML 形式を選択するとイベント メッセージは、次に示すようにスキーマに従います。

以下は、設定変更通知の例です。

```
<n:configChangeEvent xmlns:n="csm" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="events.xsd">
    <eventType>syslog</eventType>
    <eventCategory>configChange</eventCategory>
    <time>1697-02-01T00:00:00Z</time>
    <content>this is a syslog event</content>
    <srcIP>12.1.1.1</srcIP>
    <srcGID>00000000-0000-0000-0000-000000000890</srcGID>
    <srcDns>cisco.com</srcDns>
    <srcOSType>asa</srcOSType>
    <deploymentType>Device</deploymentType>
    <updateType>NO_OOB</updateType>
</n:configChangeEvent>
```

図 90：設定変更通知の例

上記のイベント通知はサーバによって登録された Syslog リスナーに送信され、次の要素と属性に基づいています。

表 76：ConfigChangeEvent のデータ要素の説明

要素名	定義
configChangeEvent	CSM 特有の設定変更イベント通知を示すルート要素。
configChangeEvent.eventType	このメッセージが syslog メッセージであることを示します。
configChangeEvent.eventCategory	イベント カテゴリ configChange を示します。
configChangeEvent.subscriptionId	現在は「エコー」バックされるクライアントによって使用される、元のサブスクリプション ID。
configChangeEvent.time	このイベントが生成された時刻。
configChangeEvent.content	設定変更に関連する追加のメッセージ。
configChangeEvent.srcIP	変更されたデバイスの IP アドレス
configChangeEvent.srcGID	変更されたデバイスの GID
configChangeEvent.srcDns	変更されたデバイスの DNS 名（存在する場合）。
configChangeEvent.srcOSType	デバイス タイプ FWSM、ASA、IOS、PIX を示します。

要素名	定義
configChangeEvent.deploymentType	展開のタイプを示します（*「不明」に設定されたアウトオブバンド変更の場合は適用不可*）。以下は許可される値です。 • **Device →** デバイスへ展開 • **File →** ファイルへ展開 • **AUS →** Auto Update Server へ展開 • **CNS →** Cisco Networking Services、Configuration Engine へ展開 • **TMS →** Token Management Server へ展開
configChangeEvent.updateType	**NO_OOB →** この設定変更が**アウトオブバンド**（OOB）変更ではないことを示します。つまり、デバイスへの設定変更は Cisco Security Manager によって行われたことを示します。 **OOB →** この設定変更が Cisco Security Manager の「外部」で実行された OOB 変更であることを示します。**CSM に ASA（バージョン 7.2 以降）デバイス単独の OOB 変更を検出する機能があります**。

次の図は、イベント通知のスキーマを示します。

```
    <xs:simpleType name="OSType">
        <xs:restriction base="xs:token">
            <xs:enumeration value="ios"/>
            <xs:enumeration value="fwsm"/>
            <xs:enumeration value="asa"/>
            <xs:enumeration value="ips"/>
            <xs:enumeration value="pix"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="event" type="Event"/>
    <xs:complexType name="Event">
        <xs:choice>
            <xs:element name="configChange" type="ConfigChangeEvent"/>
        </xs:choice>
    </xs:complexType>
    <xs:simpleType name="UpdateType">
        <xs:restriction base="xs:token">
            <xs:enumeration value="NO_OOB"/>
            <xs:enumeration value="OOB"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="DeploymentType">
        <xs:restriction base="xs:token">
            <xs:enumeration value="Device"/>
            <xs:enumeration value="File"/>
            <xs:enumeration value="AUS"/>
            <xs:enumeration value="CNS"/>
            <xs:enumeration value="TMS"/>
            <xs:enumeration value="Unknown"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="BaseEventDetails">
        <xs:sequence>
            <xs:element name="eventType" type="EventType" minOccurs="1" maxOccurs="1"/>
            <xs:element name="eventCategory" type="EventCategory" minOccurs="1" maxOccurs="1"/>
            <xs:element name="time" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
            <xs:element name="content" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="DeviceSpecificEvent">
        <xs:complexContent>
            <xs:extension base="BaseEventDetails">
                <xs:sequence>
                    <xs:element name="srcIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="srcGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="srcDns" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="srcOSType" type="OSType" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!-- Config Change Event -->
    <xs:element name="configChangeEvent" type="ConfigChangeEvent"/>
    <xs:complexType name="ConfigChangeEvent">
        <xs:complexContent>
            <xs:extension base="DeviceSpecificEvent">
                <xs:sequence>
                    <xs:element name="deploymentType" type="DeploymentType" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="updateType" type="UpdateType" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
```

図 91：イベント XML スキーマ

以下は、デバイス ステータスの変更通知の例です。

```
<n: deviceStatusEvent xmlns:n="csm" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="events.xsd">
        < subscriptionId >100</ subscriptionId >
        <eventType>syslog</eventType>
        <eventCategory>deviceStatus</eventCategory>
        <time>2013-01-25T04:47:59.082Z</time>
        <content>Connection Error</content>
        <srcIP>12.1.1.1</srcIP>
        <srcGID>00000000-0000-0000-0000-000000000890</srcGID>
        <srcDns>cisco.com</srcDns>
        <srcOSType>asa</srcOSType>
        <updateType>DEVICE_DOWN </updateType>
</n: deviceStatusEvent>
```

図 92：デバイス ステータス通知の例

上記のイベント通知はサーバによって登録された Syslog リスナーに送信され、次の要素と属性に基づいています。

表 77：DeviceStatusEvent のデータ要素の説明

要素名	定義
deviceStatusEvent	デバイス ステータス イベント通知を示すルート要素。
deviceStatusEvent.eventType	このメッセージが syslog メッセージであることを示します。
deviceStatusEvent.eventCategory	イベント カテゴリ deviceStatus を示します。
deviceStatusEvent.subscriptionId	現在は「エコー」バックされるクライアントによって使用される、元のサブスクリプション ID。
deviceStatusEvent.time	このイベントが生成された時刻。
deviceStatusEvent.content	設定変更に関連する追加のメッセージ。
deviceStatusEvent.srcIP	変更されたデバイスの IP アドレス
deviceStatusEvent.srcGID	変更されたデバイスの GID。
deviceStatusEvent.srcDns	変更されたデバイスの DNS 名（存在する場合）。
deviceStatusEvent.srcOSType	デバイス タイプ FWSM、ASA、IOS、PIX を示します。
deviceStatusEvent.updateType	**DEVICE_DOWN** → デバイスは CSM サーバから到達可能ではありません。 **DEVICE_UP** → デバイスが起動し CSM サーバから到達可能です。

次の図は、イベント通知のスキーマを示します。

```
<xs:simpleType name="OSType">
        <xs:restriction base="xs:token">
                <xs:enumeration value="ios"/>
                <xs:enumeration value="fwsm"/>
                <xs:enumeration value="asa"/>
                <xs:enumeration value="ips"/>
                <xs:enumeration value="pix"/>
        </xs:restriction>
</xs:simpleType>
<xs:element name="event" type="Event"/>
<xs:complexType name="Event">
        <xs:choice>
                <xs:element name="configChange" type="ConfigChangeEvent"/>
        </xs:choice>
</xs:complexType>
<xs:simpleType name="UpdateType">
        <xs:restriction base="xs:token">
                <xs:enumeration value="NO_OOB"/>
                <xs:enumeration value="OOB"/>
                <xs:enumeration value="DEVICE_DOWN"/>
                <xs:enumeration value="DEVICE_UP"/>
        </xs:restriction>
</xs:simpleType>
<xs:simpleType name="DeploymentType">
        <xs:restriction base="xs:token">
                <xs:enumeration value="Device"/>
                <xs:enumeration value="File"/>
                <xs:enumeration value="AUS"/>
                <xs:enumeration value="CNS"/>
                <xs:enumeration value="TMS"/>
                <xs:enumeration value="Unknown"/>
        </xs:restriction>
</xs:simpleType>
<xs:complexType name="BaseEventDetails">
        <xs:sequence>
                <xs:element name="eventType" type="EventType" minOccurs="1" maxOccurs="1"/>
                <xs:element name="eventCategory" type="EventCategory" minOccurs="1" maxOccurs="1"/>
                <xs:element name="time" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
                <xs:element name="content" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="DeviceSpecificEvent">
        <xs:complexContent>
                <xs:extension base="BaseEventDetails">
                        <xs:sequence>
                                <xs:element name="srcIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
                                <xs:element name="srcGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="srcDns" type="xs:string" minOccurs="0" maxOccurs="1"/>
                                <xs:element name="srcOSType" type="OSType" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                </xs:extension>
        </xs:complexContent>
</xs:complexType>
<!—Device Status Event -->
<xs:element name=" deviceStatusEvent " type=" DeviceStatusEvent "/>
<xs:complexType name=" DeviceStatusEvent ">
        <xs:complexContent>
                <xs:extension base="DeviceSpecificEvent">
                        <xs:sequence>
                                <xs:element name="updateType" type="UpdateType" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                </xs:extension>
        </xs:complexContent>
</xs:complexType>
```

図 93：イベント XML スキーマ

4.1.2.4 Syslog PlainText イベント通知

登録された **filterEventFormat** がイベント サブスクリプションの間に**プレーン テキスト**として指定されると、ネイティブ形式の通知は syslog プロトコルを使用して送信されます。

次は設定変更通知の例を表しています。

```
[Mon Aug 29 08:30:21 IST 2011]syslog-configChange-101:10.104.52.71 SUCCEEDED in job
admin_job_2011-08-29 08:30:12.148,10.104.52.71,00000000-0000-0000-0000-017179869189,NO DOMAIN
NAME DEFINED,ios,Device,NO_OOB.
```

次はデバイス ステータスの変更通知の例です。

```
[Thu Jan 24 20:47:59 PST 2013]syslog-deviceStatus-102:,10.104.52.71,00000000-0000-0000-0000-
042949673287,default.domain.invalid,asa,DEVICE_DOWN
```

このメッセージの標準形式は以下となります。

```
[time-stamp]<eventType>-<eventCategory>-<subscriptionId>:{Comma separated list of event
details - <content>,<srcIp>,<srcGID>,<srcDNS>,<srcOSType>,<deploymentType>,<updateType>}
```

コンマ区切りリスト内の要素は、XML スキーマで定義された特定のイベント タイプの要素の順に並んでいます。デバイスに IP アドレスがない場合は、文字列「NO IP DEFINED」（srcIP）がメッセージに含まれます。ソース DNS（srcDNS）がない場合は、「NO DOMAIN NAME DEFINED」が使用されます。

5 CSM Utility Service API

ユーティリティ サービスでは、ネットワーク デバイスの機能へのアクセスに対して汎用ユーティリティが用意されています。Utility Service API は常にデバイスからデバイス設定データを直接取り出します。一方、Configuration Service API （セクション3を参照）は常に CSM データベースからデータを取り出します。

Utility Service API よりも Configuration Service API を使用したほうが、応答が迅速になり、効率が向上します。これは、Utility Service API はメソッド要求中にデバイスと通信し、CSM アプリケーションとネットワーク デバイス間に追加の通信オーバーヘッドあるためです。このことは、ネットワーク デバイスで負荷をさらに増加させる可能性があります。

そのため次のような条件下で Utility Service API を使用することを推奨します。

- 対応する設定データが Configuration ServiceAPI によってサポートされていない場合。
- 対応する設定データが CSM で管理されていない場合。デフォルトでは、すべての適用可能なポリシーが CSM で管理されます（[**Policy Management** CSM Administration] 画面を参照してください）。
- アウトオブバンド変更がエンド デバイスで実行されている場合（この場合、CSM はデータベースで更新された設定がない場合があります）。

5.1 オブジェクト モデル

次のオブジェクトは Utility Service で定義されます。

- DeviceReadOnlyCLICmds
    - デバイスおよびそのデバイスに対して実行された CLI コマンドを示します。
- DeviceCmdResults
    - 一連のデバイス コマンドの結果を示します。
- DeviceCmdResult
    - 単一デバイスおよびコマンドの結果を示します。

```
<xs:simpleType name="Result">
    <xs:restriction base="xs:token">
        <xs:enumeration value="ok"/>
        <xs:enumeration value="timeout"/>
        <xs:enumeration value="failed"/>
    </xs:restriction>
</xs:simpleType>
```

図 94：結果 XML スキーマ

5.2 メソッド

Utility Service は次のメソッドを定義します。

1. GetServiceInfo
    a. サービス名、バージョン、日付などを含むサービス固有の情報を返します。
2. execDeviceReadOnlyCLICmd
    a. メソッド引数で指定されたデバイスで識別された CLI コマンドを実行します。

5.2.1 メソッド GetServiceInfo

GetServiceInfo メソッドは、サービスに関連するサービスの説明、バージョン情報、および該当する属性を返します。この要求の URL は、*https://hostname/nbi/utilservice/GetServiceInfo* です。要求、応答、およびオブジェクト モデルは、セクション 3.2.1 で説明したものと同じです。

5.2.2 メソッド execDeviceReadOnlyCLICmds

execDeviceReadOnlyCLICmds メソッドはリストの特定されたデバイスに対して一連のコマンドを実行し、CSM クライアントに各コマンドの結果を返します。このメソッドは、CSM サーバが持つ読み取り専用クレデンシャルを使用します。そのため、デバイスに対して読み取り専用を超えるクレデンシャルが必要なコマンドの実行に使用することはできません。**このメソッドは、IPS/IDS にデバイスに対して実行されません。**

このメソッドで実行できるコマンドのセットは、統計情報、特定のデバイスの操作に関する追加情報を提供するモニタリング コマンドなどの読み取り専用コマンドです。たとえば、クライアント アプリケーションは、このコマンドによってデバイスから統計情報を収集する CLI コマンドを呼び出す可能性があります。CSM の *view_cli* および *view_device* 権限はこのメソッドを使用する必要があります。このメソッドは、1 台のデバイスの設定情報を一度に取得します。

注：show コマンド出力に W3C XML 規格予約記号がある場合、API 応答は、World Wide Web Consortium（W3C）標準による上記特殊文字を符号化します。たとえば、マルチ コンテキスト デバイスから出力される「show version」に <context> という出力があると、応答は <context> という名前付き文字参照を表示します。詳細については、http://www.w3.org/TR/REC-xml/#syntax を参照してください。

5.2.2.1 要求

メソッド execDeviceReadOnlyCLICmds 要求の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
URL:

https://hostname/nbi/utilservice/execDeviceReadOnlyCLICmds

HTTP Header:

Set-Cookie: asCookie=732423sdfs73242; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/;
domain=.hostdomain.com

XML Argument:

<?xml version=”1.0” encoding=”UTF-8”?>
      <execDeviceReadOnlyCLICmdsRequest>
         <protVersion>1.0</protVersion>
         <reqId>123</reqId>
         <deviceReadOnlyCLICmd>
           <deviceIP>12.1.1.1</deviceIP>
           <cmd>show</cmd>
           <argument>run all</argument>
           <execTimeout>5</execTimeout>
         </deviceReadOnlyCLICmd>
      </execDeviceReadOnlyCLICmdsRequest>
```

図 95：メソッド execDeviceReadOnlyCLICmds 要求の例

表 78：メソッド execDeviceReadOnlyCLICmds 要求の要素と属性の説明

要素.属性名	定義
execDeviceReadOnlyCLICmdsRequest	execDeviceReadOnlyCLICmds 要求は、特定されたデバイスのセットのコマンドを実行します。
deviceReadOnlyCLICmd	実行するデバイス コマンドの詳細。deviceIP、deviceName、または deviceGID を選択し、コマンドが実行されるデバイスを識別します。デバイス クレデンシャルの検索を避けるために、deviceName および deviceGID は deviceIP より効率的に実行します。
deviceIP	コマンドが実行されるデバイスの IP アドレス。
deviceName	コマンドが実行されるデバイスの名前。
deviceGID	コマンドが実行されるデバイスのオブジェクト ID。
cmd	固定コマンド「show」。regex は「sS」「hH」「oO」「wW」のように大小文字の混合が可能です。
argument	show コマンドの引数。デバイスの実行設定を表示する「実行」や、アクセスリストの詳細を表示する「アクセスリスト」など。
execTimeout	execTimeout 属性は任意に選択され数秒でタイムアウトになります。execTimeout 属性は、符号なし整数です。デフォルトは 180 秒に設定されます。 CSM は 15 秒の遅延で、3 回の EXEC コマンドを試みます。それぞれの試行は execTimeout 属性に指定された値でタイムアウトします。
reqId	CSM クライアントと CSM サーバ間の要求/応答トランザクション ペアを一意的に識別する要求 ID。
HTTP メソッド	POST
HTTP ヘッダー：**asCookie**	Cookie は、認証セッションを識別するログイン メソッドによって返されます。
戻り値	200 OK + XML
	401 Unauthorized

```
    <xs:complexType name="DeviceReadOnlyCLICmd">
        <xs:sequence>
            <xs:choice minOccurs="1" maxOccurs="1">
                <xs:element name="deviceIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="deviceName" type="xs:string" minOccurs="1" maxOccurs="1" />
                <xs:element name="deviceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
            </xs:choice>
            <xs:element name="cmd" minOccurs="1" maxOccurs="1">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:pattern value="[sS][hH][oO][wW]"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="argument" type="xs:string" minOccurs="1" maxOccurs="1">
            <xs:element name="execTimeout" type="xs: unsignedInt" minOccurs="0" maxOccurs="1">
        </xs:element>
       </xs:sequence>
    </xs:complexType>

    <xs:element name="execDeviceReadOnlyCLICmdsRequest"
type="ExecDeviceReadOnlyCLICmdsRequest"/>
    <xs:complexType name="ExecDeviceReadOnlyCLICmdsRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence>
                    <xs:element name="DeviceReadOnlyCLICmd" type="DeviceReadOnlyCLICmd"
minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 96：ExecDeviceReadOnlyCLICmdsRequest XML スキーマ

5.2.2.2 応答

execDeviceReadOnlyCLICmds 応答の例を次の図に示します。これらのメッセージのフィールドを次の表に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
      <execDeviceReadOnlyCLICmdsResponse>
         <protVersion>1.0</protVersion>
         <reqId>123</reqId>
         <deviceCmdResult>
           <deviceIP>12.1.1.1</deviceIP>
           <deviceName>rtr.cisco.com</deviceName>
           <deviceGID>00000000-0000-0000-0000-261993005068</deviceGID>
           <result>ok</result>
           <resultContent>
              FWSM Firewall Version 3.1(16) &lt;context>
              Device Manager Version 5.1(1)
              Compiled on Wed 29-Jul-09 02:10 by fwsmbld
              U27-FWSM up 5 days 3 hours
              Hardware:   WS-SVC-FWM-1, 1024 MB RAM, CPU Pentium III 1000 MHz
              Flash STI Flash 8.0.0 @ 0xc321, 20MB
               0: Int: Not licensed        : irq 5
               1: Int: Not licensed        : irq 7
               2: Int: Not licensed        : irq 11
              Licensed features for this platform:
              Maximum Interfaces          : 100
              Inside Hosts                : Unlimited
              Failover                    : Active/Active
              VPN-DES                     : Enabled
              VPN-3DES-AES                : Enabled
              Cut-through Proxy           : Enabled
              Guards                      : Enabled
              URL Filtering               : Enabled
              Security Contexts           : 250
              GTP/GPRS                    : Disabled
              VPN Peers                   : Unlimited
              Serial Number: SAD11420A0N
              Running Activation Key: 0xe3bbe77c 0x0b82b2ba 0x4014b998 0x6bef38ad
              Configuration has not been modified since last system restart.
           <resultContent>
         </deviceCmdResult>
      </execDeviceReadOnlyCLICmdsResponse>
```

図 97：execDeviceReadOnlyCLICmds 応答の例

表 79：execDeviceReadOnlyCLICmds 応答の要素と属性の説明

要素.属性名	定義
execDeviceReadOnlyCLICmdsResponse	コマンドの結果を返します。
serviceVersion	コンフィギュレーション サービス実行のサービス バージョン。
deviceCmdResult	特定のデバイスのコマンドの結果の詳細
deviceIP	デバイスの IP アドレス（利用できる場合）。
deviceName	結果を返したデバイスのデバイス名
deviceGID	結果を返したデバイス オブジェクト ID
result	コマンドの結果の列挙 { success, generalFailure, timeout }
resultContent	コマンドの結果内容は「コマンドが正常に終了」となります。

```
    <xs:complexType name="DeviceCmdResult">
        <xs:sequence>
            <xs:element name="deviceIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
            <xs:element name="deviceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
            <xs:element name="deviceName" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="result" type="Result" minOccurs="1" maxOccurs="1"/>
            <xs:element name="resultContent" type="xs:string" minOccurs="0" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>


<xs:element name="execDeviceReadOnlyCLICmdsResponse"
type="ExecDeviceReadOnlyCLICmdsResponse"/>
    <xs:complexType name="ExecDeviceReadOnlyCLICmdsResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence>
                    <xs:element name="deviceCmdResult" type="DeviceCmdResult" minOccurs="1"
maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

図 98：ExecDeviceReadOnlyCLICmdsRequest XML スキーマ

次は、このメソッドに特有のエラー コードです。エラーの場合にこのメソッドが返す追加の汎用エラーが存在することがあります。

コード	説明
3000	このエラーは、API がネットワーク デバイスで show コマンドを実行できない場合に返されます。
3001	このエラーは、CSM サーバで要求された IP アドレスを持つデバイスがない場合に返されます。
3002	このエラーは、要求されたデバイスが IPS デバイスの場合に返されます。
3003	このエラーは、デバイスへのアクセス時に、API に何らかの問題が発生した場合、たとえば無効なクレデンシャルでデバイスに要求を送信する場合などに返されます。
27	Exectimeout Failure：タイムアウト内にデバイスからの応答がない。

表 80：ExecDeviceReadOnlyCLICmdsRequest メソッドのエラー コード

6 API スケーリング

API は、さまざまな展開をサポートします。次のガイドラインに従う必要があります。

1) API によってサポートされるデバイスの数に制限はありません。現在の CSM の推奨事項は、イベント管理およびその他の機能でサーバあたり約 500 デバイスをイネーブルにすることです。

7 CSM Client Protocol State Machine

7.1.1 概要

CSM クライアントが CSM サーバ上のサービスを使用する前に、2 つの前提条件があります。

1 番目の前提条件は、クライアントが CSM サーバに認証されることです。2 番目の前提条件は、必要なサービスがすべて CSM サーバでアクティブになっていて、これらのサービスのバージョンがクライアントで予期されるバージョンに対応することをクライアントが確認することです。このプロセスは、クライアントがインターフェイスに対する後続のコールで使用する必要がある認証 Cookie を戻します。このフローは、次の図に示します。

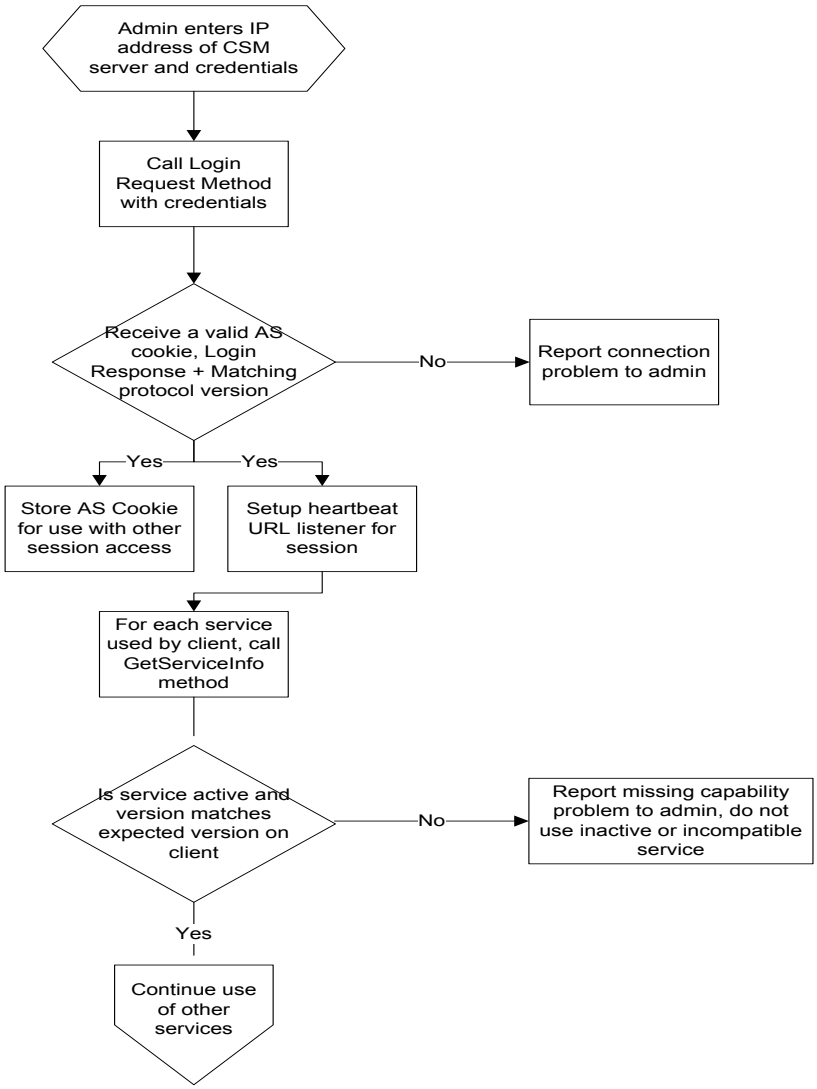


図 99：クライアント セッションの開始のフロー チャート

セッションが設定されている場合は、クライアントは、設定、イベント、ユーティリティ サービスに対するサービス メソッドにアクセスする可能性があります。

ハート ビート コールバックに登録済みのクライアントは以下に定義されたフローに従う必要があります。

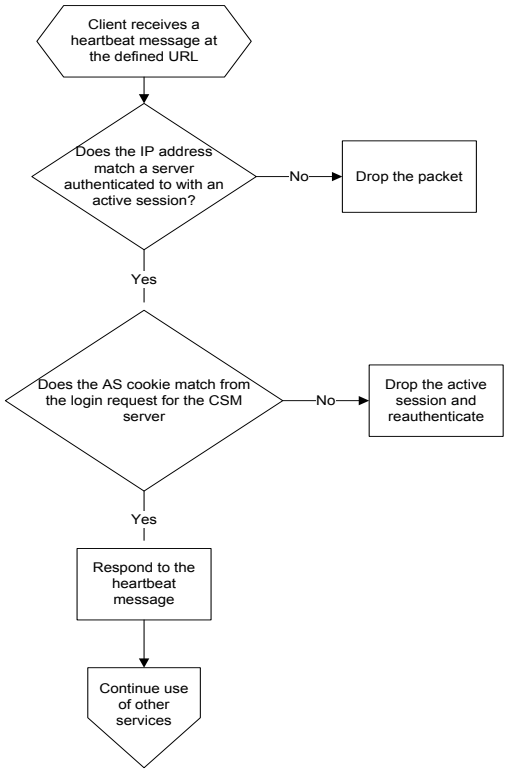


図 100：クライアント ハート ビート プロセス フローチャート

クライアントがハート ビート コールバックの登録は完了していないが CSM サーバでのアクティブ セッションを維持する場合は、ping メソッドを定期的に呼び出す必要があります。

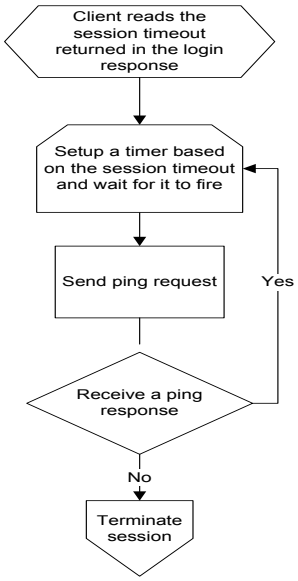


図 101：クライアントの ping プロセス フローチャート

7.1.2 設定とイベント サービスの使用

CSM クライアントは CSM サーバでサポートされるデバイスの設定を読み込むための API にアクセスすることができます。またクライアントは、設定が変更されるたびに変更通知を登録することができます。次のフロー チャートはクライアントの詳細な処理を示します。

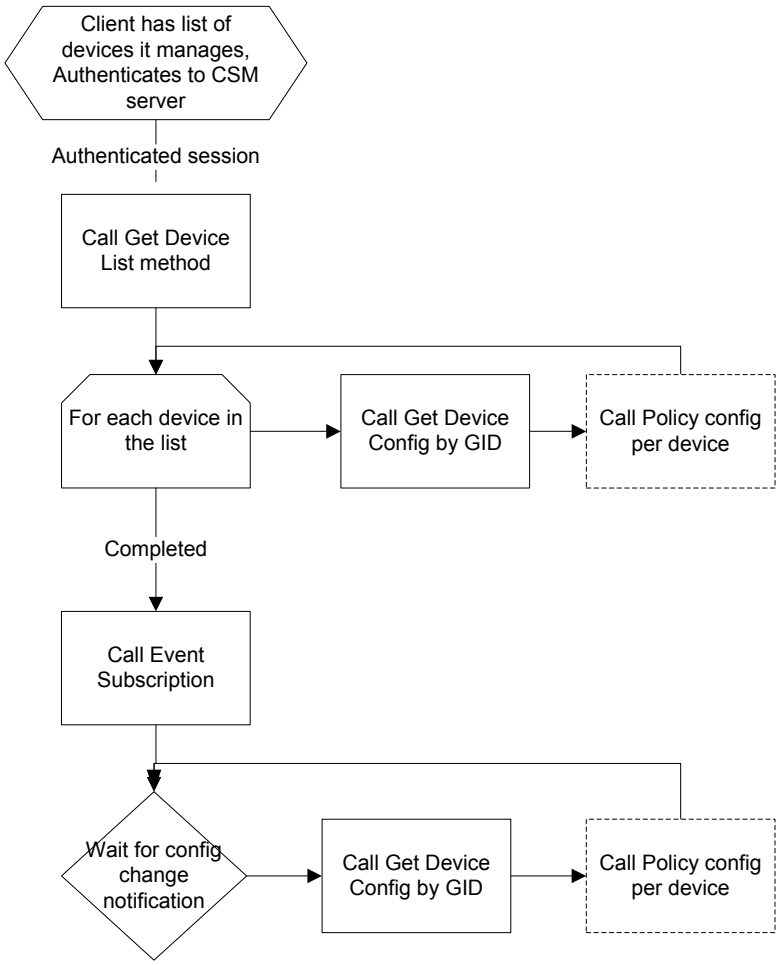


図 102：クライアント コンフィギュレーションのフロー チャート

未処理形式でポリシー データを処理する一般的なフローを以下に示します。

a. ログイン（およびバックグラウンド スレッドまたはプロセスをセットアップして ping またはハート ビートでセッションのアライブを保持）
b. システムのデバイス リストを取得するために GetGroupList（または）GetDeviceListByCapability（または）GetDeviceListByGroup の 1 つを使用します。
c. リスト内のデバイスそれぞれにつき
    i. RAW 設定データを取得するには GetDeviceConfigByGID（または）GetDeviceConfigByName を呼び出します。
d. Logout

設定が更新された場合にクライアントが設定をリフレッシュできるよう、クライアントは追加でイベントをサブスクライブする必要があります（このドキュメントのイベント サービス API を参照して

ください）。GetDeviceConfigByGID および GetDeviceConfigByName は、CSM データベース内にアーカイブされたデータのみを返します。CSM の外部デバイスへの更新（アウトオブバンド）は使用できません。Utility Service API を使用してのみ取得できます。

以下はポリシー オブジェクト モデル**の構造化形式**でポリシー データを処理する一般的なフローです。

a. ログイン（およびバックグラウンド スレッドまたはプロセスをセットアップして ping またはハート ビートでセッションのアライブを保持）
b. システムのデバイス リストを取得するために GetGroupList（または）GetDeviceListBy 機能（または）GetDeviceListByGroup の 1 つを使用します。
c. リスト内のデバイスそれぞれにつき
    i. GetPolicyListByDeviceGID を呼び出して、デバイスで設定またはサポートされている「ポリシー タイプ」を検出します。
    ii. これらの「ポリシー タイプ」ごとに GetPolicyConfigByDeviceGID または GetPolicyConfigByName を呼び出します（名前または共有ポリシーの場合）。
d. Logout

設定が更新された場合にクライアントが設定をリフレッシュできるよう、クライアントは追加でイベントをサブスクライブする必要があります（このドキュメントのイベント サービス API を参照してください）。GetDeviceConfigByGID および GetDeviceConfigByName は、CSM データベース内にアーカイブされたデータのみを返します。CSM の外部デバイスへの更新（アウトオブバンド）は使用できません。Utility Service API を使用してのみ取得できます。

8　サンプル API クライアント プログラム

注：このセクションのすべてのサンプル プログラムは単純なデモンストレーションの目的で提供されます。プログラムは実稼働環境のシステムでの使用に際し拡張または修正が必要な場合があります。

以下に記載する Java のサンプル プログラムの実行には以下をセットアップする必要があります。

1. Apache 共通 http://hc.apache.org からの **http-clien** jar および **http-core** jar は Java クラスパスに含める必要があります。
2. Apache 共通 http://commons.apache.org/logging/index.html から共通ロギング jar を Java クラスパスに含める必要があります。
3. 必要に応じて、必須 jar すべてとその他のフォルダを含むコマンド シェルに **CLASSPATH** 環境変数を定義します。
4. **client.properties** ファイルは Java プログラムの引数と同等のパラメータとして定義され、渡される必要があります。このファイルの形式は、次のようになります。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:pingRequest  xmlns:csm=\"csm\">\
        <protVersion>1.0</protVersion>\
        <reqId>3</reqId>\
</csm:pingRequest>

# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI=https://localhost/nbi/ping
```

プロパティ定義は次のとおりです。

- **USER**：ユーザのログイン用のユーザ名を定義します。
- **PASSWORD**：ユーザ パスワード
- **HOST**：接続されるホスト サーバ
- **XML_REQUEST**：送信する必要のある XML 要求（この場合は ping）
- **LOGIN_REQUIRED**：true の場合、ログインは XML_REQUEST を送信する前に実行されます。
- **URI**：呼び出される必要があるサービスを示すオプションの URI パラメータ。これがプロパティ ファイルで指定されていない場合は、Java プログラムの引数として渡す必要があります。

client.properties ファイルのプロパティ値（HOST、USER、PASSWORD など）、XML_REQUEST および URI 自体のその他の値は、展開および実行中のサンプルに合わせて変更する必要がある場合があることに注意してください。以下に記載したプログラムのいずれかを実行する前に、このファイルを適切に編集します。

8.1 CSM API 事前設定の確認

Java で実行される次のサンプル プログラムは、CSM API が使用可能になっているかどうかを確認する REST クライアント プログラムを示しています。上記で定義されている client.properties ファイルを使用します。

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
  * Sample Program to test if CSM server is correctly configured for API
 */
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

import java.io.FileInputStream;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }
```

```
    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
```

```
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
```

```
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.2 ログインおよび ping テスト

Java で実行される次の簡単なサンプル プログラムは、CSM API を使用して CSM サーバにログインし、「ping」を要求する REST クライアントを示します。上記で定義したとおり client.properties ファイルを使用します。

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
  * Sample Program to login to the CSM Server and send a ping request
 */
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.ParseException;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;

import java.io.FileInputStream;
import java.util.Properties;
```

```
public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param PayLoad
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
ClientProtocolException, IOException {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);
```

```
    }

    public void processResponse (HttpResponse httpresponse) throws ParseException,
IOException {
         HttpEntity ent = httpresponse.getEntity();
         String response = EntityUtils.toString(ent);

         if(response != null && response.trim().length() != 0){
              StatusLine sl;
              if ((sl = httpresponse.getStatusLine()) != null) {
                   if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                        System.out.println("Hit Authorization exception");
                        System.out.println(response);
                        //Do something...
                   }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                        System.out.println("The request is a success...");
                        System.out.println(response);
                        //Do something...
                   }else{
                        System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                        System.out.println(response);
                        //Do something...
                   }
              }
         }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
         try{
              //Load the basic properties
              FileInputStream fis = null;
              Properties prop = new Properties();
              fis = new FileInputStream(args[0]);
              prop.load(fis);

              String host = prop.getProperty("HOST");
              String payload = prop.getProperty("XML_REQUEST");
              String username = prop.getProperty("USER");
              String password = prop.getProperty("PASSWORD");
              String path = prop.getProperty("URI");
              //If URI is not passed on commandline see if its defined in properties file
              URI uri = new URI((args.length == 2)?args[1] : path);
              String temp = prop.getProperty("LOGIN_REQUIRED");
              boolean autoLogin = false;
              if(null != temp && temp.trim().length() != 0){
                   autoLogin = Boolean.valueOf(temp);
              }

              RestClient client = new RestClient();
              if(uri.toString().endsWith("login")){
                   client.doPost(uri, payload, host, false);
              }else{
                   //Step 1 :
                   if(autoLogin){
                        String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                        client.doPost(new URI("https://"+host+"/nbi/login"), login_payload,
host, false);
                   }
                   //Step 2:
                   client.doPost(uri, payload, host, true);
              }
         }catch(Exception ex){ex.printStackTrace(); usage();}
    }
```

```
    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.3　ファイアウォールのフェッチ CLI 設定

Java で実行される次の簡単なサンプル プログラムは、CSM データベースからファイアウォールの未処理の CLI 設定を読み込む際に CSM API を使用する REST クライアント プログラムを示します。入力として次の client.properties を使用します（サーバ インベントリのデバイス名に一致するようにデバイス名を変更する）。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"utf-8\"?>\
<n:deviceConfigByNameRequest xmlns:n=\"csm\"  xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <name>firewall_device</name>\
</n:deviceConfigByNameRequest>

# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI= https://localhost/nbi/configservice/getDeviceConfigByName
```

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample Program to get entire CLI of a firewall
 */
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
```

```
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;

import java.io.FileInputStream;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
```

```
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
```

```
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.4 ファイアウォール デバイスでの show access-list の実行

Java で実行される次の簡単なサンプル プログラムは、ファイアウォール デバイスで show access-list コマンドを実行する際に CSM API を使用する REST クライアントを示します。入力として次の client.properties を使用します（deviceIP を変更してサーバ インベントリの有効なデバイスの IP に一致させます）。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:execDeviceReadOnlyCLICmdsRequest  xmlns:csm=\"csm\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <deviceReadOnlyCLICmd>\
     <deviceIP>192.168.1.1</deviceIP>\
     <cmd>show</cmd>\
     <argument>access-list</argument>\
  </deviceReadOnlyCLICmd>\
</csm:execDeviceReadOnlyCLICmdsRequest>

# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
```

LOGIN_REQUIRED=true
URI=https://localhost/nbi/utilservice/execDeviceReadOnlyCLICmds

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample program to execute a show access-list command on a firewall
 */
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

import java.io.FileInputStream;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
```

```
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
```

```
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
```

```
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.5 フェッチ CSM によって定義されているファイアウォール ポリシー

Java で実行される次の簡単なサンプル プログラムは、CSM UI で定義される通り、CSM のファイアウォール ポリシーをフェッチする REST クライアントを示します。次の client.properties ファイルを使用します（サーバ インベントリのデバイスの GID に一致するように GID 値を変更してください）。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:policyConfigByDeviceGIDRequest  xmlns:csm=\"csm\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <gid>00000000-0000-0000-0000-004294967307</gid>\
 <policyType>DeviceAccessRuleFirewallPolicy</policyType>\
</csm:policyConfigByDeviceGIDRequest>

# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI=https://localhost/nbi/configservice/getPolicyConfigById
```

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample Program to get access rules defined on a firewall as it appears in the
 * CSM UI.
 */
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
```

```
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

import java.io.FileInputStream;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
```

```
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);
```

```
            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.6 すべてのデバイスに割り当てられている共有ポリシーのリスト

Java で実行される次の簡単なサンプル プログラムは、CSM のデバイス インベントリを繰り返し、すべてのデバイスに直接割り当てられている共有ポリシーをリストする REST クライアントを示します。次の client.properties ファイルを使用してください。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:deviceListByCapabilityRequest xmlns:csm=\"csm\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <deviceCapability>*</deviceCapability>\
</csm:deviceListByCapabilityRequest>
XML_REQUEST1=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:policyListByDeviceGIDRequest xmlns:csm=\"csm\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <gid>DEVICE_ID</gid>\
</csm:policyListByDeviceGIDRequest>
```

```
# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI=https://localhost/nbi/configservice/getDeviceListByType
URI1=https://localhost/nbi/configservice/getPolicyListByDeviceGID
```

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample program to collect the Shared Access Rules applied to Devices in CSM.
 */
import java.io.*;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

import java.util.ArrayList;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
```

```
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
```

```
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            String policy_req = prop.getProperty("XML_REQUEST1");
            String policy_req_path = prop.getProperty("URI1");

            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2: Get All Devices
                ArrayList devices = client.getDeviceList(uri, payload, host, true);

                //Step3: Get Shared Policies on Devices.
                ArrayList sharedAccessRules = new ArrayList();
                for (int i = 0; i < devices.size(); i++) {
                    String device = (String) devices.get(i);
                    String policy_request = policy_req.replace("DEVICE_ID", device);
                    ArrayList policies = client.getPolicyList(new URI(policy_req_path),
policy_request, host, true);
                    if(policies!= null) {
                        sharedAccessRules.addAll(policies);
                    }
                    System.out.println(sharedAccessRules);
                }
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }
```

```
    private ArrayList getPolicyList(URI uri, String payload, String host, boolean
isCookieNeeded) throws Exception {
        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        //processResponse(httpresponse);
        return getPolicyList(httpresponse);
    }

    private ArrayList getPolicyList(HttpResponse httpresponse) throws IOException,
SAXException, ParserConfigurationException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        System.out.println(response);
        ArrayList<String> retArr = new ArrayList();
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    DocumentBuilder domp =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
                            Document doc = domp.parse(new
ByteArrayInputStream(response.getBytes()));

                    NodeList errorNodes =
doc.getDocumentElement().getElementsByTagName("policyDesc");
                    for (int i = 0; i < errorNodes.getLength(); i++) {
                        Element element = (Element) errorNodes.item(i);
                        String text = element.getTextContent();
                        if(text != null && !text.equals("")&&
text.indexOf("DeviceAccessRuleFirewallPolicy")>=0) {
                            text = text.replaceAll("DeviceAccessRuleFirewallPolicy", "");
                            if(!text.equals("-- local --"))// Filter out Local rules.
                                retArr.add(text);
                        }
                    }

                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
        return retArr;
    }

    private ArrayList getDeviceList(URI uri, String payload, String host, boolean
isCookieNeeded) throws Exception {
        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
```

```
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        //processResponse(httpresponse);
        return getDeviceList(httpresponse);
    }

    private ArrayList getDeviceList(HttpResponse httpresponse) throws IOException,
SAXException, ParserConfigurationException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        ArrayList<String> retArr = new ArrayList();
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    DocumentBuilder domp =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
                            Document doc = domp.parse(new
ByteArrayInputStream(response.getBytes()));

                    NodeList errorNodes =
doc.getDocumentElement().getElementsByTagName("gid");
                    for (int i = 0; i < errorNodes.getLength(); i++) {
                        Element element = (Element) errorNodes.item(i);
                        if(element.getTextContent() != null
&& !element.getTextContent().equals("")) {
                            retArr.add(element.getTextContent());
                        }
                    }

                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
        return retArr;
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.7 特定の共有ポリシーのリストの内容

Java で実行される次の簡単なサンプル プログラムは、REST クライアントがファイアウォール アクセス ルール共有ポリシーの名前を指定してその内容をリストする方法を示します。次に示すように client.properties ファイルを使用します（プロパティ ファイルで適切に定義された変更の共有ポリシーの名前）。

```
USER=admin
```

```
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:policyConfigByNameRequest  xmlns:csm=\"csm\">\
  <protVersion>1.0</protVersion>\
  <reqId>123</reqId>\
  <name>ACL1</name>\
  <policyType>DeviceAccessRuleFirewallPolicy</policyType>\
</csm:policyConfigByNameRequest>
# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI=https://localhost/nbi/configservice/getPolicyConfigByName
```

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample program to get the contents of a shared policy given the shared policy name.
 */
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

import java.io.FileInputStream;
import java.util.Properties;

public class RestClient {

    public static CookieStore ascookie = null;
```

```
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());


            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
```

```
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
```

```
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

8.8 変更通知のサブスクライブ：展開、OOB

Java で実行される次の簡単なサンプル プログラムは、REST クライアントが CSM からの変更通知を登録し、展開および Out of Band （OOB）イベントを受け取る方法を示します。変更通知を受信するには、クライアントは、**CSM が「非同期」通知を送信する特定の syslog サービス（IP およびポート）**を登録することに注意してください。実際の通知を確認するには、オープン ソースの Syslog サーバを使用する、または指定された IP/Port で簡易 UDP リスナーを実行して通知の内容をリストします（次の SyslogServer 要素を参照してください）。

次のように client.properties ファイルを使用してください。

```
USER=admin
PASSWORD=admin
HOST=localhost
XML_REQUEST=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\
<csm:eventSubRequest xmlns:csm=\"csm\">\
  <op>add</op>\
  <subscriptionId>123454</subscriptionId>\
  <eventFilterItem>\
  <filterEventType>syslog</filterEventType>\
  <filterEventFormat>xml</filterEventFormat>\
  <filterEventCategory>configChange</filterEventCategory>\
  </eventFilterItem>\
  <syslogServer>\
     <port>514</port>\
     <destAddress>10.10.10.10</destAddress>\
  </syslogServer>\
</csm:eventSubRequest>

# Set LOGIN_REQUIRED to true if the URI supplied
# requires login to be done as a prerequisite.
LOGIN_REQUIRED=true
URI=https://localhost/nbi/eventservice/eventSubscription
```

コンパイルの後、プログラムを実行するには、次のコマンドを使用します。

```
Command Prompt> java RestClient <path_to_client.properties> [<uri>]
```

クラス RestClient.java

```
/**
 * Sample program to subscribe to change events and get more details on latest change.
 */
import java.io.ByteArrayInputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.net.URI;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.X509Certificate;
import java.util.Properties;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.StatusLine;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.CookieStore;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.conn.ClientConnectionManager;
import org.apache.http.conn.scheme.Scheme;
import org.apache.http.conn.scheme.SchemeRegistry;
import org.apache.http.conn.ssl.SSLSocketFactory;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpParams;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
import org.xml.sax.SAXException;


public class RestClient {

    public static CookieStore ascookie = null;
    public static DefaultHttpClient httpclient;

    static{
        initSSL();
    }


    private static void initSSL() {
        SSLContext sslContext = null;
        try {
            sslContext = SSLContext.getInstance("SSL");
            sslContext.init(null, new TrustManager[] { new X509TrustManager() {
                public X509Certificate[] getAcceptedIssuers() {
                    System.out.println("getAcceptedIssuers =============");
                    return null;
                }
                public void checkClientTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkClientTrusted =============");
                }
                public void checkServerTrusted(X509Certificate[] certs, String authType) {
                    System.out.println("checkServerTrusted =============");
                }
            }}, new SecureRandom());
```

```
            SSLSocketFactory sf = new SSLSocketFactory(sslContext);

            Scheme httpsScheme = new Scheme("https", 443, sf);
            SchemeRegistry schemeRegistry = new SchemeRegistry();
            schemeRegistry.register(httpsScheme);
            HttpParams params = new BasicHttpParams();
            ClientConnectionManager cm = new ThreadSafeClientConnManager(params,
schemeRegistry);
            httpclient = new DefaultHttpClient(cm, params);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        } catch (KeyManagementException e) {
            e.printStackTrace();
        }
    }

    /**
     * This method will send the XML payload and return the XML response as a string.
     * @param uri
     * @param host
     * @param isCookieNeeded
     * @return
     * @throws IOException
     * @throws ClientProtocolException
     */
    public void doPost (URI uri, String payload, String host, boolean isCookieNeeded) throws
Exception {

        HttpResponse httpresponse = null;
        HttpPost httppost = new HttpPost (uri);

        if (isCookieNeeded) {
            httpclient.setCookieStore(ascookie);
        }
        httppost.addHeader("Content-Type", "text/xml");
        StringEntity strEntity = new StringEntity (payload, "UTF-8");
        httppost.setEntity(strEntity);
        System.out.println("Calling : "+uri.toString());
        httpresponse = httpclient.execute(httppost);
        ascookie = httpclient.getCookieStore();
        processResponse(httpresponse);

    }

    public void processResponse (HttpResponse httpresponse) throws Exception, IOException,
SAXException {
        HttpEntity ent = httpresponse.getEntity();
        String response = EntityUtils.toString(ent);
        DocumentBuilder domp = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = domp.parse(new ByteArrayInputStream(response.getBytes()));

        NodeList errorNodes = doc.getDocumentElement().getElementsByTagName("code");
        for (int i = 0; i < errorNodes.getLength(); i++) {
            Element element = (Element) errorNodes.item(i);
            if(element.getTextContent() != null && !element.getTextContent().equals("")) {
                NodeList nodes = doc.getDocumentElement().getElementsByTagName("description");
                for (int j = 0; j < nodes.getLength(); j++) {
                    Element element2 = (Element) nodes.item(j);
                    throw new Exception(element2.getTextContent());
                }
            }
        }
        if(response != null && response.trim().length() != 0){
            StatusLine sl;
            if ((sl = httpresponse.getStatusLine()) != null) {
                if (sl.getStatusCode() == HttpStatus.SC_UNAUTHORIZED) {
                    System.out.println("Hit Authorization exception");
                    System.out.println(response);
                    //Do something...
                }else if (sl.getStatusCode() == HttpStatus.SC_OK) {
```

```
                    System.out.println("The request is a success...");
                    System.out.println(response);
                    //Do something...
                }else{
                    System.out.println("Some issue, Obtained HTTP status
code :"+sl.getStatusCode());
                    System.out.println(response);
                    //Do something...
                }
            }
        }
    }

    /**
     * Main method processing the request/response
     * @param args
     */
    public static void main(String[] args){
        try{
            //Load the basic properties
            FileInputStream fis = null;
            Properties prop = new Properties();
            fis = new FileInputStream(args[0]);
            prop.load(fis);

            String host = prop.getProperty("HOST");
            String payload = prop.getProperty("XML_REQUEST");
            String username = prop.getProperty("USER");
            String password = prop.getProperty("PASSWORD");
            String path = prop.getProperty("URI");
            //If URI is not passed on commandline see if its defined in properties file
            URI uri = new URI((args.length == 2)?args[1] : path);
            String temp = prop.getProperty("LOGIN_REQUIRED");
            boolean autoLogin = false;
            if(null != temp && temp.trim().length() != 0){
                autoLogin = Boolean.valueOf(temp);
            }

            RestClient client = new RestClient();
            if(uri.toString().endsWith("login")){
                client.doPost(uri, payload, host, false);
            }else{
                //Step 1 :
                if(autoLogin){
                    String login_payload = "<?xml version=\"1.0\" encoding=\"UTF-
8\"?><csm:loginRequest
xmlns:csm=\"csm\"><protVersion>1.0</protVersion><reqId>123</reqId><username>"+username+"</user
name><password>"+password+"</password></csm:loginRequest>";
                    client.doPost(new URI("https://"+host+"/nbi/login"), login_payload, host,
false);
                }
                //Step 2:
                client.doPost(uri, payload, host, true);
            }
        }catch(Exception ex){
            System.out.println(ex.getMessage()); usage();}
    }

    public static void usage(){
        System.out.println("Please check the data entered in the properties file");
        System.out.println("Usage : ");
        System.out.println("java RestClient <path_to_client.properties> [<uri>]");
    }
}
```

9　トラブルシューティング（共通シナリオ）

症状：サーバと通信する際の証明書エラー

CSM サーバは自己署名証明書を使用します。自己署名証明書を受け入れるようにクライアント プログラムを変更します。また API アクセスは HTTPS 経由でのみ可能で、セキュリティ上の理由から HTTP アクセスはディセーブルです。

症状：ログイン成功後も認証エラー

セッションがタイム アウトになるか無効になった場合に、断続セッションエラーが表示されます。デフォルトのセッションの非アクティブ タイムアウトは 15 分です。セッションは ping またはハートビート機能を使用してアライブ状態を保持できます。

症状：CSM GUI で入力されるインライン IP がオブジェクトとして取得される

便宜上、CSM はポリシー オブジェクト内の特定のインライン値（IP、インターフェイス名など）を自動的にカプセル化します。

症状：API 経由で取得した場合、CSM クライアント UI に表示されるデータが、応答データに表示されない

これは、さまざまな理由が原因です。そのうちのいくつかを次にリストします。

- API はコミット済みのデータのみを返します（つまりすべての変更を送信する必要があります）。このため、クライアントに表示されているすべてのデータがコミットされたかどうかを確認します。
- API アクセスに関連しない可能性があるため、CSM 特有の設定が応答で返されないことがあります。
- データにアクセスするのに十分な Role Based Access Control（RBAC）特権をユーザが持っているかどうかを確認します。

症状：クライアントがサーバからイベント通知を受信しない

以下はこの問題の原因の一部です。

- イベント サブスクリプションが実施されていない、または通知を受信する不正な syslog サーバで登録されています。
- イベント サブスクリプションの登録に使用されるセッションがタイム アウトになりました。すべてのイベント通知がユーザ セッションに関連付けられます。ユーザ セッションが無効になると、そのユーザ セッションのすべてのイベント通知が停止します。

10 XML スキーマ

XML スキーマは 4 個のファイルに分割されます。

10.1 Common XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="csm" targetNamespace="csm">
	<xs:simpleType name="ObjectIdentifier">
		<xs:restriction base="xs:string">
			<xs:pattern value="[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:complexType name="ObjectIdentifierList">
		<xs:sequence>
			<xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="BaseObject">
		<xs:sequence>
			<xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
			<xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="lastUpdateTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
			<xs:element name="parentGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
			<xs:element name="updatedByUser" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="lastCommitTime" type="xs:dateTime" minOccurs="0" maxOccurs="1"/>
			<xs:element name="ticketId" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="activityName" type="xs:string" minOccurs="0" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="BaseError">
		<xs:sequence>
			<xs:element name="code" type="xs:unsignedLong" maxOccurs="1"/>
			<xs:element name="description" type="xs:string" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="BaseReqResp">
		<xs:sequence>
			<xs:element name="protVersion" type="xs:double" minOccurs="0" maxOccurs="1"/>
			<xs:element name="reqId" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="startIndex" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
			<xs:element name="endIndex" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
			<xs:element name="totalCount" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
			<xs:element name="error" type="BaseError" minOccurs="0" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="EntityDescriptor">
		<xs:sequence>
			<xs:element name="name" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="device" type="Device"/>
	<xs:complexType name="Device">
		<xs:complexContent>
			<xs:extension base="BaseObject">
				<xs:sequence>
					<xs:element name="osType" type="OSType" minOccurs="1" maxOccurs="1"/>
					<xs:element name="osVersion" type="xs:string" minOccurs="1" maxOccurs="1"/>
					<xs:element name="imageName" type="xs:string" minOccurs="1" maxOccurs="1"/>
					<xs:element name="sysObjectID" type="xs:string" minOccurs="1" maxOccurs="1"/>
					<xs:element name="fullConfig" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="mgmtInterface" type="Interface" minOccurs="0" maxOccurs="1"/>
					<xs:element name="interfaceList" type="InterfaceList" minOccurs="0" maxOccurs="1"/>
					<xs:element name="virtualContextList" type="Device" minOccurs="0"
maxOccurs="unbounded"/>
```

```
                        <xs:element name="configState" type="ConfigurationState" minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="PortIdentifier">
            <xs:sequence>
                <!-- for non-modular chassis or chassis with a continuous port numbering scheme slot/module are not
included -->
                <xs:element name="slotNum" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                <xs:element name="moduleNum" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                <xs:element name="portNum" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <xs:simpleType name="ProtocolPort">
            <xs:restriction base="xs:string"/>
        </xs:simpleType>
        <xs:complexType name="InterfaceList">
            <xs:sequence>
                <xs:element name="interface" type="Interface" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="Interface">
            <xs:sequence>
                <xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="identifier" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="ipInterface" type="IPInterfaceAttrs" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="macInterface" type="MACInterfaceAttrs" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="MACInterfaceAttrs">
            <xs:sequence>
                <xs:element name="macAddress" type="xs:string" minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="IPInterfaceAttrs">
            <xs:sequence>
                <xs:element name="domainName" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="ipAddress" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="isNatAddress" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                <xs:element name="realIpAddress" type="xs:string" minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <xs:simpleType name="OSType">
            <xs:restriction base="xs:token">
                <xs:enumeration value="ios"/>
                <xs:enumeration value="fwsm"/>
                <xs:enumeration value="asa"/>
                <xs:enumeration value="ips"/>
                <xs:enumeration value="pix"/>
                <xs:enumeration value="undefined"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="IPTransportProtocol">
            <xs:restriction base="xs:token">
                <xs:enumeration value="TCP"/>
                <xs:enumeration value="UDP"/>
                <xs:enumeration value="IP"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="ConfigurationState">
            <xs:restriction base="xs:token">
                <xs:enumeration value="undefined"/>
                <xs:enumeration value="committed"/>
                <xs:enumeration value="deployed"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="DeviceCapability">
            <xs:restriction base="xs:token">
                <xs:enumeration value="firewall"/>
                <xs:enumeration value="ids"/>
```

```
                <xs:enumeration value="router"/>
                <xs:enumeration value="switch"/>
                <xs:enumeration value="*"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:complexType name="DeviceGroup">
            <xs:complexContent>
                <xs:extension base="BaseObject">
                    <xs:sequence>
                        <xs:element name="path" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="device" type="Device" minOccurs="0" maxOccurs="unbounded"/>
                        <xs:element name="deviceGroup" type="DeviceGroup" minOccurs="0"
maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="DeviceGroupPath">
            <xs:sequence>
                <xs:element name="pathItem" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:simpleType name="SubscriptionOperation">
            <xs:restriction base="xs:string">
                <xs:enumeration value="add"/>
                <xs:enumeration value="delete"/>
            </xs:restriction>
        </xs:simpleType>
        <!-- Common Service Methods
    -->
        <!-- Generic error -->
        <xs:element name="baseError" type="BaseError"/>
        <xs:element name="loginRequest" type="LoginRequest"/>
        <xs:complexType name="LoginRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence>
                        <xs:element name="username" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="password" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="heartbeatRequested" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="callbackUrl" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="loginResponse" type="LoginResponse"/>
        <xs:complexType name="LoginResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence>
                        <xs:element name="serviceVersion" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="sessionTimeoutInMins" type="xs:positiveInteger" minOccurs="1"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="heartbeatCallbackRequest" type="HeartbeatCallbackRequest"/>
        <xs:complexType name="HeartbeatCallbackRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="logoutRequest" type="LogoutRequest"/>
        <xs:complexType name="LogoutRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="logoutResponse" type="LogoutResponse"/>
        <xs:complexType name="LogoutResponse">
```

```
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="pingRequest" type="PingRequest"/>
        <xs:complexType name="PingRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="pingResponse" type="PingResponse"/>
        <xs:complexType name="PingResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="getServiceInfoRequest" type="GetServiceInfoRequest"/>
        <xs:complexType name="GetServiceInfoRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp"/>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="getServiceInfoResponse" type="GetServiceInfoResponse"/>
        <xs:complexType name="GetServiceInfoResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence>
                        <xs:element name="serviceVersion" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="serviceName" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="serviceDesc" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
</xs:schema>
```

10.2 Config XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="csm" targetNamespace="csm">
	<xs:include schemaLocation="common.xsd"/>
	<xs:complexType name="BasePolicy" >
		<xs:complexContent>
			<xs:extension base="BaseObject">
				<xs:sequence minOccurs="1" maxOccurs="1">
					<xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
					<xs:element name="orderId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
					<xs:element name="isMandatoryAggregation" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
					<xs:element name="description" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="configState" type="ConfigurationState" minOccurs="0" maxOccurs="1"/>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="BasePolicyObject">
		<xs:complexContent>
			<xs:extension base="BaseObject">
				<xs:sequence minOccurs="1" maxOccurs="1">
					<xs:element name="type" type="xs:string" minOccurs="1" maxOccurs="1"/>
					<xs:element name="comment" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="nodeGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
					<xs:element name="isProperty" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
					<xs:element name="subType" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="isGroup" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
					<xs:element name="refGIDs" type="ObjectIdentifierList" minOccurs="0" maxOccurs="1"/>
					<xs:element name="configState" type="ConfigurationState" minOccurs="0" maxOccurs="1"/>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<xs:complexType name="NetworkInterfaceObjectsRefs">
		<xs:sequence>
			<xs:element name="networkObjectGIDs" type="ObjectIdentifierList" minOccurs="0" maxOccurs="1"/>
			<xs:element name="interfaceRoleObjectGIDs" type="ObjectIdentifierList" minOccurs="0" maxOccurs="1"/>
			<xs:choice>
			    <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
			    <xs:element name="ipData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
			</xs:choice>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="NetworkObjectsRefs">
		<xs:sequence>
			<xs:element name="networkObjectGIDs" type="ObjectIdentifierList" minOccurs="0" maxOccurs="1"/>
		 <xs:choice>
			 <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
			 <xs:element name="ipData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
		 </xs:choice>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="SecurityGrpObjectsRef">
	    <xs:sequence>
	      <xs:choice>
	        <xs:element name="securityGrpObjectGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
	        <xs:element name="secName" type="xs:string" minOccurs="0" maxOccurs="1"/>
	        <xs:element name="secTag" type="xs:string" minOccurs="0" maxOccurs="1"/>
	      </xs:choice>
	    </xs:sequence>
	</xs:complexType>
	<xs:complexType name="SecurityGrpObjectsRefs">
	    <xs:sequence>
	      <xs:element name="securityTag" type="SecurityGrpObjectsRef" minOccurs="1" maxOccurs="unbounded"/>
	    </xs:sequence>
	</xs:complexType>
	<xs:complexType name="IdentityUserGrpObjectsRefs">
```

```
            <xs:sequence>
                <xs:element name="identityUserGrpObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
                <xs:element name="userNameData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="userGroupData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="NetworkObjectRefs">
            <xs:sequence>
                <xs:element name="networkObjectGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
                <xs:choice>
                  <xs:element name="ipv4Data" type="xs:string" minOccurs="0" maxOccurs="1"/>
                  <xs:element name="ipData" type="xs:string" minOccurs="0" maxOccurs="1"/>
                </xs:choice>
                <xs:element name="interfaceKeyword" type="xs:string" fixed="interface" minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="NetworkOrIPRef">
            <xs:choice>
                <xs:element name="hostOrNetworkObjectGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                <xs:choice>
                  <xs:element name="ipv4Data" type="xs:string" minOccurs="1" maxOccurs="1"/>
                  <xs:element name="ipData" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:choice>
            </xs:choice>
        </xs:complexType>
        <xs:complexType name="NetworkPolicyObject">
            <xs:complexContent>
                <xs:extension base="BasePolicyObject">
                    <xs:sequence>
                            <xs:choice>
                                <xs:element name="ipv4Data" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                                <xs:element name="ipData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                            </xs:choice>                                <xs:element name="fqdnData" minOccurs="0"
maxOccurs="1">
                              <xs:complexType>
                              <xs:sequence>
                                        <xs:element name="value" type="xs:string" minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="isIPv4Only" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                              </xs:sequence>
                              </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="IdentityUserGroupPolicyObject">
            <xs:complexContent>
                <xs:extension base="BasePolicyObject">
                    <xs:sequence>
                        <xs:element name="userNameData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
                        <xs:element name="userGroupData" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="SecurityGroupPolicyObject">
          <xs:complexContent>
            <xs:extension base="BasePolicyObject">
              <xs:sequence>
                <xs:element name="securityTag" type="SecurityGrpObjectsRef" minOccurs="1" maxOccurs="unbounded"/>
              </xs:sequence>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="PortListPolicyObject">
            <xs:complexContent>
                <xs:extension base="BasePolicyObject">
```

```
                        <xs:sequence minOccurs="0" maxOccurs="unbounded">
                            <xs:element name="startPort" type="PortIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="endPort" type="PortIdentifier" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="ServiceParameters">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="protocol" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="sourcePort" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="port" type="ProtocolPort"/>
                                <xs:element name="portRefGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="destinationPort" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="port" type="ProtocolPort"/>
                                <xs:element name="portRefGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="icmpMessage" type="xs:string" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:complexType>
            <xs:complexType name="ServicePolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="0" maxOccurs="1">
                            <xs:element name="serviceParameters" type="ServiceParameters" minOccurs="1"
maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="InterfaceRolePolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="pattern" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="TimeRangePolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="startTime" type="xs:dateTime" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="endTime" type="xs:dateTime" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="recurrence" minOccurs="0" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:choice>
                                        <xs:element name="dayOfWeekInterval">
                                            <xs:complexType>
                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                    <xs:element name="dayOfWeek" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                                    <xs:element name="startTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                                    <xs:element name="endTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                                </xs:sequence>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="weeklyInterval">
                                            <xs:complexType>
```

```
                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                    <xs:element name="startDay" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                                    <xs:element name="startTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                                    <xs:element name="endDay" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                                    <xs:element name="endTime" type="xs:time"
minOccurs="1" maxOccurs="1"/>
                                                </xs:sequence>
                                            </xs:complexType>
                                        </xs:element>
                                    </xs:choice>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="SLAMonitorPolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="slaId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="monitoredAddress" type="xs:string" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="dataSizeInBytes" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="thresholdInMilliSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                            <xs:element name="timeoutInMilliSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                            <xs:element name="frequencyInSeconds" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                            <xs:element name="toS" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="numberOfPackets" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="StandardACEPolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="networkGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="doLogging" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="permit" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="ExtendedACEPolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence>
                            <xs:element name="sourceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="destinationGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="serviceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="doLogging" type="xs:string" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="logInterval" type="xs:string" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="logLevel" type="xs:string" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="logOption" type="xs:string" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="permit" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="ACLPolicyObject">
                <xs:complexContent>
                    <xs:extension base="BasePolicyObject">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="references" minOccurs="1" maxOccurs="unbounded">
```

```
                                        <xs:complexType>
                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                        <xs:element name="sequenceNumber" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                                                        <xs:choice>
                                                                <xs:element name="aclObjectReferenceGID" type="ObjectIdentifier"/>
                                                                <xs:element name="aceReferenceGID" type="ObjectIdentifier"/>
                                                        </xs:choice>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                        </xs:sequence>
                </xs:extension>
        </xs:complexContent>
</xs:complexType>
<xs:complexType name="DeviceAccessRuleFirewallPolicy">
        <xs:complexContent>
                <xs:extension base="BasePolicy">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                                <xs:element name="isEnabled" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                                <xs:element name="direction" minOccurs="0" maxOccurs="1">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:enumeration value="In"/>
                                                        <xs:enumeration value="Out"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
                                <xs:element name="permit" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                                <xs:element name="interfaceRoleObjectGIDs" type="ObjectIdentifierList" minOccurs="0"
maxOccurs="1"/>
                                <xs:element name="users" type="IdentityUserGrpObjectsRefs" minOccurs="0" maxOccurs="1"/>
                                <xs:element name="sources" type="NetworkInterfaceObjectsRefs" minOccurs="0"
maxOccurs="1"/>
                                <xs:element name="destinations" type="NetworkInterfaceObjectsRefs" minOccurs="0"
maxOccurs="1"/>
                                <xs:element name="services">
                                        <xs:complexType>
                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                        <xs:element name="serviceObjectGIDs" type="ObjectIdentifierList"
minOccurs="0" maxOccurs="1"/>
                                                        <xs:element name="serviceParameters" type="ServiceParameters"
minOccurs="0" maxOccurs="unbounded"/>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                <xs:element name="logOptions" minOccurs="0" maxOccurs="1">
                                        <xs:complexType>
                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                        <xs:element name="isFirewallLoggingEnabled" type="xs:boolean"
minOccurs="0" maxOccurs="1"/>
                                                        <xs:choice>
                                                                <xs:element name="isDefaultLogging" type="xs:boolean"
minOccurs="0" maxOccurs="1"/>
                                                                <xs:sequence minOccurs="1" maxOccurs="1">
                                                                        <xs:element name="loggingInterval" type="xs:unsignedInt"
minOccurs="0" maxOccurs="1"/>
                                                                        <xs:element name="loggingLevel" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                                                </xs:sequence>
                                                        </xs:choice>
                                                        <xs:element name="isIOSLoggingEnabled" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                                                        <xs:element name="isLogInput" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                <xs:element name="iosOptions" minOccurs="0" maxOccurs="1">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
```

```
                                        <xs:enumeration value="None"/>
                                        <xs:enumeration value="Established"/>
                                        <xs:enumeration value="Fragment"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="timeRangeObjectGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="DeviceAccessRuleUnifiedFirewallPolicy">
              <xs:complexContent>
                <xs:extension base="DeviceAccessRuleFirewallPolicy">
                  <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="sourceSG" type="SecurityGrpObjectsRefs" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="destinationSG" type="SecurityGrpObjectsRefs" minOccurs="0" maxOccurs="1"/>
                  </xs:sequence>
                </xs:extension>
              </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="DeviceStaticRoutingFirewallPolicy">
                <xs:complexContent>
                    <xs:extension base="BasePolicy">
                        <xs:sequence>
                            <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="networks" type="NetworkObjectsRefs" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="gateway" type="NetworkObjectRefs" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="metric" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="tunnelled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="slaMonitorGID" type="ObjectIdentifier" minOccurs="0" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="DeviceStaticRoutingRouterPolicy">
                <xs:complexContent>
                    <xs:extension base="BasePolicy">
                        <xs:sequence>
                            <xs:element name="destinationNetwork" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:choice>
                                        <xs:element name="useAsDefaultRoute" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="prefix" type="NetworkOrIPRef" minOccurs="1"
maxOccurs="1"/>
                                    </xs:choice>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="fowarding" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:choice>
                                        <xs:element name="forwardingInterfaceGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="forwardingIPAddress" type="NetworkOrIPRef"
minOccurs="1" maxOccurs="1"/>
                                    </xs:choice>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="distanceMetric" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="isPermanentRoute" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="DeviceBGPRouterPolicy">
                <xs:complexContent>
                    <xs:extension base="BasePolicy">
                        <xs:sequence>
                            <xs:element name="asNumber" type="xs:unsignedLong" minOccurs="1" maxOccurs="1"/>
```

```
                        <xs:element name="networks" type="NetworkObjectsRefs" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="neighbors" minOccurs="1" maxOccurs="1">
                              <xs:complexType>
                                    <xs:sequence minOccurs="1" maxOccurs="unbounded">
                                          <xs:element name="ipAddress" type="NetworkObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                                          <xs:element name="asNumber" type="xs:unsignedLong" minOccurs="1"
maxOccurs="1"/>
                                    </xs:sequence>
                              </xs:complexType>
                        </xs:element>
                        <xs:element name="autoSummary" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="synchronization" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="logNeighbor" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="redistributionEntry" minOccurs="0" maxOccurs="unbounded">
                              <xs:complexType>
                                    <xs:sequence>
                                          <xs:element name="protocol" minOccurs="1" maxOccurs="1">
                                                <xs:complexType>
                                                      <xs:choice>
                                                            <xs:element name="static" minOccurs="1" maxOccurs="1">
                                                                  <xs:simpleType>
                                                                        <xs:restriction base="xs:string">
                                                                              <xs:enumeration value="IP"/>
                                                                              <xs:enumeration value="OSI"/>
                                                                        </xs:restriction>
                                                                  </xs:simpleType>
                                                            </xs:element>
                                                            <xs:element name="connected" type="xs:string"
minOccurs="1" maxOccurs="1"/>
                                                            <xs:element name="rip" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                                            <xs:element name="eigrp" minOccurs="1" maxOccurs="1">
                                                                  <xs:complexType>
                                                                        <xs:sequence>
                                                                              <xs:element name="asNumber"
type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                                                                        </xs:sequence>
                                                                  </xs:complexType>
                                                            </xs:element>
                                                            <xs:element name="ospf" minOccurs="1" maxOccurs="1">
                                                                  <xs:complexType>
                                                                        <xs:sequence>
                                                                              <xs:element name="processId"
type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                                                                              <xs:element name="match"
minOccurs="0" maxOccurs="unbounded">
                                                                                    <xs:simpleType>
                                                                                          <xs:restriction
base="xs:string">
                                                                                                <xs:enumeration
value="Internal"/>
                                                                                                <xs:enumeration
value="External1"/>
                                                                                                <xs:enumeration
value="External2"/>
                                                                                                <xs:enumeration
value="NSSAExternal1"/>
                                                                                                <xs:enumeration
value="NSSAExternal2"/>
                                                                                          </xs:restriction>
                                                                                    </xs:simpleType>
                                                                              </xs:element>
                                                                        </xs:sequence>
                                                                  </xs:complexType>
                                                            </xs:element>
                                                      </xs:choice>
                                                </xs:complexType>
                                          </xs:element>
                                          <xs:element name="metric" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
```

```
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:extension>
                    </xs:complexContent>
                </xs:complexType>
                <xs:complexType name="InterfaceNATRouterPolicy">
                    <xs:complexContent>
                        <xs:extension base="BasePolicy">
                            <xs:sequence>
                                <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="isNatInside" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            </xs:sequence>
                        </xs:extension>
                    </xs:complexContent>
                </xs:complexType>
                <xs:complexType name="InterfaceNATStaticRulesRouterPolicy">
                    <xs:complexContent>
                        <xs:extension base="BasePolicy">
                            <xs:sequence>
                                <xs:element name="staticRuleType" minOccurs="1" maxOccurs="1">
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:enumeration value="Static Host"/>
                                            <xs:enumeration value="Static Network"/>
                                            <xs:enumeration value="Static Port"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
                                <xs:element name="original" type="NetworkOrIPRef" minOccurs="1" maxOccurs="1"/>
                                <xs:element name="translated" minOccurs="1" maxOccurs="1">
                                    <xs:complexType>
                                        <xs:choice>
                                            <xs:element name="originalIP" type="NetworkOrIPRef" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                        </xs:choice>
                                    </xs:complexType>
                                </xs:element>
                                <xs:element name="portRedirection" minOccurs="0" maxOccurs="1">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="protocol" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="localPort" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="globalPort" type="xs:unsignedInt" minOccurs="1"
maxOccurs="1"/>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                                <xs:element name="settings" minOccurs="0" maxOccurs="1">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="noAlias" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="noPayload" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                            <xs:element name="createExtTransEntry" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:extension>
                    </xs:complexContent>
                </xs:complexType>
                <xs:complexType name="InterfaceNATDynamicRulesRouterPolicy">
                    <xs:complexContent>
```

```
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="trafficFlowAclObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="translated" minOccurs="1" maxOccurs="1">
                            <xs:complexType>
                                <xs:choice>
                                    <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="addressPool" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
                                </xs:choice>
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="settings" minOccurs="0" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="enablePortTrans" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="noTransVPN" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="DeviceNATTimeoutsRouterPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="maxEntriesInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="timeoutInSecs" type="xs:unsignedLong" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="udpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="dnsTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="tcpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="finRstTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="icmpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="pptpTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="synTimeoutInSecs" type="xs:unsignedLong" minOccurs="0"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!--
    Firewall NAT
 -->
        <!--  Reusable Firewall advanced options -->
        <xs:complexType name="FirewallNATAdvancedOptions">
            <xs:sequence>
                <xs:element name="isTransDNSReplies" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                <xs:element name="maxTCPConnPerRule" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                <xs:element name="maxUDPConnPerRule" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                <xs:element name="maxEmbConnections" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                <xs:element name="randomizeSeqNum" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <!-- Reusable NAT Type -->
        <xs:simpleType name="NATType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Static"/>
                <xs:enumeration value="Dynamic"/>
```

```
            </xs:restriction>
        </xs:simpleType>
        <!-- Reusable Protocol Type -->
        <xs:complexType name="InterfaceNATAddressPoolFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="interfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="ipAddressRange" type="NetworkObjectsRefs" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="interfaceKeyword" type="xs:string" fixed="interface" minOccurs="0"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="DeviceNATTransOptionsFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="isEnableTrafficWithoutTrans" type="xs:boolean" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="isXlateByPass" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNATTransExemptionsFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="isExempt" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="original" type="NetworkInterfaceObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="destinations" type="NetworkInterfaceObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="fwsmAdvancedOptions" type="FirewallNATAdvancedOptions"
minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNATDynamicRulesFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="original" type="NetworkObjectsRefs" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions" minOccurs="0"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNATPolicyDynamicRulesFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="poolId" type="xs:unsignedInt" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="original" type="NetworkInterfaceObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="outsideNAT" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
```

```
                        <xs:element name="destinations" type="NetworkInterfaceObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="services" minOccurs="1" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="serviceData" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                                    <xs:element name="serviceObjectGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="unbounded"/>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions" minOccurs="0"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNATStaticRulesFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="BasePolicy">
                    <xs:sequence>
                        <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="translationType" minOccurs="1" maxOccurs="1">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:enumeration value="NAT"/>
                                    <xs:enumeration value="PAT"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                        <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="mappedInterfaceGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="original" type="NetworkOrIPRef" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="translated" type="NetworkObjectRefs" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="policyNAT" minOccurs="0" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="destAddress" type="NetworkObjectsRefs" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="services" minOccurs="1" maxOccurs="1">
                                        <xs:complexType>
                                            <xs:sequence>
                                                <xs:element name="serviceData" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="serviceObjectGID"
type="ObjectIdentifier" minOccurs="0" maxOccurs="unbounded"/>
                                            </xs:sequence>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="protocol" type="IPTransportProtocol" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="originalPort" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="translatedPort" type="xs:unsignedInt" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="advancedOptions" type="FirewallNATAdvancedOptions" minOccurs="0"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="PatOptions">
            <xs:sequence>
                <xs:element name="patAddressPool" minOccurs="0" maxOccurs="1">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="patPoolAddressGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
```

```
                                    <xs:element name="interfaceKeyword" type="xs:string" fixed="interface" minOccurs="0"
maxOccurs="1"/>
                                </xs:choice>
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="isPatAllocatedInRoundRobin" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:complexType>
            <xs:complexType name="InterfaceNATManualFirewallPolicy">
                <xs:complexContent>
                    <xs:extension base="BasePolicy">
                        <xs:sequence>
                            <xs:element name="isRuleEnabled" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="section" minOccurs="1" maxOccurs="1">
                                <xs:simpleType>
                                    <xs:restriction base="xs:unsignedInt">
                                        <xs:enumeration value="1"/>
                                        <xs:enumeration value="2"/>
                                        <xs:enumeration value="3"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="realInterface" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:choice>
                                        <xs:element name="realInterfaceGID" type="ObjectIdentifier" minOccurs="0"
maxOccurs="1"/>
                                        <xs:element name="realInterfaceName" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                    </xs:choice>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="mappedInterface" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:choice>
                                        <xs:element name="mappedInterfaceGID" type="ObjectIdentifier"
minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="mappedInterfaceName" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                    </xs:choice>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="source" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="natType" type="NATType" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="originalObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="translated" minOccurs="0" maxOccurs="1">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element name="objectGID" type="NetworkObjectRefs"
minOccurs="0" maxOccurs="1"/>
                                                    <xs:element name="patPool" type="PatOptions"
minOccurs="0" maxOccurs="1"/>
                                                </xs:sequence>
                                            </xs:complexType>
                                        </xs:element>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="destination" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="natType" type="NATType" fixed="Static" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="originalObject" type="NetworkObjectRefs" minOccurs="0"
maxOccurs="1"/>
                                        <xs:element name="translatedObjectGID" type="ObjectIdentifier"
minOccurs="1" maxOccurs="1"/>
```

```
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="service" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="originalObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="transObjectGID" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="isTransDNSReplies" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="direction" minOccurs="0" maxOccurs="1">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="Unidirectional"/>
                                        <xs:enumeration value="Bidirectional"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="isNoProxyARP" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="isRouteLookUp" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:complexType name="InterfaceNATObjectFirewallPolicy">
                <xs:complexContent>
                    <xs:extension base="BasePolicy">
                        <xs:sequence>
                            <xs:element name="section" fixed="2" minOccurs="1" maxOccurs="1">
                                <xs:simpleType>
                                    <xs:restriction base="xs:unsignedInt">
                                        <xs:enumeration value="1"/>
                                        <xs:enumeration value="2"/>
                                        <xs:enumeration value="3"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="realInterface" type="xs:string" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="mappedInterface" type="xs:string" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="natType" type="NATType" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="originalObjectGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                             <xs:element name="translated" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="objectGID" type="NetworkObjectRefs" minOccurs="0"
maxOccurs="1"/>
                                        <xs:element name="patPool" type="PatOptions" minOccurs="0"
maxOccurs="1"/>
                                    </xs:sequence>
                                </xs:complexType>
                             </xs:element>
                            <xs:element name="isTransDNSReplies" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                            <xs:element name="isNoProxyARP" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="isRouteLookUp" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                            <xs:element name="service" minOccurs="0" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="protocol" type="IPTransportProtocol" minOccurs="1"
maxOccurs="1"/>
                                        <xs:element name="originalPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                                        <xs:element name="transPort" type="xs:unsignedInt" minOccurs="0"
maxOccurs="1"/>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
```

```
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <!-- Config Service Methods
-->
        <xs:element name="groupListRequest" type="GroupListRequest"/>
        <xs:complexType name="GroupListRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="includeEmptyGroups" type="xs:boolean"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNAT64ManualFirewallPolicy">
            <xs:complexContent>
                <xs:extension base="InterfaceNATManualFirewallPolicy">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="isInterfaceIpv6" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="isNetToNet" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="groupListResponse" type="GroupListResponse"/>
        <xs:complexType name="GroupListResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="deviceGroup" type="DeviceGroup"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="deviceListByCapabilityRequest" type="DeviceListByCapabilityRequest"/>
        <xs:complexType name="DeviceListByCapabilityRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="deviceCapability" type="DeviceCapability" minOccurs="1"
maxOccurs="unbounded"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="deviceListResponse" type="DeviceListResponse"/>
        <xs:complexType name="DeviceListResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="deviceId" minOccurs="0" maxOccurs="unbounded">
                            <xs:complexType>
                                <xs:sequence minOccurs="1" maxOccurs="1">
                                    <xs:element name="deviceCapability" type="DeviceCapability" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="deviceName" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="ipv4Address" type="xs:string" minOccurs="0"
maxOccurs="1"/>
                                    <xs:element name="gid" type="ObjectIdentifier" minOccurs="1"
maxOccurs="1"/>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:complexType name="InterfaceNAT64ObjectFirewallPolicy">
            <xs:complexContent>
```

```
            <xs:extension base="InterfaceNATObjectFirewallPolicy">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="isInterfaceIpv6" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="isNetToNet" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="deviceListByGroupRequest" type="DeviceListByGroupRequest"/>
    <xs:complexType name="DeviceListByGroupRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="deviceGroupPath" type="DeviceGroupPath"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="deviceConfigByGIDRequest" type="DeviceConfigByGIDRequest"/>
    <xs:complexType name="DeviceConfigByGIDRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="deviceConfigByNameRequest" type="DeviceConfigByNameRequest"/>
    <xs:complexType name="DeviceConfigByNameRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="deviceConfigResponse" type="DeviceConfigResponse"/>
    <xs:complexType name="DeviceConfigResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="device" type="Device" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="policyConfigByNameRequest" type="PolicyConfigByNameRequest"/>
    <xs:complexType name="PolicyConfigByNameRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="policyType" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="policyConfigResponse" type="PolicyConfigResponse"/>
    <xs:complexType name="PolicyConfigResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence minOccurs="1" maxOccurs="1">
                    <xs:element name="policy" type="BasePolicy" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="policyListByDeviceGIDRequest" type="PolicyListByDeviceGIDRequest"/>
    <xs:complexType name="PolicyListByDeviceGIDRequest">
```

```
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="policyListDeviceResponse" type="PolicyListDeviceResponse"/>
        <xs:complexType name="PolicyListDeviceResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="policyList" minOccurs="1" maxOccurs="1">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="policyDesc" type="EntityDescriptor" minOccurs="0"
maxOccurs="unbounded"/>
                                </xs:sequence>
                            </xs:complexType>
                        </xs:element>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="policyNamesByTypeRequest" type="policyNamesByTypeRequest"/>
            <xs:complexType name="policyNamesByTypeRequest">
                <xs:complexContent>
                    <xs:extension base="BaseReqResp">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="policyType" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
        </xs:complexType>
        <xs:element name="policyConfigByDeviceGIDRequest" type="PolicyConfigByDeviceGIDRequest"/>
        <xs:complexType name="PolicyConfigByDeviceGIDRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="gid" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="policyType" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="policyNamesResponse" type="PolicyNamesResponse"/>
        <xs:complexType name="PolicyNamesResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence minOccurs="1" maxOccurs="1">
                        <xs:element name="policyType" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="policy" minOccurs="1" maxOccurs="unbounded">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element name="policyName" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                                    <xs:element name="deviceAssignments" minOccurs="0" maxOccurs="1">
                                        <xs:complexType>
                                            <xs:sequence>
                                                <xs:element name="device" minOccurs="0"
maxOccurs="unbounded">
                                                    <xs:complexType>
                                                        <xs:sequence>
                                                            <xs:element name="deviceGID"
type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                                                            <xs:element name="deviceName"
type="xs:string" minOccurs="1" maxOccurs="1"/>
                                                        </xs:sequence>
                                                    </xs:complexType>
                                                </xs:element>
```

```
                                                </xs:sequence>
                                            </xs:complexType>
                                        </xs:element>
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:extension>
                </xs:complexContent>
            </xs:complexType>
            <xs:element name="policyConfigDeviceResponse" type="PolicyConfigDeviceResponse"/>
            <xs:complexType name="PolicyConfigDeviceResponse">
                <xs:complexContent>
                    <xs:extension base="BaseReqResp">
                        <xs:sequence minOccurs="1" maxOccurs="1">
                            <xs:element name="policy" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence minOccurs="1" maxOccurs="1">
                                        <!-- ALL DEFINED POLICY TYPES -->
                                        <xs:element name="deviceAccessRuleFirewallPolicy"
type="DeviceAccessRuleFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceStaticRoutingRouterPolicy"
type="DeviceStaticRoutingRouterPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceStaticRoutingFirewallPolicy"
type="DeviceStaticRoutingFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceAccessRuleUnifiedFirewallPolicy"
type="DeviceAccessRuleUnifiedFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceBgpRouterPolicy" type="DeviceBGPRouterPolicy"
minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="interfaceNATRouterPolicy"
type="InterfaceNATRouterPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATStaticRulesRouterPolicy"
type="InterfaceNATStaticRulesRouterPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATDynamicRulesRouterPolicy"
type="InterfaceNATDynamicRulesRouterPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceNATTimeoutsRouterPolicy"
type="DeviceNATTimeoutsRouterPolicy" minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="interfaceNATAddressPoolFirewallPolicy"
type="InterfaceNATAddressPoolFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="deviceNATTransOptionsFirewallPolicy"
type="DeviceNATTransOptionsFirewallPolicy" minOccurs="0" maxOccurs="1"/>
                                        <xs:element name="interfaceNATTransExemptionsFirewallPolicy"
type="InterfaceNATTransExemptionsFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATDynamicRulesFirewallPolicy"
type="InterfaceNATDynamicRulesFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATPolicyDynamicRulesFirewallPolicy"
type="InterfaceNATPolicyDynamicRulesFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATStaticRulesFirewallPolicy"
type="InterfaceNATStaticRulesFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATManualFirewallPolicy"
type="InterfaceNATManualFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNATObjectFirewallPolicy"
type="InterfaceNATObjectFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNAT64ManualFirewallPolicy"
type="InterfaceNAT64ManualFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="interfaceNAT64ObjectFirewallPolicy"
type="InterfaceNAT64ObjectFirewallPolicy" minOccurs="0" maxOccurs="unbounded"/>

                                        <!-- ..... all other policies .. -->
                                    </xs:sequence>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name="policyObject" minOccurs="1" maxOccurs="1">
                                <xs:complexType>
                                    <xs:sequence minOccurs="1" maxOccurs="1">
                                        <!-- ALL DEFINED POLICY OBJECT TYPES -->
                                        <xs:element name="networkPolicyObject" type="NetworkPolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
                                        <xs:element name="portListPolicyObject" type="PortListPolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
```

```
                                                <xs:element name="servicePolicyObject" type="ServicePolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="interfaceRolePolicyObject"
type="InterfaceRolePolicyObject" minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="timeRangePolicyObject" type="TimeRangePolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="slaMonitorPolicyObject" type="SLAMonitorPolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="aclPolicyObject" type="ACLPolicyObject" minOccurs="0"
maxOccurs="unbounded"/>
                                                <xs:element name="stdAcePolicyObject" type="StandardACEPolicyObject"
minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="extendedACEPolicyObject"
type="ExtendedACEPolicyObject" minOccurs="0" maxOccurs="unbounded"/>
                                                <xs:element name="identityUserGroupPolicyObject"
type="IdentityUserGroupPolicyObject" minOccurs="0" maxOccurs="unbounded"/>
                                                 <xs:element name="securityGroupPolicyObject"
type="SecurityGroupPolicyObject" minOccurs="0" maxOccurs="unbounded"/>
                                                <!-- ..... all other policy objects .. -->
                                            </xs:sequence>
                                        </xs:complexType>
                                    </xs:element>
                                </xs:sequence>
                            </xs:extension>
                        </xs:complexContent>
                    </xs:complexType>
</xs:schema>
```

10.3 Event XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="csm" targetNamespace="csm">
	<xs:include schemaLocation="common.xsd"/>
	<xs:simpleType name="EventType">
		<xs:restriction base="xs:token">
			<xs:enumeration value="syslog"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="EventFormat">
		<xs:restriction base="xs:token">
			<xs:enumeration value="xml"/>
			<xs:enumeration value="plainText"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="EventCategory">
		<xs:restriction base="xs:token">
			<xs:enumeration value="configChange"/>
			<xs:enumeration value="deviceStatus"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:element name="event" type="Event"/>
	<xs:complexType name="Event">
		<xs:choice>
			<xs:element name="configChange" type="ConfigChangeEvent"/>
		</xs:choice>
	</xs:complexType>
	<xs:simpleType name="UpdateType">
		<xs:restriction base="xs:token">
			<xs:enumeration value="NO_OOB"/>
			<xs:enumeration value="OOB"/>
			<xs:enumeration value="DEVICE_DOWN"/>
			<xs:enumeration value="DEVICE_UP"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:simpleType name="DeploymentType">
		<xs:restriction base="xs:token">
			<xs:enumeration value="Device"/>
			<xs:enumeration value="File"/>
			<xs:enumeration value="AUS"/>
			<xs:enumeration value="CNS"/>
			<xs:enumeration value="TMS"/>
			<xs:enumeration value="Unknown"/>
			<xs:enumeration value="NOT_APPLICABLE"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:complexType name="BaseEventDetails">
		<xs:sequence>
			<xs:element name="subscriptionId" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="eventType" type="EventType" minOccurs="1" maxOccurs="1"/>
			<xs:element name="eventCategory" type="EventCategory" minOccurs="1" maxOccurs="1"/>
			<xs:element name="time" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
			<xs:element name="content" type="xs:string" minOccurs="1" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="DeviceSpecificEvent">
		<xs:complexContent>
			<xs:extension base="BaseEventDetails">
				<xs:sequence>
					<xs:element name="srcIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="srcGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
					<xs:element name="srcDns" type="xs:string" minOccurs="0" maxOccurs="1"/>
					<xs:element name="srcOSType" type="OSType" minOccurs="1" maxOccurs="1"/>
				</xs:sequence>
			</xs:extension>
		</xs:complexContent>
	</xs:complexType>
	<!-- Config Change Event -->
```

```
    <xs:element name="configChangeEvent" type="ConfigChangeEvent"/>
    <xs:complexType name="ConfigChangeEvent">
        <xs:complexContent>
            <xs:extension base="DeviceSpecificEvent">
                <xs:sequence>
                    <xs:element name="deploymentType" type="DeploymentType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="updateType" type="UpdateType" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="deviceStatusEvent" type="DeviceStatusEvent"/>
    <xs:complexType name="DeviceStatusEvent">
        <xs:complexContent>
            <xs:extension base="DeviceSpecificEvent">
                <xs:sequence>
                    <xs:element name="updateType" type="UpdateType" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name="EventFilterItem">
        <xs:sequence>
            <xs:element name="filterEventType" type="EventType" minOccurs="1" maxOccurs="1"/>
            <xs:element name="filterEventFormat" type="EventFormat" minOccurs="1" maxOccurs="1"/>
            <xs:element name="filterEventCategory" type="EventCategory" minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="SyslogServer">
        <xs:sequence>
            <xs:element name="port" type="xs:positiveInteger" default="514" minOccurs="0" maxOccurs="1"/>
            <xs:element name="destAddress" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="eventSubRequest" type="EventSubRequest"/>
    <xs:complexType name="EventSubRequest">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence>
                    <xs:element name="op" type="SubscriptionOperation" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="subscriptionId" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="eventFilterItem" type="EventFilterItem" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="syslogServer" type="SyslogServer" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="eventSubResponse" type="EventSubResponse"/>
    <xs:complexType name="EventSubResponse">
        <xs:complexContent>
            <xs:extension base="BaseReqResp">
                <xs:sequence>
                    <xs:element name="subscriptionId" type="xs:string" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

10.4 Utility XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2011 (http://www.altova.com) by BRIAN MCMAHON (CISCO) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="csm" targetNamespace="csm">
    <xs:include schemaLocation="common.xsd"/>
    <xs:simpleType name="Result">
        <xs:restriction base="xs:token">
            <xs:enumeration value="ok"/>
```

```
                <xs:enumeration value="timeout"/>
                <xs:enumeration value="failed"/>
            </xs:restriction>
        </xs:simpleType>
        <xs:complexType name="DeviceReadOnlyCLICmd">
            <xs:sequence>
                <xs:choice minOccurs="1" maxOccurs="1">
                    <xs:element name="deviceIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
                    <xs:element name="deviceName" type="xs:string" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="deviceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                </xs:choice>
                <xs:element name="cmd" minOccurs="1" maxOccurs="1">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:pattern value="[sS][hH][oO][wW]"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element name="argument" type="xs:string" minOccurs="1" maxOccurs="1">
                <xs:element name="execTimeout" type="xs:unsignedInt" minOccurs="0" maxOccurs="1">
                    <xs:simpleType>
                        <xs:restriction base="xs:unsignedInt">
                            <xs:minInclusive value="1"/>
                        </xs:restriction>
                    </xs:simpleType>
              </xs:element>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="DeviceCmdResult">
            <xs:sequence>
                <xs:element name="deviceIP" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="deviceGID" type="ObjectIdentifier" minOccurs="1" maxOccurs="1"/>
                <xs:element name="deviceName" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="result" type="Result" minOccurs="1" maxOccurs="1"/>
                <xs:element name="resultContent" type="xs:string" minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
        <xs:element name="execDeviceReadOnlyCLICmdsRequest" type="ExecDeviceReadOnlyCLICmdsRequest"/>
        <xs:complexType name="ExecDeviceReadOnlyCLICmdsRequest">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence>
                        <xs:element name="deviceReadOnlyCLICmd" type="DeviceReadOnlyCLICmd" minOccurs="1"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
        <xs:element name="execDeviceReadOnlyCLICmdsResponse" type="ExecDeviceReadOnlyCLICmdsResponse"/>
        <xs:complexType name="ExecDeviceReadOnlyCLICmdsResponse">
            <xs:complexContent>
                <xs:extension base="BaseReqResp">
                    <xs:sequence>
                        <xs:element name="deviceCmdResult" type="DeviceCmdResult" minOccurs="1"
maxOccurs="1"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
</xs:schema>
```

ドキュメントの最後

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/