Configuration Web サービスの使用

この章では、Configuration Web サービスを使用するために設定が必要な環境と、このサービスを使用する方法について説明します。

Configuration Web サービスは、HTTPS 上で REST インターフェイスとして実装されます。HTTP はサポートされていません。

Configuring REST Web サービスは、導入環境内のすべての ACS サーバで使用できます。ただし、ACS プライマリ インスタンスだけが、読み取り操作と書き込み操作をサポートするすべてのサービスを提供します。セカンダリ ACS インスタンスは設定データへの読み取り専用アクセスを提供します。

Monitoring and Report Viewer には、すべての REST アクティビティに対するメッセージと監査ログが表示されます。

ACS CLI での REST Web インターフェイスのイネーブル化

REST Web サービスを使用する前に、ACS で Web インターフェイスをイネーブルにする必要があります。ACS で Web インターフェイスをイネーブルにするには、ACS CLI から次のように入力します。

```
acs config-web-interface rest enable
```

acs config-web-interface コマンドの詳細については、『*CLI Reference Guide for Cisco Secure Access Control System 5.7*』を参照してください。

ACS CLI からの REST Web インターフェイスのステータス表示

Web インターフェイスのステータスを表示するには、ACS CLI から次のように入力します。

```
show acs-config-web-interface
```

acs config-web-interface コマンドの詳細については、『*CLI Reference Guide for Cisco Secure Access Control System 5.7*』を参照してください。

ACS Configuration REST サービスと対話するアプリケーションでは、REST サービスを認証するために管理者アカウントを使用できます。使用するアカウントに対する許可は、REST クライアントによって実行されるすべてのアクティビティを許可するように設定する必要があります。

サポートされる Configuration オブジェクト

ACS の Rest PI は、ACS を設定するためのサービスを提供します。これは、設定機能ごとにまとめられています。ACS 5.7 では、ACS 設定の次の 5 つのサブセットがサポートされています。

- Common Configuration オブジェクト
- Users コンフィギュレーション オブジェクト、Hosts コンフィギュレーション オブジェクト、Identity group コンフィギュレーション オブジェクト。
- Network Device コンフィギュレーション オブジェクト
- Device Group コンフィギュレーション オブジェクト
- Device Group Type コンフィギュレーション オブジェクト
- 最大ユーザ セッション数

表 1（2 ページ）に、サポートされる Configuration オブジェクトを示します。

表 1　　サポートされる Configuration オブジェクト

機能	サポートされる主なクラス	注
Common	Attribute Info	AV ペアのダイナミック属性とも呼ばれます。Attribute Info は、Protocol User 内で作成されます。
	ACS Version	Get メソッドだけをサポートしています。
	Service Location	getall メソッドだけをサポートしています。 プライマリ インスタンスとしての役割を果たす ACS インスタンスと、Monitoring and Troubleshooting Viewer を提供する ACS インスタンスを検索できます。
	Error Message	getall メソッドだけをサポートしています。 REST インターフェイスで使用されるすべての ACS メッセージ コードとメッセージ テキストを取得できます。
Identity	Protocol User	CRUD のすべて（作成、読み取り、更新、および削除）とクエリーをサポートします。
	Identity Group	CRUD のすべてとクエリーをサポートします。 クエリーは、特定のノードのサブグループを取得するために使用されます。各グループのユーザのリストは、ユーザに関してクエリーを実行することにより取得されます。
	Internal Host	CRUD のすべてとクエリーをサポートします。
Network Device	Network Device	CRUD のすべてとクエリーをサポートします。
Device Group	Network Device Group	CRUD のすべてとクエリーをサポートします。
Device Group Type	Network Device Group Type	CRUD のすべてとクエリーをサポートします。

この項の構成は、次のとおりです。

ID グループ

ID グループ オブジェクトは、ID グループ階層でノードを操作するために使用します。グループ名によって、階層内のノードのフル パスが定義されます。新しいノードを追加する場合、ノードの名前（フル パスを含む）は、そのノードを追加すべき階層内の場所を指定していることに注意する必要があります。次に例を示します。

- All Groups:CDO:PMBU
- All Groups:CDO
- All Groups:CDO:PMBU:ACS-Dev

注：上位レベルの階層（親ノード）を作成してから、リーフ ノードを作成する必要があります。
たとえば、階層 All Groups:US:WDC を作成するには、All Groups:US を作成してから、階層内で次のレベルの作成に進む必要があります。

特定のグループの子を取得するには、「start with All groups:CDO」としてフィルタを設定できます。

Attribute Info

AttributeInfo 構造は、属性名と属性値のペアの配列になっています。

属性名は、ユーザ ディクショナリを参照します。このディクショナリで、値の型などの属性の定義を検索できます。属性の値は、ディクショナリの定義に従う必要があります。

次に、2 つの属性があるユーザに対する Java での表現の例を示します。

```
User user = new User();
        user.setDescription(description);
        user.setPassword(password);
        user.setName(userName);
                user.setAttributeInfo(new AttributeInfo[]{
        new AttributeInfo("Department","Dev"),
        new AttributeInfo("Clock","10 Nov 2008 12:12:34")
        });
```

グループの関連付け

REST インターフェイス スキーマは、ユーザ オブジェクト上のグループ名プロパティとして、ID グループに対するユーザの関連付けを示します。

次に、ユーザを ID グループに関連付ける例を示します。

```
User user = new User();
        user.setIdenityGroupName("IdentityGroup:All Groups:Foo");
        user.setDescription(description);
        user.setPassword(password);
        user.setName(userName);
```

Device Info

次に、ネットワーク デバイス グループの設定およびネットワーク デバイスの作成に関連する情報の Java 表現の例を示します。これは、デバイス グループを定義するために GroupInfo オブジェクトを使用する良い例でもあります。

```
IPSubnetType ip1 = new IPSubnetType();
                ip1.setIpAddress("1.1.1.1-5");
                ip1.setNetMask(32);
                ip1.setIpSubnetExclude(new IPSubnetExcludeType().setIpAddress("1.1.1.3");
IPSubnetType ip2 = new IPSubnetType();
                ip1.setIpAddress("3.3.3.1-5");
                ip1.setNetMask(32);
                ip1.setIpSubnetExclude(new IPSubnetExcludeType().setIpAddress("3.3.3.3");

TACACSConnection tacacsCon = new TACACSConnection();
                tacacsCon.setSharedSecret("secret");
                tacacsCon.setSingleConnect("true");
                tacacsCon.setLegacyTACACS("true");

RADIUSConnection radiusCon = new RADIUSConnection();
                radiusCon.setSharedSecret("secret");
                radiusCon.setPortCoA(1700);
                radiusCon.setKeyWrap(true);
                radiusCon.setKeyEncryption("Key");
                radiusCon.setMessageAuthenticationCodeKey("AuthKey");
                radiusCon.setDisplayedInHex(false);

Device device = new Device();
                device.setName(deviceName);
```

```
                device.setdescription(deviceDescription);
                device.setGroupInfo(new GroupInfo[]{new GroupInfo("All
Locations:Chennai","Location"),new GroupInfo("All Device Types:Router","Device Type"});
                device.setSubnets(new IPSubnetType[]{ip1,ip2});
                device.setTacacsConnection();
                device.setRadiusConnection(radiusCon);
```

マスクまたは範囲を使用せずに単一 IP を作成するには、IPSubnetType を作成して、上記の例のように、このオブジェクトにサブネットを関連付けるだけです。

注：REST インターフェイスを使用してネットワーク デバイスを更新するときに、ネットワーク デバイスが関連付けられている少なくとも 1 グループについての情報を提供する必要があります。ネットワーク デバイスは複数のグループに関連付けられている可能性があります。提供されていないグループ情報は、自動的にデフォルト値で更新されます。

クエリー オブジェクト

REST インターフェイス スキーマは、基準およびその他のクエリー パラメータを定義するためのクエリー オブジェクトを公開しています。クエリー オブジェクトは、Users、Identity Groups、Network Device、Network Device Groups、Internal Hosts、および Maximum User Sessions に使用されます。

クエリー オブジェクトには、次の項目に適用されるパラメータが含まれます。

- フィルタリング（4 ページ）
- ソート（5 ページ）
- ページング（5 ページ）

フィルタリング

クエリー オブジェクトを使用すると、フィルタ後の結果セットを取得できます。次の基準に基づいて、ユーザまたは ID グループをフィルタリングできます。

- 単純な条件：プロパティ名、操作、および値が含まれます。たとえば、name STARTS_WITH "A" とします。

  フィルタでは、次の操作がサポートされています。

  - CONTAINS
  - DOES_NOT_CONTAIN
  - ENDS_WITH
  - IS_EMPTY
  - EQUALS
  - NOT_EMPTY
  - NOT_EQUALS
  - STARTS_WITH

- *And* 条件：単純な条件のセットが含まれます。*And* 条件と一致するためには、すべての単純な条件が True に評価される必要があります。

次に「And」フィルタに対する XML ベースの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
      <ns2:query xmlns:ns2="query.rest.mgmt.acs.nm.cisco.com">
            <criteria xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi:type="ns2:AndFilter">
```

```
                <simpleFilters>
                      <propertyName>name</propertyName>
                      <operation>STARTS_WITH</operation>
                      <value>user</value>
                </simpleFilters>
                <simpleFilters>
                      <propertyName>name</propertyName>
                      <operation>ENDS_WITH</operation>
                      <value>1</value>
                </simpleFilters>
          </criteria>
          <numberOfItemsInPage>100</numberOfItemsInPage>
          <startPageNumber>1</startPageNumber>
    </ns2:query>
```

注： XML タグ numberOfItemsInPage は、パッチ 3 で numberofItemsInPage に（大文字「O」が小文字「o」に）変更されました。この変更を ACS に反映するには、パッチ 3 をインストールする必要があります。

次に「And」フィルタに対する Java ベースの例を示します。

```
Query query = new Query();
query.setStartPageNumber(1);
query.setNumberOfItemsInPage(100);

SimpleFilter simpleFilter = new SimpleFilter();
simpleFilter.setOperation(FilterOperation.STARTS_WITH);
simpleFilter.setPropertyName("name");
simpleFilter.setValue("user");

SimpleFilter simpleFilter1 = new SimpleFilter();
simpleFilter1.setOperation(FilterOperation.ENDS_WITH);
simpleFilter1.setPropertyName("name");
simpleFilter1.setValue("1");

AndFilter andFilter = new AndFilter();
andFilter.setSimpleFilters(new SimpleFilter[] { simpleFilter,
            simpleFilter1 });

query.setCriteria(andFilter);
```

ソート

クエリー オブジェクトを使用して、結果をソートできます。次の基準に基づいてソートを実行できます。

- ソートの基準とする 1 つのプロパティ
- ソートの方向（昇順または降順）

ページング

次のページング パラメータで、クエリー オブジェクトを設定できます。

- 要求するページのページ番号
- ページ内のオブジェクト数

ページングは、ステートレスです。要求されたページは、要求ごとに新規計算されます。このため、オブジェクトが同時に追加または削除された場合、ページングによってオブジェクトがスキップされたり、オブジェクトが 2 回返されることがあります。

要求構造

ACS REST 要求は次の項目から構成されます。

- URL パス（6 ページ）
- HTTP メソッド（7 ページ）
- コンテンツ：要求するメソッドに適用される場合は、ACS オブジェクトが含まれます。ACS オブジェクトは、XML で表現されます。

URL パス

URL には、次の項目が含まれます。

- サービス名：Rest
- パッケージ名：Identity、Common または Network Device
- オブジェクト タイプ：User、Identity Group など
- オブジェクト ID は、GET および DELETE メソッドで有効です。
- 操作名は、クエリーなどの CRUD 以外の操作で必要です。

表 2（6 ページ）には、各オブジェクトの URL を示します。

オブジェクト ID

オブジェクトは、名前またはオブジェクト ID によって識別されます。基本オブジェクト キーはオブジェクト名です。また、GET および DELETE メソッドではオブジェクト ID を使用できます。POST メソッドと PUT メソッドでは、ID を含むオブジェクト自体が取得されます。

次の方法により URL で ID を指定できます。

- キーとしての名前：Rest/{package}/{ObjectType}/name/{name}
- キーとしてのオブジェクト ID：Rest//{package}/{ObjectType}/id/{id}
- オブジェクト タイプごとにインスタンスが単一の場合、キーは不要です。例：`REST/Common/AcsVersion`

表 2　URL の概要表

オブジェクト	URL	コメント
ACS Version	../Rest/Common/AcsVersion	1 つのオブジェクトが存在
Service Location	../Rest/Common/ServiceLocation	—
Error Message	../Rest/Common/ErrorMessage	—
User	../Rest/Identity/User/…..	一部のメソッドでは、URL に追加のデータがあります。表 3（7 ページ）を参照してください。
Identity Group	../Rest/Identity/IdentityGroup/…..	一部のメソッドでは、URL に追加のデータがあります。表 3（7 ページ）を参照してください。
Network Device	../Rest/NetworkDevice/Device/.....	一部のメソッドでは、URL に追加のデータがあります。表 3（7 ページ）を参照してください。
Device Group	..Rest/NetworkDevice/DeviceGroup/.....	一部のメソッドでは、URL に追加のデータがあります。表 3（7 ページ）を参照してください。
Internal Host	../Rest/Identity/Host/.....	一部のメソッドでは、URL に追加のデータがあります。表 3（7 ページ）を参照してください。

HTTP メソッド

HTTP メソッドは、設定操作（CRUD：作成、読み取り、更新、および削除）にマッピングされます。

一般的な固有のメソッドは、URL 内で指定されず、HTTP 要求メソッドによって決定されます。その他の場合は、URL に設定操作を追加する必要があります。HTTP メソッドは、ACS 操作にマッピングされます。

- HTTP GET：1 つのオブジェクトまたは複数のオブジェクトを表示します。
- HTTP POST：新しいオブジェクトを作成します。
- HTTP DELETE：オブジェクトを削除します。
- HTTP PUT：既存のオブジェクトを更新します。PUT は、外部のメソッド（CRUD 以外）を呼び出すためにも使用されます。

CRUD 以外の操作に対して HTTP PUT メソッドを使用する場合は、URL で要求する操作を指定します。これは、更新のための PUT メソッドからのメッセージを区別するためにも使用されます。キーワード「op」は、次のように URL に含まれます。

Rest/{package}/{ObjectType}/op/{operation}

例：/Rest/Identity/IdentityGroup/op/query

表 3（7 ページ）では、プライマリ ACS REST メソッドと、HTTP メッセージに対する各メソッドのマッピングについて説明します。

表 3　　HTTP メソッドのまとめ

機能	HTTP メソッド	URL	要求するコンテンツ	成功時の応答
getAll	GET	/{ObjectType}	なし	オブジェクトの収集
getAllDevices	GET	/{ObjectType}	なし	基本情報を含み、ネットワーク デバイス グループの情報は含まないすべてのデバイスのリスト。
getByName	GET	/{ObjectType}/name/{name}[1]	なし	オブジェクト
getById	GET	/{ObjectType}/id/{id}	なし	オブジェクト
create	POST	/{ObjectType}	オブジェクト 注：作成では、オブジェクト ID プロパティを設定しないでください。	オブジェクト ID を含む REST 応答結果。
delete	DELETE	/{ObjectType}/name/{name}[1]	なし	REST 結果
delete	DELETE	/{ObjectType}/ id/{id}	なし	REST 結果
update[2]	PUT	/{ObjectType}	オブジェクト	REST 結果
Query	PUT	/{ObjectType}/op/query	QueryObject	オブジェクトのリスト

1. URL 内の名前は、フルネームです。ACS REST サービスでは、ワイルドカードまたは正規表現はサポートされていません。
2. Update メソッドは、要求の本体で提供されたオブジェクトで、オブジェクト全体を置き換えます。ただし、機密プロパティを除きます。

注：MAC アドレスを使用する GET および DELETE メソッドは Host オブジェクト タイプにだけ使用できます。例：`/Host/macaddress/{macaddress}`。これらのメソッドの name 属性は、MAC アドレスに置き換えられます。これは、Host オブジェクトが name 属性を持たないためです。

注：障害時の応答については、ACS REST 結果（10 ページ）を参照してください。

getAllDevices メソッド

ACS 5.7 では、“getAllDevices” という軽量な REST API のメソッドが導入され、ネットワーク デバイスと AAA クライアントのネットワーク デバイス グループなしの情報のリストを取得できます。このメソッドは、すべてのネットワーク デバイスについて、基本的な、最小限の情報のリストを返します。基本情報には、名前、IPアドレス、説明、デバイス ID、バージョン、認証関連の詳細が含まれています。この新しい軽量な “getAllDevices” メソッドでは、ACS からデータを取得する応答時間が増加します。

ACS からデバイス情報を取得するために、既存の “getAll” メソッドを使用すると、10 ～ 100台のデバイスが存在する場合、ネットワーク デバイス グループを含むすべての情報をすばやく取得します。ただし、1 万台のデバイスがある場合に “getAll” メソッドを使用してデータを取得しようとすると、要求を処理するために約 30 分かかります。したがって、このような状況では、軽量な “getAllDevices” メソッドを使用する必要があります。軽量な “getAllDevices” メソッドは、ACS から制限付きの基本情報をすばやく取得します。デバイスの名前または ID を使用して、“getByName” メソッドまたは “getById” メソッドで、そのデバイスのすべての関連情報を取得できます。ここで、受信したオブジェクトに CRUD 操作を実行できます。

軽量な “getAllDevices” API メソッドを使用する場合は、データベースにバージョン フィールドを追加する必要があります。バージョン フィールドは、オブジェクトが変更された回数を示します。このバージョン フィールドは、そのオブジェクトを更新するたびに増加させる必要があります。“getAllDevices” メソッドを使用してデータを取得すると、バージョン フィールドも ACS から取得されます。データベースのオブジェクトのバージョンを、ACS から取得したオブジェクトのバージョンと比較する必要があります。データベースのバージョンを、ACS から取得されるレコードと比較する必要があります。取得したオブジェクトのバージョンがデータベース内のものと異なっている場合は、“getByName” または “getById” API メソッドを使用して、そのオブジェクトを更新する必要があります。

複数のパラレル REST 要求が ACS でサービス拒否になる：

プライマリ ACS ノードが REST 要求を受信すると、管理者のクレデンシャルを確認し、REST クライアントと TCP 接続を確立して、認証結果、および最終ログイン時間を保存し、結果をセカンダリ ノードに複製します。

ACS が複数のパラレル REST 要求を同時に受信すると、ACS は認証結果をデータベースに保存して、結果をセカンダリ ノードに複製しようとします。これにより、ACS のDenial of Service（DoS）状態が発生します。

安定性の問題を回避するために、ACS に対して接続を確立し、それ以降のすべての REST 要求に対してこの接続を使用することを推奨します。

- Java クライアントを使用している場合は、ACS に対してクライアントを登録して TCP 接続を確立し、それ以降のすべての要求に対してこの接続を使用します。詳細については、ACS Web インターフェイスの SDK の例に含まれる REST クライアントの Java クラスを参照してください（[System Administration] > [Downloads] > [REST Service] > [SDK samples]）。
- CURL クライアントを使用している場合は、CURL プログラムから Cookie を保存し、それ以降のすべての REST 要求に対して Cookie を使用する必要があります。

CURL クッキーを保存して使用するには、次の手順を実行します。

1. 次のコマンドを実行して、cookies.txt ファイルのクッキーを保存します。

```
curl -c cookies.txt https://<ACS_IP_ADRESS>/REST/Common/AcsVersion
```

2. 次のコマンドを実行して、cookies.txt ファイルの Cookie を使用します。

```
curl -b cookies.txt https://<ACS_IP_ADRESS>/REST/Common/AcsVersion
```

応答構造

REST 要求に対する応答は、Web サービスによって返される HTTP ステータス コードとその他のデータを含む標準 HTTP 応答です。また、応答には、要求のタイプに従って、ACS REST 結果オブジェクトまたは ACS コンフィギュレーション オブジェクトが含まれることがあります。

応答の本文に予想されるオブジェクトのタイプを確認するには、HTTP ステータス コードをチェックする必要があります。

- 401 と 404 を除く 4xx HTTPS ステータス コードでは、REST 結果オブジェクトが返されます。
- 500 以外の 5xx ステータス コードでは、メッセージ コンテンツに、サーバ エラーについて説明するテキストが含まれます。
- 500 HTTP ステータス コードでは、REST 結果が返されます。
- 200 と 201 の HTTP ステータス コードでは、特定のメソッドまたはオブジェクト タイプに応じたオブジェクトが返されます。
- 204 HTTP ステータス コードでは、オブジェクトは返されません。

HTTP ステータス コード

ACS では、次のタイプのステータス コードが返されます。

- 成功は 2xx
- クライアント エラーは 4xx
- サーバ エラーは 5xx

ACS では、次のタイプのステータス コードは返されません。

- 1xx
- 3xx

HTTP ステータス コードは、HTTP 応答ヘッダー内と、REST 結果オブジェクト内で返されます。

表 4（9 ページ）には、ACS によって返される HTTP ステータス コードを示します。

表 4　　HTTP ステータス コードの用途

ステータス コード	メッセージ	ACS での使用	コメント
200	OK	取得、作成、およびクエリーの成功	—
204	OK with no content	削除および更新の成功	応答の本体ではデータが返されません。
400	Bad Request	要求エラー：オブジェクトの検証の失敗、XML 構文エラー、およびその他の要求メッセージに関するエラー	要求に不正な構文が含まれているか、要求を実行できません。 たとえば、すでに存在する名前でオブジェクトを作成しようとした場合、オブジェクトの検証は失敗します。 詳細な原因は、REST 結果オブジェクトに示されていることがあります。

表 4　　HTTP ステータス コードの用途 （続き）

ステータス コード	メッセージ	ACS での使用	コメント
401	Unauthorized	認証の失敗またはタイムアウト	403 エラーに似ていますが、特に、認証が失敗した場合またはクレデンシャルを使用できない場合に使用されます。
403	Forbidden	ACS がセカンダリであり、要求を実行できないか、または管理者の許可によって操作が認められていません。	要求は有効ですが、サーバが要求への応答を拒否しています。 401 エラーとは異なり、認証による影響はありません。また、このエラーは、読み取り以外の要求がセカンダリ インスタンスに送信された場合にも表示されます。
404	Not Found	URL が正しくないか、REST サービスがイネーブルになっていない場合。	—
410	Gone	リソースを使用できなくなりました。	存在しないオブジェクトに対して要求が実行されました。たとえば、存在しないオブジェクトを削除しようとしました。
500	Internal Server Error	特定の HTTP コードがないすべてのサーバ エラー。	—

ACS REST 結果

REST 要求の HTTP 応答には、要求されたオブジェクトまたは REST 結果オブジェクトが含まれています。詳細については、表 3（7 ページ）を参照してください。ACS 結果には、次の項目が含まれます。

- HTTP ステータス コード
- HTTP ステータス テキスト
- ACS メッセージ コード
- ACS メッセージ
- 成功した CREATE メソッドのオブジェクト ID

返されるオブジェクト

ACS は、GET メソッドおよびクエリー操作に対してオブジェクトを返します。返されるオブジェクトのタイプは、要求 URL によって決定されます。

GET メソッドが複数のオブジェクトを返す場合、これらのオブジェクトは応答に含まれます。返されるリストが長すぎる場合は、フィルタリング オプションまたはページング オプションを使用する必要があります。

WADL ファイル

WADL ファイルには、すべてのオブジェクトに対するオブジェクト構造（スキーマ）とメソッドが含まれます。

WADL ファイルは、主に文書化に役立ちます。WADL ファイルを使用して、クライアント アプリケーションを生成することはできません。

WADL ファイル構造は、W3C 仕様に従っています。詳細については、http://www.w3.org/Submission/wadl/ を参照してください。

WADL ファイルをダウンロードするには、次の手順を実行します。

1. ACS ユーザ インターフェイスで、[System Administration] > [Downloads] > [Rest Service] の順に進みます。
2. ACS Rest Service WADL ファイルの下で、[Common] または [Identity] をクリックし、ローカル ドライブにファイルを保存します。

スキーマ ファイル

ACS には、ACS 5.7 REST インターフェイスでサポートされるオブジェクトの構造を記述する 4 つの XSD ファイルが付属しています。

次の 4 つの XSD ファイルがあります。

- 次のオブジェクトが記述された Common.xsd
  - Version
  - AttributeInfo
  - Error Message
  - ResultResult、RestCreateResult
  - BaseObject
  - Service Location
  - Status
  - RestCommonOperationType
- 次のオブジェクトが記述された Identity.xsd
  - Users
  - Hosts
  - IdentityGroup
- クエリー オブジェクトの構造が記述された Query.xsd
- 次のオブジェクトが記述された NetworkDevice.xsd
  - Network Devices
  - Network Device Groups
  - Network Device Group Types

スキーマ ファイルは、WADL ファイルのダウンロードと同じ方法でダウンロードできます。JAXB などの使用可能なツールとともにスキーマを使用して、スキーマ クラスを生成できます。

HTTP クライアントを開発し、またはサード パーティ HTTP クライアント コードを使用して、XSD ファイルから生成されたスキーマ クラスと統合できます。

注 : XML をコーディングしたり手動で作成したりするよりも、XSD ファイルから REST クライアント クラスを生成することを強く推奨します。

コードの例

ACS にはクライアント アプリケーション用のコード例が用意されており、ACS REST インターフェイスと連携動作するアプリケーションの開発に役立ちます。コード例は、WADL ファイルおよびスキーマ ファイルと同じ方法でダウンロードできます。

コード例は、Apache HTTP Client（http://hc.apache.org/httpcomponents-client-ga/index.html）および XSD ファイルを利用した JAXB（xjc コマンド）で生成される Java コードに基づいています。次のコード例が含まれています。

- ACS バージョンの取得
- すべてのユーザの取得
- すべてのサービス ロケーションの取得
- フィルタリングしたユーザ リストの取得
- エラー メッセージのリストの取得
- ID と名前によるユーザの取得
- ユーザの作成、削除、更新
- ID グループの作成、削除、および更新
- 名前または ID による ID グループの取得
- ID グループのサブツリーの取得
- ID グループのすべてのユーザの取得
- ネットワーク デバイスの作成、削除、更新
- ID と名前によるネットワーク デバイス ID の取得
- すべてのネットワーク デバイスの取得
- ネットワーク デバイスのクエリー
- ネットワーク デバイス グループの作成、削除、更新
- ID または名前によるネットワーク デバイス グループの取得
- ネットワーク デバイス グループのサブツリーの取得
- ネットワーク デバイス グループのクエリー
- ネットワーク デバイス グループ タイプの作成、削除、更新
- ID と名前によるネットワーク デバイス グループ タイプの取得
- すべてのネットワーク デバイス グループ タイプの取得
- ネットワーク デバイス グループ タイプのクエリー
- 内部ホストの作成、削除、更新
- ID または名前による内部ホストの取得
- すべての内部ホストの取得
- 内部ホストのクエリー

REST API からの内部ユーザのパスワードの変更

注：この機能は ACS 5.7 パッチ 1 のインストール後にのみ機能します。

ACS では、REST API からユーザのパスワードを変更することができます。REST API の GET メソッドを使用して、ACS からパスワード変更の XML ファイルを取得できます。取得した XML ファイルに古いパスワードと新しいパスワードを入力し、PUT メソッドを使用してパスワードを更新できます。この機能は内部ユーザにのみ適用できます。

はじめる前に

REST API が ACS で有効になっていることを確認します。REST API を有効にするには、ACS CLI のEXEC モードで **cs config-web-interface rest enable** コマンドを実行します。

REST API からユーザのパスワードを変更するには：

1. REST API クライアントを開きます。
2. [Request URL] フィールドに **https://<IP address>/Rest/UCP/User/name/<*username*>** と入力します。<IP address> はユーザアカウントが存在する ACS インスタンスの IP アドレス、<username> はパスワードを変更するユーザのユーザ名です。
3. [Method] ドロップダウンリストから [GET] を選択して、[SEND] をクリックします。

   REST API により、次の XML コードを含む [Response] ダイアログボックスが表示されます。

   ```
   <?xml version"1.0" encoding "UTF-8" standalone "yes"?>
   <ns2:userPassword
   xmlns:ns2-"ucp.rest.mgmt.acs.nm.cisco.com">
   <version>0</version><newPassword>*******</newPassword>
   <oldPassword>*******</oldPassword>
   (userName><acsuser></userName></ns2:userPassword>
   ```

   古いパスワードと新しいパスワードのフィールドでアスタリスクが表示されていることを確認します。
4. XML コードで古いパスワードと新しいパスワードを入力します。

   次に例を示します。

   ```
   <?xml version"1.0" encoding "UTF-8" standalone "yes"?>
   <ns2:userPassword
   xmlns:ns2-"ucp.rest.mgmt.acs.nm.cisco.com">
   <version>0</version><newPassword>123456</newPassword>
   <oldPassword>abcdefg</oldPassword>
   <userName><acsuser></userName></ns2:userPassword>
   ```
5. [Request URL] フィールドを [https://<IP address>/Rest/UCP/User] に変更します。
6. [Method] ドロップダウンリストから [PUT] を選択して、[SEND] をクリックします。

正常にパスワードを変更すると、REST API により、[204 No Content] というステータスの応答のダイアログボックスが表示されます。

パスワード変更操作が失敗すると、REST API により、XML コードの対応するエラー コードとエラー メッセージが表示されます。エラー コードの詳細については、表 4（9 ページ）を参照してください。

REST API からの内部ユーザのパスワードの変更