CWMP サービス セキュリティの設定

この章では、認証および暗号化を含む、Broadband Access Center（BAC）の CWMP サービスのセキュリティ オプションについて説明します。この章では、DPE で CWMP サービスおよび HTTP ファイル サービスの HTTP over SSL 転送（この章では SSL と呼びます）をイネーブルにする方法と、証明書管理ツールを使用して DPE に証明書ストアを作成する方法について説明します。

この章では、次のトピックについて説明します。

- 概要（P.13-2）
- BAC における鍵と証明書の管理（P.13-3）
- SSL サービスの設定（P.13-4）
- DPE サービスのセキュリティの設定（P.13-13）

概要

BAC は、RFC 2246 で定義されているとおり、SSL（特に SSL 3.0 および TLS 1.0）を使用することにより、CWMP デバイスのセキュア プロビジョニングをサポートします。RFC 2617 で定義されているとおり、Basic および Digest 認証を使用することにより、DPE との共有秘密情報に基づいてデバイス認証をサポートします。

BAC は、DPE で複数のサービスを提供します。それには、CWMP サービスの 2 つのインスタンスと、HTTP ファイル サービスの 2 つのインスタンスが含まれます。各サービスは、個別にイネーブルにして設定します。その際、同じ DPE で、各種デバイスを異なる方法で柔軟に処理するための、さまざまなセキュリティ オプションを指定できます。

BAC はさらに、一意または汎用のクライアント証明書を使用することにより、セキュアなデバイス認証も提供します。

- 汎用：顧客宅内装置（CPE）すべてまたは CPE の大部分に共通の汎用証明書を使用することにより、SSL 経由のデバイス証明書認証をイネーブルにします。たとえば、特定の配備内のすべての VoIP デバイスが、サービス プロバイダーが発行する同一の汎用証明書を持つことができます。クライアント証明書は、署名機関鍵に対して検証されますが、所定のデバイスのアイデンティティは確立しません。デバイス識別子は、HTTP Basic または Digest 認証を介して提供されるデータを使用するか、CWMP Inform メッセージのデータを使用して形成されます。
- 一意：各デバイスが提供する一意の証明書を使用することにより、SSL 経由のクライアント証明書認証をイネーブルにします。署名認証局の公開鍵を使用してクライアント証明書が検証された後、クライアント証明書の CN フィールドを使用してデバイスの一意の識別子が形成されます。

BAC は、Cisco 11500 Content Services Switch（CSS）などのハードウェア ロード バランサおよび SSL アクセラレータで、クライアント証明書認証を実行する固有のオプションをサポートします。このシナリオでは、ダウンストリーム Cisco CSS は証明書の検証を実行し、SSL セッション（特にクライアント証明書フィールド）に関する情報をデバイス証明書から抽出し、そのデータを特殊 HTTP ヘッダーに挿入します。次に BAC は、Cisco CSS ヘッダー ClientCert-Subject-CN から CN フィールドを取得し、一意のデバイス識別子を形成します。

BAC では、認証オプションを組み合せて使用できます。たとえば、SSL を使用してデバイスが汎用証明書を持っていることを検証し、SSL 接続が確立された後に、追加の HTTP Basic または Digest 認証を実行することができます。

BAC は、no-device 認証のオプションも提供しています。このオプションは、デバイスが DPE のダウンストリームで認証され、デバイスが提示するアイデンティティが信頼できる場合に便利です。BAC DPE が認証なしで実行するよう設定されている場合、BAC DPE は Inform メッセージからデバイスのアイデンティティを抽出して、それを信頼します。

SSL を使用すると、デバイスと DPE の間のトラフィックを暗号化できます。BAC は、デバイスに対して使用する暗号化アルゴリズムや暗号鍵長さを判別する、さまざまな暗号スイートをサポートします。CLI を使用すると、DPE 上で受け入れ可能な暗号スイートを設定できます。別のオプションは、ハードウェア ロード バランサおよびアクセラレータで SSL を終端させることにより、スケーラビリティを高めることです。

（注）DPE CLI を使用して、CWMP サービスおよび HTTP ファイル サービスのセキュリティ オプションを設定します。設定手順については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

BAC における鍵と証明書の管理

DPE は、SSL プロトコルが認証に必要とする証明書を鍵ストアに格納します。この鍵ストアはファイル形式のデータベースで、秘密鍵とそれに関連付けられている公開鍵の X.509 証明書が含まれます。

DPE サーバには 2 つの鍵ストアがあります。これらの鍵ストアは、cacerts 鍵ストアとサーバ証明書鍵ストアです。cacerts 鍵ストアには、デバイスのクライアント証明書を認証するときに DPE が信頼できる公開鍵証明書が含まれます。サーバ証明書鍵ストアには、デバイスに対して DPE を認証するために使用される、サーバ側の証明書の秘密鍵および関連付けられている証明書チェーンが含まれます。

すべての DPE SSL サービスは、単一の cacerts 鍵ストアを共有します。この鍵ストアには、任意の数の署名機関証明書を含めることができます。cacerts 鍵ストアの名前は固定されており、常に *BPR_HOME/jre/lib/security* ディレクトリに存在している必要があります。BAC には、デフォルトの cacerts 鍵ストアが付属しています。これは署名機関証明書を追加および削除することにより操作できます。

cacerts 鍵ストアとは対照的に、サーバ証明書鍵ストアは複数存在することができ、異なるサーバ証明書鍵ストアを使用するように DPE 内の各 SSL サービスを設定できます。各サーバ証明書鍵ストアには、1 つの証明書チェーンのみ含めることができます。サーバ証明書鍵ストアは、*BPR_HOME/dpe/conf* ディレクトリに存在している必要があります。

SSL が DPE で終端され、プロビジョニング グループに複数の DPE が含まれる場合、同一の鍵ストアですべての DPE を設定する必要があります。プロビジョニング グループのすべての DPE を解決するために自動構成サーバ（ACS）の同じ完全修飾ドメイン名（FQDN）URL が使用されるので、同一の鍵ストアが必要です。TR-069 仕様で定義されているとおり、ACS URL は、鍵ストアにインポートされるサーバ証明書の通常名（CN）の値と同一である必要があります。

（注）DPE には、servercerts という名前のデフォルトのサンプル サーバ証明書鍵ストアが付属しています。この鍵ストアには、自己署名サーバ証明書が含まれています。しかし、通常、CWMP デバイスは、自己署名証明書を信頼しないので、デバイス プロビジョニングの SSL をイネーブルにするためにサンプルの鍵ストアを使用することはできません。その代わり、秘密鍵を使用して署名付き ACS 証明書を取得し、これを使用して新しいサーバ証明書鍵ストアを作成する必要があります（P.13-4 の「keytool を使用した DPE 鍵ストアの設定」を参照してください）。ACS 証明書を取得して設定する前に、デフォルトの鍵ストアを使用して SSL サービス リンクをテストできます。

SSL サービスの設定

BAC で SSL をイネーブルにするには、次の手順に従います。

ステップ 1　秘密鍵および関連付けられている ACS 公開鍵証明書を含む、サーバ証明書鍵ストアを作成します。このプロセスでは、サーバ証明書の署名機関の公開証明書をロードするために cacerts 鍵ストアを更新することも必要です。

ステップ 2　オプションで、クライアント証明書を使用するように CPE 認証を設定する場合は、CPE 証明書を検証できる CPE 認証局のルート証明書の公開鍵を使用して、cacerts 鍵ストアを更新します。詳細については、P.13-4 の「keytool を使用した DPE 鍵ストアの設定」を参照してください。

ステップ 3　DPE CLI から、新しいサーバ証明書鍵ストアを使用するように DPE を設定します。詳細については、P.13-13 の「DPE サービスのセキュリティの設定」を参照してください。

ステップ 4　DPE CLI を使用して、CWMP サービスまたは HTTP ファイル サービスの SSL 転送をイネーブルにします。詳細については、P.13-13 の「DPE サービスのセキュリティの設定」を参照してください。

（注）　鍵ストアに対する変更が有効であることを確認するには、CLI から **dpe reload** コマンドを使用するか、ウォッチドッグ エージェントのコマンドラインから **/etc/init.d/bprAgent restart dpe** コマンドを使用して、DPE を再起動する必要があります（P.9-2 の「コマンドラインからの BAC プロセス ウォッチドッグの使用」を参照してください）。

keytool を使用した DPE 鍵ストアの設定

BAC では、keytool ユーティリティを使用してサーバ証明書鍵ストアおよび cacerts 鍵ストアを設定します。keytool は、DPE サーバ上の証明書を管理するために使用する、鍵と証明書管理のユーティリティです。

keytool ユーティリティは、BAC のデフォルトのインストール ディレクトリ */opt/CSCObac/jre/bin/keytool* にあります。DPE サーバへのセキュア接続を使用して、keytool ユーティリティを実行します。

（注）　鍵ストアのファイル形式は keytool のリリースによって異なるので、この BAC のバージョンに付属している keytool ツールを実行する必要があります。

サーバ証明書を設定するには、次の手順に従います。

ステップ 1　keytool を使用して、新しい秘密鍵で新しいサービス鍵ストアを作成します。詳細については、P.13-7 の「新しい証明書のサーバ証明書鍵ストアおよび秘密鍵の生成」を参照してください。

ステップ 2　証明書署名要求（CSR）を生成します。詳細については、P.13-9 の「証明書署名要求の生成」を参照してください。

ステップ 3　CSR を使用して、署名機関に公開証明書を要求します。詳細については、P.13-9 の「証明書署名要求の生成」を参照してください。

ステップ 4　署名機関の公開鍵を cacerts 鍵ストアにロードします。詳細については、P.13-10 の「cacerts 鍵ストアへの署名機関証明書のインポート」を参照してください。

ステップ 5　署名付きサーバ証明書をサーバ鍵ストアにロードします。詳細については、P.13-10 の「サーバ証明書鍵ストアへの署名付き証明書のインポート」を参照してください。

ステップ 6　新しい鍵ストア ファイルを DPE の *BPR_HOME/dpe/conf* ディレクトリに入れます。

ステップ 7　CLI で、新しい鍵ストアを使用するように DPE サービスの 1 つを設定します。詳細については、P.13-4 の「SSL サービスの設定」を参照してください。

ステップ 8　CLI から **dpe reload** コマンドを使用するか、ウォッチドッグ エージェントのコマンドラインから **/etc/init.d/bprAgent restart dpe** コマンドを使用して DPE を再起動します（P.9-2 の「コマンドラインからの BAC プロセス ウォッチドッグの使用」を参照してください）。

（注）　デバイス認証でクライアント証明書の使用をイネーブルにするには、デバイス証明書の署名機関の公開証明書が cacerts 鍵ストアにロードされていることを確認してください。詳細については、P.13-10 の「cacerts 鍵ストアへの署名機関証明書のインポート」に示す手順に従います。

既存の署名付きサーバ証明書のインポート

署名付きサーバ証明書があり、それを鍵ストアにロードする場合は、証明書に関連付けられている秘密鍵を知っている必要があります。この場合、上記の手順ではなく、この項の手順に従ってください。秘密鍵と署名付き証明書の両方を組み合せた、PCKS#12 ファイル形式を使用します。このファイルを **keystore import-pkcs12** コマンドを使用して鍵ストアにロードできます。

既存の署名付きサーバ証明書でサーバ証明書を設定するには、次の手順に従います。

ステップ 1　**keystore import-pkcs12** コマンドを使用して、SSL クライアントに対する DPE の認証で使用された DPE 互換ファイルに既存の秘密鍵と証明書をロードします。

このコマンドを使用する場合、構文は次のようになります。

```
keystore import-pkcs12 keystore-filename pkcs12-filename keystore-password
key-password
export-password export-key-password
```

- *keystore-filename*：作成する鍵ストア ファイルを示します。ファイルがすでに存在する場合は、上書きされます。

  （注）　必ず鍵ストア ファイルのフル パスを指定してください。

- *pkcs12-filename*：鍵と証明書をインポートする PKCS#12 ファイルを示します。

- *keystore-password*：鍵ストア ファイルを作成したときに使用した秘密鍵パスワードと鍵ストア パスワードを示します。このパスワードの長さは、6 文字から 30 文字にする必要があります。
- *key-password*：DPE 鍵ストア内の鍵にアクセスするために使用されるパスワードを示します。このパスワードの長さは、6 文字から 30 文字にする必要があります。
- *export-password*：PKCS#12 ファイルの鍵を復号化するために使用されるパスワードを示します。このエクスポート パスワードの長さは、6 文字から 30 文字にする必要があります。
- *export-key-password*：PKCS#12 鍵ストア内の鍵にアクセスするために使用されるパスワードを示します。このパスワードの長さは、6 文字から 30 文字にする必要があります。

次に例を示します。

```
dpe# keystore import-pkcs12 example.keystore example.pkcs12 changeme changeme changeme
changeme
% Reading alias [1]

% Reading alias [1]: key with format [PKCS8] algorithm [RSA]

% Reading alias [1]: cert type [X.509]

% Created JKS keystore: example.keystore

% OK
```

ステップ 2　新しい鍵ストア ファイルを *BPR_HOME/dpe/conf* ディレクトリにコピーします。

ステップ 3　CLI で、新しい鍵ストアを使用するように DPE サービスの 1 つを設定します。詳細については、P.13-4 の「SSL サービスの設定」を参照してください。

ステップ 4　CLI から **dpe reload** コマンドを使用するか、ウォッチドッグ エージェントのコマンドラインから **/etc/init.d/bprAgent restart dpe** コマンドを使用して DPE を再起動します（P.9-2 の「コマンドラインからの BAC プロセス ウォッチドッグの使用」を参照してください）。

Keytool コマンドの使用

keytool ユーティリティでは、コマンド パラメータを使用して DPE 鍵ストアを設定します。表 13-1 は、keytool のコマンドと説明を示しています。

表 13-1　Keytool のコマンド

-alias *alias*	証明書チェーンと秘密鍵を格納する鍵ストア エントリに割り当てられたアイデンティティを示します。これ以降の keytool コマンドでは、エンティティを参照するときに同じエイリアスを使用する必要があります。
-dname *dname*	サブジェクトおよび発行元により指定された名前など、エンティティを識別するときに使用する X.500 認定者名を示します。
-file *csr_file*	エクスポートする CSR ファイルを示します。
-file *cert_file*	証明書が読み取られるファイルを示します。
-keyalg *keyalg*	鍵ペアの作成に使用されるアルゴリズムを示します。値は DSA（デフォルト）および RSA です。

表 13-1　Keytool のコマンド（続き）

-keysize *keysize*	鍵サイズを指定します。値は 64 ビットの倍数にする必要があります。
-keypass *keypass*	鍵ペアに割り当てられているパスワードを示します。
-keystore *keystore*	鍵ストアの名前と場所をカスタマイズします。
-noprompt	インポート操作中にプロンプトが発行されないように指定します。
-provider *provider_class_name*	サービス プロバイダーがセキュリティ プロパティ ファイルにリストされていない場合、暗号化サービス プロバイダーのマスター クラス ファイルの名前を示します。
-rfc	証明書の MD5 フィンガープリントの出力が、印刷可能な符号化形式で表示されるように指定します。
-storepass *storepass*	鍵ストアに割り当てられているパスワードを示します。
-sigalg *sigalg*	証明書に署名するために使用されるアルゴリズムを示します。
-storetype *storetype*	鍵ストアに割り当てられているタイプまたは鍵ストアへのエントリを示します。
-trustcacerts	信頼のチェーンで、追加の証明書が考慮されるよう指定します。
-v	証明書の MD5 フィンガープリントの出力が、人間が判読可能な形式で印刷されるように指定します。
-validity *valDays*	有効期間を示します。デフォルトは 90 日 です。

（注）　keytool および一般的な証明書管理の概念の詳細については、Sun Microsystems のマニュアルを参照してください。

新しい証明書のサーバ証明書鍵ストアおよび秘密鍵の生成

keytool -genkey コマンドは、鍵ペア（公開鍵および関連付けられている秘密鍵）を生成し、公開鍵を X.509 の自己署名証明書にラップします。これが単一要素証明書チェーンとして格納されます。この証明書チェーンと秘密鍵は、エイリアスによって識別される新しい鍵ストアのエントリに格納されます。

（注）　エイリアスに使用する名前は、重要ではありません。その目的は、複数の鍵ペアを持つ場合に鍵ストア内の鍵ペアを識別することです。BAC では、サーバ証明書鍵ストアには 1 つの鍵ペアを定義しますが、鍵ストアは複数持つことができます。

例 13-1　Keytool -genkey

次の例では、鍵ストア ファイルの名前として *train-1.keystore* を使用します。任意のファイル名を使用できますが、*servercerts* は、BAC が提供するサンプルの鍵ストアと競合するので推奨されていません。

```
dpe# ./keytool -keystore train-1.keystore -alias train-1 -genkey -keyalg RSA
Enter keystore password:  changeme
What is your first and last name?
  [Unknown]:  train-1.bac.test
What is the name of your organizational unit?
  [Unknown]:  BAC Training
What is the name of your organization?
  [Unknown]:  Acme Device, Inc.
What is the name of your City or Locality?
  [Unknown]:  Boxborough
What is the name of your State or Province?
  [Unknown]:  MA
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=train-1.bac.test, OU=BAC Training, O="Acme Device, Inc.", L=Boxborough, ST=MA,
C=US correct?
  [no]:  yes

Enter key password for <train-1>
        (RETURN if same as keystore password):
```

この例では、証明書の CN フィールドに *train-1.bac.test* が挿入され、ACS URL に含まれている FQDN を表します。TR-069 仕様に従って、デバイスは証明書の署名を検証し、証明書内の CN フィールドがアクセス先の URL のフィールドと一致することを確認します。

自己署名証明書の表示

keytool -list 引数は、エイリアスによって識別される鍵ストア エントリの内容を表示します。エイリアスを指定しない場合、鍵ストアの内容全体が表示されます。

-list を **-v** と組み合せると、エイリアスに関連付けられている証明書チェーンが表示されます。次の **keytool -list** のサンプル出力は、単一の自己署名証明書が含まれる鍵ストアを示しています。

例 13-2　Keytool -list

```
# ./keytool -keystore train-1.keystore -list -v
Enter keystore password:  changeme

Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: train-1
Creation date: Nov 8, 2005
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=train-1.bac.test, OU=BAC Training, O="Acme Device, Inc.", L=Boxborough,
ST=MA, C=US
Issuer: CN=train-1.bac.test, OU=BAC Training, O="Acme Device, Inc.", L=Boxborough,
ST=MA, C=US
Serial number: 43714f22
Valid from: Tue Nov 08 20:21:38 EST 2005 until: Mon Feb 06 20:21:38 EST 2006
Certificate fingerprints:
         MD5:  CF:D4:CB:D1:20:6F:8C:12:ED:EA:2F:21:53:57:E5:1D
         SHA1: DD:AE:96:02:71:55:F8:1F:14:4F:D7:64:9C:FE:91:DE:65:C9:BB:49
```

証明書署名要求の生成

一連の手順におけるこの時点では、鍵ストアに秘密鍵と X.509 自己署名証明書が含まれています。DPE ACS がこの証明書でデバイスの初期ハンドシェークに応答を試みる場合、デバイスは、CPE が信頼した認証局が証明書に署名しなかったことを示す TLS アラート `bad CA` により証明書を拒否します。したがって、CPE が信頼する署名機関が証明書に署名する必要があります。

（注）SSL をサポートするには、デバイスに、信頼する署名機関の事前に設定された公開証明書のリストを含める必要があります。デバイスが信頼する機関の 1 つが、ACS 証明書に署名する必要があります。

keytool -certreq コマンド パラメータは、証明書署名要求（CSR）を生成します。このコマンドは、業界標準の PKCS#10 形式で CRS を生成します。

例 13-3　Keytool -certreq

次の例では、**alias** *train-1* に事前に存在する自己署名証明書を鍵ストアとともに使用して、証明書署名要求を生成し、要求を `train-1.csr` ファイルに出力します。

```
dpe# ./keytool -keystore train-1.keystore -alias train-1 -certreq -file train-1.csr
Enter keystore password:  changeme
```

次のステップでは、CSR ファイルを署名機関に送信します。署名機関、またはその署名機関向けの秘密鍵を所有する管理者は、この要求に基づいて署名付き証明書を生成します。管理者からは、署名機関の公開証明書を取得する必要もあります。

署名付き証明書の検証

署名された証明書を受け取った後、**keytool -printcert** コマンドを使用して、自己署名証明書のファイル形式が正しいかどうか、また正しいオーナーおよび発行元フィールドを使用しているかどうかを確認します。コマンドは、**-file** *cert_file* パラメータから証明書を読み取り、その内容を人間が判読可能な形式で出力します。

例 13-4　Keytool -printcert

この例の *train-1.crt* ファイルは、管理者が提供する署名付き証明書を示します。

```
dpe# ./keytool -printcert -file train-1.crt
Owner: CN=train-1.bac.test, OU=BAC Training, O="Acme Device, Inc.", ST=MA, C=US
Issuer: EMAILADDRESS=linksys-certadmin@cisco.com, CN=Acme Device Provisioning Root
Authority 1, OU=Acme Device Certificate Authority, O="Acme Device, Inc.", L=Irvine,
ST=California, C=US
Serial number: 1
Valid from: Tue Nov 08 12:40:28 EST 2005 until: Thu Nov 08 12:40:28 EST 2007
Certificate fingerprints:
         MD5:  25:8E:98:C5:5C:23:5C:A0:4D:51:CF:2A:AA:2A:FC:42
         SHA1: 05:C1:2D:C6:94:78:D1:40:88:6A:55:67:43:27:68:D3:AC:43:C6:A5
```

（注）keytool は、X.509 v1、v2、および v3 の証明書と、そのタイプの証明書から成る PKCS#7 形式の証明書チェーンを出力できます。出力されるデータは、バイナリ符号化形式で提供されるか、RFC 1421 で定義されているように、印刷可能符号化形式（Base64 符号化とも呼ばれる）で提供される必要があります。

cacerts 鍵ストアへの署名機関証明書のインポート

証明書をサーバ証明書鍵ストアにインポートする前に、署名機関の公開証明書を cacerts 鍵ストアにインポートする必要があります。証明書を鍵ストアにインポートするときに、keytool は、証明書とその署名機関との間で信頼チェーンを確立できるかどうかを確認するからです。信頼のチェーンを確立できない場合は、エラー メッセージが表示されます。

（注）BAC に組み込まれている cacerts ファイルには、一般的な第三者署名機関のいくつかのルート証明書が付属しています。cacerts 鍵ストアは、keytool ユーティリティを使用して管理できます。デフォルトの cacerts 鍵ストアのパスワードは **changeit** です。cacerts データベース ファイルは、*BPR_HOME/jre/lib/security* ディレクトリにあります。
cacerts 鍵ストアをいずれかの場所にコピーする必要はありません。DPE は、再起動の直後に新しい鍵ストアを使用します。

例 13-5　Keytool -import（署名機関証明書）

```
# ./keytool -import -alias DeviceProvRoot -file rootCA4.crt -keystore
/opt/CSCObac/jre/lib/security/cacerts
Enter keystore password:  changeit
Owner: EMAILADDRESS=linksys-certadmin@cisco.com, CN=Acme Device Provisioning Root
Authority 1, OU=Acme Device Certificate Authority, O="Acme Device, Inc.", L=Irvine,
ST=California, C=US
Issuer: EMAILADDRESS=linksys-certadmin@cisco.com, CN=Acme Device Provisioning Root
Authority 1, OU=Acme Device Certificate Authority, O="Acme Device, Inc.", L=Irvine,
ST=California, C=US
Serial number: 8bcbc07a0768c1eb78e6c5c93c0c2ff0
Valid from: Fri Jul 01 21:22:12 EDT 2005 until: Mon Jun 29 21:22:12 EDT 2015
Certificate fingerprints:
         MD5:  C4:D4:09:6A:60:34:A0:00:96:4F:4D:47:23:86:8C:FA
         SHA1: B0:CC:6D:CD:BB:62:1B:A1:15:D3:2D:68:7E:D0:4A:0C:91:C2:A5:FD
Trust this certificate? [no]:  yes
Certificate was added to keystore.
```

（注）keytool は、X.509 v1、v2、および v3 の証明書と、そのタイプの証明書から成る PKCS#7 形式の証明書チェーンをインポートできます。インポートされるデータは、バイナリ符号化形式で提供されるか、RFC 1421 で定義されているように、印刷可能符号化形式（Base64 符号化とも呼ばれる）で提供される必要があります。

サーバ証明書鍵ストアへの署名付き証明書のインポート

署名機関の公開証明書を cacerts 鍵ストアにインポートした後は、署名付きサーバ証明書を DPE サーバ証明書鍵ストアにインポートする必要があります。鍵ストアには、秘密鍵および対応する自己署名証明書（公開鍵）がすでに格納されています。署名応答（署名付き証明書）をインポートすることにより、鍵ストアは、署名付き証明書をサーバ証明書鍵ストア内の既存の秘密鍵に関連付けるように変更されます。

証明書応答を鍵ストアにインポートする場合、*cacerts* ファイル内の証明書に対する **-import** コマンドで **-trustcacerts** フラグを使用し、サブジェクトの鍵ストア内の証明書応答との信頼チェーンを確立する必要があります。

例 13-6　Keytool -import（署名付きサーバ証明書）

```
dpe# ./keytool -import -trustcacerts -file train-1.crt -keystore train-1.keystore
-alias train-1
Enter key password: changeme2
Enter keystore password:  changeme
Certificate reply was installed in keystore.
Certificate was added to keystore.
```

署名付きサーバ証明書を DPE サーバ証明書鍵ストアにインポートした後、**keytool -printcert** コマンドを使用して鍵ストアの内容を確認します。詳細は、P.13-9 の「署名付き証明書の検証」に記載されています。これで、**-printcert** の出力が、発行元が署名認証局であり、ルート信頼証明書を持つ署名機関を介して信頼チェーンが確立されたことを示すようになりました。

クライアント認証の証明書のインポート

このステップを実行する必要があるのは、DPE でクライアント証明書を使用してクライアント認証を設定した場合だけです。クライアント証明書を使用してクライアント認証をイネーブルにした場合、cacerts 鍵ストアには、CPE クライアント証明書に署名した署名機関の公開証明書が含まれている必要があります。DPE で、提供された証明書の検証をイネーブルにするには、この証明書が必要です。

例 13-7　Keytool -import（署名機関証明書）

```
# ./keytool -import -alias DeviceClientRoot -file rootCA3.crt -keystore
/opt/CSCObac/jre/lib/security/cacerts
Enter keystore password:  changeit
Owner: EMAILADDRESS=linksys-certadmin@cisco.com, CN=Acme Device Client Root Authority
1, OU=Acme Device Certificate Authority, O=Acme Device LLC., L=Irvine, ST=California,
C=US
Issuer: EMAILADDRESS=linksys-certadmin@cisco.com, CN=Acme Device Client Root Authority
1, OU=Acme Device Certificate Authority, O=Acme Device LLC., L=Irvine, ST=California,
C=US
Serial number: d07d8a7badba7cb6446998b1ea89879f
Valid from: Fri Jul 01 21:19:50 EDT 2005 until: Mon Jun 29 21:19:50 EDT 2015
Certificate fingerprints:
         MD5:  40:B0:40:49:37:3A:51:1F:0D:78:B6:B3:E2:2C:1A:E8
         SHA1: 96:F5:84:71:84:CC:0A:A2:1E:7B:44:A2:B6:F5:B7:3D:C4:9F:81:3B
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

（注） この手順は、P.13-10 の「cacerts 鍵ストアへの署名機関証明書のインポート」で説明されている手順とまったく同じです。いずれの場合も、署名機関の公開証明書をロードします。サーバ証明書の署名機関が、デバイス証明書の署名機関と同じ場合は、署名は 1 度だけ追加します。

DPE へのサービス プロバイダー鍵ストアの提供

署名付き公開鍵証明書が含まれる新しいサービス証明書鍵ストアを取得したら、鍵ストア ファイルを DPE にコピーする必要があります。ファイルは、*BPR_HOME/dpe/conf* ディレクトリにコピーする必要があります。

例 13-8　DPE 設定ディレクトリへの鍵ストアのコピー

```
dpe# cp train-1.keystore /opt/CSCObac/dpe/conf
```

このステップが完了したら、DPE CLI を使用して、新しい鍵ストアを使用するように DPE サービスを設定できます。

（注）　cacerts 鍵ストアをコピーする必要はありません。DPE は、再起動されるとただちに新しい鍵ストアを使用します。

詳細については、P.13-13 の「DPE サービスのセキュリティの設定」を参照してください。DPE 設定コマンドの詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

DPE サービスのセキュリティの設定

この項では、認証オプションを設定する方法、および DPE サービスに SSL を設定する方法について説明します。

DPE セキュリティ オプションは、DPE CLI から設定できます。詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

DPE は、実行中の 2 つの CWMP サービスと 2 つの HTTP ファイル サービスを並行してサポートします。各サービスには異なる設定のセキュリティ オプションを定義でき、異なるポートで実行できます。デフォルトでは、1 つの CWMP サービスと 1 つの HTTP サービスのみがイネーブルにされ、SSL なしで設定されています。2 つの追加のサーバは SSL 用に設定されていますが、デフォルトではディセーブルになっています。

表 13-2 では、CWMP サービスおよび HTTP ファイル サービスの各インスタンスのデフォルト設定を指定します。

表 13-2　CWMP 技術のデフォルト設定

	CWMP サービス		HTTP ファイル サービス	
	サービス 1	**サービス 2**	**ファイル サービス 1**	**ファイル サービス 2**
デフォルト モード	イネーブル	ディセーブル	イネーブル	ディセーブル
認証	Digest	Digest	Digest	Digest
ポート番号	7547	7548	7549	7550
SSL プロトコル	ディセーブル	イネーブル	ディセーブル	イネーブル

DPE での SSL の設定

所定のサービスで SSL をイネーブルにするには、次の手順に従います。

ステップ 1　HTTP クライアント認証を設定します。Basic または Digest モードで認証をイネーブルにするか、HTTP 認証をディセーブルにできます。

ステップ 2　サービスの SSL プロトコルをイネーブルにします。

ステップ 3　デバイスが DPE 上のサービスにアクセスするときに使用するポートを設定します。

ステップ 4　鍵ストアのファイル名、鍵ストアのパスワード、および鍵のパスワードを設定します。

ステップ 5　SSL を使用してクライアント証明書認証を設定します。クライアント認証は、汎用または一意のクライアント証明書を使用するように設定できます。

ステップ 6　オプションで、CWMP サービスまたは HTTP ファイル サービスの他のインスタンスをディセーブルにします。

ステップ 7　1 つのサービスのインスタンスを 1 つイネーブルにします。サービスは、CWMP サービスまたは HTTP ファイル サービスのいずれかです。

ステップ 8　**dpe reload** コマンドを使用して DPE を再起動し、変更が有効になっていることを確認します。

CWMP サービスの SSL のイネーブル化

次の例は、CWMP サービスの 1 つのインスタンスで SSL をイネーブルにするためのコマンドを示しています。この例では、クライアント証明書および Basic モードの HTTP 認証を使用することにより、SSL クライアントの二重の認証をイネーブルにします。

```
dpe# service cwmp 2 client-auth basic
% OK (Basic authentication was enabled. Digest authentication was disabled. Requires
DPE restart "> dpe reload")

dpe# service cwmp 2 port 7548
% OK (Requires DPE restart "> dpe reload")

dpe# service cwmp 2 ssl enable true
% OK (Requires DPE restart "> dpe reload")

dpe# service cwmp 2 ssl keystore train-1.keystore changeme changeme2
% OK (Requires DPE restart "> dpe reload")

dpe# service cwmp 2 ssl client-auth client-cert-unique
% OK (Requires DPE restart "> dpe reload")

dpe# service cwmp 1 enabled false
% OK (Requires DPE restart "> dpe reload")

dpe# service cwmp 2 enable true
% OK (Requires DPE restart "> dpe reload")

dpe# dpe reload
Process dpe has been restarted.

% OK
```

（注） この例では、CWMP インスタンス 2 の SSL 転送を設定します。例では、`train-1.keystore` が、署名付きのサーバ用の公開鍵証明書とともにプリロードされていること、また鍵ストア ファイルが DPE の *BPR_HOME/dpe/conf* ディレクトリに移動されたことを想定しています。

HTTP ファイル サービスの SSL のイネーブル化

次の例は、HTTP ファイル サービスの 1 つのインスタンスで SSL プロトコルをイネーブルにするためのコマンドを示しています。この例では、クライアント認証はディセーブルにされるので、認証チャレンジなしでアクセスできます。

```
dpe# service http 2 client-auth none
% OK (Requires DPE restart "> dpe reload")

dpe# service http 2 port 7550
% OK (Requires DPE restart "> dpe reload")

dpe# service http 2 ssl enable true
% OK (Requires DPE restart "> dpe reload")

dpe# service http 2 ssl keystore train-1.keystore changeme changeme2
% OK (Requires DPE restart "> dpe reload")

dpe# service http 2 ssl client-auth none
% OK (Requires DPE restart "> dpe reload")

dpe# service http 1 enable false
% OK (Requires DPE restart "> dpe reload")

dpe# service http 2 enable true
% OK (Requires DPE restart "> dpe reload")

dpe# dpe reload
Process dpe has been restarted.

% OK
```

（注）　詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

CPE 認証の設定

CPE 認証は、次の事項を通じてサポートされます。

- HTTP Basic または Digest 認証を使用する共有秘密情報。
- SSL を使用するクライアント証明書。共有秘密情報 HTTP ベースの認証の代わりとして、または追加機能として、証明書ベースのクライアント認証を使用できます。
- 外部クライアント証明書認証。このシナリオでは、SSL　接続は、クライアントの認証も行うハードウェア ロード バランサで終端します。

認証の目的は、信頼されるデバイス ID を確立することです。この ID は、DPE キャッシュでデバイス命令を検索するために使用されます。このデバイス ID は、BAC RDU データベースのデバイス レコードに事前プロビジョニングされたデバイス識別子と相関関係にあります。共有秘密情報 HTTP ベースの認証の場合、ユーザ名はデバイスのアイデンティティを確立するために使用され、デバイス ID として処理されます。一意のクライアント証明書を使用する認証の場合、デバイス ID はデバイス証明書の CN フィールドから取得されます。外部エンティティ（Cisco CSS 15000 シリーズ ロード バランサなど）によるクライアント証明書の場合、CN フィールドを含む証明書データは、HTTP ヘッダーで DPE に渡されます。

（注）クライアントを信頼できる場合、クライアント認証を行わないように選択することもできます。たとえば、加入者の物理ラインに基づいて、クライアントによるネットワーク アクセスがすでに認証されている場合もあります。この場合は、BAC を認証なしで設定することができます。Inform メッセージから、信頼されるデバイス ID が取得されます。

DPE 認証オプションは、DPE CLI から設定できます。

共有秘密情報の認証

BAC は、CPE と DPE との間の共有パスワードに基づく HTTP 認証をサポートします。この認証は、Basic および Digest の 2 つのモードで使用可能です。

（注）クライアント認証中のセキュリティ リスクを限定するため、Digest モード（デフォルトの設定）の使用が推奨されています。Basic モードではクライアント認証を使用しないようにしてください。

DPE CLI から Basic または Digest モードで認証を設定するには、次のコマンドを使用します。

```
# service {cwmp | http} num client-auth mode
```

- *num*：サービスのインスタンスを指定します。1 または 2 になります。
- *mode*：サービスのクライアント認証モードを示します。クライアント認証モードには、次のものがあります。
  - basic：Basic HTTP 認証をイネーブルにします。
  - digest：Digest HTTP 認証をイネーブルにします。これがデフォルト設定です。
  - none：Basic および Digest 認証をディセーブルにします。

詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

デバイス パスワードの変更

BAC では、デバイスの共有秘密情報を設定できます。共有秘密情報は、`IPDeviceKeys.CPE_PASSWORD` プロパティを使用してデバイス レコードに格納されます。CPE は、HTTP ベースの認証中に、このパスワードを知っていることを証明する必要があります。Basic モードでは、パスワードは符号化クリア テキストとして送信されますが、Digest モードでは、デバイスはパスワードを送信することなくパスワードに関する知識を持っていることを証明できます。

パスワードは、次の方法で設定できます。

- API 上では、`IPDeviceKeys.CPE_PASSWORD` プロパティを使用します。
- 管理者のユーザ インターフェイス上では、**Devices > Add Device** ページまたは **Modify Device** ページにある CPE Password フィールドを使用します。

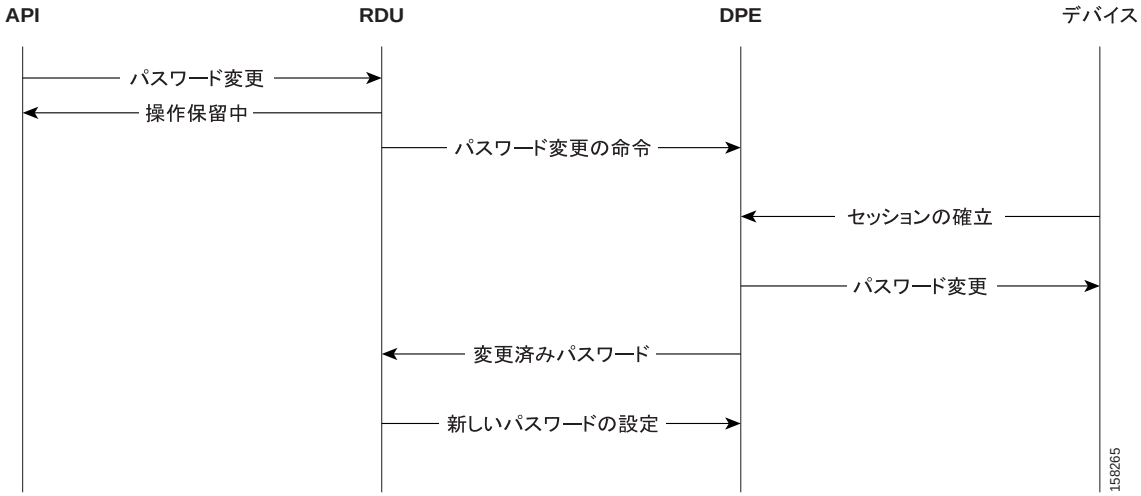
（注）RDU への DPE 接続が使用可能でない場合、CPE パスワードは変更できません。

SSL を使用してクライアント証明書によりクライアント認証をイネーブルにした場合、CPE パスワードはオプションです。

BAC によって使用されるパスワードの変更と、デバイスによって使用されるパスワードの変更を、区別する必要があります。BAC のデバイス レコードでパスワードを設定する場合、以前のパスワードの値に応じて結果は異なります。パスワードがデバイス レコードですでに設定されている場合、BAC はこれを変更して、実際のデバイスでパスワードを変更するプロセスを開始します。ただし、デバイス レコードに以前のパスワード値が存在しない（または空の文字列であった）場合、BAC はデバイス レコードに新しいパスワードを設定しますが、実際のデバイスのパスワードの変更は開始しません。したがって、BAC 内でのみパスワードを既存の値から別の値に変更する場合は、最初に BAC で値をリセット（空の文字列に設定）してから、それを新しい値に設定する必要があります。

図 13-1 は、実際のデバイスでパスワードを変更するために BAC が使用するプロセスを示しています。BAC が、デバイスを認証するために最初に古いパスワードを使用してから、新しいパスワードを設定する必要があるため、このプロセスは複雑になっています。パスワードの変更が確認されてからのみ、BAC は以前のパスワードを無効にしてデータベースから削除します。

図 13-1　パスワード変更のフロー



RDU のデバイス オブジェクトでパスワードが変更された後、パスワード変更命令はデバイス設定に含められます。この時点では、デバイスを認証するために引き続き古いパスワードが使用されます。次に、新しい設定命令がデバイスのプロビジョニング グループ内のすべての DPE に転送されます。

デバイスが BAC と次のセッションを作成するとき、デバイスは古いパスワードで認証されます。その後、新しいパスワードは SetParameterValues RPC を介して設定されます。DPE が変更を RDU に通知すると、RDU はデバイス パスワードを変更し、新しいパスワードが含まれる新しい設定命令を生成します。その後、古いパスワードは BAC に格納されなくなります。

新しいパスワードは、デバイスで設定されるまで変更可能です。たとえば、デバイスの元のパスワード（*password*）が *newpassword* に変更されたとします。しかし、新しいパスワードを設定するためにデバイスが BAC に接続する前であれば、パスワードは引き続き変更できます。この例を示すため、ここでパスワードを *n3wpa550d* に変更したと仮定します。デバイスが BAC に接続するまで、デバイスは引き続き古いパスワード（*password*）で認証されます。新しいパスワード（*n3wpa550d*）は、デバイスの BAC との次のセッション中に設定されます。

デバイス パスワードの修正

スペルが間違っているパスワードを、RDU で変更することもできます。たとえば、*password* の代わりに *passwrod* と入力した場合、最初にパスワードを削除して送信します。この時点で、デバイス オブジェクトには、関連付けられているパスワードはありません。次に、デバイスにパスワードを追加します。

（注）　この方法は、デバイス内のパスワードではなく、BAC 内のパスワードを変更する場合に使用します。

管理者のユーザ インターフェイスからデバイス パスワードを修正するには、次の手順に従います。

ステップ 1　**Devices > Manage Devices** から、適切なデバイスに対応する Identifier リンクをクリックします。

ステップ 2　Modify Device ページが表示されます。CPE Password フィールドのデバイス パスワードを削除します。

ステップ 3　**Submit** をクリックします。

ステップ 4　**Modify Device** ページに戻ります。正しいパスワードを入力します。

ステップ 5　**Submit** をクリックします。

これで、RDU のデバイス レコードでパスワードが変更されました。この手順の後、実際のデバイス上でパスワードの変更は開始されません。

クライアント証明書認証

認証のために、デバイスにより提供された検証済みの証明書を要求するように、DPE を設定できます。次の証明書が含まれます。

- 汎用：すべての CPE または CPE の大部分に共通の汎用証明書を使用して、SSL 経由のデバイス証明書認証をイネーブルにします。
- 一意：各 CPE が提供する一意の証明書を使用することにより SSL 経由のクライアント証明書認証をイネーブルにします。

デバイス証明書が一意である場合、HTTP 認証を使用する必要はありません。しかし、すべてのデバイスで同じ証明書が使用される場合（つまり、サービス プロバイダーが使用する単一デバイス証明書）、追加の HTTP 認証を設定する必要があります。

DPE CLI からクライアント証明書認証を設定するには、次のコマンドを使用します。

```
# service {cwmp | http} num ssl client-auth mode
```

- *num :* サービスのインスタンスを示します。1 または 2 になります。デフォルトで、SSL によるクライアント証明書認証は次のようになります。
  - サービス 1 の場合：ディセーブル
  - サービス 2 の場合：ディセーブル

- *mode*：クライアント証明書認証のモードを示します。BAC は次のモードをサポートします。
  - client-cert-generic：デバイスのセットに共通の証明書を使用します。
  - client-cert-unique：デバイスに一意の証明書を使用します。
  - none：クライアント証明書認証をディセーブルにします。

外部クライアント証明書認証

SSL 接続は、ロード バランサなど、DPE の外側で終端させることができます。この場合、DPE は SSL サービスなしで設定できます。

SSL 接続の終端として CSS 11500 などのロード バランサを使用する場合、DPE は一意の証明書を受け取らないので、HTTP 認証（Basic または Digest）を持たないデバイスを識別することは不可能になります。この問題を解決するため、ロード バランサは、証明書情報が含まれる追加の HTTP ヘッダーを挿入します。

単一のセッションに複数の TCP 接続が含まれる場合、それぞれの接続が認証されます（イネーブル時）。セッションの cookie により、デバイスと既存のセッション状態がバインドされます。

CSS でクライアント証明書認証を設定する方法の詳細については、P.12-18 の「Cisco CSS でのクライアント証明書認証の設定」を参照してください。

BAC における認証オプション

この項では、BAC で CWMP サービスおよび HTTP ファイル サービスに対して使用可能なクライアント認証オプションの組み合せの概要を説明します。これらのサービスの各インスタンスは、要件に合せて DPE CLI から個別に設定できます。

BAC は、CPE と DPE との間の共有パスワードに基づく Basic および Digest モードの HTTP 認証をサポートします。HTTP ベースの認証を使用する場合は、Digest モードを使用することをお勧めします。

各 CPE に一意の証明書を使用して CPE 認証を設定することもできます。この場合、HTTP 認証は必要ありません。ただし、すべての CPE または CPE の大部分に共通の汎用証明書を使用して CPE 認証を設定する場合、追加の HTTP 認証を要求するように DPE を設定することをお勧めします。

表 13-3 は、BAC がサポートするさまざまなオプションと、認証を設定するために DPE CLI から使用するコマンドを示しています。各コマンドの詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

表 13-3　BAC における認証オプション

オプション	使用するコマンド
HTTP を使用	
Basic または Digest モードで HTTP を使用してデバイス認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth** {**basic** | **digest**}
HTTP を使用するデバイス認証をディセーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth none**
（注）　HTTP を使用するデバイス認証がディセーブルになっている場合、信頼されるデバイスのアイデンティティは、デバイスからの Inform メッセージの値を使用して形成されます。	
SSL を使用	
SSL 接続で、HTTP Basic または Digest モードのデバイス認証をイネーブルにする。クライアント証明書認証は使用しない。	**service** {**cwmp** | **http**} *num* **client-auth** {**basic** | **digest**} **service** {**cwmp** | **http**} *num* **ssl client-auth none**
一意のクライアント証明書に基づく SSL 接続で、HTTP Basic または Digest モードでデバイス認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth** {**basic** | **digest**} **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-unique**
（注）　HTTP ベース（Basic または Digest）の認証、および一意の証明書を使用する認証を要求するように DPE を設定した場合、デバイスは両方のメカニズムを使用して認証されます。この二重認証シナリオでは、デバイスの一意の識別子は、クライアント証明書の CN フィールドを使用して形成されます。その結果、信頼されるデバイス ID が確立されます。	
汎用のクライアント証明書に基づく SSL 接続で、HTTP Basic または Digest モードでデバイス認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth** {**basic** | **digest**} **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-generic**
HTTP Basic または Digest 認証、および Cisco CSS 11500 によるクライアント証明書認証を使用してデバイス認証をイネーブルにする。	**service** {**cwmp**| **http**} *num* **client-auth** {**basic** | **digest**} **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-css-ext**
（注）　この二重の認証のシナリオでは、信頼されるデバイス ID は、Cisco CSS による認証に基づいて確立され、HTTP ヘッダーを使用する DPE に伝達されます。	
Cisco CSS 11500 によって検証されたクライアント証明書に基づくデバイス認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-css-ext**
デバイス認証およびクライアント証明書認証をディセーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth none** **service** {**cwmp** | **http**} *num* **ssl client-auth none**
（注）　デバイス認証およびクライアント証明書認証がディセーブルになっている場合、デバイスは信頼済み、または事前認証済みと見なされ、DPE は CWMP Inform メッセージからのデータを使用して、信頼されるデバイスのアイデンティティを確立します。	

表 13-3　BAC における認証オプション（続き）

オプション	使用するコマンド
HTTP ベースのデバイス認証をディセーブルにし、各 CPE が提供する一意の証明書を使用して SSL 経由のクライアント認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth none** **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-unique**
HTTP ベースのデバイス認証をディセーブルにし、汎用の証明書を使用して SSL 経由のクライアント認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth none** **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-generic**
HTTP ベースのデバイス認証をディセーブルにし、Cisco CSS 11500 によって検証されたクライアント証明書に基づく認証をイネーブルにする。	**service** {**cwmp** | **http**} *num* **client-auth none** **service** {**cwmp** | **http**} *num* **ssl client-auth client-cert-css-ext**

（注）　HTTP ベースのデバイス認証がディセーブルで、汎用の証明書を使用するようにクライアント証明書認証がイネーブルになっている場合、デバイスは信頼済み、または事前認証済みであると見なされ、DPE は CWMP Inform メッセージからのデータを使用して、信頼されるデバイスのアイデンティティを確立します。