CHAPTER 12

CWMP サービスの設定

この章では、Broadband Access Center（BAC）で CWMP サービスを設定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。

（注） プロビジョニング API を使用すると、管理者のユーザ インターフェイスから可能な多くの操作を実行できます。

CWMP サービスの設定値

CWMP は、GetParameterValues、SetParameterValues などの、リモート プロシージャ コール（RPC）のセットの仕様です。これらの RPC は、BAC が顧客宅内装置（CPE）を管理するためにパラメータの読み取りまたは書き込みを行う汎用メカニズムを定義します。次のパラメータがあります。

- デバイス構成情報
- ステータス情報
- パフォーマンス統計情報

DPE CLI を使用すると、DPE 上で CWMP 機能をイネーブルまたはディセーブルにできます。

DPE 上で設定できる機能には、次のものがあります。

- HTTP ベースの Basic または Digest 認証
- 証明書ベースの認証
- HTTP over SSL/TLS サービス設定
- 不明なデバイスの処理
- デバッグ設定
- セッション管理設定
- CWMP サービス設定
- HTTP ファイル サービス設定

これらのプロパティを設定する方法については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

DPE でのサービス ポートの設定

CWMP サービスがデバイスと通信するときに使用するポートを設定できます。要件に応じて、CWMP サービスの各インスタンス（CWMP RPC サービスおよび HTTP ファイル サービス）を個別に設定できます。表 12-1 は、サービスごとに、ポートを設定する方法を示しています。

表 12-1　サービスポートの設定

コマンド	構文の説明	デフォルト
CWMP RPC サービスの設定		
service cwmp *num* **port** *port*	• *num*：CWMP サービスを示します。1 または 2 になります。 • *port*：サービスで使用するポート番号を示します。	デフォルトでは、CWMP サービスは次のポートでリッスンするように設定されています。 • サービス 1 の場合：ポート 7547 • サービス 2 の場合：ポート 7548

	例：
	`dpe# service http 1 port 7547`
	`% OK (Requires DPE restart "# dpe reload")`	

表 12-1　サービスポートの設定（続き）

コマンド	構文の説明	デフォルト
HTTP ファイル サービスの設定		
service http *num* **port** *port*	• *num*：HTTP ファイル サービスを示します。1 または 2 になります。 • *port*：サービスで使用するポート番号を示します。	デフォルトでは、HTTP ファイル サービスは次のポートでリッスンするように設定されています。 • サービス 1 の場合：ポート 7549 • サービス 2 の場合：ポート 7550

	例：
	`dpe# service http 1 port 7549`
	`% OK (Requires DPE restart "# dpe reload")`	

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

接続要求サービス

接続要求は、DPE との CWMP セッションを確立するようにデバイスに指示します。BAC 接続要求サービスを使用すると、デバイス上で設定を有効にしたり、デバイスのファームウェア変更を実行したり、デバイスで即時操作を実行したりできます。

DPE によって開始された場合、接続要求は、DPE がデバイスとのセッションを確立するときに使用可能な唯一の方法です。セッションが確立されると、デバイスまたは DPE は、デバイス操作および設定変更を含む、任意の RPC を実行できます。

図 12-1　BAC における接続要求

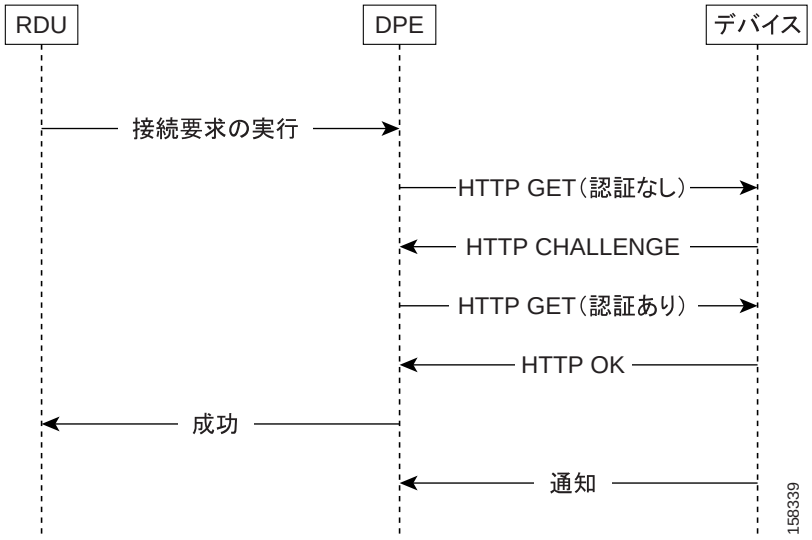

図 12-1 では、BAC における接続要求のフローについて説明します。RDU は、接続要求を、デバイスのプロビジョニング グループ内で使用可能な最適な DPE に委任します。接続要求が終了すると、DPE は結果を RDU に通知します。

接続要求オプションの設定

BAC を使用すると、次のプリファレンスを設定することにより接続要求の動作を制御できます。

- 認証の設定
- 接続要求方式の設定
- 到達可能性の設定

（注）　プリファレンスは、デバイス オブジェクト上またはそのプロパティ階層内で設定できます。

認証の設定

RDU のデバイス オブジェクトで設定した 2 つのプロパティは、認証に影響を及ぼします。これらのプロパティは次のとおりです。

- API の場合：
  - `IPDeviceKeys.CONNECTION_REQUEST_USERNAME`
  - `IPDeviceKeys.CONNECTION_REQUEST_PASSWORD`
- 管理者のユーザ インターフェイスの場合：
  - **Devices > Modify Device >** Connection Request User Name field
  - **Devices > Modify Device >** Connection Request Password field

    Add Device ページでデバイスを追加するときに接続要求ユーザ名およびパスワードを設定したり、Modify Device ページでユーザ名およびパスワードを変更したりすることもできます。

どちらのプロパティも、DPE-CPE 認証に使用される接続要求ユーザ名およびパスワードを制御します。このユーザ名とパスワードは、DPE とデバイスの間で CWMP セッションを認証するために使用されるユーザ名およびパスワードとは異なります。これらのプロパティは単一のデバイス用なので、デバイス オブジェクトでしか設定できません。

接続要求パスワードを指定していない場合は、DPE に対してデバイスを認証する CWMP セッション パスワードが使用されます。接続要求ユーザ名も指定されていない場合、デバイス ID が使用されます。

（注）　接続要求の認証中に認証チャレンジを発行するかどうかは、図 12-1 が示すとおり、デバイスによって異なります。DPE は、HTTP Digest 認証によりチャレンジされることを予期しています。接続要求の処理のための DPE 設定はありません。

（注）　API プロパティでは、デバイス パラメータは自動的には更新されません。対応する値をデバイス上で事前に設定するか、これらのプロパティを参照可能な設定テンプレートを使用して値を設定する必要があります。

接続要求方式の設定

プロビジョニング API または管理者のユーザ インターフェイスを使用して、BAC が接続要求の実行を試みる方式を指定できます。選択した方式により、デバイスにアクセスするときに使用される接続要求 URL を BAC が判別する方法が決定されます。

API プロパティ `IPDeviceKeys.CONNECTION_REQUEST_METHOD` では、接続要求の方式を指定します（各方式については次に説明します）。

（注）　このプロパティは、デバイス階層の任意の場所で指定できます。

管理者のユーザ インターフェイスを使用してデフォルトの接続要求方式を設定するには、**Configuration > Default > CWMP Defaults** を選択し、ドロップダウン リストのオプションを選択します。

BAC は、接続要求を設定する次の 3 つの方式をサポートします。

- Discovered
- Using FQDN
- Using IP

方式によってパフォーマンス レベルと管理性が異なるので、接続要求の方式を選択するときは、パフォーマンスと管理性を考慮する必要があります。接続要求の方式を選択する前に、各方式の推奨事項を参照してください。

Discovered 方式の接続要求

Discovered 方式では、CPE が DPE と対話する間、DPE がデバイスと対話するたびに、InternetGatewayDevice.ManagementServer.ConnectionRequestURL パラメータに対応するデバイスの接続要求 URL を検出するようにデータ同期命令を修正します。RDU は、このパラメータに対するすべての更新を記録し、接続要求を行うときにその情報を使用します。

（注）　接続要求が試行される前に、この値が検出されている必要があります。

推奨事項：このパラメータ値はデバイスの WAN IP アドレスが変更されるたびに変化し、すべての更新が RDU に格納される必要があるので、これは接続要求で最適な方法ではありません。

Use FQDN 方式の接続要求

Use FQDN 方式は、RDU でそのデバイスに指定されている完全修飾ドメイン名（FQDN）を使用して、デバイスの接続要求 URL を構築します。この方式は、FQDN を、API で次のプロパティに指定されている値とともに使用します。

- `IPDeviceKeys.CONNECTION_REQUEST_PORT`
- `IPDeviceKeys.CONNECTION_REQUEST_PATH`

これらのプロパティは、管理者のユーザ インターフェイス上でも指定できます。手順は次のとおりです。

ステップ 1　**Devices > Manage Devices** を選択します。

ステップ 2　次の方法のいずれかを使用します。

- デバイス レコードを追加します。追加するには、**Add** ボタンをクリックします。Add Device ページが表示されます。
- デバイス レコードを検索します。検索するには、Search Type を指定し、選択する検索タイプに固有の画面コンポーネントの値を入力します。デバイスのリストが表示されます。目的のデバイスに対応する Identifier リンクをクリックします。Modify Device ページが表示されます。

ステップ 3　Property Name ドロップダウン リストから /IPDevice/connectionRequestPort および /IPDevice/connectionRequestPath のプロパティを選択し、Property Value フィールドに適切な値を入力します。

（注） /IPDevice/connectionRequestPort および /IPDevice/connectionRequestPath の API 定数は、それぞれ `IPDeviceKeys.CONNECTION_REQUEST_PORT` および `IPDeviceKeys.CONNECTION_REQUEST_PATH` です。

ステップ 4　**Add** をクリックします。

（注） ポートとパスのプロパティは、プロパティ階層の任意の場所で指定できます。

推奨事項：Use FQDN 方式は、デバイスの正しい IP アドレスを使用して更新している DNS に依存するので、デバイスの IP アドレスが変更されるたびに BAC を更新する必要はありません。それ以降、このオプションは、接続要求で最もスケーラブルなオプションです。

Use IP 方式の接続要求

Use IP 方式は、Discovered 方式と同じメカニズムを使用してデバイスの WAN IP アドレスを検出します。その後、次の API プロパティの値を使用してデバイスの接続要求 URL を構築します。

- `IPDeviceKeys.CONNECTION_REQUEST_PORT`
- `IPDeviceKeys.CONNECTION_REQUEST_PATH`

これらのプロパティは、管理者のユーザ インターフェイス上でも指定できます。手順は次のとおりです。

ステップ 1　**Devices > Manage Devices** を選択します。

ステップ 2　次の方法のいずれかを使用します。

- デバイス レコードを追加します。追加するには、**Add** ボタンをクリックします。Add Device ページが表示されます。
- デバイス レコードを検索します。検索するには、Search Type を指定し、選択する検索タイプに固有の画面コンポーネントの値を入力します。デバイスのリストが表示されます。目的のデバイスに対応する Identifier リンクをクリックします。Modify Device ページが表示されます。

ステップ 3　Property Name ドロップダウン リストから /IPDevice/connectionRequestPort および /IPDevice/connectionRequestPath のプロパティを選択し、Property Value フィールドに適切な値を入力します。

（注）　/IPDevice/connectionRequestPort および /IPDevice/connectionRequestPath の API 定数は、それぞれ `IPDeviceKeys.CONNECTION_REQUEST_PORT` および `IPDeviceKeys.CONNECTION_REQUEST_PATH` です。

ステップ 4　**Add** をクリックします。

推奨事項：Use IP 方式は、デバイスの WAN IP アドレスを持つ RDU に依存しているので、接続要求を試行する前に WAN IP アドレスが検出される必要があります。また、デバイスの WAN IP アドレスは変更されるので、新しい IP アドレスを使用して RDU を更新する必要があります。したがって、このオプションは、接続要求に最適な方式ではありません。

接続要求のディセーブル化

接続要求サービスをディセーブルにするように選択できます。このオプションは、デバイスが NAT を使用しており、接続要求が可能ではない場合に便利です。

（注）　接続要求がディセーブルになっている場合でも、API デバイス操作を介して ConnectionRequest を使用できます。

到達可能性の設定

到達可能性は、接続要求の設定で重要な役割を果たします。デバイスの報告された IP アドレスとその発信元 IP アドレスが一致しない場合、それはデバイスが NAT 標準を使用していることを示すので、BAC は接続要求を拒否します。そのため、通常、設定要求は正常に実行されません。この動作は、管理者のユーザ インターフェイスまたは API から変更できます。

API を使用して、`IPDeviceKeys.FORCE_ROUTABLE_IP_ADDRESS` プロパティを *true* に設定し、デバイスの発信元 IP アドレスに不一致があるかどうかに関係なく接続要求を許可します。

管理者のユーザ インターフェイスから次の手順に従います。

ステップ 1　**Devices > Manage Devices** を選択します。

ステップ 2　次の方法のいずれかを使用します。

- デバイス レコードを追加します。追加するには、**Add** ボタンをクリックします。Add Device ページが表示されます。
- デバイス レコードを検索します。検索するには、Search Type を指定し、選択する検索タイプに固有の画面コンポーネントの値を入力します。デバイスのリストが表示されます。目的のデバイスに対応する Identifier リンクをクリックします。Modify Device ページが表示されます。

ステップ 3　Property Value ドロップダウン リストから、/IPDevice/forceRouteIPAddress を選択し、プロパティ値を設定します。

（注）　/IPDevice/forceRouteIPAddress の API 定数は、`IPDeviceKeys.FORCE_ROUTABLE_IP_ADDRESS` です。

ステップ 4　**Add** をクリックします。

（注）　このプロパティは、デバイスの階層の任意の場所で指定できます。

デバイスからのデータの検出

この項では、Discovery of Data 機能について説明します。この機能は、デバイスからパラメータの事前定義セットを取得し、それらのパラメータを後で使用できるように RDU に格納します。検出されたこのデータを使用すると、デバイスの一部のキー属性とその現在の構成を提供することにより、デバイス ファームウェアおよびデバイス構成を管理できます。検出されたパラメータは、これらの値がデバイス上で変更されるたびに RDU で更新されます。

データ検出は、RDU で設定できます。RDU は、以前の検出プロセス中に各デバイスについて検出された、検出ポリシー命令およびパラメータの値を適切な DPE に転送します。デバイスとの対話中に、DPE は、デバイスに固有の検出ポリシー命令を参照し、どのパラメータを検出する必要があるかを判別します。

パラメータ値が検出されると、既存のデバイス パラメータが、デバイスにすでに格納されているパラメータと比較されます。値が変更されている場合、または初めて取得される場合、このデータは RDU で更新されます。更新を受信するときに RDU が使用可能でない場合、新しく検出されたデータは破棄され、次回デバイスが DPE に接続するときにデータ検出の全プロセスが開始されます。

検出されたパラメータはデバイス レコードに格納され、管理者のユーザ インターフェイスを使用して表示するか、API `IPDevice.getDetails()` コールを介して取得することができます。検出されたパラメータをユーザ インターフェイス経由で表示するには、プライマリ ナビゲーション バーの **Devices** タブの下にある Manage Devices ページにアクセスします。検索オプションを使用してデバイスを見つけ、デバイスに対応する **View Details** アイコン（  ）をクリックします。デバイスについて検出されたパラメータの詳細を示す Device Details ページが表示されます。

以降の項では、次のトピックについて説明します。

- データ検出の設定（P.12-9）
- データ検出のトラブルシューティング（P.12-10）

データ検出の設定

データ検出ポリシーは、RDU から設定します。このポリシーには、いくつかのパラメータが含まれており、そのパラメータがチェックされるタイミングは次のとおりです。

- デバイスにアクセスするたび
- ファームウェアのアップグレード時のみ

データ検出プロセスは、デバイスが DPE にアクセスするたびに実行されるので、デバイスが新しいバージョンのファームウェアを報告した場合に限り、特定のパラメータの検証が実行されるように設定できます。このチェックにより、ファームウェアのアップグレード時にしか値が変更されないパラメータを検証する必要はなくなります。たとえば、ファームウェアのアップグレードがない場合、デバイス モデル名は変更されません。そのため、ファームウェアのアップグレードが実行されない限り、デバイスでこのパラメータをチェックする必要はありません。

BAC は、データ検出のデフォルト設定で出荷されます。このデフォルト設定は、次の 2 つの方法で拡張できます。

- パラメータをカスタム リストに追加する。
- デフォルトのリストを変更する。ただし、このオプションは推奨されていません。

アクセスごとにチェックされるパラメータの設定

`ServerDefaultsKeys.CWMP_DISCOVER_PARAMETERS` プロパティを使用すると、デバイスが DPE に接続するたびにパラメータが検出されるように、パラメータのデフォルトのリストを設定できます。`IPDeviceKeys.CWMP_CUSTOM_DISCOVER_PARAMETERS` プロパティの値としてパラメータのカンマ区切りリストを提供することにより、デフォルトのリストにパラメータを追加できます。デフォルトのリストには、次のパラメータが含まれます。

- Inform.DeviceId.Manufacturer
- Inform.DeviceId.ManufacturerOUI
- Inform.DeviceId.ProductClass
- InternetGatewayDevice.DeviceInfo.HardwareVersion
- InternetGatewayDevice.DeviceInfo.SoftwareVersion
- InternetGatewayDevice.ManagementServer.ParameterKey

次に例を示します。

```
IPDeviceKeys.CWMP_CUSTOM_DISCOVER_PARAMETERS=
InternetGatewayDevice.ManagementServer.URL,
InternetGatewayDevice.ManagementServer.PeriodicInformEnable
```

ファームウェアのアップグレード時にチェックされるパラメータの設定

`ServerDefaultsKeys.CWMP_FIRMWARE_CHANGED_CPE_PARAMETERS` プロパティを使用すると、ファームウェアのアップグレードのたびにパラメータが検出されるように、パラメータのデフォルトのリストを設定できます。このデフォルト リストには、InternetGatewayDevice.DeviceInfo.ModelName パラメータが含まれます。

このリストをカスタマイズしてさらに多くのパラメータを含めるには、`IPDeviceKeys.CWMP_CUSTOM_FIRMWARE_CHANGED_PARAMETERS` プロパティを使用します。

データ検出のトラブルシューティング

表 12-2 に示されているタスクのいずれかを使用して、DPE CLI からデータ検出のトラブルシューティングを実行することもできます。

表 12-2　DPE からのデータ検出のトラブルシューティング

タスク	使用するコマンド	説明
情報レベル ロギングをイネーブルにする	dpe# **log level 6-info**	情報メッセージを表示します。
デバイス ログを表示する	dpe# **show log** dpe# **show log last 100** dpe# **show log run** **（注）** リストされている 3 つのコマンドのどれでも使用できます。	直近のいくつかの DPE ログ エントリを表示します。 DPE ログの最後の 100 行を表示します。 実行中の DPE ログを表示します。

	例
	
	この例は説明を目的としているため、**show log run** コマンドの出力は短縮されています。
	
	`dpe# show log run`
	
	`% Press <enter> to stop.`
	
	`2006 08 04 00:47:01 EDT: %BAC-DPE-6-0104: Obtained configuration for device [0014BF-CJJ005B00009] from RDU.`
	
	`2006 08 04 00:47:21 EDT: %BAC-CWMP-6-5129: Device [0014BF-CJJ005B00009]. Source IP [10.86.147.149]. Retrieving [1] discovered CPE parameters.`
	
	`2006 08 04 00:47:21 EDT: %BAC-CWMP-6-5107: Device [0014BF-CJJ005B00009]. Source IP [10.86.147.149]. Sent [GetParameterValues] message.`
	
	`2006 08 04 00:47:21 EDT: %BAC-CWMP-6-5106: Device [0014BF-CJJ005B00009]. Source IP [10.86.147.149]. Received [GetParameterValuesResponse] message.`
	
	`2006 08 04 00:47:21 EDT: %BAC-CWMP-6-5120: Device [0014BF-CJJ005B00009]. Source IP [10.86.147.149]. New data discovered from CPE. Queued update of [7] parameters to RDU.`
	
	（注） **show log** コマンドの出力が、ここに示されているサンプル出力と同様であれば、データ検出は正常に完了しました。	

デバッグをイネーブルにする	dpe# **debug on** dpe# **debug service cwmp** *num* **data-sync** *num* : CWMP 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。	CWMP サービスのデータ同期プロセスのデバッグ ロギングをイネーブルにします。
特定のデバイスのデータ検出設定を表示する	dpe# **show device-config** *device-id* *device-id* : デバイスの ID を指定します。	DPE でキャッシュされるデバイス設定を表示します。

（注） これらのコマンドの使用方法の詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。

プロビジョニング グループのスケーラビリティとフェールオーバー

この項では、この BAC リリースで提供されるスケーラビリティとフェールオーバーの機能について説明します。

BAC のスケーラビリティとフェールオーバーは、数百万のデバイスが配備されているネットワークを含む、実質的にどのような規模のネットワークにも適合する高度のアベイラビリティを実現します。BAC は、DPE 冗長性やフェールオーバーの保護などの重要な機能も提供します。

BAC 配備のスケーラビリティは、プロビジョニング グループを通じて実現されます。各プロビジョニング グループは、一群の CPE と通信する複数の冗長 DPE サーバから成るクラスタです。プロビジョニング グループを使用すると、各プロビジョニング グループがデバイスのサブセットにのみ責任を負うようになるので、BAC ネットワークのスケーラビリティが高まります。このデバイスのパーティション化は、地域別のグループ化や、サービス プロバイダーによって定義されている他のポリシーと並行して実現できます。

配備を拡張するには、管理者は次の操作を行うことができます。

- 既存の DPE サーバ ハードウェアをアップグレードする
- DPE サーバをプロビジョニング グループに追加する
- プロビジョニング グループを追加して、デバイスをそれらのグループに再分散する

BAC は、プロビジョニング グループへのデバイスの明示的な割り当てと自動メンバシップをサポートします。詳細については、P.4-1 の「CPE 管理の概要」を参照してください。

BAC における冗長性

冗長性により、次の事項が保証されます。

- ネットワーク アプリケーションに高いアベイラビリティが提供される。
- ユーザは、単一のポイント障害による長いネットワーク遅延やブラック ホールを経験することがない。

BAC は、ローカル サーバおよび地域別サーバの冗長性をサポートします。

ローカルでの冗長性

BAC プロビジョニング グループは、一群の CPE と通信する複数の冗長 DPE サーバから成るクラスタです。単一の URL が、各プロビジョニング グループを識別します。プロビジョニング グループへのすべての CPE 要求は、使用可能な DPE 間でロードバランスされます。各 DPE は、プロビジョニング グループ内の任意のデバイスを処理できます。

地域別の冗長性

地域別の冗長性を使用すると、地域的な障害が発生した場合に、異なる場所にある DPE が一時的に CPE 要求を処理できるようになります。この配備を推進する最も簡単な方法は、プロビジョニング グループ内の各 DPE を地理的に異なるさまざまな位置に設定することです。そのような設定では、CPE 要求はさまざまな地域にある DPE によって処理され、プロビジョニング グループの範囲内で地域的なフェールオーバーが提供されます。

ただし、配備によっては、CPE 要求を 1 つの場所にあるサーバで処理し、障害が発生した場合にのみ別の場所にある DPE にフェールオーバーする必要が生じる場合があります。そのような場合、別々の地域に存在する DPE を 1 つのプロビジョニング グループ内で見つけ、通常の状態では CPE 要求が特定地域の DPE のサブセットにのみ方向付けられるようにネットワークを設定することができます。

通常は、DNS 技術を使用して地域別の冗長性を設定できます。地域別の冗長性のためにネットワークを設定するには、所定のプロビジョニング名の BAC ホスト名が DPE の 1 つのセットを表す IP アドレスに正常に解決されることを確認します。しかし、そのセット内の DPE サーバのいずれもキープアライブに応答しない場合（ICMP、HTTP GET、または TCP ハンドシェークなど）、DNS サーバは、2 番目のセットに含まれる DPE の IP アドレスに BAC ホスト名を解決する必要があります。次に、CPE 要求は DPE の別のセットに方向付けられます。両方のセットの DPE は同じプロビジョニング グループに属しているので、新しい DPE は要求にすぐに応答できます。それ以降の DNS ルックアップ要求は、使用可能になるとただちに DPE のプライマリ セットにフォールバックされます。

地域別の冗長性は、Cisco 11500 Content Services Switch などのロード バランシング機能を備えるソフトウェアまたはハードウェアを使用して設定できます。

DPE ロード バランシング

BAC は、DPE 冗長性およびロード バランシングの次のメカニズムをサポートします。

- DNS ラウンド ロビンの使用方法（P.12-12）
- ハードウェア ロード バランサの使用方法（P.12-12）

DNS ラウンド ロビンの使用方法

DNS ラウンド ロビン メカニズムを使用すると、DNS サーバは、そのデバイスの自動構成サーバ（ACS）ホスト名を解決するときに DPE IP のリストをシャッフルします。次に、デバイスは、リスト内の最初の IP アドレスを ACS ホスト名として使用します。

サービス プロバイダーが DNS キャッシング サーバを制御しない場合、このオプションは推奨されません。また、停電が発生した場合、多数のデバイスが、DNS サーバによってキャッシュされた同じオーダー IP アドレスを受け取ることによりパフォーマンスが影響を受け、DNS ラウンド ロビンが適切に動作しなくなる場合があります。この問題に対処するため、DNS サーバの TTL を非常に短い値（1 秒）に設定することをお勧めします。

ハードウェア ロード バランサの使用方法

ハードウェア ロード バランサの使用時、ACS URL は、IP アドレスを含むか、または、DNS サーバによって単一の IP アドレスに解決されます。プロビジョニング グループのすべての DPE は、Cisco 11500 Content Services Switch（CSS）などの、ハードウェア ロード バランサの単一仮想 IP アドレスの背後に隠されています。ハードウェア ロード バランサは、その仮想 IP アドレスを、任意の数のロード バランシング アルゴリズムに基づいて固有の DPE IP アドレスに変換するように設定します。ロード バランサの冗長ペアを使用すると、冗長性が向上し、複数のプロビジョニング グループを処理できるようになる場合があります。

プロビジョニング グループへの DPE の追加

この項では、DPE を新しいプロビジョニング グループに追加する方法について説明します。DPE をプロビジョニング グループに追加する場合、次の 3 つのオプションがあります。

- DPE をプロビジョニング グループに追加する。
- デバイスと DPE の間に Cisco 11500 CSS などのハードウェア ロード バランサを使用する配備で、プロビジョニング グループに DPE を追加する。この場合、ロード バランサを更新する必要があります。
- DNS サーバがラウンド ロビンを使用してプロビジョニング グループの複数の DPE に解決する配備で、プロビジョニング グループに DPE を追加する。

DPE をプロビジョニング グループに追加するには、次の手順に従います。

ステップ 1　DPE CLI から DPE を設定します。実行する必要がある設定には、次のものがあります。

- DPE が属する必要があるプロビジョニング グループの指定。次のように入力します。

  ```
  dpe# dpe provisioning-group primary name
  ```

  - *name*：割り当てられているプライマリ プロビジョニング グループを示します。
- DPE が接続する RDU の指定。次のように入力します。

  ```
  dpe# dpe rdu-server {host | ip} port
  ```

  - *host*：RDU が実行しているホストの FQDN を示します。
  - *ip*：RDU の IP アドレスを示します。
  - *port*：DPE 接続で RDU がリッスンするポート番号を示します。デフォルトでは、このポート番号は 49187 です。
- 特定のインターフェイスの FQDN の指定。プロビジョニング FQDN は、特定の DPE インターフェイスにアクセスするデバイスに与えられる FQDN です。次のように入力します。

  ```
  dpe# interface ethernet {intf0 | intf1} provisioning fqdn fqdn
  ```

  - *intf0* | *intf1*：イーサネット インターフェイスを示します。
  - *fqdn*：指定されたインターフェイスで設定される FQDN を示します。

    特定のプロビジョニング グループ内では、すべての DPE に対して同じ FQDN を使用する必要があります。DPE がロード バランサの背後に位置する場合は、ロード バランサの FQDN をインターフェイス FQDN として使用し、同一のロード バランシング グループに属するすべての DPE で FQDN が同じであることを確認します。

（注）　CWMP サービスおよび HTTP ファイル サービスを 1 つの DPE で設定して、他の DPE の設定と一致させる必要もあります。設定オプションの詳細については、『*Cisco Broadband Access Center DPE CLI Reference, Release 3.0*』を参照してください。P.3-1 の「設定のワークフローとチェックリスト」も参照してください。

ステップ 2　**dpe start** コマンドを使用して DPE を起動し、DPE が同期をとり、RDU からのデバイス設定命令を読み込めるようにします。

ステップ 3　ロード バランサを使用している場合、オプションで、DPE アドレスをロード バランサに追加します。

ステップ 4　DNS ラウンド ロビン技術を使用している場合、オプションで、DPE アドレスを DNS サーバに追加します。

Cisco CSS を使用する DPE ロード バランシング

この項では、BAC トラフィックのロード バランシングのために Cisco 11501 CSS を設定する方法について説明します。Cisco CSS フロントエンド サーバは、BAC がサポートする CWMP デバイスからの着信 TCP トラフィックを処理し、Cisco CSS バックエンド サーバ（この場合は DPE）全体で TCP セッションをロードバランシングします。

（注）　Cisco CSS が設定されている場合、Cisco CSS はフロントエンド サーバとバックエンド サーバで HTTP over SSL（この項では SSL と呼びます）をサポートします。

初期 CSS 設定

Cisco CSS に初めて電源を入れると起動時設定メニューが表示されます。これは、イーサネット管理ポートの IP アドレス、サブネット マスク、デフォルトのゲートウェイといった最小設定を行うためのメニューです。この情報は設定しないでください。

次に、デフォルトのユーザ名（**admin**）とパスワード（**system**）を変更するよう求めるメッセージが表示されます。オプションで、これらの資格情報を変更できます。ユーザ名の長さは 1 文字から 16 文字、パスワードの長さは 6 文字から 16 文字にする必要があります。

次に、追加の設定情報を求めるメッセージが表示されます。実行時設定および起動時設定を後でクリアする場合は、追加の設定情報を入力する必要があります。

例 12-1 は、Cisco CSS の初期設定を示しています。

例 12-1　Cisco CSS の初期設定

```
Checking for Existing Config...

No startup-config was found, continue with the setup script [y/n]? y

Note: Pressing 'q' after any prompt quits setup.
      Pressing <CR> after any [y/n] defaults to 'y'.


Warning: All circuit VLAN IP addresses must be on a different
         subnet than the Ethernet Mgt port IP address.
         The existing Ethernet Mgt port IP address is: 0.0.0.0

Add an IP address to VLAN1:     [default = 192.168.10.1]? 10.86.147.51
Add an IP subnet mask to VLAN1: [default = 255.255.255.0]? 255.255.255.224

Would you like to specify a default gateway?  [y/n]? y

Warning: The default gateway IP address must be on the same subnet
         as VLAN1. VLAN1 IP address is: 10.86.147.51

Add IP address for default gateway: [default = 10.86.147.2]? 10.86.147.33
Pinging the default gateway: 0% Success.

Which feature do you want to configure?
[1] Layer3 load balancing
[2] Layer5 load balancing
[3] Proxy cache
[4] Transparent cache
[5] Exit script
Enter the number of the feature you want to configure: 5

Showing the Running Config

CSS11501#  show running-config
!Generated on 07/17/2006 13:00:30
!Active version: sg0750103

configure


!************************** GLOBAL **************************
  ip route 0.0.0.0 0.0.0.0 10.86.147.33 1

!************************* CIRCUIT **************************
circuit VLAN1

  ip address 10.86.147.51 255.255.255.224

CSS11501#
```

VLAN 1 はデフォルトの VLAN であるため、Cisco CSS の 8 個のポートはすべて Cisco CSS のフロントエンド ポートとして使用できます。しかし、この例の目的に合せて、イーサネット ポート 0 はフロントエンド サーバ ポートとして使用されます。フロントエンド ルータまたはスイッチは、顧客のネットワークから Cisco CSS または DPE にアクセスするのを許可するために使用します。Cisco CSS に接続されたこのスイッチまたはルータ上のインターフェイスの IP アドレスは、デフォルトのゲートウェイの Cisco CSS で設定されたアドレスです（10.86.147.33）。FTP サーバ（IP アドレス 10.86.147.53）は、Cisco CSS のイーサネット ポート 8 に接続されます。この FTP サーバは、後の設定で使用します。

Cisco CSS での DPE ロード バランシングの設定

この項で説明する設定を使用すると、バックエンド DPE サーバ間の顧客ネットワーク内で、デバイスからのトラフィックの簡易ロード バランシングを実行できます。使用されるロード バランシングのアルゴリズムは、単純ラウンドロビンです。CWMP デバイスは HTTP を使用して Cisco CSS に接続し、接続は HTTP 形式で DPE サーバに渡されます。

（注）　CSS の設定の詳細については、『*Cisco Content Services Switch SSL Configuration Guide (Software Version 7.40)*』を参照してください。

例 12-2　Cisco CSS での DPE ロード バランシングの設定

次の例は、同じサブネット上の 2 台の DPE サーバを使用する設定を示しています。この例の DPE サーバの IP アドレスは 11.32.0.26/12 および 11.32.0.27/12 です。イーサネット ポートに割り当てられたバックエンド サーバ Cisco CSS VLAN の IP アドレスは 11.32.0.1/12 です。コンテンツ ルールで設定されている VIP アドレスおよび TCP ポートは、10.86.147.52 および 7547 です。CWMP デバイスが Cisco CSS に接続し、DPE 間のロード バランシングを実行するときに使用する URL は、*http://10.86.147.52:7547/acs* です。

ステップ 1　設定モードで Cisco CSS にログインします。

```
CSS11501#config t<cr>
```

ステップ 2　DPE が使用する 2 つのインターフェイスを設定し、次のコマンドを使用してそれらを VLAN2 に割り当てます。

```
CSS11501(config)#interface e1<cr>
CSS11501(config-if[e1])#bridge vlan 2<cr>
CSS11501(config-if[e1])#interface e2<cr>
CSS11501(config-if[e2])#bridge vlan 2<cr>
CSS11501(config-if[e2]}#exit<cr>
```

ステップ 3　2 つの DPE が使用する VLAN 2 を設定します。

```
CSS11501(config)#circuit VLAN2<cr>
CSS11501(config-circuit[VLAN2])#ip address 11.32.0.1/12<cr>
Create ip interface <11.32.0.1>, [y/n]:y<cr>
CSS11501(config-circuit-ip[VLAN2-11.32.0.1])#exit<cr>
CSS11501(config-circuit[VLAN2])#exit<cr>
```

ステップ 4　それぞれの DPE に割り当てられるサービスを設定し、サービスを有効にします。

（注）　2 つのサービスは後にコンテンツ ルールに追加され、コンテンツ ルールは 2 つの DPE にわたって HTTP トラフィックをロード バランシングするように設定されます。

```
CSS11501(config)# service DPE-1<cr>
Create service <DPE-1>, [y/n]:y
CSS11501(config-service[DPE-1])#keepalive type tcp<cr>
CSS11501(config-service[DPE-1])#keepalive port tcp 7547<cr>
CSS11501(config-service[DPE-1])#ip address 11.32.0.26<cr>
CSS11501(config-service[DPE-1])#active<cr>
CSS11501(config-service[DPE-1])# service DPE-2<cr>

Create service <DPE-2>, [y/n]:y
CSS11501(config-service[DPE-2])# keepalive type tcp<cr>
CSS11501(config-service[DPE-2])# keepalive port 7547<cr>
CSS11501(config-service[DPE-2])# ip address 11.32.0.27<cr>
CSS11501(config-service[DPE-2])# active<cr>
CSS11501(config-service[DPE-2])#exit<cr>
```

ステップ 5　着信 HTTP トラフィックをロード バランシングするために使用されるコンテンツ ルールを含むオーナーを設定します。オーナー名として任意の名前を選択できます。

```
CSS11501(config)# owner User1<cr>
Create owner <User1>, [y/n]:y
CSS11501(config-owner[User1])#
```

ステップ 6　次のコマンドを使用してコンテンツ ルールを設定し、それを有効にします。

（注）　コンテンツ ルールの VIP アドレスは、CWMP デバイスが Cisco CSS に接続するときに使用する IP アドレスです。

```
CSS11501(config-owner[User1])# content Clear-text<cr>
Create content <Clear-text>, [y/n]:y
CSS11501(config-owner-content[User1-Clear-text])# protocol tcp<cr>
CSS11501(config-owner-content[User1-Clear-text])# port 7547<cr>
CSS11501(config-owner-content[User1-Clear-text])# add service DPE-1<cr>
CSS11501(config-owner-content[User1-Clear-text])# add service DPE-2<cr>
CSS11501(config-owner-content[User1-Clear-text])# vip address 10.86.147.52<cr>
CSS11501(config-owner-content[User1-Clear-text])#active<cr>
CSS11501(config-owner-content[User1-Clear-text])#exit<cr>
CSS11501(config-owner[User1])#exit<cr>
CSS11501(config)#exit<cr>
CSS11501#
```

ステップ 7　これで、設定は完了しました。実行時設定を起動時設定にコピーし、Cisco CSS のリブートのたびにロードされる永続的な設定レコードを作成します。

```
CSS11501# copy running-config startup-config<cr>
Working..(\) 100%
CSS11501#
```

Cisco CSS でのクライアント証明書認証の設定

次の例では、デバイスは SSL を使用して Cisco CSS に接続します。接続は、HTTP 形式で DPE サーバまで渡されます。Cisco CSS は、デバイスから送信された証明書を認証します。オプションで、証明書が HTTP ヘッダー経由で DPE に転送されるようにすることができます。

例 12-3　Cisco CSS でのクライアント証明書認証の設定

次の例は、同じサブネット上の 2 台の DPE サーバを使用する設定を示しています。この例の DPE サーバの IP アドレスは、11.32.0.26/12 および 11.32.0.27/12 です。DPE が接続するイーサネット ポートに割り当てられたバックエンド サーバ Cisco CSS VLAN の IP アドレスは 11.32.0.1/12 です。コンテンツ ルールで設定されている VIP アドレスおよび TCP ポートは、10.86.147.52 および 7548 です。デバイスが Cisco CSS に接続するときに使用する URL は、*http://10.86.147.52:7548/acs* です。

ステップ 1　設定モードで Cisco CSS にログインします。

```
CSS11501#config t<cr>
```

ステップ 2　DPE が使用する 2 つのインターフェイスを設定し、次のコマンドを使用してそれらを VLAN2 に割り当てます。

```
CSS11501(config)#interface e1<cr>
CSS11501(config-if[e1])#bridge vlan 2<cr>
CSS11501(config-if[e1])#interface e2<cr>
CSS11501(config-if[e2])#bridge vlan 2<cr>
CSS11501(config-if[e2]}#exit<cr>
```

ステップ 3　2 つの DPE が使用する VLAN 2 を設定します。

```
CSS11501(config)#circuit VLAN2<cr>
CSS11501(config-circuit[VLAN2])#ip address 11.32.0.1/12<cr>
Create ip interface <11.32.0.1>, [y/n]:y<cr>
CSS11501(config-circuit-ip[VLAN2-11.32.0.1])#exit<cr>
CSS11501(config-circuit[VLAN2])#exit<cr>
```

ステップ 4　それぞれの DPE に割り当てられるサービスを設定し、サービスを有効にします。

（注）　2 つのサービスは後にコンテンツ ルールに追加され、コンテンツ ルールは 2 つの DPE にわたって HTTP トラフィックをロード バランシングするように設定されます。

```
CSS11501(config)# service DPE-1<cr>
Create service <DPE-1>, [y/n]:y
CSS11501(config-service[DPE-1])#keepalive type tcp<cr>
CSS11501(config-service[DPE-1])#keepalive port tcp 7547<cr>
CSS11501(config-service[DPE-1])#ip address 11.32.0.26<cr>
CSS11501(config-service[DPE-1])#active<cr>
CSS11501(config-service[DPE-1])# service DPE-2<cr>

Create service <DPE-2>, [y/n]:y
CSS11501(config-service[DPE-2])# keepalive type tcp<cr>
CSS11501(config-service[DPE-2])# keepalive port 7547<cr>
CSS11501(config-service[DPE-2])# ip address 11.32.0.27<cr>
CSS11501(config-service[DPE-2])# active<cr>
CSS11501(config-service[DPE-2])#exit<cr>
```

ステップ 5　着信 HTTP トラフィックをロード バランシングするために使用されるコンテンツ ルールを含むオーナーを設定します。オーナー名として任意の名前を選択できます。

```
CSS11501(config)# owner User1<cr>
Create owner <User1>, [y/n]:y
CSS11501(config-owner[User1])#
```

ステップ 6　FTP サーバの ftp レコードを設定します。ftp レコードは、デバイスのルート証明書をダウンロードするために使用されます。Cisco CSS はこの設定を使用して、デバイスからのクライアント証明書と、Cisco CSS サーバを認証するためにデバイスに送信される Cisco CSS サーバ証明書および秘密鍵を認証します。

（注）　この例で使用する ssl-machine という名前の ftp レコードは、ユーザ定義の名前であり、証明書が Cisco CSS にロードされたときに FTP サーバの参照としてのみ使用されます。

この例の目的に合せて、FTP 接続を作成するために使用されるユーザ名とパスワードは、それぞれ root および cisco です。証明書と鍵が格納される FTP サーバ上のディレクトリは、*/var/tmp/ftp* です。

```
CSS11501(config)#ftp-record ssl-machine 10.86.147.53 root "cisco" /var/tmp/ftp<cr>
CSS11501(config)#exit<cr>
CSS11501#
```

ステップ 7　FTP サーバから証明書と鍵をロードします。

この例で使用する CWMP ルート証明書は cwmp_root.cer、Cisco CSS サーバ証明書は css.cer、Cisco CSS 秘密鍵は css.key です。

```
CSS11501#copy ssl ftp ssl-machine import cwmp_root.cer PEM "password"<cr>
CSS11501#copy ssl ftp ssl-machine import css.cer PEM "password"<cr>
CSS11501#copy ssl ftp ssl-mcahine import css.key PKCS12 "password"<cr>
```

（注）　この例で使用される証明書は PEM 形式、Cisco CSS 秘密鍵は PKCS12 形式です。これらのファイルのサポートされる形式は、DER、PEM、および PKCS12 です。ユーザ定義可能なパスワード（**password**）は、データ暗号規格ファイル内のファイルを符号化します。

ステップ 8　ロードした証明書と鍵を、後で SSL プロキシ リストを設定するときに参照可能な名前に関連付けます。次のコマンドを使用します。

```
CSS11501#config t<cr?
CSS11501(config)#ssl associate cert ca-root-cert cwmp_root.cer <cr>
CSS11501(config)#ssl associate cert css-server-cert css.cer <cr>
CSS11501(config)#ssl associate rsakey css-server-key css.key <cr>
CSS11501(config)#
```

ステップ 9　SSL プロキシ リストを設定します。このリストでは、次の事項を設定します。

- 使用される証明書と鍵
- バックエンド DPE サーバ向けのコンテンツ ルールの VIP アドレスを使用した優先暗号方式
- デバイスが Cisco CSS に接続するときに使用する VIP アドレスおよび TCP ポート

（注）　プロキシ リストには、任意の名前を定義できます。

```
CSS11501(config)#ssl-proxy-list SSL<cr>
Create Ssl-list <SSL>, [y/n]:y
CSS11501(config-ssl-proxy-list[SSL]}#ssl-server 1<cr>
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 port 7548<cr>
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 rsakey css-server-key<cr>
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 rsacert css-server-cert<cr>
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 cacert ca-root-cert<cr>
CSS11501(config-ssl-proxy-list[SSL]}#ssl-server 1 cipher rsa-with-rc4-128-md5
10.86.147.60 7547
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 authentication enable<cr>
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 vip address 10.86.147.52<cr>
```

ステップ 10　HTTP ヘッダーのクライアント証明書を DPE に渡す場合は、次のコマンドを追加します。

```
CSS11501(config-ssl-proxy-list[SSL])#ssl-server 1 http-header client-cert<cr>
```

ステップ 11　**ssl-proxy-list** を有効にします。

```
CSS11501(config-ssl-proxy-list[SSL])#active<cr>
CSS11501(config-ssl-proxy-list[SSL])#exit<cr>
CSS11501(config)#
```

ステップ 12　**ssl-proxy-list** を参照する SSL サービスを設定します。サービス名は、ユーザが定義できます。

```
CSS11501(config)# service SSL<cr>
Create service <SSL>, [y/n]:y
CSS11501(config-service[SSL])#type ssl-accel<cr>
CSS11501(config-service[SSL])#add ssl-proxy-list SSL <cr>
CSS11501(config-service[SSL])#keepalive type none<cr>
CSS11501(config-service[SSL])#slot 2<cr>
CSS11501(config-service[SSL])#active<cr>
CSS11501(config-service[SSL])#exit<cr>
CSS11501(config)#
```

ステップ 13　**ssl-proxy-list** から出力された VIP アドレスおよびポート番号と一致するコンテンツ ルールを定義します。

```
CSS11501(config)#owner User1<cr>
CSS11501(config-owner[User1])#content Clear-text<cr>
CSS11501(config-owner[User1])#protocol tcp<cr>
CSS11501(config-owner[User1])#port 7547<cr>
CSS11501(config-owner[User1])# vip address 10.86.147.60<cr>
CSS11501(config-owner[User1])#add service DPE-1<cr>
CSS11501(config-owner[User1])#add service DPE-2<cr>
CSS11501(config-owner[User1])#active<cr>
CSS11501(config-owner[User1])#exit<cr>
```

ステップ 14 デバイスから送信された URL 内のアドレスおよびポート番号と一致するコンテンツ ルールを定義します。

```
CSS11501(config)#owner User1<cr>
CSS11501(config-owner[User1])#content SSL
Create content <SSL>, [y/n]:y
CSS11501(config-owner-content[User1-SSL])#protocol tcp<cr>
CSS11501(config-owner-content[User1-SSL])#port 7548<cr>
CSS11501(config-owner-content[User1-SSL])#add service SSL<cr>
CSS11501(config-owner-content[User1-SSL])#vip address 10.86.147.52<cr>
CSS11501(config-owner-content[User1-SSL])#active<cr>
CSS11501(config-owner-content[User1-SSL])#exit<cr>
CSS11501(config-owner[User1])#exit<cr>
CSS11501(config)#exit<cr>
CSS11501#
```

ステップ 15 最後に、実行時設定を起動時設定に保存します。

```
CSS11501#copy running-config startup-config<cr>
Working..(\) 100%
CSS11501#
```