Red Hat Satellite 6.15

Puppet 統合を使用した設定の管理

Satellite で Puppet 統合を設定し、Puppet クラスを使用してホストを設定する

Last Updated: 2024-08-30

Red Hat Satellite 6.15 Puppet 統合を使用した設定の管理

Satellite で Puppet 統合を設定し、Puppet クラスを使用してホストを設定する

Red Hat Satellite Documentation Team
satellite-doc-list@redhat.com

法律上の通知

概要

このガイドでは、Satellite と Puppet の統合の有効化、ホストへの Puppet エージェントの設定、Puppet モジュールのインポート、および Puppet モジュールを使用した、Red Hat Satellite インフラストラクチャーで管理されるホストへの設定の適用方法を説明します。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

フィードバックを提供するには、Red Hat Jira の **Create Issue** フォームを使用します。Jira の問題は Red Hat Satellite Jira プロジェクトに作成され、その進捗状況を追跡できます。

前提条件

- Red Hat アカウント が登録されている。

手順

1. **Create Issue** にアクセスします。Jira でログインエラーが表示された場合は、フォームにリダイレクトされた後、ログインして続行します。
2. **Summary** フィールドと **Description** フィールドに入力します。**Description** フィールドに、ドキュメントの URL、章またはセクション番号、および問題の詳しい説明を入力します。フォーム内の他のフィールドは変更しないでください。
3. **Create** をクリックします。

第1章 PUPPET を使用した設定管理の概要

Puppet を使用して、ホスト設定の管理および自動化を行うことができます。Puppet は宣言型言語を使用してホストの **あるべき状態** を記述します。

複数のホストを同時に管理できるため、Puppet は生産性を高めます。同時に、Puppet を使用するとホストの状態を容易に確認し、場合によってはそれを修正できるため、設定に要する労力が軽減されます。

関連情報

- Open Source Puppet のドキュメント
- Puppet Forge – 事前にビルドされた Puppet モジュールのリポジトリー

1.1. PUPPET と SATELLITE の統合方法

Puppet はサーバー/エージェントアーキテクチャーを使用します。Puppet サーバーは、設定定義を保存する中心的なコンポーネントです。Satellite Server または Capsule は通常 Puppet サーバーと共にデプロイされ、Satellite はそのような Puppet サーバーの External Node Classifier (ENC: 外部ノードの分類子) として機能します。ホストは、Puppet サーバーと通信する Puppet エージェントを実行します。

Puppet エージェントはホストの **ファクト** を収集し、それらを実行ごとに Puppet サーバーに報告します。ホストで **puppet facts** を実行して、Puppet **ファクト** を JSON 形式で表示できます。

Puppet サーバーは **ファクト** を Satellite に転送し、Satellite は後で使用できるようにファクトを保存します。**ファクト** およびその他の定義に基づいて、Satellite は Puppet サーバーへの ENC 応答を構築します。Puppet サーバーは、ENC 応答に基づいて **カタログ** をコンパイルし、**カタログ** を Puppet エージェントに送信します。

Puppet エージェントは、ホストのシステム状態を評価します。Puppet エージェントが、**カタログ** で定義されている **あるべき状態** と実際の状態の間に **ドリフト** と呼ばれる違いを検出すると、ホストの状態の修正を適用します。Puppet エージェントは、修正結果を Puppet サーバーに報告し、Puppet サーバーはそれらを Satellite に報告します。

Puppet モジュール

ホストの **あるべき状態** が **カタログ** で定義されます。**カタログ** は、ホストに割り当てられた 1 つまたは複数の Puppet モジュールの Puppet マニフェストからコンパイルされます。Puppet モジュールは、クラス、マニフェスト、リソース、ファイル、およびテンプレートのコレクションです。Puppet モジュールは、ホスト設定定義のコンポーネントとして機能します。

スマートクラスパラメーター

モジュールがパラメーターの使用をサポートしている場合は、スマートクラスパラメーターを使用して Puppet モジュールのパラメーターをオーバーライドできます。Satellite でパラメーターを **キー/値** ペアとして定義できます。これは、ホストパラメーターや Ansible 変数と同様に動作します。

Puppet 環境

複数の Puppet 環境を作成して、設定定義のバージョンを制御したり定義のバリアントを管理したりすることや、実稼働環境にデプロイする前に定義をテストすることもできます。

統合手順の概要

Satellite と Puppet の統合手順の概要は以下のとおりです。

1. Puppet 統合を有効にします。
2. Puppet エージェントパッケージを Satellite にインポートします。Puppet エージェントパッケージは、Red Hat リポジトリーを有効化 し、アクティベーションキー と コンテンツビュー を使用して、Satellite の他のコンテンツと同様に管理できます。
3. プロビジョニング 時、登録 時、そして 手動 またはリモートジョブの実行により、ホストに Puppet エージェントをインストールします。

関連情報

- コンテンツの管理
- **ホストの管理** ガイドの ホストの登録
- **ホストの管理** ガイドの リモートジョブの設定およびセットアップ

次の手順では、Puppet モジュールを使用して **ntp** サービスをインストール、設定、および管理する方法を概説し、その例を示します。

1.2. サポートされている PUPPET のバージョンとシステム要件

Puppet の統合を開始する前に、サポートされている Puppet のバージョンとシステム要件を確認してください。

サポート対象の Puppet バージョン

Satellite は Puppet 7 をサポートします。ホストの設定に使用される Puppet モジュールが、Puppet 7 と互換性があることを確認してください。

システム要件

Puppet と Satellite の統合を開始する前に、システム要件を満たしていることを確認してください。詳細は、**オープンソース Puppet** ドキュメントの System Requirements for Puppet 7 を参照してください。

1.3. SATELLITE と PUPPET の統合の有効化

デフォルトでは、Satellite に Puppet 統合は設定されていません。状況に応じて統合を有効にする必要があります。これは、Satellite Server または Capsule Server で、Puppet サーバーを管理およびデプロイするように Satellite を設定できることを意味します。さらに、Puppet サーバーを Satellite の外部にデプロイし、レポート、ファクト、External Node Classification (ENC) のために Satellite と統合できます。

手順

1. Puppet 統合を有効にし、Satellite Server に Puppet サーバーをインストールします。

```
# satellite-installer \
--enable-foreman-cli-puppet \
--enable-foreman-plugin-puppet \
--enable-puppet \
--foreman-proxy-puppet true \
--foreman-proxy-puppetca true \
--puppet-server true
```

2. Capsule Server で Puppet 統合を使用する場合は、Puppet 統合を有効にし、Capsule Server に Puppet サーバーをインストールします。

```
# satellite-installer \
--enable-puppet \
--foreman-proxy-puppet true \
--foreman-proxy-puppetca true \
--puppet-server true
```

1.4. ホストのプロビジョニング中の PUPPET エージェントのインストールと設定

プロビジョニングプロセス中に、ホストに Puppet エージェントをインストールして設定できます。Satellite と Puppet を統合するには、設定済みの Puppet エージェントがホストに必要です。

前提条件

- Satellite で Puppet が有効になっている。詳細は、「Satellite と Puppet の統合の有効化」 を参照してください。
- ホストのオペレーティングシステムバージョン用の Red Hat Satellite Client 6 リポジトリーが、Satellite Server 上で同期され、使用するアクティベーションキーで有効になっている。詳細は、**コンテンツの管理** の コンテンツのインポート を参照してください。
- アクティベーションキーがある。詳細は、**コンテンツの管理** の アクティベーションキーの管理 を参照してください。

手順

1. **Hosts** > **Templates** > **Provisioning Templates** に移動します。
2. ホストのプロビジョニング方法に応じて、プロビジョニングテンプレートを選択します。詳細は、**ホストのプロビジョニング** の プロビジョニングテンプレートのタイプ を参照してください。
3. **puppet_setup** スニペットが以下のように含まれていることを確認します。

```
<%= snippet 'puppet_setup' %>
```

このスニペットは、Satellite に同梱されているテンプレート (**Kickstart default** や **Preseed default など)** にすでに含まれている点に注意してください。

4. グローバルパラメーター、ホストグループ、または単一ホストのホストパラメーターを使用して、Puppet エージェントを有効化します。**enable-puppet7** という名前のホストパラメーターを追加し、**boolean** タイプを選択して、値を **true** に設定します。
5. Puppet エージェントの設定を指定します。
   - 統合された Puppet サーバーを使用する場合は、ホストの作成時に Puppet Capsule、Puppet CA Capsule、および Puppet 環境を必ず選択してください。
   - 非統合 Puppet サーバーを使用する場合は、グローバルパラメーターまたはホストグループで次のホストパラメーターを設定するか、ホストの作成時に次のホストパラメーターを設定します。

- **puppet_server** という名前のホストパラメーターを追加し、**string** タイプを選択して、値を Puppet サーバーのホスト名 (**puppet.example.com** など) に設定します。
- オプション: **puppet_ca_server** という名前のホストパラメーターを追加し、**string** タイプを選択して、値を Puppet CA サーバーのホスト名 (**puppet-ca.example.com** など) に設定します。**puppet_ca_server** が設定されていない場合、Puppet エージェントは **puppet_server** と同じサーバーを使用します。
- オプション: **puppet_environment** という名前のホストパラメーターを追加し、**string** タイプを選択して、ホストで使用する Puppet 環境に値を設定します。

6. 適切なアクティベーションキーを使用して、ホストが Satellite Server から Puppet エージェントパッケージにアクセスできることを確認します。

1.5. ホスト登録時の PUPPET AGENT のインストールと設定

登録時に、ホストに Puppet エージェントをインストールして設定できます。Satellite と Puppet を統合するには、設定済みの Puppet エージェントがホストに必要です。

前提条件

- Satellite で Puppet が有効になっている。詳細は、「Satellite と Puppet の統合の有効化」 を参照してください。
- ホストのオペレーティングシステムバージョン用の Red Hat Satellite Client 6 リポジトリーが、Satellite Server 上で同期され、使用するアクティベーションキーで有効になっている。詳細は、**コンテンツの管理** の コンテンツのインポート を参照してください。
- アクティベーションキーがある。詳細は、**コンテンツの管理** の アクティベーションキーの管理 を参照してください。

手順

1. Satellite Web UI で、**Configure** > **Global Parameters** に移動して、ホストパラメーターをグローバルに追加します。あるいは、**Configure** > **Host Groups** に移動し、ホストグループを編集または作成して、ホストパラメーターをホストグループにのみ追加することもできます。
2. グローバルパラメーターまたはホストグループのホストパラメーターを使用して、Puppet エージェントを有効化します。**enable-puppet7** という名前のホストパラメーターを追加し、**boolean** タイプを選択して、値を **true** に設定します。
3. グローバルパラメーターまたはホストグループで次のホストパラメーターを使用して、Puppet エージェントの設定を指定します。
   - **puppet_server** という名前のホストパラメーターを追加し、**string** タイプを選択して、値を Puppet サーバーのホスト名 (**puppet.example.com** など) に設定します。
   - オプション: **puppet_ca_server** という名前のホストパラメーターを追加し、**string** タイプを選択して、値を Puppet CA サーバーのホスト名 (**puppet-ca.example.com** など) に設定します。**puppet_ca_server** が設定されていない場合、Puppet エージェントは **puppet_server** と同じサーバーを使用します。
   - オプション: **puppet_environment** という名前のホストパラメーターを追加し、**string** タイプを選択して、ホストで使用する Puppet 環境に値を設定します。

BZ2177730 が解決されるまでは、Puppet サーバーが Capsule Server となっている統合セットアップでも、ホストパラメーターを使用して Puppet エージェント設定を指定する必要があります。

4. **Hosts** > **Register Host** に移動し、適切なアクティベーションキーを使用してホストを登録します。詳細は、**ホストの管理** の ホストの登録 を参照してください。
5. **Infrastructure** > **Capsules** に移動します。
6. 必要な Capsule Server の **Actions** コラムの一覧から、**Certificates** を選択します。
7. 必要なホストの右にある **Sign** をクリックして、Puppet エージェントの SSL 証明書に署名します。

1.6. PUPPET AGENT の手動でのインストールと設定

Puppet エージェントを手動でホストにインストールし、設定できます。Satellite と Puppet を統合するには、設定済みの Puppet エージェントがホストに必要です。

前提条件

- Satellite で Puppet が有効になっている。詳細は、「Satellite と Puppet の統合の有効化」 を参照してください。
- ホストに Puppet 環境が割り当てられている。
- ホストのオペレーティングシステムバージョン用の Red Hat Satellite Client 6 リポジトリーが、Satellite Server 上で同期され、ホストのコンテンツビューおよびライフサイクル環境で使用可能であり、ホストに対して有効になっている。詳細は、**コンテンツの管理** の Satellite でホストのリポジトリーセットのステータスを変更する を参照してください。

手順

1. **root** ユーザーで、ホストにログインします。
2. Puppet エージェントパッケージをインストールします。
   - Red Hat Enterprise Linux 8 以降を実行しているホストの場合:

     ```
     # dnf install puppet-agent
     ```

   - Red Hat Enterprise Linux 7 以前を実行しているホストの場合:

     ```
     # yum install puppet-agent
     ```

3. 次のスクリプトを使用して、Puppet エージェントを現在のシェルの **PATH** に追加します。

   ```
   . /etc/profile.d/puppet-agent.sh
   ```

4. Puppet エージェントを設定します。ホストの所属先の Puppet 環境名に **environment** パラメーターを設定します。

   ```
   # puppet config set server satellite.example.com --section agent
   # puppet config set environment My_Puppet_Environment --section agent
   ```

5. Puppet エージェントサービスを開始します。

```
# puppet resource service puppet ensure=running enable=true
```

6. ホストの証明書を作成します。

```
# puppet ssl bootstrap
```

7. Satellite Web UI で、**Infrastructure** > **Capsules** に移動します。
8. 必要な Capsule Server の **Actions** コラムの一覧から、**Certificates** を選択します。
9. 必要なホストの右にある **Sign** をクリックして、Puppet エージェントの SSL 証明書に署名します。
10. ホスト上で、Puppet エージェントを再度実行します。

```
# puppet ssl bootstrap
```

1.7. 設定管理の実施

ホストに Puppet エージェントをデプロイしたら、Puppet を使用して設定管理を開始できます。その概略的な手順は以下のとおりです。

1. Puppet モジュールを Puppet サーバーで管理する (インストールおよび更新)。
2. Puppet クラスおよび環境を Puppet モジュールから Satellite にインポートする。
3. オプション: Puppet クラスから設定グループを作成する。
4. さまざまなレベルでスマートクラスパラメーターのオーバーライドを設定する。
5. ホストグループまたは個別ホストに Puppet クラスまたは設定グループを割り当てる。
6. ホストで Puppet エージェントを実行する間隔および Puppet サーバーの設定を適用する間隔を設定する。
7. Satellite Web UI でレポートを使用して設定管理を監視する。詳細は、**Red Hat Satellite の管理** の リソースの監視 を参照してください。
8. メール通知を設定する。詳細は、**Red Hat Satellite の管理** の メール通知の設定 を参照してください。

Puppet クラスまたは設定グループを割り当てた後、Satellite は設定された間隔で設定管理を自動的に実行し、ホストに Puppet 設定を適用します。または、**Run Puppet Once**機能を使用して、オンデマンドで手動で設定適用を開始することもできます。詳細は、「SSH を使用した 1 回の Puppet 実行」 を参照してください。

1.8. SATELLITE と PUPPET の統合の無効化

Satellite で Puppet の使用を停止するには、以下の手順に従います。

コマンドで **--remove-all-data** 引数を指定しないと、Satellite データベース内のすべての Puppet 関連データが削除されることに注意してください。**--remove-all-data** 引数を指定すると、このコマンドは Puppet 環境を含む Puppet サーバーデータファイルをさらに削除します。

警告

--remove-all-data 引数を使用して Puppet を無効にすると、後で Puppet を再度有効にすることはできなくなります。これは既知の問題です。Bug 2087067 を参照してください。

前提条件

- Puppet が Satellite で有効化されている。

手順

1. いずれかの Capsule で Puppet サーバーを使用している場合は、すべての Capsule で Puppet サーバーを無効化します。

```
# satellite-maintain plugin purge-puppet --remove-all-data
```

2. Satellite Server で Puppet サーバーを無効化します。

```
# satellite-maintain plugin purge-puppet --remove-all-data
```

第2章 PUPPET モジュールの管理

2.1. SATELLITE SERVER への PUPPET モジュールのインストール

事前にビルドされた Puppet モジュールを Puppet Forge からインストールできます。Puppet Forge は、コミュニティーが提供する Puppet モジュールを提供するリポジトリーです。**サポート対象** のフラグが付いた Puppet モジュールは、Puppet Inc によって正式にサポートされ、テストされています。

この例では、ホストに **ntp モジュール** を追加する方法を示します。

手順

1. forge.puppet.com に移動し、**ntp** を検索します。最初のモジュールの 1 つは **puppetlabs/ntp** です。
2. SSH を使用して Satellite Server に接続し、Puppet モジュールをインストールします。

```
# puppet module install puppetlabs-ntp -i
/etc/puppetlabs/code/environments/production/modules
```

-i パラメーターを使用してパスおよび Puppet 環境を指定します (例:**production**)。

インストールが完了すると、出力は以下のようになります。

```
Notice: Preparing to install into /etc/puppetlabs/code/environments/production/modules ...
Notice: Created target directory /etc/puppetlabs/code/environments/production/modules
Notice: Downloading from https://forgeapi.puppet.com ...
Notice: Installing -- do not interrupt ...
/etc/puppetlabs/code/environments/production/modules
|-| puppetlabs-ntp (v8.3.0)
  |-- puppetlabs-stdlib (v4.25.1) [/etc/puppetlabs/code/environments/production/modules]
```

Puppet モジュールをインストールする別の方法は、Puppet モジュールを含むディレクトリーを上記のモジュールパスにコピーすることです。依存関係を手動で解決してください。

2.2. PUPPET モジュールの更新

puppet コマンドを使用して、既存の Puppet モジュールを更新できます。

手順

1. SSH を使用して Puppet サーバーに接続し、Puppet モジュールの場所を探します。

```
# puppet config print modulepath
```

これにより、以下のような出力が返されます。

```
/etc/puppetlabs/code/environments/production/modules:/etc/puppetlabs/code/environments/com
mon:/etc/puppetlabs/code/modules:/opt/puppetlabs/puppet/modules:/usr/share/puppet/modules
```

2. モジュールが上記のパスにある場合、以下のコマンドによりモジュールが更新されます。

```
# puppet module upgrade module name
```

第3章 PUPPET クラスおよび環境の SATELLITE へのインポート

ホストにクラスを割り当てる前に、インストール済みの Puppet モジュールから Satellite Server または アタッチされた Capsule Server に Puppet クラスおよび環境をインポートします。

前提条件

- **Any Organization** および **Any Location** をコンテキストとして選択してください。そうでないと、インポートが失敗する可能性があります。

手順

1. Satellite Web UI で、**Configure** > **Puppet ENC** > **Classes** または **Configure** > **Puppet ENC** > **Environments** に移動します。
2. 右上隅の **Import** をクリックし、モジュールのインポート元となる Capsule を選択します。通常、Satellite Server またはアタッチされている Capsule Server のいずれかを選択できます。
3. 左側のチェックボックスを使用して、インポートする Puppet 環境を選択します。
4. **Update** をクリックして、Puppet 環境とクラスを Satellite にインポートします。
5. インポートにより、以下のように通知が表示されます。

> Successfully updated environments and Puppet classes from the on-disk Puppet installation

第4章 カスタム PUPPET 環境の作成

Satellite 内に Puppet 環境を作成できます。

手順

1. Satellite Web UI で、**Configure > Puppet Environments** に移動します。
2. **Create Puppet Environment** をクリックして Puppet 環境を作成します。
3. **Name** を入力します。**example_environment** のように、英数字とアンダースコアを使用できます。
4. オプション: ロケーションのコンテキストを設定します。
5. オプション: 組織のコンテキストを設定します。
6. **Submit** をクリックして、Puppet 環境を作成します。

Puppet モジュールの Satellite へのインポートを実行する前に、環境が Puppet サーバーの **/etc/puppetlabs/code/environments/example_environment** フォルダーとしてすでに存在し、インストール済みの Puppet モジュールが含まれている必要があります。

第5章 PUPPET 設定グループの作成

Puppet 設定グループは、Puppet クラスの名前付きリストです。このグループを使用すると、Puppet クラスの機能を組み合わせて、簡単にホストに割り当てることができます。これは、純粋な Puppet のプロファイルの概念と同じです。

手順

1. Satellite Web UI で、**Configure** > **Puppet ENC** > **Config Groups** に移動します。
2. **Create Config Group** をクリックします。
3. 設定グループに追加するクラスを選択します。
    a. Puppet 設定グループに意味のある **名前** を選択します。
    b. 選択した Puppet クラスを **Included Classes** フィールドに追加します。
4. **Submit** をクリックして変更を保存します。

第6章 PUPPET スマートクラスパラメーターの設定

6.1. PUPPET パラメーターの階層

Puppet パラメーターは階層的に構造化されています。下位レベルのパラメーターは、上位レベルのパラメーターをオーバーライドします。

1. グローバルパラメーター
2. 組織パラメーター
3. ロケーションパラメーター
4. ホストグループパラメーター
5. ホストパラメーター

たとえば、ホスト固有のパラメーターはすべての上位レベルのパラメーターをオーバーライドし、ロケーションパラメーターは組織またはグローバルレベルのパラメーターのみをオーバーライドします。この機能は、ロケーションまたは組織を使用してホストをグループ化する場合に特に便利です。

6.2. スマートクラスパラメーターのグローバルなオーバーライド

Puppet クラスは、Satellite Server にインポートした後で設定できます。この例では、ntp サーバーのデフォルトリストをオーバーライドします。

手順

1. Satellite Web UI で、**Configure** > **Puppet ENC** > **Classes** に移動します。
2. 設定を変更する**ntp** Puppet クラスを選択します。
3. **Smart Class Parameter** タブを選択し、**servers** を検索します。
4. **Override** チェックボックスが選択されていることを確認します。
5. **Parameter Type** ドロップダウンメニューを **array** に設定します。
6. **Default Value** として ntp サーバーのリストを挿入します。

   ```
   ["0.de.pool.ntp.org","1.de.pool.ntp.org","2.de.pool.ntp.org","3.de.pool.ntp.org"]
   ```

   配列を記述する別の方法は、**yaml** 構文です。

   ```
   - 0.de.pool.ntp.org
   - 1.de.pool.ntp.org
   - 2.de.pool.ntp.org
   - 3.de.pool.ntp.org
   ```

7. **Submit** をクリックして、Puppet モジュール **ntp** のデフォルト設定を変更します。

6.3. 組織のスマートクラスパラメーターのオーバーライド

ホストのグループを使用して、一度に複数のホストの Puppet パラメーターをオーバーライドできます。以下の例では、**組織** コンテキストを選択して、コンテキストベースのパラメーターの設定を説明します。

組織 レベルの Puppet パラメーターは、**ロケーション**レベルの Puppet パラメーターでオーバーライドされることに注意してください。

手順

1. Satellite Web UI で、**Configure** > **Puppet ENC** > **Classes** に移動します。
2. クラス名をクリックしてクラスを選択します。
3. **Smart Class Parameter** タブで、パラメーターを選択します。
4. **Order** リストを使用して Puppet パラメーターの階層を定義します。個々のホスト (**fqdn**) は最も関連性が強いとマークし、組織コンテキスト (**organization**) は最も関連性が弱いとマークします。
5. 最初のマッチが検索された後、それ以降のマッチするパラメーターをすべて追加する場合は、**Merge Overrides** をチェックします。
6. より具体的な値が定義されている場合でもデフォルト値も含める場合は、**Merge Default** をオンにします。
7. 選択したパラメーターの一意の値のリストを作成する場合は、**Avoid Duplicates** をオンにします。
8. **matcher** フィールドには、**order** リストからの **attribute タイプ** が必要です。
9. オプション: マッチャーをさらに追加するには、**Add Matcher** をクリックします。
10. **Submit** をクリックして変更を保存します。

6.4. ロケーションのスマートクラスパラメーターのオーバーライド

ホストのグループを使用して、一度に複数のホストの Puppet パラメーターをオーバーライドできます。以下の例では、**ロケーション** コンテキストを選択して、コンテキストベースのパラメーターの設定を説明します。

手順

1. Satellite Web UI で、**Configure** > **Puppet ENC** > **Classes** に移動します。
2. クラス名をクリックしてクラスを選択します。
3. **Smart Class Parameter** タブで、パラメーターを選択します。
4. **Order** リストを使用して Puppet パラメーターの階層を定義します。個々のホスト (**fqdn**) は最も関連性が強いとマークし、ロケーションコンテキスト (**location**) は最も関連性が弱いとマークします。
5. 最初のマッチが検索された後、それ以降のマッチするパラメーターをすべて追加する場合は、**Merge Overrides** をチェックします。

6. より具体的な値が定義されている場合でもデフォルト値も含める場合は、**Merge Default** をオンにします。
7. 選択したパラメーターの一意の値のリストを作成する場合は、**Avoid Duplicates** をオンにします。
8. **matcher** フィールドには、**order** リストからの **attribute タイプ** が必要です。たとえば、**場所** のコンテキストとして **Paris** を選択し、値をフランスの ntp サーバーに設定できます。
9. オプション: マッチャーをさらに追加するには、**Add Matcher** をクリックします。
10. **Submit** をクリックして変更を保存します。

6.5. 各ホストのスマートクラスパラメーターのオーバーライド

各ホストのパラメーターをオーバーライドできます。これは、複数のホストがあり、1 つのホストのみに変更を加えたい場合に推奨されます。

手順

1. Satellite Web UI で、**Hosts > All Hosts** に移動します。
2. ホスト名をクリックしてホストを選択します。
3. **Edit** をクリックします。
4. **Host** タブで、**Puppet 環境** を選択します。
5. **Puppet ENC** タブを選択します。
6. **Override** をクリックして、Puppet パラメーターを編集します。
7. **Submit** をクリックして変更を保存します。

第7章 ホストグループへの PUPPET クラスの割り当て

ホストグループを使用して、**ntp** Puppet クラスを一度に複数のホストに割り当てます。このホストグループに基づいてデプロイするすべてのホストには、この Puppet クラスがインストールされます。

手順

1. Satellite Web UI で **Configure** > **Host Groups** に移動して、ホストグループを作成するか、既存のホストグループを編集します。
2. **Host Group** タブで、以下のパラメーターを設定します。
    a. **Lifecycle Environment** は、コンテンツの特定のバージョンがホストで利用可能になるステージを表します。
    b. **Content View** は、製品で構成されており、コンテンツリポジトリーのバージョン管理を可能にします。
    c. **Environment** では、ホストのグループに独自の専用設定を提供できます。
3. **Puppet ENC** タブに移動します。
4. Puppet クラスを **Included Classes** に追加するか、Puppet 設定グループが設定されている場合は **Included Config Groups** に追加します。
5. **Submit** をクリックして変更を保存します。

第8章 個々のホストへの PUPPET クラスの割り当て

手順

1. Satellite Web UI で、**Hosts** > **All Hosts** に移動します。
2. **ntp** Puppet クラスを追加するホストを見つけて、**Edit** をクリックします。
3. **Puppet ENC** タブを選択し、ntp クラスを探します。
4. **ntp** の横にある + 記号をクリックして、**included classes** の一覧に **ntp submodule** を追加します。
5. **Submit** をクリックして変更を保存します。

   ヒント

   個々のホストの **Puppet classes** タブが空の場合は、適切な Puppet 環境に割り当てられているかどうかを確認します。

6. Puppet 設定を確認します。
   a. **Hosts** > **All Hosts** に移動し、ホストを選択します。
   b. 上部のオーバーフローメニューから **Legacy UI** を選択します。
   c. **Details** で、**Puppet YAML** をクリックします。これにより、以下のような出力が生成されます。

   ```
   ---
   parameters:
     // shortened YAML output
   classes:
     ntp:
       servers:
   '["0.de.pool.ntp.org","1.de.pool.ntp.org","2.de.pool.ntp.org","3.de.pool.ntp.org"]'
   environment: production
   ...
   ```

7. ntp 設定を確認します。
   SSH を使用してホストに接続し、**/etc/ntp.conf** の内容を確認します。

   この例では、ホストが **CentOS 7** を実行していることを前提としています。他のオペレーティングシステムでは、ntp 設定ファイルが別のパスに保存されている場合があります。

   ヒント

   以下のコマンドを実行して、ホストで Puppet エージェントを実行しなければならない場合があります。

   ```
   # puppet agent -t
   ```

8. ホストで以下のコマンドを実行し、クロックの同期に使用される ntp サーバーを確認します。

```
# cat /etc/ntp.conf
```

これにより、以下のような出力が返されます。

```
# ntp.conf: Managed by puppet.
server 0.de.pool.ntp.org
server 1.de.pool.ntp.org
server 2.de.pool.ntp.org
server 3.de.pool.ntp.org
```

これで、ntp モジュールが機能し、ホストまたはホストのグループに追加して、ntp 設定を自動的にロールアウトできます。

第9章 ホストでの PUPPET 設定の適用

オンデマンドで手動で (run once)、または設定可能な間隔で自動的に、Satellite からの設定を適用することができます。

9.1. SSH を使用した 1 回の PUPPET 実行

ホストで Puppet を実行するには、適切なジョブテンプレートを **Run Puppet Once** 機能に割り当てます。

手順

1. Satellite Web UI で、**Administer** > **Remote Execution Features** に移動します。
2. **puppet_run_host** リモート実行機能を選択します。
3. **Run Puppet Once – SSH Default** ジョブテンプレートを割り当てます。

ジョブを実行し、カテゴリー **Puppet** およびテンプレート **Run Puppet Once - SSH Default** を選択して、ホストで Puppet を実行します。または、ホストの詳細ページの **Schedule Remote Job** ドロップダウンメニューで **Run Puppet Once** をクリックします。

9.2. 自動適用の間隔について

最後の Puppet レポートが、分単位で設定された **outofsync_interval** と **puppet_interval** を組み合わせた値より古い場合に、Satellite はホストが同期されていないと見なします。デフォルトでは、ホストの Puppet エージェントは 30 分ごとに実行されます。**puppet_interval** は 35 分に設定され、グローバルの **outofsync_interval** は 30 分に設定されています。

ホストが同期されていないと見なされるまでの実質的な時間は、**outofsync_interval** と **puppet_interval** の合計です。たとえば、グローバルの **outofsync_interval** を 30 に設定し、**puppet_interval** を 60 に設定すると、合計の 90 分が経過するとホストのステータスが **out of sync** に変わります。

9.3. ホストでの PUPPET エージェント実行間隔の設定

Puppet エージェントを実行してレポートを Satellite に送信する間隔を設定します。

手順

1. SSH を使用してホストに接続します。
2. Puppet エージェントの実行間隔を **/etc/puppetlabs/puppet/puppet.conf** に追加します (例:**runinterval = 1h**)。

9.4. グローバルの非同期間隔の設定

手順

1. Satellite Web UI で、**Administer** > **Settings** に移動します。
2. **General** タブで、**Out of sync interval** を編集します。ホストが同期されていないと見なされるまでの期間を分単位で設定します。

outofsync_interval パラメーターを追加して、ホストグループ または 各ホスト でこの間隔をオーバーライドすることもできます。

9.5. PUPPET の非同期間隔の設定

手順

1. Satellite Web UI で、**Administer > Settings** に移動して、**Config Management** タブをクリックします。
2. **Puppet interval** フィールドで、Puppet を使用して報告するホストが同期されていないと見なされるまでの期間に分単位の値を設定します。

9.6. ホストグループの非同期間隔のオーバーライド

手順

1. Satellite Web UI で、**Configure > Host Groups** に移動します。
2. ホストグループを選択します。
3. **Parameters** タブで **Add Parameter** をクリックします。
4. **Name** フィールドに **outofsync_interval** を入力します。
5. **Type** ドロップダウンメニューから、**integer** を選択します。
6. **Value** フィールドに新しい間隔を分単位で入力します。
7. **Submit** をクリックします。

9.7. 各ホストの非同期間隔のオーバーライド

手順

1. Satellite Web UI で、**Hosts > All Hosts** に移動します。
2. 選択したホストの **Edit** をクリックします。
3. **Parameters** タブで **Add Parameter** をクリックします。
4. **Name** フィールドに **outofsync_interval** を入力します。
5. **Type** ドロップダウンメニューから、**integer** を選択します。
6. **Value** フィールドに新しい間隔を分単位で入力します。
7. **Submit** をクリックします。