Red Hat Satellite 6.6

コンテンツ管理ガイド

Red Hat およびカスタムソースのコンテンツ管理ガイド

Last Updated: 2020-04-28

Red Hat Satellite 6.6 コンテンツ管理ガイド

Red Hat およびカスタムソースのコンテンツ管理ガイド

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

法律上の通知

概要

本書を使用して、RPM ファイル、ISO イメージ、Puppet モジュールなどの、Satellite 6 のコンテンツを理解し、管理します。Red Hat Satellite 6 は、アプリケーションライフサイクル全体でプロモートされた一連のコンテンツビューを使用してこのコンテンツを管理します。本書では、それぞれの組織に合わせたアプリケーションライフサイクルとライフサイクル環境内でホストの状態に合致するコンテンツビューの作成方法を説明します。これらのコンテンツビューは、最終的に Red Hat Satellite 6 環境でホストをプロビジョニングおよび更新する基礎となります。

目次

第1章 はじめに

Satellite 6 では、**コンテンツ** は、システムにインストールされたソフトウェアとして定義されます。コンテンツには、ベースオペレーティングシステム、ミドルウェアサービス、エンドユーザーアプリケーションなどが含まれますが、これらに限定はされません。Red Hat Satellite 6 を使用して、ソフトウェアライフサイクルの各段階で Red Hat Enterprise Linux システムのさまざまな種類のコンテンツを管理できます。

Red Hat Satellite 6 では、以下のコンテンツを管理します。

サブスクリプション管理

サブスクリプション管理は、Red Hat サブスクリプション情報を管理する方法を組織に提供します。

コンテンツ管理

コンテンツ管理は、組織が Red Hat コンテンツを保存してさまざまな方法で整理する手段を提供します。

1.1. コンテンツ管理タイプの概要

Red Hat Satellite 6 を使用して、次の Red Hat コンテンツタイプを管理できます。

RPM パッケージ

RPM ファイルは、Red Hat サブスクリプションに関連するリポジトリーからインポートします。Satellite Server を使用して、Red Hat のコンテンツ配信ネットワークから RPM ファイルをダウンロードし、ローカルに保存します。これらのリポジトリーと RPM ファイルはコンテンツビューで使用できます。

キックスタートツリー

システムの作成には、キックスタートツリーをインポートします。新しいシステムは、ネットワーク経由でこれらのキックスタートツリーにアクセスしてインストールのベースコンテンツとして使用します。また、Red Hat Satellite 6 には、事前定義済みのキックスタートテンプレートが複数含まれ、独自のキックスタートテンプレートを作成することもでき、これらのテンプレートを使用してシステムのプロビジョニングやインストールをカスタマイズできます。

Red Hat Satellite 6 では、カスタムコンテンツを管理することもできます。以下に例を示します。

ISO および KVM イメージ

インストールおよびプロビジョニング向けのメディアをダウンロードおよび管理します。たとえば、Satellite は、特定の Red Hat Enterprise Linux および Red Hat 以外のオペレーティングシステム向けの ISO イメージおよびゲストイメージをダウンロード、保存、および管理します。

Puppet モジュール

RPM コンテンツとともに Puppet モジュールをアップロードできるため、Puppet は、プロビジョニング後のシステムの状態を設定できます。また、ユーザーはプロビジョニングプロセスの一部として Puppet クラスとパラメーターを管理することもできます。

OSTree

OSTree ブランチをインポートし、このコンテンツを HTTP の場所に公開できます。

この手順を使用して、SSL 証明書や OVAL ファイルなど、必要なタイプのコンテンツのカスタムコンテンツを追加できます。

第2章 組織の管理

組織は、所有者、目的、コンテンツ、セキュリティーレベルなどに基づいて Red Hat Satellite 6 リソースを論理グループに分割します。Red Hat Satellite 6 では複数の組織を作成および管理し、Red Hat サブスクリプションを分割して、各個別組織に割り当てることができます。これにより、1 つの管理システムで複数の個別組織のコンテンツを管理できるようになります。以下に、組織管理の例をいくつか示します。

1 つの組織

システム管理チェーンが単純な小企業。この場合は、このビジネスに対して組織を 1 つ作成し、コンテンツをその組織に割り当てることができます。

複数の組織

複数の小規模な事業単位を所有する大企業 (たとえば、独立したシステム管理およびソフトウェア開発グループがある会社)。この場合は、企業に対して、またこの企業が所有する各事業単位に対して組織を作成できます。組織を作成することで、それぞれのシステムインフラストラクチャーを分離できます。各組織に、それぞれのニーズに基づいてコンテンツを割り当てることができます。

外部組織

他の組織の外部システムを管理する会社 (たとえば、クラウドコンピューティングと Web ホスティングリソースを顧客に提供する会社)。この場合は、会社の独自のシステムインフラストラクチャーの組織に加え、外部の各会社に対して組織を作成できます。必要に応じて、各組織にコンテンツを割り当てることができます。

Red Hat Satellite 6 のデフォルトのインストールには、**Default_Organization** と呼ばれるデフォルトの組織が含まれます。

新しいユーザー

新しいユーザーに、デフォルトの組織が割り当てられていない場合には、このユーザーのアクセス権限は制限されます。ユーザーにシステムの権限を付与するには、ユーザーをデフォルトの組織に割り当てます。ユーザーが次に Satellite にログオンしたときに、ユーザーのアカウントに適切なシステム権限が付与されます。

2.1. 組織の作成

以下の手順を使用して組織を作成します。

手順

組織を作成するには、以下の手順を行います。

1. Satellite Web UI で、**管理** > **組織** に移動します。
2. **新規組織** をクリックします。
3. **名前** フィールドに、組織の名前を入力します。
4. **ラベル** フィールドに、組織の一意な ID を入力します。これは、コンテンツストレージ用ディレクトリーなどの特定のアセットを作成およびマッピングする場合に使用されます。文字、数字、アンダースコア、およびダッシュを使用し、スペースは使用しないでください。
5. オプション: **説明** フィールドに、組織の説明を入力します。
6. **送信** をクリックします。

7. ホストに組織が割り当てられていない場合は、組織に追加するホストを選択し、**Proceed to Edit (編集に進む)** をクリックします。

8. **編集** ページで、組織に追加するインフラストラクチャーリソースを割り当てます。このインフラストラクチャーリソースには、ネットワークリソース、インストールメディア、キックスタートテンプレートなどのパラメーターが含まれます。このページには、**管理** > **組織** に移動し、編集する組織を選択するといつでも戻ることができます。

9. **送信** をクリックします。

CLI をご利用の場合

1. 組織を作成するには、以下のコマンドを入力します。

```
# hammer organization create \
--name "your_organization_name" \
--label "your_organization_label \
--description "your_organization_description"
```

2. オプション: 組織を編集するには、**hammer organization update** コマンドを入力します。たとえば、以下のコマンドはコンピュートリソースを組織に割り当てます。

```
# hammer organization update \
--name "your_organization_name" \
--compute-resource-ids 1
```

2.2. 組織コンテキストの設定

ホストに使用する組織や関連するリソースを定義する組織のコンテキスト

手順

組織メニューは、Satellite Web UI の左上にあるメニューバーの最初のメニュー項目です。現在の組織を選択していない場合には、メニューには**任意の組織**と表示されます。**任意の組織**ボタンをクリックして、使用する組織を選択します。

CLI をご利用の場合

CLI を使用する場合は、オプションとして **--organization "your_organization_name"** または **--organization-label "your_organization_label"** を追加できます。以下に例を示します。

```
# hammer subscription list --organization "Default_Organization"
```

このコマンドは、Default_Organization に割り当てられたサブスクリプションを出力します。

2.3. 組織のデバッグ証明書の作成

組織のデバッグ証明書が必要な場合は、以下の手順を使用します。

手順

組織のデバッグ証明書を作成するには、以下の手順を実行します。

1. Satellite Web UI で、**管理** > **組織** に移動します。

2. デバッグ証明書を生成する組織を選択します。
3. **生成してダウンロード** をクリックします。
4. 証明書ファイルを安全な場所に保存します。

プロビジョニングテンプレートのデバッグ証明書

デバッグ証明書が組織内にダウンロードされていない場合は、プロビジョニングテンプレートのダウンロード時に自動的に生成されます。

2.4. 組織のデバッグ証明書を使用したリポジトリーコンテンツの参照

組織のデバッグ証明書がある場合には、Web ブラウザーまたは API を使用して、組織のリポジトリーコンテンツを表示できます。

前提条件

1. 「組織のデバッグ証明書の作成」 に記載されている手順で、組織の証明書を作成およびダウンロードする。
2. たとえば、デフォルトの組織の X.509 証明書を開く。

```
$ vi 'Default Organization-key-cert.pem'
```

3. このファイルの **-----BEGIN RSA PRIVATE KEY-----** から **-----END RSA PRIVATE KEY-----** までを **key.pem** ファイルにコピーする。
4. このファイルの **-----BEGIN CERTIFICATE-----** から **-----END CERTIFICATE-----** までを **cert.pem** ファイルにコピーする。

手順

ブラウザーを使用するには、X.509 証明書をご利用のブラウザーがサポートする形式にまず変換してから、証明書をインポートする必要があります。

Firefox をご利用の場合

Firefox で組織のデバッグ証明書を使用するには、以下の手順を行います。

1. PKCS12 形式の証明書を作成するには、以下のコマンドを入力します。

```
$ openssl pkcs12 -keypbe PBE-SHA1-3DES -certpbe PBE-SHA1-3DES -export -in cert.pem
-inkey key.pem -out organization_label.pfx -name organization_name
```

2. Firefox ブラウザーで、**編集** > **設定** > **詳細タブ** に移動します。
3. **証明書の表示** を選択し、**証明書** タブをクリックします。
4. **インポート** をクリックし、読み込む **.pfx** ファイルを選択します。
5. アドレスバーに、リポジトリーの参照先の URL を以下の形式で入力します。

```
http://satellite.example.com/pulp/repos/organization_label
```

Pulp は組織ラベルを使用するため、組織ラベルを URL に入力する必要があります。

CURL をご利用の場合

CURL で組織のデバッグ証明書を使用するには、以下のコマンドを入力します。

```
$ curl -k --cert cert.pem --key key.pem \
http://satellite.example.com/pulp/repos/Default_Organization/Library/content/dist/rhel/server/7/7Server/x86_64/sat-tools/6.6/os/
```

cert.pem と **key.pem** へのパスが適切な絶対パスであることを確認します。間違っていると、エラーメッセージなしにコマンドが失敗します。

2.5. 組織の削除

組織は、ライフサイクル環境またはホストグループに関連付けられていない場合に削除できます。削除する組織にライフサイクル環境またはホストグループが関連付けられている場合は、**管理** > **組織**> に移動して関連の組織をクリックします。インストール中に作成されたデフォルトの組織は、Satellite 環境で関連付けられていないホストへのプレースホルダーであるため、削除しないでください。環境には常に 1 つ以上の組織が必要です。

手順

組織を削除するには、以下の手順を行います。

1. Satellite Web UI で、**管理** > **組織** に移動します。
2. 削除する組織名の右側にあるリストから **削除** を選択します。
3. **OK** をクリックして、組織を削除します。

CLI をご利用の場合

1. 以下のコマンドを入力して、削除する組織の ID を取得します。

   ```
   # hammer organization list
   ```

   出力から、削除する組織の ID をメモします。

2. 以下のコマンドを入力して組織を削除します。

   ```
   # hammer organization delete --id Organization_ID
   ```

第3章 ロケーションの管理

ロケーションは、リソースを分類し、ホストを割り当てる方法を提供する点で組織に似ています。組織とロケーションは、以下の点が異なります。

- ロケーションは、物理または地理的設定をベースにしています。
- ロケーションの構造は階層的です。

3.1. ロケーションの作成

以下の手順を使用して、ホストとリソースをロケーション別に管理できるようにロケーションを作成します。

手順

ロケーションを作成するには、以下の手順を行います。

1. Satellite Web UI で、**管理** > **ロケーション** に移動します。
2. **新規組ロケーション**をクリックします。
3. オプション: **親** リストから、親ロケーションを選択します。これにより、ロケーションの階層が作成されます。
4. **名前** フィールドに、ロケーションの名前を入力します。
5. オプション: **説明** フィールドに、ロケーションの説明を入力します。
6. **送信** をクリックします。
7. ホストにロケーションが割り当てられていない場合は、新しいロケーションに割り当てるホストを追加し、**Proceed to Edit (編集に進む)** をクリックします。
8. ロケーションに追加するインフラストラクチャーリソースを割り当てます。このインフラストラクチャーリソースには、ネットワークリソース、インストールメディア、キックスタートテンプレートなどのパラメーターが含まれます。**管理** > **ロケーション** に移動して、編集するロケーションを選択すると、いつでもこのページに戻ることができます。
9. **送信** をクリックして変更を保存します。

CLI をご利用の場合

以下のコマンドを実行してロケーションを作成します。

```
# hammer location create \
--parent-id "parent_location_id" \
--name "your_location_name" \
--description "your_location_description"
```

3.2. 複数ロケーションの作成

以下の Bash スクリプトでは、3 つのロケーション (ロンドン、ミュンヘン、ボストン) を作成して、これらの場所を Example Organization に割り当てます。

```
ORG="Example Organization"
LOCATIONS="London Munich Boston"

for LOC in ${LOCATIONS}
do
  hammer location create --name "${LOC}"
  hammer location add-organization --name "${LOC}" --organization "${ORG}"
done
```

3.3. ロケーションコンテキストの設定

ホストに使用するロケーションや関連するリソースを定義するロケーションのコンテキスト

手順

ロケーションメニューは、Satellite Web UI の左上にあるメニューバーの 2 番目のメニュー項目です。現在のロケーションを選択していない場合には、メニューには **任意のロケーション** と表示されます。**任意のロケーション** をクリックして、使用するロケーションを選択します。

CLI をご利用の場合

CLI を使用する場合は、オプションとして、**--location "your_location_name"** または **--location-id "your_location_id"** を追加します。以下に例を示します。

```
# hammer subscription list --location "Default_Location"
```

このコマンドは、**Default_Location** に割り当てられたサブスクリプションを出力します。

3.4. ロケーションの削除

ロケーションは、ライフサイクル環境またはホストグループに関連付けられていない場合に削除できます。削除するロケーションにライフサイクル環境またはホストグループが関連付けられている場合は、**管理** > **ロケーション**> に移動して関連のロケーションをクリックします。インストール中に作成されたデフォルトのロケーションは、Satellite 環境で関連付けられていないホストへのプレースホルダーであるため、削除しないでください。環境には常に 1 つ以上のロケーションが必要です。

手順

ロケーションを削除するには、以下の手順を行います。

1. Satellite Web UI で、**管理** > **ロケーション** に移動します。
2. 削除するロケーションの名前の右側にあるリストから**削除**を選択します。
3. **OK** をクリックして、ロケーションを削除します。

CLI をご利用の場合

1. 以下のコマンドを入力して、削除するロケーションの ID を取得します。

   ```
   # hammer location list
   ```

   出力から、削除するロケーションの ID をメモします。

2. 以下のコマンドを入力して、ロケーションを削除します。

```
# hammer location delete --id Location ID
```

第4章 サブスクリプションの管理

Red Hat Satellite 6 では、Red Hat のコンテンツ配信ネットワーク (CDN) からコンテンツをインポートします。Satellite 6 では、対応のリポジトリーからコンテンツの検索、アクセス、ダウンロードを行うサブスクリプションマニフェストが必要です。サブスクリプションマニフェストには、Satellite Server の組織ごとにサブスクリプション割り当てを含める必要があります。サブスクリプション情報はすべて、Red Hat カスタマーポータルのアカウントで確認できます。

本章のタスクを完了するには事前にカスタマーポータルでサブスクリプションマニフェストを作成する必要があります。

カスタマーポータルでサブスクリプションマニフェストを作成、管理、エクスポートするには、『**Red Hat Subscription Management の使用**』ガイドの「マニフェストの使用」を参照してください。

本章を参照して、Satellite Web UI 内でサブスクリプションマニフェストをインポートし、管理します。

サブスクリプションの割り当てと組織

複数のサブスクリプション割り当てがある場合は、複数の組織を管理できます。Satellite 6 では、Satellite Server で設定した組織ごとに 1 つの割り当てが必要です。この利点は、各組織が独立したサブスクリプションを保持するため、複数の組織をそれぞれ独自の Red Hat アカウントでサポートできることです。

未来の日付のサブスクリプション

サブスクリプション割り当てでは、未来の日付のサブスクリプションを使用できます。既存のサブスクリプションの有効期限前に、未来の日付のサブスクリプションをコンテンツホストに追加する場合は、リポジトリーへのアクセスが中断されず、そのまま利用できます。

現在のサブスクリプションの有効期限前に、コンテンツホストに未来の日付のサブスクリプションを手動でアタッチします。自動アタッチ機能は別の目的で設計されており、機能しない可能性があるので、この機能に依存しないでください。詳細は、「コンテンツホストへのサブスクリプションのアタッチ」を参照してください。

4.1. SATELLITE SERVER へのサブスクリプションマニフェストのインポート

以下の手順を使用して、サブスクリプションマニフェストを Satellite Server にインポートします。

前提条件

- サブスクリプションマニフェストファイルがカスタマーポータルからエクスポートされていること。詳細は、『**Red Hat Subscription Management の使用**』ガイドの「マニフェストの使用」を参照してください。

手順

サブスクリプションマニフェストをインポートするには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **コンテンツ** > **サブスクリプション** に移動して、**マニフェストの管理** をクリックします。
3. マニフェストの管理ウィンドウで、**参照**をクリックします。

4. サブスクリプションマニフェストファイルが保存されている場所に移動して、**表示** をクリックします。マニフェストの管理ウィンドウが自動的に終了しない場合は、**終了** をクリックしてサブスクリプションウィンドウに戻ります。

CLI をご利用の場合

1. サブスクリプションマニフェストファイルをクライアントから Satellite Server にコピーします。

```
$ scp ~/manifest_file.zip root@satellite.example.com:~/.
```

2. Satellite Server に root ユーザーとして接続し、サブスクリプションマニフェストファイルをインポートします。

```
# hammer subscription upload \
--file ~/manifest_file.zip \
--organization "organization_name"
```

リポジトリーを有効化して、Red Hat コンテンツをインポートできるようになります。詳細は、*5章Red Hat コンテンツのインポート* を参照してください。

4.2. SATELLITE WEB UI でのサブスクリプションの検索

サブスクリプションマニフェストを Satellite Server にインポートすると、マニフェストからのサブスクリプションがサブスクリプションウィンドウに表示されます。サブスクリプションが大量にある場合には、結果をフィルタリングして、特定のサブスクリプションを検索できます。

前提条件

Satellite Server にサブスクリプションマニフェストファイルをインポートしておくこと。詳細は、「Satellite Server へのサブスクリプションマニフェストのインポート」 を参照してください。

手順

サブスクリプションを検索するには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **コンテンツ** > **サブスクリプション** に移動します。
3. サブスクリプションウィンドウで、**検索** フィールドをクリックし、検索条件の一覧を表示して検索クエリーをビルドします。
4. 検索条件を選択して、他のオプションを表示します。
5. 検索クエリーをビルドしたら、検索アイコンをクリックします。

たとえば、**検索** フィールドにカーソルを置き、**expires (期限切れ)** を選択して、スペースキーを押すと、別のリストが表示され、>、< または = 文字を選択できます。> を選択してスペースキーを押すと、自動オプションの別のリストが表示されます。独自の条件を入力することも可能です。

4.3. SATELLITE WEB UI でのサブスクリプション割り当てへのサブスクリプションの追加

以下の手順を使用して、Satellite Web UI でサブスクリプション割り当てにサブスクリプションを追加します。

前提条件

Satellite Server にサブスクリプションマニフェストファイルをインポートしておくこと。詳細は、「Satellite Server へのサブスクリプションマニフェストのインポート」 を参照してください。

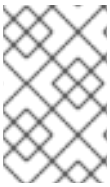
手順

サブスクリプション割り当てにサブスクリプションを追加するには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **コンテンツ** > **サブスクリプション** に移動します。
3. サブスクリプションウィンドウで **サブスクリプションの追加** をクリックします。
4. 追加するサブスクリプションの各行で、**Quantity to Allocate (割り当てる数量)** 列に数量を入力します。
5. **送信** をクリックします。

4.4. SATELLITE WEB UI でのサブスクリプション割り当てからのサブスクリプションの削除

以下の手順を使用して、Satellite Web UI でサブスクリプション割り当てからサブスクリプションを削除します。



注記

マニフェストは削除しないでください。Red Hat カスタマーポータルまたは Satellite Web UI でマニフェストを削除すると、コンテンツホストのエンタイトルメントがすべて解除されます。

前提条件

Satellite Server にサブスクリプションマニフェストファイルをインポートしておくこと。詳細は、「Satellite Server へのサブスクリプションマニフェストのインポート」 を参照してください。

手順

サブスクリプションを削除するには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **コンテンツ** > **サブスクリプション** に移動します。
3. 削除するサブスクリプションの各行で、該当のチェックボックスを選択します。
4. **削除**をクリックして、削除を確定します。

4.5. サブスクリプションマニフェストの更新およびリフレッシュ

サブスクリプションの割り当てを変更した場合は、マニフェストをリフレッシュして変更を反映させる必要があります。たとえば、以下のいずれかを行った場合はマニフェストをリフレッシュしてください。

- サブスクリプションの更新
- サブスクリプション数量の調整
- 追加のサブスクリプションの購入

Satellite Web UI で直接マニフェストをリフレッシュすることも、変更内容が含まれる更新済みのマニフェストをインポートすることもできます。

手順

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **コンテンツ** > **サブスクリプション** に移動します。
3. サブスクリプションウィンドウで、**マニフェストの管理** をクリックします。
4. マニフェストの管理ウィンドウで、**リフレッシュ** をクリックします。

4.6. コンテンツホストへのサブスクリプションのアタッチ

プロビジョニングプロセスで、コンテンツホストにサブスクリプションをアタッチするには通常、アクティベーションキーを使用します。ただし、アクティベーションキーは既存のホストを更新できません。新規または追加のサブスクリプション (たとえば未来の日付が指定されたサブスクリプション) を 1 台のホストにアタッチする必要がある場合は、以下の手順を使用します。

複数ホストの更新に関する詳細情報は、「コンテンツホストのサブスクリプションの一括更新」 を参照してください。

アクティベーションキーに関する詳細情報は、10章 *アクティベーションキーの管理* を参照してください。

Smart Management サブスクリプション

Satellite 6 では、管理するすべての Red Hat Enterprise Linux ホストの Red Hat Enterprise Linux Smart Management サブスクリプションを保持する必要があります。

ただし、コンテンツホストごとに Smart Management サブスクリプションをアタッチする必要はありません。Smart Management サブスクリプションは、製品の証明書に関連付けられていないため、自動的に Satellite のコンテンツホストにアタッチできません。コンテンツホストに Smart Management サブスクリプションを追加しても、コンテンツやリポジトリーへのアクセスができるようになりません。ご希望であれば、独自の記録用または追跡の目的で Smart Management サブスクリプションをマニフェストに追加することができます。

前提条件

Satellite Server にサブスクリプションマニフェストファイルをインポートしておくこと。

手順

コンテンツホストにサブスクリプションをアタッチするには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。

2. **ホスト** > **コンテンツホスト**に移動します。
3. 変更するサブスクリプションのコンテンツホストの行ごとに、該当するチェックボックスを選択してください。
4. **アクションの選択**一覧から**サブスクリプションの管理**を選択します。
5. 任意で、**検索**フィールドにキーと値を入力し、表示するサブスクリプションを絞り込みます。
6. 追加または削除するサブスクリプションの左側にあるチェックボックスを選択し、必要に応じて **選択した項目を追加** または **選択した項目を削除** をクリックします。
7. **完了**をクリックして変更を保存します。

CLI をご利用の場合

1. root ユーザーとして Satellite Server に接続してから、利用可能なサブスクリプションを表示します。

```
# hammer subscription list \
--organization-id 1
```

2. サブスクリプションをホストにアタッチします。

```
# hammer host subscription attach \
--host host_name \
--subscription-id subscription_id
```

4.7. コンテンツホストのサブスクリプションの一括更新

この手順は、インストール後の複数のコンテンツホストへの同時変更に使用します。**hammer csv** コマンドは Satellite 6.6 以降では非推奨であるため、ホストを一括更新するには Satellite Web UI を使用する必要があることに注意してください。

手順

複数のコンテンツホストを更新するには、以下の手順を行います。

1. Satellite Web UI で、コンテキストが、使用する組織に設定されていることを確認します。
2. **ホスト** > **コンテンツホスト**に移動します。
3. 変更するサブスクリプションのコンテンツホストの行ごとに、該当するチェックボックスを選択してください。
4. **アクションの選択**一覧から**サブスクリプションの管理**を選択します。
5. 任意で、**検索**フィールドにキーと値を入力し、表示するサブスクリプションを絞り込みます。
6. 追加または削除するサブスクリプションの左側にあるチェックボックスを選択し、**選択した項目を追加**または**選択した項目を削除**をクリックします。
7. **完了**をクリックして変更を保存します。

第5章 RED HAT コンテンツのインポート

このセクションでは、Satellite の製品とリポジトリーを使用する方法、Satellite コンテンツが Red Hat コンテンツ配信ネットワーク (CDN) のコンテンツに合わせて最新の状態に保たれるように、同期プランを作成する方法を説明します。

5.1. RED HAT SATELLITE の製品

Satellite では、**製品** は複数のリポジトリーをグループ化する組織単位です。たとえば、Satellite では、Red Hat Enterprise Linux Server は **製品**で、この製品のリポジトリーは、さまざまなバージョン、アーキテクチャー、アドオンで構成されます。製品を使用すると、相互に依存関係にあるリポジトリーが、まとめて同期されます。Red Hat リポジトリーの場合は、製品は、リポジトリーを有効にした後に自動的に作成されます。

5.2. コンテンツの同期の概要

Satellite Server は、独自のリポジトリーと Red Hat CDN 上のリポジトリーを同期し、Satellite Server で Red Hat のリポジトリーの同一コピーが保持されるようにします。Satellite Server はこのリポジトリー情報を取得し、Satellite Server のファイルシステムに保存します。最初の同期後に、リポジトリーが CDN のリポジトリーに合わせて最新の状態が保たれるようにチェックする同期プランを作成できます。

ISO イメージを使用して最初の同期を実行できます。コンテンツ ISO の使用の詳細は、付録E *接続済み Satellite Server へのコンテンツ ISO のインポート*を参照してください。

5.3. グローバル HTTP プロキシーの無効化

設定されたプロキシーを使用せずにローカルリポジトリーを同期する場合には、**satellite-installer** ツールの実行時に設定されるグローバルプロキシーを無視できます。

手順

グローバル HTTP プロキシーを無視するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、変更する製品を選択します。
2. **リポジトリー** タブをクリックして、変更するリポジトリーを選択します。
3. **同期設定** エリアで、**グローバル HTTP Capsule を無視** に移動します。編集アイコンをクリックし、チェックボックスを選択して、**保存** をクリックします。

CLI をご利用の場合

```
# hammer repository update --id Repository_ID --ignore-global-proxy yes
```

5.4. ダウンロードポリシーの概要

Red Hat Satellite には、コンテンツメタデータのみをダウンロードし、実際のコンテンツのダウンロードは後で行うなど、RPM コンテンツの同期に関する複数のダウンロードポリシーが含まれています。

Satellite Server には以下のポリシーがあります。

即時

Satellite Server は、同期時にメタデータとパッケージをすべてダウンロードします。

オンデマンド

Satellite Server は同期時にメタデータのみをダウンロードし、Capsule、または直接接続されているクライアントが要求した場合に限り、ファイルシステムにパッケージを取得して保存します。Capsule の対応のリポジトリーを **即時** に設定した場合には、Satellite Server が強制的に全パッケージをダウンロードするため、この設定は効果がありません。

背景

Satellite Server は、最初の同期後にすべてのパッケージをダウンロードする背景タスクを作成します。

オンデマンド と **背景** のポリシーは、コンテンツの同期時間を短縮するので、**遅延同期** 機能として動作します。遅延同期機能は **yum** リポジトリーにのみ使用してください。通常と同じように、コンテンツビューにパッケージを追加し、通常通りにライフサイクル環境にプロモートすることができます。

Capsule Server に、以下のポリシーを提供します。

即時

Capsule Server は、同期時にメタデータとパッケージすべてをダウンロードします。Satellite Server で対応するリポジトリーを **オンデマンド** に設定した場合は、Satellite Server が強制的にすべてのパッケージをダウンロードするので、この設定を使用しないでください。

オンデマンド

Capsule Server は、同期時にメタデータのみをダウンロードします。Capsule Server は、直接接続されたクライアントから要求されると、ファイルシステムだけにパッケージを取得して保存します。**オンデマンド** ダウンロードポリシーを使用すると、Capsule Server でコンテンツを入手できない場合には、コンテンツが Satellite Server からダウンロードされます。

背景

Capsule Server は、最初の同期後にすべてのパッケージをダウンロードする背景タスクを作成します。

継承

Capsule Server は、Satellite Server で対応するリポジトリーから、リポジトリーのダウンロードポリシーを継承します。

このポリシーは、**--enable-foreman-proxy-plugin-pulp** を false に設定して Capsule をインストールまたは更新した場合は利用できません。

5.5. デフォルトのダウンロードポリシーの変更

Satellite が全組織に作成した新規リポジトリーに対して適用するデフォルトのダウンロードポリシーを設定できます。デフォルト値を変更しても、既存の設定は変更されません。

手順

リポジトリーのデフォルトのダウンロードポリシーを変更するには、以下の手順を行います。

1. Satellite Web UI で、**管理** > **設定**に移動します。
2. **コンテンツ** タブをクリックし、**デフォルトのリポジトリーダウンロードポリシー** を見つけます。
3. **値** フィールドで、編集アイコンをクリックします。
4. 必要なダウンロードポリシーを選択し、**保存** をクリックします。

CLI をご利用の場合

デフォルトのダウンロードポリシーを**immediate**、**on_demand**、**background** の 1 つに変更するには、以下のコマンドを入力します。

```
# hammer settings set \
--name default_download_policy \
--value immediate
```

5.6. リポジトリーのダウンロードポリシーの変更

リポジトリーのダウンロードポリシーも設定できます。

手順

リポジトリーのダウンロードポリシーを設定するには、以下の手順を行います。

1. Web UI で **コンテンツ** > **製品**に移動して、該当製品名をクリックします。
2. **リポジトリー** タブで必要なリポジトリー名をクリックし、**ポリシーのダウンロード** フィールドを見つけて、編集アイコンをクリックします。
3. リストから、必要なダウンロードポリシーを選択して、**Save** をクリックします。

CLI をご利用の場合

1. 組織のリポジトリーを一覧表示します。

```
# hammer repository list \
--organization-label organization-label
```

2. リポジトリーのダウンロードポリシーを **immediate**、**on_demand**、**background** の 1 つに変更します。

```
# hammer repository update \
--organization-label organization-label  \
--product "Red Hat Enterprise Linux Server" \
--name "Red Hat Enterprise Linux 7 Server Kickstart x86_64 7.5"  \
--download-policy immediate
```

5.7. RED HAT リポジトリーの有効化

同期するリポジトリーを選択するには、対象のリポジトリーが含まれる製品を先に特定して、適切なリリースバージョンとベースのアーキテクチャーをもとにリポジトリーを有効化する必要があります。Red Hat Enterprise Linux 8 の場合は、AppStream と BaseOS の両方のリポジトリーを有効化する必要があります。

オフラインの Satellite

オフラインの Satellite Serverを使用する場合は、コンテンツを同期する前に、コンテンツをローカル CDN サーバーと同期するように Satellite を設定する必要があります。詳細は、付録B *コンテンツをローカル CDN サーバーと同期するための Satellite の設定* を参照してください。

リポジトリーのバージョン管理

Red Hat Enterprise Linux オペレーティングシステムに 7 Server リポジトリー、または 7.**X** リポジトリーのいずれかを関連付けることの相違点は、7 Server リポジトリーには最新更新がすべて含まれますが、Red Hat Enterprise Linux 7.**X** リポジトリーは次のマイナーバージョンリリース以降の更新を取得しなくなる点です。キックスタートリポジトリーには、マイナーバージョンのみが含まれることに注意してください。

Red Hat Enterprise Linux 8 クライアントの場合

Red Hat Enterprise Linux 8 クライアントをプロビジョニングするには、**Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMS)** および **Red Hat Enterprise Linux 8 for x86_64 - BaseOS (RPMs)** のリポジトリーが必要です。

Red Hat Enterprise Linux 7 クライアントの場合

Red Hat Enterprise Linux 7 クライアントをプロビジョニングするには、**Red Hat Enterprise Linux 7 Server (RPMs)** リポジトリーが必要です。

手順

1. Satellite Web UI で、**コンテンツ** > **Red Hat リポジトリー**に移動します。
2. リポジトリーを検索するには、リポジトリー名を入力するか、**推奨のリポジトリー** ボタンをオンの位置に切り替えて、必要なリポジトリーの一覧を表示します。
3. 利用可能なリポジトリーのペインで、リポジトリーをクリックしてリポジトリーセットを展開します。
4. 必要な基本アーキテクチャーとリリースバージョンの横にある **有効** アイコンをクリックします。

CLI をご利用の場合

1. 製品を検索するには、以下のコマンドを実行します。

```
# hammer product list --organization "My_Organization"
```

2. 製品のリポジトリーのセットを一覧表示します。

```
# hammer repository-set list \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
```

3. 名前または ID 番号のいずれかを使用してリポジトリーを有効にします。リリースバージョンに **7Server**、基本アーキテクチャーに **x86_64** のように含めます。例を示します。

```
# hammer repository-set enable \
--name "Red Hat Enterprise Linux 7 Server (RPMs)" \
--releasever "7Server" \
--basearch "x86_64" \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
```

5.8. RED HAT リポジトリーの同期

リポジトリーを Red Hat CDN のリポジトリーと同期します。

Web UI をご利用の場合

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、同期が必要なリポジトリーを含む製品を選択します。
2. 同期が必要なリポジトリーを選択し、**今すぐ同期** をクリックします。

Web UI で同期の進捗状況を表示するには、**コンテンツ** > **同期の状態** に移動して、対応する製品またはリポジトリーツリーを展開します。

CLI をご利用の場合

Red Hat Enterprise Linux Server 製品内の有効済みリポジトリーを同期します。

```
# hammer product synchronize \
--name "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
```

また、各リポジトリーを個別に同期することもできます。製品内のすべてのリポジトリーを一覧表示し、対応するリポジトリーの ID 番号を使用して同期します。以下は例となります。

```
# hammer repository list \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
# hammer repository synchronize \
--name "Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server" \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
```

同期にかかる時間は、各リポジトリーのサイズとネットワーク接続の速度によって異なります。以下の表は、利用可能なインターネット帯域幅に応じてコンテンツの同期にかかる推定時間を示しています。

	単一パッケージ (10Mb)	マイナーリリース (750Mb)	メジャーリリース (6Gb)
256 Kbps	5 分 27 秒	6 時間 49 分 36 秒	2 日と 7 時間 55 分
512 Kbps	2 分 43.84 秒	3 時間 24 分 48 秒	1 日と 3 時間 57 分
T1 (1.5 Mbps)	54.33 秒	1 時間 7 分 54.78 秒	9 時間 16 分 20.57 秒
10 Mbps	8.39 秒	10 分 29.15 秒	1 時間 25 分 53.96 秒
100 Mbps	0.84 秒	1 分 2.91 秒	8 分 35.4 秒
1000 Mbps	0.08 秒	6.29 秒	51.54 秒

定期的に更新されるように同期プランを作成します。

5.9. 組織の全リポジトリーの同期

以下の手順を使用して、組織内の全リポジトリーを同期します。

手順

組織内の全リポジトリーを同期するには、Satellite Server で次の Bash スクリプトを実行します。

```
ORG="Your_Organization"

for i in $(hammer --no-headers --csv repository list --organization $ORG | awk -F, {'print $1'})
do
  hammer repository synchronize --id ${i} --organization $ORG --async
done
```

5.10. リポジトリーの復旧

リポジトリーが破損した場合は、高度な同期を使用してそれを復旧できます。3 つのオプションの中から選択できます。

最適同期

リポジトリーの同期時に、アップストリームの RPM との違いが検出されない RPM は回避します。

完全同期

検出した変更に関係なく、すべての RPM を同期します。特定の RPM が、アップストリームリポジトリーに存在しても、ローカルリポジトリーにダウンロードできなかった場合はこのオプションを使用します。

コンテンツの同期検証

すべての RPM を同期してから、すべての RPM のチェックサムをローカルで検証します。RPM のチェックサムがアップストリームと異なる場合は、RPM をもう一度ダウンロードします。このオプションは **yum** リポジトリーにのみ該当します。以下のいずれかのエラーが発生した場合に限りこのオプションを使用します。

- **yum** との同期中に、特定の RPM で **404** エラーが発生した場合。
- 特定のエラーが破損していることを示す **Package does not match intended download** エラーが発生した場合。

手順

高度なオプションを使用して特定のリポジトリーを同期するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品**に移動します。
2. 破損したリポジトリーを含む製品を選択します。
3. 同期するリポジトリーの名前を選択します。
4. **アクションの選択** メニューから、**高度な同期** を選択します。
5. オプションを選択し、**同期**をクリックします。

CLI をご利用の場合

1. リポジトリー ID の一覧を取得します。

```
# hammer repository list --organization "My_Organization"
```

2. 必要なオプションを使用して破損したリポジトリーを同期します。

   - 最適な同期の場合:

     ```
     # hammer repository synchronize --incremental true --id 1
     ```

   - 完全な同期の場合:

     ```
     # hammer repository synchronize --skip-metadata-check true --id 1
     ```

   - コンテンツを同期する場合:

     ```
     # hammer repository synchronize --validate-contents true --id 1
     ```

5.11. 同期速度の制限

同期の速度を制限して、利用可能な帯域幅の枯渇や、その他のパフォーマンスの問題を回避できます。これには、**PULP_CONCURRENCY** パラメーターおよび **max_speed** パラメーターを設定します。この設定はアップグレード時に上書きされる点に注意してください。アップグレード前に、変更したファイルをバックアップし、設定を復元できるようにしてください。

1. 並行して実行する同期ジョブの数を制御するには、**/etc/default/pulp_workers** ファイルの **PULP_CONCURRENCY** パラメーターを設定します。たとえば、平行して実行するジョブの数を 1 に設定するには、**PULP_CONCURRENCY** を 1 に設定します。

   ```
   PULP_CONCURRENCY=1
   ```

   デフォルトでは、CPU が 8 個より少なくなるシステムでは、**PULP_CONCURRENCY** が CPU の数に設定されます。8 個以上の場合は、8 に設定されます。

2. 同期するネットワークの最大速度 (バイト毎秒) を設定するには、**max_speed** パラメーターを設定します。このパラメーターは、**/etc/pulp/server/plugins.conf.d/** ディレクトリーで、インポーターごとに設定する必要があります。たとえば、RPM コンテンツを同期する最大速度を毎秒 10 バイトに設定するには、**/etc/pulp/server/plugins.conf.d/yum_importer.json** ファイルの **"max_speed"** パラメーターを 10 に設定します。

   ```
   # cat /etc/pulp/server/plugins.conf.d/yum_importer.json
   {
       "proxy_host": null,
       "proxy_port": null,
       "proxy_username": null,
       "proxy_password": null,
       "max_speed": 10
   }
   ```

3. 編集後にファイルの構文を検証します。

   ```
   # json_verify < /etc/pulp/server/plugins.conf.d/yum_importer.json
   JSON is valid
   ```

4. **satellite-maintain** サービスを再起動して変更を適用します。

   ```
   # satellite-maintain service restart
   ```

5.12. 同期プランの作成

同期プランでは、スケジュールされた日時にコンテンツをチェックし、更新します。Red Hat Satellite 6 では、同期プランを作成して製品を割り当てることができます。

手順

同期プランを作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **同期プラン** に移動して、**新規同期プラン** をクリックします。
2. **名前** フィールドに、プランの名前を入力します。
3. **説明** フィールドに、プランの説明を入力します。
4. **間隔** の一覧から、プランを実行する間隔を選択します。
5. **開始日** と **開始時間** のリストから、同期プランの実行を開始する日時を開始します。
6. **保存**をクリックします。
7. **製品** タブをクリックし、次に **追加** をクリックします。**Red Hat Enterprise Linux Server**製品を選択し、**選択した項目を追加** をクリックします。

CLI をご利用の場合

1. 同期プランを作成するには、以下のコマンドを実行します。

```
# hammer sync-plan create \
--name "Red Hat Products 2" \
--description "Example Plan for Red Hat Products" \
--interval daily \
--sync-date "2016-02-01 01:00:00" \
--enabled true \
--organization "My_Organization"
```

2. Red Hat Enterprise Linux Server 製品を同期プランに割り当てます。

```
# hammer product set-sync-plan \
--name "Red Hat Enterprise Linux Server" \
--sync-plan "Red Hat Products" \
--organization "My_Organization"
```

3. 組織の同期プランで利用可能なものを表示して、同期プランが作成されたことを確認します。

```
# hammer sync-plan list --organization "Default Organization"
```

5.13. 複数製品への同期プランの割り当て

以下の手順を使用して、最低でも 1 度同期され、1 つ以上リポジトリーが含まれる組織の製品に、同期プランを割り当てます。

手順

選択した製品に同期プランを割り当てるには、次の手順を実行します。

1. 次の Bash スクリプトを実行します。

```
ORG="Your_Organization"
SYNC_PLAN="daily_sync_at_3_a.m"

for i in $(hammer --no-headers --csv product list --organization $ORG --per-page 999 | grep -vi not_synced | awk -F, {'{ if ($5!=0) print $1}'})
do
  hammer sync-plan create --name $SYNC_PLAN --interval daily --sync-date "2018-06-20 03:00:00" --enabled true --organization $ORG
  hammer product set-sync-plan --sync-plan $SYNC_PLAN --organization $ORG --id $i
done
```

2. スクリプトの実行後、同期プランを割り当てた製品を表示します。

```
# hammer product list --organization $ORG --sync-plan $SYNC_PLAN
```

第6章 カスタムコンテンツのインポート

本章では、さまざまな種類のカスタムコンテンツを Satellite にインポートする方法を概説します。Satellite に ISO、Puppet モジュール、またはさまざまなコンテンツタイプをインポートする場合は、この章とほぼ同じ手順を実行します。

たとえば、特定のタイプのカスタムコンテンツの情報については、以下の章を使用できますが、基本的な手順は同じです。

- 12章*OSTree コンテンツの管理*
- 14章*ISO イメージの管理*
- 15章 *カスタムファイルタイプコンテンツの管理*
- 16章 *カスタム Puppet コンテンツの管理*

6.1. SATELLITE でのカスタム製品の使用

Red Hat Satellite 6 の Red Hat コンテンツとカスタムコンテンツはいずれも、次の点が類似しています。

- 製品とそのリポジトリー間の関係は同じであり、リポジトリーは引き続き同期する必要があります。
- カスタム製品には、クライアントがアクセスするサブスクリプション (Red Hat 製品に対するサブスクリプションと同様) が必要です。Red Hat Satellite 6 では、作成する各カスタム製品に対してサブスクリプションが作成されます。

RPM の作成およびパッケージングの詳細は、Red Hat Enterprise Linux ドキュメンテーションの『RPM パッケージングガイド』を参照してください。

6.2. カスタム SSL 証明書のインポート

カスタムコンテンツを作成する前に、必要なカスタム SSL 証明書をすべてインポートしてください。

RPM のダウンロードに SSL 証明書とキーが必要な場合は、Satellite に追加することができます。

1. Satellite Web UI で **コンテンツ** > **コンテンツの認証情報** に移動します。コンテンツ認証ウィンドウで、**コンテンツの認証情報の作成** をクリックします。
2. **名前** フィールドに、SSL 証明書の名前を入力します。
3. **タイプ** の一覧から、**SSL 証明書** を選択します。
4. **コンテンツの認証情報** フィールドに SSL 証明書を貼り付けるか、**参照** をクリックして SSL 証明書をアップロードします。
5. **保存**をクリックします。

6.3. カスタム GPG キーのインポート

カスタムコンテンツを作成する前に、必要な GPG キーをすべてインポートしてください。

前提条件

1. バージョン固有のリポジトリーパッケージのコピーをクライアントシステムにダウンロードする。

```
$ wget http://www.example.com/9.5/example-9.5-2.noarch.rpm
```

2. RPM ファイルをインストールせずに抽出する。

```
$ rpm2cpio example-9.5-2.noarch.rpm | cpio -idmv
```

GPG キーは、その抽出ファイルに相対的な場所である **etc/pki/rpm-gpg/RPM-GPG-KEY-EXAMPLE-95** に存在します。

手順

GPG キーをインポートするには、以下の手順を実行してください。

1. Satellite Web UI で**コンテンツ** > **コンテンツの認証情報**に移動して、ウィンドウの右上の**コンテンツの認証情報の作成**をクリックします。
2. リポジトリーの名前を入力し、**タイプ** の一覧から **GPG キー** を選択します。
3. GPG キーを **コンテンツ認証情報の内容** フィールドに貼り付けるか、**参照**をクリックし、インポートする GPG キーファイルを選択します。
   カスタムリポジトリーに複数の GPG キーで署名されたコンテンツが含まれている場合、必要なすべての GPG キーを **コンテンツ認証情報の内容** フィールドに入力しますが、この時、各キーの間に新たな行を入れる必要があります。以下に例を示します。

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFy/HE4BEADttv2TCPzVrre+aJ9f5QsR6oWZMm7N5Lwxjm5x5zA9BLiPPGFN
4aTUR/g+K1S0aqCU+ZS3Rnxb+6fnBxD+COH9kMqXHi3M5UNzbp5WhCdUpISXjjpU
XIFFWBPuBfyr/FKRknFH15P+9kLZLxCpVZZLsweLWCuw+JKCMmnA
=F6VG
-----END PGP PUBLIC KEY BLOCK-----

-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFw467UBEACmREzDeK/kuScCmfJfHJa0Wgh/2fbJLLt3KSvsgDhORIptf+PP
OTFDlKuLkJx99ZYG5xMnBG47C7ByoMec1j94YeXczuBbynOyyPlvduma/zf8oB9e
Wl5GnzcLGAnUSRamfqGUWcyMMinHHIKIc1X1P4I=
=WPpI
-----END PGP PUBLIC KEY BLOCK-----
```

4. **保存**をクリックします。

CLI をご利用の場合

1. GPG キーを Satellite Server にコピーします。

```
$ scp ~/etc/pki/rpm-gpg/RPM-GPG-KEY-EXAMPLE-95 root@satellite.example.com:~/.
```

2. GPG キーを Satellite にアップロードします。

```
# hammer gpg create \
```

```
--key ~/RPM-GPG-KEY-EXAMPLE-95 \
--name "My_Repository" \
--organization "My_Organization"
```

6.4. カスタム製品の作成

以下の手順を使用してカスタム製品を作成し、後でリポジトリーを追加できます。

手順

カスタム製品を作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **製品** に移動して、**製品の作成** をクリックします。
2. **名前** フィールドで、製品の名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容に基づいて **ラベル** フィールドに自動的に入力されます。
3. オプション: **GPG キー** の一覧から、製品の GPG キーを選択します。
4. オプション: **SSL CA 証明書** の一覧から、製品の SSL CA 証明書を選択します。
5. オプション: **SSL クライアント証明書** の一覧から、製品の SSL クライアント 証明書を選択します。
6. オプション: **SSL クライアントキー** の一覧から、製品の SSL クライアントキーを選択します。
7. オプション: **同期プラン** の一覧から、既存の同期プランを選択するか、**同期プランの作成** をクリックし、製品要件の同期プランを作成します。
8. **説明**フィールドには、製品の説明を入力します。
9. **保存**をクリックします。

CLI をご利用の場合

製品を作成するには、以下のコマンドを実行します。

```
# hammer product create \
--name "My_Product" \
--sync-plan "Example Plan" \
--description "Content from My Repositories" \
--organization "My_Organization"
```

6.5. カスタム RPM リポジトリーの追加

以下の手順を使用して、Satellite でカスタム RPM リポジトリーを追加します。

Satellite Web UI の製品ウィンドウには、**リポジトリーの検出** 機能があり、URL からすべてのリポジトリーを見つけ、どのリポジトリーを自分のカスタム製品に追加するかを選択できます。たとえば、**リポジトリーの検出** を使用して、**http://yum.postgresql.org/9.5/redhat/** を検索し、Red Hat Enterprise Linux の各バージョンおよびアーキテクチャー用のすべてのリポジトリーを一覧表示することができるので、1つのソースから複数のリポジトリーをインポートする時間を節約できます。

カスタム RPM のサポート

Red Hat では、サードパーティーのサイトに、直接接続されているアップストリーム RPM はサポートしていません。これらの RPM は同期プロセスのデモに使用されます。これらの RPM について問題がある場合は、サードパーティーの開発者に連絡してください。

手順

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、使用する製品を選択して、**リポジトリーの作成** をクリックします。
2. **名前** フィールドで、リポジトリーの名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容に基づいて **ラベル** フィールドに値が自動的に入力されます。
3. **タイプ** の一覧から、リポジトリーのタイプを選択します。RPM ファイル (**yum**)、Puppet モジュール (**puppet**)、または Docker イメージ (**docker**) のいずれかのリポジトリーを選択できます。
4. **URL** フィールドに、ソースとして使用する外部リポジトリーの URL を入力します。
5. **ダウンロードポリシー** の一覧から、Satellite Server が実行する同期の種類を選択します。
6. **同期時のミラー** チェックボックスが選択されていることを確認します。アップストリームのリポジトリーにないコンテンツが同期中に削除されるようにします。
7. **チェックサム** の一覧から、リポジトリーのチェックサムタイプを選択します。
8. オプション: 必要に応じて、**HTTP 経由で公開** チェックボックスの選択を解除して、このリポジトリーの HTTP 経由での公開を無効にできます。
9. オプション: **GPG キー** の一覧から、製品の GPG キーを選択します。
10. **保存**をクリックします。

製品のウィンドウで即時同期を実行する場合は、**今すぐ同期** をクリックします。

CLI をご利用の場合

1. 以下のコマンドを実行してリポジトリーを作成します。

```
# hammer repository create \
--name "My_Repository" \
--content-type "yum" \
--publish-via-http true \
--url http://yum.postgresql.org/9.5/redhat/rhel-7-x86_64/ \
--gpg-key "My_Repository" \
--product "My_Product" \
--organization "My_Organization"
```

2. リポジトリーを同期します。

```
# hammer repository synchronize \
--name "My_Repository" \
--product "My Product" \
--organization "My_Organization"
```

第7章 アプリケーションライフサイクルの管理

本章では、Satellite のアプリケーションライフサイクルと、Satellite と Capsule のアプリケーションライフサイクルの作成と削除の方法を概説します。

7.1. アプリケーションライフサイクルの概要

アプリケーションライフサイクル は、Red Hat Satellite 6 のコンテンツ管理機能の中心となる概念です。アプリケーションライフサイクルは、特定の段階で特定のシステムとソフトウェアがどのように見えるかを定義します。たとえば、アプリケーションライフサイクルが単純な場合には、開発段階と実稼働段階のみになります。このような場合に、アプリケーションライフサイクルは以下のようになります。

- 開発
- 実稼働

ただし、テストやベータリリースなど、より多くの段階が含まれ、アプリケーションライフサイクルが複雑になる場合があります。

- 開発
- テスト
- Beta リリース
- 実稼働

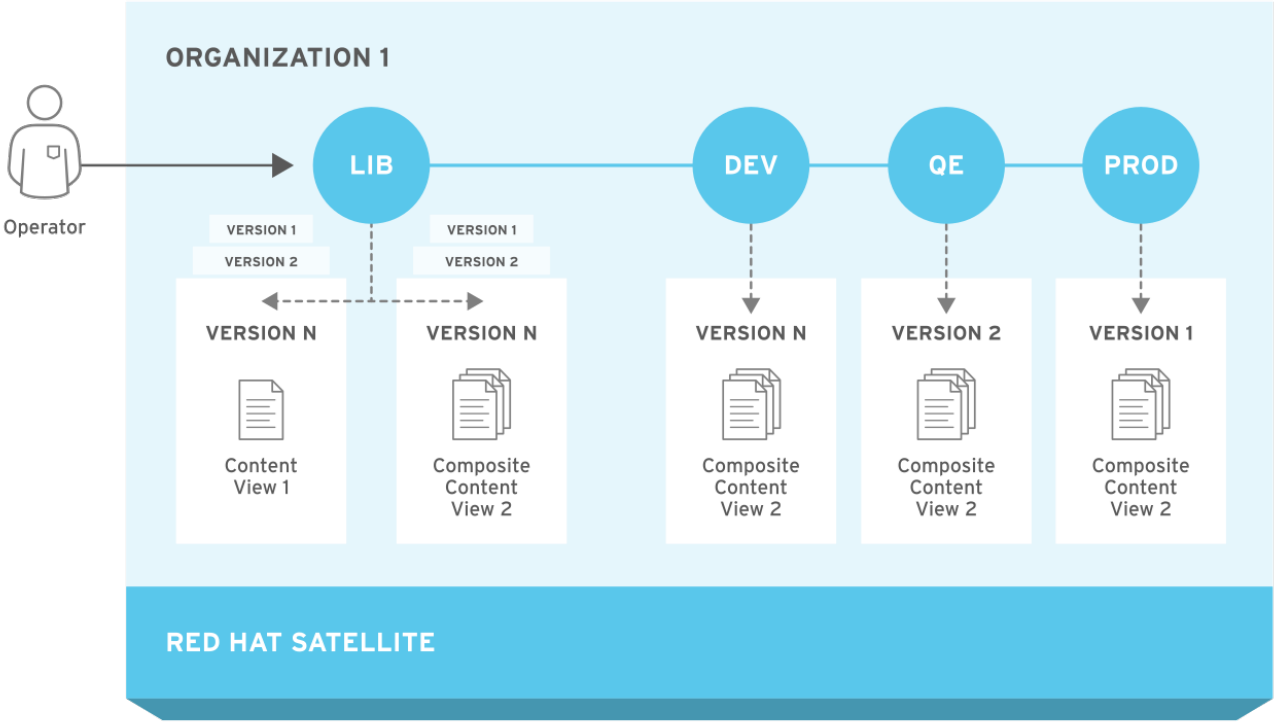
Red Hat Satellite 6 は、仕様に合わせて各アプリケーションライフサイクルの段階をカスタマイズする方法を提供します。

Red Hat Satellite 6 では、アプリケーションライフサイクルの各段階は**環境**と呼ばれます。各環境はコンテンツの特定のコレクションを使用します。Red Hat Satellite 6 では、これらのコンテンツコレクションはコンテンツビューとして定義されます。各コンテンツビューは、特定の環境に含めるリポジトリー、パッケージ、および Puppet モジュールを定義できるフィルターとなります。これにより、ユーザーは各環境に指定する特定のコンテンツセットを定義できるようになります。

たとえば、メールサーバーの場合は、実際に使用する実稼働レベルのサーバーと、最新のメールサーバーパッケージをテストするテストサーバーという、単純なアプリケーションライフサイクルのみを必要とします。テストサーバーが初期段階をパスすると、実稼働レベルのサーバーで新パッケージを使用するように設定できます。

別の例としては、ソフトウェア製品の開発ライフサイクルがあります。開発環境でソフトウェアの新しい部分を開発するには、品質保証環境でソフトウェアをテストしてベータ版としてプレリリースした後に、実稼働レベルのアプリケーションとしてソフトウェアをリリースします。

図7.1 Red Hat Satellite 6 アプリケーションライフサイクル



7.2. アプリケーションライフサイクルでのコンテンツのプロモーション

アプリケーションライフサイクルチェーンで、ある環境から次の環境へコンテンツが移動することは、**プロモーション** と呼ばれます。

Satellite ライフサイクル環境のコンテンツプロモーションの例

各環境には、Red Hat Satellite 6 に登録したシステムが含まれ、そのシステムがアクセスできるのは、各環境に関連するリポジトリーに限られます。別の環境にパッケージをプロモートすると、プロモート先の環境のリポジトリーはパッケージの新バージョンを受け取り、その結果、プロモート先の環境にある各システムはパッケージを新バージョンに更新できます。

開発	テスト	実稼働
example_software-1.1-0.noarch.rpm	**example_software**-1.0-0.noarch.rpm	**example_software**-1.0-0.noarch.rpm

パッチ開発の完了後には、RPM をテスト環境にプロモートして、品質保証エンジニアチームがパッチをレビューできるようにします。この時点では、アプリケーションライフサイクルに以下のパッケージバージョンが各環境に含まれます。

開発	テスト	実稼働
example_software-1.1-0.noarch.rpm	**example_software-1.1-0.noarch.rpm**	**example_software**-1.0-0.noarch.rpm

品質保証エンジニアチームがパッチのレビューを行う間、開発チームは **example_software** 2.0 の作業に着手します。このため、アプリケーションライフサイクルは以下のようになります。

開発	テスト	実稼働
exampleware-2.0-0.noarch.rpm	exampleware-1.1-0.noarch.rpm	exampleware-1.0-0.noarch.rpm

品質保証エンジニアチームがパッチのレビューを完了します。これで、**example_software** 1.1 をリリースする準備が完了します。1.1 を**実稼働**環境にプロモートします。

開発	テスト	実稼働
example_software-2.0-0.noarch.rpm	**example_software**-1.1-0.noarch.rpm	**example_software-1.1-0.noarch.rpm**

開発チームが **example_software** 2.0 の作業を完了し、テスト環境にプロモートします。

開発	テスト	実稼働
example_software-2.0-0.noarch.rpm	**example_software-2.0-0.noarch.rpm**	**example_software**-1.1-0.noarch.rpm

最後に品質保証エンジニアチームがこのパッケージのレビューを行います。レビューが完了したら、パッケージを **実稼働** 環境にプロモートします。

開発	テスト	実稼働
exampleware-2.0-0.noarch.rpm	exampleware-2.0-0.noarch.rpm	**exampleware-2.0-0.noarch.rpm**

詳細については、「コンテンツビューのプロモート」を参照してください。

7.3. ライフサイクル環境パスの作成

ソフトウェアを開発およびリリースするためのアプリケーションライフサイクルを作成するには、**ライブラリー** 環境を初期環境として使用して、環境パスを作成します。次に、オプションで環境パスに環境を追加します。

手順

1. Satellite Web UI で、**コンテンツ** > **ライフサイクル環境** に移動します。
2. **新規環境パス** をクリックして、新しいアプリケーションライフサイクルを開始します。
3. **名前** フィールドに、環境の名前を入力します。
4. **説明** フィールドに、環境の説明を入力します。
5. **保存**をクリックします。
6. オプション: 環境パスに環境を追加するには、**新しい環境の追加** をクリックし、**名前** と **説明** フィールドを入力します。続いて、**以前の環境** 一覧から以前の環境を選択します。

CLI をご利用の場合

1. 環境パスを作成するには、**hammer lifecycle-environment create** コマンドを入力し、**--prior** オプションを使用してライブラリー環境を指定します。

```
# hammer lifecycle-environment create \
--name "Environment Path Name" \
--description "Environment Path Description" \
--prior "Library" \
--organization "My_Organization"
```

2. オプション: 環境パスに環境を追加するには、**hammer lifecycle-environment create** コマンドを入力し、**--prior** オプションを使用して親環境を指定します。

```
# hammer lifecycle-environment create \
--name "Environment Name" \
--description "Environment Description" \
--prior "Prior Environment Name" \
--organization "My_Organization"
```

3. ライフサイクル環境のチェーンを表示するには、以下のコマンドを入力します。

```
# hammer lifecycle-environment paths --organization "My_Organization"
```

7.4. SATELLITE SERVER からのライフサイクル環境の削除

以下の手順を使用して、ライフサイクル環境を削除します。

手順

ライフサイクル環境を削除するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **ライフサイクル環境** に移動します。
2. 削除するライフサイクル環境の名前をクリックし、**環境の削除**をクリックします。
3. **削除** をクリックして環境を削除します。

CLI をご利用の場合

1. 組織のライフサイクル環境を一覧表示し、削除するライフサイクル環境の名前を確認します。

```
# hammer lifecycle-environment list --organization "My_Organization"
```

2. **hammer lifecycle-environment delete** コマンドを使用して、環境を削除します。

```
# hammer lifecycle-environment delete \
--name "your_environment" \
--organization "My_Organization"
```

7.5. CAPSULE SERVER からのライフサイクル環境の削除

ライフサイクル環境がホストシステムに適さなくなった場合や、環境が Capsule Server に誤って追加された場合、Capsule Server からライフサイクル環境を削除できます。

Capsule からライフサイクル環境を削除するには、Satellite Web UI と Hammer の両方を使用できます。

手順

ライフサイクル環境を Capsule Server から削除するには、以下の手順を行います。

1. Satellite Web UI で、**インフラストラクチャー** > **Capsules** に移動し、ライフサイクルを削除する Capsule を選択します。
2. **Edit** をクリックしてから、**Life Cycle Environments** タブをクリックします。
3. 右のメニューから、Capsule から削除するライフサイクル環境を選択し、**送信** をクリックします。
4. Capsule のコンテンツを同期するには、**概要**タブをクリックしてから**同期**ボタンをクリックします。
5. **最適化された同期**または **完全な同期**を選択します。

CLI をご利用の場合

ライフサイクル環境を Capsule Server から削除するには、以下の手順を行います。

1. 一覧から使用する Capsule Server を選択し、その **ID** を書き留めます。

```
# hammer capsule list
```

2. Capsule Server の詳細を確認するには、以下のコマンドを実行します。

```
# hammer capsule info --id capsule_id
```

3. Capsule Server に現在アタッチされているライフサイクル環境の一覧を確認し、**環境 ID** を書き留めます。

```
# hammer capsule content lifecycle-environments \
--id capsule_id
```

4. Capsule Server からのライフサイクル環境を削除します。

```
# hammer capsule content remove-lifecycle-environment \
--id capsule_id \
--lifecycle-environment-id lifecycle_environment_id
```

   Capsule Server から削除するすべてのライフサイクル環境に対してこの手順を繰り返します。

5. Satellite Server の環境にあるコンテンツを Capsule Server に同期します。

```
# hammer capsule content synchronize \
--id capsule_id
```

7.6. CAPSULE SERVER へのライフサイクル環境の追加

Capsule Server でコンテンツ機能が有効な場合は、環境を追加して、Capsule が Satellite Server のコンテンツを同期し、コンテンツをホストシステムに提供できるようにする必要があります。

リポジトリーが CDN から更新されるたびに自動で Capsule が同期されるようになるので、**ライブラリー**ライフサイクル環境を Capsule Server に割り当てないでください。自動的に同期されると、Capsule 上の複数のシステムリソースや Satellite と Capsule 間のネットワーク帯域幅、および Capsule 上の利用可能なディスク領域が消費される可能性があります。

Satellite Server の Hammer CLI または Satellite Web UI を使用できます。

手順

ライフサイクル環境を Capsule Server に追加するには、以下の手順を実行します。

1. Satellite Web UI で、**インフラストラクチャー** > **Capsule** に移動し、ライフサイクルを追加する Capsule を選択します。
2. **Edit** をクリックしてから、**Life Cycle Environments** タブをクリックします。
3. 左のメニューから、Capsule に追加するライフサイクル環境を選択し、**送信**をクリックします。
4. Capsule のコンテンツを同期するには、**概要**タブをクリックしてから、**同期** をクリックします。
5. **最適化された同期**または **完全な同期**を選択します。
   各同期タイプの定義については、「リポジトリーの復旧」 を参照してください。

CLI をご利用の場合

1. Satellite Server で、Capsule Server の全一覧を表示するには、以下のコマンドを入力します。

```
# hammer capsule list
```

   ライフサイクルを追加する Capsule の Capsule ID を書き留めます。

2. その ID を使用して、Capsule Server の詳細を確認します。

```
# hammer capsule info --id capsule_id
```

3. 利用可能なライフサイクル環境を確認し、環境 ID を書き留めます。

```
# hammer capsule content available-lifecycle-environments \
--id capsule_id
```

4. Capsule Server で利用可能なライフサイクル環境を表示するには、以下のコマンドを入力して、組織名と ID をメモします。

```
# hammer capsule content available-lifecycle-environments --id capsule_id
```

5. ライフサイクル環境を Capsule Server に追加します。

```
# hammer capsule content add-lifecycle-environment \
--id capsule_id --organization "My_Organization" \
--environment-id environment_id
```

Capsule Server に追加する各ライフサイクル環境に対して手順を繰り返します。

6. Satellite から Capsule にコンテンツを同期します。

   - Satellite Server 環境のすべてのコンテンツを Capsule Server に同期するには、以下のコマンドを実行します。

     ```
     # hammer capsule content synchronize --id capsule_id
     ```

   - Satellite Server 環境の特定のライフサイクル環境を Capsule Server と同期するには、以下のコマンドを実行します。

     ```
     # hammer capsule content synchronize --id external_capsule_id \
     --environment-id environment_id
     ```

第8章 コンテンツビューの管理

Red Hat Satellite 6 では、コンテンツビューを使用して リポジトリーからカスタマイズリポジトリーを作成します。作成するには、使用するリポジトリーを定義し、特定のフィルターをコンテンツに適用します。このフィルターにはパッケージフィルター、パッケージグループフィルター、およびエラータフィルターが含まれます。コンテンツビューを使用して、特定の環境が使用するソフトウェアのバージョンを定義できます。たとえば、**実稼働** 環境では古いバージョンのパッケージを含むコンテンツビューを使用し、**開発** 環境では新しいバージョンのパッケージを含むコンテンツビューを使用するなどです。

コンテンツビューは各環境でリポジトリーセットを作成し、Satellite Server が保存して管理します。アプリケーションライフサイクルの次の環境にコンテンツビューをプロモートすると、それに対応する Satellite Server のリポジトリーがパッケージを更新して公開します。

	開発	テスト	実稼働
コンテンツビューのバージョンとコンテンツ	バージョン 2 - **example_software**-1.1-0.noarch.rpm	バージョン 1 - **example_software**-1.0-0.noarch.rpm	バージョン 1 - **example_software**-1.0-0.noarch.rpm

テストと実稼働のリポジトリーには **example_software-1.0-0.noarch.rpm** パッケージが含まれています。コンテンツビューのバージョン 2 を開発環境からテスト環境にプロモートすると、テスト環境のリポジトリーが再作成され、**example_software-1.1-0.noarch.rpm** パッケージが含まれるようになります。

	開発	テスト	実稼働
コンテンツビューのバージョンとコンテンツ	バージョン 2 - **example_software**-1.1-0.noarch.rpm	**バージョン 2 - example_software-1.1-0.noarch.rpm**	バージョン 1 - **example_software**-1.0-0.noarch.rpm

こうすることで、システムは特定の環境専用となり、その環境が新しいコンテンツビューを使用する際には更新を受け取ることができます。

スナップショットのフィルタリングと作成に使用するコンテンツビューを作成する一般的なワークフローは、以下の通りです。

1. コンテンツビューを作成します。
2. コンテンツビューに使用するリポジトリーと Puppet モジュールを追加します。
3. 任意で、コンテンツビューのコンテンツを絞り込むフィルターを 1 つまたは複数作成します。
4. オプションで、コンテンツビューのパッケージの依存関係を解決します。
5. コンテンツビューを公開します。
6. 任意で、コンテンツビューを別の環境にプロモートします。
7. コンテンツホストをコンテンツビューにアタッチします。

コンテンツビューでリポジトリーを割り当てないと、**/etc/yum.repos.d/redhat.repo** ファイルは空になり、登録済みのシステムで更新を受け取ることができません。

ホストを関連付けるコンテンツビューは1つだけにすることができます。複数のコンテンツビューにホストを関連付けるには、複合コンテンツビューを作成します。詳細は「複合コンテンツビューの作成」を参照してください。

パッケージの依存関係の解決

パッケージの依存関係は、パッケージ管理を複雑化させます。コンテンツビューでパッケージの依存関係を管理する方法は、「パッケージの依存関係の解決」を参照してください。

8.1. コンテンツビューの作成

以下の手順を使用してシンプルなコンテンツビューを作成します。

前提条件

コンテンツビューごとに、コンテンツビューでパッケージの依存関係を解決するかどうかを指定できますが、Satellite のデフォルト設定を変更して、全コンテンツビューでパッケージの解決を有効化または無効化できます。詳細情報は、「パッケージの依存関係の解決」を参照してください。

手順

コンテンツビューを作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **コンテンツビュー**に移動して、**新規ビューの作成**をクリックします。
2. **名前** フィールドで、ビューの名前を入力します。Satellite では、**ラベル** フィールドは、入力した名前をもとに自動的に入力されます。
3. **説明** フィールドに、ビューの説明を入力します。
4. オプション: 対象のコンテンツビューを公開するたびに自動的に依存関係を解決する場合には、**依存関係の解決** のチェックボックスにチェックを入れます。依存関係の解決を選択すると、公開にかかる時間が長くなり、使用するコンテンツビューフィルターが無視される可能性があります。また、その結果、エラータの依存関係を解決する時に、エラーが発生する可能性があります。
5. **保存** をクリックして、コンテンツビューを作成します。
6. **リポジトリーの選択** エリアで、コンテンツビューに追加するリポジトリーを選択して、**リポジトリーの追加** をクリックします。
7. **新規バージョンの公開** をクリックし、**説明** フィールドに、変更をログに記録するバージョンについての情報を入力します。
8. **保存**をクリックします。
9. オプション: Yum リポジトリーでメタデータを再生成させるには、コンテンツビューバージョンの **アクション** リストから、**リポジトリーメタデータの再生成** を選択します。

コンテンツビューウィンドウでコンテンツビューを表示できます。コンテンツビューに関する詳細情報を表示するには、コンテンツビュー名をクリックします。

コンテンツビューにホストを登録するには、『**ホストの管理**』ガイドの「ホストの登録」を参照してください。

Hammer CLI を使用したコンテンツビューの作成

1. リポジトリー ID の一覧を取得します。

```
# hammer repository list --organization "My_Organization"
```

2. コンテンツビューを作成し、リポジトリーに追加します。

```
# hammer content-view create \
--name "Example_Content_View" \
--description "Example Content View" \
--repository-ids 1,2 \
--organization "My_Organization"
```

--repository-ids オプションを使用すると、**hammer repository list** コマンドの出力で ID が分かります。

3. ビューを公開します。

```
# hammer content-view publish \
--name "Example_Content_View" \
--description "Example Content View" \
--organization "My_Organization"
```

4. オプション: 既存のコンテンツビューにリポジトリーを追加するには、以下のコマンドを入力します。

```
# hammer content-view add-repository \
--name "Example_Content_View" \
--organization "My_Organization" \
--repository-id repository_ID
```

Satellite Server がビューの新バージョンを作成し、ライブラリー環境に公開します。

8.2. モジュールストリームの表示

Satellite で、コンテンツビューにリポジトリーのモジュールストリームを表示できます。

手順

コンテンツビューにリポジトリーのモジュールストリームを表示するには、次の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **コンテンツビュー** に移動して、表示するモジュールが含まれるコンテンツビューを選択します。
2. **バージョン** タブをクリックし、表示するコンテンツビューバージョンを選択します。
3. **モジュールストリーム** タブをクリックして、コンテンツビューに利用できるモジュールストリームを表示します。
4. **フィルター** フィールドを使用して、モジュールのリストを絞り込みます。
5. モジュールについての情報を表示するには、モジュールをクリックします。

8.3. PUPPET モジュールを含むコンテンツビューの作成

以下の手順を使用して、フィルターなしで、リポジトリーを 1 つ使用するコンテンツビューを作成します。

前提条件

開始前に、カスタムの製品内のリポジトリーに、必要な Puppet モジュールをアップロードします。詳細は、『**Puppet Guide**』の「Adding Puppet Modules to Red Hat Satellite 6 」を参照してください。

手順

Puppet モジュールありのコンテンツビューを作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **コンテンツビュー**に移動して、**新規ビューの作成**をクリックします。
2. **名前** フィールドで、ビューの名前を入力します。Red Hat Satellite 6 では、入力した名前から、**ラベル** フィールドに自動的に入力されます。
3. **説明** フィールドに、ビューの説明を入力します。
4. **保存**をクリックします。
5. **リポジトリーの選択** エリアで、コンテンツビューに追加するリポジトリーを選択して、**リポジトリーの追加** をクリックします。
6. **Puppet モジュール** タブで **新規モジュールの追加** をクリックします。
7. 追加するモジュールを探し、**バージョンの選択** をクリックします。
8. **最新を使用** のエントリーに移動し、**アクション** コラムの **バージョンの選択** をクリックします。
9. 公開するには、**バージョン** タブをクリックし、**新規バージョンの公開** をクリックします。**説明** フィールドに、変更をログに記録する説明を入力し、**保存** をクリックします。

コンテンツビューにホストを登録するには、『**ホストの管理**』ガイドの「ホストの登録」を参照してください。

CLI をご利用の場合

Puppet モジュールをコンテンツビューに追加するには、以下のコマンドを実行します。

```
# hammer content-view puppet-module add \
--content-view cv_name \
--name module_name
```

8.4. コンテンツビューのプロモート

以下の手順を使用して、異なるライフサイクル環境全体に、コンテンツビューをプロモートします。

コンテンツビュープロモーションのパーミッション要件

環境にコンテンツビューをプロモートするために、管理者以外のユーザーには以下の 2 つのパーミッションが必要になります。

1. **promote_or_remove_content_views**
2. **promote_or_remove_content_views_to_environment**

promote_or_remove_content_views パーミッションで、ユーザーがプロモートできるコンテンツビューを制限します。

promote_or_remove_content_views_to_environment パーミッションで、コンテンツビューのプロモート先となる環境を制限します。

上記のパーミッションを使用すると、指定の環境に指定のコンテンツビューをプロモートし、それ以外の環境にはプロモートできないように、ユーザーパーミッションを割り当てることができます。たとえば、テスト環境へのプロモーションはできるが、実稼働環境にはできないようにユーザーを制限できます。

コンテンツビューをプロモートできるようにするには、ユーザーに両パーミッションを割り当てる必要があります。

手順

コンテンツビューをプロモートするには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **コンテンツビュー** に移動して、プロモートするコンテンツビューを選択します。
2. コンテンツビューの **バージョン** タブをクリックします。
3. プロモートするバージョンを選択し、**アクション** 列で、**プロモート** をクリックします。
4. コンテンツビューをプロモートする環境を選択し、**バージョンのプロモート** をクリックします。
5. **プロモート** ボタンを再度クリックします。今度は **テスト** 環境を選択して **バージョンのプロモート** をクリックします。
6. 最後に **プロモート** ボタンを再度押します。Production 環境を選択し、**バージョンのプロモート** をクリックします。

これでこのコンテンツビューのリポジトリーが全環境に表示されます。

CLI をご利用の場合

- コンテンツビューのプロモートには、毎回 **hammer content-view version promote** を使用します。

```
# hammer content-view version promote \
--content-view "Database" \
--version 1 \
--to-lifecycle-environment "Development" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Database" \
--version 1 \
--to-lifecycle-environment "Testing" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Database" \
--version 1 \
--to-lifecycle-environment "Production" \
--organization "My_Organization"
```

これで Database のコンテンツが全環境で利用可能になります。

コンテンツビューにホストを登録するには、『**ホストの管理**』ガイドの「ホストの登録」を参照してください。

8.5. 組織内の全ライフサイクル環境へのコンテンツビューのプロモート

以下の手順を使用して、組織内の全ライフサイクル環境にコンテンツビューをプロモートします。

手順

組織内の全ライフサイクル環境にコンテンツビューのバージョンをプロモートするには、次の手順を実行します。

1. 組織内の全ライフサイクル環境に、ライブラリーから選択したコンテンツビューバージョンをプロモートするには、次の Bash スクリプトを実行します。

```
ORG="Your_Organization"
CVV_ID=3

for i in $(hammer --no-headers --csv lifecycle-environment list --organization $ORG | awk -F, {'print $1'} | sort -n)
do
  hammer content-view version promote --organization $ORG --to-lifecycle-environment-id $i --id $CVV_ID
done
```

2. コンテンツビューのバージョンに関する情報を表示して、必要なライフサイクル環境にプロモートされていることを確認します。

```
# hammer content-view version info --id 3
```

8.6. 複合コンテンツビューの概要

複合コンテンツビューは、複数のコンテンツビューのコンテンツを組み合わせます。たとえば、オペレーティングシステムとアプリケーションの管理に別々のコンテンツビューを使用していたとします。複合コンテンツビューを使用して、この 2 つのコンテンツビューのコンテンツを新規リポジトリーに統合できます。元のコンテンツビューのリポジトリーはそのまま存在し、統合したコンテンツには新規リポジトリーを使用します。

さまざまなデータベースサーバーをサポートするアプリケーションを開発する場合には、**example_application** は次のように表示されます。

example_software
アプリケーション
データベース
オペレーティングシステム

4 つの異なるコンテンツビューの例:

- Red Hat Enterprise Linux (オペレーティングシステム)
- PostgreSQL (データベース)
- MariaDB (データベース)
- **example_software** (アプリケーション)

以前のコンテンツビューから、2 つの複合コンテンツビューを作成できます。

PostgreSQL データベースの複合コンテンツビューの例:

複合コンテンツビュー 1: PostgreSQL の example_software
example_software (アプリケーション)
PostgreSQL (データベース)
Red Hat Enterprise Linux (オペレーティングシステム)

MariaDB の複合コンテンツビューの例:

複合コンテンツビュー 2: MariaDB の example_software
example_software (アプリケーション)
MariaDB (データベース)
Red Hat Enterprise Linux (オペレーティングシステム)

これで各コンテンツビューは別個に管理して公開されます。アプリケーションのバージョンを作成すると、複合コンテンツビューの新バージョンを公開することになります。複合コンテンツビューの作成時に、**自動公開** オプションを選択することも可能です。自動公開オプションを選択して、複合コンテンツビューに含まれるコンテンツビューが再公開されると、自動的に複合コンテンツビューが再公開されます。

リポジトリーの制限事項

複合コンテンツビューは、各リポジトリーで 1 つしか許可されません。たとえば、同じリポジトリーを使用したコンテンツビューを 2 つ追加しようとすると、Satellite Server はエラーをレポートします。

8.7. 複合コンテンツビューの作成

手順

複合コンテンツビューを作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **コンテンツビュー**に移動して、**新規ビューの作成**をクリックします。

2. **名前** フィールドで、ビューの名前を入力します。Red Hat Satellite 6 では、入力した名前から、**ラベル** フィールドに自動的に入力されます。

3. **説明** フィールドに、ビューの説明を入力します。

4. **複合ビュー?** チェックボックスを選択して、複合コンテンツビューを作成します。

5. オプション: コンテンツビューが再公開されると複合コンテンツビューが自動的に再公開されるようにするには、**自動公開** チェックボックスを選択します。

6. **保存**をクリックします。

7. **コンテンツビューの追加** エリアで、複合コンテンツビューに追加するコンテンツビューを選択して、**コンテンツビューの追加** をクリックします

8. **新規バージョンの公開** をクリックして、複合コンテンツビューを公開します。**説明** フィールドに説明を入力し、**保存** をクリックします。

9. **プロモート** をクリックし、複合コンテンツビューをプロモートするライフサイクル環境を選択し、説明を入力して、**バージョンのプロモート** をクリックします。

CLI をご利用の場合

1. 複合コンテンツビューを作成する前に、既存のコンテンツビューのバージョン ID を一覧表示します。

```
# hammer content-view version list \
--organization "My_Organization"
```

2. 新しい複合コンテンツビューを作成します。**--auto-publish** オプションを **yes** に設定すると、そのコンテンツビューが含まれるコンテンツビューが再公開されると、複合コンテンツビューは自動的に再公開されます。

```
# hammer content-view create \
--composite \
--auto-publish yes \
--name "Example_Composite_Content_View" \
--description "Example Composite Content View" \
--organization "My_Organization"
```

3. コンポーネントのコンテンツビューを複合コンテンツビューに追加します。コンテンツビューのバージョン ID を追加して、**--latest** オプションを使用する必要があります。複合コンテンツビューに、複数のコンポーネントのコンテンツビューを追加するには、追加する全コンテンツビューに対して、この手順を繰り返し実行します。

```
# hammer content-view component add \
--component-content-view-id Content_View_Version_ID \
--latest \
--composite-content-view "Example_Composite_Content_View"
```

4. 複合コンテンツビューを公開します。

```
# hammer content-view publish \
--name "Example_Composite_Content_View" \
--description "Initial version of Composite Content View" \
```

```
--organization "My_Organization"
```

5. 複合コンテンツビューを全環境にプロモートします。

```
# hammer content-view version promote \
--content-view "Example_Composite_Content_View" \
--version 1 \
--to-lifecycle-environment "Development" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Example_Composite_Content_View" \
--version 1 \
--to-lifecycle-environment "Testing" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Example_Composite_Content_View" \
--version 1 \
--to-lifecycle-environment "Production" \
--organization "My_Organization"
```

8.8. コンテンツフィルターの概要

コンテンツビューでは、フィルターを使って特定の RPM コンテンツを追加したり制限したりします。フィルターを使用しないと、選択したリポジトリーからのすべてのコンテンツが含まれてしまいます。

コンテンツフィルターには、次の 2 種類があります。

表8.1 フィルタータイプ

フィルタータイプ	説明
包含	コンテンツなしの状態から開始し、選択したリポジトリーから追加するコンテンツを選択します。このフィルターを使用して、複数のコンテンツアイテムを組み合わせます。
除外	選択したリポジトリーからのコンテンツがすべてある状態から開始し、除外するコンテンツを選択します。リポジトリーのほとんどのコンテンツを使用するものの、ブラックリスト化されたパッケージなど、特定のパッケージは除外する場合にこのフィルターを使用します。このフィルターでは、選択したコンテンツ以外、リポジトリーにあるコンテンツすべてを使用します。

包含と除外のフィルターの組み合わせ

包含と除外のフィルターの組み合わせを使用してコンテンツビューを公開すると、最初に包含フィルターが適用され、次に除外フィルターが適用されます。この場合には、先に、含めるコンテンツを選択してから、このサブセットから除外するコンテンツを選択することになります。

コンテンツタイプ

また、フィルターの対象となるコンテンツには以下の 4 つのタイプがあります。

表8.2 コンテンツタイプ

コンテンツタイプ	説明
パッケージ	名前とバージョン番号に基づいてパッケージをフィルタリングします。
パッケージグループ	パッケージグループ別に、パッケージをフィルタリングします。パッケージグループの一覧は、コンテンツビューに追加されたリポジトリー別になっています。
エラータ - ID 別	フィルターに追加する特定のエラータを選択します。エラーター覧は、コンテンツビューに追加されたリポジトリー別になっています。
エラータ - 日付およびタイプ別	フィルターに追加する発行済みまたは更新済みの日付範囲およびエラータタイプ (バグ修正、機能強化、またはセキュリティー) を選択します。

8.9. パッケージの依存関係の解決

Satellite では、パッケージの依存関係解決機能を使用して、コンテンツビューに含まれるパッケージの依存関係が、コンテンツビューの公開プロセスの一部として、依存するリポジトリーに追加されているようにします。

任意のコンテンツビューのパッケージ依存関係を解決するか、デフォルト設定を変更して、新規コンテンツビューすべてのパッケージ依存関係の解決を有効化または無効化できます。

パッケージの依存関係を解決すると、コンテンツビューのプロモートにかなり時間がかかる可能性がある点に注意してください。パッケージの依存関係解決機能では、コンテンツビューから独立してシステムにインストールされたパッケージを考慮されず、リポジトリー間の依存関係も解決されません。

パッケージの依存関係の解決とフィルター

フィルターは、フィルター内に記載されているパッケージの依存関係を解決するものではありません。必要な依存関係の判定には、テストが必要になる場合があります。

必要なパッケージを除外するフィルターを追加しており、かつコンテンツビューで依存関係の解決が有効になっている場合には、Satellite はフィルターで作成したルールを無視して、パッケージの依存関係解決を優先します。

セキュリティーの目的でコンテンツフィルターを作成している場合に、Satellite は、パッケージの依存関係を解決するためにセキュリティーが低いパッケージを追加する可能性があります。

手順

デフォルトでパッケージの依存関係を解決するには、次の手順を実行します。

1. Satellite Web UI で、**管理** > **設定** に移動して、**コンテンツ** タブをクリックします。
2. **コンテンツビューの依存関係をデフォルトで解決する** という項目を探し、**はい** を選択します。

任意の依存関係解決に関してデフォルトレベルを設定することもできます。必要なパッケージが存在しない場合には、依存関係を解決するパッケージを追加することも可能です。または、リポジトリーにパッケージが存在する場合でも依存関係を解決するための最新のパッケージを追加できます。

依存関係解決のデフォルトレベルを設定するには、次の手順を実行します。

1. Satellite Web UI で、**管理** > **設定** に移動して、**コンテンツ** タブをクリックします。
2. **コンテンツビューの依存関係解決アルゴリズム** という項目を探し、以下のオプションのいずれかを選択します。
   - リポジトリーに存在しない場合にのみ、依存関係を解決するパッケージを追加する場合には、**Conservative** を選択します。
   - リポジトリーにパッケージが存在するかどうかにかかわらず、依存関係を解決するパッケージを追加するには、**Greedy** を選択します。

8.10. コンテンツフィルターの例

カスタムコンテンツフィルターをビルドするには、以下のいずれかの例の手順に従います。

例 1

ベースの Red Hat Enterprise Linux パッケージでリポジトリーを作成します。このフィルターでは、Red Hat Enterprise Linux リポジトリーがコンテンツビューに追加されている必要があります。

フィルター:

- **包含タイプ:** 組み込み
- **コンテンツタイプ:** パッケージグループ
- **フィルター: Base** パッケージグループのみを選択します。

例 2

セキュリティー更新を除く、特定日以降の全エラータを除外するリポジトリーを作成します。これは、重要なセキュリティー更新は即座に適用する必要があるが、その他のシステム更新は定期的に実行する場合に便利です。このフィルターでは、Red Hat Enterprise Linux リポジトリーをコンテンツビューに追加しておく必要があります。

フィルター:

- **包含タイプ:** 除外
- **コンテンツタイプ:** エラータ - 日付およびタイプ別
- **フィルター: バグ修正**と**機能強化**のエラータタイプのみを選択し、**セキュリティー**の選択は解除します。**日付タイプ**を**更新日**に設定します。エラータを制限する日付を**開始日**に設定します。**終了日**は空白にして、セキュリティー以外の新たなエラータにフィルターが適用されないようにします。

例 3

例 1 と例 2 の組み合わせで、オペレーティングシステムパッケージのみが必要で、最近のバグ修正と機能強化エラータを除外します。この場合には、同一のコンテンツビューにアタッチされた 2 つのフィルターが必要です。コンテンツビューは組み込みフィルターを最初に適用してから、除外フィルターを適用します。

フィルター 1:

- **包含タイプ:** 組み込み
- **コンテンツタイプ:** パッケージグループ
- **フィルター:** Base パッケージグループのみを選択します。

フィルター 2:

- **包含タイプ:** 除外
- **コンテンツタイプ:** エラータ - 日付およびタイプ別
- **フィルター:** **バグ修正**と**機能強化**のエラータタイプのみを選択し、**セキュリティー**の選択は解除します。**日付タイプ**を**更新日**に設定します。エラータを制限する日付を**開始日**に設定します。**終了日**は空白にして、セキュリティー以外の新たなエラータにフィルターが適用されないようにします。

コンテンツフィルターの機能例については、How do content filters work in Satellite 6 を参照してください。

8.11. コンテンツフィルターの作成

以下の手順を使用して、コンテンツフィルターを作成します。フィルターをビルドする方法の例については、「コンテンツフィルターの例」を参照してください。

手順

コンテンツフィルターを作成するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **コンテンツビュー** に移動して、コンテンツビューを選択します。
2. **Yum コンテンツ** > **フィルター**に移動し、**新規フィルター**をクリックします。
3. **名前** フィールドに、フィルターの名前を入力します。
4. **コンテンツタイプ** の一覧から、絞り込むコンテンツタイプを選択します。新しいフィルターのコンテンツタイプに選択した内容に応じて、異なるオプションが表示されます。
5. **包含タイプ** リストから、**包含** または **除外** を選択します。
6. **説明** フィールドに、フィルターの説明を入力し、**保存** をクリックします。
7. **コンテンツタイプ** に入力した内容に応じて、必要なフィルターを作成するルールを追加します。
8. **影響のあるリポジトリー** タブをクリックして、どのリポジトリーがこのフィルターを使用するかを選択します。
9. **新規バージョンの公開** をクリックして、フィルタリングされたリポジトリーを公開します。**説明** フィールドに変更の説明を入力し、**保存** をクリックします。

このコンテンツビューを全環境にプロモートできます。

CLI をご利用の場合

1. フィルターをコンテンツビューに追加します。**--inclusion false** オプションを使用して、フィルターを除外フィルターに設定します。

```
# hammer content-view filter create \
--name "Errata Filter" \
--type erratum --content-view "Example_Content_View" \
--description "My latest filter" \
--inclusion false \
--organization "My_Organization"
```

2. フィルターにルールを追加します。

```
# hammer content-view filter rule create \
--content-view "Example_Content_View" \
--content-view-filter "Errata Filter" \
--start-date "YYYY-MM-DD" \
--types enhancement,bugfix \
--date-type updated \
--organization "My_Organization"
```

3. コンテンツビューを公開します。

```
# hammer content-view publish \
--name "Example_Content_View" \
--description "Adding errata filter" \
--organization "My_Organization"
```

4. ビューを各環境にプロモートします。

```
# hammer content-view version promote \
--content-view "Example_Content_View" \
--version 1 \
--to-lifecycle-environment "Development" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Example_Content_View" \
--version 1 \
--to-lifecycle-environment "Testing" \
--organization "My_Organization"
# hammer content-view version promote \
--content-view "Example_Content_View" \
--version 1 \
--to-lifecycle-environment "Production" \
--organization "My_Organization"
```

第9章 SATELLITE SERVER 間でのコンテンツ同期

Red Hat Satellite 6.6 は Inter-Satellite Synchronization (ISS) を使用して、コンテンツの Satellite Server 間、または Satellite Server の組織間を同期します。

次のシナリオで ISS を使用することができます。

- オンラインの Satellite Server とオフラインの Satellite Server の両方があり、オンラインのサーバーからオフラインのサーバーへコンテンツをコピーする場合。たとえば、セキュリティーなどの理由で、管理インフラストラクチャーを完全に分離する必要がある場合などです。
- Satellite Server の一部のコンテンツのみを他の Satellite Server にコピーする場合。たとえば、IT 部門が検証するコンテンツビューが Satellite Server であり、このコンテンツビューのコンテンツを他の Satellite Server へコピーする場合などです。
- ある組織のコンテンツビューのクローンを Satellite Server の別の組織に作成する場合。

ISS は、Satellite Server から Capsule Server へのコンテンツの同期には使用できません。Capsule Server はネイティブで同期をサポートします。詳細は、『**Red Hat Satellite 6 のプランニング**』の「Capsule Server の概要」を参照してください。

9.1. コンテンツビューバージョンのエクスポート

コンテンツビューのバージョンを、Satellite Server からアーカイブファイルにエクスポートして、このアーカイブファイルを使用し、別の Satellite Server か、別の Satellite Server の組織に同じコンテンツビューバージョンを作成します。Satellite では、複合コンテンツビューはエクスポートされません。エクスポートしたアーカイブファイルには、以下のデータが含まれます。

- コンテンツビューバージョンのメタデータが含まれる JSON ファイル
- コンテンツビューバージョンに組み込まれる全リポジトリーを含むアーカイブファイル

Satellite Server は、コンテンツビューのバージョンに追加された RPM およびキックスタートファイルのみをエクスポートします。Satellite では、以下の内容はエクスポートされません。

- Puppet コンテンツ
- Docker コンテンツ
- OSTree コンテンツ
- パッケージフィルターなど、コンテンツビューの定義およびメタデータ

hammer content-view version export コマンドの変更

hammer content-view version export および **hammer content-view version import** の新しいコマンドは、以前の Satellite のバージョンのコマンドと機能の仕方が違います。以前の機能は、**hammer content-view version export-legacy** コマンドで利用でき、新しい機能には存在しない機能が、以前の機能には含まれています。

1. オンラインの Satellite Server からオフラインの Satellite Server に直接パッチを適用できます。**hammer content-view version export-legacy** は、CDN の構造をエクスポートするので、Red Hat カスタマーポータルからの DVD ISO を使用する必要はありません。
2. yum 以外のコンテンツを含むコンテンツビューをエクスポートする場合は、**hammer content-view version export-legacy** を使用すると、yum 以外のコンテンツをスキップしてコンテンツ

ビューをエクスポートしますが、**hammer content-view version export** は、yum 以外のリポジトリーを削除するようにプロンプトを表示して失敗します。

以前の機能に関する詳細情報は、『Satellite 6.4 コンテンツ管理ガイド』の「Satellite Server 間でのコンテンツ同期」を参照してください。

前提条件

コンテンツビューをエクスポートするには、エクスポートする Satellite Server が、以下の条件を満たしていることを確認します。

- エクスポートディレクトリーに、エクスポートに対応できる空き容量があることを確認する。
- **/var/lib/pulp/** ディレクトリーに、エクスポートプロセス中に作成された一時ファイルに、エクスポートされるリポジトリーのサイズと同じ空き容量があることを確認する。
- **/var/cache/pulp** ディレクトリーに、エクスポートプロセス中に作成される一時ファイルに、エクスポートされるリポジトリーのサイズの 2 倍の空き容量があることを確認する。
- エクスポートするコンテンツビュー内の全リポジトリーでダウンロードポリシーを**即時** に設定していることを確認する。詳細については、「ダウンロードポリシーの概要」を参照してください。
- リポジトリー設定ページで、インポートするリポジトリーの **Mirror on Sync** チェックボックスの選択が解除されていることを確認する。
- エクスポートする製品が、必要な日付に同期されることを確認する。

コンテンツビューバージョンをエクスポートする方法

1. コンテンツビューを一覧表示して、エクスポートするコンテンツビューバージョンの ID を特定します。

```
# hammer content-view version list \
--organization "Default Organization"
```

2. コンテンツビューのバージョンをエクスポートします。**--export-dir** オプションを使用してエクスポートを保存するディレクトリー、**--id** オプションを使用してエクスポートするコンテンツビューバージョン の ID を指定します。**pulp_export_destination** 設定は、この手順では動作しません。

```
# hammer content-view version export --export-dir export_directory \
--id content_view_version_ID
```

3. エクスポートしたコンテンツビューバージョンが含まれるアーカイブが、エクスポートディレクトリーにあることを確認します。

```
# ls export_directory
export-1.tar
```

9.2. コンテンツビューバージョンのインポート

hammer content-view version export コマンドが出力するアーカイブを使用して、エクスポートしたコンテンツビューバージョンとコンテンツが同じコンテンツビューバージョンを作成できます。コンテンツビューバージョンのエクスポートの詳細は、「コンテンツビューバージョンのエクスポート」 を参

照してください。

コンテンツビューバージョンをインポートすると、メジャーバージョン番号、マイナーバージョン番号が同じで、同じパッケージ、エラータを含む同じリポジトリーが含まれます。エクスポートされたアーカイブ内の **json** ファイルの **major** および **minor** 設定を変更して、バージョン番号をカスタマイズできます。

前提条件

コンテンツビューをインポートするには、インポート先の Satellite Server が、以下の条件を満たしていることを確認します。

- オフライン環境でコンテンツビューを Satellite にインポートする場合は、コンテンツをローカル CDN サーバーと同期するように Satellite を設定してから、エクスポートする CV が含まれるコンテンツを同期する必要がある。詳細は、*付録B コンテンツをローカル CDN サーバーと同期するための Satellite の設定* を参照してください。
- エクスポートするコンテンツビュー内の全リポジトリーでダウンロードポリシーを**即時** に設定していることを確認する。詳細については、「ダウンロードポリシーの概要」を参照してください。
- リポジトリー設定ページで、インポートするリポジトリーの **Mirror on Sync** チェックボックスの選択が解除されていることを確認する。

手順

1. インポート先の Satellite Server の **/var/lib/pulp/katello-export** ディレクトリーに、エクスポートしたコンテンツビューバージョンが含まれるアーカイブファイルをコピーします。
2. インポートする Satellite Server で、エクスポートしたコンテンツビューと同じ名前とラベルで、コンテンツビューを作成します。詳細情報は、Hammer CLI を使用したコンテンツビューの作成を参照してください。
3. エクスポートしたコンテンツビューバージョンの製品に含まれるリポジトリーを有効にしてください。詳細は、「Red Hat リポジトリーの有効化」 を参照してください。
4. Satellite Web UI で **コンテンツ** > **製品** に移動し、**Yum コンテンツ** タブで、エクスポートしたコンテンツビューに含まれる **Yum** コンテンツと同じものを追加します。
5. BZ#1745081 が解決されるまで、**/var/lib/pulp/katello-export** ディレクトリーに移動します。

```
# cd /var/lib/pulp/katello-export
```

6. コンテンツビューバージョンを Satellite Server にインポートするには、次のコマンドを入力します。

```
# hammer content-view version import \
--export-tar /var/lib/pulp/katello-export/exported_CV_archive.tar \
--organization-id Your_Organization_ID
```

BZ#1745081 が解決されるまで、完全なパス **/var/lib/pulp/katello-export/** を入力する必要があります。相対パスは機能しません。

7. 組織のコンテンツビューを一覧表示して、コンテンツビューバージョンのインポートが成功したことを確認します。

```
# hammer content-view version list --organization "Your_Organization"
```

第10章 アクティベーションキーの管理

アクティベーションキーは、システム登録とサブスクリプションのアタッチを自動化する方法を提供します。複数のキーを作成して、異なる環境とコンテンツビューに関連付けることができます。たとえば、Red Hat Enterprise Linux ワークステーション用のサブスクリプションで基本のアクティベーションキーを作成し、これを特定の環境のコンテンツビューに関連付けることができます。

コンテンツホストの登録時にアクティベーションキーを使用して、プロセスのスピードアップ、単純化、一貫性の向上を図ることができます。

アクティベーションキーを使用して、コンテンツホストの次のプロパティーを定義できます。

- 関連付けるサブスクリプションおよびサブスクリプションのアタッチの動作。
- 利用可能な製品およびリポジトリー。
- ライフサイクル環境およびコンテンツビュー。
- ホストコレクションのメンバーシップ。

アクティベーションキーは、ホストが登録されている場合にのみ使用できます。アクティベーションキーに変更が加えられた場合には、それ以降、改訂されたアクティベーションキーで登録されるホストにのみ、変更が適用され、既存のホストには加えられません。

ホストの作成と登録の間のコンテンツビューの競合

Satellite は、ホストのプロビジョニング時に、プロビジョニングテンプレートと、ホストグループまたはホスト設定で設定したコンテンツホストビューからの他のコンテンツを使用します。ホストの登録時には、アクティベーションキーからのコンテンツビューは、ホストグループまたはホストの設定をベースにしたもとのコンテンツビューを上書きし、Satellite は、ホストの再構築時など、その後のタスクすべてにアクティベーションキーからのコンテンツビューを使用します。

ホストの再構築時に、使用するコンテンツビューを、ホストグループやホスト設定ではなく、アクティベーションキーで設定するようにします。

複数のコンテンツホストでの同じアクティベーションキーの使用

サブスクリプションが十分にある場合には、同じアクティベーションキーを複数のコンテンツホストに適用できます。ただし、アクティベーションキーはコンテンツホストの初期設定のみを行います。コンテンツホストを組織に登録した後は、組織のコンテンツをコンテンツホストに手動でアタッチできます。

コンテンツホストでの複数のアクティベーションキーの使用

コンテンツホストは、ホストの設定を定義する g ために組み合わせて使用する複数のアクティベーションキーと関連付けることができます。設定の競合が発生した場合には、最後の指定下アクティベーションキーが優先されます。以下のようにホストグループのパラメーターを設定して優先順位を指定できます。

```
$ hammer hostgroup set-parameter \
--name kt_activation_keys \
--value name_of_first_key, name_of_second_key,... \
--hostgroup hostgroup_name
```

10.1. アクティベーションキーの作成

アクティベーションキーを使用して、登録時にホストにアタッチするサブスクリプションの特定のセットを定義できます。アクティベーションキーに追加するサブスクリプションは、関連するコンテンツビュー内で利用可能である必要があります。

サブスクリプションマネージャーは、以下の要因に応じたさまざまな方法で、サブスクリプションをアタッチします。

- サブスクリプションがアクティベーションキーに関連付けられているか?
- 自動アタッチオプションは有効になっているか?

上記の要因をもとに、アクティベーションキーを使用してサブスクライブするシナリオを 3 つ想定できます。

1. サブスクリプションが指定されていないアクティベーションキー。
   サブスクリプションの指定なしで、自動アタッチが有効な場合に、アクティベーションキーを使用するホストは、アクティベーションキーと関連するコンテンツビューが提供する最適なサブスクリプションを検索します。これは、**subscription-manager --auto-attach** コマンドを実行する場合と類似しています。

2. 自動アタッチ用にカスタムサブスクリプションプールを指定するアクティベーションキー。
   サブスクリプションが指定されていて自動アタッチが有効な場合に、アクティアクティベーションキーを使用するホストは、アクティベーションキーで指定された一覧から最適なサブスクリプションを選択します。

3. サブスクリプションセットが指定されたアクティベーションキー。
   サブスクリプションが指定されており、自動アタッチが無効な場合に、アクティベーションキーを使用するホストは、アクティベーションキーに指定されたすべてのサブスクリプションに関連付けられます。

カスタム製品

カスタム製品 (通常は Red Hat が提供しないコンテンツを含む製品) がアクティベーションキーに割り当てられている場合には、この製品は、自動アタッチの設定の有無にかかわらず、登録されたコンテンツホストに対して常に有効になります。

手順

アクティベーションキーを作成するには、以下の手順を行います。

1. Satellite Web UI で **コンテンツ** > **アクティベーションキー** に移動して、**アクティベーションキーの作成** をクリックします。

2. **名前** フィールドに、アクティベーションキーの名前を入力します。

3. 制限を設定しない場合は、**Unlimited hosts** チェックボックスの選択を解除して、**Limit** フィールドに、アクティベーションキーを使って登録できるシステムの最大数を入力します。アクティベーションキーを使って登録するホストに制限を設けない場合は、**Unlimited Hosts** チェックボックスが選択されていることを確認します。

4. **説明** フィールドに、アクティベーションキーの説明を入力します。

5. **環境** 一覧から、使用する環境を選択します。

6. **コンテンツビュー** リストから、使用するコンテンツビューを選択します。このアクティベーションキーを使用してホストを登録する場合は、**katello-agent** をインストールする必要があるため、コンテンツビューに Satellite Tools リポジトリーが含まれている必要があります。

7. **保存** をクリックし、アクティベーションキーウィンドウに新しいアクティベーションキーが表示されたら、編集する名前をクリックします。

CLI をご利用の場合

1. アクティベーションキーを作成します。

```
# hammer activation-key create \
--name "My_Activation_Key" \
--unlimited-hosts \
--description "Example Stack in the Development Environment" \
--lifecycle-environment "Development" \
--content-view "Stack" \
--organization "My_Organization"
```

2. サブスクリプション ID 一覧を取得します。

```
# hammer subscription list --organization "My_Organization"
```

3. Red Hat Enterprise Linux サブスクリプション UUID をアクティベーションキーにアタッチします。

```
# hammer activation-key add-subscription \
--name "My_Activation_Key" \
--subscription-id ff808181533518d50152354246e901aa \
--organization "My_Organization"
```

4. アクティベーションキーに関連付けられている製品コンテンツを一覧表示します。

```
# hammer activation-key product-content \
--name "My_Activation_Key" \
--organization "My_Organization"
```

5. Red Hat Satellite Tools 6.6 リポジトリーのデフォルトの自動有効化ステータスを上書きします。デフォルトでは無効になっています。有効にするには、以下のコマンドを実行します。

```
# hammer activation-key content-override \
--name "My_Activation_Key" \
--content-label rhel-7-server-satellite-tools-6.6-rpms \
--value 1 \
--organization "My_Organization"
```

10.2. アクティベーションキーを使用して関連するサブスクリプションの更新

Web UI を使用するか、Hammer コマンドラインツールを使用して、アクティベーションキーに関連付けられたサブスクリプションを変更できます。

アクティベーションキーへの変更は、変更後にプロビジョニングしたマシンにのみ適用されます。既存のコンテンツホストでサブスクリプションを更新する方法は 「コンテンツホストのサブスクリプションの一括更新」 を参照してください。

手順

アクティベーションキーに関連付けられたサブスクリプションを更新するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **アクティベーションキー** に移動し、アクティベーションキーの名前をクリックします。
2. **サブスクリプション** タブをクリックします。
3. サブスクリプションを削除するには **一覧/削除** を選択してから、削除するサブスクリプションの左側にあるチェックボックスを選択し、**選択した項目を追加** をクリックします。
4. サブスクリプションを追加するには、**追加** を選択してから、追加するサブスクリプションの左側のチェックボックスを選択し、**選択した項目を追加** をクリックします。
5. **リポジトリーセット** タブをクリックし、リポジトリーのステータス設定を確認します。
6. リポジトリーを有効または無効にするには、リポジトリーに対してチェックボックスを選択し、**アクションの選択** リストを使用してステータスを変更します。
7. **詳細** タブをクリックし、このアクティベーションキーにコンテンツビューを選択し、**保存** をクリックします。

CLI をご利用の場合

1. 現在アクティベーションキーが含まれているサブスクリプションを一覧表示します。

```
# hammer activation-key subscriptions \
--name My_Activation_Key \
--organization "My_Organization"
```

2. アクティベーションキーから必要なサブスクリプションを削除します。

```
# hammer activation-key remove-subscription \
--name "My_Activation_Key" \
--subscription-id ff808181533518d50152354246e901aa \
--organization "My_Organization"
```

--subscription-id オプションで、UUID またはサブスクリプションの ID のいずれかを使用できます。

3. 新しいサブスクリプションをアクティベーションキーにアタッチします。

```
# hammer activation-key add-subscription \
--name "My_Activation_Key" \
--subscription-id ff808181533518d50152354246e901aa \
--organization "My_Organization"
```

--subscription-id オプションで、UUID またはサブスクリプションの ID のいずれかを使用できます。

4. アクティベーションキーに関連付けられている製品コンテンツを一覧表示します。

```
# hammer activation-key product-content \
--name "My_Activation_Key" \
--organization "My_Organization"
```

5. 必要なリポジトリーのデフォルトの自動有効化ステータスを上書きします。

```
# hammer activation-key content-override \
--name "My_Activation_Key" \
--content-label content_label \
--value 1 \
--organization "My_Organization"
```

有効化する場合は、**--value** オプションに **1** を、無効化する場合は **0** を入力します。

10.3. アクティベーションキーを使用したホストの登録

アクティベーションキーを使用して、以下のタスクを完了できます。

- Red Hat Satellite 6 を使用したプロビジョニング中に新規ホストを登録する。Red Hat Satellite 6 のキックスタートプロビジョニングテンプレートには、ホストの作成時に定義されるアクティベーションキーを使用してホストを登録するコマンドが含まれています。
- 既存の Red Hat Enterprise Linux ホストを登録する。Red Hat サブスクリプションマネージャーが登録に Satellite Server を使用するように設定し、**subscription-manager register** コマンドの実行時にアクティベーションキーを指定します。

手順

アクティベーションキーを使用して、既存の Red Hat Enterprise Linux 7 ホストを、Satellite Server に登録するには、以下の手順を行います。

1. Satellite Server 用のコンシューマー RPM をダウンロードします。これは、ホストの Web サーバーの **pub** ディレクトリーに配置されています。たとえば、ホスト名が **satellite.example.com** の Satellite Server の場合は、登録するホストで以下のコマンドを実行します。

```
# rpm -Uvh http://satellite.example.com/pub/katello-ca-consumer-latest.noarch.rpm
```

この RPM は Satellite Server 上のリポジトリーにアクセスするために必要な証明書をインストールし、Red Hat サブスクリプションマネージャーがサーバーの URL を使用するように設定します。

2. ホストで、アクティベーションキーを使用して Satellite にホストを登録するには、以下のコマンドを実行します。

```
# subscription-manager register --activationkey="My_Activation_Key" \
--org="My_Organization"
```

3. 組織のホストの一覧を表示するには、Satellite Server で、以下のコマンドを実行します。

```
# hammer host list --organization "My_Organization"
```

4. Satellite Server にホストを登録したら、ホストに **katello-agent** パッケージをインストールして、ホストから Satellite Server にレポートできるようにします。

```
# yum install katello-agent
```

このパッケージは、Red Hat Satellite Tools 6.6 リポジトリーに含まれます。

複数のアクティベーションキー

コンテンツホストの登録時に複数のアクティベーションキーを使用できます。特定のサブスクリプションセット用にアクティベーションキーを作成し、コンテンツホストの要件に合わせて、これらのアクティベーションキーを組み合わせることができます。たとえば、以下のコマンドは VDC と OpenShift の両方のサブスクリプションでコンテンツホストを組織に登録します。

```
# subscription-manager register --org="My_Organization" \
--activationkey="ak-VDC,ak-OpenShift"
```

競合の設定

アクティベーションキーの設定で競合が生じた場合は、右端のキーが優先されます。

- 競合する設定: **サービスレベル**、**リリースバージョン**、**環境**、**コンテンツビュー**、および **製品コンテンツ**。
- 競合しない設定と、ホストがその統合を取得: **サブスクリプション** および **ホストコレクション**。
- キーそのものの動作に影響を与えるが、ホストの設定には影響を与えない設定: **コンテンツホストの制限** および **自動アタッチ**。

10.4. 自動アタッチの有効化

アクティベーションキーで自動アタッチを有効にし、キーに関連付けられているサブスクリプションがある場合は、サブスクリプション管理サービスが、現在インストールされている製品、アーキテクチャー、およびサービスレベルなどの設定に基づいて、最適な関連サブスクリプションを選択してアタッチします。

自動アタッチを有効にして、キーに関連付けられたサブスクリプションを持たないことができます。このタイプのキーは、仮想マシンが物理サブスクリプションを消費するのではなく、ハイパーバイザーからホストベースのサブスクリプションを継承する場合に、仮想マシンを登録するために一般的に使用されます。詳細は、『Red Hat Satellite での仮想マシンサブスクリプションの設定』を参照してください。

自動アタッチはデフォルトで有効になっています。アクティベーションキーに関連付けられているすべてのサブスクリプションを強制的にアタッチする場合は、このオプションを無効にします。

手順

自動アタッチを有効にするには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **アクティベーションキー** に移動します。
2. 編集するアクティベーションキーの名前をクリックします。
3. **サブスクリプション** タブをクリックします。
4. **自動アタッチ**の隣にある編集アイコンをクリックします。
5. チェックボックスにチェックを入れて自動アタッチを有効にするか、チェックを外して無効にします。
6. **保存**をクリックします。

CLI をご利用の場合

アクティベーションキーで自動アタッチを有効にするには、以下を実行します。

```
# hammer activation-key update --name "My_Activation_Key" \
--organization "My_Organization" --auto-attach true
```

10.5. サービスレベルの設定

アクティベーションキーで作成した新規ホストのデフォルトのサービスレベルを定義するように、アクティベーションキーを設定できます。デフォルトのサービスレベルを設定すると、ホストにアタッチするのに適したサブスクリプションのみが選択されます。たとえば、アクティベーションキーのデフォルトのサービスレベルが Premium に設定されている場合には、Premium サービスレベルのサブスクリプションのみが、登録時にホストにアタッチされます。

手順

サービスレベルを設定するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **アクティベーションキー** に移動します。
2. 編集するアクティベーションキーの名前をクリックします。
3. **サービスレベル**の隣にある編集アイコンをクリックします。
4. リストから必要なサービスレベルを選択します。このリストには、アクティベーションキーで利用できるサービスレベルだけが含まれます。
5. **保存**をクリックします。

CLI をご利用の場合

アクティベーションキーでデフォルトのサービスレベルを Premium に設定するには、以下を実行します。

```
# hammer activation-key update --name "My_Activation_Key" \
--organization "My_Organization" --service-level premium
```

第11章 エラータの管理

Red Hat では、品質管理およびリリースプロセスの一部として、お客様に Red Hat RPM の公式リリースの更新を提供しています。Red Hat では、更新を説明するアドバイザリーと共に、関連パッケージのグループを**エラータ**にコンパイルします。アドバイザリーには以下の 3 種類があります (重要度の高い順)。

セキュリティーアドバイザリー

パッケージで見つかったセキュリティー問題の修正を説明。セキュリティー問題の重大度のレベルは、低、中、重要、重大に分かれています。

バグ修正アドバイザリー

パッケージのバグ修正を説明。

製品の機能強化アドバイザリー

パッケージに追加された機能強化および新機能を説明。

Red Hat Satellite 6 は、リポジトリーを Red Hat の Content Delivery Network (CDN) と同期する際にこれらのエラータ情報をインポートします。Red Hat Satellite 6 ではエラータを検証しフィルタリングするためのツールも提供しており、更新の管理が正確にできます。このようにして、関連のある更新を選択し、コンテンツビューから選択したコンテンツホストに伝達することができます。

エラータには、それらに含まれる最も重要なアドバイザリータイプに応じてラベルが付けられます。そのため、**製品の機能強化アドバイザリー** というラベルが付けられたエラータには機能強化の更新のみが含まれ、**バグ修正アドバイザリー** エラータにはバグ修正と機能強化の両方が含まれ、**セキュリティーアドバイザリー** にはこれら 3 つのタイプが含まれる場合があります。

Red Hat Satellite では、エラータと利用可能なコンテンツホストとの関係を表す 2 つのキーワードがあります。

適用可能

1 つ以上のコンテンツホストに適用されるエラータ。これは、コンテンツホストに存在するパッケージを更新することを意味します。これらのエラータはコンテンツホストに適用されますが、状態が **インストール可能** に変わるまでは、エラータをインストールする準備はできていません。インストール可能なエラータは自動的に適用されます。

インストール可能

1 つ以上のコンテンツホストに適用され、コンテンツホストにインストールできるエラータ。インストール可能なエラータは、ライフサイクル環境および関連するコンテンツビューからコンテンツホストで利用できますが、まだインストールされていません。

本章では、エラータの管理方法と 1 つのホストまたは複数のホストへの適用方法を説明します。

11.1. 利用可能なエラータの検出

以下の手順では、利用可能なエラータを表示し、フィルタリングする方法や、選択したアドバイザリーのメタデータを表示する方法を説明します。

1. **コンテンツ** > **エラータ** に移動して、利用可能なエラータの一覧を表示します。
2. ページ上部のフィルターツールを使用して、表示されるエラータの数を制限します。
   - 調べるリポジトリーをリストから選択します。デフォルトでは **すべてのリポジトリー** が選択されます。

- **適用可能** チェックボックスがデフォルトで選択され、選択されたリポジトリーに適用可能なエラータだけが表示されます。**インストール可能** チェックボックスを選択すると、インストール可能のマークが付いたエラータのみが表示されます。

- エラータの表を検索するには、以下の形式で **検索** フィールドにクエリーを入力します。

  ```
  parameter operator value
  ```

  検索に使用できるパラメーターの一覧は、表11.1「エラータ検索で利用できるパラメーター」 を参照してください。適用可能な演算子の一覧は、『**Red Hat Satellite の管理**』の「詳細な検索に対してサポートされる演算子」を参照してください。入力時に自動サジェスト機能が利用できます。**and** 演算子と **or** 演算子を使用してクエリーを組み合わせることもできます。たとえば、**kernel** パッケージに関するセキュリティーアドバイザリーのみを表示するには、以下を入力します。

  ```
  type = security and package_name = kernel
  ```

  Enter を押して検索を開始します。

3. 調べるエラータの **Errata ID** をクリックします。

   - **詳細** タブには、更新されたパッケージの説明や、更新によって提供される重要な修正および機能強化が記載されています。

   - **コンテンツホスト** タブでは、「複数ホストへのエラータの適用」で説明したように、選択したコンテンツホストにエラータを適用できます。

   - **リポジトリー** タブには、エラータが含まれているリポジトリーの一覧が表示されます。リポジトリーはフィルターを使用して環境やコンテンツビューで絞り込むことができ、リポジトリー名で検索できます。

CLI をご利用の場合

- 全組織で利用可能なエラータを表示するには、以下のコマンドを実行します。

  ```
  # hammer erratum list
  ```

- 特定のエラータの詳細を表示するには、以下のコマンドを実行します。

  ```
  # hammer erratum info --id erratum_ID
  ```

- **--search** オプションを指定してクエリーを入力し、エラータを検索します。たとえば、選択した製品に適用可能なエラータで、指定したバグが含まれるものを順番に表示し、セキュリティーエラータが一番上に表示されるようにするには、以下のコマンドを入力します。

  ```
  # hammer erratum list \
  --product-id 7 \
  --search "bug = 1213000 or bug = 1207972" \
  --errata-restrict-applicable 1 \
  --order "type desc"
  ```

表11.1 エラータ検索で利用できるパラメーター

パラメーター	説明	例
bug	Bugzilla 番号での検索。	**bug = 1172165**
cve	CVE 番号での検索。	**cve = CVE-2015-0235**
id	エラータ ID での検索。自動サジェストシステムにより、入力時に利用可能な ID の一覧が表示されます。	**id = RHBA-2014:2004**
issued	発行日による検索。正確な日付 (「Feb16,2015」など) を指定したり、キーワード (「Yesterday」、「1 hour ago」など) を使用したりできます。時間の範囲は、「<」演算子と「>」演算子を使用して指定できます。	**issued < "Jan 12,2015"**
package	完全なパッケージビルド名による検索。自動サジェストシステムにより、入力時に利用可能なパッケージの一覧が表示されます。	**package = glib2-2.22.5-6.el6.i686**
package_name	パッケージ名による検索。自動サジェストシステムにより、入力時に利用可能なパッケージの一覧が表示されます。	**package_name = glib2**
severity	セキュリティー更新によって修正される問題の重大度による検索。**Critical**、**Important**、または **Moderate** を指定します。	**severity = Critical**
title	アドバイザリーのタイトルによる検索。	**title ~ openssl**
type	アドバイザリーのタイプによる検索。**security**、**bugfix**、または **enhancement** を指定します。	**type = bugfix**
updated	最新更新日による検索。**issued** パラメーターの同じ形式を使用できます。	**updated = "6 days ago"**

11.2. エラータ通知のサブスクライブ

Satellite ユーザー向けにメール通知を設定することができます。ユーザーには、リポジトリーの同期後に、適用可能かつインストール可能なエラータのまとめ、通知が、コンテンツビュープロモーションで送信されます。詳しい情報は、『**Red Hat Satellite の管理**』ガイドの「電子メール通知の設定」を参照してください。

11.3. リポジトリー依存関係の解決の制限

Satellite 6 には、リポジトリー依存関係の解決が必要な問題が複数あり、これは既知の問題です。詳細は、BZ#1508169、BZ#1640420、BZ#1508169、BZ#1629462 を参照してください。Satellite でコンテンツビューの増分更新を使用すると、リポジトリー依存関係の問題がいくつか解決しますが、リポジトリーレベルの依存関係の解決で問題が残る場合があります。

新しい依存関係でリポジトリーの更新が利用できるようになると、Satellite は、既存のリポジトリーパッケージで利用可能な古いバージョンがある場合でも、依存関係を解決するために、パッケージの最新バージョンを取得します。これにより、パッケージのインストール時に依存関係の解決が必要な問題がさらに発生することがあります。

シナリオ例

クライアント上のリポジトリーには、依存関係 **example_repository-libs-1.0** のパッケージ **example_repository-1.0** があります。リポジトリーには、別のパッケージ **example_tools-1.0** もあります。

セキュリティーエラータは、パッケージ **example_tools-1.1** で利用できるようになります。**example_tools-1.1** パッケージは、依存関係として **example_repository-libs-1.1** パッケージが必要です。

コンテンツビューを増分更新すると、**example_tools-1.1**、**example_tools-1.0**、**example_repository-libs-1.1** がリポジトリーに含まれます。リポジトリーには、**example_repository-1.0** と **example_repository-libs-1.0** のパッケージもあります。コンテンツビューの増分更新で、パッケージ **example_repository-1.1** が追加されなかったことに注意してください。yum を使用してこれらのすべてのパッケージをインストールできるため、潜在的な問題は検出されません。ただし、クライアントが **example_tools-1.1** パッケージをインストールすると、**example_repository-libs-1.0** と **example_repository-libs-1.1** の両方をインストールできないため、依存関係の解決が必要な問題が発生します。

現在、この問題の回避策はありません。RPM の基本セットから、適用されるエラータまでのメジャー Y リリースの期間が長いほど、依存関係の解決の問題が発生する可能性が高くなります。

11.4. エラータ用のコンテンツビューフィルターの作成

コンテンツフィルターを使用して、エラータを制限できます。以下のようなフィルターを使用します。

- **ID**: 結果として表示されるリポジトリーに含めることができるように、特定のエラータを選択します。
- **日付の範囲**: 日付の範囲を定義して、その範囲内にリリースされたエラータを追加します。
- **タイプ**: バグ修正、機能強化、セキュリティーなどのエラータのタイプを選択して追加します。

特定日より後のエラータを除外するコンテンツフィルターを作成します。これにより、アプリケーションライフサイクルの実稼働システムがある時点まで最新に保たれたことになります。その後にこのフィルターの開始日を変更し、テスト環境に新たなエラータを導入します。こうすることで、新パッケージにアプリケーションライフサイクルとの互換性があるかどうかをテストできます。

前提条件

- 必要なエラータを含むリポジトリーを指定してコンテンツビューを作成しておく。詳細情報は、「コンテンツビューの作成」を参照してください。

手順

1. Satellite web UI で、**コンテンツ** > **コンテンツビュー** に移動して、エラータ適用に使用するコンテンツビューを選択します。
2. **Yum コンテンツ** > **フィルター**に移動し、**新規フィルター**をクリックします。
3. **名前** フィールドで、**Errata Filter** を入力します。
4. **コンテンツタイプ** リストから **エラータ - 日付およびタイプ** を選択します。
5. **含有タイプ** リストから **除外** を選択してください。
6. **説明** フィールドに **Exclude errata items from YYYY-MM-DD** を入力します。
7. **保存**をクリックします。
8. **エラータタイプ** には、除外するエラータタイプのチェックボックスを選択します。たとえば、特定の日付以降の機能拡張やバグ修正エラータを除外し、セキュリティーエラータすべてを含めるには、**機能拡張** および **バグ修正** のチェックボックスを選択し、**セキュリティー** チェックボックスの選択を解除します。
9. **日付タイプ** では、2 つのチェックボックスからいずれかを選択します。
   - エラータの発行日については **発行日** を選択します。
   - エラータの最終更新日については **更新日** を選択します。
10. **開始日** を選択して、すべてのエラータを除外するか、選択した日付以降のエラータを除外します。
11. **終了日** フィールドは空白にしておきます。
12. **保存**をクリックします。
13. **新規バージョンの公開** をクリックして、表示されているリポジトリーを公開します。
14. **説明** フィールドに **Adding errata filter** と入力します。
15. **保存**をクリックします。
    コンテンツビューが公開されると、**コンテンツ** コラムのパッケージとエラータの数が公開前のリポジトリーと比べて少なくなります。これは、前年のセキュリティー以外のエラータがフィルターにより正常に除外されたためです。
16. **バージョン** タブをクリックします。
17. 公開バージョンの右側にある **プロモート** をクリックします。
18. コンテンツビューのプロモート先の環境を選択します。
19. **説明** フィールドに、プロモートの説明を入力します。
20. **バージョンのプロモート** をクリックして、必要とされる環境全体に、このコンテンツビューバージョンをプロモートします。

CLI をご利用の場合

1. エラータのフィルターを作成します。

```
# hammer content-view filter create --name "Filter Name" \
--description "Exclude errata items from the YYYY-MM-DD" \
--content-view "CV Name" --organization "Default Organization" \
--type "erratum"
```

2. フィルタールールを作成して、指定の **開始日** 以降のエラータすべてを除外します。

```
# hammer content-view filter rule create --start-date "YYYY-MM-DD" \
--content-view "CV Name" --content-view-filter="Filter Name" \
--organization "Default Organization" --types=security,enhancement,bugfix
```

3. コンテンツビューを公開します。

```
# hammer content-view publish --name "CV Name" \
--organization "Default Organization"
```

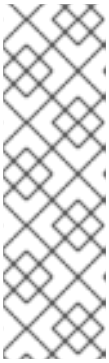
4. コンテンツビューをライフサイクル環境にプロモートし、そこに含まれるエラータをそのライフサイクル環境で利用できるようにします。

```
# hammer content-view version promote \
--content-view "CV Name" \
--organization "Default Organization" \
--to-lifecycle-environment "Lifecycle Environment Name"
```

11.5. 増分コンテンツビューへのエラータの追加

エラータが利用できるがインストールできない場合には、増分のコンテンツビューバージョンを作成して、エラータをコンテンツホストに追加できます。たとえば、コンテンツビューがバージョン 1.0 の場合は、コンテンツビューバージョン 1.1 になり、公開時に、コンテンツビューバージョン 2.0 になります。

1. Satellite Web UI で、**コンテンツ** > **エラータ** に移動します。
2. **エラータ** の一覧から、適用するエラータの名前をクリックします。
3. エラータを適用するコンテンツホストを選択し、**ホストに適用** をクリックします。これにより、コンテンツビューの増分更新が作成されます。
4. エラータをコンテンツホストに適用する場合は、**公開直後にコンテンツホストにエラータを適用する** チェックボックスを選択します。



注記

BZ#1459807 が解決されるまで、Capsule Server に登録されているホストに、インストール不可のエラータを適用する場合には、**公開直後にコンテンツホストにエラータを適用する** のチェックボックスは選択しないでください。

代わりに、**確認** をクリックした後に、エラータのコンテンツビューのプロモートと、Capsule の同期タスクが完了するまで待機します。次に、エラータが **Installable** とマークされるので、この手順を使用して、もう一度エラータを適用可能です。

5. **確認** をクリックして、エラータを適用します。

CLI をご利用の場合

1. エラータと対応する ID を一覧表示します。

```
# hammer erratum list
```

2. 異なるコンテンツビューバージョンと対応する ID を一覧表示します。

```
# hammer content-view version list
```

3. コンテンツビューバージョンに単一のエラータを適用します。コンマ区切りのリストとして、さらに ID を追加できます。

```
# hammer content-view version incremental-update \
--content-view-version-id 319 --errata-ids 34068b
```

11.6. ホストへのエラータの適用

以下の手順を使用して、エラータをレビューし、ホストに適用します。

前提条件

- Red Hat から利用可能な最新のエラータと、Red Hat Satellite リポジトリーを同期しておく。詳細は、「Red Hat リポジトリーの同期」 を参照してください。
- Satellite Server の環境およびコンテンツビューにホストを登録しておく。詳細は、『**ホストの管理**』ガイドの「ホストの登録」を参照してください。
- RHEL 7 ホストに、**katello-agent** パッケージを必ずインストールしてください。詳細は、『**ホストの管理**』ガイドの「Katello エージェントのインストール」を参照してください。

Red Hat Enterprise Linux 8 の場合

RHEL 8 ホストにエラータを適用するには、Satellite Server でリモート実行ジョブを実行するか、ホストを更新できます。リモート実行ジョブの実行の詳細は、『**ホストの管理**ガイドの「ホストでのジョブの実行」を参照してください。

RHEL 8 ホストにエラータを適用するには、以下の手順を行います。

1. Satellite で、ホストのすべてのエラータを一覧表示します。

```
# hammer host errata list \
--host client.example.com
```

2. エラータが含まれるモジュールのストリームを検索します。

```
# hammer erratum info --id ERRATUM_ID
```

3. ホストで、モジュールストリームを更新します。

```
# yum update Module_Stream_Name
```

Red Hat Enterprise Linux 7 の場合

RHEL 7 ホストにエラータを適用するには、以下の手順を行います。

1. Satellite Web UI で、**ホスト** > **コンテンツホスト** に移動し、エラータを適用するホストを選択します。
2. **エラータ** タブに移動してエラータのリストを表示します。
3. 適用するエラータを選択し、**Apply Selected (選択した項目を適用)** をクリックします。確認画面で、**適用** をクリックします。
4. 選択したエラータに関連付けられた全パッケージを更新するタスクが完了したら、**詳細** タブをクリックして更新済みのパッケージを表示します。

CLI をご利用の場合

RHEL 7 ホストにエラータを適用するには、以下の手順を行います。

1. ホストのすべてのエラータを一覧表示します。

```
# hammer host errata list \
--host client.example.com
```

2. ホストに最新のエラータを適用します。エラータ ID を使用して適用するエラータを特定します。

```
# hammer host errata apply --host "Host Name" \
--errata-ids ERRATUM_ID1,ERRATUM_ID2...
```

11.7. 複数ホストへのエラータの適用

以下の手順を使用して、エラータをレビューし、複数の RHEL 7 ホストに適用します。

前提条件

- Red Hat から利用可能な最新のエラータと、Red Hat Satellite リポジトリーを同期しておく。詳細は、「Red Hat リポジトリーの同期」 を参照してください。
- Satellite Server の環境およびコンテンツビューにホストを登録します。詳細は、『**ホストの管理**』ガイドの「ホストの登録」を参照してください。
- ホストに **katello-agent** パッケージをインストールしてください。詳細は、『**ホストの管理**』ガイドの「Katello エージェントのインストール」を参照してください。

手順

1. **コンテンツ** > **エラータ** に移動します。
2. 適用するエラータ名をクリックします。
3. **コンテンツホスト** タブをクリックします。
4. エラータの適用先のホストを選択し、**ホストへの適用** をクリックします。
5. **確認** をクリックします。

CLI をご利用の場合

CLI には Web UI と同じツールがあるわけではありませんが、同様の手順を CLI コマンドで使用することができます。

1. インストール可能な全エラータを表示します。

```
# hammer erratum list \
--errata-restrict-installable true \
--organization "Default Organization"
```

2. 使用するエラータを選択し、このエラータを適用可能なホストを一覧表示します。

```
# hammer host list \
--search "applicable_errata = ERRATUM_ID" \
--organization "Default Organization"
```

3. エラータを 1 つのホストに適用します。

```
# hammer host errata apply \
--host client.example.com \
--organization "Default Organization" \
--errata-ids ERRATUM_ID1,ERRATUM_ID2...
```

4. 次の Bash スクリプトを使用して、このエラータが利用可能な各ホストにエラータを適用します。

```
for HOST in hammer --csv --csv-separator "|" host list --search "applicable_errata = ERRATUM_ID" --organization "Default Organization" | tail -n+2 | awk -F "|" '{ print $2 }' ;
do
  echo "== Applying to $HOST ==" ; hammer host errata apply --host $HOST --errata-ids ERRATUM_ID1,ERRATUM_ID2 ;
done
```

このコマンドは、**erratum_IDs** を適用できるホストをすべて特定し、このエラータを各ホストに適用します。

5. エラータが正しく適用されたことを確認するには、以下のコマンドの出力で適切なタスクを検索します。

```
# hammer task list
```

6. 選択したタスクの状態を表示します。

```
# hammer task progress --id task_ID
```

11.8. ホストコレクションへのエラータの適用

ホストコレクションに選択したエラータを適用するには、以下のコマンドを入力します。

```
# hammer host-collection erratum install \
--errata "erratum_ID1,erratum_ID2,..." \
--name "host_collection_name"\
```

--organization "**Your_Organization**"

第12章 OSTREE コンテンツの管理

OSTree は、起動可能で、変更しない、バージョン付きファイルシステムツリーを管理するツールです。ビルドシステムにカスタムの OSTree コンテンツを使用して、OSTree リポジトリーを静的 HTTP にエクスポートできます。Red Hat Enterprise Linux Atomic Server は、RPM ファイルから作成された OSTree コンテンツを使用して、オペレーティングシステムを最新の状態に保ちます。

Red Hat Satellite 6 を使用して、OSTree リポジトリーの OSTree ブランチを同期してブランチを管理することができます。

Satellite Server 6.6 では、OSTree 管理ツールがデフォルトで有効になっています。ツールを有効化する必要がある場合には、以下のコマンドを入力します。

```
# satellite-installer --katello-enable-ostree=true
```

12.1. 同期する RED HAT OSTREE コンテンツの選択

Red Hat の CDN で OSTree コンテンツを選択して同期します。

手順

OSTree コンテンツを検索して同期するには以下の手順を実行します。

1. Satellite Web UI で、**コンテンツ** > **Red Hat リポジトリー**に移動します。
2. リストから **OSTree** のコンテンツタイプを選択します。
3. 利用可能なリポジトリーペインで、**Red Hat Enterprise Linux Atomic Host**の製品グループの **Red Hat Enterprise Linux Atomic Host Trees**セットなど、使用する OSTree リポジトリーセットを特定します。
4. **有効化** アイコンをクリックして、使用するリポジトリーを有効化します。
5. **コンテンツ** > **製品** に移動して、**Red Hat Enterprise Linux Atomic Host** など、使用する製品をクリックします。
6. このリポジトリーのアップストリームの同期ポリシーを選択します。デフォルトでは、Satellite は、最新の OSTree ブランチのみを同期します。
   a. 同期するリポジトリーをクリックします。
   b. **アップストリームの同期ポリシー** メニューから、次のポリシーの 1 つを選択して、このリポジトリーの OSTree ブランチを同期します。
      - **最新のみ**: 最新の OSTree ブランチのみを同期します。
      - **すべての履歴**: すべての OSTree ブランチを同期します。
      - **カスタム**: 指定した数の OSTree ブランチを同期します。下のフィールドに必要な数を入力します。
   c. **保存**をクリックします。
7. **アクションの選択** メニューから **同期開始** を選択します。

同期の状態の表示方法

- Satellite Web UI で、**コンテンツ** > **同期の状態** に移動し、たとえば、**Red Hat Enterprise Linux Atomic Host** を展開します。

CLI をご利用の場合

1. Red Hat Enterprise Linux Server 製品の **ostree** リポジトリーを検索します。

```
# hammer repository-set list \
--product "Red Hat Enterprise Linux Atomic Host" \
--organization "My_Organization" | grep "ostree"
```

2. Red Hat Enterprise Linux Atomic Host や使用する製品の **ostree** リポジトリーを有効にします。

```
# hammer repository-set enable \
--product "Red Hat Enterprise Linux Atomic Host" \
--name "Red Hat Enterprise Linux Atomic Host (Trees)" \
--organization "My_Organization"
```

3. 製品のリポジトリーの場所を特定し、同期します。

```
# hammer repository list \
--product "Red Hat Enterprise Linux Atomic Host" \
--organization "My_Organization"
# hammer repository synchronize \
--name "Red Hat Enterprise Linux Atomic Host Trees" \
--product "Red Hat Enterprise Linux Atomic Host" \
--organization "My_Organization"
```

12.2. カスタム OSTREE コンテンツのインポート

Red Hat の CDN から OSTree コンテンツをインポートするほかに、他のソースからコンテンツをインポートすることもできます。これには、公開済みの HTTP の場所が必要になります。

手順

カスタムの OSTree コンテンツをインポートするには、以下の手順を実行します。

1. Satellite Web UI で **コンテンツ** > **製品** に移動して、**製品の作成** をクリックします。
2. **名前** フィールドには、OSTree コンテンツの名前を入力します。これにより、自動的に **ラベル** フィールドにデータが投入されます。
3. オプション: **GPG キー** フィールドには、製品全体の GPG キーを入力します。
4. **同期プラン** メニューから製品に関連付ける同期プランを選択します。
5. **説明** フィールドには、製品の説明を入力し、**保存** をクリックします。
6. 製品の作成が完了したら、**リポジトリーの作成** をクリックします。
7. **名前** フィールドには、リポジトリーの名前を入力します。これにより、自動的に **ラベル** フィールドにデータが投入されます。
8. **タイプ** の一覧から **ostree** を選択します。

9. **URL** フィールドには、ソースとして使用するレジストリーの URL を入力します。たとえば **http://www.example.com/rpm-ostree/** と入力します。
10. **アップストリームの同期ポリシー** メニューから、次のポリシーの 1 つを選択して、このリポジトリーの OSTree ブランチを同期します。
    - **最新のみ**: 最新の OSTree ブランチのみを同期します。
    - **すべての履歴**: すべての OSTree ブランチを同期します。
    - **カスタム**: 指定した数の OSTree ブランチを同期します。下のフィールドに必要な数を入力します。
11. **保存**をクリックします。
12. リポジトリー作成が完了したら、新規リポジトリーを選択して、**同期開始** をクリックして同期プロセスを開始します。

同期の状態の表示方法

- Satellite Web UI で、**コンテンツ** > **同期の状態** に移動し、表示するエントリーを展開します。

CLI をご利用の場合

1. カスタムの **OSTree Content** 製品を作成します。

```
# hammer product create \
--name "Custom OSTree Content" \
--sync-plan "Example_Plan" \
--description "OSTree Content" \
--organization "My_Organization"
```

2. OSTree 用のリポジトリーを作成します。

```
# hammer repository create \
--name "Custom OSTree" \
--content-type "ostree" \
--url "http://www.example.com/rpm-ostree/" \
--product "OSTree Content" \
--organization "My_Organization"
```

3. リポジトリーを同期します。

```
# hammer repository synchronize \
--name "Custom OStree" \
--product "OSTree Content" \
--organization "My_Organization"
```

12.3. コンテンツビューによる OSTREE コンテンツの管理

コンテンツビューを使用して、アプリケーションライフサイクルで OSTree ブランチを管理します。このプロセスでは、RPM および Puppet モジュールが使用するものと同じ公開とプロモーションのメソッドを使用します。

手順

OSTree のコンテンツビューを作成して、リポジトリーを追加するには、以下の手順を実行します。

1. Satellite Web UI で **コンテンツ** > **コンテンツビュー**に移動して、**新規ビューの作成**をクリックします。
2. **名前** フィールドには、プレーンテキスト形式のビューの名前を入力します。これにより、自動的に **ラベル** フィールドにデータが投入されます。
3. **説明** フィールドには、OSTree コンテンツビューの説明を加えます。
4. 複合コンテンツビューを使用する場合には、**複合ビュー** のチェックボックスを選択します。
5. **保存**をクリックします。
6. **OSTree コンテンツ** タブに移動して、**追加** をクリックします。
7. 使用するコンテンツの OSTree リポジトリーを選択します。**リポジトリーの追加** をクリックして、OSTree コンテンツをこのリポジトリーからコンテンツビューに追加します。
8. **バージョン** に移動して、**新規バージョンの公開** をクリックします。
9. **説明** フィールドには、バージョンの説明を入力し、**保存** をクリックします。

プロモート をクリックして、アプリケーションライフサイクルの環境でこのコンテンツビューをプロモートすることもできます。

CLI をご利用の場合

1. リポジトリー ID の一覧を取得します。

```
# hammer repository list --organization "_My_Organization_"
```

2. コンテンツビューを作成して、リポジトリーを追加します。

```
# hammer content-view create \
--name "OSTree" \
--description "OSTree for Red Hat Enterprise Linux Atomic Host" \
--repository-ids 5 \
--organization "My_Organization"
```

3. ビューを公開します。

```
# hammer content-view publish \
--name "OSTree" \
--description "Example Content View for the OSTree" \
--organization "My_Organization"
```

第13章 コンテナーイメージの管理

Red Hat Satellite 6 では、さまざまなソースからコンテナーイメージをインポートして、コンテンツビューを使用して外部コンテナーに分散できます。

コンテナーに関する情報は、「**Red Hat Enterprise Linux Atomic Host 7**の Getting Started with Containers」を参照してください。

13.1. コンテナーイメージのインポート

Red Hat レジストリーまたは他のイメージレジストリーからコンテナーイメージリポジトリーをインポートできます。

以下の手順では、リポジトリー検出を使用して、コンテナーイメージを検索し、リポジトリーとしてインポートします。手動での製品とリポジトリーの作成方法は、6章 *カスタムコンテンツのインポート*を参照してください。

手順

コンテナーイメージリポジトリーをインポートして、製品を作成するか、製品と関連付けるには、以下の手順を実行します。

1. Satellite Web UI で **コンテンツ** > **製品** に移動して、**リポジトリーの作成** をクリックします。
2. **リポジトリータイプ** リストから **コンテナーイメージ** を選択します。
3. **検出するレジストリー** フィールドには、イメージのインポート元となるレジストリーの URL を入力します。
4. **レジストリーのユーザー名** フィールドには、コンテナーのイメージレジストリーのユーザ名に対応する名前を入力します。
5. **レジストリーのパスワード** フィールドには、入力したユーザ名に対応するパスワードを入力します。
6. **レジストリー検索パラメーター** フィールドには、検索の絞り込みに使用する検索条件を入力して、**検出** をクリックします。
7. オプション: **検出されたリポジトリー** リストをさらに絞り込むには、**フィルター** フィールドに、使用する追加の検索条件を入力します。
8. **検出されたリポジトリー** リストからインポートするリポジトリーを選択して、**選択項目の作成** クリックします。
9. オプション: **製品** リストから製品を作成するには、**新しい製品** を選択します。
10. **名前** フィールドに製品名を入力します。
11. オプション: **リポジトリー名** と **リポジトリーラベル** のコラムで、リポジトリー名とラベルを編集できます。
12. **リポジトリー作成の実行** をクリックします。
13. リポジトリーの作成が完了したら、各新規リポジトリーをクリックして詳細情報を確認できます。

14. オプション: リポジトリーにインポートするコンテンツをフィルタリングするには、リポジトリーをクリックして、**同期タグの制限** に移動します。これをクリックし、Satellite への同期コンテンツを制限するタグを編集または追加します。

15. **コンテンツ** > **製品** に移動し、製品名を選択します。

16. 新規リポジトリーを選択し、**同期開始** をクリックして同期プロセスを開始します。

同期の進捗状況を表示するには、**コンテンツ** > **同期の状態** に移動して、リポジトリーツリーを展開します。

同期が完了したら、**コンテナーイメージのマニフェスト** をクリックして利用可能なマニフェストを一覧表示します。また、必要のなくなったマニフェストは、このリストから削除できます。

CLI をご利用の場合

1. カスタムの **Red Hat Container Catalog** 製品を作成します。

```
# hammer product create \
--name "Red Hat Container Catalog" \
--sync-plan "Example Plan" \
--description "Red Hat Container Catalog content" \
--organization "My_Organization"
```

2. コンテナーイメージ用のリポジトリーを作成します。

```
# hammer repository create \
--name "RHEL7" \
--content-type "docker" \
--url "http://registry.access.redhat.com/" \
--docker-upstream-name "rhel7" \
--product "Red Hat Container Catalog" \
--organization "My_Organization"
```

3. リポジトリーを同期します。

```
# hammer repository synchronize \
--name "RHEL7" \
--product "Red Hat Container Catalog" \
--organization "My_Organization"
```

13.2. コンテナー名のパターンの管理

Satellite を使用してコンテナーの作成や管理を行う場合には、コンテナーはコンテンツビューのバージョン間や異なるステージの Satellite ライフサイクル環境間を移動するので、コンテナー名はステージごとに変化します。たとえば、アップストリームのリポジトリーから、**ssh** 名を使用してコンテナーイメージを同期する場合には、そのイメージを Satellite 製品と組織に追加してコンテンツビューの一部として公開する時に、コンテナーイメージは **my_organization_production-custom_spin-my_product-custom_ssh** という名前になる可能性があります。これが原因で、コンテナーイメージをプルする時に問題が発生する可能性があります。理由は、コンテナーレジストリーに、コンテナー名のインスタンスが 1 つしか含まれていない可能性があるためです。Satellite の命名規則の問題を回避するには、デフォルト名を上書きするようにレジストリー名のパターンを設定して、コンテナー名が後で使用するときに明確になるようにします。

制限事項

レジストリー名のパターンを使用してコンテナーの命名規則を管理する場合には、レジストリーの命名パターンが理由でグローバルー意名を生成する必要があるので、命名時に競合の問題が発生する可能性があります。以下の例を示します。

- **repository.docker_upstream_name** のレジストリー名パターンを設定した場合に、**Production** ライフサイクルと同じリポジトリー名のコンテナーコンテンツが含まれるコンテンツビューを公開またはプロモートできません。
- **lifecycle_environment.name** のレジストリー名パターンを設定した場合には、同じ名前を指定して 2 つ目のコンテナーリポジトリーを作成することができません。

コンテナーに対してレジストリーの命名パターンを定義する場合には、注意を払って進めて行く必要があります。

手順

レジストリー名パターンで、コンテナーの命名を管理するには、以下の手順を実行します。

1. Satellite Web UI で、**コンテンツ** > **ライフサイクル環境** に移動して、ライフサイクル環境を作成するか、編集するライフサイクル環境を選択します。
2. **コンテナーイメージのレジストリー** エリアで、**レジストリー名のパターン** の右側にある編集アイコンをクリックします。
3. 変数一覧と例を使用して、必要とされるレジストリー名のパターンを判断します。
4. **レジストリー名のパターン** フィールドに、使用するレジストリー名のパターンを入力します。たとえば、**repository.docker_upstream_name** を使用するには、以下を入力します。

```
<%= repository.docker_upstream_name %>
```

5. **保存**をクリックします。

13.3. コンテナーレジストリーの認証管理

デフォルトでは、Satellite のコンテナーイメージにアクセスするには認証が必要です。

ライフサイクル環境の Satellite イメージレジストリーに含まれるコンテナーイメージにアクセスするのにユーザーの認証をするかどうかを指定できます。たとえば、認証要件なしに **Production** ライフサイクルからコンテナーイメージにアクセスできるようにしたり、認証済みのユーザーだけに **Development** および **QA** 環境へのアクセスを制限したりすることもできます。

手順

Satellite からコンテナーイメージにアクセスするための認証設定を管理するには、以下の手順を実行します。

1. Satellite Web UI で、**コンテンツ** > **ライフサイクル環境** に移動して、認証管理するライフサイクル環境を選択します。
2. このライフサイクル環境に認証なしでアクセスできるようにするには、**非認証のプル** のチェックボックスを選択します。認証なしのアクセスを制限するには、**非認証のプル** チェックボックスのチェックを外します。
3. **保存**をクリックします。

第14章 ISO イメージの管理

Red Hat Satellite 6 を使用して、Red Hat のコンテンツ配信ネットワーク (CDN) または他のソースからの ISO イメージを保存できます。仮想マシンイメージなどの他のファイルをアップロードしたり、リポジトリーに公開したりすることも可能です。

14.1. RED HAT からの ISO イメージのインポート

Red Hat CDN では、特定製品の ISO イメージを提供しています。このコンテンツをインポートする手順は、RPM コンテンツのリポジトリーを有効にする手順と似ています。

手順

Red Hat ISO イメージをインポートするには、以下の手順を実行します。

1. Satellite Web UI で、**コンテンツ** > **Red Hat リポジトリー**に移動します。
2. **検索** フィールドで、**Red Hat Enterprise Linux 7 Server (ISOs)** などのイメージ名を入力します。
3. 利用可能なリポジトリーウィンドウで、**Red Hat Enterprise Linux 7 Server (ISOs)**を展開します。
4. **x86_64 7.2** エントリーでは、**有効化** アイコンをクリックして、対象のイメージのリポジトリーを有効にします。
5. **コンテンツ** > **製品** に移動して、**Red Hat Enterprise Linux Server**をクリックします。
6. Red Hat Enterprise Linux Server ウィンドウの **Repositories** タブをクリックして、**Red Hat Enterprise Linux 7 Server ISOs x86_64 7.2** をクリックします。
7. Red Hat Enterprise Linux 7 Server ISOs x86_64 7.2 ウィンドウの右上で、**アクションの選択** をクリックして、**同期開始** を選択します。

同期の状態の表示方法

- Web UI で、**コンテンツ** > **同期の状態** に移動し、たとえば、**Red Hat Enterprise Linux Server** を展開します。

CLI をご利用の場合

1. **file** リポジトリーの Red Hat Enterprise Linux Server 製品を特定します。

```
# hammer repository-set list \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization" | grep "file"
```

2. Red Hat Enterprise Linux 7.2 Server ISO の **file** リポジトリーを有効にします。

```
# hammer repository-set enable \
--product "Red Hat Enterprise Linux Server" \
--name "Red Hat Enterprise Linux 7 Server (ISOs)" \
--releasever 7.2 \
--basearch x86_64 \
--organization "My_Organization"
```

3. 製品のリポジトリーの場所を特定し、同期します。

```
# hammer repository list \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
# hammer repository synchronize \
--name "Red Hat Enterprise Linux 7 Server ISOs x86_64 7.2" \
--product "Red Hat Enterprise Linux Server" \
--organization "My_Organization"
```

14.2. 個別の ISO イメージとファイルのインポート

以下の手順を使用して、ISO コンテンツとその他のファイルを Satellite Server に手動でインポートします。カスタムファイルをインポートするには、Web UI または Hammer CLI を使用して、以下の手順を実行してください。ただし、アップロードするファイルのサイズが 15 MB よりも大きい場合は、Hammer CLI を使用してリポジトリーにアップロードする必要があります。

1. カスタム製品を作成します。
2. ファイルのリポジトリーを製品に追加します。
3. ファイルをリポジトリーにアップロードします。

手順

カスタムの ISO イメージをインポートするには、以下の手順を実行します。

1. Satellite Web UI で **コンテンツ** > **製品** に移動して、製品ウィンドウで **製品の作成** をクリックします。
2. **名前** フィールドで製品を識別するための名前を入力します。この名前が **ラベル** フィールドに投入されます。
3. **GPG キー** フィールドには、製品の GPG キーを入力します。
4. **同期プラン** リストから製品の同期プランを選択します。
5. **説明**フィールドには、製品の説明を入力します。
6. **保存**をクリックします。
7. 製品ウィンドウで、新製品をクリックし、**リポジトリーの作成** をクリックします。
8. **名前** フィールドには、リポジトリーの名前を入力します。これにより、自動的に **ラベル** フィールドにデータが投入されます。
9. **タイプ** の一覧から **ファイル** を選択します。
10. **アップストリーム URL** フィールドに、ソースとして使用するレジストリーの URL を入力します。**アップストリームのユーザー名** と **アップストリームのパスワード** フィールドには対応するユーザー名とパスワードを追加します。
11. **保存**をクリックします。
12. 新しいリポジトリーをクリックします。
13. **ファイルのアップロード** に移動し、**参照** をクリックします。

14. **.iso** ファイルを選択して **アップロード** をクリックします。

CLI をご利用の場合

1. カスタム製品を作成します。

```
# hammer product create \
--name "My_ISOs" \
--sync-plan "Example Plan" \
--description "My_Product" \
--organization "My_Organization"
```

2. リポジトリーを作成します。

```
# hammer repository create \
--name "My_ISOs" \
--content-type "file" \
--product "My_Product" \
--organization "My_Organization"
```

3. ISO ファイルをリポジトリーにアップロードします。

```
# hammer repository upload-content \
--path ~/bootdisk.iso \
--name "My_ISOs" \
--organization "My_Organization"
```

第15章 カスタムファイルタイプコンテンツの管理

Satellite で、SSH キーおよびソースコードファイル、仮想マシンイメージや ISO ファイルなどの大容量ファイルの管理と配布が必要になる場合があります。これには、Red Hat Satellite のカスタム製品にカスタムファイルタイプのリポジトリーを追加します。こうすることで、製品に任意のファイルを組み込む一般的な方法が提供されます。

リポジトリーにファイルをアップロードし、アップストリームの Satellite Server からファイルを同期できます。ファイルをカスタムのファイルタイプリポジトリーに追加すると、特定のバージョンをコンテンツビューに追加して、さまざまな Capsule Server でファイルのリポジトリーを利用可能にするなど、通常の Satellite 管理機能を使用できます。クライアントは、**curl -O** で、HTTP または HTTPS からファイルをダウンロードする必要があります。

Satellite Server のファイルタイプリポジトリーはカスタム製品に対してのみ作成できますが、ファイルタイプリポジトリーは柔軟に作成できます。Satellite がインストールされているシステム、またはリモートの HTTP サーバーのディレクトリーに個別のファイルタイプリポジトリーを作成して、そのディレクトリーのコンテンツを同期できます。この方法は、Satellite リポジトリーに追加するファイルが複数ある場合に便利です。

15.1. RED HAT SATELLITE でカスタムファイルタイプリポジトリーの作成

カスタムファイルタイプリポジトリーを作成する手順は、リポジトリー作成時に **ファイル** タイプを選択する以外は、カスタムコンテンツの作成手順と同じです。製品を作成してから、カスタムリポジトリーを追加する必要があります。

手順

カスタム製品を作成するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、**製品の作成** をクリックして、以下の詳細情報を入力します。
2. **名前** フィールドで、製品の名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容に基づいて **ラベル** フィールドに自動的に入力されます。
3. オプション: **GPG キー** の一覧から、製品の GPG キーを選択します。
4. オプション: **同期プラン** リストから製品の同期プランを選択します。
5. **説明** フィールドに、製品の説明を入力し、**保存** をクリックします。

カスタム製品のリポジトリーを作成するには、以下の手順を行います。

1. 製品ウィンドウで、リポジトリーを作成する製品の名前を選択します。
2. **リポジトリー** タブをクリックして、**新規リポジトリー** をクリックします。
3. **名前** フィールドに、リポジトリーの名前を入力します。Red Hat Satellite 6 では、名前に基づいて、**ラベル** フィールドに値が自動的に入力されます。
4. **タイプ** の一覧から **ファイル** を選択します。
5. **アップストリーム URL** フィールドに、ソースとして使用するアップストリームリポジトリーの URL を入力します。
6. アップストリームのリポジトリーの SSL 証明書が信頼できる認証機関 (CA) によって署名されていることを確認する場合、**SSL の検証** チェックボックスを選択します。

7. **アップストリームのユーザー名** フィールドに、認証に必要な場合にアップストリームリポジトリーのユーザー名を入力します。リポジトリーに認証が必要ない場合はこのフィールドを空にします。

8. **アップストリームのパスワード** フィールドに、アップストリームリポジトリーのパスワードを入力します。リポジトリーに認証が必要ない場合はこのフィールドを空にします。

9. **保存**をクリックします。

CLI をご利用の場合

1. カスタム製品の作成

```
# hammer product create \
--name "My File Product" \
--sync-plan "Example Plan" \
--description "My files" \
--organization "My_Organization"
```

表15.1 hammer product create コマンドのオプションパラメーター

オプション	説明
--gpg-key gpg_key_name	検索するキー名
--gpg-key-id gpg_key_id	GPG キー数値 ID
--sync-plan sync_plan_name	検索する同期プラン名
--sync-plan-id sync_plan_id	同期プランの数値 ID

2. ファイルタイプリポジトリーの作成

```
# hammer repository create \
--name "My Files" \
--content-type "file" \
--product "My File Product" \
--organization "My_Organization"
```

表15.2 hammer repository create コマンドのオプションパラメーター

オプション	説明
--checksum-type sha_version	リポジトリーのチェックサムです。現在、'sha1' および 'sha256' がサポートされています。
--download-policy policy_name	yum リポジトリーのダウンロードポリシーです ('immediate'、'on_demand'、または 'background')。
--gpg-key gpg_key_name	検索するキー名

オプション	説明
--gpg-key-id gpg_key_id	GPG キー数値 ID
--mirror-on-sync boolean	同期する場合に、このリポジトリーをソースからミラーリングし、古い RPM を削除する必要がありますか? **true** または **false**、**yes** または **no**、もしくは **1** または **0** に設定します。
--publish-via-http boolean	HTTP を使用して公開する必要がありますか? **true** または **false**、**yes** または **no**、**1** または **0** に設定します。
--upstream-username repository_username	認証に必要な場合は、アップストリームリポジトリーユーザー
--upstream-password repository_password	アップストリームリポジトリーユーザーのパスワード
--url source_repo_url	ソースリポジトリーの URL
--verify-ssl-on-sync boolean	URL の SSL 証明書が信頼できる CA によって署名されているのを Katello が確認する必要がありますか? **true** または **false**、**yes** または **no**、もしくは **1** または **0** に設定します。

15.2. ローカルディレクトリーにカスタムのファイルタイプリポジトリーの作成

Satellite が **pulp-manifest** コマンドを使用してインストールされているベースシステムで、ファイルのディレクトリーから、カスタムファイルタイプリポジトリーを作成できます。その後、Satellite Server にファイルを同期します。ファイルタイプリポジトリーにファイルを追加すると、他のリポジトリーと同じようにファイルを操作できます。

以下の手順を使用して、Satellite がインストールされているベースシステムのディレクトリーにリポジトリーを設定します。リモートサーバーのディレクトリーにファイルタイプリポジトリーを作成するには、「リモートファイルタイプリポジトリーの作成」を参照してください。

手順

ローカルディレクトリーにファイルタイプリポジトリーを作成するには、以下の手順を行います。

1. サーバーおよび Satellite Tools リポジトリーが有効になっていることを確認します。

```
# subscription-manager repos --enable=rhel-7-server-rpms \
--enable=rhel-7-server-satellite-tools-6.6-rpms
```

2. Pulp マニフェストパッケージをインストールします。

```
# satellite-maintain packages install python-pulp-manifest
```

3. HTTP サーバーのパブリックフォルダーのファイルタイプリポジトリーとして使用するディレクトリーを作成します。

```
# mkdir my_file_repo
```

4. ディレクトリーにファイルを追加して、テストファイルを作成します。

```
# touch my_file_repo/test.txt
```

5. Pulp マニフェストコマンドを入力して、マニフェストを作成します。

```
# pulp-manifest my_file_repo
```

6. マニフェストが作成されたことを確認します。

```
# ls my_file_repo
PULP_MANIFEST test.txt
```

ファイルタイプリポジトリーからのファイルのインポート

ファイルタイプリポジトリーからファイルをローカルディレクトリーにインポートするには、以下の手順を行います。

1. カスタム製品が Satellite Server に存在することを確認します。
2. Satellite Web UI で、**コンテンツ** > **製品**に移動します。
3. 製品の名前を選択します。
4. **リポジトリー** タブをクリックして、**新規リポジトリー** を選択します。
5. **名前** フィールドに、リポジトリーの名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容をもとに、このフィールドに値が自動的に入力されます。
6. **タイプ** リストから、リポジトリーのコンテンツタイプを選択します。
7. **アップストリーム URL** フィールドに、ソースとして使用するリポジトリーを使用したローカルディレクトリーを入力します (**file:///my_file_repo** の形式)。
8. **SSL の検証** チェックボックスを選択してリポジトリーの SSL 証明書をチェックするか、**SSL の検証** チェックボックスの選択を解除します。
9. オプション: **アップストリームのユーザー名** フィールドに、必要なアップストリームユーザー名を入力します。
10. オプション: **アップストリームのパスワード** フィールドに、アップストリームユーザー名のパスワードを入力します。
11. **保存** をクリックして、このリポジトリーエントリーを保存します。

ファイルタイプリポジトリーの更新

ファイルタイプリポジトリーを更新するには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品**に移動します。

2. 製品の名前を選択します。
3. 更新するリポジトリーの名前を選択します。
4. **アクションの選択** メニューから **同期開始** を選択します。
5. リポジトリーを公開した URL を開いて、ファイルを表示します。

15.3. リモートファイルタイプリポジトリーの作成

pulp-manifest を使用して、Satellite Server の外部にあるファイルのディレクトリーから、カスタムファイルタイプリポジトリーを作成します。その後、HTTP または HTTPS 経由で Satellite Server にファイルを同期します。ファイルタイプリポジトリーにファイルを追加すると、他のリポジトリーと同じようにファイルを操作できます。

以下の手順を使用して、リモートサーバーのディレクトリーにリポジトリーを設定します。Satellite Server がインストールされているベースシステムのディレクトリーにファイルタイプリポジトリーを作成するには、「ローカルディレクトリーにカスタムのファイルタイプリポジトリーの作成」を参照してください。

前提条件

リモートファイルタイプリポジトリーを作成する前に、以下の条件が存在することを確認します。

- Red Hat Enterprise Linux 7 サーバーが Satellite または Red Hat CDN に登録されている。
- Red Hat Enterprise Linux Server および Satellite Tools リポジトリーにエンタイトルメントがある。
- HTTP サーバーがインストールされている。Web サーバーの設定方法は『**システム管理者のガイド**』における Red Hat Enterprise Linux 7 での「 Apache HTTP サーバー」を参照してください。

手順

リモートディレクトリーにファイルタイプリポジトリーを作成するには、以下の手順を行います。

1. リモートサーバーで、サーバーおよび Satellite Tools リポジトリーが有効になっていることを確認します。

```
# subscription-manager repos --enable=rhel-7-server-rpms \
--enable=rhel-7-server-satellite-tools-6.6-rpms
```

2. Pulp マニフェストパッケージをインストールします。

```
# yum install python-pulp-manifest
```

3. HTTP サーバーのパブリックフォルダーのファイルタイプリポジトリーとして使用するディレクトリーを作成します。

```
# mkdir /var/www/html/pub/my_file_repo
```

4. ディレクトリーにファイルを追加して、テストファイルを作成します。

```
# touch /var/www/html/pub/my_file_repo/test.txt
```

5. Pulp マニフェストコマンドを入力して、マニフェストを作成します。

```
# pulp-manifest /var/www/html/pub/my_file_repo
```

6. マニフェストが作成されたことを確認します。

```
# ls /var/www/html/pub/my_file_repo
PULP_MANIFEST test.txt
```

リモートファイルタイプリポジトリーからのファイルのインポート

リモートファイルタイプリポジトリーからファイルをインポートするには、以下の手順を行います。

1. カスタム製品が Satellite Server に存在することを確認します。ない場合には、カスタム製品を作成します。詳細については、「Red Hat Satellite でカスタムファイルタイプリポジトリーの作成」を参照してください。
2. Satellite Web UI で、**コンテンツ** > **製品**に移動します。
3. 製品の名前を選択します。
4. **リポジトリー** タブをクリックして、**新規リポジトリー** を選択します。
5. **名前** フィールドに、リポジトリーの名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容をもとに、このフィールドに値が自動的に入力されます。
6. **タイプ** の一覧から **ファイル** を選択します。
7. **アップストリーム URL** フィールドに、ソースとして使用するアップストリームリポジトリーの URL を入力します。
8. アップストリームのリポジトリーの SSL 証明書が信頼できる認証機関 (CA) によって署名されていることを確認する場合、**SSL の検証** チェックボックスを選択します。
9. **アップストリームのユーザー名** フィールドに、認証に必要な場合にアップストリームリポジトリーのユーザー名を入力します。リポジトリーに認証が必要ない場合はこのフィールドを空にします。
10. **アップストリームのパスワード** フィールドに、アップストリームリポジトリーのパスワードを入力します。リポジトリーに認証が必要ない場合はこのフィールドを空にします。
11. **保存**をクリックします。
12. **コンテンツ** > **製品** に移動します。更新するリポジトリーが含まれる製品の名前を選択します。
13. 製品のウィンドウで、更新するリポジトリーの名前を選択します。
14. **アクションの選択** メニューから **同期開始** を選択します。

リポジトリーを公開した URL を開いて、ファイルを表示します。

15.4. RED HAT SATELLITE へのカスタムファイルタイプリポジトリーへのファイルのアップロード

手順

ファイルをカスタムファイルタイプリポジトリーにアップロードするには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品**に移動します。
2. カスタム製品の名前を選択します。
3. ファイルタイプリポジトリーの名前を選択します。
4. **参照** をクリックして、アップロードするファイルを選択します。
5. **アップロード** をクリックして、選択したファイルを Satellite Server にアップロードします。
6. リポジトリーを公開した URL を開いて、ファイルを表示します。

CLI をご利用の場合

```
# hammer repository upload-content \
--name "My Files" \
--organization "My_Organization" \
--path example_file
```

--path オプションは、ファイル、ファイルディレクトリー、またはファイルの glob 表現を示します。glob は、一重引用符または二重引用符でエスケープする必要があります。

15.5. RED HAT SATELLITE のカスタムファイルタイプリポジトリーからホストにファイルのダウンロード

curl -O を実行して、HTTPS (**HTTP での公開** リポジトリーを選択している場合は HTTP でも可能) でクライアントにファイルをダウンロードします。

前提条件

- カスタムファイルタイプリポジトリーがある。詳細は「Red Hat Satellite でカスタムファイルタイプリポジトリーの作成」を参照してください。
- ファイルタイプリポジトリーから、クライアントにダウンロードするファイル名を把握しておく。
- HTTPS を使用するためには、クライアントで以下の証明書を用意しておく。
    1. **katello-server-ca.crt**。詳細については、『**Red Hat Satellite の管理**』ガイドの「Katello ルート CA 証明書のインストール」を参照してください。
    2. 組織のデバッグ証明書。詳細については、「組織のデバッグ証明書の作成」を参照してください。

手順

ファイルをカスタムファイルタイプリポジトリーからホストにダウンロードするには、以下の手順を行います。

1. Satellite Web UI で、**コンテンツ** > **製品**に移動します。
2. カスタム製品の名前を選択します。
3. ファイルタイプリポジトリーの名前を選択します。

4. **HTTP 経由での公開** が有効になっているかどうかを確認します。有効になっていない場合は、HTTPS を使用するための証明書が必要です。

5. リポジトリーを公開した URL をコピーします。

CLI をご利用の場合

1. ファイルタイプリポジトリーを一覧表示します。

```
# hammer repository list --content-type file
---|----------|----------------|--------------|----
ID | NAME     | PRODUCT        | CONTENT TYPE | URL
---|----------|----------------|--------------|----
7  | My Files | My File Product | file         |
---|----------|----------------|--------------|----
```

2. リポジトリー情報を表示します。

```
# hammer repository info --name "My Files" --product "My File Product" --organization-id 1
```

HTTP が有効になっている場合には、出力は次のようになります。

```
Publish Via HTTP:   yes
Published At:       http://satellite.example.com/pulp/isos/uuid/
```

HTTP が有効になっていない場合には、出力は次のようになります。

```
Publish Via HTTP:   no
Published At:       https://satellite.example.com/pulp/isos/uuid/
```

3. クライアントに、適切な HTTP または HTTPS の形式でコマンドを入力します。
HTTP の場合:

```
# curl -O satellite.example.com/pulp/isos/uuid/my_file
```

HTTPS の場合:

```
# curl -O --cert ./Default\ Organization-key-cert.pem --cacert katello-server-ca.crt
satellite.example.com/pulp/isos/uuid/my_file
```

第16章 カスタム PUPPET コンテンツの管理

Satellite で、Puppet モジュールを使用してホストの状態設定を組み込む場合は、Puppet モジュールが状態設定を組み込むのに使用するリポジトリーで、カスタム製品を作成します。

16.1. カスタム PUPPET リポジトリーの作成

カスタム Puppet モジュールリポジトリーを作成する手順は、リポジトリー作成時に "strong">puppet タイプを選択することを除き、カスタムコンテンツの作成手順と同じです。製品を作成してから、カスタムリポジトリーを追加する必要があります。

手順

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、使用する製品をクリックします。
2. **リポジトリーの作成** をクリックします。
3. **名前** フィールドで、リポジトリーの名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容に基づいて **ラベル** フィールドに値が自動的に入力されます。
4. **タイプ** の一覧から **puppet** を選択します。
5. **URL** フィールドに、ソースとして使用する外部リポジトリーの URL を入力します。Puppet モジュールの同期にはリポジトリーソースを使用できます。
6. **保存**をクリックします。

CLI をご利用の場合

1. 以下のコマンドを実行して Puppet モジュールリポジトリーを作成します。

```
# hammer repository create \
--name "PostgreSQL Puppet Modules" \
--content-type "puppet" \
--product "PostgreSQL" \
--organization "My_Organization"
```

16.2. PUPPET モジュールの個別管理

RPM コンテンツと Puppet モジュールの両方が含まれるカスタム製品を作成して、カスタム RPM コンテンツを使用するサーバーをインストールして設定する場合には、「カスタム Puppet リポジトリーの作成」の手順を使用してから、以下の手順に従い、Puppet モジュールをアップロードします。

カスタム RPM のサポート

Red Hat では、Puppet Forge からのモジュールをサポートしていません。これらのモジュールに問題がある場合は、モジュール開発者に連絡してください。

前提条件

1. https://forge.puppetlabs.com/puppetlabs/postgresql など、Puppet Forge の Web サイトから、使用するモジュールをダウンロードしておく。

2. Web ブラウザーで、**download latest tar.gz** をクリックしてローカルのファイルシステムに保存しておく。

手順

1. Satellite Web UI で、**コンテンツ** > **製品** に移動し、管理する Puppet リポジトリーが含まれる製品を選択します。
2. リポジトリーウィンドウで、新しい Puppet リポジトリーをクリックすると、そのリポジトリーの詳細ページが表示されます。
3. **Puppet モジュールのアップロード** エリアに移動し、**参照** をクリックします。新たにダウンロードして抽出された Puppet モジュールを選択し、**アップロード** をクリックします。

Puppet モジュールを管理し、製品から削除するには、以下の手順を行います。

1. Puppet モジュールリポジトリーのウィンドウで、ウィンドウの右上の **コンテンツ数** エリアに移動します。**Puppet モジュール** 行で、Puppet モジュールに表示されている数値をクリックします。
2. Puppet モジュールリポジトリーウィンドウの **Puppet モジュールの管理** で、管理するモジュールを選択して、**アクションの選択** をクリックし、アクションを実行するか、**Puppet モジュールの削除** を選択します。

CLI をご利用の場合

1. 以下のコマンドで、使用中の Satellite Server のファイルシステムに Puppet モジュールをコピーします。

```
$ scp ~/puppet_module.tar.gz root@satellite.example.com:~/.
```

2. Puppet モジュールを Puppet Modules リポジトリーにインポートします。

```
# hammer repository upload-content \
--path ~/puppet_module.tar.gz \
--name "My Puppet Modules" \
--organization "My_Organization"
```

16.3. PUPPET リポジトリーの同期

Satellite Server は、アップロードした Puppet モジュールのリポジトリーを作成するだけでなく、完全な Puppet モジュールリポジトリーの同期ができます。この例では、Satellite Server は Puppet Forge リポジトリー全体を同期します。

カスタム RPM のサポート

Red Hat では、Puppet Forge からのモジュールをサポートしていません。モジュールは、同期プロセスのデモのために使用されています。これらのモジュールに問題がある場合は、モジュール開発者に連絡してください。

手順

1. Satellite Web UI で **コンテンツ** > **製品** に移動して、**製品の作成** をクリックします。

2. **名前** フィールドで、製品の名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した内容に基づいて **ラベル** フィールドに自動的に入力されます。
3. オプション: **GPG キー** の一覧から、製品の GPG キーを選択します。
4. オプション: **同期プラン** リストから製品の同期プランを選択します。
5. **説明**フィールドには、製品の説明を入力します。
6. **保存**をクリックします。
7. **リポジトリーの作成** をクリックすると、新しいリポジトリーのフォームが表示されます。
8. **名前** フィールドに、リポジトリーの名前を入力します。Red Hat Satellite 6 では、 **名前** に入力した名前をもとに、このフィールドに値が自動的に入力されます。
9. **タイプ** の一覧から **puppet** を選択します。
10. **URL** フィールドに、**http://forge.puppetlabs.com/** と入力します。
11. **保存** をクリックします。
12. 新しい Puppet リポジトリーを選択し、**今すぐ同期** をクリックして、Puppet Forge から Satellite Server に全モジュールをインポートします。この処理には時間がかかることがあります。

CLI をご利用の場合

1. 製品を作成します。

```
# hammer product create \
--name "Puppet Forge" \
--sync-plan "Example Plan" \
--description "All modules from Puppet Forge" \
--organization "My_Organization"
```

2. Puppet Forge リポジトリーを作成します。

```
# hammer repository create \
--name "Puppet Forge Modules" \
--content-type "puppet" \
--product "Puppet Forge" \
--organization "My_Organization" \
--url http://forge.puppetlabs.com/
```

3. リポジトリーを同期します。

```
# hammer repository synchronize \
--name "Puppet Forge Modules" \
--product "Puppet Forge" \
--organization "My_Organization"
```

Puppet Forge リポジトリーには数千のモジュールが含まれてるため、同期には時間がかかる場合があります。

16.4. GIT リポジトリーからの PUPPET MODULES の同期

Red Hat Satellite 6 には **pulp-puppet-module-builder** と呼ばれるユーティリティーが含まれており、これは **pulp-puppet-tools** RPM から他のシステムにインストールできます。このツールは Git リポジトリーをチェックアウトし、全モジュールをビルドして、それらを Satellite 6 が同期できる構造で公開します。一般的な方法の 1 つは、Satellite Server 上でこのユーティリティーを実行し、ローカルディレクトリーに公開して、そのディレクトリーに対して同期するというものです。以下は例となります。

```
# mkdir /modules
# chmod 755 /modules
# pulp-puppet-module-builder \
--output-dir=/modules \
--url=git@mygitserver.com:mymodules.git \
--branch=develop
```

この例では、Git リポジトリーの **develop** ブランチを **git@mygitserver.com:mymodules.git** からチェックアウトし、**/modules** に公開します。このディレクトリーを Satellite Server の新規リポジトリーの URL (**file:///modules**) として追加します。

リモート HTTP サーバー上の Puppet モジュールの公開

リモートの HTTP サーバー上にモジュールを公開する場合でも同じプロセスを実行します。たとえば、Puppet モジュールを公開する標準 Web ホストとして **webserver.example.com** を使用します。

```
# mkdir /var/www/html/modules/
# chmod 755 /var/www/html/modules/
# pulp-puppet-module-builder \
--output-dir=/var/www/html/modules/ \
--url=git@mygitserver.com:mymodules.git \
--branch=develop
```

Satellite Server では、リポジトリーの URL を **http://webserver.example.com/modules/** に設定します。

Web UI を使用した Git リポジトリーからの Puppet モジュールの同期

以下の手順を使用して、Git リポジトリーから Puppet モジュールを同期します。

手順

1. カスタム製品を作成し、**リポジトリーの作成** をクリックします。
2. **タイプ** の一覧から **puppet** を選択します。
3. **URL** フィールドに、ソースとして使用する外部 Git リポジトリーの URL を **file:///modules** の形式で入力します。

CLI をご利用の場合

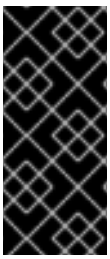
1. Puppet Forge リポジトリーを作成します。

```
# hammer repository create \
--name "Modules from Git" \
--content-type "puppet" \
```

```
--product "MyProduct" \
--organization "My_Organization" \
--url file:///modules
```

付録A コンテンツストレージ向け NFS 共有の使用

使用する環境ではコンテンツのストレージに十分な容量のハードディスクが必要になります。場合によっては、コンテンツのストレージに NFS 共有を使用することが便利なこともあります。本付録では、Satellite Server のコンテンツ管理コンポーネントに NFS 共有をマウントする方法を説明します。



重要

NFS 共有に、**/var/lib/pulp** すべてをマウントしないでください。**/var/lib/pulp** のファイルシステムは、帯域幅が高く、レイテンシーの低いストレージを使用するので、レイテンシーが高く、帯域幅の低いストレージでは、パフォーマンスが低下する問題が発生する可能性があります。NFS 共有は、 **/var/lib/pulp/content** ディレクトリーにのみ使用してください。

1. NFS 共有を作成します。この例では、**nfs.example.com:/satellite/content** で共有を使用します。この共有で適切なパーミッションが Satellite Server とその **apache** ユーザーに提供されるようにしてください。

2. Satellite ホストで **satellite-maintain** サービスを停止します。

```
# satellite-maintain service stop
```

3. Satellite Server に **nfs-utils** パッケージがインストールされていることを確認します。

```
# satellite-maintain packages install nfs-utils
```

4. **/var/lib/pulp/content** の既存のコンテンツを NFS 共有にコピーします。まず、NFS 共有を一時的な場所にマウントします。

```
# mkdir /mnt/temp
# mount -o rw nfs.example.com:/satellite/content /mnt/temp
```

/var/lib/pulp/content の既存コンテンツを一時的な場所にコピーします。

```
# cp -r /var/lib/pulp/content/* /mnt/temp/.
```

5. 共有上の全ファイルで **apache** ユーザーを使用するようにパーミッションを設定します。通常、このユーザーの ID は 48 になります。

6. 一時的なストレージの場所をアンマウントします。

```
# umount /mnt/temp
```

7. **/var/lib/pulp/content** の既存コンテンツを削除します。

```
# rm -rf /var/lib/pulp/content/*
```

8. **/etc/fstab** ファイルに以下の行を追加します。

```
nfs.example.com:/satellite/content    /var/lib/pulp/content    nfs
rw,hard,intr,context="system_u:object_r:httpd_sys_rw_content_t:s0"
```

これでシステムの再起動後もマウントが維持されます。SELinux コンテキストを含めることを忘れないでください。

9. マウントを有効にします。

```
# mount -a
```

10. NFS 共有が **var/lib/pulp/content** にマウントしていることを確認します。

```
# df
Filesystem                      1K-blocks     Used Available Use% Mounted on
...
nfs.example.com:/satellite/content 309506048 58632800 235128224  20%
/var/lib/pulp/content
...
```

既存のコンテンツが **var/lib/pulp/content** のマウントにあることを確認します。

```
# ls /var/lib/pulp/content
```

11. Satellite ホストで **satellite-maintain** サービスを起動します。

```
# satellite-maintain service start
```

これで Satellite Server はコンテンツの保存に NFS 共有を使用します。コンテンツの同期を実行して (「コンテンツの同期の概要」を参照) NFS 共有が予想どおり機能することを確認してください。

付録B コンテンツをローカル CDN サーバーと同期するための SATELLITE の設定

オフライン環境で、最新のセキュリティー更新、エラータ、パッケージをシステムにプロビジョニングするために必要なコンテンツが Satellite Server に含まれていることを確認する必要があります。これには、Red Hat カスタマーポータルからコンテンツの ISO イメージをダウンロードして、ローカルの CDN サーバーにインポートする手順を実行してください。Satellite Server のベースオペレーティングシステムか、HTTP 経由で Satellite にアクセス可能なシステムで、ローカル CDN サーバーをホストできます。次に、Satellite Server がローカルの CDN サーバーとコンテンツを同期するように設定する必要があります。

手順

1. Red Hat カスタマーポータル https://access.redhat.com へ移動し、ログインします。
2. 画面の左上で、**ダウンロード** をクリックし、**Red Hat Satellite** を選択します。
3. **コンテンツ ISO** タブをクリックします。このページには、サブスクリプションで利用できるすべての製品が一覧表示されます。
4. **Red Hat Enterprise Linux 7 Server (x86_64)**などの製品名のリンクをクリックして、ISO イメージをダウンロードします。
5. すべての Satellite コンテンツ ISO イメージを、ローカル CDN サーバーとして使用するシステムにコピーします。たとえば、Satellite Server の **/root/isos** ディレクトリーなどです。Satellite がインストールされているシステムにコンテンツを保存する必要はない点にご留意ください。CDN は、HTTP 経由で Satellite Server にアクセスできる限り、同じオフラインネットワーク内の別のシステムでホストできます。
6. ローカル CDN サーバーとして使用するシステムで、httpd 経由でアクセス可能なローカルディレクトリーを作成します (例: **/var/www/html/pub/sat-import/**)。

```
# mkdir -p /var/www/html/pub/sat-import/
```

7. マウントポイントを作成し、その場所に ISO イメージを一時的にマウントします。

```
# mkdir /mnt/iso
# mount -o loop /root/isos/first_iso /mnt/iso
```

8. 最初の ISO イメージのコンテンツをローカルディレクトリーに再帰的にコピーします。

```
# cp -ruv /mnt/iso/* /var/www/html/pub/sat-import/
```

9. マウントされたバイナリー DVD ISO イメージを使用する予定がない場合は、マウントポイントをアンマウントして削除します。

```
# umount /mnt/iso
# rmdir /mnt/iso
```

10. 各 ISO で上記の作業を繰り返して、コンテンツ ISO イメージからすべてのデータを **/var/www/html/pub/sat-import/** にコピーします。
11. ディレクトリーに正しい SELinux コンテキストが設定されていることを確認します。

```
# restorecon -rv /var/www/html/pub/sat-import/
```

12. Satellite Web UI で、**コンテンツ** > **サブスクリプション** に移動します。
13. **マニフェストの管理** をクリックします。
14. 以下の例のように、ローカルの CDN サーバーとして使用するシステムのホスト名に、新規作成したディレクトリーを指定して参照するように、**Red Hat CDN URL** フィールドを編集します。
    http://server.example.com/pub/sat-import/
15. **更新** をクリックして、マニフェストを Satellite にアップロードします。

付録C キックスタートリポジトリーのインポート

キックスタートリポジトリーは、コンテンツ ISO イメージでは提供されません。オフラインの Satellite でキックスタートリポジトリーを使用するには、使用する Red Hat Enterprise Linux のバージョンのバイナリー DVD ISO ファイルをダウンロードし、キックスタートファイルを Satellite にコピーする必要があります。

手順

1. Red Hat カスタマーポータル https://access.redhat.com/ へ移動し、ログインします。
2. ウィンドウの右上隅で **ダウンロード** をクリックします。
3. 使用する Red Hat Enterprise Linux のバージョンを探してクリックします (たとえば、**Red Hat Enterprise Linux 8** など)。
4. Red Hat Enterprise Linux のダウンロードウィンドウで、ISO イメージのバイナリー DVD バージョン (たとえば、**Red Hat Enterprise Linux 8.1 バイナリー DVD**) を見つけて、**今すぐダウンロード** をクリックします。
5. ダウンロードが完了したら、ISO イメージをSatellite Server にコピーします。
6. Satellite Server で、マウントポイントを作成し、その場所に ISO イメージを一時的にマウントします。

```
# mkdir /mnt/iso
# mount -o loop rhel-8.1-x86_64-dvd.iso /mnt/iso
```

7. AppStream および BaseOS のキックスタートディレクトリーを作成します。

```
# mkdir /var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart

# mkdir /var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart
```

Red Hat Enterprise Linux 7 を使用する場合は、1 つのディレクトリーのみで、以下のすべての手順を作成して完了する必要があることに注意してください。**/var/www/html/pub/sat-import/content/dist/rhel/server/7/7.7/x86_64/kickstart/**。

8. **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/listing** および **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/listing** のリストファイルに対して、**kickstart** を追加して改行します。

```
kickstart
```

9. **/var/www/html/pub/sat-import/content/dist/rhel8/listing** のリストファイルに対して、使用するオペレーティングシステムの ISO のバージョン番号を追加して改行します。たとえば、RHEL 8.1 バイナリー ISO の場合、**8.1** を追加して改行します。

```
8.1
```

10. ISO イメージから **kickstart** ファイルをコピーします。

```
# cp -a /mnt/iso/AppStream/* \
/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart
```

```
# cp -a /mnt/iso/BaseOS/* /mnt/iso/images/ \
/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart
```

BaseOS の場合は、**/mnt/iso/images/** ディレクトリーのコンテンツもコピーする必要があることに注意してください。

11. ISO イメージから **.treeinfo** ファイルをコピーします。

```
# cp /mnt/iso/.treeinfo \
/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart/treeinfo

# cp /mnt/iso/.treeinfo \
/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart/treeinfo
```

12. **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart/treeinfo** ファイルを開いて編集します。

13. **[general]** セクションで、以下の変更を加えます。

    - **packagedir = AppStream/Packages** を **packagedir = Packages** に変更します。
    - **repository = AppStream** を **repository = .** に変更します。
    - **variant = AppStream** を **variant = BaseOS** に変更します。
    - **variants = AppStream,BaseOS** を **variants = BaseOS** に変更します。

14. **[tree]** セクションで、**variants = AppStream,BaseOS** を **variants = BaseOS** に変更します。

15. **[variant-BaseOS]** セクションで、以下の変更を加えます。

    - **packages = BaseOS/Packages** を**packages = Packages** に変更します。
    - **repository = BaseOS** を **repository = .** に変更します。

16. **[media]** および **[variant-AppStream]** のセクションを削除します。

17. ファイルを保存して閉じます。

18. **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart/treeinfo** ファイルが以下の形式であることを確認します。

```
[checksums]
images/efiboot.img =
sha256:9ad9beee4c906cd05d227a1be7a499c8d2f20b3891c79831325844c845262bb6
images/install.img =
sha256:e246bf4aedfff3bb54ae9012f959597cdab7387aadb3a504f841bdc2c35fe75e
images/pxeboot/initrd.img =
sha256:a66e3c158f02840b19c372136a522177a2ab4bd91cb7269fb5bfdaaf7452efef
images/pxeboot/vmlinuz =
sha256:789028335b64ddad343f61f2abfdc9819ed8e9dfad4df43a2694c0a0ba780d16

[general]
; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
```

```
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 8.1.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = BaseOS
variants = BaseOS
version = 8.1.0

[header]
type = productmd.treeinfo
version = 1.2

[images-x86_64]
efiboot.img = images/efiboot.img
initrd = images/pxeboot/initrd.img
kernel = images/pxeboot/vmlinuz

[images-xen]
initrd = images/pxeboot/initrd.img
kernel = images/pxeboot/vmlinuz

[release]
name = Red Hat Enterprise Linux
short = RHEL
version = 8.1.0

[stage2]
mainimage = images/install.img

[tree]
arch = x86_64
build_timestamp = 1571146127
platforms = x86_64,xen
variants = BaseOS

[variant-BaseOS]
id = BaseOS
name = BaseOS
packages = Packages
repository = .
type = variant
uid = BaseOS
```

19. **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart/treeinfo** ファイルを開いて編集します。

20. **[general]** セクションで、以下の変更を加えます。

    - **packagedir = AppStream/Packages** を **packagedir = Packages** に変更します。
    - **repository = AppStream** を **repository = .** に変更します。
    - **variants = AppStream,BaseOS** を **variants = AppStream** に変更します。

21. **[tree]** セクションで、**variants = AppStream,BaseOS** を **variants = AppStream** に変更します。

22. **[variant-AppStream]** セクションで、以下の変更を加えます。

    - **packages = AppStream/Packages** を **packages = Packages** に変更します。
    - **repository = AppStream** を **repository = .** に変更します。

23. ファイルから次のセクションを削除します: **[checksums]**、**[images-x86_64]**、**[images-xen]**、**[media]**、**[stage2]**、**[variant-BaseOS]**。

24. ファイルを保存して閉じます。

25. **/var/www/html/pub/sat-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart/treeinfo** ファイルが以下の形式であることを確認します。

```
[general]
; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 8.1.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = AppStream
variants = AppStream
version = 8.1.0

[header]
type = productmd.treeinfo
version = 1.2

[release]
name = Red Hat Enterprise Linux
short = RHEL
version = 8.1.0

[tree]
arch = x86_64
build_timestamp = 1571146127
platforms = x86_64,xen
variants = AppStream

[variant-AppStream]
id = AppStream
name = AppStream
packages = Packages
repository = .
type = variant
uid = AppStream
```

26. マウントされたバイナリー DVD ISO イメージを使用する予定がない場合は、ディレクトリーをアンマウントして削除します。

```
# umount /mnt/iso
# rmdir /mnt/iso
```

27. Satellite Web UI で、Kickstart リポジトリーを有効にします。

付録D RED HAT CDN からコンテンツをダウンロードするために SATELLITE を戻す

お使いの環境がオフラインからオンラインに変更された場合は、Red Hat CDN から直接コンテンツをダウンロードするようにオフラインの Satellite を再設定することができます。

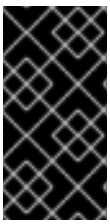
手順

1. Satellite Web UI で、**コンテンツ** > **サブスクリプション** に移動します。
2. **マニフェストの管理** をクリックします。
3. **Red Hat CDN URL**フィールドを編集して、Red Hat CDN URL をポイントします。
   https://cdn.redhat.com
4. **保存**をクリックします。

これで Satellite Server は、次にリポジトリーを同期するときに CDN からコンテンツをダウンロードするように設定されました。

付録E 接続済み SATELLITE SERVER へのコンテンツ ISO のインポート

Satellite Server が Red Hat カスタマーポータルに直接接続できる場合でも、初回同期はローカルにマウントされたコンテンツ ISO から実行することができます。この同期が完了すると、ネットワーク接続でのコンテンツのダウンロードに切り替えることができます。これを実行するには、Red Hat Satellite 向けのコンテンツ ISO を Red Hat カスタマーポータルからダウンロードし、Satellite Server にこれをインポートします。帯域幅に制限がある場合は、**オンデマンド** または **背景** ダウンロードポリシーを使用する方が、上記の方法よりも効率的な場合があります。



重要

Red Hat Enterprise Linux 8 のコンテンツ ISO イメージのみをインポートできます。これは、CDN のリポジトリーデータのチェックサムと、Red Hat Enterprise Linux 7 以前のコンテンツ ISO イメージからのリポジトリーデータのチェックサムが一致しないためです。

Red Hat Enterprise Linux ISO を同期する場合には、Red Hat Enterprise Linux のすべてのマイナーバージョンも同期される点に注意します。このため、Satellite に十分なストレージが必要です。

重要

お使いの Satellite Server がインターネットに接続している場合は、本セクションは必要ありません。

この例の手順では、コンテンツ ISO から Red Hat Enterprise Linux 8 リポジトリーを初めて同期する方法を説明します。

手順

1. Red Hat カスタマーポータル https://access.redhat.com へ移動し、ログインします。
2. 画面の左上で、**ダウンロード** をクリックし、**Red Hat Satellite** を選択します。
3. コンテンツ ISOタブをクリックします。このページには、サブスクリプションで利用できるすべての製品が一覧表示されます。
4. RHEL 8 (x86_64) などの製品名のリンクをクリックして、ISO イメージのダウンロードリンクを表示します。
5. ISO イメージをダウンロードします。
6. Satellite Server で、必要な Satellite コンテンツ ISO イメージすべてを一時的に保存するディレクトリーを作成します。この例では、**/tmp/isos/rhel8** を使用します。

```
# mkdir -p /tmp/isos/rhel8
```

7. ワークステーションで、ISO ファイルを Satellite Server にコピーします。

```
$ scp ~/Downloads/iso_file root@satellite.example.com:/tmp/isos/rhel8
```

8. Satellite Server で、ISO のマウントポイントとなるディレクトリーを作成します。

```
# mkdir /mnt/iso
```

9. ISO イメージを格納する作業ディレクトリーを作成します。

```
# mkdir /mnt/rhel8
```

10. 最初の ISO イメージを一時的にマウントします。

```
# mount -o loop /tmp/isos/iso_file /mnt/iso
```

11. 最初の ISO のコンテンツを作業ディレクトリーに再帰的にコピーします。

```
# cp -ruv /mnt/iso/* /mnt/rhel8/
```

12. ISO イメージをアンマウントします。

```
# umount /mnt/iso
```

13. 各 ISO で上記の作業を繰り返して、コンテンツ ISO イメージから全データを **/mnt/rhel8** にコピーします。

14. 必要に応じて、マウントポイントに使用した空のディレクトリーを削除します。

```
# rmdir /mnt/iso
```

15. 必要に応じて、一時的な作業ディレクトリーとそのコンテンツを削除して、スペースを確保します。

```
# rm -rf /tmp/isos/
```

16. ディレクトリーの所有者、SELinux コンテキスト、そのコンテンツを **/var/lib/pulp** と同じものにします。

```
# chcon -R --reference /var/lib/pulp  /mnt/rhel8/
# chown -R apache:apache /mnt/rhel8/
```

17. **/etc/pulp/content/sources/conf.d/local.conf** ファイルを作成または編集し、以下のテキストを追加します。

```
[rhel-8-server]
enabled: 1
priority: 0
expires: 3d
name: Red Hat Enterprise Linux 8
type: yum
base_url: file:///mnt/rhel8/content/dist/rhel/server/8/x86_64/os/
```

base_url のパスはコンテンツ ISO によって異なる場合があります。**base_url** で指定するディレクトリー内に **repodata** ディレクトリーが必要です。これがないと、同期は失敗します。複数のリポジトリーを同期するには、**/etc/pulp/content/sources/conf.d/local.conf** 設定ファイルで各リポジトリー向けの個別エントリーを作成します。

18. Satellite Web UI で、**コンテンツ** > **Red Hat リポジトリー**に移動し、以下のリポジトリーを有効にします。
    - **Red Hat Enterprise Linux 8 for x86_64 - BaseOS RPMs 8**
    - **Red Hat Enterprise Linux 8 for x86_64 - AppStream RPMs 8**
19. **コンテンツ** > **同期の状態** に移動して、同期するリポジトリーを選択し、**今すぐ同期** をクリックします。

Satellite Web UI では、使用されているソースが表示されないことに留意してください。ローカルのソースに問題がある場合は、Satellite はネットワーク経由でコンテンツをプルします。このプロセスを監視するには、Satellite で以下のコマンドを入力します。

```
# journalctl -f -l SYSLOG_IDENTIFIER=pulp | grep -v worker[\-,\.]heartbeat
```

上記のコマンドを実行すると対話的なログが表示されます。まず Satellite Server が Red Hat カスタマーポータルに接続してリポジトリーのメタデータをダウンロードして処理します。次に、ローカルリポジトリーが読み込まれます。エラーが発生したら Satellite Web UI で同期をキャンセルして、設定を確認してください。

同期が成功したら、**/etc/pulp/content/sources/conf.d/local.conf** からローカルソースのエントリーを削除して、このソースの接続を解除します。