Red Hat Satellite 6.15

ロードバランサーを使用したカプセルの設定

Capsules 間での負荷分散

Last Updated: 2024-04-24

Red Hat Satellite 6.15 ロードバランサーを使用したカプセルの設定

Capsules 間での負荷分散

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

法律上の通知

概要

このガイドでは、ロードバランサーを使用して Capsule Server 間で負荷を分散するように Red Hat Satellite を設定する方法について説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。この取り組みは膨大な作業を要するため、これらの変更による更新は可能な範囲で段階的に行われます。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。

Bugzilla でチケットを作成することでフィードバックを送信できます。

1. Bugzilla のWeb サイトに移動します。
2. **Component** フィールドで、**Documentation** を使用します。
3. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
4. **Submit Bug** をクリックします。

第1章 負荷分散ソリューションアーキテクチャー

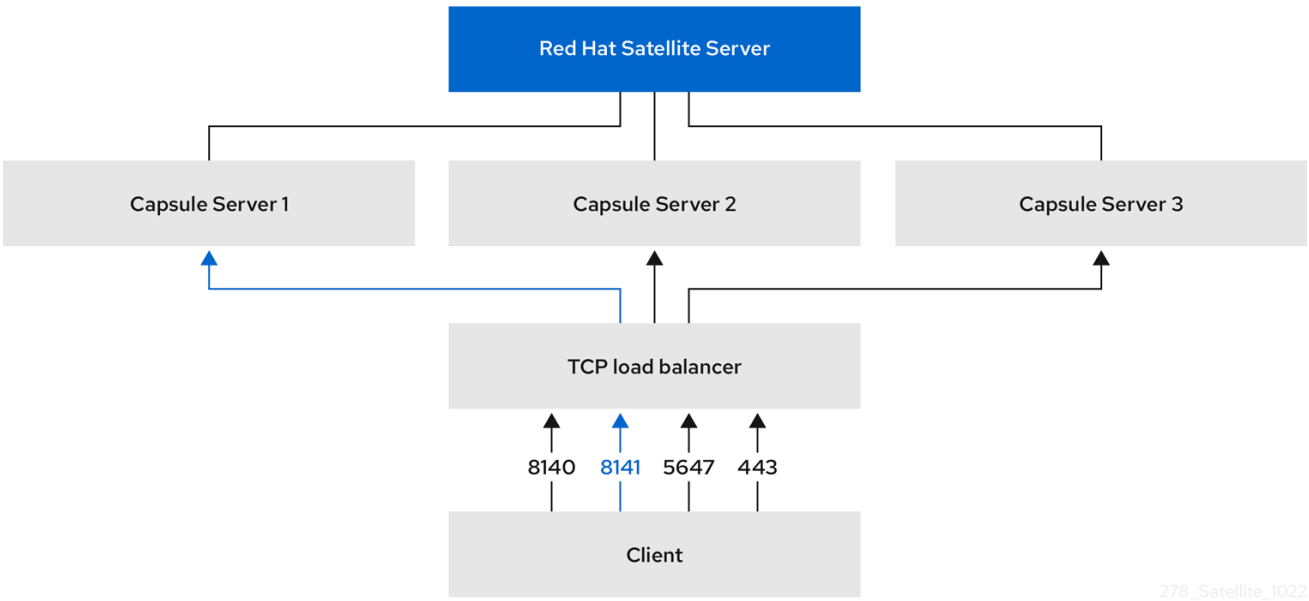
Satellite Server がロードバランサーを使用して複数の Capsule Server 間でクライアント要求とネットワーク負荷を分散するように設定できます。このように設定すると、Capsule Server の全体的なパフォーマンスが向上します。

本書では、ロードバランシングを使用できるように Satellite Server と Capsule Server を準備する方法を概説し、負荷分散型のセットアップで、クライアントを登録する方法やロードバランサーを設定する方法についてガイドラインを提供します。

負荷分散の設定は、以下のコンポーネントで設定されます。

- Satellite Server
- Capsule Server 2 台以上
- ロードバランサー 1 つ
- クライアント複数台

図1.1 Satellite 負荷分散ソリューションアーキテクチャー



負荷分散型の設定では、予定されたメンテナンスや予定外のメンテナンスで、Capsule Server が 1 台停止しても、ほぼすべての Capsule 機能は想定どおりに動作し続けます。ロードバランサーは、次のサービスおよび機能で動作します。

- **subscription-manager** での登録
- Yum リポジトリーによるコンテンツ管理
- オプション: Puppet

注記

負荷分散の設定では、ロードバランサーは上記のサービスと機能に対してのみ負荷を分散します。プロビジョニングや virt-who などの他のサービスが個別の Capsule で実行されている場合は、ロードバランサーではなく Capsule から直接アクセスする必要があります。

Puppet の制限の管理

Puppet 認証局 (CA) の管理では、負荷分散型の設定における証明書署名をサポートしていません。Puppet CA では、シリアル番号カウンターや CRL などの証明書情報がファイルシステムに保存されます。複数の書き込みプロセスで同一のデータを使用しようとすると、データが破損する可能性があります。

Puppet のこの制限を管理するには、次の手順を実行します。

1. Capsule Server 1 台 (通常、ロードバランシング用に Capsule Server を設定する最初のシステム) に Puppet 証明書署名を設定します。
2. ロードバランサー上のポート 8141 に CA 要求を送信するようにクライアントを設定します。
3. Puppet 証明書に署名するために Capsule Server を設定したシステムで、ポート 8141 からポート 8140 に CA 要求をリダイレクトするようにロードバランサーを設定します。

第2章 負荷分散の考慮事項

複数の Capsule Server 間で負荷を分散すると、1 つの Capsule が単一障害点になることを防ぎます。ロードバランサーを使用するように Capsule を設定すると、予定および予定外のシステム停止に適応できるので、可用性および応答性が向上します。

ロードバランシングを設定する場合は、次のガイドラインを考慮します。

- Puppet を使用する場合、Puppet 証明書署名は、設定する最初の Capsule に割り当てられます。最初の Capsule が停止していると、クライアントは Puppet コンテンツを取得できません。
- このソリューションでは、すべての Capsule で 1 つの状態を維持するために Pacemaker または他の同様の HA ツールを使用しません。問題をトラブルシュートするには、ロードバランサーを使用せずに Capsule ごとに問題を再現します。

負荷分散には追加のメンテナンスが必要

ロードバランサーを使用するように Capsule を設定すると、環境が複雑になり、新たなメンテナンスが必要になります。

ロードバランシングには、次の追加手順が必要です。

- すべてのカプセルが同じコンテンツビューを持ち、すべてのカプセルが同じコンテンツビューバージョンに同期されていることを確認する必要があります。
- 各 Capsule を順にアップグレードする必要があります。
- 設定した Capsule ごとに定期的にバックアップする必要があります。

負荷分散設定での Capsule Server のアップグレード

Capsule Server を 6.14 から 6.15 にアップグレードするには、**接続された Red Hat Satellite の 6.15 へのアップグレード** の Capsule Server のアップグレード 手順を完了します。ロードバランシング設定の Capsule Server では、他に必要な手順はありません。

第3章 負荷分散用に CAPSULE SERVER を設定するための前提条件

Capsule Server をロードバランシング用に設定するには、**Capsule Server のインストール** に記載の以下の手順を実行します。Satellite には、既存の Capsule Server をロードバランシング用に設定するサポートはありません。

1. Capsule Server の Satellite Server への登録
2. Satellite Infrastructure サブスクリプションのアタッチ
3. リポジトリーの設定
4. chronyd とシステムクロックの同期
5. Capsule Server パッケージのインストール

第4章 負荷分散のためのカプセルサーバーの設定

この章では、ロードバランシング用に Capsule Server を設定する方法を概説します。お使いの Satellite Server の設定に合わせて、次のいずれかのセクションに進んでください。

- 「Puppet を使用せずに負荷分散を行うために、デフォルトの SSL 証明書を使用して Capsule Server を設定する」
- 「Puppet による負荷分散のためにデフォルトの SSL 証明書を使用して Capsule Server を設定する」
- 「Puppet を使用せずに負荷分散を行うために、カスタム SSL 証明書を使用して Capsule Server を設定する」
- 「Puppet による負荷分散のためにカスタム SSL 証明書を使用して Capsule Server を設定する」

作成する Katello 証明書には、Capsule Server ごとに異なるファイル名を使用します。たとえば、Capsule Server FQDN を使用して証明書アーカイブファイルに名前を付けます。

4.1. PUPPET を使用せずに負荷分散を行うために、デフォルトの SSL 証明書を使用して CAPSULE SERVER を設定する

次のセクションでは、Puppet を使用せず、デフォルト SSL 証明書を使用する Capsule Server をロードバランシング用に設定する方法を説明します。この手順は、ロードバランシング用に設定する Capsule Server ごとに実行します。

手順

1. Satellite Server で、Capsule Server の Katello 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar "/root/capsule.example.com-certs.tar" \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule.example.com
```

capsule-certs-generate コマンドの出力である **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

2. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

```
# scp /root/capsule.example.com-certs.tar
root@capsule.example.com:/root/capsule.example.com-certs.tar
```

3. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--certs-cname "loadbalancer.example.com" \
--enable-foreman-proxy-plugin-remote-execution-script
```

4. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
```

```
--certs-tar-file "capsule.example.com-certs.tar" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
--foreman-proxy-trusted-hosts "capsule.example.com"
```

4.2. PUPPET による負荷分散のためにデフォルトの SSL 証明書を使用して CAPSULE SERVER を設定する

次のセクションでは、Puppet を使用し、デフォルト SSL 証明書を使用する Capsule Server をロードバランシング用に設定する方法を説明します。

Satellite 設定で Puppet を使用する場合は、次の手順を実行する必要があります。

1. 「Puppet 証明書を生成して署名するために、デフォルトの SSL 証明書を使用して Capsule Server を設定する」
2. 「負荷分散のためにデフォルトの SSL 証明書を使用して残りの Capsule Server を設定する」

4.2.1. Puppet 証明書を生成して署名するために、デフォルトの SSL 証明書を使用して Capsule Server を設定する

この手順は、ロードバランシング用に設定した他の Capsule Server すべてに、Puppet 証明書を生成して署名するように Capsule Server を設定するシステムにのみ、実行してください。

手順

1. Satellite Server で、Puppet 証明書を生成し、署名するように Capsule Server を設定するシステムの Katello 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar "/root/capsule-ca.example.com-certs.tar" \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule-ca.example.com
```

   capsule-certs-generate コマンドの出力である **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

2. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

```
# scp /root/capsule-ca.example.com-certs.tar root@capsule-ca.example.com:capsule-ca.example.com-certs.tar
```

3. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--certs-cname "loadbalancer.example.com" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-puppetca "true" \
```

```
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "true"
```

4. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
--certs-tar-file "capsule-ca.example.com-certs.tar" \
--enable-foreman-proxy-plugin-remote-execution-script \
--enable-puppet \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-puppetca "true" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
--foreman-proxy-trusted-hosts "capsule-ca.example.com" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server true \
--puppet-server-ca "true"
```

5. Capsule Server で、Puppet サーバーを停止します。

```
# puppet resource service puppetserver ensure=stopped
```

6. ロードバランシングを設定する他のすべての Capsule Server に対して Puppet 証明書を生成します。ただし、Puppet 証明書署名を設定する最初のシステムを除きます。

```
# puppetserver ca generate \
--ca-client \
--certname capsule.example.com \
--subject-alt-names loadbalancer.example.com
```

このコマンドは、Capsule Server が Puppet 証明書に署名するように設定するシステムで、次のファイルを作成します。

- **/etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem**
- **/etc/puppetlabs/puppet/ssl/certs/ca.pem**
- **/etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem**
- **/etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem**

7. Puppet サーバーを再開します。

```
# puppet resource service puppetserver ensure=running
```

4.2.2. 負荷分散のためにデフォルトの SSL 証明書を使用して残りの Capsule Server を設定する

この手順は、Capsule Server が Puppet 証明書を署名するように設定するシステムを除き、各 Capsule Server で実行します。

手順

1. Satellite Server で、Capsule Server の Katello 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar "/root/capsule.example.com-certs.tar" \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule.example.com
```

capsule-certs-generate コマンドの出力である **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

2. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

```
# scp /root/capsule.example.com-certs.tar
root@capsule.example.com:/root/capsule.example.com-certs.tar
```

3. Capsule Server で、**puppetserver** パッケージをインストールします。

```
# satellite-maintain packages install puppetserver
```

4. Capsule Server で、Puppet 証明書用のディレクトリーを作成します。

```
# mkdir -p /etc/puppetlabs/puppet/ssl/certs/ \
/etc/puppetlabs/puppet/ssl/private_keys/ \
/etc/puppetlabs/puppet/ssl/public_keys/
```

5. Capsule Server で、Capsule Server を設定するシステムから、対象の Capsule Server の Puppet 証明書をコピーして、Puppet 証明書を署名します。

```
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem
/etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/certs/ca.pem
/etc/puppetlabs/puppet/ssl/certs/ca.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem
/etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem
/etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem
```

6. Capsule Server で、**/etc/puppetlabs/puppet/ssl/** ディレクトリーの所有権をユーザー **puppet** およびグループ **puppet** に変更します。

```
# chown -R puppet:puppet /etc/puppetlabs/puppet/ssl/
```

7. Capsule Server で、**/etc/puppetlabs/puppet/ssl/** ディレクトリーの SELinux コンテキストを設定します。

```
# restorecon -Rv /etc/puppetlabs/puppet/ssl/
```

8. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--certs-cname "loadbalancer.example.com" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-puppetca "false" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "false"
```

9. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
--certs-tar-file "capsule.example.com-certs.tar" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-puppetca "false" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
--foreman-proxy-trusted-hosts "capsule.example.com" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "false"
```

4.3. PUPPET を使用せずに負荷分散を行うために、カスタム SSL 証明書を使用して CAPSULE SERVER を設定する

次のセクションでは、Puppet を使用せず、カスタム SSL 証明書を使用する Capsule Server をロードバランシング用に設定する方法を説明します。

4.3.1. Capsule Server 用のカスタム SSL 証明書の作成

以下の手順では、Certificate Signing Request (CSR: 証明書署名要求) の設定ファイルを作成して、Subject Alternative Names (SAN: サブジェクトの別名) としてロードバランサーと Capsule Server を追加する方法を説明します。この手順は、ロードバランシング用に設定する Capsule Server ごとに実行します。

手順

1. ソースの証明書ファイルすべてを保存するには、**root** ユーザーだけがアクセスできるディレクトリーを作成します。

```
# mkdir /root/capsule_cert
```

2. 証明書署名要求 (CSR) に署名する秘密鍵を作成します。
秘密鍵は暗号化する必要がないことに注意してください。パスワードで保護された秘密鍵を使用する場合は、秘密鍵のパスワードを削除します。

   この Capsule Server の秘密鍵がすでにある場合は、この手順を省略します。

```
# openssl genrsa -out /root/capsule_cert/capsule_cert_key.pem 4096
```

3. CSR 用の **/root/capsule_cert/openssl.cnf** 設定ファイルを作成して、以下のコンテンツを追加します。

```
[ req ]
req_extensions = v3_req
distinguished_name = req_distinguished_name
x509_extensions = usr_cert
prompt = no

[ req_distinguished_name ]
commonName = capsule.example.com (1)

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
extendedKeyUsage = serverAuth, clientAuth, codeSigning, emailProtection
subjectAltName = @alt_names

[alt_names] (2)
DNS.1 = loadbalancer.example.com
DNS.2 = capsule.example.com
```

(1) 証明書の共通名は、Capsule Server の FQDN と一致する必要があります。ロードバランシング用に設定する各 Capsule Server でコマンドを実行するたびに、必ず変更します。ワイルドカードの値 * を設定することもできます。ワイルドカードの値を設定しており、**katello-certs-check** コマンドを使用する場合には、**-t capsule** オプションを追加する必要があります。

(2) **[alt_names]** で、ロードバランサーの FQDN は **DNS.1**、Capsule Server の FQDN は **DNS.2** として追加します。

4. オプション: CSR に識別名 (DN) の詳細を追加する場合は、**[req_distinguished_name]** セクションに次の情報を追加します。

```
[req_distinguished_name]
CN = capsule.example.com
countryName =My_Country_Name (1)
stateOrProvinceName = My_State_Or_Province_Name (2)
localityName = My_Locality_Name (3)
organizationName = My_Organization_Or_Company_Name
organizationalUnitName = My_Organizational_Unit_Name (4)
```

(1) 2 文字コード

(2) 名前

(3) フルネーム (例: ニューヨーク)

(4) 証明書を担当する部門 (例:IT 部門)

5. CSR を生成します。

```
# openssl req -new \
-key /root/capsule_cert/capsule_cert_key.pem \ 1
-config /root/capsule_cert/openssl.cnf \ 2
-out /root/capsule_cert/capsule_cert_csr.pem 3
```

1. 秘密鍵へのパス
2. 設定ファイルへのパス
3. 生成する CSR へのパス

6. 認証局 (CA) に証明書署名要求を送信します。Satellite Server と Capsule Server の証明書には同じ CA が署名する必要があります。
   要求を送信する場合は、証明書の有効期限を指定してください。証明書要求の送信方法にはさまざまなものがあるため、推奨される方法について CA にお問い合わせください。要求すると、CA バンドルと署名済み証明書を別々のファイルで受け取ることになります。
7. 認証局バンドル、認証局から受け取る Capsule Server の証明書ファイル、Capsule Server の秘密鍵を、Satellite Server にコピーします。
8. Satellite Server で、Capsule Server 証明書入力ファイルを検証します。

```
# katello-certs-check \
-c /root/capsule_cert/capsule_cert.pem \ 1
-k /root/capsule_cert/capsule_cert_key.pem \ 2
-b /root/capsule_cert/ca_cert_bundle.pem 3
```

1. 認証局により提供された Capsule Server 証明書ファイル
2. 証明書の署名に使用した Capsule Server の秘密鍵
3. 認証局により提供された認証局バンドル

commonName= をワイルドカードの値 * に設定する場合には、**-t capsule** オプションを **katello-certs-check** コマンドに追加する必要があります。

katello-certs-check コマンドの出力である **capsule-certs-generate** コマンドの例をメモして、この Capsule Server の認証アーカイブファイルを作成します。

4.3.2. Puppet を使用せずに負荷分散を行うために、カスタム SSL 証明書を使用して Capsule Server を設定する

次のセクションでは、Puppet を使用せず、カスタム SSL 証明書を使用する Capsule Server をロードバランシング用に設定する方法を説明します。この手順は、ロードバランシング用に設定する Capsule Server ごとに実行します。

手順

1. **katello-certs-check** コマンドの出力から取得する **capsule-certs-generate** コマンドに次のオプションを追加します。

```
--foreman-proxy-cname loadbalancer.example.com
```

2. Satellite Server で、**capsule-certs-generate** コマンドを入力して Capsule 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar /root/capsule_cert/capsule.tar \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule.example.com \
--server-ca-cert /root/capsule_cert/ca_cert_bundle.pem \
--server-cert /root/capsule_cert/capsule.pem \
--server-key /root/capsule_cert/capsule.pem
```

出力からの **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

3. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

```
# scp /root/capsule.example.com-certs.tar
root@capsule.example.com:capsule.example.com-certs.tar
```

4. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--certs-cname "loadbalancer.example.com" \
--enable-foreman-proxy-plugin-remote-execution-script
```

5. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
--certs-tar-file "capsule.example.com-certs.tar" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
--foreman-proxy-trusted-hosts "capsule.example.com"
```

4.4. PUPPET による負荷分散のためにカスタム SSL 証明書を使用して CAPSULE SERVER を設定する

Satellite 設定で Puppet を使用する場合は、次の手順を実行する必要があります。

1. 「Puppet 証明書を生成して署名するために、カスタム SSL 証明書を使用して Capsule Server を設定する」
2. 「負荷分散のために残りの Capsule Server をカスタム SSL 証明書で設定する」

4.4.1. Capsule Server 用のカスタム SSL 証明書の作成

以下の手順では、Certificate Signing Request (CSR: 証明書署名要求) の設定ファイルを作成して、Subject Alternative Names (SAN: サブジェクトの別名) としてロードバランサーと Capsule Server を追加する方法を説明します。この手順は、ロードバランシング用に設定する Capsule Server ごとに実行し

ます。

手順

1. ソースの証明書ファイルすべてを保存するには、**root** ユーザーだけがアクセスできるディレクトリーを作成します。

```
# mkdir /root/capsule_cert
```

2. 証明書署名要求 (CSR) に署名する秘密鍵を作成します。
秘密鍵は暗号化する必要がないことに注意してください。パスワードで保護された秘密鍵を使用する場合は、秘密鍵のパスワードを削除します。

この Capsule Server の秘密鍵がすでにある場合は、この手順を省略します。

```
# openssl genrsa -out /root/capsule_cert/capsule_cert_key.pem 4096
```

3. CSR 用の **/root/capsule_cert/openssl.cnf** 設定ファイルを作成して、以下のコンテンツを追加します。

```
[ req ]
req_extensions = v3_req
distinguished_name = req_distinguished_name
x509_extensions = usr_cert
prompt = no

[ req_distinguished_name ]
commonName = capsule.example.com (1)

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
extendedKeyUsage = serverAuth, clientAuth, codeSigning, emailProtection
subjectAltName = @alt_names

[alt_names] (2)
DNS.1 = loadbalancer.example.com
DNS.2 = capsule.example.com
```

(1) 証明書の共通名は、Capsule Server の FQDN と一致する必要があります。ロードバランシング用に設定する各 Capsule Server でコマンドを実行するたびに、必ず変更します。ワイルドカードの値 * を設定することもできます。ワイルドカードの値を設定しており、**katello-certs-check** コマンドを使用する場合には、**-t capsule** オプションを追加する必要があります。

(2) **[alt_names]** で、ロードバランサーの FQDN は **DNS.1**、Capsule Server の FQDN は **DNS.2** として追加します。

4. オプション: CSR に識別名 (DN) の詳細を追加する場合は、**[req_distinguished_name]** セクションに次の情報を追加します。

```
[req_distinguished_name]
CN = capsule.example.com
```

```
countryName =My_Country_Name (1)
stateOrProvinceName = My_State_Or_Province_Name (2)
localityName = My_Locality_Name (3)
organizationName = My_Organization_Or_Company_Name
organizationalUnitName = My_Organizational_Unit_Name (4)
```

(1) 2 文字コード

(2) 名前

(3) フルネーム (例: ニューヨーク)

(4) 証明書を担当する部門 (例:IT 部門)

5. CSR を生成します。

```
# openssl req -new \
-key /root/capsule_cert/capsule_cert_key.pem \ (1)
-config /root/capsule_cert/openssl.cnf \ (2)
-out /root/capsule_cert/capsule_cert_csr.pem (3)
```

(1) 秘密鍵へのパス

(2) 設定ファイルへのパス

(3) 生成する CSR へのパス

6. 認証局 (CA) に証明書署名要求を送信します。Satellite Server と Capsule Server の証明書には同じ CA が署名する必要があります。
要求を送信する場合は、証明書の有効期限を指定してください。証明書要求の送信方法にはさまざまなものがあるため、推奨される方法について CA にお問い合わせください。要求すると、CA バンドルと署名済み証明書を別々のファイルで受け取ることになります。

7. 認証局バンドル、認証局から受け取る Capsule Server の証明書ファイル、Capsule Server の秘密鍵を、Satellite Server にコピーします。

8. Satellite Server で、Capsule Server 証明書入力ファイルを検証します。

```
# katello-certs-check \
-c /root/capsule_cert/capsule_cert.pem \ (1)
-k /root/capsule_cert/capsule_cert_key.pem \ (2)
-b /root/capsule_cert/ca_cert_bundle.pem (3)
```

(1) 認証局により提供された Capsule Server 証明書ファイル

(2) 証明書の署名に使用した Capsule Server の秘密鍵

(3) 認証局により提供された認証局バンドル

commonName= をワイルドカードの値 * に設定する場合には、**-t capsule** オプションを **katello-certs-check** コマンドに追加する必要があります。

katello-certs-check コマンドの出力である **capsule-certs-generate** コマンドの例をメモして、この Capsule Server の認証アーカイブファイルを作成します。

4.4.2. Puppet 証明書を生成して署名するために、カスタム SSL 証明書を使用して Capsule Server を設定する

この手順は、ロードバランシング用に設定した他の Capsule Server すべてに、Puppet 証明書を生成するように Capsule Server を設定するシステムにのみ、実行してください。

手順

1. **katello-certs-check** コマンドの出力から取得する **capsule-certs-generate** コマンドに次のオプションを追加します。

```
--foreman-proxy-cname loadbalancer.example.com
```

2. Satellite Server で、**capsule-certs-generate** コマンドを入力して Capsule 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar /root/capsule_cert/capsule-ca.tar \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule-ca.example.com \
--server-ca-cert /root/capsule_cert/ca_cert_bundle.pem \
--server-cert /root/capsule_cert/capsule-ca.pem \
--server-key /root/capsule_cert/capsule-ca.pem
```

出力からの **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

3. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

4. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-puppetca "true" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "true"
```

5. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
--certs-tar-file "certs.tgz" \
--enable-foreman-proxy-plugin-remote-execution-script \
--enable-puppet \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-puppetca "true" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
```

```
--foreman-proxy-trusted-hosts "capsule-ca.example.com" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server true \
--puppet-server-ca "true"
```

6. Capsule Server で、ロードバランシングを設定する他のすべての Capsule に対して Puppet 証明書を生成します。ただし、Puppet 証明書署名を設定する最初のシステムを除きます。

```
# puppet cert generate capsule.example.com \
--dns_alt_names=loadbalancer.example.com
```

このコマンドは、Puppet 証明書署名を行う Capsule Server インスタンスに次のファイルを作成します。

- **/etc/puppetlabs/puppet/ssl/certs/ca.pem**
- **/etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem**
- **/etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem**
- **/etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem**

4.4.3. 負荷分散のために残りの Capsule Server をカスタム SSL 証明書で設定する

この手順は、Puppet 証明書に署名するために Capsule Server を設定するシステムを除き、各 Capsule Server で実行します。

手順

1. **katello-certs-check** コマンドの出力から取得する **capsule-certs-generate** コマンドに次のオプションを追加します。

```
--foreman-proxy-cname loadbalancer.example.com
```

2. Satellite Server で、**capsule-certs-generate** コマンドを入力して Capsule 証明書を生成します。

```
# capsule-certs-generate \
--certs-tar /root/capsule_cert/capsule.tar \
--foreman-proxy-cname loadbalancer.example.com \
--foreman-proxy-fqdn capsule.example.com \
--server-ca-cert /root/capsule_cert/ca_cert_bundle.pem \
--server-cert /root/capsule_cert/capsule.pem \
--server-key /root/capsule_cert/capsule.pem
```

出力からの **satellite-installer** コマンド例のコピーを保持し、Capsule Server 証明書をインストールします。

3. 証明書アーカイブファイルを Satellite Server から Capsule Server にコピーします。

```
# scp /root/capsule.example.com-certs.tar
root@capsule.example.com:capsule.example.com-certs.tar
```

4. Capsule Server で、**puppetserver** パッケージをインストールします。

```
# satellite-maintain packages install puppetserver
```

5. Capsule Server で、Puppet 証明書用のディレクトリーを作成します。

```
# mkdir -p /etc/puppetlabs/puppet/ssl/certs/ \
/etc/puppetlabs/puppet/ssl/private_keys/ \
/etc/puppetlabs/puppet/ssl/public_keys/
```

6. Capsule Server で、Capsule Server を設定するシステムから、対象の Capsule Server の Puppet 証明書をコピーして、Puppet 証明書を署名します。

```
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem /etc/puppetlabs/puppet/ssl/certs/capsule.example.com.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/certs/ca.pem /etc/puppetlabs/puppet/ssl/certs/ca.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem /etc/puppetlabs/puppet/ssl/private_keys/capsule.example.com.pem
# scp root@capsule-ca.example.com:/etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem /etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.pem
```

7. Capsule Server で、**/etc/puppetlabs/puppet/ssl/** ディレクトリーの所有権をユーザー **puppet** およびグループ **puppet** に変更します。

```
# chown -R puppet:puppet /etc/puppetlabs/puppet/ssl/
```

8. Capsule Server で、**/etc/puppetlabs/puppet/ssl/** ディレクトリーの SELinux コンテキストを設定します。

```
# restorecon -Rv /etc/puppetlabs/puppet/ssl/
```

9. **capsule-certs-generate** コマンドの出力から取得する **satellite-installer** コマンドに次のオプションを追加します。

```
--certs-cname "loadbalancer.example.com" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-puppetca "false" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "false"
```

10. Capsule Server で、**satellite-installer** コマンドを入力します。

```
# satellite-installer --scenario capsule \
--certs-cname "loadbalancer.example.com" \
--certs-tar-file "capsule.example.com-certs.tar" \
--enable-foreman-proxy-plugin-remote-execution-script \
--foreman-proxy-foreman-base-url "https://satellite.example.com" \
--foreman-proxy-oauth-consumer-key "oauth key" \
```

```
--foreman-proxy-oauth-consumer-secret "oauth secret" \
--foreman-proxy-puppetca "false" \
--foreman-proxy-register-in-foreman "true" \
--foreman-proxy-trusted-hosts "satellite.example.com" \
--foreman-proxy-trusted-hosts "capsule.example.com" \
--puppet-ca-server "capsule-ca.example.com" \
--puppet-dns-alt-names "loadbalancer.example.com" \
--puppet-server-ca "false"
```

第5章 ホスト登録用のロードバランサーの設定

ホスト登録機能を使用する場合、ロードバランサーを介してクライアントを登録するように Satellite を設定できます。

Capsule の代わりにロードバランサーにホストを登録できるようになります。ロードバランサーは、リクエスト時にホストを登録する Capsule を決定します。登録時に、ホスト上のサブスクリプションマネージャーがロードバランサーを介してコンテンツを管理するように設定されます。

前提条件

- すべての Capsule Server で SSL 証明書を設定している。詳細は、*4章 負荷分散のためのカプセルサーバーの設定* を参照してください。
- すべての Capsule Server で登録およびテンプレートプラグインを有効にしました。

```
# satellite-installer --scenario capsule \
--foreman-proxy-registration true \
--foreman-proxy-templates true
```

手順

1. すべての Capsule Server で、**satellite-installer** を使用して登録 URL とテンプレート URL を設定します。

```
# satellite-installer --scenario capsule \
--foreman-proxy-registration-url "https://loadbalancer.example.com:9090" \
--foreman-proxy-template-url "https://loadbalancer.example.com:8000"
```

2. Satellite Web UI で、**Infrastructure** > **Capsules** に移動します。
3. 各 Capsule について、**Actions** 列のドロップダウンメニューをクリックし、**Refresh** を選択します。

第6章 ロードバランサーのインストール

以下の例は、Red Hat Enterprise Linux 8 サーバーを使用して HAProxy ロードバランサーを設定するための一般的なガイダンスを提供します。ただし、適切なロードバランシングソフトウェアソリューションをインストールして、TCP 転送をサポートすることができます。

手順

1. HAProxy をインストールします。

```
# dnf install haproxy
```

2. **semanage** ツールが含まれる、次のパッケージをインストールします。

```
# dnf install policycoreutils-python-utils
```

3. SELinux で HAProxy がどのポートでもバインドできるように設定します。

```
# semanage boolean --modify --on haproxy_connect_any
```

4. 表6.1「ロードバランサーのポート設定」 の説明に従って、ポートのネットワーク負荷を分散するようにロードバランサーを設定します。たとえば、HAProxy のポートを設定するには、**/etc/haproxy/haproxy.cfg** ファイルを表に合わせて編集します。詳細は、**Red Hat ナレッジベース** の Configuration example for haproxy.cfg for HAProxy load balancer with Satellite 6 を参照してください。

表6.1 ロードバランサーのポート設定

サービス	ポート	モード	バランスモード	宛先
HTTP	80	TCP	roundrobin	すべての Capsule Server のポート 80
HTTPS および RHSM	443	TCP	source	すべての Capsule Server のポート 443
テンプレート取得用の Anaconda	8000	TCP	roundrobin	すべての Capsule Server のポート 8000
Puppet (**オプション**)	8140	TCP	roundrobin	すべての Capsule Server のポート 8140
PuppetCA (**オプション**)	8141	TCP	roundrobin	Puppet 証明書に署名するように Capsule Server を設定するシステムだけのポート 8140
ホスト登録用および OpenSCAP (オプション) 用の Capsule HTTPS	9090	TCP	roundrobin	すべての Capsule Server のポート 9090

5. SSL オフロードを無効にして、クライアント側の SSL 証明書がバックエンドサーバーにパススルーできるようにロードバランサーを設定します。クライアントから Capsule Server への通信はクライアント側の SSL 証明書に依存するので、この設定が必要です。

6. HAProxy サービスを起動し、有効にします。

```
# systemctl enable --now haproxy
```

第7章 負荷分散設定の検証

この手順を使用して、各 Capsule Server のロードバランシング設定を確認します。

手順

1. Capsule Server のベースオペレーティングシステムをシャットダウンします。
2. この Capsule に登録されているクライアントでコンテンツまたは Subscription Management 機能が利用できることを確認します。たとえば、クライアントで **subscription-manager refresh** コマンドを入力します。
3. Capsule Server のベースオペレーティングシステムを再起動します。

第8章 ロードバランサーへのクライアントの登録

クライアントからのネットワークトラフィックの負荷を分散するには、クライアントをロードバランサーに登録する必要があります。

クライアントを登録するには、次のいずれかの手順を実行します。

- 「ホスト登録を使用してクライアントを登録する」
- 「**(非推奨)** ブートストラップスクリプトを使用してクライアントを登録する」

8.1. ホスト登録を使用してクライアントを登録する

Satellite Web UI、Hammer CLI、または Satellite API のホスト登録機能を使用して、Satellite にホストを登録できます。詳細は、**ホストの管理** の ホストの登録 を参照してください。

前提条件

- ホスト登録用のロードバランサーを設定している。詳細は、*5章 ホスト登録用のロードバランサーの設定* を参照してください。

手順

1. Satellite Web UI で、**Hosts > Register Host** に移動します。
2. **Capsule** ドロップダウンリストから、ロードバランサーを選択します。
3. 以前に Capsule Server に登録されていたホストを登録するには、**Force** を選択します。
4. **Activation Keys** リストから、ホストに割り当てるアクティベーションキーを選択します。
5. **Generate** をクリックして登録コマンドを作成します。
6. **ファイル** アイコンをクリックして、コマンドをクリップボードにコピーします。
7. SSH を使用してホストに接続し、登録コマンドを実行します。
8. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なリポジトリーが有効であることを確認します。

CLI 手順

1. Hammer CLI を使用してホスト登録コマンドを生成します。

```
# hammer host-registration generate-command \
--activation-keys "My_Activation_Key"
```

ホストが Satellite Server の SSL 証明書を信頼しない場合は、登録コマンドに **--insecure** フラグを追加して SSL 検証を無効にすることができます。

```
# hammer host-registration generate-command \
--activation-keys "My_Activation_Key" \
--insecure true
```

--smart-proxy-id My_Capsule_ID オプションを含めます。ホスト登録のロードバランシング用に設定した任意の Capsule Server の ID を使用できます。Satellite はロードバランサーを登録コマンドに自動的に適用します。

以前に Capsule Server に登録されていたホストを登録するには、**--force** オプションを追加します。

2. SSH を使用してホストに接続し、登録コマンドを実行します。
3. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なリポジトリーが有効であることを確認します。

API の手順

1. Satellite API を使用してホスト登録コマンドを生成します。

```
# curl -X POST https://satellite.example.com/api/registration_commands \
--user "My_User_Name" \
-H 'Content-Type: application/json' \
-d '{ "registration_command": { "activation_keys": ["My_Activation_Key_1,
My_Activation_Key_2"] }}'
```

ホストが Satellite Server の SSL 証明書を信頼しない場合は、登録コマンドに **--insecure** フラグを追加して SSL 検証を無効にすることができます。

```
# curl -X POST https://satellite.example.com/api/registration_commands \
--user "My_User_Name" \
-H 'Content-Type: application/json' \
-d '{ "registration_command": { "activation_keys": ["My_Activation_Key_1,
My_Activation_Key_2"], "insecure": true }}'
```

アクティベーションキーを使用すると、その環境を簡単に指定できます。詳細は、**コンテンツの管理** の アクティベーションキーの管理 を参照してください。

{ "smart_proxy_id": My_Capsule_ID } を含めます。ホスト登録のロードバランシング用に設定した任意の Capsule Server の ID を使用できます。Satellite はロードバランサーを登録コマンドに自動的に適用します。

以前に Capsule Server に登録されていたホストを登録するには、**{ "force": true }** を追加します。

コマンドライン引数としてパスワードを入力するには、**username:password** 構文を使用します。これにより、パスワードがシェル履歴に保存される可能性があることに注意してください。または、パスワードの代わりに一時的なパーソナルアクセストークンを使用することもできます。Satellite Web UI でトークンを生成するには、**My Account** > **Personal Access Tokens** に移動します。

2. SSH を使用してホストに接続し、登録コマンドを実行します。
3. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なリポジトリーが有効であることを確認します。

8.2. (非推奨) ブートストラップスクリプトを使用してクライアントを登録する

クライアントを登録するには、クライアント上で次のコマンドを入力します。クライアントごとに登録の手順を実行する必要があります。

前提条件

- クライアントにブートストラップスクリプトをインストールし、スクリプトのファイル権限を実行可能に変更していることを確認している。詳細は、**ホストの管理** の ブートストラップスクリプトを使用して Red Hat Satellite にホストを登録する を参照してください。

手順

- Red Hat Enterprise Linux 8 の場合は、以下のコマンドを入力します。

```
# /usr/libexec/platform-python bootstrap.py \
--activationkey="My_Activation_Key" \
--enablerepos=satellite-client-6-for-rhel-8-<arch>-rpms \ 1
--force \ 2
--hostgroup="My_Host_Group" \
--location="My_Location" \
--login=admin \
--organization="My_Organization" \
--puppet-ca-port 8141 \ 3
--server loadbalancer.example.com
```

1. **<arch>** をクライアントアーキテクチャー (**x86** など) に置き換えます。
2. スタンドアロンの Capsule に以前に登録されていたクライアントを登録するには、**--force** オプションを追加します。
3. Puppet を使用している場合は、**--puppet-ca-port 8141** オプションを含めます。

- Red Hat Enterprise Linux 7 または 6 の場合は、以下のコマンドを入力します。

```
# python bootstrap.py --login=admin \
--activationkey="My_Activation_Key" \
--enablerepos=rhel-7-server-satellite-client-6-rpms \
--force \ 1
--hostgroup="My_Host_Group" \
--location="My_Location" \
--organization="My_Organization" \
--puppet-ca-port 8141 \ 2
--server loadbalancer.example.com
```

1. スタンドアロンの Capsule に以前に登録されていたクライアントを登録するには、**--force** オプションを追加します。
2. Puppet を使用している場合は、**--puppet-ca-port 8141** オプションを含めます。

このスクリプトでは、**--login** オプションで入力した Satellite ユーザー名に対応するパスワードの入力が求められます。

第9章 ロードバランサーを介して SCAP コンテンツを伝播する

OpenSCAP を使用してクライアントのセキュリティーコンプライアンスを管理する場合は、ARF レポートを Capsule ではなくロードバランサーに送信するように SCAP クライアントを設定する必要があります。設定手順は、コンプライアンスポリシーをデプロイするために選択した方法によって異なります。

9.1. ANSIBLE デプロイメントを使用した SCAP コンテンツの伝播

この手順を使用すると、Ansible デプロイメント方法の範囲内で、ロードバランサーを介して Security Content Automation Protocol (SCAP) コンテンツをプロモートできます。

前提条件

- コンプライアンスポリシーの Ansible デプロイメント向けに Satellite を設定している。詳細は、**セキュリティーコンプライアンスの管理** の コンプライアンスポリシーのデプロイメント方法の設定 を参照してください。

手順

1. Satellite Web UI で、**設定** > **Ansible** > **変数** に移動します。
2. **foreman_scap_client_port** 変数を検索し、その名前をクリックします。
3. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
4. **Parameter Type** リストで、**integer** を選択します。
5. **Default Value** フィールドに、**9090** と入力します。
6. **Specify Matchers** エリアで、デフォルト値をオーバーライドするすべてのマッチャーを削除します。
7. **Submit** をクリックします。
8. **foreman_scap_client_server** 変数を検索し、その名前をクリックします。
9. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
10. **Parameter Type** リストで、**string** を選択します。
11. **Default Value** フィールドに、ロードバランサーの FQDN (**loadbalancer.example.com** など) を入力します。
12. **Specify Matchers** エリアで、デフォルト値をオーバーライドするすべてのマッチャーを削除します。
13. **Submit** をクリックします。
14. Ansible を使用したコンプライアンスポリシーのデプロイに進みます。詳細は以下を参照してください。
    - **セキュリティーコンプライアンスの管理** における Ansible を使用したホストグループへのポリシーの導入

- **セキュリティーコンプライアンスの管理** における Ansible を使用したホストへのポリシーの導入

検証

- クライアント上で、**/etc/foreman_scap_client/config.yaml** ファイルに次の行が含まれていることを確認します。

```
# Foreman proxy to which reports should be uploaded
:server: 'loadbalancer.example.com'
:port: 9090
```

9.2. PUPPET デプロイメントを使用した SCAP コンテンツの伝播

この手順を使用すると、Puppet デプロイメント方法の範囲内で、ロードバランサーを介して Security Content Automation Protocol (SCAP) コンテンツをプロモートできます。

前提条件

- コンプライアンスポリシーの Puppet デプロイメント向けに Satellite を設定している。詳細は、**セキュリティーコンプライアンスの管理** の コンプライアンスポリシーのデプロイメント方法の設定 を参照してください。

手順

1. Satellite Web UI で、**設定** > **Puppet ENC** > **クラス** に移動します。
2. **foreman_scap_client を** クリックします。
3. **Smart Class Parameter** タブをクリックします。
4. **Smart Class Parameter** ウィンドウの左側のペインで、**port** をクリックします。
5. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
6. **Key Type** の一覧から **integer** を選択します。
7. **Default Value** フィールドに、**9090** と入力します。
8. **Smart Class Parameter** ウィンドウの左側のペインで、**server** をクリックします。
9. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
10. **Key Type** の一覧から **string** を選択します。
11. **Default Value** フィールドに、ロードバランサーの FQDN (**loadbalancer.example.com** など) を入力します。
12. **Smart Class Parameter** ウィンドウの左下のペインで、**Submit** をクリックします。
13. Puppet を使用したコンプライアンスポリシーのデプロイに進みます。詳細は以下を参照してください。

    - **セキュリティーコンプライアンスの管理** における Puppet を使用したホストグループへのポリシーの導入

- **セキュリティーコンプライアンスの管理** における Puppet を使用したホストへのポリシーの導入

検証

- クライアント上で、**/etc/foreman_scap_client/config.yaml** ファイルに次の行が含まれていることを確認します。

```
# Foreman proxy to which reports should be uploaded
:server: 'loadbalancer.example.com'
:port: 9090
```