



Red Hat Satellite 6.12

ロードバランサーを使用した Capsule の設定

Capsule Server 間での負荷分散

Red Hat Satellite 6.12 ロードバランサーを使用した Capsule の設定

Capsule Server 間での負荷分散

Red Hat Satellite Documentation Team

satellite-doc-list@redhat.com

法律上の通知

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

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

RED HAT ドキュメントへのフィードバック (英語のみ)	3
第1章 ロードバランシングソリューションのアーキテクチャー	4
第2章 ロードバランシングに関する考慮事項	6
第3章 ロードバランシング用に CAPSULE SERVER を設定するための前提条件	7
第4章 CAPSULE SERVER のロードバランシング用の設定	8
4.1. デフォルト SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用しない)	8
4.2. デフォルト SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用する)	9
4.3. カスタム SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用しない)	12
4.4. カスタム SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用する)	15
第5章 ロードバランサーのインストール	21
第6章 ロードバランシング設定の確認	23
第7章 ロードバランサーへのクライアントの登録	24
7.1. ホスト登録機能を使用したクライアントの登録	24
7.2. (非推奨) ブートストラップスクリプトを使用したクライアントの登録	25
7.3. (非推奨) KATELLO-CA-CONSUMER RPM を使用したクライアントの手動登録	27
第8章 ロードバランサーを介した SCAP コンテンツの伝播	28
8.1. ANSIBLE デプロイメントを使用した SCAP コンテンツの伝播	28
8.2. PUPPET デプロイメントを使用した SCAP コンテンツの伝播	29

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 ロードバランシングソリューションのアーキテクチャー

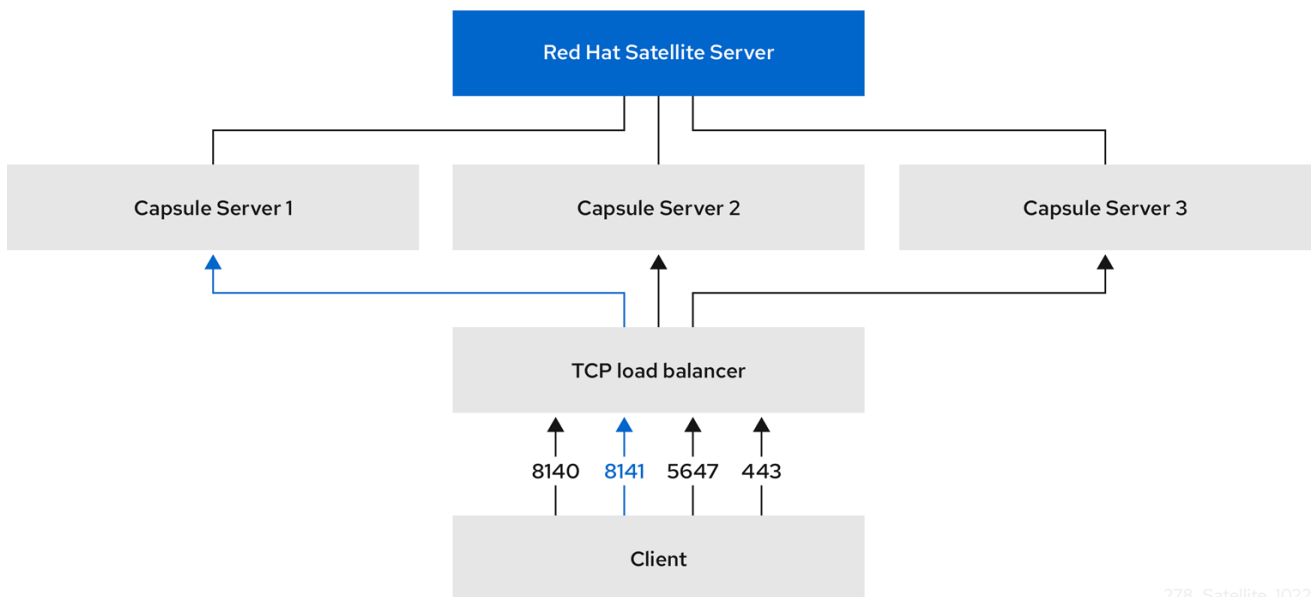
Satellite Server がロードバランサーを使用して複数の Capsule Server 間でクライアント要求とネットワーク負荷を分散するように設定できます。このように設定すると、Capsule Server の全体的なパフォーマンスが向上します。

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 ロードバランシングソリューションのアーキテクチャー



278_Satellite_1022

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、以下のサービスおよび機能と連携します。

- **subscription-manager** での登録
- **yum** リポジトリでのコンテンツ管理
- 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、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 ロードバランシングに関する考慮事項

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 に割り当てられません。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ティングするには、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。

ロードバランシングに必要な追加メンテナンス

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 Capsule に同じコンテンツビューがあることを確認し、すべての Capsule を同一のコンテンツビューバージョンに同期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。

ロードバランシング設定の Capsule Server のアップグレード

Capsule Servers 6.11 から 6.12 にアップグレードするには、[Red Hat Satellite のアップグレードおよび更新の Capsule Server のアップグレード](#) の手順を実行します。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 ロードバランシング用に 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 CAPSULE SERVER のロードバランシング用の設定

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。

- 「[デフォルト SSL 証明書を使用する Capsule Server のロードバランシング用の設定 \(Puppet を使用しない\)](#)」
- 「[デフォルト SSL 証明書を使用する Capsule Server のロードバランシング用の設定 \(Puppet を使用する\)](#)」
- 「[カスタム SSL 証明書を使用する Capsule Server のロードバランシング用の設定 \(Puppet を使用しない\)](#)」
- 「[カスタム SSL 証明書を使用する Capsule Server のロードバランシング用の設定 \(Puppet を使用する\)](#)」

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. デフォルト SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用しない)

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. デフォルト SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用する)

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 証明書を生成および署名するためのデフォルトの SSL 証明書を使用した Capsule Server の設定](#)」
2. 「[ロードバランシング向けにデフォルトの SSL 証明書を使用した残りの Capsule Server の設定](#)」

4.2.1. Puppet 証明書を生成および署名するためのデフォルトの SSL 証明書を使用した Capsule Server の設定

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"
```

- 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" \
--puppet-server-foreman-url "https://satellite.example.com"
```

- Capsule Server で、Puppet サーバーを停止します。

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

- 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

- Puppet サーバーを再開します。

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

4.2.2. ロードバランシング向けにデフォルトの SSL 証明書を使用した残りの Capsule Server の設定

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" \  
--puppet-server-foreman-url "https://satellite.example.com"
```

4.3. カスタム SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用しない)

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 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. Capsule Server で、すべてのソース証明書ファイルを含むディレクトリーを作成し、**root** ユーザーのみがアクセスできるようにします。

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

2. Certificate Signing Request (CSR) を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
```

- 証明書要求設定ファイルを作成して、次の内容を追加します。

```
[ req ]
default_bits      = 4096
distinguished_name = req_distinguished_name
req_extensions    = req_ext
prompt = no

[ req_distinguished_name ]
countryName=2 Letter Country Code
stateOrProvinceName=State or Province Full Name
localityName=Locality Name
0.organizationName=Organization Name
organizationalUnitName=Capsule Organization Unit Name
commonName=capsule.example.com 1
emailAddress=Email Address

[ req_ext ]
#authorityKeyIdentifier=keyid,issuer
#basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
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。

- SAN 証明書の Certificate Signing Request (CSR) を作成します。

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

- 1** 証明書を署名するために使用される Capsule Server の秘密鍵
- 2** 証明書要求の設定ファイル

3 Certificate Signing Request ファイル

- 証明書要求を認証局に送信します。
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。証明書要求を送信する方法は異なるため、推奨される方法について認証局にお問い合わせください。要求への応答で、認証局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。
- 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。
- 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 \  
-k /root/capsule_cert/capsule_cert_key.pem \  
-b /root/capsule_cert/ca_cert_bundle.pem
```

- 認証局により提供された Capsule Server 証明書ファイル
- 証明書の署名に使用した Capsule Server の秘密鍵
- 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. カスタム SSL 証明書を使用する Capsule Server のロードバランシング用の設定 (Puppet を使用しない)

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

- 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
```

- 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. カスタム SSL 証明書を使用する CAPSULE SERVER のロードバランシング用の設定 (PUPPET を使用する)

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 証明書を生成および署名するためのカスタム SSL 証明書を使用した Capsule Server の設定](#)」
2. 「[ロードバランシング向けにカスタム SSL 証明書を使用した残りの Capsule Server の設定](#)」

4.4.1. Capsule Server 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. Capsule Server で、すべてのソース証明書ファイルを含むディレクトリを作成し、**root** ユーザーのみがアクセスできるようにします。

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

2. Certificate Signing Request (CSR) を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
```

- 証明書要求設定ファイルを作成して、次の内容を追加します。

```
[ req ]
default_bits      = 4096
distinguished_name = req_distinguished_name
req_extensions    = req_ext
prompt = no

[ req_distinguished_name ]
countryName=2 Letter Country Code
stateOrProvinceName=State or Province Full Name
localityName=Locality Name
0.organizationName=Organization Name
organizationalUnitName=Capsule Organization Unit Name
commonName=capsule.example.com 1
emailAddress=Email Address

[ req_ext ]
#authorityKeyIdentifier=keyid,issuer
#basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
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。

- SAN 証明書の Certificate Signing Request (CSR) を作成します。

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

- 1** 証明書を署名するために使用される Capsule Server の秘密鍵
- 2** 証明書要求の設定ファイル

3 Certificate Signing Request ファイル

- 証明書要求を認証局に送信します。
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。証明書要求を送信する方法は異なるため、推奨される方法について認証局にお問い合わせください。要求への応答で、認証局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。
- 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。
- 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 証明書を生成および署名するためのカスタム SSL 証明書を使用した Capsule Server の設定

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" \
--puppet-server-foreman-url "https://satellite.example.com"
```

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. ロードバランシング向けにカスタム SSL 証明書を使用した残りの Capsule Server の設定

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
```

- 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/
```

- 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/
```

- 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"
```

- 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" \  
--puppet-server-foreman-url "https://satellite.example.com"
```


第5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5.1「ロードバランサー用のポート設定」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5.1 ロードバランサー用のポート設定

サービス	ポート	モード	バランスマード	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
AMQP	5647	TCP	roundrobin	全 Capsule Server のポート 5647
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
SmartProxy (OpenScap のオプション)	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6章 ロードバランシング設定の確認

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7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。

- [「ホスト登録機能を使用したクライアントの登録」](#)
- [「\(非推奨\) ブートストラップスクリプトを使用したクライアントの登録」](#)
- [「\(非推奨\) katello-ca-consumer RPM を使用したクライアントの手動登録」](#)

7.1. ホスト登録機能を使用したクライアントの登録

ホスト登録機能、Satellite API、または Hammer CLI を使用して、ホストを Satellite に登録できます。

手順

1. Satellite Web UI で、**ホスト > ホストの登録** に移動します。
2. **Generate** をクリックして登録コマンドを作成します。
3. **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。
4. 登録するホストにログインして、以前に作成したコマンドを実行します。
5. **rhsm.baseurl** および **server.hostname** のサブスクリプションマネージャーの設定を更新します。

```
# subscription-manager config \
--rhsm.baseurl=https://loadbalancer.example.com/pulp/content \
--server.hostname=loadbalancer.example.com
```

6. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なりポジトリ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
```

2. 登録するホストにログインして、以前に作成したコマンドを実行します。

3. **rhsm.baseurl** および **server.hostname** のサブスクリプションマネージャーの設定を更新します。

```
# subscription-manager config \
--rhsm.baseurl=https://loadbalancer.example.com/pulp/content \
--server.hostname=loadbalancer.example.com
```

4. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なりポジトリ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。

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。

ホストの登録に関する情報は [ホストの管理](#) の [Red Hat Satellite へのホストの登録](#) を参照してください。

2. 登録するホストにログインして、以前に作成したコマンドを実行します。
3. **rhsm.baseurl** および **server.hostname** のサブスクリプションマネージャーの設定を更新します。

```
# subscription-manager config \
--rhsm.baseurl=https://loadbalancer.example.com/pulp/content \
--server.hostname=loadbalancer.example.com
```

4. **/etc/yum.repos.d/redhat.repo** ファイルをチェックして、適切なりポジトリが有効であることを確認します。

7.2. (非推奨) ブートストラップスクリプトを使用したクライアントの登録

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](#) ガイドの [ブートストラップスクリプトを使用した Red Hat Satellite へのホストの登録](#) 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_Hostgroup" \
--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_Hostgroup" \
--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。

7.3. (非推奨) KATELLO-CA-CONSUMER RPM を使用したクライアントの手動登録

クライアントを手動で登録するには、登録する各クライアントで次の手順を実行します。

手順

1. **katello-ca-consumer** パッケージがインストールされている場合は削除します。

```
# dnf remove 'katello-ca-consumer'
```

2. ロードバランサーから **katello-ca-consumer** パッケージをインストールします。

```
# dnf install http://loadbalancer.example.com/pub/katello-ca-consumer-latest.noarch.rpm
```

3. クライアントを登録し、**--serverurl** と **--baseurl** のオプションを追加します。

```
# subscription-manager register \  
--activationkey="My_Activation_Key" \  
--baseurl=https://loadbalancer.example.com/pulp/content/ \  
--org="My_Organization" \  
--serverurl=https://loadbalancer.example.com/rhsm
```

第8章 ロードバランサーを介した SCAP コンテンツの伝播

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。

8.1. 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、[Red Hat Satellite 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、**Configure > Variables** 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。
 - [Red Hat Satellite の管理の Ansible を使用したホストグループへのポリシーのデプロイ](#)
 - [Red Hat Satellite の管理の Ansible を使用したホストへのポリシーのデプロイ](#)

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
```

8.2. 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、[Red Hat Satellite 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 **Configure > Classes** に移動して、**foreman_scap_client** をクリックします。
2. **Smart Class Parameter** タブをクリックします。
3. **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。
4. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
5. **Key Type** の一覧から **integer** を選択します。
6. **Default Value** フィールドに、**9090** と入力します。
7. **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。
8. **Default Behavior** エリアで、**Override** チェックボックスを選択します。
9. **Key Type** の一覧から **string** を選択します。
10. **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。
11. **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。
12. 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。
 - [Red Hat Satellite の管理の Puppet を使用したホストグループへのポリシーのデプロイ](#)
 - [Red Hat Satellite の管理の Puppet を使用したホストへのポリシーのデプロイ](#)

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
```