Configuring Capsules with a Load Balancer

Distribute load among Capsules
Red Hat Satellite 6.14 Configuring Capsules with a Load Balancer

Distribute load among Capsules

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

This guide describes how to configure Red Hat Satellite to use a load balancer to distribute load among Capsule Servers.
## Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION ............................................. 3

CHAPTER 1. LOAD BALANCING SOLUTION ARCHITECTURE ........................................ 4

CHAPTER 2. LOAD BALANCING CONSIDERATIONS ..................................................... 6

CHAPTER 3. PREREQUISITES FOR CONFIGURING CAPSULE SERVERS FOR LOAD BALANCING .... 7

CHAPTER 4. CONFIGURING CAPSULE SERVERS FOR LOAD BALANCING .......................... 8
  4.1. CONFIGURING CAPSULE SERVER WITH DEFAULT SSL CERTIFICATES FOR LOAD BALANCING
       WITHOUT PUPPET .................................................................................. 8
  4.2. CONFIGURING CAPSULE SERVER WITH DEFAULT SSL CERTIFICATES FOR LOAD BALANCING
       WITH PUPPET .................................................................................. 9
     4.2.1. Configuring Capsule Server with Default SSL Certificates to Generate and Sign Puppet
            Certificates .............................................................................. 9
     4.2.2. Configuring Remaining Capsule Servers with Default SSL Certificates for Load Balancing .... 11
  4.3. CONFIGURING CAPSULE SERVER WITH CUSTOM SSL CERTIFICATES FOR LOAD BALANCING
       WITHOUT PUPPET ........................................................................... 12
     4.3.1. Creating Custom SSL Certificates for Capsule Server ............................................. 12
     4.3.2. Configuring Capsule Server with Custom SSL Certificates for Load Balancing without Puppet .... 14
  4.4. CONFIGURING CAPSULE SERVER WITH CUSTOM SSL CERTIFICATES FOR LOAD BALANCING
       WITH PUPPET ................................................................................ 15
     4.4.1. Creating Custom SSL Certificates for Capsule Server ............................................. 15
     4.4.2. Configuring Capsule Server with Custom SSL Certificates to Generate and Sign Puppet
            Certificates .............................................................................. 17
     4.4.3. Configuring Remaining Capsule Servers with Custom SSL Certificates for Load Balancing .... 18

CHAPTER 5. SETTING THE LOAD BALANCER FOR HOST REGISTRATION ....................... 21

CHAPTER 6. INSTALLING THE LOAD BALANCER ....................................................... 22

CHAPTER 7. VERIFYING THE LOAD BALANCING CONFIGURATION ............................. 24

CHAPTER 8. REGISTERING CLIENTS TO THE LOAD BALANCER ................................. 25
  8.1. REGISTERING CLIENTS USING HOST REGISTRATION ........................................ 25
  8.2. (DEPRECATED) REGISTERING CLIENTS USING THE BOOTSTRAP SCRIPT .......... 26
  8.3. (DEPRECATED) REGISTERING CLIENTS MANUALLY USING KATELLO-CA-CONSUMER RPM .. 27

CHAPTER 9. PROPAGATING SCAP CONTENT THROUGH THE LOAD BALANCER .......... 29
  9.1. PROPAGATING SCAP CONTENT USING ANSIBLE DEPLOYMENT ..................... 29
  9.2. PROPAGATING SCAP CONTENT USING PUPPET DEPLOYMENT ....................... 30
We appreciate your input on our documentation. Please let us know how we could make it better.

You can submit feedback by filing a ticket in Bugzilla:

1. Navigate to the Bugzilla website.
2. In the Component field, use Documentation.
3. In the Description field, enter your suggestion for improvement. Include a link to the relevant parts of the documentation.
4. Click Submit Bug.
CHAPTER 1. LOAD BALANCING SOLUTION ARCHITECTURE

You can configure Satellite Server to use a load balancer to distribute client requests and network load across multiple Capsule Servers. This results in an overall performance improvement on Capsule Servers.

This guide outlines how to prepare Satellite Server and Capsule Server for load balancing, and provides guidelines on how to configure a load balancer and register clients in a load-balanced setup.

A load-balanced setup consists of the following components:

- Satellite Server
- Two or more Capsule Servers
- A load balancer
- Multiple clients

In a load-balanced setup, nearly all Capsule functionality continues to work as expected when one Capsule Server is down, for planned or unplanned maintenance. Load balancer works with the following services and features:

- Registration using `subscription-manager`
- Content Management with `yum` repositories
- Optional: Puppet

**NOTE**

In the load-balanced setup, a load balancer distributes load only for the services and features mentioned above. If other services, such as provisioning or virt-who, are running on the individual Capsules, you must access them directly through Capsules and not through the load balancer.


Managing Puppet Limitations

Puppet Certificate Authority (CA) management does not support certificate signing in a load-balanced setup. Puppet CA stores certificate information, such as the serial number counter and CRL, on the file system. Multiple writer processes that attempt to use the same data can corrupt it.

To manage this Puppet limitation, complete the following steps:

1. Configure Puppet certificate signing on one Capsule Server, typically the first system where you configure Capsule Server for load balancing.
2. Configure the clients to send CA requests to port 8141 on a load balancer.
3. Configure a load balancer to redirect CA requests from port 8141 to port 8140 on the system where you configure Capsule Server to sign Puppet certificates.
CHAPTER 2. LOAD BALANCING CONSIDERATIONS

Distributing load between several Capsule Servers prevents any one Capsule from becoming a single point of failure. Configuring Capsules to use a load balancer can provide resilience against planned and unplanned outages. This improves availability and responsiveness.

Consider the following guidelines when configuring load balancing:

- If you use Puppet, Puppet certificate signing is assigned to the first Capsule that you configure. If the first Capsule is down, clients cannot obtain Puppet content.
- This solution does not use Pacemaker or other similar HA tools to maintain one state across all Capsules. To troubleshoot issues, reproduce the issue on each Capsule, bypassing the load balancer.

Additional Maintenance Required for Load Balancing

Configuring Capsules to use a load balancer results in a more complex environment and requires additional maintenance.

The following additional steps are required for load balancing:

- You must ensure that all Capsules have the same Content Views and synchronize all Capsules to the same Content View versions
- You must upgrade each Capsule in sequence
- You must backup each Capsule that you configure regularly

Upgrading Capsule Servers in a Load Balancing Configuration

To upgrade Capsule Servers from 6.13 to 6.14, complete the Upgrading Capsule Servers procedure in Upgrading Red Hat Satellite to 6.14. There are no additional steps required for Capsule Servers in a load balancing configuration.
To configure Capsule Servers for load balancing, complete the following procedures described in *Installing Capsule Server*. Satellite does not support configuring existing Capsule Servers for load balancing.

1. **Registering Capsule Server to Satellite Server**
2. **Attaching the Satellite Infrastructure Subscription**
3. **Configuring Repositories**
4. **Synchronizing the System Clock With chronyd**
5. **Installing Capsule Server Packages**
CHAPTER 4. CONFIGURING CAPSULE SERVERS FOR LOAD BALANCING

This chapter outlines how to configure Capsule Servers for load balancing. Proceed to one of the following sections depending on your Satellite Server configuration:

- **Section 4.1, “Configuring Capsule Server with Default SSL Certificates for Load Balancing without Puppet”**
- **Section 4.2, “Configuring Capsule Server with Default SSL Certificates for Load Balancing with Puppet”**
- **Section 4.3, “Configuring Capsule Server with Custom SSL Certificates for Load Balancing without Puppet”**
- **Section 4.4, “Configuring Capsule Server with Custom SSL Certificates for Load Balancing with Puppet”**

Use different file names for the Katello certificates you create for each Capsule Server. For example, name the certificate archive file with Capsule Server FQDN.

### 4.1. CONFIGURING CAPSULE SERVER WITH DEFAULT SSL CERTIFICATES FOR LOAD BALANCING WITHOUT PUPPET

The following section describes how to configure Capsule Servers that use default SSL certificates for load balancing without Puppet. Complete this procedure on each Capsule Server that you want to configure for load balancing.

**Procedure**

1. On Satellite Server, generate Katello certificates for Capsule Server:

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

   Retain a copy of the example `satellite-installer` command that is output by the `capsule-certs-generate` command for installing Capsule Server certificate.

2. Copy the certificate archive file from Satellite Server to Capsule Server.

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

3. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

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

4. On Capsule Server, enter the `satellite-installer` command:
4.2. CONFIGURING CAPSULE SERVER WITH DEFAULT SSL CERTIFICATES FOR LOAD BALANCING WITH PUPPET

The following section describes how to configure Capsule Servers that use default SSL certificates for load balancing with Puppet.

If you use Puppet in your Satellite configuration, you must complete the following procedures:

1. Section 4.2.1, “Configuring Capsule Server with Default SSL Certificates to Generate and Sign Puppet Certificates”

2. Section 4.2.2, “Configuring Remaining Capsule Servers with Default SSL Certificates for Load Balancing”

4.2.1. Configuring Capsule Server with Default SSL Certificates to Generate and Sign Puppet Certificates

Complete this procedure only for the system where you want to configure Capsule Server to generate and sign Puppet certificates for all other Capsule Servers that you configure for load balancing.

Procedure

1. On Satellite Server, generate Katello certificates for the system where you configure Capsule Server to generate and sign Puppet certificates:

```
# 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 \
--foreman-proxy-trusted-hosts "capsule.ca.example.com"
```

Retain a copy of the example `satellite-installer` command that is output by the `capsule-certs-generate` command for installing Capsule Server certificate.

2. Copy the certificate archive file from Satellite Server to Capsule Server:

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

3. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

```
--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. On Capsule Server, enter the satellite-installer command:

```bash
# 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"
```

5. On Capsule Server, stop the Puppet server:

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

6. Generate Puppet certificates for all other Capsule Servers that you configure for load balancing, except the first system where you configure Puppet certificates signing:

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

This command creates the following files on the system where you configure Capsule Server to sign Puppet certificates:

- `/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. Resume the Puppet server:

```bash
# puppet resource service puppetserver ensure=running
```
4.2.2. Configuring Remaining Capsule Servers with Default SSL Certificates for Load Balancing

Complete this procedure on each Capsule Server excluding the system where you configure Capsule Server to sign Puppet certificates.

Procedure

1. On Satellite Server, generate Katello certificates for Capsule Server:

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

   Retain a copy of the example `satellite-installer` command that is output by the `capsule-certs-generate` command for installing Capsule Server certificate.

2. Copy the certificate archive file from Satellite Server to Capsule Server:

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

3. On Capsule Server, install the `puppetserver` package:

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

4. On Capsule Server, create directories for puppet certificates:

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

5. On Capsule Server, copy the Puppet certificates for this Capsule Server from the system where you configure Capsule Server to sign Puppet certificates:

   ```
   # 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/capsule.example.com.pem /etc/puppetlabs/puppet/ssl/public_keys/capsule.example.com.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
   ```

6. On Capsule Server, change the `/etc/puppetlabs/puppet/ssl/` directory ownership to user `puppet` and group `puppet`:

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

7. On Capsule Server, set the SELinux context for the `/etc/puppetlabs/puppet/ssl/` directory:
8. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

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

9. On Capsule Server, enter the `satellite-installer` command:

```bash
# 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. CONFIGURING CAPSULE SERVER WITH CUSTOM SSL CERTIFICATES FOR LOAD BALANCING WITHOUT PUPPET

The following section describes how to configure Capsule Servers that use custom SSL certificates for load balancing without Puppet.

#### 4.3.1. Creating Custom SSL Certificates for Capsule Server

This procedure outlines how to create a configuration file for the Certificate Signing Request and include the load balancer and Capsule Server as Subject Alternative Names (SAN). Complete this procedure on each Capsule Server that you want to configure for load balancing.

**Procedure**

1. On Capsule Server, create a directory to contain all the source certificate files, accessible to only the `root` user:

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

2. Create a private key with which to sign the Certificate Signing Request (CSR).
   Note that the private key must be unencrypted. If you use a password-protected private key, remove the private key password.
If you already have a private key for this Capsule Server, skip this step.

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

3. Create the certificate request configuration file with the following content:

```ini
[ 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. The certificate’s common name must match the FQDN of Capsule Server. Ensure to change this when running the command on each Capsule Server that you configure for load balancing. You can also set a wildcard value `*`. If you set a wildcard value, you must add the `-t capsule` option when you use the `katello-certs-check` command.

2. Under [alt_names], include the FQDN of the load balancer as DNS.1 and the FQDN of Capsule Server as DNS.2.

4. Create a Certificate Signing Request (CSR) for the SAN certificate:

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

1. Capsule Server’s private key, used to sign the certificate
2. The certificate request configuration file
3. Certificate Signing Request file

5. Send the certificate request to the Certificate Authority:
When you submit the request, specify the lifespan of the certificate. The method for sending the certificate request varies, so consult the Certificate Authority for the preferred method. In response to the request, you can expect to receive a Certificate Authority bundle and a signed certificate, in separate files.

6. Copy the Certificate Authority bundle and Capsule Server certificate file that you receive from the Certificate Authority, and Capsule Server private key to your Satellite Server.

7. On Satellite Server, validate Capsule Server certificate input files:

```
# 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. Capsule Server certificate file, provided by your Certificate Authority
2. Capsule Server’s private key that you used to sign the certificate
3. Certificate Authority bundle, provided by your Certificate Authority

If you set the `commonName` to a wildcard value `*`, you must add the `-t capsule` option to the `katello-certs-check` command.

Retain a copy of the example `capsule-certs-generate` command that is output by the `katello-certs-check` command for creating the Certificate Archive File for this Capsule Server.

### 4.3.2. Configuring Capsule Server with Custom SSL Certificates for Load Balancing without Puppet

The following section describes how to configure Capsule Servers that use custom SSL certificates for load balancing without Puppet. Complete this procedure on each Capsule Server that you want to configure for load balancing.

**Procedure**

1. Append the following option to the `capsule-certs-generate` command that you obtain from the output of the `katello-certs-check` command:

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

2. On Satellite Server, enter the `capsule-certs-generate` command to generate Capsule certificates:

```
# 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
```

Retain a copy of the example `satellite-installer` command from the output for installing Capsule Server certificates.
3. Copy the certificate archive file from Satellite Server to Capsule Server:

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

4. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

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

5. On Capsule Server, enter the `satellite-installer` command:

```
# 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. CONFIGURING CAPSULE SERVER WITH CUSTOM SSL CERTIFICATES FOR LOAD BALANCING WITH PUPPET

If you use Puppet in your Satellite configuration, then you must complete the following procedures:

1. Section 4.4.2, “Configuring Capsule Server with Custom SSL Certificates to Generate and Sign Puppet Certificates”

2. Section 4.4.3, “Configuring Remaining Capsule Servers with Custom SSL Certificates for Load Balancing”

### 4.4.1. Creating Custom SSL Certificates for Capsule Server

This procedure outlines how to create a configuration file for the Certificate Signing Request and include the load balancer and Capsule Server as Subject Alternative Names (SAN). Complete this procedure on each Capsule Server that you want to configure for load balancing.

**Procedure**

1. On Capsule Server, create a directory to contain all the source certificate files, accessible to only the `root` user:

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

2. Create a private key with which to sign the Certificate Signing Request (CSR). Note that the private key must be unencrypted. If you use a password-protected private key, remove the private key password.
If you already have a private key for this Capsule Server, skip this step.

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

3. Create the certificate request configuration file with the following content:

```ini
[ 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. The certificate’s common name must match the FQDN of Capsule Server. Ensure to change this when running the command on each Capsule Server that you configure for load balancing. You can also set a wildcard value `*`. If you set a wildcard value, you must add the `-t capsule` option when you use the `katello-certs-check` command.

2. Under [alt_names], include the FQDN of the load balancer as DNS.1 and the FQDN of Capsule Server as DNS.2.

4. Create a Certificate Signing Request (CSR) for the SAN certificate:

```bash
# 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. Capsule Server’s private key, used to sign the certificate

2. The certificate request configuration file

3. Certificate Signing Request file

5. Send the certificate request to the Certificate Authority:
When you submit the request, specify the lifespan of the certificate. The method for sending the certificate request varies, so consult the Certificate Authority for the preferred method. In response to the request, you can expect to receive a Certificate Authority bundle and a signed certificate, in separate files.

6. Copy the Certificate Authority bundle and Capsule Server certificate file that you receive from the Certificate Authority, and Capsule Server private key to your Satellite Server.

7. On Satellite Server, validate Capsule Server certificate input files:

```
# 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. Capsule Server certificate file, provided by your Certificate Authority
2. Capsule Server’s private key that you used to sign the certificate
3. Certificate Authority bundle, provided by your Certificate Authority

If you set the `commonName=` to a wildcard value `*`, you must add the `-t capsule` option to the `katello-certs-check` command.

Retain a copy of the example `capsule-certs-generate` command that is output by the `katello-certs-check` command for creating the Certificate Archive File for this Capsule Server.

### 4.4.2. Configuring Capsule Server with Custom SSL Certificates to Generate and Sign Puppet Certificates

Complete this procedure only for the system where you want to configure Capsule Server to generate Puppet certificates for all other Capsule Servers that you configure for load balancing.

**Procedure**

1. Append the following option to the `capsule-certs-generate` command that you obtain from the output of the `katello-certs-check` command:

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

2. On Satellite Server, enter the `capsule-certs-generate` command to generate Capsule certificates:

```
# 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
```

Retain a copy of the example `satellite-installer` command from the output for installing Capsule Server certificates.
3. Copy the certificate archive file from Satellite Server to Capsule Server.

4. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

   ```
   --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. On Capsule Server, enter the `satellite-installer` command:

   ```
   # 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. On Capsule Server, generate Puppet certificates for all other Capsules that you configure for load balancing, except this first system where you configure Puppet certificates signing:

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

   This command creates the following files on the Puppet certificate signing Capsule Server instance:

   - `/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. Configuring Remaining Capsule Servers with Custom SSL Certificates for Load Balancing

Complete this procedure for each Capsule Server excluding the system where you configure Capsule Server to sign Puppet certificates.
1. Append the following option to the `capsule-certs-generate` command that you obtain from the output of the `katello-certs-check` command:

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

2. On Satellite Server, enter the `capsule-certs-generate` command to generate Capsule certificates:

```
# 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
```

Retain a copy of the example `satellite-installer` command from the output for installing Capsule Server certificates.

3. Copy the certificate archive file from Satellite Server to Capsule Server.

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

4. On Capsule Server, install the `puppetserver` package:

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

5. On Capsule Server, create directories for puppet certificates:

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

6. On Capsule Server, copy the Puppet certificates for this Capsule Server from the system where you configure Capsule Server to sign Puppet certificates:

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

7. On Capsule Server, change the `/etc/puppetlabs/puppet/ssl/` directory ownership to user `puppet` and group `puppet`: 

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

8. On Capsule Server, set the SELinux context for the `/etc/puppetlabs/puppet/ssl/` directory:

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

9. Append the following options to the `satellite-installer` command that you obtain from the output of the `capsule-certs-generate` command:

    ```
    --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. On Capsule Server, enter the `satellite-installer` command:

    ```
    # 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"
    ```
CHAPTER 5. SETTING THE LOAD BALANCER FOR HOST REGISTRATION

You can configure Satellite to register clients through a load balancer when using the host registration feature.

You will be able to register hosts to the load balancer instead of Capsule. The load balancer will decide through which Capsule to register the host at the time of request. Upon registration, the subscription manager on the host will be configured to manage content through the load balancer.

Prerequisites

- You configured SSL certificates on all Capsule Servers. For more information, see Chapter 4, Configuring Capsule Servers for Load Balancing.
- You enabled Registration and Templates plug-ins on all Capsule Servers:

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

Procedure

1. On all Capsule Servers, set the registration and template URLs using satellite-installer:

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

2. In the Satellite web UI, navigate to Infrastructure > Capsules.

3. For each Capsule, click the dropdown menu in the Actions column and select Refresh.
CHAPTER 6. INSTALLING THE LOAD BALANCER

The following example provides general guidance for configuring an HAProxy load balancer using Red Hat Enterprise Linux 8 server. However, you can install any suitable load balancing software solution that supports TCP forwarding.

Procedure

1. Install HAProxy:
   ```bash
   # dnf install haproxy
   ```

2. Install the following package that includes the `semanage` tool:
   ```bash
   # dnf install policycoreutils-python-utils
   ```

3. Configure SELinux to allow HAProxy to bind any port:
   ```bash
   # semanage boolean --modify --on haproxy_connect_any
   ```

4. Configure the load balancer to balance the network load for the ports as described in Table 6.1, “Ports Configuration for the Load Balancer”. For example, to configure ports for HAProxy, edit the `/etc/haproxy/haproxy.cfg` file to correspond with the table.

<table>
<thead>
<tr>
<th>Service</th>
<th>Port</th>
<th>Mode</th>
<th>Balance Mode</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>80</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 80 on all Capsule Servers</td>
</tr>
<tr>
<td>HTTPS and RHSM</td>
<td>443</td>
<td>TCP</td>
<td>source</td>
<td>port 443 on all Capsule Servers</td>
</tr>
<tr>
<td>AMQP</td>
<td>5647</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 5647 on all Capsule Servers</td>
</tr>
<tr>
<td>Anaconda for template retrieval</td>
<td>8000</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 8000 on all Capsule Servers</td>
</tr>
<tr>
<td>Puppet (Optional)</td>
<td>8140</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 8140 on all Capsule Servers</td>
</tr>
<tr>
<td>PuppetCA (Optional)</td>
<td>8141</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 8140 only on the system where you configure Capsule Server to sign Puppet certificates</td>
</tr>
<tr>
<td>Capsule HTTPS for Host Registration and optionally OpenSCAP</td>
<td>9090</td>
<td>TCP</td>
<td>roundrobin</td>
<td>port 9090 on all Capsule Servers</td>
</tr>
</tbody>
</table>
5. Configure the load balancer to disable SSL offloading and allow client-side SSL certificates to pass through to back end servers. This is required because communication from clients to Capsule Servers depends on client-side SSL certificates.

6. Start and enable the HAProxy service:

```bash
# systemctl enable --now haproxy
```
CHAPTER 7. VERIFYING THE LOAD BALANCING CONFIGURATION

Use this procedure to verify the load balancing configuration for each Capsule Server.

Procedure

1. Shut down the base operating system for your Capsule Server.

2. Verify that content or subscription management features are available on clients registered to this Capsule. For example, enter the `subscription-manager refresh` command on a client.

3. Restart the base operating system for your Capsule Server.
CHAPTER 8. REGISTERING CLIENTS TO THE LOAD BALANCER

To balance the load of network traffic from clients, you must register the clients to the load balancer.

To register clients, proceed with one of the following procedures:

- Section 8.1, “Registering Clients Using Host Registration”
- Section 8.2, “(Deprecated) Registering Clients Using the Bootstrap Script”
- Section 8.3, “(Deprecated) Registering Clients Manually Using katello-ca-consumer RPM”

8.1. REGISTERING CLIENTS USING HOST REGISTRATION

You can register hosts with Satellite using the host registration feature in the Satellite web UI, Hammer CLI, or the Satellite API. For more information, see Registering Hosts in Managing Hosts.

Procedure

1. In the Satellite web UI, navigate to Hosts > Register Host.
2. From the Capsule dropdown list, select the load balancer.
3. Select Force to register a host that has been previously registered to a Capsule Server.
4. From the Activation Keys list, select the activation keys to assign to your host.
5. Click Generate to create the registration command.
6. Click on the files icon to copy the command to your clipboard.
7. Connect to your host using SSH and run the registration command.
8. Check the /etc/yum.repos.d/redhat.repo file and ensure that the appropriate repositories have been enabled.

CLI procedure

1. Generate the host registration command using the Hammer CLI:

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

   If your hosts do not trust the SSL certificate of Satellite Server, you can disable SSL validation by adding the --insecure flag to the registration command.

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

   Include the --smart-proxy-id My_Capsule_ID option. You can use the ID of any Capsule Server that you configured for host registration load balancing. Satellite will apply the load balancer to the registration command automatically.
Include the --force option to register a host that has been previously registered to a Capsule Server.

2. Connect to your host using SSH and run the registration command.

3. Check the /etc/yum.repos.d/redhat.repo file and ensure that the appropriate repositories have been enabled.

API procedure

1. Generate the host registration command using the Satellite API:

   ```bash
   # 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"] } }'
   
   If your hosts do not trust the SSL certificate of Satellite Server, you can disable SSL validation by adding the --insecure flag to the registration command.

   ```bash
   # 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 } }'
   ```

   Use an activation key to simplify specifying the environments. For more information, see Managing Activation Keys in Managing Content.

   Include { "smart_proxy_id": My_Capsule_ID } You can use the ID of any Capsule Server that you configured for host registration load balancing. Satellite will apply the load balancer to the registration command automatically.

   Include { "force": true } to register a host that has been previously registered to a Capsule Server.

   To enter a password as a command line argument, use username:password syntax. Keep in mind this can save the password in the shell history. Alternatively, you can use a temporary personal access token instead of a password. To generate a token in the Satellite web UI, navigate to My Account > Personal Access Tokens.

2. Connect to your host using SSH and run the registration command.

3. Check the /etc/yum.repos.d/redhat.repo file and ensure that the appropriate repositories have been enabled.

8.2. (DEPRECATED) REGISTERING CLIENTS USING THE BOOTSTRAP SCRIPT

To register clients, enter the following command on the client. You must complete the registration procedure for each client.

Prerequisite
• Ensure that you install the bootstrap script on the client and change file permissions of the script to executable. For more information, see Registering Hosts to Red Hat Satellite Using The Bootstrap Script in Managing Hosts.

Procedure

• On Red Hat Enterprise Linux 8, enter the following command:

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

1 Replace `<arch>` with the client architecture, for example `x86`.
2 Include the `--force` option to register the client that has been previously registered to a standalone Capsule.
3 Include the `--puppet-ca-port 8141` option if you use Puppet.

• On Red Hat Enterprise Linux 7 or 6, enter the following command:

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

1 Include the `--force` option to register the client that has been previously registered to a standalone Capsule.
2 Include the `--puppet-ca-port 8141` option if you use Puppet.

The script prompts for the password corresponding to the Satellite user name you entered with the `--login` option.

8.3. (DEPRECATED) REGISTERING CLIENTS MANUALLY USING KATELLO-CA-CONSUMER RPM

To register clients manually, complete the following procedure on each client that you want to register.

Procedure
1. Remove the `katello-ca-consumer` package if it is installed:

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

2. Install the `katello-ca-consumer` package from the load balancer:

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

3. Register the client and include the `--serverurl` and `--baseurl` options:

   ```
   # subscription-manager register \
   --activationkey="My_Activation_Key" \
   --baseurl=https://loadbalancer.example.com/pulp/content/ \
   --org="My_Organization" \
   --serverurl=https://loadbalancer.example.com/rhsm
   ```
CHAPTER 9. PROPAGATING SCAP CONTENT THROUGH THE LOAD BALANCER

If you use OpenSCAP to manage security compliance on your clients, you must configure the SCAP client to send ARF reports to the load balancer instead of Capsule. The configuration procedure depends on the method you have selected to deploy compliance policies.

9.1. PROPAGATING SCAP CONTENT USING ANSIBLE DEPLOYMENT

Using this procedure, you can promote Security Content Automation Protocol (SCAP) content through the load balancer in the scope of the Ansible deployment method.

Prerequisite

- Ensure that you have configured Satellite for Ansible deployment of compliance policies. For more information, see Configuring Compliance Policy Deployment Methods in Managing Security Compliance.

Procedure

1. In the Satellite web UI, navigate to Configure > Variables.
2. Search for the `foreman_scap_client_port` variable and click its name.
3. In the Default Behavior area, ensure that the Override checkbox is selected.
4. In the Parameter Type list, ensure that integer is selected.
5. In the Default Value field, enter 9090.
6. In the Specify Matchers area, remove all matchers that override the default value.
7. Click Submit.
8. Search for the `foreman_scap_client_server` variable and click its name.
9. In the Default Behavior area, ensure that the Override checkbox is selected.
10. In the Parameter Type list, ensure that string is selected.
11. In the Default Value field, enter the FQDN of your load balancer, such as `loadbalancer.example.com`.
12. In the Specify Matchers area, remove all matchers that override the default value.
13. Click Submit.
14. Continue with deploying a compliance policy using Ansible. For more information, see:
   - Deploying a Policy in a Host Group Using Ansible in Managing Security Compliance
   - Deploying a Policy on a Host Using Ansible in Managing Security Compliance

Verification
On the client, verify that the `/etc/foreman_scap_client/config.yaml` file contains the following lines:

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

### 9.2. PROPAGATING SCAP CONTENT USING PUPPET DEPLOYMENT

Using this procedure, you can promote Security Content Automation Protocol (SCAP) content through the load balancer in the scope of the Puppet deployment method.

**Prerequisite**

- Ensure that you have configured Satellite for Puppet deployment of compliance policies. For more information, see Configuring Compliance Policy Deployment Methods in Managing Security Compliance.

**Procedure**

1. In the Satellite web UI, navigate to Configure > Classes and click `foreman_scap_client`.
2. Click the Smart Class Parameter tab.
3. In the pane to the left of the Smart Class Parameter window, click `port`.
4. In the Default Behavior area, select the Override checkbox.
5. From the Key Type list, select integer.
6. In the Default Value field, enter 9090.
7. In the pane to the left of the Smart Class Parameter window, click `server`.
8. In the Default Behavior area, select the Override checkbox.
9. From the Key Type list, select string.
10. In the Default Value field, enter the FQDN of your load balancer, such as `loadbalancer.example.com`.
11. In the lower left of the Smart Class Parameter window, click Submit.
12. Continue with deploying a compliance policy using Puppet. For more information, see:
   - Deploying a Policy in a Host Group Using Puppet in Managing Security Compliance
   - Deploying a Policy on a Host Using Puppet in Managing Security Compliance

**Verification**

- On the client, verify that the `/etc/foreman_scap_client/config.yaml` file contains the following lines:
# Foreman proxy to which reports should be uploaded
:server: 'loadbalancer.example.com'
:port: 9090