Red Hat Satellite 6.10

Provisioning Guide

A guide to provisioning physical and virtual hosts on Red Hat Satellite Servers.

Last Updated: 2023-04-24

Red Hat Satellite 6.10 Provisioning Guide

A guide to provisioning physical and virtual hosts on Red Hat Satellite Servers.

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

Legal Notice

Abstract

The Red Hat Satellite Provisioning Guide has instructions on provisioning physical and virtual hosts. This includes setting up the required network topology, configuring the necessary services, and providing all of the other configuration information to provision hosts on your network.

Table of Contents

CHAPTER 1. INTRODUCTION

1.1. PROVISIONING OVERVIEW

Provisioning is a process that starts with a bare physical or virtual machine and ends with a fully configured, ready-to-use operating system. Using Red Hat Satellite, you can define and automate fine-grained provisioning for a large number of hosts.

There are many provisioning methods. For example, you can use Satellite Server's integrated Capsule or an external Capsule Server to provision bare metal hosts using PXE based methods. You can also provision cloud instances from specific providers through their APIs. These provisioning methods are part of the Red Hat Satellite application life cycle to create, manage, and update hosts.

Red Hat Satellite has different methods for provisioning hosts:

Bare Metal Provisioning

Satellite provisions bare metal hosts primarily through PXE boot and MAC address identification. You can create host entries and specify the MAC address of the physical host to provision. You can also boot blank hosts to use Satellite's discovery service, which creates a pool of ready-to-provision hosts.

Cloud Providers

Satellite connects to private and public cloud providers to provision instances of hosts from images that are stored with the Cloud environment. This also includes selecting which hardware profile or flavor to use.

Virtualization Infrastructure

Satellite connects to virtualization infrastructure services such as Red Hat Virtualization and VMware to provision virtual machines from virtual image templates or using the same PXE-based boot methods as bare metal providers.

1.2. NETWORK BOOT PROVISIONING WORKFLOW

PXE booting assumes that a host, either physical or virtual, is configured to boot from network as the first booting device, and from the hard drive as the second booting device.

The provisioning process follows a basic PXE workflow:

1. You create a host and select a domain and subnet. Satellite requests an available IP address from the DHCP Capsule Server that is associated with the subnet or from the PostgreSQL database in Satellite. Satellite loads this IP address into the **IP address** field in the Create Host window. When you complete all the options for the new host, submit the new host request.

2. Depending on the configuration specifications of the host and its domain and subnet, Satellite creates the following settings:

    - A DHCP record on Capsule Server that is associated with the subnet.
    - A forward DNS record on Capsule Server that is associated with the domain.
    - A reverse DNS record on the DNS Capsule Server that is associated with the subnet.
    - PXELinux, Grub, Grub2, and iPXE configuration files for the host in the TFTP Capsule Server that is associated with the subnet.
    - A Puppet certificate on the associated Puppet server.

- A realm on the associated identity server.

3. The new host requests a DHCP reservation from the DHCP server.
4. The DHCP server responds to the reservation request and returns TFTP **next-server** and **filename** options.
5. The host requests the boot loader and menu from the TFTP server according to the PXELoader setting.
6. A boot loader is returned over TFTP.
7. The boot loader fetches configuration for the host through its provisioning interface MAC address.
8. The boot loader fetches the operating system installer kernel, init RAM disk, and boot parameters.
9. The installer requests the provisioning template from Satellite.
10. Satellite renders the provision template and returns the result to the host.
11. The installer performs installation of the operating system.
    - The installer registers the host to Satellite using Red Hat Subscription Manager.
    - The installer installs management tools such as **katello-agent** and **puppet**.
    - The installer notifies Satellite of a successful build in the **postinstall** script.
12. The PXE configuration files revert to a local boot template.
13. The host reboots.
14. The new host requests a DHCP reservation from the DHCP server.
15. The DHCP server responds to the reservation request and returns TFTP **next-server** and **filename** options.
16. The host requests the bootloader and menu from the TFTP server according to the PXELoader setting.
17. A boot loader is returned over TFTP.
18. The boot loader fetches the configuration for the host through its provision interface MAC address.
19. The boot loader initiates boot from the local drive.
20. If you configured the host to use any Puppet classes, the host configures itself using the modules.

This workflow differs depending on custom options. For example:

Discovery

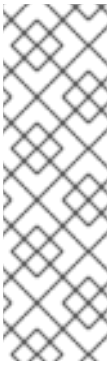
If you use the discovery service, Satellite automatically detects the MAC address of the new host and restarts the host after you submit a request. Note that TCP port 8443 must be reachable by the Capsule to which the host is attached for Satellite to restart the host.

PXE-less Provisioning

After you submit a new host request, you must boot the specific host with the boot disk that you download from Satellite and transfer using a USB port of the host.

Compute Resources

Satellite creates the virtual machine and retrieves the MAC address and stores the MAC address in Satellite. If you use image-based provisioning, the host does not follow the standard PXE boot and operating system installation. The compute resource creates a copy of the image for the host to use. Depending on image settings in Satellite, seed data can be passed in for initial configuration, for example using **cloud-init**. Satellite can connect using SSH to the host and execute a template to finish the customization.



NOTE

By default, deleting the provisioned profile host from Satellite does not destroy the actual VM on the external compute resource. To destroy the VM when deleting the host entry on Satellite, navigate to **Administer** > **Settings** > **Provisioning** and configure this behavior using the **destroy_vm_on_host_delete** setting. If you do not destroy the associated VM and attempt to create a new VM with the same resource name later, it will fail because that VM name already exists in the external compute resource. You can still register the existing VM to Satellite using the standard host registration workflow you would use for any already provisioned host.

CHAPTER 2. CONFIGURING PROVISIONING RESOURCES

2.1. PROVISIONING CONTEXTS

A provisioning context is the combination of an organization and location that you specify for Satellite components. The organization and location that a component belongs to sets the ownership and access for that component.

Organizations divide Red Hat Satellite 6 components into logical groups based on ownership, purpose, content, security level, and other divisions. You can create and manage multiple organizations through Red Hat Satellite 6 and assign components to each individual organization. This ensures Satellite Server provisions hosts within a certain organization and only uses components that are assigned to that organization. For more information about organizations, see Managing Organizations in the *Content Management Guide*.

Locations function similar to organizations. The difference is that locations are based on physical or geographical setting. Users can nest locations in a hierarchy. For more information about locations, see Managing Locations in the *Content Management Guide*.

2.2. SETTING THE PROVISIONING CONTEXT

When you set a provisioning context, you define which organization and location to use for provisioning hosts.

The organization and location menus are located in the menu bar, on the upper left of the Satellite web UI. If you have not selected an organization and location to use, the menu displays: **Any Organization** and **Any Location**.

Procedure

1. Click **Any Organization** and select the organization.
2. Click **Any Location** and select the location to use.

Each user can set their default provisioning context in their account settings. Click the user name in the upper right of the Satellite web UI and select **My account** to edit your user account settings.

CLI procedure

- When using the CLI, include either **--organization** or **--organization-label** and **--location** or **--location-id** as an option. For example:

```
# hammer host list --organization "Default_Organization" --location "Default_Location"
```

This command outputs hosts allocated for the Default_Organization and Default_Location.

2.3. CREATING OPERATING SYSTEMS

An operating system is a collection of resources that define how Satellite Server installs a base operating system on a host. Operating system entries combine previously defined resources, such as installation media, partition tables, provisioning templates, and others.

Importing operating systems from Red Hat's CDN creates new entries on the **Hosts** > **Operating Systems** page.

You can also add custom operating systems using the following procedure. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Operating systems** and click **New Operating system**.
2. In the **Name** field, enter a name to represent the operating system entry.
3. In the **Major** field, enter the number that corresponds to the major version of the operating system.
4. In the **Minor** field, enter the number that corresponds to the minor version of the operating system.
5. In the **Description** field, enter a description of the operating system.
6. From the **Family** list, select the operating system's family.
7. From the **Root Password Hash** list, select the encoding method for the root password.
8. From the **Architectures** list, select the architectures that the operating system uses.
9. Click the **Partition table** tab and select the possible partition tables that apply to this operating system.
10. Optional: if you use non-Red Hat content, click the Installation Media tab and select the installation media that apply to this operating system. For more information, see Adding Installation Media to Satellite.
11. Click the **Templates** tab and select a **PXELinux template**, a **Provisioning template**, and a **Finish template** for your operating system to use. You can select other templates, for example an **iPXE template**, if you plan to use iPXE for provisioning.
12. Click **Submit** to save your provisioning template.

CLI procedure

- Create the operating system using the **hammer os create** command:

```
# hammer os create --name "MyOS" \
--description "My_custom_operating_system" \
--major 7 --minor 3 --family "Redhat" --architectures "x86_64" \
--partition-tables "My_Partition" --media "Red_Hat" \
--provisioning-templates "My_Provisioning_Template"
```

2.4. UPDATING THE DETAILS OF MULTIPLE OPERATING SYSTEMS

Use this procedure to update the details of multiple operating systems. This example shows you how to assign each operating system a partition table called **Kickstart default**, a configuration template called **Kickstart default PXELinux**, and a provisioning template called **Kickstart Default**.

Procedure

1. On Satellite Server, run the following Bash script:

```
PARTID=$(hammer --csv partition-table list | grep "Kickstart default," | cut -d, -f1)
PXEID=$(hammer --csv template list --per-page=1000 | grep "Kickstart default PXELinux" |
cut -d, -f1)
SATID=$(hammer --csv template list --per-page=1000 | grep "provision" | grep ",Kickstart
default" | cut -d, -f1)

for i in $(hammer --no-headers --csv os list | awk -F, {'print $1'})
do
  hammer partition-table add-operatingsystem --id="${PARTID}" --operatingsystem-id="${i}"
  hammer template add-operatingsystem --id="${PXEID}" --operatingsystem-id="${i}"
  hammer os set-default-template --id="${i}" --config-template-id=${PXEID}
  hammer os add-config-template --id="${i}" --config-template-id=${SATID}
  hammer os set-default-template --id="${i}" --config-template-id=${SATID}
done
```

2. Display information about the updated operating system to verify that the operating system is updated correctly:

```
# hammer os info --id 1
```

2.5. CREATING ARCHITECTURES

An architecture in Satellite represents a logical grouping of hosts and operating systems. Architectures are created by Satellite automatically when hosts check in with Puppet. Basic i386 and x86_64 architectures are already preset in Satellite.

Use this procedure to create an architecture in Satellite.

Supported Architectures

Only Intel x86_64 architecture is supported for provisioning using PXE, Discovery, and boot disk. For more information, see the Red Hat Knowledgebase solution Supported architectures and provisioning scenarios in Satellite 6.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Architectures** and click **Create Architecture**.
2. In the **Name** field, enter a name for the architecture.
3. From the **Operating Systems** list, select an operating system. If none are available, you can create and assign them under **Hosts** > **Operating Systems**.
4. Click **Submit**.

CLI procedure

- Enter the **hammer architecture create** command to create an architecture. Specify its name and operating systems that include this architecture:

```
# hammer architecture create --name "Architecture_Name" \
--operatingsystems "os"
```

2.6. CREATING HARDWARE MODELS

Use this procedure to create a hardware model in Satellite so that you can specify which hardware model a host uses.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Hardware Models** and click **Create Model**.
2. In the **Name** field, enter a name for the hardware model.
3. Optionally, in the **Hardware Model** and **Vendor Class** fields, you can enter corresponding information for your system.
4. In the **Info** field, enter a description of the hardware model.
5. Click **Submit** to save your hardware model.

CLI procedure

- Create a hardware model using the **hammer model create** command. The only required parameter is **--name**. Optionally, enter the hardware model with the **--hardware-model** option, a vendor class with the **--vendor-class** option, and a description with the **--info** option:

```
# hammer model create --name "model_name" --info "description" \
--hardware-model "hardware_model" --vendor-class "vendor_class"
```

2.7. USING A SYNCED KICKSTART REPOSITORY FOR A HOST'S OPERATING SYSTEM

Satellite contains a set of synchronized kickstart repositories that you use to install the provisioned host's operating system. For more information about adding repositories, see Syncing Repositories in the *Content Management Guide*.

Use this procedure to set up a kickstart repository.

Prerequisites

You must enable both **BaseOS** and **Appstream Kickstart** before provisioning.

Procedure

1. Add the synchronized kickstart repository that you want to use to the existing Content View, or create a new Content View and add the kickstart repository.
   For Red Hat Enterprise Linux 8, ensure that you add both **Red Hat Enterprise Linux 8 for x86_64 - AppStream Kickstart x86_64 8** and **Red Hat Enterprise Linux 8 for x86_64 - BaseOS Kickstart x86_64 8** repositories.

   If you use a disconnected environment, you must import the Kickstart repositories from a Red Hat Enterprise Linux binary DVD. For more information, see Importing Kickstart Repositories in the *Content Management Guide*.
2. Publish a new version of the Content View where the kickstart repository is added and promote it to a required lifecycle environment. For more information, see Managing Content Views in the *Content Management Guide*.

3. When you create a host, in the **Operating System** tab, for **Media Selection**, select the **Synced Content** check box.

To view the kickstart tree, enter the following command:

```
# hammer medium list --organization "your_organization"
```

2.8. ADDING INSTALLATION MEDIA TO SATELLITE

Installation media are sources of packages that Satellite Server uses to install a base operating system on a machine from an external repository. You can use this parameter to install third-party content. Red Hat content is delivered through repository syncing instead.

Installation media must be in the format of an operating system installation tree, and must be accessible to the machine hosting the installer through an HTTP URL. You can view installation media by navigating to **Hosts** > **Installation Media** menu.

By default, Satellite includes installation media for some official Linux distributions. Note that some of those installation media are targeted for a specific version of an operating system. For example **CentOS mirror (7.x)** must be used for CentOS 7 or earlier, and **CentOS mirror (8.x)** must be used for CentOS 8 or later.

If you want to improve download performance when using installation media to install operating systems on multiple host, you must modify the installation medium's **Path** to point to the closest mirror or a local copy.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Installation Media** and click **Create Medium**.
2. In the **Name** field, enter a name to represent the installation media entry.
3. In the **Path** enter the URL or NFS share that contains the installation tree. You can use following variables in the path to represent multiple different system architectures and versions:
   - **$arch** - The system architecture.
   - **$version** - The operating system version.
   - **$major** - The operating system major version.
   - **$minor** - The operating system minor version.
     Example HTTP path:

     ```
     http://download.example.com/centos/$version/Server/$arch/os/
     ```

     Example NFS path:

     ```
     nfs://download.example.com:/centos/$version/Server/$arch/os/
     ```

     Synchronized content on Capsule Servers always uses an HTTP path. Capsule Server managed content does not support NFS paths.

4. From the **Operating system family** list, select the distribution or family of the installation medium. For example, CentOS and Fedora are in the **Red Hat** family.

5. Click the **Organizations** and **Locations** tabs, to change the provisioning context. Satellite Server adds the installation medium to the set provisioning context.

6. Click **Submit** to save your installation medium.

CLI procedure

- Create the installation medium using the **hammer medium create** command:

```
# hammer medium create --name "CustomOS" --os-family "Redhat" \
--path 'http://download.example.com/centos/$version/Server/$arch/os/' \
--organizations "My_Organization" --locations "My_Location"
```

2.9. CREATING PARTITION TABLES

A partition table is a type of template that defines the way Satellite Server configures the disks available on a new host. A Partition table uses the same ERB syntax as provisioning templates. Red Hat Satellite contains a set of default partition tables to use, including a **Kickstart default**. You can also edit partition table entries to configure the preferred partitioning scheme, or create a partition table entry and add it to the operating system entry.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Partition Tables** and click **Create Partition Table**.

2. In the **Name** field, enter a name for the partition table.

3. Select the **Default** check box if you want to set the template to automatically associate with new organizations or locations.

4. Select the **Snippet** check box if you want to identify the template as a reusable snippet for other partition tables.

5. From the **Operating System Family** list, select the distribution or family of the partitioning layout. For example, Red Hat Enterprise Linux, CentOS, and Fedora are in the Red Hat family.

6. In the **Template editor** field, enter the layout for the disk partition. For example:

```
zerombr
clearpart --all --initlabel
autopart
```

   You can also use the **Template** file browser to upload a template file.

   The format of the layout must match that for the intended operating system. For example, Red Hat Enterprise Linux 7.2 requires a layout that matches a kickstart file.

7. In the **Audit Comment** field, add a summary of changes to the partition layout.

8. Click the **Organizations** and **Locations** tabs to add any other provisioning contexts that you want to associate with the partition table. Satellite adds the partition table to the current provisioning context.

9. Click **Submit** to save your partition table.

CLI procedure

1. Before you create a partition table with the CLI, create a plain text file that contains the partition layout. This example uses the ~/**my-partition** file.

2. Create the installation medium using the **hammer partition-table create** command:

```
# hammer partition-table create --name "My Partition" --snippet false \
--os-family Redhat --file ~/my-partition --organizations "My_Organization" \
--locations "My_Location"
```

2.10. DYNAMIC PARTITION EXAMPLE

Using an Anaconda kickstart template, the following section instructs Anaconda to erase the whole disk, automatically partition, enlarge one partition to maximum size, and then proceed to the next sequence of events in the provisioning process:

```
zerombr
clearpart --all --initlabel
autopart <%= host_param('autopart_options') %>
```

Dynamic partitioning is executed by the installation program. Therefore, you can write your own rules to specify how you want to partition disks according to runtime information from the node, for example, disk sizes, number of drives, vendor, or manufacturer.

If you want to provision servers and use dynamic partitioning, add the following example as a template. When the **#Dynamic** entry is included, the content of the template loads into a **%pre** shell scriplet and creates a **/tmp/diskpart.cfg** that is then included into the Kickstart partitioning section.

```
#Dynamic (do not remove this line)

MEMORY=$((`grep MemTotal: /proc/meminfo | sed 's/^MemTotal: *//'|sed 's/ .*//'` / 1024))
if [ "$MEMORY" -lt 2048 ]; then
   SWAP_MEMORY=$(($MEMORY * 2))
elif [ "$MEMORY" -lt 8192 ]; then
   SWAP_MEMORY=$MEMORY
elif [ "$MEMORY" -lt 65536 ]; then
   SWAP_MEMORY=$(($MEMORY / 2))
else
   SWAP_MEMORY=32768
fi

cat <<EOF > /tmp/diskpart.cfg
zerombr yes
clearpart --all --initlabel
part /boot --fstype ext4 --size 200 --asprimary
part swap --size "$SWAP_MEMORY"
part / --fstype ext4 --size 1024 --grow
EOF
```

2.11. PROVISIONING TEMPLATES

A provisioning template defines the way Satellite Server installs an operating system on a host.

Red Hat Satellite includes many template examples. In the Satellite web UI, navigate to **Hosts** > **Provisioning templates** to view them. You can create a template or clone a template and edit the clone. For help with templates, navigate to **Hosts** > **Provisioning templates** > **Create Template** > **Help**.

Templates accept the Embedded Ruby (ERB) syntax. For more information, see Template Writing Reference in *Managing Hosts*.

You can download provisioning templates. Before you can download the template, you must create a debug certificate. For more information, see Creating an Organization Debug Certificate in the *Content Management Guide*.

You can synchronize templates between Satellite Server and a Git repository or a local directory. For more information, see Synchronizing Templates Repositories in the *Managing Hosts* guide.

To view the history of changes applied to a template, navigate to **Hosts** > **Provisioning templates**, select one of the templates, and click **History**. Click **Revert** to override the content with the previous version. You can also revert to an earlier change. Click **Show Diff** to see information about a specific change:

- The **Template Diff** tab displays changes in the body of a provisioning template.
- The **Details** tab displays changes in the template description.
- The **History** tab displays the user who made a change to the template and date of the change.

2.12. TYPES OF PROVISIONING TEMPLATES

There are various types of provisioning templates:

Provision

The main template for the provisioning process. For example, a kickstart template. For more information about kickstart template syntax, see the Kickstart Syntax Reference in the *Red Hat Enterprise Linux 7 Installation Guide*.

PXELinux, PXEGrub, PXEGrub2

PXE-based templates that deploy to the template Capsule associated with a subnet to ensure that the host uses the installer with the correct kernel options. For BIOS provisioning, select **PXELinux** template. For UEFI provisioning, select **PXEGrub2**.

Finish

Post-configuration scripts to execute using an SSH connection when the main provisioning process completes. You can use Finishing templates only for imaged-based provisioning in virtual or cloud environments that do not support user_data. Do not confuse an image with a foreman discovery ISO, which is sometimes called a Foreman discovery image. An image in this context is an install image in a virtualized environment for easy deployment.
When a finish script successfully exits with the return code **0**, Red Hat Satellite treats the code as a success and the host exits the build mode. Note that there are a few finish scripts with a build mode that uses a *call back* HTTP call. These scripts are not used for image-based provisioning, but for post configuration of operating-system installations such as Debian, Ubuntu, and BSD.

user_data

Post-configuration scripts for providers that accept custom data, also known as seed data. You can use the user_data template to provision virtual machines in cloud or virtualised environments only. This template does not require Satellite to be able to reach the host; the cloud or virtualization platform is responsible for delivering the data to the image.
Ensure that the image that you want to provision has the software to read the data installed and set to start during boot. For example, **cloud-init**, which expects YAML input, or **ignition**, which expects JSON input.

cloud_init

Some environments, such as VMWare, either do not support custom data or have their own data format that limits what can be done during customization. In this case, you can configure a cloud-init client with the **foreman** plug-in, which attempts to download the template directly from Satellite over HTTP or HTTPS. This technique can be used in any environment, preferably virtualized.
Ensure that you meet the following requirements to use the **cloud_init** template:

- Ensure that the image that you want to provision has the software to read the data installed and set to start during boot.
- A provisioned host is able to reach Satellite from the IP address that matches the host's provisioning interface IP.
  Note that cloud-init does not work behind NAT.

Bootdisk

Templates for PXE-less boot methods.

Kernel Execution (kexec)

Kernel execution templates for PXE-less boot methods.

NOTE

Kernel Execution is a Technology Preview feature. Technology Preview features are not fully supported under Red Hat Subscription Service Level Agreements (SLAs), may not be functionally complete, and are not intended for production use. However, these features provide early access to upcoming product innovations, enabling customers to test functionality and provide feedback during the development process.

Script

An arbitrary script not used by default but useful for custom tasks.

ZTP

Zero Touch Provisioning templates.

POAP

PowerOn Auto Provisioning templates.

iPXE

Templates for **iPXE** or **gPXE** environments to use instead of PXELinux.

2.13. CREATING PROVISIONING TEMPLATES

A provisioning template defines the way Satellite Server installs an operating system on a host. Use this procedure to create a new provisioning template.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates** and click **Create Template**.
2. In the **Name** field, enter a name for the provisioning template.
3. Fill in the rest of the fields as required. The **Help** tab provides information about the template syntax and details the available functions, variables, and methods that can be called on different types of objects within the template.

CLI procedure

1. Before you create a template with the CLI, create a plain text file that contains the template. This example uses the **~/my-template** file.
2. Create the template using the **hammer template create** command and specify the type with the **--type** option:

```
# hammer template create --name "My Provisioning Template" \
--file ~/my-template --type provision --organizations "My_Organization" \
--locations "My_Location"
```

2.14. CLONING PROVISIONING TEMPLATES

A provisioning template defines the way Satellite Server installs an operating system on a host. Use this procedure to clone a template and add your updates to the clone.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates** and search for the template that you want to use.
2. Click **Clone** to duplicate the template.
3. In the **Name** field, enter a name for the provisioning template.
4. Select the **Default** check box to set the template to associate automatically with new organizations or locations.
5. In the **Template editor** field, enter the body of the provisioning template. You can also use the **Template** file browser to upload a template file.
6. In the **Audit Comment** field, enter a summary of changes to the provisioning template for auditing purposes.
7. Click the **Type** tab and if your template is a snippet, select the **Snippet** check box. A snippet is not a standalone provisioning template, but a part of a provisioning template that can be inserted into other provisioning templates.
8. From the **Type** list, select the type of the template. For example, **Provisioning template**.

9. Click the **Association** tab and from the **Applicable Operating Systems** list, select the names of the operating systems that you want to associate with the provisioning template.
10. Optionally, click **Add combination** and select a host group from the **Host Group** list or an environment from the **Environment** list to associate provisioning template with the host groups and environments.
11. Click the **Organizations** and **Locations** tabs to add any additional contexts to the template.
12. Click **Submit** to save your provisioning template.

2.15. CREATING COMPUTE PROFILES

You can use compute profiles to predefine virtual machine hardware details such as CPUs, memory, and storage. A default installation of Red Hat Satellite contains three predefined profiles:

- **1-Small**
- **2-Medium**
- **3-Large**

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles** and click **Create Compute Profile**.
2. In the **Name** field, enter a name for the profile.
3. Click **Submit**. A new window opens with the name of the compute profile.
4. In the new window, click the name of each compute resource and edit the attributes you want to set for this compute profile.

CLI procedure

The compute profile CLI commands are not yet implemented in Red Hat Satellite 6.10.

2.16. SETTING A DEFAULT ENCRYPTED ROOT PASSWORD FOR HOSTS

If you do not want to set a plain text default root password for the hosts that you provision, you can use a default encrypted password.

Procedure

1. Generate an encrypted password:

```
# python -c 'import crypt,getpass;pw=getpass.getpass(); print(crypt.crypt(pw)) if
(pw==getpass.getpass("Confirm: ")) else exit()'
```

2. Copy the password for later use.
3. In the Satellite web UI, navigate to **Administer** > **Settings**.

4. On the **Settings** page, select the **Provisioning** tab.
5. In the **Name** column, navigate to **Root password**, and click **Click to edit**
6. Paste the encrypted password, and click **Save**.

2.17. USING NOVNC TO ACCESS VIRTUAL MACHINES

You can use your browser to access the VNC console of VMs created by Satellite.

Satellite supports using noVNC on the following virtualization platforms:

- VMware
- Libvirt
- Red Hat Virtualization

Prerequisites

- You must have a virtual machine created by Satellite.
- For existing virtual machines, ensure that the **Display type** in the **Compute Resource** settings is **VNC**.
- You must import the Katello root CA certificate into your Satellite Server. Adding a security exception in the browser is not enough for using noVNC. For more information, see the Installing the Katello Root CA Certificate section in the *Administering Red Hat Satellite* guide.

Procedure

1. On the VM host system, configure the firewall to allow VNC service on ports 5900 to 5930:
   - On Red Hat Enterprise Linux 6:

     ```
     # iptables -A INPUT -p tcp --dport 5900:5930 -j ACCEPT
     # service iptables save
     ```

   - On Red Hat Enterprise Linux 7:

     ```
     # firewall-cmd --add-port=5900-5930/tcp
     # firewall-cmd --add-port=5900-5930/tcp --permanent
     ```

2. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and select the name of a compute resource.
3. In the **Virtual Machines** tab, select the name of a VM host. Ensure the machine is powered on and then select **Console**.

CHAPTER 3. CONFIGURING NETWORKING

Each provisioning type requires some network configuration. Use this chapter to configure network services in your integrated Capsule on Satellite Server.

New hosts must have access to your Capsule Server. Capsule Server can be either your integrated Capsule on Satellite Server or an external Capsule Server. You might want to provision hosts from an external Capsule Server when the hosts are on isolated networks and cannot connect to Satellite Server directly, or when the content is synchronized with Capsule Server. Provisioning using the external Capsule Server can save on network bandwidth.

Configuring Capsule Server has two basic requirements:

1. Configuring network services. This includes:
    - Content delivery services
    - Network services (DHCP, DNS, and TFTP)
    - Puppet configuration
2. Defining network resource data in Satellite Server to help configure network interfaces on new hosts.

The following instructions have similar applications to configuring standalone Capsules managing a specific network. To configure Satellite to use external DHCP, DNS, and TFTP services, see Configuring External Services in *Installing Satellite Server from a Connected Network* .

3.1. NETWORK RESOURCES

Satellite contains networking resources that you must set up and configure to create a host. Satellite includes the following networking resources:

Domain

You must assign every host that is managed by Satellite to a domain. Using the domain, Satellite can manage A, AAAA, and PTR records. Even if you do not want Satellite to manage your DNS servers, you still must create and associate at least one domain. Domains are included in the naming conventions Satellite hosts, for example, a host with the name **test123** in the **example.com** domain has the fully qualified domain name **test123.example.com**.

Subnet

You must assign every host managed by Satellite to a subnet. Using subnets, Satellite can then manage IPv4 reservations. If there are no reservation integrations, you still must create and associate at least one subnet. When you manage a subnet in Satellite, you cannot create DHCP records for that subnet outside of Satellite. In Satellite, you can use IP Address Management (IPAM) to manage IP addresses with one of the following options:

- **DHCP**: DHCP Capsule manages the assignment of IP addresses by finding the next available IP address starting from the first address of the range and skipping all addresses that are reserved. Before assigning an IP address, Capsule sends an ICMP and TCP pings to check whether the IP address is in use. Note that if a host is powered off, or has a firewall configured to disable connections, Satellite makes a false assumption that the IP address is available. This check does not work for hosts that are turned off, therefore, the **DHCP** option can only be used with subnets that Satellite controls and that do not have any hosts created externally.

The Capsule DHCP module retains the offered IP addresses for a short period of time to prevent collisions during concurrent access, so some IP addresses in the IP range might remain temporarily unused.

- **Internal DB**: Satellite finds the next available IP address from the Subnet range by excluding all IP addresses from the Satellite database in sequence. The primary source of data is the database, not DHCP reservations. This IPAM is not safe when multiple hosts are being created in parallel; in that case, use **DHCP** or **Random DB** IPAM instead.
- **Random DB**: Satellite finds the next available IP address from the Subnet range by excluding all IP addresses from the Satellite database randomly. The primary source of data is the database, not DHCP reservations. This IPAM is safe to use with concurrent host creation as IP addresses are returned in random order, minimizing the chance of a conflict.
- **EUI-64**: Extended Unique Identifier (EUI) 64bit IPv6 address generation, as per RFC2373, is obtained through the 48-bit MAC address.
- **External IPAM**: Delegates IPAM to an external system through Capsule feature. Satellite currently does not ship with any external IPAM implementations, but several plug-ins are in development.
- **None**: IP address for each host must be entered manually.
  Options DHCP, Internal DB and Random DB can lead to DHCP conflicts on subnets with records created externally. These subnets must be under exclusive Satellite control.

  For more information about adding a subnet, see Section 3.7, “Adding a Subnet to Satellite Server”.

DHCP Ranges

You can define the same DHCP range in Satellite Server for both discovered and provisioned systems, but use a separate range for each service within the same subnet.

3.2. SATELLITE AND DHCP OPTIONS

Satellite manages DHCP reservations through a DHCP Capsule. Satellite also sets the **next-server** and **filename** DHCP options.

The next-server option

The **next-server** option provides the IP address of the TFTP server to boot from. This option is not set by default and must be set for each TFTP Capsule. You can use the **satellite-installer** command with the **--foreman-proxy-tftp-servername** option to set the TFTP server in the **/etc/foreman-proxy/settings.d/tftp.yml** file:

```
# satellite-installer --foreman-proxy-tftp-servername 1.2.3.4
```

Each TFTP Capsule then reports this setting through the API and Satellite can retrieve the configuration information when it creates the DHCP record.

When the PXE loader is set to **none**, Satellite does not populate the **next-server** option into the DHCP record.

If the **next-server** option remains undefined, Satellite uses reverse DNS search to find a TFTP server address to assign, but you might encounter the following problems:

- DNS timeouts during provisioning

- Querying of incorrect DNS server. For example, authoritative rather than caching
- Errors about incorrect IP address for the TFTP server. For example, **PTR record was invalid**

If you encounter these problems, check the DNS setup on both Satellite and Capsule, specifically the PTR record resolution.

The filename option

The **filename** option contains the full path to the file that downloads and executes during provisioning. The PXE loader that you select for the host or host group defines which **filename** option to use. When the PXE loader is set to **none**, Satellite does not populate the **filename** option into the DHCP record. Depending on the PXE loader option, the **filename** changes as follows:

PXE loader option	filename entry	Notes
PXELinux BIOS	**pxelinux.0**	
PXELinux UEFI	**pxelinux.efi**	
iPXE Chain BIOS	**undionly.kpxe**	
PXEGrub2 UEFI	**grub2/grubx64.efi**	x64 can differ depending on architecture
iPXE UEFI HTTP	**\http://*capsule.example.com*:8000/httpboot/ipxe-x64.efi**	Requires the **httpboot** feature and renders the **filename** as a full URL where *capsule.example.com* is a known host name of Capsule in Satellite.
Grub2 UEFI HTTP	**\http://*capsule.example.com*:8000/httpboot/grub2/grubx64.efi**	Requires the **httpboot** feature and renders the **filename** as a full URL where *capsule.example.com* is a known host name of Capsule in Satellite.

3.3. TROUBLESHOOTING DHCP PROBLEMS IN SATELLITE

Satellite can manage an ISC DHCP server on internal or external DHCP Capsule. Satellite can list, create, and delete DHCP reservations and leases. However, there are a number of problems that you might encounter on occasions.

Out of sync DHCP records

When an error occurs during DHCP orchestration, DHCP records in the Satellite database and the DHCP server might not match. To fix this, you must add missing DHCP records from the Satellite database to the DHCP server and then remove unwanted records from the DHCP server as per the following steps:

1. To preview the DHCP records that are going to be added to the DHCP server, enter the following command:

```
# foreman-rake orchestration:dhcp:add_missing subnet_name=NAME
```

2. If you are satisfied by the preview changes in the previous step, apply them by entering the above command with the **perform=1** argument:

```
# foreman-rake orchestration:dhcp:add_missing subnet_name=NAME perform=1
```

3. To keep DHCP records in Satellite and in the DHCP server synchronized, you can remove unwanted DHCP records from the DHCP server. Note that Satellite assumes that all managed DHCP servers do not contain third-party records, therefore, this step might delete those unexpected records. To preview what records are going to be removed from the DHCP server, enter the following command:

```
# foreman-rake orchestration:dhcp:remove_offending subnet_name=NAME
```

4. If you are satisfied by the preview changes in the previous step, apply them by entering the above command with the **perform=1** argument:

```
# foreman-rake orchestration:dhcp:remove_offending subnet_name=NAME perform=1
```

PXE loader option change

When the PXE loader option is changed for an existing host, this causes a DHCP conflict. The only workaround is to overwrite the DHCP entry.

Incorrect permissions on DHCP files

An operating system update can update the **dhcpd** package. This causes the permissions of important directories and files to reset so that the DHCP Capsule cannot read the required information.

For more information, see DHCP error while provisioning host from Satellite server Error ERF12-6899 ProxyAPI::ProxyException: Unable to set DHCP entry RestClient::ResourceNotFound 404 Resource Not Found on Red Hat Knowledgebase.

Changing the DHCP Capsule entry

Satellite manages DHCP records only for hosts that are assigned to subnets with a DHCP Capsule set. If you create a host and then clear or change the DHCP Capsule, when you attempt to delete the host, the action fails.

If you create a host without setting the DHCP Capsule and then try to set the DHCP Capsule, this causes DHCP conflicts.

Deleted hosts entries in the dhcpd.leases file

Any changes to a DHCP lease are appended to the end of the **dhcpd.leases** file. Because entries are appended to the file, it is possible that two or more entries of the same lease can exist in the **dhcpd.leases** file at the same time. When there are two or more entries of the same lease, the last entry in the file takes precedence. Group, subgroup and host declarations in the lease file are processed in the same manner. If a lease is deleted, **{ deleted; }** is appended to the declaration.

3.4. PREREQUISITES FOR IMAGE BASED PROVISIONING

Post-Boot Configuration Method

Images that use the **finish** post-boot configuration scripts require a managed DHCP server, such as Satellite's integrated Capsule or an external Capsule. The host must be created with a subnet associated with a DHCP Capsule, and the IP address of the host must be a valid IP address from the DHCP range.

It is possible to use an external DHCP service, but IP addresses must be entered manually. The SSH credentials corresponding to the configuration in the image must be configured in Satellite to enable the post-boot configuration to be made.

Check the following items when troubleshooting a virtual machine booted from an image that depends on post-configuration scripts:

- The host has a subnet assigned in Satellite Server.
- The subnet has a DHCP Capsule assigned in Satellite Server.
- The host has a valid IP address assigned in Satellite Server.
- The IP address acquired by the virtual machine using DHCP matches the address configured in Satellite Server.
- The virtual machine created from an image responds to SSH requests.
- The virtual machine created from an image authorizes the user and password, over SSH, which is associated with the image being deployed.
- Satellite Server has access to the virtual machine via SSH keys. This is required for the virtual machine to receive post-configuration scripts from Satellite Server.

Pre-Boot Initialization Configuration Method

Images that use the **cloud-init** scripts require a DHCP server to avoid having to include the IP address in the image. A managed DHCP Capsule is preferred. The image must have the **cloud-init** service configured to start when the system boots and fetch a script or configuration data to use in completing the configuration.

Check the following items when troubleshooting a virtual machine booted from an image that depends on initialization scripts included in the image:

- There is a DHCP server on the subnet.
- The virtual machine has the **cloud-init** service installed and enabled.

For information about the differing levels of support for **finish** and **cloud-init** scripts in virtual-machine images, see the Red Hat Knowledgebase Solution What are the supported compute resources for the finish and cloud-init scripts on the Red Hat Customer Portal.

3.5. CONFIGURING NETWORK SERVICES

Some provisioning methods use Capsule Server services. For example, a network might require Capsule Server to act as a DHCP server. A network can also use PXE boot services to install the operating system on new hosts. This requires configuring Capsule Server to use the main PXE boot services: DHCP, DNS, and TFTP.

Use the **satellite-installer** command with the options to configure these services on Satellite Server.

To configure these services on an external Capsule Server, run **satellite-installer --scenario capsule**.

Satellite Server uses **eth0** for external communication, such as connecting to Red Hat's CDN.

Procedure

To configure network services on Satellite's integrated Capsule, complete the following steps:

1. Enter the **satellite-installer** command to configure the required network services:

```
# satellite-installer --foreman-proxy-dhcp true \
--foreman-proxy-dhcp-managed true \
--foreman-proxy-dhcp-gateway "192.168.140.1" \
--foreman-proxy-dhcp-interface "eth1" \
--foreman-proxy-dhcp-nameservers "192.168.140.2" \
--foreman-proxy-dhcp-range "192.168.140.10 192.168.140.110" \
--foreman-proxy-dhcp-server "192.168.140.2" \
--foreman-proxy-dns true \
--foreman-proxy-dns-managed true \
--foreman-proxy-dns-forwarders "8.8.8.8; 8.8.4.4" \
--foreman-proxy-dns-interface "eth1" \
--foreman-proxy-dns-reverse "140.168.192.in-addr.arpa" \
--foreman-proxy-dns-server "127.0.0.1" \
--foreman-proxy-dns-zone "example.com" \
--foreman-proxy-tftp true \
--foreman-proxy-tftp-managed true
```

2. Find Capsule Server that you configure:

```
# hammer proxy list
```

3. Refresh features of Capsule Server to view the changes:

```
# hammer proxy refresh-features --name "satellite.example.com"
```

4. Verify the services configured on Capsule Server:

```
# hammer proxy info --name "satellite.example.com"
```

3.5.1. Multiple Subnets or Domains via Installer

The satellite-installer options allow only for a single DHCP subnet or DNS domain. One way to define more than one subnet is by using a custom configuration file.

For every additional subnet or domain, create an entry in **/etc/foreman-installer/custom-hiera.yaml** file:

```
dhcp::pools:
 isolated.lan:
   network: 192.168.99.0
   mask: 255.255.255.0
   gateway: 192.168.99.1
   range: 192.168.99.5 192.168.99.49

dns::zones:
  # creates @ SOA $::fqdn root.example.com.
  # creates $::fqdn A $::ipaddress
```

```
example.com: {}

# creates @ SOA test.example.net. hostmaster.example.com.
# creates test.example.net A 192.0.2.100
example.net:
  soa: test.example.net
  soaip: 192.0.2.100
  contact: hostmaster.example.com.

# creates @ SOA $::fqdn root.example.org.
# does NOT create an A record
example.org:
  reverse: true

# creates @ SOA $::fqdn hostmaster.example.com.
2.0.192.in-addr.arpa:
  reverse: true
  contact: hostmaster.example.com.
```

Execute satellite-installer to perform the changes and verify that the **/etc/dhcp/dhcpd.conf** contains appropriate entries. Subnets must be then defined in Satellite database.

3.5.2. DHCP, DNS, and TFTP Options for Network Configuration

DHCP Options

--foreman-proxy-dhcp

Enables the DHCP service. You can set this option to **true** or **false**.

--foreman-proxy-dhcp-managed

Enables Foreman to manage the DHCP service. You can set this option to **true** or **false**.

--foreman-proxy-dhcp-gateway

The DHCP pool gateway. Set this to the address of the external gateway for hosts on your private network.

--foreman-proxy-dhcp-interface

Sets the interface for the DHCP service to listen for requests. Set this to **eth1**.

--foreman-proxy-dhcp-nameservers

Sets the addresses of the nameservers provided to clients through DHCP. Set this to the address for Satellite Server on **eth1**.

--foreman-proxy-dhcp-range

A space-separated DHCP pool range for Discovered and Unmanaged services.

--foreman-proxy-dhcp-server

Sets the address of the DHCP server to manage.

DNS Options

--foreman-proxy-dns

Enables DNS service. You can set this option to **true** or **false**.

--foreman-proxy-dns-managed

Enables Foreman to manage the DNS service. You can set this option to **true** or **false**.

--foreman-proxy-dns-forwarders

Sets the DNS forwarders. Set this to your DNS servers.

--foreman-proxy-dns-interface

Sets the interface to listen for DNS requests. Set this to **eth1**.

--foreman-proxy-dns-reverse

The DNS reverse zone name.

--foreman-proxy-dns-server

Sets the address of the DNS server to manage.

--foreman-proxy-dns-zone

Sets the DNS zone name.

TFTP Options

--foreman-proxy-tftp

Enables TFTP service. You can set this option to **true** or **false**.

--foreman-proxy-tftp-managed

Enables Foreman to manage the TFTP service. You can set this option to **true** or **false**.

--foreman-proxy-tftp-servername

Sets the TFTP server to use. Ensure that you use Capsule's IP address.

Run **satellite-installer --help** to view more options related to DHCP, DNS, TFTP, and other Satellite Capsule services

3.5.3. Using TFTP Services through NAT

You can use Satellite TFTP services through NAT. To do this, on all NAT routers or firewalls, you must enable a TFTP service on UDP port 69 and enable the TFTP state tracking feature. For more information, see the documentation for your NAT device.

Using NAT on Red Hat Enterprise Linux 7:

Use the following command to allow TFTP service on UDP port 69, load the kernel TFTP state tracking module, and make the changes persistent:

```
# firewall-cmd --add-service=tftp && firewall-cmd --runtime-to-permanent
```

For a NAT running on Red Hat Enterprise Linux 6:

1. Configure the firewall to allow TFTP service UDP on port 69.

```
# iptables -A OUTPUT -i eth0 -p udp --sport 69 -m state \
--state ESTABLISHED -j ACCEPT
# service iptables save
```

2. Load the **ip_conntrack_tftp** kernel TFTP state module. In the **/etc/sysconfig/iptables-config** file, locate **IPTABLES_MODULES** and add **ip_conntrack_tftp** as follows:

```
IPTABLES_MODULES="ip_conntrack_tftp"
```

3.6. ADDING A DOMAIN TO SATELLITE SERVER

Satellite Server defines domain names for each host on the network. Satellite Server must have information about the domain and Capsule Server responsible for domain name assignment.

Checking for Existing Domains

Satellite Server might already have the relevant domain created as part of Satellite Server installation. Switch the context to **Any Organization** and **Any Location** then check the domain list to see if it exists.

DNS Server Configuration Considerations

During the DNS record creation, Satellite performs conflict DNS lookups to verify that the host name is not in active use. This check runs against one of the following DNS servers:

- The system-wide resolver if **Administer** > **Settings** > **Query local nameservers** is set to **true**.
- The nameservers that are defined in the subnet associated with the host.
- The authoritative NS-Records that are queried from the SOA from the domain name associated with the host.

If you experience timeouts during DNS conflict resolution, check the following settings:

- The subnet nameservers must be reachable from Satellite Server.
- The domain name must have a Start of Authority (SOA) record available from Satellite Server.
- The system resolver in the **/etc/resolv.conf** file must have a valid and working configuration.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

To add a domain to Satellite, complete the following steps:

1. In the Satellite web UI, navigate to **Infrastructure** > **Domains** and click **Create Domain**.
2. In the **DNS Domain** field, enter the full DNS domain name.
3. In the **Fullname** field, enter the plain text name of the domain.
4. Click the **Parameters** tab and configure any domain level parameters to apply to hosts attached to this domain. For example, user defined Boolean or string parameters to use in templates.
5. Click **Add Parameter** and fill in the **Name** and **Value** fields.
6. Click the **Locations** tab, and add the location where the domain resides.
7. Click the **Organizations** tab, and add the organization that the domain belongs to.
8. Click **Submit** to save the changes.

CLI procedure

- Use the **hammer domain create** command to create a domain:

```
# hammer domain create --name "domain_name.com" \
--description "My example domain" --dns-id 1 \
--locations "My_Location" --organizations "My_Organization"
```

In this example, the **--dns-id** option uses **1**, which is the ID of your integrated Capsule on Satellite Server.

3.7. ADDING A SUBNET TO SATELLITE SERVER

You must add information for each of your subnets to Satellite Server because Satellite configures interfaces for new hosts. To configure interfaces, Satellite Server must have all the information about the network that connects these interfaces.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Subnets**, and in the Subnets window, click **Create Subnet**
2. In the **Name** field, enter a name for the subnet.
3. In the **Description** field, enter a description for the subnet.
4. In the **Network address** field, enter the network address for the subnet.
5. In the **Network prefix** field, enter the network prefix for the subnet.
6. In the **Network mask** field, enter the network mask for the subnet.
7. In the **Gateway address** field, enter the external gateway for the subnet.
8. In the **Primary DNS server** field, enter a primary DNS for the subnet.
9. In the **Secondary DNS server**, enter a secondary DNS for the subnet.
10. From the **IPAM** list, select the method that you want to use for IP address management (IPAM). For more information about IPAM, see Section 3.1, “Network Resources”.
11. Enter the information for the IPAM method that you select. Click the **Remote Execution** tab and select the Capsule that controls the remote execution.
12. Click the **Domains** tab and select the domains that apply to this subnet.
13. Click the **Capsules** tab and select the Capsule that applies to each service in the subnet, including DHCP, TFTP, and reverse DNS services.
14. Click the **Parameters** tab and configure any subnet level parameters to apply to hosts attached to this subnet. For example, user defined Boolean or string parameters to use in templates.
15. Click the **Locations** tab and select the locations that use this Capsule.
16. Click the **Organizations** tab and select the organizations that use this Capsule.
17. Click **Submit** to save the subnet information.

CLI procedure

- Create the subnet with the following command:

```
# hammer subnet create --name "My_Network" \
--description "your_description" \
--network "192.168.140.0" --mask "255.255.255.0" \
--gateway "192.168.140.1" --dns-primary "192.168.140.2" \
--dns-secondary "8.8.8.8" --ipam "DHCP" \
--from "192.168.140.111" --to "192.168.140.250" --boot-mode "DHCP" \
--domains "example.com" --dhcp-id 1 --dns-id 1 --tftp-id 1 \
--locations "My_Location" --organizations "My_Organization"
```

NOTE

In this example, the **--dhcp-id**, **--dns-id**, and **--tftp-id** options use 1, which is the ID of the integrated Capsule in Satellite Server.

CHAPTER 4. USING INFOBLOX AS DHCP AND DNS PROVIDERS

You can use Capsule Server to connect to your Infoblox application to create and manage DHCP and DNS records, and to reserve IP addresses.

The supported Infoblox version is NIOS 8.0 or higher and Satellite 6.10 or higher.

4.1. LIMITATIONS

All DHCP and DNS records can be managed only in a single Network or DNS view. After you install the Infoblox modules on Capsule and set up the view using the **satellite-installer** command, you cannot edit the view.

Capsule Server communicates with a single Infoblox node using the standard HTTPS web API. If you want to configure clustering and High Availability, make the configurations in Infoblox.

Hosting PXE-related files using Infoblox's TFTP functionality is not supported. You must use Capsule as a TFTP server for PXE provisioning. For more information, see Chapter 3, *Configuring Networking*.

Satellite IPAM feature cannot be integrated with Infoblox.

4.2. PREREQUISITES

You must have Infoblox account credentials to manage DHCP and DNS entries in Satellite.

Ensure that you have Infoblox administration roles with the names: **DHCP Admin** and **DNS Admin**.

The administration roles must have permissions or belong to an admin group that permits the accounts to perform tasks through the Infoblox API.

4.3. INSTALLING THE INFOBLOX CA CERTIFICATE ON CAPSULE SERVER

You must install Infoblox HTTPS CA certificate on the base system for all Capsules that you want to integrate with Infoblox applications.

You can download the certificate from the Infoblox web UI, or you can use the following OpenSSL commands to download the certificate:

```
# update-ca-trust enable
# openssl s_client -showcerts -connect infoblox.example.com:443 </dev/null | \
openssl x509 -text >/etc/pki/ca-trust/source/anchors/infoblox.crt
# update-ca-trust extract
```

- The ***infoblox.example.com*** entry must match the host name for the Infoblox application in the X509 certificate.

To test the CA certificate, use a CURL query:

```
# curl -u admin:password https://infoblox.example.com/wapi/v2.0/network
```

Example positive response:

```
[
    {
        "_ref":
"network/ZG5zLm5ldHdvcmskMTkyLjE2OC4yMDIuMC8yNC8w:infoblox.example.com/24/default",
        "network": "192.168.202.0/24",
        "network_view": "default"
    }
]
```

Use the following Red Hat Knowledgebase article to install the certificate: How to install a CA certificate on Red Hat Enterprise Linux 6 / 7.

4.4. INSTALLING THE DHCP INFOBLOX MODULE

Use this procedure to install the DHCP Infoblox module on Capsule. Note that you cannot manage records in separate views.

You can also install DHCP and DNS Infoblox modules simultaneously by combining this procedure and Section 4.5, “Installing the DNS Infoblox Module”

DHCP Infoblox Record Type Considerations

Use only the **--foreman-proxy-plugin-dhcp-infoblox-record-type fixedaddress** option to configure the DHCP and DNS modules.

Configuring both DHCP and DNS Infoblox modules with the **host** record type setting causes DNS conflicts and is not supported. If you install the Infoblox module on Capsule Server with the **--foreman-proxy-plugin-dhcp-infoblox-record-type** option set to **host**, you must unset both DNS Capsule and Reverse DNS Capsule options because Infoblox does the DNS management itself. You cannot use the **host** option without creating conflicts and, for example, being unable to rename hosts in Satellite.

Procedure

To install the Infoblox module for DHCP, complete the following steps:

1. On Capsule, enter the following command:

```
# satellite-installer --enable-foreman-proxy-plugin-dhcp-infoblox \
--foreman-proxy-dhcp true \
--foreman-proxy-dhcp-managed false \
--foreman-proxy-dhcp-provider infoblox \
--foreman-proxy-plugin-dhcp-infoblox-record-type fixedaddress \
--foreman-proxy-dhcp-server infoblox.example.com \
--foreman-proxy-plugin-dhcp-infoblox-username admin \
--foreman-proxy-plugin-dhcp-infoblox-password infoblox \
--foreman-proxy-plugin-dhcp-infoblox-network-view default \
--foreman-proxy-plugin-dhcp-infoblox-dns-view default
```

2. In the Satellite web UI, navigate to **Infrastructure** > **Capsules** and select the Capsule with the Infoblox DHCP module and click **Refresh**.

3. Ensure that the **dhcp** features are listed.

4. For all domains managed through Infoblox, ensure that the DNS Capsule is set for that domain. To verify, in the Satellite web UI, navigate to **Infrastructure** > **Domains**, and inspect the settings of each domain.

5. For all subnets managed through Infoblox, ensure that DHCP Capsule and Reverse DNS Capsule is set. To verify, in the Satellite web UI, navigate to **Infrastructure** > **Subnets**, and inspect the settings of each subnet.

4.5. INSTALLING THE DNS INFOBLOX MODULE

Use this procedure to install the DNS Infoblox module on Capsule. You can also install DHCP and DNS Infoblox modules simultaneously by combining this procedure and Section 4.4, "Installing the DHCP Infoblox module".

DNS records are managed only in the default DNS view, it's not possible to specify which DNS view to use.

To install the DNS Infoblox module, complete the following steps:

1. On Capsule, enter the following command to configure the Infoblox module:

```
# satellite-installer --enable-foreman-proxy-plugin-dns-infoblox \
--foreman-proxy-dns true \
--foreman-proxy-dns-managed false \
--foreman-proxy-dns-provider infoblox \
--foreman-proxy-plugin-dns-infoblox-dns-server infoblox.example.com \
--foreman-proxy-plugin-dns-infoblox-username admin \
--foreman-proxy-plugin-dns-infoblox-password infoblox \
--foreman-proxy-plugin-dns-infoblox-dns-view default
```

   Optionally, you can change the value of the **--foreman-proxy-plugin-dns-infoblox-dns-view** option to specify a DNS Infoblox view other than the default view.

2. In the Satellite web UI, navigate to **Infrastructure** > **Capsules** and select the Capsule with the Infoblox DNS module and click **Refresh**.

3. Ensure that the **dns** features are listed.

CHAPTER 5. CONFIGURING IPXE TO REDUCE PROVISIONING TIMES

You can use Satellite to configure PXELinux to chainboot iPXE in BIOS mode and boot using the HTTP protocol if you have the following restrictions that prevent you from using PXE:

- A network with unmanaged DHCP servers.
- A PXE service that is blacklisted on your network or restricted by a firewall.
- An unreliable TFTP UDP-based protocol because of, for example, a low-bandwidth network.

For more information about iPXE support, see Supported architectures for provisioning article.

iPXE Overview

iPXE is an open source network boot firmware. It provides a full PXE implementation enhanced with additional features, including booting from HTTP server. For more information, see ipxe.org.

There are three methods of using iPXE with Red Hat Satellite:

1. Booting virtual machines using hypervisors that use iPXE as primary firmware.
2. Using PXELinux through TFTP to chainload iPXE directly on bare metal hosts.
3. Using PXELinux through UNDI, which uses HTTP to transfer the kernel and the initial RAM disk on bare-metal hosts.

Security Information

The iPXE binary in Red Hat Enterprise Linux is built without any security features. For this reason, you can only use HTTP, and cannot use HTTPS.

All security-related features of iPXE in Red Hat Enterprise Linux are not supported. For more information, see Red Hat Enterprise Linux HTTPS support in iPXE .

Prerequisites

Before you begin, ensure that the following conditions are met:

- A host exists on Red Hat Satellite to use.
- The MAC address of the provisioning interface matches the host configuration.
- The provisioning interface of the host has a valid DHCP reservation.
- The NIC is capable of PXE booting. For more information, see supported hardware on ipxe.org for a list of hardware drivers expected to work with an iPXE-based boot disk.
- The NIC is compatible with iPXE.

To prepare iPXE environment, you must perform this procedure on all Capsules.

Procedure

1. Enable the **tftp** and **httpboot** services:

```
# satellite-installer --foreman-proxy-httpboot true --foreman-proxy-tftp true
```

2. Install the **ipxe-bootimgs** RPM package:

```
# yum install ipxe-bootimgs
```

- Copy the iPXE firmware with the Linux kernel header to the TFTP directory:

```
# cp /usr/share/ipxe/ipxe.lkrn /var/lib/tftpboot/
```

- Copy the UNDI iPXE firmware to the TFTP directory:

```
# cp /usr/share/ipxe/undionly.kpxe /var/lib/tftpboot/undionly-ipxe.0
```

3. Correct the SELinux file contexts:

```
# restorecon -RvF /var/lib/tftpboot/
```

4. Optionally, configure Foreman discovery. For more information, see Chapter 7, *Configuring the Discovery Service*.

- In the Satellite web UI, navigate to **Administer** > **Settings**, and click the **Provisioning** tab.
- Locate the **Default PXE global template entry** row and, in the **Value** column, change the value to **discovery**.

5.1. BOOTING VIRTUAL MACHINES

Some virtualization hypervisors use iPXE as primary firmware for PXE booting. Because of this, you can boot virtual machines without TFTP and PXELinux.

Chainbooting virtual machine workflow

Using virtualization hypervisors removes the need for TFTP and PXELinux. It has the following workflow:

1. Virtual machine starts
2. iPXE retrieves the network credentials using DHCP
3. iPXE retrieves the HTTP address using DHCP
4. iPXE loads the iPXE bootstrap template from Capsule
5. iPXE loads the iPXE template with MAC as a URL parameter from Capsule
6. iPXE loads the kernel and initial RAM disk of the installer

Ensure that the hypervisor that you want to use supports iPXE. The following virtualization hypervisors support iPXE:

- libvirt
- Red Hat Virtualization
- RHEV

Configuring Satellite Server to use iPXE

You can use the default template to configure iPXE booting for hosts. If you want to change the default values in the template, clone the template and edit the clone.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates**, enter **Kickstart default iPXE** and click **Search**.
2. Optional: If you want to change the template, click **Clone**, enter a unique name, and click **Submit**.
3. Click the name of the template you want to use.
4. If you clone the template, you can make changes you require on the **Template** tab.
5. Click the **Association** tab and select the operating systems that your host uses.
6. Click the **Locations** tab and add the location where the host resides.
7. Click the **Organizations** tab and add the organization that the host belongs to.
8. Click **Submit** to save the changes.
9. Navigate to **Hosts** > **Operating systems** and select the operating system of your host.
10. Click the **Templates** tab.
11. From the **iPXE Template** list, select the template you want to use.
12. Click **Submit** to save the changes.
13. Navigate to **Hosts** > **All Hosts**.
14. In the **Hosts** page, select the host that you want to use.
15. Select the **Operating System** tab.
16. Set **PXE Loader** to **iPXE Embedded** or **None**.
17. Select the **Templates** tab.
18. From the **iPXE template** list, select **Review** to verify that the **Kickstart default iPXE** template is the correct template.
19. Configure the **dhcpd.conf** file as follows:

```
if exists user-class and option user-class = "iPXE" {
  filename "http://capsule.example.com:8000/unattended/iPXE?bootstrap=1";
} # elseif existing statements if non-iPXE environment should be preserved
```

If you use an isolated network, use a Capsule Server URL with TCP port **8000**, instead of the URL of Satellite Server.

NOTE

If you have changed the port using the **--foreman-proxy-http-port installer** option, use your custom port. You must update the **/etc/dhcp/dhcpd.conf** file after every upgrade.

5.2. CHAINBOOTING IPXE FROM PXELINUX

Use this procedure to set up iPXE to use a built-in driver for network communication or UNDI interface. To use HTTP with iPXE, use iPXE build with built-in drivers (**ipxe.lkrn**). Universal Network Device Interface (UNDI) is a minimalistic UDP/IP stack that implements TFTP client, however, cannot support other protocols like HTTP (**undionly-ipxe.0**). You can choose to either load **ipxe.lkrn** or **undionly-ipxe.0** file depending on the networking hardware capabilities and iPXE driver availablity.

Chainbooting iPXE directly or with UNDI workflow

1. Host powers on
2. PXE driver retrieves the network credentials using DHCP
3. PXE driver retrieves the PXELinux firmware **pxelinux.0** using TFTP
4. PXELinux searches for the configuration file on the TFTP server
5. PXELinux chainloads iPXE **ipxe.lkrn** or **undionly-ipxe.0**
6. iPXE retrieves the network credentials using DHCP again
7. iPXE retrieves HTTP address using DHCP
8. iPXE chainloads the iPXE template from the template Capsule
9. iPXE loads the kernel and initial RAM disk of the installer

Configuring Red Hat Satellite Server to use iPXE

You can use the default template to configure iPXE booting for hosts. If you want to change the default values in the template, clone the template and edit the clone.

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates**.
2. Enter **PXELinux chain iPXE** to use **ipxe.lkrn** or, for BIOS systems, enter **PXELinux chain iPXE UNDI** to use **undionly-ipxe.0**, and click **Search**.
3. Optional: If you want to change the template, click **Clone**, enter a unique name, and click **Submit**.
4. Click the name of the template you want to use.
5. If you clone the template, you can make changes you require on the **Template** tab.
6. Click the **Association** tab and select the operating systems that your host uses.
7. Click the **Locations** tab and add the location where the host resides.
8. Click the **Organizations** tab and add the organization that the host belongs to.

9. Click **Submit** to save the changes.
10. In the **Provisioning Templates** page, enter **Kickstart default iPXE** into the search field and click **Search**.
11. Optional: If you want to change the template, click **Clone**, enter a unique name, and click **Submit**.
12. Click the name of the template you want to use.
13. If you clone the template, you can make changes you require on the **Template** tab.
14. Click the **Association** tab and associate the template with the operating system that your host uses.
15. Click the **Locations** tab and add the location where the host resides.
16. Click the **Organizations** tab and add the organization that the host belongs to.
17. Click **Submit** to save the changes.
18. Navigate to **Hosts** > **Operating systems** and select the operating system of your host.
19. Click the **Templates** tab.
20. From the **PXELinux template** list, select the template you want to use.
21. From the **iPXE template** list, select the template you want to use.
22. Click **Submit** to save the changes.
23. Navigate to **Hosts** > **All Hosts**, and select the host you want to use.
24. Select the **Operating System** tab.
25. Set **PXE Loader** to **PXELinux BIOS** to chainboot iPXE via PXELinux, or to **iPXE Chain BIOS** to load **undionly-ipxe.0** directly.
26. Select the **Templates** tab, and from the **PXELinux template** list, select **Review** to verify the template is the correct template.
27. From the **iPXE template** list, select **Review** to verify the template is the correct template. If there is no PXELinux entry, or you cannot find the new template, navigate to **Hosts** > **All Hosts**, and on your host, click **Edit**. Click the **Operating system** tab and click the Provisioning Template **Resolve** button to refresh the list of templates.
28. Configure the **dhcpd.conf** file as follows:

```
if exists user-class and option user-class = "iPXE" {
  filename "http://capsule.example.com:8000/unattended/iPXE?bootstrap=1";
} # elseif existing statements if non-iPXE environment should be preserved
```

NOTE

If you have changed the port using the **--foreman-proxy-http-port** installer option, use your custom port. You must update the **/etc/dhcp/dhcpd.conf** file after every upgrade.

CHAPTER 6. USING PXE TO PROVISION HOSTS

There are four main ways to provision bare metal instances with Red Hat Satellite 6.10:

Unattended Provisioning

New hosts are identified by a MAC address and Satellite Server provisions the host using a PXE boot process.

Unattended Provisioning with Discovery

New hosts use PXE boot to load the Satellite Discovery service. This service identifies hardware information about the host and lists it as an available host to provision. For more information, see Chapter 7, *Configuring the Discovery Service*.

PXE-less Provisioning with Discovery

New hosts use an ISO boot disk that loads the Satellite Discovery service. This service identifies hardware information about the host and lists it as an available host to provision. For more information, see Section 7.7, "Implementing PXE-less Discovery".

BIOS and UEFI Support

With Red Hat Satellite, you can perform both BIOS and UEFI based PXE provisioning.

Both BIOS and UEFI interfaces work as interpreters between the computer's operating system and firmware, initializing the hardware components and starting the operating system at boot time.

For information about supported workflows, see Supported architectures and provisioning scenarios.

In Satellite provisioning, the PXE loader option defines the DHCP **filename** option to use during provisioning. For BIOS systems, use the **PXELinux BIOS** option to enable a provisioned node to download the **pxelinux.0** file over TFTP. For UEFI systems, use the **PXEGrub2 UEFI** option to enable a TFTP client to download **grub2/grubx64.efi** file.

For BIOS provisioning, you must associate a PXELinux template with the operating system.

For UEFI provisioning, you must associate a PXEGrub2 template with the operating system.

If you associate both PXELinux and PXEGrub2 templates, Satellite 6 can deploy configuration files for both on a TFTP server, so that you can switch between PXE loaders easily.

6.1. PREREQUISITES FOR BARE METAL PROVISIONING

The requirements for bare metal provisioning include:

- A Capsule Server managing the network for bare metal hosts. For unattended provisioning and discovery-based provisioning, Satellite Server requires PXE server settings.
  For more information about networking requirements, see Chapter 3, *Configuring Networking*.

  For more information about the Discovery service, Chapter 7, *Configuring the Discovery Service*.
- A bare metal host or a blank VM.
- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.

6.2. CONFIGURING THE SECURITY TOKEN VALIDITY DURATION

To adjust the token's duration of validity, in the Satellite web UI, navigate to **Administer** > **Settings**, and click the **Provisioning** tab. Find the **Token duration** option, and click the edit icon and edit the duration, or enter **0** to disable token generation.

If token generation is disabled, an attacker can spoof client IP address and download kickstart from Satellite Server, including the encrypted root password.

6.3. CREATING HOSTS WITH UNATTENDED PROVISIONING

Unattended provisioning is the simplest form of host provisioning. You enter the host details on Satellite Server and boot your host. Satellite Server automatically manages the PXE configuration, organizes networking services, and provides the operating system and configuration for the host.

This method of provisioning hosts uses minimal interaction during the process.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter a name for the host.
3. Click the **Organization** and **Location** tabs and change the context to match your requirements.
4. From the **Host Group** list, select a host group that you want to use to populate the form.
5. Click the **Interface** tab, and on the host's interface, click **Edit**.
6. Verify that the fields are populated with values. Note in particular:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.
7. In the **MAC address** field, enter a MAC address for the host. This ensures the identification of the host during the PXE boot process.
8. Ensure that Satellite Server automatically selects the **Managed**, **Primary**, and **Provision** options for the first interface on the host. If not, select them.
9. In the **MAC address** field, enter a MAC address of the host's provisioning interface. This ensures the identification of the host during the PXE boot process.
10. Click **OK** to save. To add another interface, click **Add Interface**. You can select only one interface for **Provision** and **Primary**.
11. Click the **Operating System** tab, and verify that all fields contain values. Confirm each aspect of the operating system.
12. Optional: Click **Resolve** in **Provisioning template** to check the new host can identify the right provisioning templates to use.
    For more information about associating provisioning templates, see Section 2.11, "Provisioning Templates".

13. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.

14. Click **Submit** to save the host details.
    For more information about network interfaces, see Adding network interfaces.

This creates the host entry and the relevant provisioning settings. This also includes creating the necessary directories and files for PXE booting the bare metal host. If you start the physical host and set its boot mode to PXE, the host detects the DHCP service of Satellite Server's integrated Capsule, receives HTTP endpoint of the Kickstart tree and installs the operating system.

When the installation completes, the host also registers to Satellite Server using the activation key and installs the necessary configuration and management tools from the Satellite Tools 6.10 repository.

CLI procedure

1. Create the host with the **hammer host create** command.

```
# hammer host create --name "My_Unattended_Host" --organization "My_Organization" \
--location "My_Location" --hostgroup "My_Host_Group" --mac "aa:aa:aa:aa:aa:aa" \
--build true --enabled true --managed true
```

2. Ensure the network interface options are set using the **hammer host interface update** command.

```
# hammer host interface update --host "test1" --managed true \
--primary true --provision true
```

6.4. CREATING HOSTS WITH UEFI HTTP BOOT PROVISIONING

You can provision hosts from Satellite using the UEFI HTTP Boot. This is the only method with which you can provision hosts in IPv6 network.

To use the CLI instead of the web UI, see the CLI procedure.

Prerequisites

- Ensure that you meet the requirements for HTTP booting. For more information, see HTTP Booting Requirements in *Planning for Satellite*.

Procedure

1. On Capsule that you use for provisioning, update the **grub2-efi** package to the latest version:

```
# satellite-maintain packages install grub2-efi
```

2. Enable **foreman-proxy-http**, **foreman-proxy-httpboot**, and **foreman-proxy-tftp** features.

```
# satellite-installer --scenario satellite \
--foreman-proxy-httpboot true \
--foreman-proxy-http true \
--foreman-proxy-tftp true
```

3. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
4. In the **Name** field, enter a name for the host.
5. Click the **Organization** and **Location** tabs and change the context to match your requirements.
6. From the **Host Group** list, select a host group that you want to use to populate the form.
7. Click the **Interface** tab, and on the host's interface, click **Edit**.
8. Verify that the fields are populated with values. Note in particular:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.
9. In the **MAC address** field, enter a MAC address of the host's provisioning interface. This ensures the identification of the host during the PXE boot process.
10. Ensure that Satellite Server automatically selects the **Managed**, **Primary**, and **Provision** options for the first interface on the host. If not, select them.
11. Click **OK** to save. To add another interface, click **Add Interface**. You can select only one interface for **Provision** and **Primary**.
12. Click the **Operating System** tab, and verify that all fields contain values. Confirm each aspect of the operating system.
13. From the **PXE Loader** list, select **Grub2 UEFI HTTP**.
14. Optional: Click **Resolve** in **Provisioning template** to check the new host can identify the right provisioning templates to use.
    For more information about associating provisioning templates, see Section 2.13, "Creating Provisioning Templates".
15. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.
16. Click **Submit** to save the host details.
    For more information about network interfaces, see Adding network interfaces.
17. Set the host to boot in UEFI mode from network.
18. Start the host.
19. From the boot menu, select **Kickstart default PXEGrub2**.

This creates the host entry and the relevant provisioning settings. This also includes creating the necessary directories and files for UEFI booting the bare metal host. When you start the physical host and set its boot mode to UEFI HTTP, the host detects the defined DHCP service, receives HTTP endpoint of Capsule with the Kickstart tree and installs the operating system.

When the installation completes, the host also registers to Satellite Server using the activation key and installs the necessary configuration and management tools from the Satellite Tools 6.10 repository.

CLI procedure

1. On Capsule that you use for provisioning, update the **grub2-efi** package to the latest version:

```
# satellite-maintain packages install grub2-efi
```

2. Enable **foreman-proxy-http**, **foreman-proxy-httpboot**, and **foreman-proxy-tftp true** features.

```
# satellite-installer --scenario satellite \
--foreman-proxy-httpboot true \
--foreman-proxy-http true \
--foreman-proxy-tftp true
```

3. Create the host with the **hammer host create** command.

```
# hammer host create --name "My_Host" \
--organization "My_Organization" \
--location "My_Location" \
--hostgroup "My_Host_Group" \
--mac "aa:aa:aa:aa:aa:aa" \
--build true \
--enabled true \
--managed true \
--pxe-loader "Grub2 UEFI HTTP"
```

4. Ensure the network interface options are set using the **hammer host interface update** command.

```
# hammer host interface update --host "My_Host" \
--managed true \
--primary true \
--provision true
```

5. Set the host to boot in UEFI mode from network.

6. Start the host.

7. From the boot menu, select **Kickstart default PXEGrub2**.

This creates the host entry and the relevant provisioning settings. This also includes creating the necessary directories and files for UEFI booting the bare metal host. When you start the physical host and set its boot mode to UEFI HTTP, the host detects the defined DHCP service, receives HTTP endpoint of Capsule with the Kickstart tree and installs the operating system.

When the installation completes, the host also registers to Satellite Server using the activation key and installs the necessary configuration and management tools from the Satellite Tools 6.10 repository.

6.5. DEPLOYING SSH KEYS DURING PROVISIONING

Use this procedure to deploy SSH keys added to a user during provisioning. For information on adding SSH keys to a user, see Managing SSH Keys for a User in *Administering Red Hat Satellite*.

Procedure

To deploy SSH keys during provisioning, complete the following steps:

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates**.

2. Create a provisioning template, or clone and edit an existing template. For more information, see Section 2.13, “Creating Provisioning Templates”.
3. In the template, click the **Template** tab.
4. In the **Template editor** field, add the **create_users** snippet to the **%post** section:

```
<%= snippet('create_users') %>
```

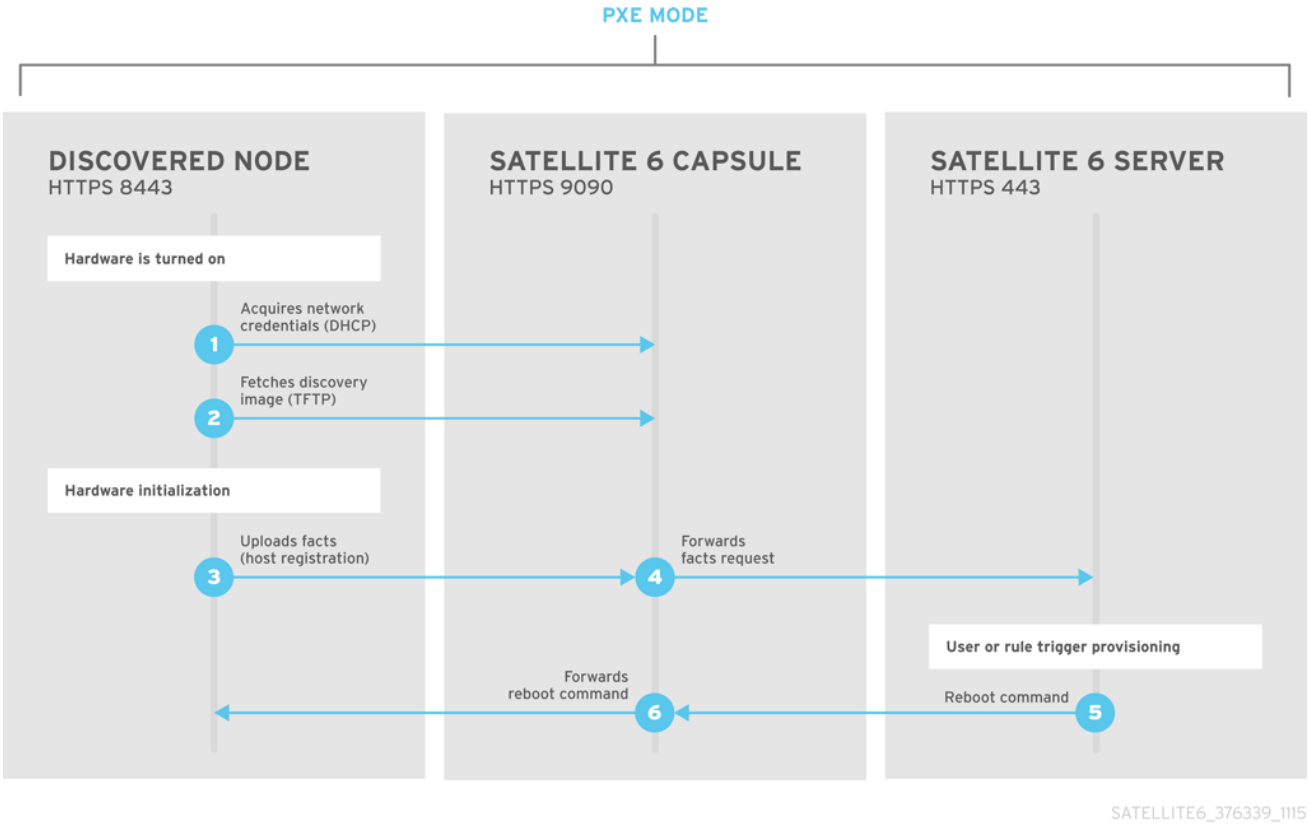
5. Select the **Default** check box.
6. Click the **Association** tab.
7. From the **Application Operating Systems** list, select an operating system.
8. Click **Submit** to save the provisioning template.
9. Create a host that is associated with the provisioning template or rebuild a host using the OS associated with the modified template. For more information, see Creating a Host in the *Managing Hosts* guide.
   The SSH keys of the **Owned by** user are added automatically when the **create_users** snippet is executed during the provisioning process. You can set **Owned by** to an individual user or a user group. If you set **Owned by** to a user group, the SSH keys of all users in the user group are added automatically.

CHAPTER 7. CONFIGURING THE DISCOVERY SERVICE

Red Hat Satellite can detect hosts on a network that are not in your Satellite inventory. These hosts boot the discovery image that performs hardware detection and relays this information back to Satellite Server. This method creates a list of ready-to-provision hosts in Satellite Server without needing to enter the MAC address of each host.

When you boot a blank bare-metal host, the boot menu has two options: **local** and **discovery**. If you select **discovery** to boot the Discovery image, after a few minutes, the Discovery image completes booting and a status screen is displayed.

The Discovery service is enabled by default on Satellite Server. However, the default setting of the global templates is to boot from the local hard drive. To change the default setting, in the Satellite web UI, navigate to **Administer** > **Settings**, and click the **Provisioning** tab. Locate the **Default PXE global template entry** row, and in the **Value** column, enter **discovery**.



To use Satellite Server to provide the Discovery image, install the following RPM packages:

- **tfm-rubygem-foreman_discovery**
- **foreman-discovery-image**
- **tfm-rubygem-smart_proxy_discovery**

The **tfm-rubygem-foreman_discovery** package contains the Satellite plug-in to handle discovered nodes, connections, and necessary database structures, and API.

The **tfm-rubygem-smart_proxy_discovery** package configures Capsule Server, such as the integrated Capsule of Satellite Server, to act as a proxy for the Discovery service.

When the installation completes, you can view the new menu option by navigating to **Hosts** > **Discovered Hosts**.

7.1. INSTALLING THE DISCOVERY SERVICE

Complete the following procedure to enable the Discovery service on Capsule Server.

Procedure

1. Enter the following commands on Capsule Server:

```
# satellite-maintain packages install foreman-discovery-image tfm-rubygem-
smart_proxy_discovery
```

2. Restart the satellite-maintain services:

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

3. In the Satellite web UI, navigate to **Infrastructure** > **Capsule**.
4. Click Capsule Server and select **Refresh** from the **Actions** list. Locate **Discovery** in the list of features to confirm the Discovery service is now running.

Subnets

All subnets with discoverable hosts require an appropriate Capsule Server selected to provide the Discovery service.

To check this, navigate to **Infrastructure** > **Capsules** and verify if Capsule Server that you want to use lists the Discovery feature. If not, click **Refresh features**.

In the Satellite web UI, navigate to **Infrastructure** > **Subnets**, select a subnet, click the Capsules tab, and select the **Discovery Proxy** that you want to use. Perform this for each subnet that you want to use.

7.2. THE PROVISIONING TEMPLATE PXELINUX DISCOVERY SNIPPET

For BIOS provisioning, the **PXELinux global default** template in the **Hosts** > **Provisioning Templates** window contains the snippet **pxelinux_discovery**. The snippet has the following lines:

```
LABEL discovery
  MENU LABEL Foreman Discovery Image
  KERNEL boot/fdi-image/vmlinuz0
  APPEND initrd=boot/fdi-image/initrd0.img rootflags=loop root=live:/fdi.iso rootfstype=auto ro
rd.live.image acpi=force rd.luks=0 rd.md=0 rd.dm=0 rd.lvm=0 rd.bootif=0 rd.neednet=0 nomodeset
proxy.url=<%= foreman_server_url %> proxy.type=foreman
  IPAPPEND 2
```

The **KERNEL** and **APPEND** options boot the Discovery image and ramdisk. The **APPEND** option contains a **proxy.url** parameter, with the **foreman_server_url** macro as its argument. This macro resolves to the full URL of Satellite Server.

For UEFI provisioning, the **PXEgrub2 global default** template in the **Hosts** > **Provisioning Templates** window contains the snippet **pxegrub2_discovery**:

```
menuentry 'Foreman Discovery Image' --id discovery {
  linuxefi boot/fdi-image/vmlinuz0 rootflags=loop root=live:/fdi.iso rootfstype=auto ro rd.live.image
acpi=force rd.luks=0 rd.md=0 rd.dm=0 rd.lvm=0 rd.bootif=0 rd.neednet=0 nomodeset proxy.url=<%=
```

```
foreman_server_url %> proxy.type=foreman BOOTIF=01-$mac
  initrdefi boot/fdi-image/initrd0.img
}
```

To use Capsule to proxy the discovery steps, edit **/var/lib/tftpboot/pxelinux.cfg/default** or **/var/lib/tftpboot/grub2/grub.cfg** and change the URL to Capsule Server FQDN you want to use.

The global template is available on Satellite Server and all Capsules that have the TFTP feature enabled.

7.3. AUTOMATIC CONTEXTS FOR DISCOVERED HOSTS

Satellite Server assigns an organization and location to discovered hosts according to the following sequence of rules:

1. If a discovered host uses a subnet defined in Satellite, the host uses the first organization and location associated with the subnet.

2. If the **discovery_organization** or **discovery_location** fact values are set, the discovered host uses these fact values as an organization and location. To set these fact values, navigate to **Administer** > **Settings** > **Discovered**, and add these facts to the **Default organization** and **Default location** fields. Ensure that the discovered host's subnet also belongs to the organization and location set by the fact, otherwise Satellite refuses to set it for security reasons.

3. If none of the previous conditions exists, Satellite assigns the first Organization and Location ordered by name.

You can change the organization or location using the bulk actions menu of the **Discovered hosts** page. Select the discovered hosts to modify and select **Assign Organization** or **Assign Location** from the **Select Action** menu.

Note that the **foreman_organization** and **foreman_location** facts are no longer valid values for assigning context for discovered hosts. You still can use these facts to configure the host for Puppet runs. To do this, navigate to **Administer** > **Settings** > **Puppet** section and add the **foreman_organization** and **foreman_location** facts to the **Default organization** and **Default location** fields.

7.4. DISCOVERY TEMPLATES AND SNIPPETS SETTINGS

To use the Discovery service, you must configure provisioning settings to set Discovery as the default service and set the templates that you want to use.

Setting Discovery Service as Default

For both BIOS and UEFI, to set the Discovery service as the default service that boots for hosts that are not present in your current Satellite inventory, complete the following steps:

1. In the Satellite web UI, navigate to **Administer** > **Settings** and click the **Provisioning** tab.

2. For the **Default PXE global template entry**, in the **Value** column, enter **discovery**.

To use a template, in the Satellite web UI, navigate to **Administer** > **Settings** and click the **Provisioning** tab and set the templates that you want to use.

Customizing Templates and Snippets

Templates and snippets are locked to prevent changes. If you want to edit a template or snippet, clone it, save it with a unique name, and then edit the clone.

When you change the template or a snippet it includes, the changes must be propagated to Satellite Server's default PXE template.

In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates** and click **Build PXE Default**.

This refreshes the default PXE template on Satellite Server.

Additional Settings

The proxy.url argument

During the Satellite installation process, if you use the default option **--enable-foreman-plugin-discovery**, you can edit the **proxy.url** argument in the template to set the URL of Capsule Server that provides the discovery service. You can change the **proxy.url** argument to the IP address or FQDN of another provisioning Capsule that you want to use, but ensure that you append the port number, for example, **9090**. If you use an alternative port number with the **--foreman-proxy-ssl-port** option during Satellite installation, you must add that port number. You can also edit the **proxy.url** argument to use a Satellite IP address or FQDN so that the discovered hosts communicate directly with Satellite Server.

The proxy.type argument

If you use a Capsule Server FQDN for the **proxy.url** argument, ensure that you set the **proxy.type** argument to **proxy**. If you use a Satellite FQDN, update the **proxy.type** argument to **foreman**.

```
proxy.url=https://capsule.example.com:9090 proxy.type=proxy
```

Rendering the Capsule's Host Name

Satellite 6 deploys the same template to all TFTP Capsules and there is no variable or macro available to render the Capsule's host name. The hard-coded **proxy.url** does not not work with two or more TFTP Capsules. As a workaround, every time you click **Build PXE Defaults**, edit the configuration file in the TFTP directory using SSH, or use a common DNS alias for appropriate subnets.

Tagged VLAN Provisioning

If you want to use tagged VLAN provisioning, and you want the discovery service to send a discovery request, add the following information to the **KERNEL** option in the discovery template:

```
fdi.vlan.primary=example_VLAN_ID
```

7.5. CREATING HOSTS FROM DISCOVERED HOSTS

Provisioning discovered hosts follows a provisioning process that is similar to PXE provisioning. The main difference is that instead of manually entering the host's MAC address, you can select the host to provision from the list of discovered hosts.

To use the CLI instead of the web UI, see the CLI procedure.

Prerequisites

- Configure a domain and subnet on Satellite. For more information about networking requirements, see Chapter 3, *Configuring Networking*.

- Configure the discovery service on Satellite. For more information, see Section 7.1, "Installing the Discovery Service".
- A bare metal host or a blank VM.
- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Discovered host**. Select the host you want to use and click **Provision** to the right of the list.
2. Select from one of the two following options:
   - To provision a host from a host group, select a host group, organization, and location, and then click **Create Host**.
   - To provision a host with further customization, click **Customize Host** and enter the additional details you want to specify for the new host.
3. Verify that the fields are populated with values. Note in particular:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.
   - Satellite Server automatically populates the MAC address from the Discovery results.
4. Ensure that Satellite Server automatically selects the **Managed**, **Primary**, and **Provision** options for the first interface on the host. If not, select them.
5. Click the **Operating System** tab, and verify that all fields contain values. Confirm each aspect of the operating system.
6. Click **Resolve** in **Provisioning template** to check the new host can identify the right provisioning templates to use. The host must resolve to the following provisioning templates:
   - **kexec Template: Discovery Red Hat kexec**
   - **provision Template: Satellite Kickstart Default**
     For more information about associating provisioning templates, see Section 2.11, "Provisioning Templates".
7. Click **Submit** to save the host details.

When the host provisioning is complete, the discovered host becomes a content host. To view the host, navigate to **Hosts** > **Content Hosts**.

CLI procedure

1. Identify the discovered host to use for provisioning:

```
# hammer discovery list
```

2. Select a host and provision it using a host group. Set a new host name with the **--new-name** option:

```
# hammer discovery provision --name "host_name" \
--new-name "new_host_name" --organization "My_Organization" \
--location "My_Location" --hostgroup "My_Host_Group" --build true \
--enabled true --managed true
```

This removes the host from the discovered host listing and creates a host entry with the provisioning settings. The Discovery image automatically resets the host so that it can boot to PXE. The host detects the DHCP service on Satellite Server's integrated Capsule and starts installing the operating system. The rest of the process is identical to normal PXE workflow described in Section 6.3, "Creating Hosts with Unattended Provisioning".

7.6. CREATING DISCOVERY RULES

As a method of automating the provisioning process for discovered hosts, Satellite provides a feature to create discovery rules. These rules define how discovered hosts automatically provision themselves, based on the assigned host group. For example, you can automatically provision hosts with a high CPU count as hypervisors. Likewise, you can provision hosts with large hard disks as storage servers.

To use the CLI instead of the web UI, see the CLI procedure.

NIC Considerations

Auto provisioning does not currently allow configuring NICs; all systems are being provisioned with the NIC configuration that was detected during discovery. However, you can set the NIC in the **kickstart** scriplet, using a script, or using configuration management at a later stage.

Procedure

1. In the Satellite web UI, navigate to **Configure** > **Discovery rules**, and select **Create Rule**.
2. In the **Name** field, enter a name for the rule.
3. In the **Search** field, enter the rules to determine whether to provision a host. This field provides suggestions for values you enter and allows operators for multiple rules. For example: **cpu_count > 8**.
4. From the **Host Group** list, select the host group to use as a template for this host.
5. In the **Hostname** field, enter the pattern to determine host names for multiple hosts. This uses the same ERB syntax that provisioning templates use. The host name can use the **@host** attribute for host-specific values and the **rand** macro for a random number or the **sequence_hostgroup_param_next** macro for incrementing the value. For more information about provisioning templates, see Section 2.11, "Provisioning Templates" and the API documentation.
    - **myhost-<%= sequence_hostgroup_param_next("EL7/MyHostgroup", 10, "discovery_host") %>**
    - **myhost-<%= rand(99999) %>**
    - **abc-<%= @host.facts['bios_vendor'] %>-<%= rand(99999) %>**
    - **xyz-<%= @host.hostgroup.name %>**

- **srv-<%= @host.discovery_rule.name %>**
- **server-<%= @host.ip.gsub('.','-') + '-' + @host.hostgroup.subnet.name %>**
  When creating host name patterns, ensure that the resulting host names are unique, do not start with numbers, and do not contain underscores or dots. A good approach is to use unique information provided by Facter, such as the MAC address, BIOS, or serial ID.

6. In the **Hosts limit** field, enter the maximum number of hosts that you can provision with the rule. Enter **0** for unlimited.
7. In the **Priority** field, enter a number to set the precedence the rule has over other rules. Rules with lower values have a higher priority.
8. From the **Enabled** list, select whether you want to enable the rule.
9. To set a different provisioning context for the rule, click the **Organizations** and **Locations** tabs and select the contexts you want to use.
10. Click **Submit** to save your rule.
11. Navigate to **Hosts** > **Discovered Host** and select one of the following two options:
    - From the **Discovered hosts** list on the right, select **Auto-Provision** to automatically provisions a single host.
    - On the upper right of the window, click **Auto-Provision All** to automatically provisions all hosts.

CLI procedure

1. Create the rule with the **hammer discovery-rule create** command:

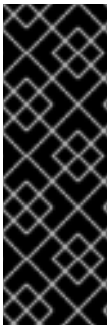
```
# hammer discovery-rule create --name "Hypervisor" \
--search "cpu_count  > 8" --hostgroup "My_Host_Group" \
--hostname "hypervisor-<%= rand(99999) %>" \
--hosts-limit 5 --priority 5 --enabled true
```

2. Automatically provision a host with the **hammer discovery auto-provision** command:

```
# hammer discovery auto-provision --name "macabcdef123456"
```

7.7. IMPLEMENTING PXE-LESS DISCOVERY

Satellite provides a PXE-less Discovery service that operates without the need for PXE-based services (DHCP and TFTP). You accomplish this using Satellite Server's Discovery image. Once a discovered node is scheduled for installation, it uses **kexec** command to reload Linux kernel with OS installer without rebooting the node.

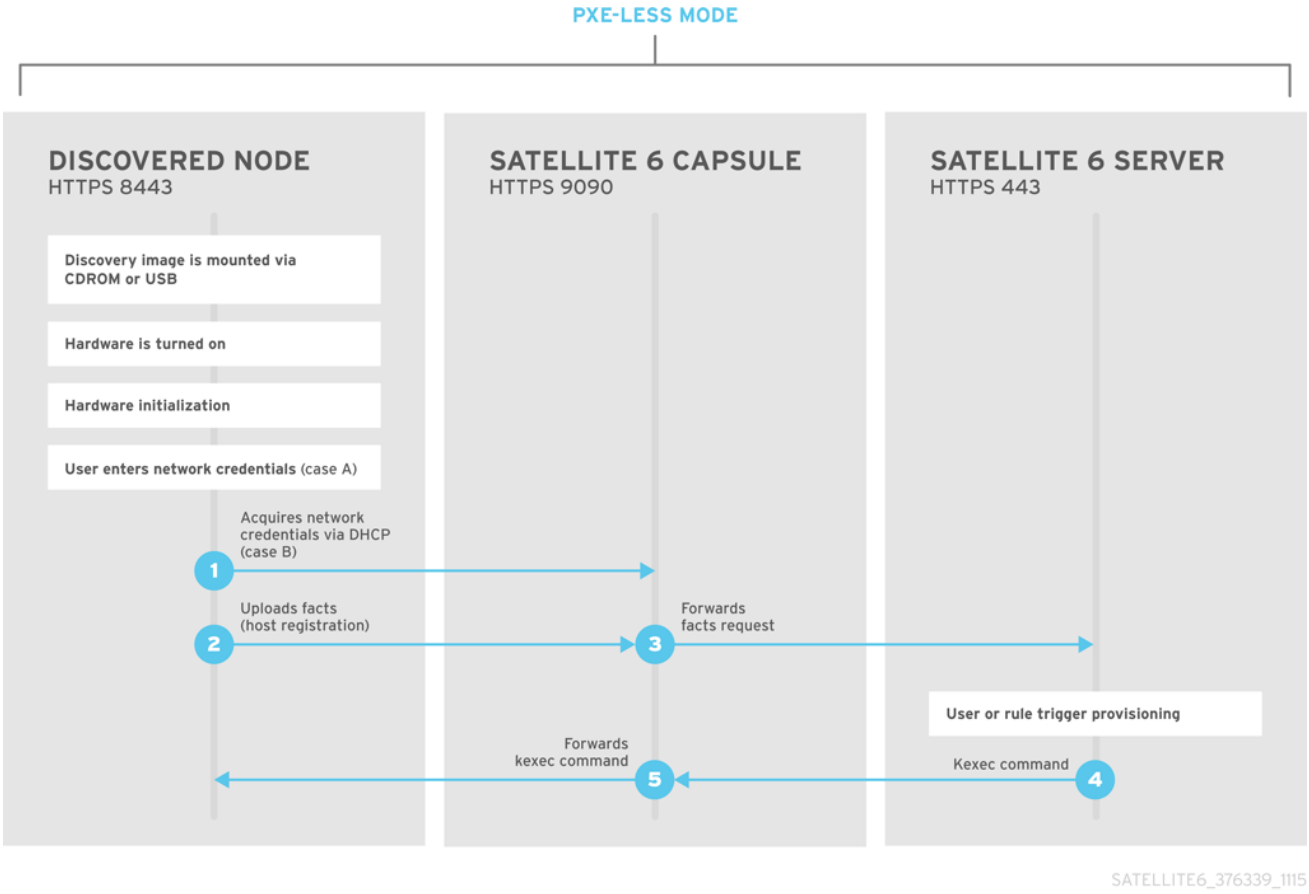


IMPORTANT

Discovery **kexec** is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see https://access.redhat.com/support/offerings/techpreview/.

Known Issues

The console might freeze during the process. On some hardware, you might experience graphical hardware problems.



The ISO for the Discovery service resides at **/usr/share/foreman-discovery-image/** and is installed using the **foreman-discovery-image** package.

Attended Use

1. Copy this media to either a CD, DVD, or a USB stick. For example, to copy to a USB stick at **/dev/sdb**:

```
# dd bs=4M \
if=/usr/share/foreman-discovery-image/foreman-discovery-image-3.4.4-5.iso \
of=/dev/sdb
```

2. Insert the Discovery boot media into a bare metal host, start the host, and boot from the media. The Discovery Image displays an option for either **Manual network setup** or **Discovery with DHCP**:

If you select **Manual network setup**, the Discovery image requests a set of network options. This includes the primary network interface that connects to Satellite Server. This Discovery image also asks for network interface configuration options, such as an **IPv4 Address**, **IPv4 Gateway**, and an **IPv4 DNS** server.

3. After entering these details, select **Next**.

4. If you select **Discovery with DHCP**, the Discovery image requests only the primary network interface that connects to Satellite Server. It attempts to automatically configure the network interface using a DHCP server, such as one that a Capsule Server provides.

5. After the primary interface configuration, the Discovery image requests the **Server URL**, which is the URL of Satellite Server or Capsule Server offering the Discovery service. For example, to use the integrated Capsule on Satellite Server, use the following URL:

```
https://satellite.example.com:9090
```

6. Set the **Connection type** to **Proxy**, then select **Next**.

7. Optional: The Discovery image also provides a set of fields to input **Custom facts** for the Facter tool to relay back to Satellite Server. These are entered in a **name-value** format. Provide any custom facts you require and select **Confirm** to continue.

Satellite reports a successful communication with Satellite Server's Discovery service. Navigate to **Hosts** > **Discovered Hosts** and view the newly discovered host.

For more information about provisioning discovered hosts, see Section 7.5, "Creating Hosts from Discovered Hosts".

7.8. UNATTENDED USE, CUSTOMIZATION, AND IMAGE REMASTERING

You can create a customized Discovery ISO to automate the image configuration process after booting. The Discovery image uses a Linux kernel for the operating system, which passes kernel parameters to configure the discovery service. These kernel parameters include the following entries:

proxy.url

The URL of Capsule Server or Satellite Server providing the Discovery service.

proxy.type

The proxy type. This is usually set to **proxy** to connect to Capsule Server. This parameter also supports a legacy **foreman** option, where communication goes directly to Satellite Server instead of a Capsule Server.

fdi.pxmac

The MAC address of the primary interface in the format of **AA:BB:CC:DD:EE:FF**. This is the interface you aim to use for communicating with Capsule Server. In automated mode, the first NIC (using network identifiers in alphabetical order) with a link is used. In semi-automated mode, a screen appears and requests you to select the correct interface.

fdi.pxip, fdi.pxgw, fdi.pxdns

Manually configures IP address (**fdi.pxip**), the gateway (**fdi.pxgw**), and the DNS (**fdi.pxdns**) for the primary network interface. If your omit these parameters, the image uses DHCP to configure the network interface.

fdi.pxfactname1, fdi.pxfactname2 ... fdi.pxfactnameN

Use to specify custom fact names.

fdi.pxfactvalue1, fdi.pxfactvalue2 ... fdi.pxfactvalueN

The values for each custom fact. Each value corresponds to a fact name. For example, **fdi.pxfactvalue1** sets the value for the fact named with **fdi.pxfactname1**.

fdi.pxauto

To set automatic or semi-automatic mode. If set to 0, the image uses semi-automatic mode, which allows you to confirm your choices through a set of dialog options. If set to 1, the image uses automatic mode and proceeds without any confirmation.

fdi.initnet

By default, the image initializes all network interfaces (value **all**). When this setting is set to **bootif**, only the network interface it was network-booted from will be initialized.

fdi.rootpw

By default, the **root** account is locked. Use this option to set a root password. You can enter both clear and encrypted passwords.

fdi.ssh

By default, the SSH service is disabled. Set this to **1** or **true** to enable SSH access.

fdi.ipv4.method

By default, NetworkManager IPv4 method setting is set to **auto**. This option overrides it, set it to **ignore** to disable the IPv4 stack. This option works only in DHCP mode.

fdi.ipv6.method

By default, NetworkManager IPv6 method setting is set to **auto**. This option overrides it, set it to **ignore** to disable the IPv6 stack. This option only works in DHCP mode.

fdi.zips

Filenames with extensions to be downloaded and started during boot. For more information, see see Section 7.9, “Extending the Discovery Image” .

fdi.zipserver

TFTP server to use to download extensions from. For more information, see see Section 7.9, “Extending the Discovery Image”.

fdi.countdown

Number of seconds to wait until the text-user interface is refreshed after the initial discovery attempt. This value defaults to 45 seconds. Increase this value if the status page reports the IP address as **N/A**.

fdi.dhcp_timeout

NetworkManager DHCP timeout. The default value is 300 seconds.

fdi.vlan.primary

VLAN tagging ID to set for the primary interface.

Using the discovery-remaster Tool to Remaster an OS Image

Satellite Server provides the **discovery-remaster** tool in the **foreman-discovery-image** package. This tool remasters the image to include these kernel parameters. To remaster the image, run the **discovery-remaster** tool. For example:

```
# discovery-remaster ~/iso/foreman-discovery-image-3.4.4-5.iso \
"fdi.pxip=192.168.140.20/24 fdi.pxgw=192.168.140.1 \
fdi.pxdns=192.168.140.2 proxy.url=https://satellite.example.com:9090 \
proxy.type=proxy fdi.pxfactname1=customhostname fdi.pxfactvalue1=myhost
fdi.pxmac=52:54:00:be:8e:8c fdi.pxauto=1"
```

Copy this media to either a CD, DVD, or a USB stick. For example, to copy to a USB stick at **/dev/sdb**:

```
# dd bs=4M \
if=/usr/share/foreman-discovery-image/foreman-discovery-image-3.4.4-5.iso \
of=/dev/sdb
```

Insert the Discovery boot media into a bare metal host, start the host, and boot from the media.

For more information about provisioning discovered hosts, see Section 7.5, "Creating Hosts from Discovered Hosts".

7.9. EXTENDING THE DISCOVERY IMAGE

You can extend the Satellite Discovery image with custom facts, software, or device drivers. You can also provide a compressed archive file containing extra code for the image to use.

Procedure

1. Create the following directory structure:

```
.
├── autostart.d
│   └── 01_zip.sh
├── bin
│   └── ntpdate
├── facts
│   └── test.rb
└── lib
    ├── libcrypto.so.1.0.0
    └── ruby
        └── test.rb
```

   - The **autostart.d** directory contains scripts that are executed in POSIX order by the image when it starts, but before the host is registered to Satellite.
   - The **bin** directory is added to the **$PATH** variable; you can place binary files in this directory and use them in the **autostart** scripts.
   - The **facts** directory is added to the **FACTERLIB** variable so that custom facts can be configured and sent to Satellite.
   - The **lib** directory is added to the **LD_LIBRARY_PATH** variable and **lib/ruby** is added to the **RUBYLIB** variable, so that binary files in **/bin** can be executed correctly.

2. After creating the directory structure, create a **.zip** file archive with the following command:

```
zip -r my_extension.zip .
```

3. To inform the Discovery image of the extensions it must use, place your zip files on your TFTP server with the Discovery image, and then update the **APPEND** line of the PXELinux template with the **fdi.zips** option where the paths are relative to the TFTP root. For example, if you have two archives at **$TFTP/zip1.zip** and **$TFTP/boot/zip2.zip**, use the following syntax:

```
fdi.zips=zip1.zip,boot/zip2.zip
```

You can append new directives and options to the existing environment variables (**PATH**, **LD_LIBRARY_PATH**, **RUBYLIB** and **FACTERLIB**). If you want to specify the path explicitly in your scripts, the **.zip** file contents are extracted to the **/opt/extension** directory on the image.

You can create multiple **.zip** files but be aware that they are extracted to the same location on the Discovery image. Files extracted from in later **.zip** files overwrite earlier versions if they have the same file name.

7.10. TROUBLESHOOTING DISCOVERY

If a machine is not listed in the Satellite web UI in **Hosts** > **Discovered Hosts**, inspect the following configuration areas to help isolate the error:

- Navigate to **Hosts** > **Provisioning Templates** and redeploy the default PXELinux template using the **Build PXE Default** button.
- Verify the **pxelinux.cfg/default** configuration file on the TFTP Capsule.
- Ensure adequate network connectivity between hosts, Capsule Server, and Satellite Server.
- Check the PXELinux template in use and determine the PXE discovery snippet it includes. Snippets are named as follows: **pxelinux_discovery**, **pxegrub_discovery**, or **pxegrub2_discovery**. Verify the **proxy.url** and **proxy.type** options in the PXE discovery snippet.
- Ensure that DNS is working correctly for the discovered nodes, or use an IP address in the **proxy.url** option in the PXE discovery snippet included in the PXELinux template you are using.
- Ensure that the DHCP server is delivering IP addresses to the booted image correctly.
- Ensure the discovered host or virtual machine has at least 1200 MB of memory. Less memory can lead to various random kernel panic errors because the image is extracted in-memory.

For gathering important system facts, use the **discovery-debug** command. It prints out system logs, network configuration, list of facts, and other information on the standard output. The typical use case is to redirect this output and copy it with the **scp** command for further investigation.

The first virtual console on the discovered host is reserved for **systemd** logs. Particularly useful system logs are tagged as follows:

- **discover-host** – initial facts upload
- **foreman-discovery** – facts refresh, reboot remote commands
- **nm-prepare** – boot script which pre-configures NetworkManager
- **NetworkManager** – networking information

Use TTY2 or higher to log in to a discovered host. The root account and SSH access are disabled by default, but you can enable SSH and set the root password using the following kernel command-line options in the Default PXELinux template on the APPEND line:

```
fdi.ssh=1 fdi.rootpw=My_Password
```

CHAPTER 8. USING A RED HAT IMAGE BUILDER IMAGE FOR PROVISIONING

In Satellite, you can integrate with Red Hat web console to perform actions and monitor your hosts. Using Red Hat web console, you can access Red Hat Image Builder and build images that you can then upload to a HTTP server and use this image to provision hosts. When you configure Satellite for image provisioning, Anaconda installer partitions disks, downloads and mounts the image and copies files over to a host. The preferred image type is TAR.

For more information about integrating Red Hat web console with Satellite, see Host Management and Monitoring Using Red Hat web console in the *Managing Hosts* guide.

Prerequisites

1. An existing TAR image created via Red Hat Image Builder.

Procedure

To use the Red Hat Image Builder image with Anaconda Kickstart in Satellite, complete the following steps:

1. On Satellite, create a custom product, add a custom file repository to this product, and upload the image to the repository. For more information, see Importing Individual ISO Images and Files in the *Content Management Guide*.
2. In the Satellite web UI, navigate to **Configure** > **Host Groups**, and select the host group that you want to use.
3. Click the **Parameters** tab, and then click **Add Parameter**.
4. In the **Name** field, enter **kickstart_liveimg**.
5. From the **Type** list, select **string**.
6. In the **Value** field, enter the absolute path or a relative path in the following format **custom/*product*/*repository*/*image_name*** that points to the exact location where you store the image.
7. Click **Submit** to save your changes.

You can use this image for bare metal provisioning and provisioning using a compute resource. For more information about bare metal provisioning, see Chapter 6, *Using PXE to Provision Hosts*. For more information about provisioning with different compute resources, see the relevant chapter for the compute resource that you want to use.

CHAPTER 9. PROVISIONING VIRTUAL MACHINES ON KVM (LIBVIRT)

Kernel-based Virtual Machines (KVMs) use an open source virtualization daemon and API called **libvirt** running on Red Hat Enterprise Linux. Red Hat Satellite can connect to the **libvirt** API on a KVM server, provision hosts on the hypervisor, and control certain virtualization functions.

You can use KVM provisioning to create hosts over a network connection or from an existing image.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- A Capsule Server managing a network on the KVM server. Ensure no other DHCP services run on this network to avoid conflicts with Capsule Server. For more information about network service configuration for Capsule Servers, see Configuring Networking.
- A Red Hat Enterprise Linux server running KVM virtualization tools (libvirt daemon). For more information, see the Red Hat Enterprise Linux 7 Virtualization Getting Started Guide .
- An existing virtual machine image if you want to use image-based provisioning. Ensure that this image exists in a storage pool on the KVM host. The **default** storage pool is usually located in **/var/lib/libvirt/images**. Only directory pool storage types can be managed through Satellite.
- Optional: The examples in these procedures use the root user for KVM. If you want to use a non-root user on the KVM server, you must add the user to the **libvirt** group on the KVM server:

```
useradd -a -G libvirt non_root_user
```

- A Satellite user account with the following roles:
  - **Edit hosts**
  - **View hosts**
    For more information, see Assigning Roles to a User in the *Administering Red Hat Satellite* guide.
- A custom role in Satellite with the following permissions:
  - **view_compute_resources**
  - **destroy_compute_resources_vms**
  - **power_compute_resources_vms**
  - **create_compute_resources_vms**
  - **view_compute_resources_vms**
  - **view_locations**
  - **view_subnets**

For more information about creating roles, see Creating a Role in the *Administering Red Hat Satellite* guide. For more information about adding permissions to a role, see Adding Permissions to a Role in the *Administering Red Hat Satellite* guide.

Procedure Overview

1. Section 9.1, "Configuring Satellite Server for KVM Connections".
2. Section 9.2, "Adding a KVM Connection to Satellite Server" .
3. Optional: Section 9.3, "Adding KVM Images to Satellite Server" . Use this procedure if you want to use image-based provisioning.
4. Section 9.4, "Adding KVM Details to a Compute Profile" .
5. Section 9.5, "Creating Hosts on KVM" .

9.1. CONFIGURING SATELLITE SERVER FOR KVM CONNECTIONS

Before adding the KVM connection, create an SSH key pair for the **foreman** user to ensure a secure connection between Satellite Server and KVM.

Procedure

1. On Satellite Server, switch to the **foreman** user:

```
# su foreman -s /bin/bash
```

2. Generate the key pair:

```
$ ssh-keygen
```

3. Copy the public key to the KVM server:

```
$ ssh-copy-id root@kvm.example.com
```

4. Exit the bash shell for the **foreman** user:

```
$ exit
```

5. Install the **libvirt-client** package:

```
# satellite-maintain packages install libvirt-client
```

6. Use the following command to test the connection to the KVM server:

```
# su foreman -s /bin/bash -c 'virsh -c qemu+ssh://root@kvm.example.com/system list'
```

9.2. ADDING A KVM CONNECTION TO SATELLITE SERVER

Use this procedure to add KVM as a compute resource in Satellite. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click **Create Compute Resource**.
2. In the **Name** field, enter a name for the new compute resource.
3. From the **Provider** list, select **Libvirt**.
4. In the **Description** field, enter a description for the compute resource.
5. In the **URL** field, enter the connection URL to the KVM server. For example:

```
qemu+ssh://root@kvm.example.com/system
```

6. From the **Display type** list, select either **VNC** or **Spice**.
7. Optional: To secure console access for new hosts with a randomly generated password, select the **Set a randomly generated password on the display connection** check box. You can retrieve the password for the VNC console to access the guest virtual machine console from the output of the following command executed on the KVM server:

```
# virsh edit your_VM_name
<graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0'
passwd='your_randomly_generated_password'>
```

   The password is randomly generated every time the console for the virtual machine is opened, for example, with virt-manager.
8. Click **Test Connection** to ensure that Satellite Server connects to the KVM server without fault.
9. Verify that the **Locations** and **Organizations** tabs are automatically set to your current context. If you want, add additional contexts to these tabs.
10. Click **Submit** to save the KVM connection.

CLI procedure

- To create a compute resource, enter the **hammer compute-resource create** command:

```
# hammer compute-resource create --name "My_KVM_Server" \
--provider "Libvirt" --description "KVM server at kvm.example.com" \
--url "qemu+ssh://root@kvm.example.com/system" --locations "New York" \
--organizations "My_Organization"
```

9.3. ADDING KVM IMAGES TO SATELLITE SERVER

To create hosts using image-based provisioning, you must add information about the image, such as access details and the image location, to your Satellite Server.

Note that you can manage only directory pool storage types through Satellite 6.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click the name of the KVM connection.
2. Click **Create Image**.
3. In the **Name** field, enter a name for the image.
4. From the **Operating System** list, select the image's base operating system.
5. From the **Architecture** list, select the operating system architecture.
6. In the **Username** field, enter the SSH user name for image access. This is normally the **root** user.
7. In the **Password** field, enter the SSH password for image access.
8. In the **Image path** field, enter the full path that points to the image on the KVM server. For example:

```
/var/lib/libvirt/images/TestImage.qcow2
```

9. Optional: Select the **User Data** check box if the image supports user data input, such as **cloud-init** data.
10. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Use the **--uuid** field to store the full path of the image location on the KVM server.

```
# hammer compute-resource image create \
--name "KVM Image" \
--compute-resource "My_KVM_Server"
--operatingsystem "RedHat version" \
--architecture "x86_64" \
--username root \
--user-data false \
--uuid "/var/lib/libvirt/images/KVMimage.qcow2" \
```

9.4. ADDING KVM DETAILS TO A COMPUTE PROFILE

Use this procedure to add KVM hardware settings to a compute profile. When you create a host on KVM using this compute profile, these settings are automatically populated.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles**.
2. In the Compute Profiles window, click the name of an existing compute profile, or click **Create Compute Profile**, enter a **Name**, and click **Submit**.
3. Click the name of the KVM compute resource.

4. In the **CPUs** field, enter the number of CPUs to allocate to the new host.
5. In the **Memory** field, enter the amount of memory to allocate to the new host.
6. From the **Image** list, select the image to use if performing image-based provisioning.
7. From the **Network Interfaces** list, select the network parameters for the host's network interface. You can create multiple network interfaces. However, at least one interface must point to a Capsule-managed network.
8. In the **Storage** area, enter the storage parameters for the host. You can create multiple volumes for the host.
9. Click **Submit** to save the settings to the compute profile.

CLI procedure

1. To create a compute profile, enter the following command:

```
# hammer compute-profile create --name "Libvirt CP"
```

2. To add the values for the compute profile, enter the following command:

```
# hammer compute-profile values create --compute-profile "Libvirt CP" \
--compute-resource "My_KVM_Server" \
--interface
"compute_type=network,compute_model=virtio,compute_network=examplenetwork" \
--volume "pool_name=default,capacity=20G,format_type=qcow2" \
--compute-attributes "cpus=1,memory=1073741824"
```

9.5. CREATING HOSTS ON KVM

In Satellite, you can use KVM provisioning to create hosts over a network connection or from an existing image:

- If you want to create a host over a network connection, the new host must be able to access either Satellite Server's integrated Capsule or an external Capsule Server on a KVM virtual network, so that the host has access to PXE provisioning services. This new host entry triggers the KVM server to create and start a virtual machine. If the virtual machine detects the defined Capsule Server through the virtual network, the virtual machine boots to PXE and begins to install the chosen operating system.
- If you want to create a host with an existing image, the new host entry triggers the KVM server to create the virtual machine using a pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

DHCP Conflicts

For network-based provisioning, if you use a virtual network on the KVM server for provisioning, select a network that does not provide DHCP assignments. This causes DHCP conflicts with Satellite Server when booting new hosts.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter a name for the host.
3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the KVM connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine settings.
7. Click the **Interface** tab and click **Edit** on the host's interface.
8. Verify that the fields are automatically populated, particularly the following items:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.
   - The **MAC address** field is blank. The KVM server assigns a MAC address to the host.
   - The **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.
   - The KVM-specific fields are populated with settings from your compute profile. Modify these settings if required.
9. Click the **Operating System** tab, and confirm that all fields automatically contain values.
10. Select the **Provisioning Method** that you want to use:
    - For network-based provisioning, click **Network Based**.
    - For image-based provisioning, click **Image Based**.
11. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.
12. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your needs.
13. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.
14. Click **Submit** to save the host entry.

CLI procedure

- To use network-based provisioning, create the host with the **hammer host create** command and include **--provision-method build**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "kvm-host1" \
```

```
--organization "My_Organization" \
--location "New York" \
--hostgroup "Base" \
--compute-resource "My_KVM_Server" \
--provision-method build \
--build true \
--enabled true \
--managed true \
--interface
"managed=true,primary=true,provision=true,compute_type=network,compute_network=exam
plenetwork" \
--compute-attributes="cpus=1,memory=1073741824" \
--volume="pool_name=default,capacity=20G,format_type=qcow2" \
--root-password "password"
```

- To use image-based provisioning, create the host with the **hammer host create** command and include **--provision-method image**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "kvm-host2" \
--organization "My_Organization" \
--location "New York" \
--hostgroup "Base" \
--compute-resource "My_KVM_Server" \
--provision-method image \
--image "KVM Image" \
--enabled true \
--managed true \
--interface
"managed=true,primary=true,provision=true,compute_type=network,compute_network=exampl
enetwork" \
--compute-attributes="cpus=1,memory=1073741824" \
--volume="pool_name=default,capacity=20G,format_type=qcow2"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

CHAPTER 10. PROVISIONING VIRTUAL MACHINES ON RED HAT VIRTUALIZATION

Red Hat Virtualization is an enterprise-grade server and desktop virtualization platform. In Red Hat Satellite, you can manage virtualization functions through Red Hat Virtualization's REST API. This includes creating virtual machines and controlling their power states.

You can use Red Hat Virtualization provisioning to create virtual machines over a network connection or from an existing image.

You can use **cloud-init** to configure the virtual machines that you provision. Using **cloud-init** avoids any special configuration on the network, such as a managed DHCP and TFTP, to finish the installation of the virtual machine. This method does not require Satellite to connect to the provisioned virtual machine using SSH to run the finish script.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- A Capsule Server managing a logical network on the Red Hat Virtualization environment. Ensure no other DHCP services run on this network to avoid conflicts with Capsule Server. For more information, see Configuring Networking.
- An existing template, other than the **blank** template, if you want to use image-based provisioning. For more information about creating templates for virtual machines, see Templates in the *Red Hat Virtualization Virtual Machine Management Guide*.
- An administration-like user on Red Hat Virtualization for communication with Satellite Server. Do not use the **admin@internal** user for this communication. Instead, create a new Red Hat Virtualization user with the following permissions:
  - **System** > **Configure System** > **Login Permissions**
  - **Network** > **Configure vNIC Profile** > **Create**
  - **Network** > **Configure vNIC Profile** > **Edit Properties**
  - **Network** > **Configure vNIC Profile** > **Delete**
  - **Network** > **Configure vNIC Profile** > **Assign vNIC Profile to VM**
  - **Network** > **Configure vNIC Profile** > **Assign vNIC Profile to Template**
  - **Template** > **Provisioning Operations** > **Import/Export**
  - **VM** > **Provisioning Operations** > **Create**
  - **VM** > **Provisioning Operations** > **Delete**
  - **VM** > **Provisioning Operations** > **Import/Export**
  - **VM** > **Provisioning Operations** > **Edit Storage**

- **Disk** > **Provisioning Operations** > **Create**
- **Disk** > **Disk Profile** > **Attach Disk Profile**

For more information about how to create a user and add permissions in Red Hat Virtualization, see Administering User Tasks From the Administration Portal in the *Red Hat Virtualization Administration Guide*.

Procedure Overview

1. Section 10.1, "Adding the Red Hat Virtualization Connection to Satellite Server".
2. Optional: Section 10.2, "Preparing Cloud-init Images in Red Hat Virtualization". Use this procedure if you want to use **cloud-init** to configure an image-based virtual machine.
3. Optional: Section 10.3, "Adding Red Hat Virtualization Images to Satellite Server". Use this procedure if you want to use image-based provisioning.
4. Optional: Section 10.4, "Preparing a Cloud-init Template". Use this procedure if you want to use **cloud-init** to configure an image-based virtual machine.
5. Section 10.5, "Adding Red Hat Virtualization Details to a Compute Profile".
6. Section 10.6, "Creating Hosts on Red Hat Virtualization".

10.1. ADDING THE RED HAT VIRTUALIZATION CONNECTION TO SATELLITE SERVER

Use this procedure to add Red Hat Virtualization as a compute resource in Satellite. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click **Create Compute Resource**.
2. In the **Name** field, enter a name for the new compute resource.
3. From the **Provider** list, select **RHV**.
4. In the **Description** field, enter a description for the compute resource.
5. In the **URL** field, enter the connection URL for the Red Hat Virtualization Manager's API in the following form: **https://rhv.example.com/ovirt-engine/api/v4**.
6. In the **User** field, enter the name of a user with permissions to access Red Hat Virtualization Manager's resources.
7. In the **Password** field, enter the password of the user.
8. Click **Load Datacenters** to populate the **Datacenter** list with data centers from your Red Hat Virtualization environment.
9. From the **Datacenter** list, select a data center.
10. From the **Quota ID** list, select a quota to limit resources available to Satellite.

11. In the **X509 Certification Authorities** field, enter the certificate authority for SSL/TLS access. Alternatively, if you leave the field blank, a self-signed certificate is generated on the first API request by the server.
12. Click the **Locations** tab and select the location you want to use.
13. Click the **Organizations** tab and select the organization you want to use.
14. Click **Submit** to save the compute resource.

CLI procedure

- Enter the **hammer compute-resource create** command with **Ovirt** for **--provider** and the name of the data center you want to use for **--datacenter**.

```
# hammer compute-resource create \
--name "My_RHV" --provider "Ovirt" \
--description "RHV server at rhv.example.com" \
--url "https://rhv.example.com/ovirt-engine/api" \
--user "Satellite_User" --password "My_Password" \
--locations "New York" --organizations "My_Organization" \
--datacenter "My_Datacenter"
```

10.2. PREPARING CLOUD-INIT IMAGES IN RED HAT VIRTUALIZATION

To use **cloud-init** during provisioning, you must prepare an image with **cloud-init** installed in Red Hat Virtualization, and then import the image to Satellite to use for provisioning.

Procedure

1. In Red Hat Virtualization, create a virtual machine to use for image-based provisioning in Satellite.
2. On the virtual machine, install **cloud-init**:

```
yum install cloud-init
```

3. To the **/etc/cloud/cloud.cfg** file, add the following information:

```
datasource_list: ["NoCloud", "ConfigDrive"]
```

4. In Red Hat Virtualization, create an image from this virtual machine.

When you add this image to Satellite, ensure that you select the **User Data** check box.

10.3. ADDING RED HAT VIRTUALIZATION IMAGES TO SATELLITE SERVER

To create hosts using image-based provisioning, you must add information about the image, such as access details and the image location, to your Satellite Server.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click the name of the Red Hat Virtualization connection.
2. Click **Create Image**.
3. In the **Name** field, enter a name for the image.
4. From the **Operating System** list, select the image's base operating system.
5. From the **Architecture** list, select the operating system architecture.
6. In the **Username** field, enter the SSH user name for image access. This is normally the **root** user.
7. In the **Password** field, enter the SSH password for image access.
8. From the **Image** list, select an image from the Red Hat Virtualization compute resource.
9. Optional: Select the **User Data** check box if the image supports user data input, such as **cloud-init** data.
10. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Use the **--uuid** option to store the template UUID on the Red Hat Virtualization server.

```
# hammer compute-resource image create \
--name "RHV_Image" \
--compute-resource "My_RHV"
--operatingsystem "RedHat version" \
--architecture "x86_64" \
--username root \
--uuid "9788910c-4030-4ae0-bad7-603375dd72b1" \
```

10.4. PREPARING A CLOUD-INIT TEMPLATE

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Provisioning Templates**, and click **Create Template**.
2. In the **Name** field, enter a name for the template.
3. In the **Editor** field, enter the following template details:

```
<%#
kind: user_data
name: Cloud-init
-%>
#cloud-config
hostname: <%= @host.shortname %>

<%# Allow user to specify additional SSH key as host paramter -%>
```

```
<% if @host.params['sshkey'].present? ||
@host.params['remote_execution_ssh_keys'].present? -%>
ssh_authorized_keys:
<% if @host.params['sshkey'].present? -%>
  - <%= @host.params['sshkey'] %>
<% end -%>
<% if @host.params['remote_execution_ssh_keys'].present? -%>
<% @host.params['remote_execution_ssh_keys'].each do |key| -%>
  - <%= key %>
<% end -%>
<% end -%>
<% end -%>
runcmd:
  - |
    #!/bin/bash
<%= indent 4 do
    snippet 'subscription_manager_registration'
end %>
<% if @host.info['parameters']['realm'] && @host.realm && @host.realm.realm_type == 'Red
Hat Identity Management' -%>
  <%= indent 4 do
    snippet 'idm_register'
  end %>
<% end -%>
<% unless @host.operatingsystem.atomic? -%>
    # update all the base packages from the updates repository
    yum -t -y -e 0 update
<% end -%>
<%
    # safemode renderer does not support unary negation
    non_atomic = @host.operatingsystem.atomic? ? false : true
    pm_set = @host.puppetmaster.empty? ? false : true
    puppet_enabled = non_atomic && (pm_set || @host.params['force-puppet'])
%>
<% if puppet_enabled %>
    yum install -y puppet
    cat > /etc/puppet/puppet.conf << EOF
  <%= indent 4 do
    snippet 'puppet.conf'
  end %>
    EOF
    # Setup puppet to run on system reboot
    /sbin/chkconfig --level 345 puppet on

    /usr/bin/puppet agent --config /etc/puppet/puppet.conf --onetime --tags no_such_tag <%=
@host.puppetmaster.blank? ? '' : "--server #{@host.puppetmaster}" %> --no-daemonize
    /sbin/service puppet start
<% end -%>
phone_home:
 url: <%= foreman_url('built') %>
 post: []
 tries: 10pp
```

4. Click the **Type** tab and from the **Type** list, select **User data template**.

5. Click the **Association** tab, and from the **Applicable Operating Systems** list, select the operating system that you want associate with the template.
6. Click the **Locations** tab, and from the **Locations** list, select the location that you want to associate with the template.
7. Click the **Organizations** tab, and from the **Organization** list, select the organization that you want to associate with the template.
8. Click **Submit**.
9. Navigate to **Hosts** > **Operating Systems**, and select the operating system you want to associate with the template.
10. Click the **Templates** tab, and from the **User data template** list, select the name of the new template.
11. Click **Submit**.

10.5. ADDING RED HAT VIRTUALIZATION DETAILS TO A COMPUTE PROFILE

Use this procedure to add Red Hat Virtualization hardware settings to a compute profile. When you create a host on KVM using this compute profile, these settings are automatically populated.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles**.
2. In the Compute Profiles window, click the name of an existing compute profile, or click **Create Compute Profile**, enter a **Name**, and click **Submit**.
3. Click the name of the Red Hat Virtualization compute resource.
4. From the **Cluster** list, select the target host cluster in the Red Hat Virtualization environment.
5. From the **Template** list, select the RHV template to use for the **Cores** and **Memory** settings.
6. In the **Cores** field, enter the number of CPU cores to allocate to the new host.
7. In the **Memory** field, enter the amount of memory to allocate to the new host.
8. From the **Image** list, select image to use for image-based provisioning.
9. In the **Network Interfaces** area, enter the network parameters for the host's network interface. You can create multiple network interfaces. However, at least one interface must point to a Capsule-managed network. For each network interface, enter the following details:
    a. In the **Name** field, enter the name of the network interface.
    b. From the **Network** list, select The logical network that you want to use.
10. In the **Storage** area, enter the storage parameters for the host. You can create multiple volumes for the host. For each volume, enter the following details:

a. In the **Size (GB)** enter the size, in GB, for the new volume.

b. From the **Storage domain** list, select the storage domain for the volume.

c. From the **Preallocate disk**, select either thin provisioning or preallocation of the full disk.

d. From the **Bootable** list, select whether you want a bootable or non-bootable volume.

11. Click **Submit** to save the compute profile.

CLI procedure

1. To create a compute profile, enter the following command:

```
# hammer compute-profile create --name "Red Hat Virtualization CP"
```

2. To set the values for the compute profile, enter the following command:

```
# hammer compute-profile values create --compute-profile "Red Hat Virtualization CP" \
--compute-resource "My_RHV" \
--interface
"compute_interface=Interface_Type,compute_name=eth0,compute_network=satnetwork" \
--volume "size_gb=20G,storage_domain=Data,bootable=true" \
--compute-attributes "cluster=Default,cores=1,memory=1073741824,start=true""
```

10.6. CREATING HOSTS ON RED HAT VIRTUALIZATION

In Satellite, you can use Red Hat Virtualization provisioning to create hosts over a network connection or from an existing image:

- If you want to create a host over a network connection, the new host must be able to access either Satellite Server's integrated Capsule or an external Capsule Server on a Red Hat Virtualization virtual network, so that the host has access to PXE provisioning services. This new host entry triggers the Red Hat Virtualization server to create and start a virtual machine. If the virtual machine detects the defined Capsule Server through the virtual network, the virtual machine boots to PXE and begins to install the chosen operating system.

- If you want to create a host with an existing image, the new host entry triggers the Red Hat Virtualization server to create the virtual machine using a pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

DHCP Conflicts

For network-based provisioning, if you use a virtual network on the Red Hat Virtualization server for provisioning, select a network that does not provide DHCP assignments. This causes DHCP conflicts with Satellite Server when booting new hosts.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.

2. In the **Name** field, enter a name for the host.

3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the Red Hat Virtualization connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine settings.
7. Click the **Interface** tab and click **Edit** on the host's interface.
8. Verify that the fields are automatically populated, particularly the following items:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.
   - The **MAC address** field is blank. The Red Hat Virtualization server assigns a MAC address to the host.
   - The **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.
   - The Red Hat Virtualization-specific fields are populated with settings from your compute profile. Modify these settings if required.
9. Click the **Operating System** tab, and confirm that all fields automatically contain values.
10. Select the **Provisioning Method** that you want to use:
    - For network-based provisioning, click **Network Based**.
    - For image-based provisioning, click **Image Based**.
11. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.
12. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your needs.
13. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.
14. Click **Submit** to save the host entry.

CLI procedure

- To use network-based provisioning, create the host with the **hammer host create** command and include **--provision-method build**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "RHV-vm1" \
--organization "My_Organization" \
--location "New York" \
--hostgroup "Base" \
```

```
--compute-resource "My_RHV" \
--provision-method build \
--build true \
--enabled true \
--managed true \
--interface
"managed=true,primary=true,provision=true,compute_name=eth0,compute_network=satnetwor
k" \
--compute-attributes="cluster=Default,cores=1,memory=1073741824,start=true" \
--volume="size_gb=20G,storage_domain=Data,bootable=true"
```

- To use image-based provisioning, create the host with the **hammer host create** command and include **--provision-method image**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "RHV-vm2" \
--organization "My_Organization" \
--location "New York" \
--hostgroup "Base" \
--compute-resource "My_RHV" \
--provision-method image \
--image "RHV_Image" \
--enabled true \
--managed true \
--interface
"managed=true,primary=true,provision=true,compute_name=eth0,compute_network=satnetwor
k" \
--compute-attributes="cluster=Default,cores=1,memory=1073741824,start=true" \
--volume="size_gb=20G,storage_domain=Data,bootable=true"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

CHAPTER 11. PROVISIONING VIRTUAL MACHINES IN VMWARE VSPHERE

VMware vSphere is an enterprise-level virtualization platform from VMware. Red Hat Satellite 6 can interact with the vSphere platform, including creating new virtual machines and controlling their power management states.

11.1. PREREQUISITES FOR VMWARE VSPHERE PROVISIONING

The requirements for VMware vSphere provisioning include:

- A Capsule Server managing a network on the vSphere environment. Ensure no other DHCP services run on this network to avoid conflicts with Capsule Server. For more information, see Chapter 3, *Configuring Networking*.
- An existing VMware template if you want to use image-based provisioning.
- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.

11.2. CREATING A VMWARE VSPHERE USER

The VMware vSphere server requires an administration-like user for Satellite Server communication. For security reasons, do not use the **administrator** user for such communication. Instead, create a user with the following permissions:

For VMware vCenter Server version 6.10, set the following permissions:

- All Privileges → Datastore → Allocate Space, Browse datastore, Update Virtual Machine files, Low level file operations
- All Privileges → Network → Assign Network
- All Privileges → Resource → Assign virtual machine to resource pool
- All Privileges → Virtual Machine → Change Config (All)
- All Privileges → Virtual Machine → Interaction (All)
- All Privileges → Virtual Machine → Edit Inventory (All)
- All Privileges → Virtual Machine → Provisioning (All)

Note that the same steps also apply to VMware vCenter Server version 7.0.

For VMware vCenter Server version 6.5, set the following permissions:

- All Privileges → Datastore → Allocate Space, Browse datastore, Update Virtual Machine files, Low level file operations
- All Privileges → Network → Assign Network

- All Privileges → Resource → Assign virtual machine to resource pool
- All Privileges → Virtual Machine → Configuration (All)
- All Privileges → Virtual Machine → Interaction (All)
- All Privileges → Virtual Machine → Inventory (All)
- All Privileges → Virtual Machine → Provisioning (All)

11.3. ADDING A VMWARE VSPHERE CONNECTION TO SATELLITE SERVER

Use this procedure to add a VMware vSphere connection in Satellite Server's compute resources. To use the CLI instead of the web UI, see the CLI procedure.

Ensure that the host and network-based firewalls are configured to allow Satellite to vCenter communication on TCP port 443. Verify that Satellite is able to resolve the host name of vCenter and vCenter is able to resolve Satellite Server's host name.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources**, and in the Compute Resources window, click **Create Compute Resource**
2. In the **Name** field, enter a name for the resource.
3. From the **Provider** list, select **VMware**.
4. In the **Description** field, enter a description for the resource.
5. In the **VCenter/Server** field, enter the IP address or host name of the vCenter server.
6. In the **User** field, enter the user name with permission to access the vCenter's resources.
7. In the **Password** field, enter the password for the user.
8. Click **Load Datacenters** to populate the list of data centers from your VMware vSphere environment.
9. From the **Datacenter** list, select a specific data center to manage from this list.
10. In the **Fingerprint** field, ensure that this field is populated with the fingerprint from the data center.
11. From the **Display Type** list, select a console type, for example, **VNC** or **VMRC**. Note that VNC consoles are unsupported on VMware ESXi 6.5 and later.
12. Optional: In the **VNC Console Passwords** field, select the **Set a randomly generated password on the display connection** check box to secure console access for new hosts with a randomly generated password. You can retrieve the password for the VNC console to access guest virtual machine console from the **libvirtd** host from the output of the following command:

```
# virsh edit your_VM_name
<graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0'
passwd='your_randomly_generated_password'>
```

The password randomly generates every time the console for the virtual machine opens, for example, with virt-manager.

13. From the **Enable Caching** list, you can select whether to enable caching of compute resources. For more information, see Section 11.8, "Caching of Compute Resources".

14. Click the **Locations** and **Organizations** tabs and verify that the values are automatically set to your current context. You can also add additional contexts.

15. Click **Submit** to save the connection.

CLI procedure

- Create the connection with the **hammer compute-resource create** command. Select **Vmware** as the **--provider** and set the instance UUID of the data center as the **--uuid**:

```
# hammer compute-resource create --name "My_vSphere" \
--provider "Vmware" \
--description "vSphere server at vsphere.example.com" \
--server "vsphere.example.com" --user "My_User" \
--password "My_Password" --locations "My_Location" --organizations "My_Organization" \
--datacenter "My_Datacenter"
```

11.4. ADDING VMWARE VSPHERE IMAGES TO SATELLITE SERVER

VMware vSphere uses templates as images for creating new virtual machines. If using image-based provisioning to create new hosts, you need to add VMware template details to your Satellite Server. This includes access details and the template name.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and in the Compute Resources window, click the VMware vSphere connection.

2. In the **Name** field, enter a name for the image.

3. From the **Operatingsystem** list, select the image's base operating system.

4. From the **Architecture** list, select the operating system architecture.

5. In the **User** field, enter the SSH user name for image access. This is normally the **root** user.

6. In the **Password** field, enter the SSH password for image access.

7. From the **User data** list, select whether you want the images to support user data input, such as **cloud-init** data.

8. In the **Image** field, enter the relative path and name of the template on the vSphere environment. Do not include the data center in the relative path.

9. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Use the **--uuid** field to store the relative template path on the vSphere environment.

```
# hammer compute-resource image create --name "Test_vSphere_Image" \
--operatingsystem "RedHat 7.2" --architecture "x86_64" \
--username root --uuid "Templates/RHEL72" \
--compute-resource "My_vSphere"
```

11.5. ADDING VMWARE VSPHERE DETAILS TO A COMPUTE PROFILE

You can predefine certain hardware settings for virtual machines on VMware vSphere. You achieve this through adding these hardware settings to a compute profile. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles** and, in the Compute Profiles window, click the name of the compute profile, and then click the vSphere connection.
2. In the **CPUs** field, enter the number of CPUs to allocate to the new host.
3. In the **Cores per socket** field, enter the number of cores to allocate to each CPU.
4. In the **Memory** field, enter the amount of memory to allocate to the new host.
5. In the **Cluster** field, enter the name of the target host cluster on the VMware environment.
6. From the **Resource pool** list, select an available resource allocations for the host.
7. In the **Folder** field, enter the folder to organize the host.
8. From the **Guest OS** list, select the operating system you want to use in VMware vSphere.
9. From the **SCSI controller** list, select the disk access method for the host.
10. From the **Virtual H/W version** list, select the underlying VMware hardware abstraction to use for virtual machines.
11. You can select the **Memory hot add** or **CPU hot add** check boxes if you want to add more resources while the virtual machine is powered.
12. From the **Image** list, select the image to use if performing image-based provisioning.
13. From the **Network Interfaces** list, select the network parameters for the host's network interface. You can create multiple network interfaces. However, at least one interface must point to a Capsule-managed network.
14. Select the **Eager zero** check box if you want to use eager zero thick provisioning. If unchecked, the disk uses lazy zero thick provisioning.
15. Click **Submit** to save the compute profile.

CLI procedure

1. To create the compute profile, enter the following command:

```
# hammer compute-profile create --name "VMWare CP"
```

2. To add the values for the compute profile, enter the following command:

```
# hammer compute-profile values create --compute-profile "VMWare CP" \
--compute-resource "My_vSphere" \
--interface "compute_type=VirtualE1000,compute_network=mynetwork \
--volume "size_gb=20G,datastore=Data,name=myharddisk,thin=true" \
--compute-attributes
"cpus=1,corespersocket=2,memory_mb=1024,cluster=MyCluster,path=MyVMs,start=true"
```

11.6. CREATING HOSTS ON A VMWARE VSPHERE SERVER

The VMware vSphere provisioning process provides the option to create hosts over a network connection or using an existing image.

For network-based provisioning, you must create a host to access either Satellite Server's integrated Capsule or an external Capsule Server on a VMware vSphere virtual network, so that the host has access to PXE provisioning services. The new host entry triggers the VMware vSphere server to create the virtual machine. If the virtual machine detects the defined Capsule Server through the virtual network, the virtual machine boots to PXE and begins to install the chosen operating system.

DHCP Conflicts

If you use a virtual network on the VMware vSphere server for provisioning, ensure that you select a virtual network that does not provide DHCP assignments. This causes DHCP conflicts with Satellite Server when booting new hosts.

For image-based provisioning, use the pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter the name that you want to become the provisioned system's host name.
3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the VMware vSphere connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine-based settings.
7. Click the **Interface** tab and click **Edit** on the host's interface.
8. Verify that the fields are automatically populated with values. Note in particular:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - Satellite Server automatically assigns an IP address for the new host.

9. Ensure that the **MAC address** field is blank. The VMware vSphere server assigns one to the host.

10. Verify that the **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.

11. In the interface window, review the VMware vSphere-specific fields that are populated with settings from our compute profile. Modify these settings to suit your needs.

12. Click the **Operating System** tab, and confirm that all fields automatically contain values.

13. Select the Provisioning Method that you want:

    - For network-based provisioning, click **Network Based**.
    - For image-based provisioning, click **Image Based**.
    - For boot-disk provisioning, click **Boot disk based**.

14. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.

15. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your requirements.

16. Click the **Parameters** tab and ensure that a parameter exists that provides an activation key. If not, add an activation key.

17. Click **Submit** to save the host entry.

CLI procedure

- Create the host from a network with the **hammer host create** command and include **--provision-method build** to use network-based provisioning.

```
# hammer host create --name "vmware-test1" --organization "My_Organization" \
--location "New York" --hostgroup "Base" \
--compute-resource "My_vSphere" --provision-method build \
--build true --enabled true --managed true \
--interface
"managed=true,primary=true,provision=true,compute_type=VirtualE1000,compute_network=my
network" \
--compute-
attributes="cpus=1,corespersocket=2,memory_mb=1024,cluster=MyCluster,path=MyVMs,start=
true" \
--volume="size_gb=20G,datastore=Data,name=myharddisk,thin=true"
```

- Create the host from an image with the **hammer host create** command and include **--provision-method image** to use image-based provisioning.

```
# hammer host create --name "vmware-test2" --organization "My_Organization" \
--location "New York" --hostgroup "Base" \
--compute-resource "My_VMware" --provision-method image \
--image "Test VMware Image" --enabled true --managed true \
--interface
"managed=true,primary=true,provision=true,compute_type=VirtualE1000,compute_network=my
```

```
network" \
--compute-
attributes="cpus=1,corespersocket=2,memory_mb=1024,cluster=MyCluster,path=MyVMs,start=
true" \
--volume="size_gb=20G,datastore=Data,name=myharddisk,thin=true"
```

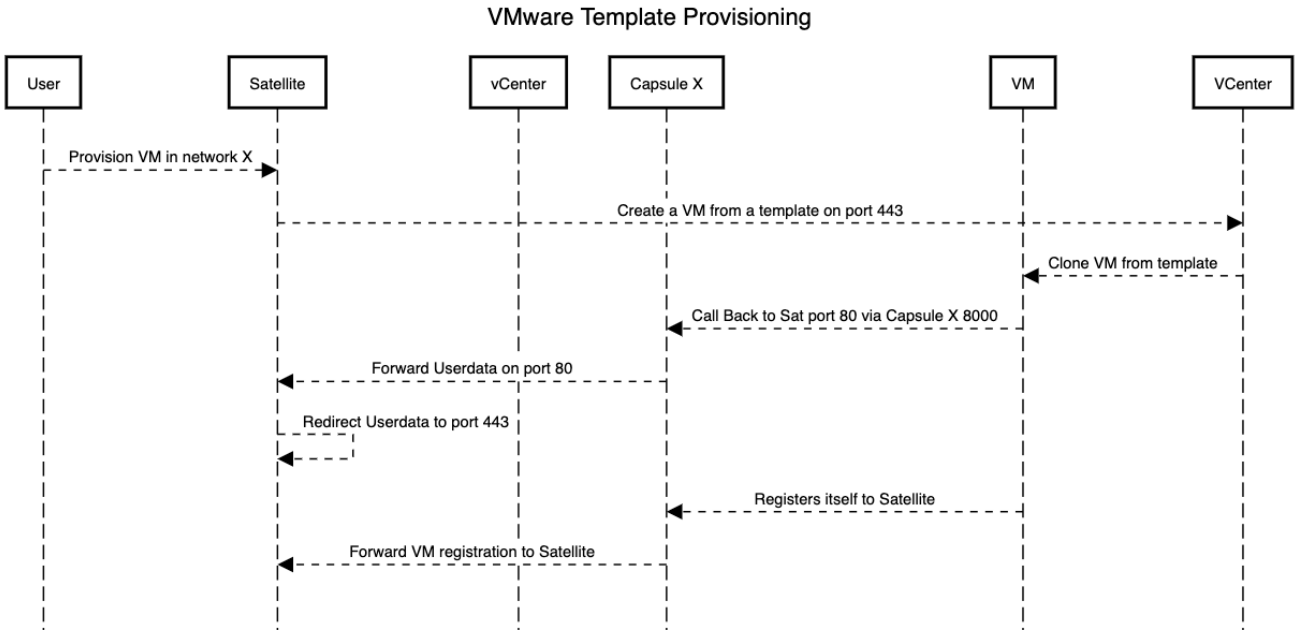
For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

11.7. USING THE VMWARE VSPHERE CLOUD-INIT AND USERDATA TEMPLATES FOR PROVISIONING

You can use VMware with the **Cloud-init** and **Userdata** templates to insert user data into the new virtual machine, to make further VMware customization, and to enable the VMware-hosted virtual machine to call back to Satellite.

You can use the same procedures to set up a VMware compute resource within Satellite, with a few modifications to the work flow.

VMware cloud-init Provisioning Overview



When you set up the compute resource and images for VMware provisioning in Satellite, the following sequence of provisioning events occur:

- The user provisions one or more virtual machines using the Satellite web UI, API, or hammer
- Satellite calls the VMware vCenter to clone the virtual machine template
- Satellite **userdata** provisioning template adds customized identity information
- When provisioning completes, the **Cloud-init** provisioning template instructs the virtual machine to call back to Capsule when **cloud-init** runs
- VMware vCenter clones the template to the virtual machine
- VMware vCenter applies customization for the virtual machine's identity, including the host name, IP, and DNS

- The virtual machine builds, **cloud-init** is invoked and calls back Satellite on port **80**, which then redirects to **443**

Port and Firewall Requirements

Because of the **cloud-init** service, the virtual machine always calls back to Satellite even if you register the virtual machine to Capsule. Ensure that you configure port and firewall settings to open any necessary connections.

For more information about port and firewall requirements, see Port and Firewall Requirements in the *Installing Satellite* and Ports and Firewalls Requirements in *Installing Capsule Server*.

Associating the userdata and Cloud-init Templates with the Operating System

1. In the Satellite web UI, navigate to **Hosts** > **Operating Systems**, and select the operating system that you want to use for provisioning.
2. Click the **Template** tab.
3. From the **Cloud-init template** list, select **Cloudinit default**.
4. From the **User data template** list, select **UserData open-vm-tools**.
5. Click **Submit** to save the changes.

Preparing an Image to use the cloud-init Template

To prepare an image, you must first configure the settings that you require on a virtual machine that you can then save as an image to use in Satellite.

To use the **cloud-init** template for provisioning, you must configure a virtual machine so that **cloud-init** is installed, enabled, and configured to call back to Satellite Server.

For security purposes, you must install a CA certificate to use HTTPs for all communication. This procedure includes steps to clean the virtual machine so that no unwanted information transfers to the image you use for provisioning.

If you have an image with **cloud-init**, you must still follow this procedure to enable **cloud-init** to communicate with Satellite because **cloud-init** is disabled by default.

1. On the virtual machine that you use to create the image, install **cloud-init**, **open-vm-tools**, and **perl**:

```
# yum -y install cloud-init open-vm-tools perl
```

2. Disable network configuration by **cloud-init**:

```
# cat << EOM > /etc/cloud/cloud.cfg.d/01_network.cfg
network:
  config: disabled
EOM
```

3. Configure **cloud-init** to fetch data from Satellite:

```
# cat << EOM > /etc/cloud/cloud.cfg.d/10_datasource.cfg
datasource_list: [NoCloud]
```

```
datasource:
  NoCloud:
    seedfrom: https://satellite.example.com/userdata/
EOM
```

4. Configure modules to use in **cloud-init**:

```
# cat << EOM > /etc/cloud/cloud.cfg
cloud_init_modules:
 - bootcmd

cloud_config_modules:
 - runcmd

cloud_final_modules:
 - scripts-per-once
 - scripts-per-boot
 - scripts-per-instance
 - scripts-user
 - phone-home

system_info:
  distro: rhel
  paths:
    cloud_dir: /var/lib/cloud
    templates_dir: /etc/cloud/templates
  ssh_svcname: sshd
EOM
```

5. Enable the CA certificates for the image:

```
# update-ca-trust enable
```

6. Download the **katello-server-ca.crt** file from Satellite Server:

```
# wget -O /etc/pki/ca-trust/source/anchors/cloud-init-ca.crt
http://satellite.example.com/pub/katello-server-ca.crt
```

7. To update the record of certificates, enter the following command:

```
# update-ca-trust extract
```

8. Use the following commands to clean the image:

```
# systemctl stop rsyslog
# systemctl stop auditd
# package-cleanup --oldkernels --count=1
# yum clean all
```

9. Use the following commands to reduce logspace, remove old logs, and truncate logs:

```
# logrotate -f /etc/logrotate.conf
# rm -f /var/log/*-???????? /var/log/*.gz
# rm -f /var/log/dmesg.old
```

```
# rm -rf /var/log/anaconda
# cat /dev/null > /var/log/audit/audit.log
# cat /dev/null > /var/log/wtmp
# cat /dev/null > /var/log/lastlog
# cat /dev/null > /var/log/grubby
```

10. Remove **udev** hardware rules:

```
# rm -f /etc/udev/rules.d/70*
```

11. Remove the **ifcfg** scripts related to existing network configurations:

```
# rm -f /etc/sysconfig/network-scripts/ifcfg-ens*
# rm -f /etc/sysconfig/network-scripts/ifcfg-eth*
```

12. Remove the SSH host keys:

```
# rm -f /etc/ssh/SSH_keys
```

13. Remove root user's SSH history:

```
# rm -rf ~root/.ssh/known_hosts
```

14. Remove root user's shell history:

```
# rm -f ~root/.bash_history
# unset HISTFILE
```

You can now create an image from this virtual machine.

You can use the Section 11.4, "Adding VMware vSphere Images to Satellite Server" section to add the image to Satellite.

Configuring Capsule to Forward the user data Template

If you deploy Satellite with the Capsule templates feature, you must configure Satellite to recognize hosts' IP addresses forwarded over the X-Forwarded-For HTTP header to serve correct template payload.

For security reasons, Satellite recognizes this HTTP header only from localhost. For each individual Capsule, you must configure a regular expression to recognize hosts' IP addresses. From the web UI, you can do this by navigating to **Administer** > **Settings** > **Provisioning**, and changing the **Remote address** setting. From the CLI, you can do this by entering the following command:

```
# hammer settings set --name remote_addr --value '(localhost(4|6|4to6)?|192.168.122.(1|2|3))'
```

11.8. CACHING OF COMPUTE RESOURCES

Caching of compute resources speeds up rendering of VMware information.

11.8.1. Enabling Caching of Compute Resources

To enable or disable caching of compute resources:

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources**.
2. Click the **Edit** button to the right of the VMware server you want to update.
3. Select the **Enable caching** check box.

11.8.2. Refreshing the Compute Resources Cache

To refresh the cache of compute resources to update compute resources information:

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources**.
2. Select a VMware server you want to refresh the compute resources cache for and click the **Refresh Cache** button.

CLI procedure

Use this API call to refresh the compute resources cache:

```
# curl -H "Accept:application/json,version=2" \
-H "Content-Type:application/json" -X PUT \
-u username:password -k \
https://satellite.example.com/api/compute_resources/compute_resource_id/refresh_cache
```

Use the **hammer compute-resource list** command to determine the ID of the VMware server you want to refresh the compute resources cache for.

CHAPTER 12. PROVISIONING VIRTUAL MACHINES ON CONTAINER-NATIVE VIRTUALIZATION

Container-native Virtualization addresses the needs of development teams that have adopted or want to adopt Kubernetes but possess existing virtual machine (VM)-based workloads that cannot be easily containerized. This technology provides a unified development platform where developers can build, modify, and deploy applications residing in application containers and VMs in a shared environment. These capabilities support rapid application modernization across the open hybrid cloud.

With Red Hat Satellite, you can create a compute resource for Container-native Virtualization so that you can provision and manage Kubernetes virtual machines using Satellite.

Note that template provisioning is not supported for this release.

IMPORTANT

The Container-native Virtualization compute resource is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see https://access.redhat.com/support/offerings/techpreview/.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- A Container-native Virtualization user that has the **cluster-admin** permissions for the Openshift Container Platform virtual cluster. For more information, see Using RBAC to Define and Apply Permissions in the *Authentication* guide of the Openshift Container Platform documentation.
- A Capsule Server managing a network on the Container-native Virtualization server. Ensure that no other DHCP services run on this network to avoid conflicts with Capsule Server. For more information about network service configuration for Capsule Servers, see Configuring Networking.
- A Satellite user account with the following roles:
  - **Edit hosts**
  - **View hosts**
    For more information, see Assigning Roles to a User in the *Administering Red Hat Satellite* guide.
- A custom role in Satellite with the following permissions:
  - **view_compute_resources**
  - **destroy_compute_resources_vms**

- **power_compute_resources_vms**
- **create_compute_resources_vms**
- **view_compute_resources_vms**
- **view_locations**
- **view_subnets**
  For more information about creating roles, see Creating a Role in the *Administering Red Hat Satellite* guide. For more information about adding permissions to a role, see Adding Permissions to a Role in the *Administering Red Hat Satellite* guide.

12.1. ADDING A CONTAINER-NATIVE VIRTUALIZATION CONNECTION TO SATELLITE SERVER

Use this procedure to add Container-native Virtualization as a compute resource in Satellite.

Procedure

1. Enter the following **satellite-installer** command to enable the Container-native Virtualization plugin for Satellite:

```
# satellite-installer --enable-foreman-plugin-kubevirt
```

2. Generate a bearer token to use for HTTP and HTTPs authentication. On the Container-native Virtualization server, list the secrets that contain tokens:

```
# kubectl get secrets
```

3. List the token for your secret:

```
# kubectl get secrets YOUR_SECRET -o jsonpath='{.data.token}' | base64 -d | xargs
```

   Make a note of this token to use later in this procedure.

4. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources**, and click **Create Compute Resource**.
5. In the **Name** field, enter a name for the new compute resource.
6. From the **Provider** list, select **KubeVirt**.
7. In the **Description** field, enter a description for the compute resource.
8. In the **Hostname** field, enter the address of the Container-native Virtualization server that you want to use.
9. In the **API Port** field, enter the port number that you want to use for provisioning requests from Satellite to Container-native Virtualization.
10. In the **Namespace** field, enter the user name of the Container-native Virtualization virtual cluster that you want to use.
11. In the **Token** field, enter the bearer token for HTTP and HTTPs authentication.

12. Optional: In the **X509 Certification Authorities** field, enter a certificate to enable client certificate authentication for API server calls.

CHAPTER 13. PROVISIONING CLOUD INSTANCES ON RED HAT OPENSTACK PLATFORM

Red Hat OpenStack Platform provides the foundation to build a private or public Infrastructure-as-a-Service (IaaS) cloud. It offers a massively scalable, fault-tolerant platform for the development of cloud-enabled workloads. In Red Hat Satellite 6, you can interact with Red Hat OpenStack Platform REST API to create cloud instances and control their power management states.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- A Capsule Server managing a network in your OpenStack environment. For more information, see Configuring Networking.
- An image added to OpenStack Image Storage (glance) service for image-based provisioning. For more information, see the Red Hat OpenStack Platform *Instances and Images Guide* .

Procedure Overview

1. Section 13.1, “Adding the Red Hat OpenStack Platform Connection to Satellite Server” .
2. Section 13.2, “Adding Red Hat OpenStack Platform Images to Satellite Server” .
3. Section 13.3, “Adding Red Hat OpenStack Platform Details to a Compute Profile” .
4. Section 13.4, “Creating Image-based Hosts on Red Hat OpenStack Platform” .

13.1. ADDING THE RED HAT OPENSTACK PLATFORM CONNECTION TO SATELLITE SERVER

Use this procedure to add Red Hat OpenStack Platform as a compute resource in Satellite. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click **Create Compute Resource**.
2. In the **Name** field, enter a name for the new compute resource.
3. From the **Provider** list, select **RHEL OpenStack Platform**.
4. In the **Description** field, enter a description for the compute resource.
5. In the **URL** field, enter the URL for the OpenStack Authentication keystone service’s API at the **tokens** resource. Use the following format: **http://openstack.example.com:5000/v3.0/tokens**.
6. In the **User** and **Password** fields, enter the authentication user and password for Satellite to access the environment.

7. In the **Domain** field, enter the domain for V3 authentication.
8. From the **Tenant** list, select the tenant or project for Satellite Server to manage.
9. To use external networks as primary networks for hosts, select the **Allow external network as main network** check box.
10. Click the **Locations** and **Organizations** tabs and verify that the location and organization that you want to use are set to your current context. Add any additional contexts that you want to these tabs.
11. Click **Submit** to save the Red Hat OpenStack Platform connection.

CLI procedure

- To create a compute resource, enter the **hammer compute-resource create** command:

```
# hammer compute-resource create --name "My_OpenStack" \
--provider "OpenStack" \
--description "My OpenStack environment at openstack.example.com" \
--url "http://openstack.example.com:5000/v3.0/tokens" --user "My_Username" \
--password "My_Password" --tenant "openstack" --locations "New York" \
--organizations "My_Organization"
```

13.2. ADDING RED HAT OPENSTACK PLATFORM IMAGES TO SATELLITE SERVER

To create hosts using image-based provisioning, you must add information about the image, such as access details and the image location, to your Satellite Server.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click the name of the Red Hat OpenStack Platform connection.
2. Click **Create Image**.
3. In the **Name** field, enter a name for the image.
4. From the **Operating System** list, select the image's base operating system.
5. From the **Architecture** list, select the operating system architecture.
6. In the **Username** field, enter the SSH user name for image access. This is normally the **root** user.
7. In the **Password** field, enter the SSH password for image access.
8. From the **Image** list, select an image from the Red Hat OpenStack Platform compute resource.
9. Optional: Select the **User Data** check box if the image supports user data input, such as **cloud-init** data.
10. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Use the **--uuid** field to store the full path of the image location on the Red Hat OpenStack Platform server.

```
# hammer compute-resource image create \
--name "OpenStack Image" \
--compute-resource "My_OpenStack_Platform"
--operatingsystem "RedHat version" \
--architecture "x86_64" \
--username root \
--user-data true \
--uuid "/path/to/OpenstackImage.qcow2"
```

13.3. ADDING RED HAT OPENSTACK PLATFORM DETAILS TO A COMPUTE PROFILE

Use this procedure to add Red Hat OpenStack Platform hardware settings to a compute profile. When you create a host on Red Hat OpenStack Platform using this compute profile, these settings are automatically populated.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles**.
2. In the Compute Profiles window, click the name of an existing compute profile, or click **Create Compute Profile**, enter a **Name**, and click **Submit**.
3. Click the name of the Red Hat OpenStack Platform compute resource.
4. From the **Flavor** list, select the hardware profile on Red Hat OpenStack Platform to use for the host.
5. From the **Availability zone** list, select the target cluster to use within the Red Hat OpenStack Platform environment.
6. From the **Image** list, select the image to use for image-based provisioning.
7. From the **Tenant** list, select the tenant or project for the Red Hat OpenStack Platform instance.
8. From the **Security Group** list, select the cloud-based access rules for ports and IP addresses.
9. From the **Internal network**, select the private networks for the host to join.
10. From the **Floating IP network**, select the external networks for the host to join and assign a floating IP address.
11. From the **Boot from volume**, select whether a volume is created from the image. If not selected, the instance boots the image directly.
12. In the **New boot volume size (GB)** field, enter the size, in GB, of the new boot volume.
13. Click **Submit** to save the compute profile.

CLI procedure

The compute profile CLI commands are not yet implemented in Red Hat Satellite 6.10. As an alternative, you can include the same settings directly during the host creation process.

13.4. CREATING IMAGE-BASED HOSTS ON RED HAT OPENSTACK PLATFORM

In Satellite, you can use Red Hat OpenStack Platform provisioning to create hosts from an existing image. The new host entry triggers the Red Hat OpenStack Platform server to create the instance using the pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter a name for the host.
3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the Red Hat OpenStack Platform connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine settings.
7. From the **Lifecycle Environment** list, select the environment.
8. Click the **Interfaces** tab and click **Edit** on the host's interface.
9. Verify that the fields are automatically populated, particularly the following items:
    - The **Name** from the **Host** tab becomes the **DNS name**.
    - The **MAC address** field is blank. Red Hat OpenStack Platform assigns a MAC address to the host during provisioning.
    - Satellite Server automatically assigns an IP address for the new host.
    - The **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.
10. Click the **Operating System** tab, and confirm that all fields automatically contain values.
11. If you want to change the image that populates automatically from your compute profile, from the **Images** list, select a different image to base the new host's root volume on.
12. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.
13. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your needs.

14. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.

15. Click **Submit** to save the host entry.

CLI procedure

- Create the host with the **hammer host create** command and include **--provision-method image**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "openstack-host1" \
--organization "My_Organization" \
--location "New York" \
--hostgroup "Base" \
--compute-resource "My_OpenStack_Platform" \
--provision-method image \
--image "OpenStack Image" \
--enabled true \
--managed true \
--interface "managed=true,primary=true,provision=true" \
--compute-
attributes="flavor_ref=m1.small,tenant_id=openstack,security_groups=default,network=mynetw
ork"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

CHAPTER 14. PROVISIONING CLOUD INSTANCES IN AMAZON EC2

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides public cloud compute resources. Using Satellite, you can interact with Amazon EC2's public API to create cloud instances and control their power management states. Use the procedures in this chapter to add a connection to an Amazon EC2 account and provision a cloud instance.

14.1. PREREQUISITES FOR AMAZON EC2 PROVISIONING

The requirements for Amazon EC2 provisioning include:

- A Capsule Server managing a network in your EC2 environment. Use a Virtual Private Cloud (VPC) to ensure a secure network between the hosts and Capsule Server.
- An Amazon Machine Image (AMI) for image-based provisioning.
- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.

14.2. ADDING AN AMAZON EC2 CONNECTION TO THE SATELLITE SERVER

Use this procedure to add the Amazon EC2 connection in Satellite Server's compute resources. To use the CLI instead of the web UI, see the CLI procedure.

Time Settings and Amazon Web Services

Amazon Web Services uses time settings as part of the authentication process. Ensure that Satellite Server's time is correctly synchronized. Ensure that an NTP service, such as **ntpd** or **chronyd**, is running properly on Satellite Server. Failure to provide the correct time to Amazon Web Services can lead to authentication failures.

For more information about synchronizing time in Satellite, see Synchronizing Time in *Installing Satellite Server from a Connected Network*.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and in the Compute Resources window, click **Create Compute Resource**.
2. In the **Name** field, enter a name to identify the Amazon EC2 compute resource.
3. From the **Provider** list, select **EC2**.
4. In the **Description** field, enter information that helps distinguish the resource for future use.
5. Optional: From the **HTTP proxy** list, select an HTTP proxy to connect to external API services. You must add HTTP proxies to Satellite before you can select a proxy from this list. For more information, see Section 14.3, "Using an HTTP Proxy with Compute Resources" .

6. In the **Access Key** and **Secret Key** fields, enter the access keys for your Amazon EC2 account. For more information, see Managing Access Keys for your AWS Account on the Amazon documentation website.
7. Optional: Click the **Load Regions** button to populate the **Regions** list.
8. From the **Region** list, select the Amazon EC2 region or data center to use.
9. Click the **Locations** tab and ensure that the location you want to use is selected, or add a different location.
10. Click the **Organizations** tab and ensure that the organization you want to use is selected, or add a different organization.
11. Click **Submit** to save the Amazon EC2 connection.
12. Select the new compute resource and then click the **SSH keys** tab, and click **Download** to save a copy of the SSH keys to use for SSH authentication. Until BZ1793138 is resolved, you can download a copy of the SSH keys only immediately after creating the Amazon EC2 compute resource. If you require SSH keys at a later stage, follow the procedure in Section 14.7, "Connecting to an Amazon EC2 instance using SSH".

CLI procedure

- Create the connection with the **hammer compute-resource create** command. Use **--user** and **--password** options to add the access key and secret key respectively.

```
# hammer compute-resource create --name "My_EC2" --provider "EC2" \
--description "Amazon EC2 Public Cloud` --user "user_name" \
--password "secret_key" --region "us-east-1" --locations "New York" \
--organizations "My_Organization"
```

14.3. USING AN HTTP PROXY WITH COMPUTE RESOURCES

In some cases, the EC2 compute resource that you use might require a specific HTTP proxy to communicate with Satellite. In Satellite, you can create an HTTP proxy and then assign the HTTP proxy to your EC2 compute resource.

However, if you configure an HTTP proxy for Satellite in **Administer** > **Settings**, and then add another HTTP proxy for your compute resource, the HTTP proxy that you define in **Administer** > **Settings** takes precedence.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **HTTP Proxies**, and select **New HTTP Proxy**.
2. In the **Name** field, enter a name for the HTTP proxy.
3. In the **URL** field, enter the URL for the HTTP proxy, including the port number.
4. Optional: Enter a username and password to authenticate to the HTTP proxy, if your HTTP proxy requires authentication.
5. Click **Test Connection** to ensure that you can connect to the HTTP proxy from Satellite.

6. Click the **Locations** tab and add a location.
7. Click the **Organization** tab and add an organization.
8. Click **Submit**.

14.4. ADDING AMAZON EC2 IMAGES TO SATELLITE SERVER

Amazon EC2 uses image-based provisioning to create hosts. You must add image details to your Satellite Server. This includes access details and image location.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and select an Amazon EC2 connection.
2. Click the **Images** tab, and then click **New Image**.
3. In the **Name** field, enter a name to identify the image for future use.
4. From the **Operating System** list, select the operating system that corresponds with the image you want to add.
5. From the **Architecture** list, select the operating system's architecture.
6. In the **Username** field, enter the SSH user name for image access. This is normally the **root** user.
7. In the **Password** field, enter the SSH password for image access.
8. In the **Image ID** field, enter the Amazon Machine Image (AMI) ID for the image. This is usually in the following format: **ami-xxxxxxxx**.
9. Optional: Select the **User Data** check box if the images support user data input, such as **cloud-init** data. If you enable user data, the Finish scripts are automatically disabled. This also applies in reverse: if you enable the Finish scripts, this disables user data.
10. Optional: In the **IAM role** field, enter the Amazon security role used for creating the image.
11. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Use the **--uuid** field to store the full path of the image location on the Amazon EC2 server.

```
# hammer compute-resource image create --name "Test Amazon EC2 Image" \
--operatingsystem "RedHat 7.2" --architecture "x86_64" --username root \
--user-data true --uuid "ami-my_ami_id" --compute-resource "My_EC2"
```

14.5. ADDING AMAZON EC2 DETAILS TO A COMPUTE PROFILE

You can add hardware settings for instances on Amazon EC2 to a compute profile.

Procedure

To add hardware settings, complete the following steps:

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles** and click the name of your profile, then click an EC2 connection.
2. From the **Flavor** list, select the hardware profile on EC2 to use for the host.
3. From the **Image** list, select the image to use for image-based provisioning.
4. From the **Availability zone** list, select the target cluster to use within the chosen EC2 region.
5. From the **Subnet** list, add the subnet for the EC2 instance. If you have a VPC for provisioning new hosts, use its subnet.
6. From the **Security Groups** list, select the cloud-based access rules for ports and IP addresses to apply to the host.
7. From the **Managed IP** list, select either a **Public** IP or a **Private** IP.
8. Click **Submit** to save the compute profile.

CLI procedure

The compute profile CLI commands are not yet implemented in Red Hat Satellite. As an alternative, you can include the same settings directly during the host creation process.

14.6. CREATING IMAGE-BASED HOSTS ON AMAZON EC2

The Amazon EC2 provisioning process creates hosts from existing images on the Amazon EC2 server. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter a name for the host.
3. From the **Host Group** list, you can select a host group to populate most of the new host's fields.
4. From the **Deploy on** list, select the EC2 connection.
5. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine-based settings.
6. Click the **Interface** tab, and then click **Edit** on the host's interface, and verify that the fields are populated with values. Leave the **Mac Address** field blank. Satellite Server automatically selects and IP address and the **Managed**, **Primary**, and **Provision** options for the first interface on the host.
7. Click the **Operating System** tab and confirm that all fields are populated with values.
8. Click the **Virtual Machine** tab and confirm that all fields are populated with values.
9. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.

10. Click **Submit** to save your changes.

This new host entry triggers the Amazon EC2 server to create the instance, using the pre-existing image as a basis for the new volume.

CLI procedure

- Create the host with the **hammer host create** command and include **--provision-method image** to use image-based provisioning.

```
# hammer host create --name "ec2-test1" --organization "My_Organization" \
--location "New York" --hostgroup "Base" \
--compute-resource "My_EC2" --provision-method image \
--image "Test Amazon EC2 Image" --enabled true --managed true \
--interface "managed=true,primary=true,provision=true,subnet_id=EC2" \
--compute-attributes="flavor_id=m1.small,image_id=TestImage,availability_zones=us-east-1a,security_group_ids=Default,managed_ip=Public"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

14.7. CONNECTING TO AN AMAZON EC2 INSTANCE USING SSH

You can connect remotely to an Amazon EC2 instance from Satellite Server using SSH. However, to connect to any Amazon Web Services EC2 instance that you provision through Red Hat Satellite, you must first access the private key that is associated with the compute resource in the Foreman database, and use this key for authentication.

Procedure

1. To locate the compute resource list, on your Satellite Server base system, enter the following command, and note the ID of the compute resource that you want to use:

```
# hammer compute-resource list
```

2. Switch user to the **postgres** user:

```
# su - postgres
```

3. Initiate the **postgres** shell:

```
$ psql
```

4. Connect to the Foreman database as the user **postgres**:

```
# postgres=# \c foreman
```

5. Select the secret from **key_pairs** where **compute_resource_id = 3**:

```
# select secret from key_pairs where compute_resource_id = 3; secret
```

6. Copy the key from after **-----BEGIN RSA PRIVATE KEY-----** until **-----END RSA PRIVATE KEY-----**.

7. Create a **.pem** file and paste your key into the file:

```
# vim Keyname.pem
```

8. Ensure that you restrict access to the **.pem** file:

```
# chmod 600 Keyname.pem
```

9. To connect to the Amazon EC2 instance, enter the following command:

```
ssh -i Keyname.pem ec2-user@example.aws.com
```

14.8. CONFIGURING A FINISH TEMPLATE FOR AN AMAZON WEB SERVICE EC2 ENVIRONMENT

You can use Red Hat Satellite finish templates during the provisioning of Red Hat Enterprise Linux instances in an Amazon EC2 environment.

If you want to use a Finish template with SSH, Satellite must reside within the EC2 environment and in the correct security group. Satellite currently performs SSH finish provisioning directly, not using Capsule Server. If Satellite Server does not reside within EC2, the EC2 virtual machine reports an internal IP rather than the necessary external IP with which it can be reached.

Procedure

1. In the Red Hat Satellite web UI, navigate to **Hosts** > **Provisioning Templates**.
2. In the **Provisioning Templates** page, enter **Kickstart default finish** into the search field and click **Search**.
3. On the **Kickstart default finish** template, select **Clone**.
4. In the **Name** field, enter a unique name for the template.
5. In the template, prefix each command that requires root privileges with **sudo**, except for **subscription-manager register** and **yum** commands, or add the following line to run the entire template as the sudo user:

```
sudo -s << EOS
_Template_ _Body_
EOS
```

6. Click the **Association** tab, and associate the template with a Red Hat Enterprise Linux operating system that you want to use.
7. Click the **Locations** tab, and add the the location where the host resides.
8. Click the **Organizations** tab, and add the organization that the host belongs to.
9. Make any additional customizations or changes that you require, then click **Submit** to save your template.
10. Navigate to **Hosts** > **Operating systems** and select the operating system that you want for your host.

11. Click the **Templates** tab, and from the **Finish Template** list, select your finish template.
12. Navigate to **Hosts** > **Create Host** and enter the information about the host that you want to create.
13. Click the **Parameters** tab and navigate to **Host parameters**.
14. In **Host parameters**, click the **Add Parameter** button three times to add three new parameter fields. Add the following three parameters:
    a. In the **Name** field, enter **remote_execution_ssh_keys**. In the corresponding **Value** field, enter the output of **cat /usr/share/foreman-proxy/.ssh/id_rsa_foreman_proxy.pub**.
    b. In the **Name** field, enter **remote_execution_ssh_user**. In the corresponding **Value** field, enter **ec2-user**.
    c. In the **Name** field, enter **activation_keys**. In the corresponding **Value** field, enter your activation key.
15. Click **Submit** to save the changes.

14.9. MORE INFORMATION ABOUT AMAZON WEB SERVICES AND SATELLITE

For information about how to locate Red Hat Gold Images on Amazon Web Services EC2, see How to Locate Red Hat Cloud Access Gold Images on AWS EC2.

For information about how to install and use the Amazon Web Service Client on Linux, see Install the AWS Command Line Interface on Linux in the Amazon Web Services documentation.

For information about importing and exporting virtual machines in Amazon Web Services, see VM Import/Export in the Amazon Web Services documentation.

CHAPTER 15. PROVISIONING CLOUD INSTANCES ON GOOGLE COMPUTE ENGINE

Red Hat Satellite can interact with Google Compute Engine (GCE), including creating new virtual machines and controlling their power management states. You can only use golden images supported by Red Hat with Satellite for creating GCE hosts.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- In your GCE project, configure a service account with the necessary IAM Compute role. For more information, see Compute Engine IAM roles in the GCE documentation.
- In your GCE project-wise metadata, set the **enable-oslogin** to **FALSE**. For more information, see Enabling or disabling OS Login in the GCE documentation.
- Optional: If you want to use Puppet with GCE hosts, navigate to **Administer** > **Settings** > **Puppet** and enable the **Use UUID for certificates** setting to configure Puppet to use consistent Puppet certificate IDs.
- Based on your needs, associate a **finish** or **user_data** provisioning template with the operating system you want to use. For more information about provisioning templates, see Provisioning Templates.

Procedure Overview

1. Section 15.1, “Adding a Google Compute Engine Connection to Satellite Server” .
2. Section 15.2, “Adding Google Compute Engine Images to Satellite Server” .
3. Section 15.3, “Adding Google Compute Engine Details to a Compute Profile” .
4. Section 15.4, “Creating Image-based Hosts on Google Compute Engine” .

15.1. ADDING A GOOGLE COMPUTE ENGINE CONNECTION TO SATELLITE SERVER

Use this procedure to add Google Compute Engine (GCE) as a compute resource in Satellite. To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In GCE, generate a service account key in JSON format and upload this file to the **/usr/share/foreman/** directory on Satellite Server.
2. On Satellite Server, change the owner for the service account key to the **foreman** user:

```
# chown foreman /usr/share/foreman/gce_key.json
```

3. Configure permissions for the service account key to ensure that the file is readable:

```
# chmod 0600 /usr/share/foreman/gce_key.json
```

4. Restore SELinux context for the service account key:

```
# restorecon -vv /usr/share/foreman/gce_key.json
```

5. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click **Create Compute Resource**.
6. In the **Name** field, enter a name for the compute resource.
7. From the **Provider** list, select **Google**.
8. Optional: In the **Description** field, enter a description for the resource.
9. In the **Google Project ID** field, enter the project ID.
10. In the **Client Email** field, enter the client email.
11. In the **Certificate Path** field, enter the path to the service account key. For example, **/usr/share/foreman/*gce_key*.json**.
12. Click **Load Zones** to populate the list of zones from your GCE environment.
13. From the **Zone** list, select the GCE zone to use.
14. Click **Submit**.

CLI procedure

1. In GCE, generate a service account key in JSON format and upload this file to the **/usr/share/foreman/** directory on Satellite Server.
2. On Satellite Server, change the owner for the service account key to the **foreman** user:

```
# chown foreman /usr/share/foreman/gce_key.json
```

3. Configure permissions for the service account key to ensure that the file is readable:

```
# chmod 0600 /usr/share/foreman/gce_key.json
```

4. Restore SELinux context for the service account key:

```
# restorecon -vv /usr/share/foreman/gce_key.json
```

5. Use the **hammer compute-resource create** command to add a GCE compute resource to Satellite:

```
# hammer compute-resource create --name 'gce_cr' \
--provider 'gce' \
--project 'gce_project_id' \
```

```
--key-path 'gce_key.json' \
--zone 'us-west1-b' \
--email 'gce_email'
```

15.2. ADDING GOOGLE COMPUTE ENGINE IMAGES TO SATELLITE SERVER

To create hosts using image-based provisioning, you must add information about the image, such as access details and the image location, to your Satellite Server.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click the name of the Google Compute Engine connection.
2. Click **Create Image**.
3. In the **Name** field, enter a name for the image.
4. From the **Operating System** list, select the image's base operating system.
5. From the **Architecture** list, select the operating system architecture.
6. In the **Username** field, enter the SSH user name for image access. Specify a user other than **root**, because the **root** user cannot connect to a GCE instance using SSH keys. The username must begin with a letter and consist of lowercase letters and numbers.
7. From the **Image** list, select an image from the Google Compute Engine compute resource.
8. Optional: Select the **User Data** check box if the image supports user data input, such as **cloud-init** data.
9. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. With the **--username** option, specify a user other than **root**, because the **root** user cannot connect to a GCE instance using SSH keys. The username must begin with a letter and consist of lowercase letters and numbers.

```
# hammer compute-resource image create \
--name 'gce_image_name' \
--compute-resource 'gce_cr' \
--operatingsystem-id 1 \
--architecture-id 1 \
--uuid '3780108136525169178' \
--username 'admin'
```

15.3. ADDING GOOGLE COMPUTE ENGINE DETAILS TO A COMPUTE PROFILE

Use this procedure to add GCE hardware settings to a compute profile. When you create a host on GCE using this compute profile, these settings are automatically populated.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles**.
2. In the Compute Profiles window, click the name of an existing compute profile, or click **Create Compute Profile**, enter a **Name**, and click **Submit**.
3. Click the name of the GCE compute resource.
4. From the **Machine Type** list, select the machine type to use for provisioning.
5. From the **Image** list, select the image to use for provisioning.
6. From the **Network** list, select the GCE network to use for provisioning.
7. Optional: Select the **Associate Ephemeral External IP** check box to assign a dynamic ephemeral IP address that Satellite uses to communicate with the host. This public IP address changes when you reboot the host. If you need a permanent IP address, reserve a static public IP address on GCE and attach it to the host.
8. In the **Size (GB)** field, enter the size of the storage to create on the host.
9. Click **Submit** to save the compute profile.

CLI procedure

1. Create a compute profile to use with the GCE compute resource:

```
# hammer compute-profile create --name gce_profile
```

2. Add GCE details to the compute profile.

```
# hammer compute-profile values create --compute-profile gce_profile \
--compute-resource 'gce_cr' \
--volume "size_gb=20" \
--compute-attributes "machine_type=f1-micro,associate_external_ip=true,network=default"
```

15.4. CREATING IMAGE-BASED HOSTS ON GOOGLE COMPUTE ENGINE

In Satellite, you can use Google Compute Engine provisioning to create hosts from an existing image. The new host entry triggers the Google Compute Engine server to create the instance using the pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.

2. In the **Name** field, enter a name for the host.
3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the Google Compute Engine connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine settings.
7. From the **Lifecycle Environment** list, select the environment.
8. Click the **Interfaces** tab and click **Edit** on the host's interface.
9. Verify that the fields are automatically populated, particularly the following items:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - The **MAC address** field is blank. Google Compute Engine assigns a MAC address to the host during provisioning.
   - Satellite Server automatically assigns an IP address for the new host.
   - The **Domain** field is populated with the required domain.
   - The **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.
10. Click the **Operating System** tab, and confirm that all fields automatically contain values.
11. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.
12. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your needs.
13. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.
14. Click **Submit** to save the host entry.

CLI procedure

- Create the host with the **hammer host create** command and include **--provision-method image**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name "GCE_VM" \
--organization "Your_Organization" \
--location "Your_Location" \
--compute-resource gce_cr_name
--compute-profile "gce_profile_name" \
--provision-method 'image' \
--image gce_image_name \
```

```
--root-password "your_root_password" \
--interface "type=interface,domain_id=1,managed=true,primary=true,provision=true" \
--puppet-environment-id 1 \
--puppet-ca-proxy-id 1 \
--puppet-proxy-id 1 \
--architecture x86_64 \
--operatingsystem "operating_system_name"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

CHAPTER 16. PROVISIONING CLOUD INSTANCES ON MICROSOFT AZURE RESOURCE MANAGER

Red Hat Satellite can interact with Microsoft Azure Resource Manager, including creating new virtual machines and controlling their power management states. Only image-based provisioning is supported for creating Azure hosts. This includes provisioning using Marketplace images, custom images, and shared image gallery.

For more information about Azure Resource Manager concepts, see Azure Resource Manager documentation.

Prerequisites

- Synchronized content repositories for Red Hat Enterprise Linux. For more information, see Syncing Repositories in the *Content Management Guide*.
- An activation key for host registration. For more information, see Creating An Activation Key in the *Content Management* guide.
- Ensure that you have the correct permissions to create an Azure Active Directory application. For more information, see Check Azure AD permissions in the *Microsoft identity platform (Azure Active Directory for developers)* documentation.
- You must create and configure an Azure Active Directory application and service principle to obtain Application or *client* ID, Directory or *tenant* ID, and Client Secret. For more information, see Use the portal to create an Azure AD application and service principal that can access resources in the *Microsoft identity platform (Azure Active Directory for developers)* documentation.
- Optional: If you want to use Puppet with Azure hosts, navigate to **Administer** > **Settings** > **Puppet** and enable the **Use UUID for certificates** setting to configure Puppet to use consistent Puppet certificate IDs.
- Based on your needs, associate a **finish** or **user_data** provisioning template with the operating system you want to use. For more information about provisioning templates, see Provisioning Templates.
- Optional: If you want the virtual machine to use a static private IP address, create a subnet in Satellite with the **Network Address** field matching the Azure subnet address.
- Before creating RHEL BYOS images, you must accept the image terms either in the Azure CLI or Portal so that the image can be used to create and manage virtual machines for your subscription.

Procedure Overview

1. Section 16.1, “Adding a Microsoft Azure Resource Manager Connection to Satellite Server” .
2. Section 16.2, “Adding Microsoft Azure Resource Manager Images to Satellite Server” .
3. Section 16.3, “Adding Microsoft Azure Resource Manager Details to a Compute Profile” .
4. Section 16.4, “Creating Image-based Hosts on Microsoft Azure Resource Manager” .

16.1. ADDING A MICROSOFT AZURE RESOURCE MANAGER CONNECTION TO SATELLITE SERVER

Use this procedure to add Microsoft Azure Resource Manager as a compute resource in Satellite. Note that you must add a separate compute resource for each Azure Resource Manager region that you want to use.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click **Create Compute Resource**.
2. In the **Name** field, enter a name for the compute resource.
3. From the **Provider** list, select **Azure Resource Manager**.
4. Optional: In the **Description** field, enter a description for the resource.
5. By default, the **Cloud** is set to Public/Standard. Azure Government Cloud supports the following regions:
   - US Government
   - China
   - Germany
6. In the **Client ID** field, enter the Application or *client* ID.
7. In the **Client Secret** field, enter the client secret.
8. In the **Subscription ID** field, enter the subscription ID.
9. In the **Tenant ID** field, enter the Directory or *tenant* ID.
10. Click **Load Regions**. This tests if your connection to Azure Resource Manager is successful and loads the regions available in your subscription.
11. From the **Azure Region** list, select the Azure region to use.
12. Click **Submit**.

CLI procedure

- Use the **hammer compute-resource create** command to add an Azure compute resource to Satellite. Note that the value for the **--region** option must be in lowercase and must not contain special characters.

```
# hammer compute-resource create --name azure_cr_name \
--provider azurerm \
--tenant tenant-id \
--app-ident client-id \
```

```
--secret-key client-secret \
--sub-id subscription-id \
--region 'region'
```

IMPORTANT

If you are using Azure Government Cloud then you must pass in the **--cloud** parameter. The values for the **cloud** parameter are:

Name of Azure Government Cloud	Value for hammer --cloud
US Goverment	azureusgovernment
China	azurechina
Germany	azuregermancloud

16.2. ADDING MICROSOFT AZURE RESOURCE MANAGER IMAGES TO SATELLITE SERVER

To create hosts using image-based provisioning, you must add information about the image, such as access details and the image location, to your Satellite Server.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Resources** and click the name of the Microsoft Azure Resource Manager connection.
2. Click **Create Image**.
3. In the **Name** field, enter a name for the image.
4. From the **Operating System** list, select the image's base operating system.
5. From the **Architecture** list, select the operating system architecture.
6. In the **Username** field, enter the SSH user name for image access. You cannot use the **root** user.
7. Optional: In the **Password** field, enter a password to authenticate with.
8. In the **Azure Image Name** field, enter an image name in the format ***prefix://UUID***.
   - For a custom image, use the prefix **custom**. For example, **custom://image-name**.
   - For a shared gallery image, use the prefix **gallery**. For example, **gallery://image-name**.
   - For public and RHEL Bring Your Own Subscription (BYOS) images, use the prefix **marketplace**. For example, **marketplace://OpenLogicCentOS:7.5:latest**.

For more information, see Find Linux VM images in the Azure Marketplace with the Azure CLI.

9. Optional: Select the **User Data** check box if the image supports user data input, such as **cloud-init** data.
10. Click **Submit** to save the image details.

CLI procedure

- Create the image with the **hammer compute-resource image create** command. Note that the username that you enter for the image must be the same that you use when you create a host with this image. The **--password** option is optional when creating an image. You cannot use the **root** user.

```
# hammer compute-resource image create \
--name Azure_image_name \
--compute-resource azure_cr_name \
--uuid 'marketplace://RedHat:RHEL:7-RAW:latest' \
--username 'azure_username' \
--user-data no
```

16.3. ADDING MICROSOFT AZURE RESOURCE MANAGER DETAILS TO A COMPUTE PROFILE

Use this procedure to add Azure hardware settings to a compute profile. When you create a host on Azure using this compute profile, these settings are automatically populated.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Infrastructure** > **Compute Profiles**.
2. In the Compute Profiles window, click the name of an existing compute profile, or click **Create Compute Profile**, enter a **Name**, and click **Submit**.
3. Click the name of the Azure compute resource.
4. From the **Resource group** list, select the resource group to provision to.
5. From the **VM Size** list, select a size of a virtual machine to provision.
6. From the **Platform** list, select **Linux**.
7. In the **Username** field, enter a user name to authenticate with. Note that the username that you enter for compute profile must be the same that you use when creating an image.
8. To authenticate the user, use one of the following options:
   - To authenticate using a password, enter a password in the **Password** field.
   - To authenticate using an SSH key, enter an SSH key in the **SSH Key** field.

9. Optional: If you want the virtual machine to use a premium virtual machine disk, select the **Premium OS Disk** check box.
10. From the **OS Disk Caching** list, select the disc caching setting.
11. Optional: In the **Custom Script Command** field, enter commands to perform on the virtual machine when the virtual machine is provisioned.
12. Optional: If you want to run custom scripts when provisioning finishes, in the **Comma separated file URIs** field, enter comma-separated file URIs of scripts to use. The scripts must contain **sudo** at the beginning because Red Hat Satellite downloads files to the **/var/lib/waagent/custom-script/download/0/** directory on the host and scripts require sudo privileges to be executed.
13. Optional: You can add a **NVIDIA Driver** by selecting the **NVIDIA driver / CUDA** checkbox. For more information, refer to the following Microsoft Azure documentation:
    - NVIDIA GPU Driver Extension for Linux
    - NVIDIA GPU Driver Extension for Windows
14. Optional: If you want to create an additional volume on the VM, click the **Add Volume** button, enter the **Size** in GB and select the **Data Disk Caching** method.
    - Note that the maximum number of these disks depends on the VM Size selected. For more information on Microsoft Azure VM storage requirements, see the Microsoft Azure documentation.
15. Click **Add Interface**.

IMPORTANT

The maximum number of interfaces depends on the VM Size selected. For more information, see the Microsoft Azure documentation link above.

1. From the **Azure Subnet** list, select the Azure subnet to provision to.
2. From the **Public IP** list, select the public IP setting.
3. Optional: If you want the virtual machine to use a static private IP, select the **Static Private IP** check box.
4. Click **Submit**.

CLI procedure

1. Create a compute profile to use with the Azure Resource Manager compute resource:

```
# hammer compute-profile create --name compute_profile_name
```

2. Add Azure details to the compute profile. With the **username** setting, enter the SSH user name for image access. Note that the username that you enter for compute profile must be the same that you use when creating an image.

```
# hammer compute-profile values create \
--compute-profile "compute_profile_name" \
--compute-resource azure_cr_name \
```

```
--compute-
attributes="resource_group=resource_group,vm_size=Standard_B1s,username=azure_user,
password=azure_password,platform=Linux,script_command=touch /var/tmp/text.txt" \
--
interface="compute_public_ip=Dynamic,compute_network=mysubnetID,compute_private_ip=fal
se"
--volume="disk_size_gb=5,data_disk_caching=None"
```

Optional: If you want to run scripts on the virtual machine after provisioning, specify the following settings:

- To enter the script directly, with the **script_command** setting, enter a command to be executed on the provisioned virtual machine.
- To run a script from a URI, with the **script_uris** setting, enter comma-separated file URIs of scripts to use. The scripts must contain **sudo** at the beginning because Red Hat Satellite downloads files to the **/var/lib/waagent/custom-script/download/0/** directory on the host and therefore scripts require sudo privileges to be executed.

16.4. CREATING IMAGE-BASED HOSTS ON MICROSOFT AZURE RESOURCE MANAGER

In Satellite, you can use Microsoft Azure Resource Manager provisioning to create hosts from an existing image. The new host entry triggers the Microsoft Azure Resource Manager server to create the instance using the pre-existing image as a basis for the new volume.

To use the CLI instead of the web UI, see the CLI procedure.

Procedure

1. In the Satellite web UI, navigate to **Hosts** > **Create Host**.
2. In the **Name** field, enter a name for the host.
3. Click the **Organization** and **Location** tabs to ensure that the provisioning context is automatically set to the current context.
4. From the **Host Group** list, select the host group that you want to use to populate the form.
5. From the **Deploy on** list, select the Microsoft Azure Resource Manager connection.
6. From the **Compute Profile** list, select a profile to use to automatically populate virtual machine settings.
7. From the **Lifecycle Environment** list, select the environment.
8. Click the **Interfaces** tab and click **Edit** on the host's interface.
9. Verify that the fields are automatically populated, particularly the following items:
   - The **Name** from the **Host** tab becomes the **DNS name**.
   - The **MAC address** field is blank. Microsoft Azure Resource Manager assigns a MAC address to the host during provisioning.
   - The **Azure Subnet** field is populated with the required Azure subnet.

- The **Managed**, **Primary**, and **Provision** options are automatically selected for the first interface on the host. If not, select them.

10. Optional: If you want to use a static private IP address, from the **IPv4 Subnet** list select the Satellite subnet with the **Network Address** field matching the Azure subnet address. In the **IPv4 Address** field, enter an IPv4 address within the range of your Azure subnet.
11. Click the **Operating System** tab, and confirm that all fields automatically contain values.
12. For **Provisioning Method**, ensure **Image Based** is selected.
13. From the **Image** list, select the Azure Resource Manager image that you want to use for provisioning.
14. In the **Root Password** field, enter the root password to authenticate with.
15. Click **Resolve** in **Provisioning templates** to check the new host can identify the right provisioning templates to use.
16. Click the **Virtual Machine** tab and confirm that these settings are populated with details from the host group and compute profile. Modify these settings to suit your needs.
17. Click the **Parameters** tab, and ensure that a parameter exists that provides an activation key. If not, add an activation key.
18. Click **Submit** to save the host entry.

CLI procedure

- Create the host with the **hammer host create** command and include **--provision-method image**. Replace the values in the following example with the appropriate values for your environment.

```
# hammer host create \
--name="Azure_VM" \
--organization "Your_Organization" \
--location "Your_Location" \
--compute-resource azure_cr_name \
--compute-profile "compute_profile_name" \
--provision-method 'image' \
--image Azure_image_name \
--domain domain_name \
--architecture x86_64 \
--operatingsystem "operating_system_name"
```

For more information about additional host creation parameters for this compute resource, enter the **hammer host create --help** command.

APPENDIX A. INITIALIZATION SCRIPT FOR PROVISIONING EXAMPLES

If you have not followed the examples in the Red Hat Satellite 6 Content Management Guide, you can use the following initialization script to create an environment for provisioning examples.

Create a script file (**content-init.sh**) and include the following:

```
#!/bin/bash

MANIFEST=$1

# Import the content from Red Hat CDN
hammer organization create --name "ACME" --label "ACME" \
--description "Our example organization for managing content."
hammer subscription upload --file ~/$MANIFEST --organization "ACME"
hammer repository-set enable \
--name "Red Hat Enterprise Linux 7 Server (RPMs)" \
--releasever "7Server" --basearch "x86_64" \
--product "Red Hat Enterprise Linux Server" --organization "ACME"
hammer repository-set enable \
--name "Red Hat Enterprise Linux 7 Server (Kickstart)" \
--releasever "7Server" --basearch "x86_64" \
--product "Red Hat Enterprise Linux Server" --organization "ACME"
hammer repository-set enable \
--name "Red Hat Satellite Tools 6.10 (for RHEL 7 Server) (RPMs)" \
--basearch "x86_64" --product "Red Hat Enterprise Linux Server" \
--organization "ACME"
hammer product synchronize --name "Red Hat Enterprise Linux Server" \
--organization "ACME"

# Create our application life cycle
hammer lifecycle-environment create --name "Development" \
--description "Environment for ACME's Development Team" \
--prior "Library" --organization "ACME"
hammer lifecycle-environment create --name "Testing" \
--description "Environment for ACME's Quality Engineering Team" \
--prior "Development" --organization "ACME"
hammer lifecycle-environment create --name "Production" \
--description "Environment for ACME's Product Releases" \
--prior "Testing" --organization "ACME"

# Create and publish our Content View
hammer content-view create --name "Base" \
--description "Base operating system" \
--repositories "Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server,Red Hat Satellite Tools 6.10
for RHEL 7 Server RPMs x86_64" \
--organization "ACME"
hammer content-view publish --name "Base" \
--description "Initial content view for our operating system" \
--organization "ACME"
hammer content-view version promote --content-view "Base" --version 1 \
--to-lifecycle-environment "Development" --organization "ACME"
hammer content-view version promote --content-view "Base" --version 1 \
```

```
--to-lifecycle-environment "Testing" --organization "ACME"
hammer content-view version promote --content-view "Base" --version 1 \
--to-lifecycle-environment "Production" --organization "ACME"
```

Set executable permissions on the script:

```
# chmod +x content-init.sh
```

Download a copy of your Subscription Manifest from the Red Hat Customer Portal and run the script on the manifest:

```
# ./content-init.sh manifest_98f4290e-6c0b-4f37-ba79-3a3ec6e405ba.zip
```

This imports the necessary Red Hat content for the provisioning examples in this guide.

APPENDIX B. PROVISIONING FIPS-COMPLIANT HOSTS

Satellite supports provisioning hosts that comply with the National Institute of Standards and Technology's Security Requirements for Cryptographic Modules standard, reference number FIPS 140-2, referred to here as FIPS.

To enable the provisioning of hosts that are FIPS-compliant, complete the following tasks:

- Change the provisioning password hashing algorithm for the operating system
- Create a host group and set a host group parameter to enable FIPS

For more information about creating host groups, see Creating a Host Group in the *Managing Hosts* guide.

The provisioned hosts have the FIPS-compliant settings applied. To confirm that these settings are enabled, complete the steps in Section B.3, "Verifying FIPS Mode is Enabled".

B.1. CHANGE THE PROVISIONING PASSWORD HASHING ALGORITHM

To provision FIPS-compliant hosts, you must first set the password hashing algorithm that you use in provisioning to SHA256. This configuration setting must be applied for each operating system you want to deploy as FIPS-compliant.

1. Identify the Operating System IDs.

```
$ hammer os list
```

2. Update each operating system's password hash value.

```
$ hammer os update --title Operating_System \
  --password-hash SHA256
```

3. Repeat this command for each of the operating systems, using the matching value in the **TITLE** column:

```
$ hammer os update --title "RedHat version_number" \
  --password-hash SHA256
```

Note that you cannot use a comma-separated list of values.

B.2. SETTING THE FIPS-ENABLED PARAMETER

To provision a FIPS-compliant host, you must create a host group and set the host group parameter **fips_enabled** to **true**. If this is not set to **true**, or is absent, the FIPS-specific changes do not apply to the system. You can set this parameter when you provision a host or for a host group.

To set this parameter when provisioning a host, append **--parameters fips_enabled=true** to the Hammer command.

```
$ hammer hostgroup set-parameter --name fips_enabled \
 --value 'true' \
 --hostgroup prod_servers
```

For more information, see the output of the command **hammer hostgroup set-parameter --help**.

B.3. VERIFYING FIPS MODE IS ENABLED

To verify these FIPS compliance changes have been successful, you must provision a host and check its configuration.

1. Log on to the host as **root** or with an admin-level account.
2. Enter the following command:

```
$ cat /proc/sys/crypto/fips_enabled
```

A value of **1** confirms that FIPS mode is enabled.

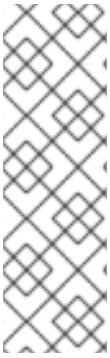
APPENDIX C. BUILDING CLOUD IMAGES FOR RED HAT SATELLITE

Use this section to build and register images to Red Hat Satellite.

You can use a preconfigured Red Hat Enterprise Linux KVM guest QCOW2 image:

- Latest RHEL 7 KVM Guest Image
- Latest RHEL 6 KVM Guest Image

These images contain **cloud-init**. To function properly, they must use ec2-compatible metadata services for provisioning an SSH key.



NOTE

For the KVM guest images:

- The **root** account in the image is disabled, but **sudo** access is granted to a special user named **cloud-user**.
- There is no **root** password set for this image.

The **root** password is locked in **/etc/shadow** by placing **!!** in the second field.

If you want to create custom Red Hat Enterprise Linux images, see Creating a Red Hat Enterprise Linux 7 Image and Creating a Red Hat Enterprise Linux 6 Image .

C.1. CREATING CUSTOM RED HAT ENTERPRISE LINUX IMAGES

Prerequisites:

- Use a Linux host machine to create an image. In this example, we use a Red Hat Enterprise Linux 7 Workstation.
- Use **virt-manager** on your workstation to complete this procedure. If you create the image on a remote server, connect to the server from your workstation with **virt-manager**.
- A Red Hat Enterprise Linux 7 or 6 ISO file (see Red Hat Enterprise Linux 7.4 Binary DVD or Red Hat Enterprise Linux 6.9 Binary DVD).

For more information about installing a Red Hat Enterprise Linux Workstation, see Red Hat Enterprise Linux 7 Installation Guide.

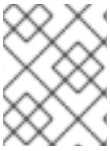
Before you can create custom images, install the following packages:

- Install **libvirt**, **qemu-kvm** and graphical tools:

```
[root@host]# yum install virt-manager virt-viewer libvirt qemu-kvm
```

- Install the following command line tools:

```
[root@host]# yum install virt-install libguestfs-tools-c
```



NOTE

In the following procedures, enter all commands with the **[root@host]#** prompt on the workstation that hosts the **libvirt** environment.

C.2. CREATING A RED HAT ENTERPRISE LINUX 7 IMAGE

Use this section to create an image in the QCOW2 format using a Red Hat Enterprise Linux 7 ISO file.

1. Using your web browser, download the Red Hat Enterprise Linux binary ISO file to a temporary location, for example, the **Downloads** directory.
2. Copy the Red Hat Enterprise Linux binary ISO file to the **/var/lib/libvirt/images/** directory.

```
[root@host]# cp ~/home/user/Downloads/rhel-server-7.4-x86_64-dvd.iso
/var/lib/libvirt/images/
```

3. Verify that **virtbr0** is the virtual bridge:

```
[root@host]# ip a
```

4. Start **libvirtd**:

```
[root@host]# systemctl start libvirtd
```

5. Navigate to the **/var/lib/libvirt/images/** directory:

```
[root@host]# cd /var/lib/libvirt/images/
```

6. Prepare the QEMU image:

```
[root@host]# qemu-img create -f qcow2 rhel7.qcow2 8G
```

7. Start the installation using **virt-install**. Use the following example as a guide:

```
[root@host]# virt-install --virt-type qemu --name rhel7 --ram 2048 \
--cdrom rhel-server-7.4-x86_64-dvd.iso \
--disk rhel7.qcow2,format=qcow2 \
--network=bridge:virbr0 --graphics vnc,listen=0.0.0.0 \
--noautoconsole --os-type=linux --os-variant=rhel7
```

NOTE

For GUI users, if the instance does not launch automatically, enter the **virt-manager** command to view the console:

```
[root@host]# virt-manager
```

8. Follow the steps of the Red Hat Enterprise Linux installation wizard.
   a. For the installation source, add an HTTP link to your repository in Red Hat Satellite, for example
      satellite.example.com/pub/export/RHEL7/content/dist/rhel/server/7/7Server/x86_64/os/

satellite.example.com/pub/export/RHEL7/content/dist/rhel/server/7/7Server/x86_64/os/

b. For the type of devices your installation uses, select **Auto-detected installation media**

c. For the type of installation destination, select **Local Standard Disks**.

d. For other storage options, select **Automatically configure partitioning**.

e. For software selection, select **Minimal Install**.

f. Set the network interface to **ON** to ensure the interface activates on system start.

g. Enter a host name, and click **Apply**.

h. Enter a **root** password.

9. When the installation completes, reboot the instance and log in as the root user.

10. Confirm that the network interface is up and that the IP address is assigned:

```
# ip a
```

11. Confirm that the hostname is correct:

```
# hostname
```

12. Create a **/etc/NetworkManager/conf.d/XX-cloud-image.conf** file where *XX* is a two-digit number that indicates order of precedence. Add the following contents to the file:

```
[main]
dns=none
```

13. Proceed to Configuring a Host for Registration .

C.3. CREATING A RED HAT ENTERPRISE LINUX 6 IMAGE

Use this section to create an image in the QCOW2 format using a Red Hat Enterprise Linux 6 ISO file.

1. Start the installation using **virt-install**:

```
[root@host]# qemu-img create -f qcow2 rhel6.qcow2 4G
[root@host]# virt-install --connect=qemu:///system --network=bridge:virbr0 \
--name=rhel6 --os-type linux --os-variant rhel6 \
--disk path=rhel6.qcow2,format=qcow2,size=10,cache=none \
--ram 4096 --vcpus=2 --check-cpu --accelerate \
--hvm --cdrom=rhel-server-6.8-x86_64-dvd.iso
```

This launches an instance and starts the installation process.

NOTE

If the instance does not launch automatically, enter the **virt-viewer** command to view the console:

```
[root@host]# virt-viewer rhel6
```

2. Set up the virtual machines as follows:
   a. At the initial Installer boot menu, select the **Install or upgrade an existing system** option.
   b. Select the appropriate **Language** and **Keyboard** options.
   c. When prompted about which type of devices your installation uses, select **Basic Storage Devices**.
   d. Select a **hostname** for your device. The default host name is **localhost.localdomain**.
   e. Set a root password.
   f. Based on the space on the disk, select the type of installation.
   g. Select the **Basic Server** install, which includes an SSH server.
3. Reboot the instance and log in as the **root** user.
4. Update the **/etc/sysconfig/network-scripts/ifcfg-eth0** file so it only contains the following values:

```
TYPE=Ethernet
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

5. Restart the service network:

```
# service network restart
```

6. Proceed to Configuring a Host for Registration .

C.4. CONFIGURING A HOST FOR REGISTRATION

Red Hat Enterprise Linux virtual machines register to Customer Portal Subscription Management by default. You must update each virtual machine configuration so that they receive updates from the correct Satellite Server or Capsule Server.

Prerequisites

- Hosts must be using the following Red Hat Enterprise Linux version:
  - 6.4 or later
  - 7.0 or later

- All architectures of Red Hat Enterprise Linux are supported (i386, x86_64, s390x, ppc_64).
- Ensure that a time synchronization tool is enabled and runs on the Satellite Servers, any Capsule Servers, and the hosts.
  - For Red Hat Enterprise Linux 6:

    ```
    # chkconfig ntpd on; service ntpd start
    ```

  - For Red Hat Enterprise Linux 7:

    ```
    # systemctl enable chronyd; systemctl start chronyd
    ```

- Ensure that the daemon **rhsmcertd** is enabled and running on the hosts.
  - For Red Hat Enterprise Linux 6:

    ```
    # chkconfig rhsmcertd on; service rhsmcertd start
    ```

  - For Red Hat Enterprise Linux 7:

    ```
    # systemctl start rhsmcertd
    ```

To Configure a Host for Registration:

1. Take note of the fully qualified domain name (FQDN) of the Satellite Server or Capsule Server, for example *server.example.com*.
2. On the host, connect to a terminal on the host as the root user
3. Install the consumer RPM from Satellite Server or Capsule Server to which the host is to be registered. The consumer RPM updates the content source location of the host and allows the host to download content from the content source specified in Red Hat Satellite.

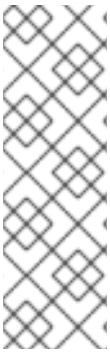
   ```
   # rpm -Uvh http://server.example.com/pub/katello-ca-consumer-latest.noarch.rpm
   ```

C.5. REGISTERING A HOST

Prerequisites

- Ensure that an activation key that is associated with the appropriate content view and environment exists for the host. For more information, see Managing Activation Keys in the *Content Management Guide*. By default, an activation key has the **auto-attach** function enabled. The feature is commonly used with hosts used as hypervisors.
- Ensure that the version of the **subscription-manager** utility is 1.10 or higher. The package is available in the standard Red Hat Enterprise Linux repository.
  1. On the Red Hat Enterprise Linux Workstation, connect to a terminal as the root user.
  2. Register the host using Red Hat Subscription Manager:

     ```
     # subscription-manager register --org="My_Organization" --activationkey="MyKey"
     ```



NOTE

You can use the **--environment** option to override the content view and life cycle environment defined by the activation key. For example, to register a host to the content view "MyView" in a "Development" life cycle environment:

```
# subscription-manager register --org="My_Organization" \
--environment=Development/MyView \
--activationkey="MyKey"
```

NOTE

For Red Hat Enterprise Linux 6.3 hosts, the release version defaults to Red Hat Enterprise Linux 6 Server and must point to the 6.3 repository.

1. On Red Hat Satellite, select **Hosts** > **Content Hosts**.
2. Select the name of the host that needs to be changed.
3. In the **Content Host Content** section click the edit icon to the right of **Release Version**.
4. Select "6.3" from the **Release Version** drop-down menu.
5. Click **Save**.

C.6. INSTALLING THE KATELLO AGENT

NOTE

The Katello agent is deprecated and will be removed in a future Satellite version. Migrate your workloads to use the remote execution feature to update clients remotely. For more information, see Installing the Katello Agent in the *Managing Hosts Guide*.

Use the following procedure to install the Katello agent on a host registered to Satellite 6. The **katello-agent** package depends on the gofer package that provides the **goferd service**.

Prerequisites

The Satellite Tools 6.10 repository must be enabled, synchronized to Satellite Server, and made available to your hosts as it provides the required packages. For more information about enabling Satellite Tools 6.10, see Installing the Katello Agent in *Managing Hosts*.

To Install the Katello Agent

1. Install the **katello-agent** RPM package using the following command:

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

2. Ensure goferd is running:

```
# systemctl start goferd
```

C.7. INSTALLING THE PUPPET AGENT

Use this section to install and configure the Puppet agent on a host. When you have correctly installed and configured the Puppet agent, you can navigate to **Hosts** > **All hosts** to list all hosts visible to Red Hat Satellite Server.

1. Install the Puppet agent RPM package using the following command:

```
# yum install puppet
```

2. Configure the puppet agent to start at boot:
   On Red Hat Enterprise Linux 6:

```
# chkconfig puppet on
```

   On Red Hat Enterprise Linux 7:

```
# systemctl enable puppet
```

C.8. COMPLETING THE RED HAT ENTERPRISE LINUX 7 IMAGE

1. Update the system:

```
# {package-update}
```

2. Install the **cloud-init** packages:

```
# yum install cloud-utils-growpart cloud-init
```

3. Open the **/etc/cloud/cloud.cfg** configuration file:

```
# vi /etc/cloud/cloud.cfg
```

4. Under the heading **cloud_init_modules**, add:

```
- resolv-conf
```

   The **resolv-conf** option automatically configures the **resolv.conf** when an instance boots for the first time. This file contains information related to the instance such as **nameservers**, **domain** and other options.

5. Open the **/etc/sysconfig/network** file:

```
# vi /etc/sysconfig/network
```

6. Add the following line to avoid problems accessing the EC2 metadata service:

```
NOZEROCONF=yes
```

7. Un-register the virtual machine so that the resulting image does not contain the same subscription details for every instance cloned based on it:

```
# subscription-manager repos --disable=*
# subscription-manager unregister
```

8. Power off the instance:

```
# poweroff
```

9. On your Red Hat Enterprise Linux Workstation, connect to the terminal as the root user and navigate to the **/var/lib/libvirt/images/** directory:

```
[root@host]# cd /var/lib/libvirt/images/
```

10. Reset and clean the image using the **virt-sysprep** command so it can be used to create instances without issues:

```
[root@host]# virt-sysprep -d rhel7
```

11. Reduce image size using the **virt-sparsify** command. This command converts any free space within the disk image back to free space within the host:

```
[root@host]# virt-sparsify --compress rhel7.qcow2 rhel7-cloud.qcow2
```

This creates a new **rhel7-cloud.qcow2** file in the location where you enter the command.

C.9. COMPLETING THE RED HAT ENTERPRISE LINUX 6 IMAGE

1. Update the system:

```
# {package-update}
```

2. Install the **cloud-init** packages:

```
# yum install cloud-utils-growpart cloud-init
```

3. Edit the **/etc/cloud/cloud.cfg** configuration file and under **cloud_init_modules** add:

```
- resolv-conf
```

The **resolv-conf** option automatically configures the **resolv.conf** configuration file when an instance boots for the first time. This file contains information related to the instance such as **nameservers**, **domain**, and other options.

4. To prevent network issues, create the **/etc/udev/rules.d/75-persistent-net-generator.rules** file as follows:

```
# echo "#" > /etc/udev/rules.d/75-persistent-net-generator.rules
```

This prevents **/etc/udev/rules.d/70-persistent-net.rules** file from being created. If **/etc/udev/rules.d/70-persistent-net.rules** is created, networking might not function properly when booting from snapshots (the network interface is created as "eth1" rather than "eth0" and IP address is not assigned).

5. Add the following line to **/etc/sysconfig/network** to avoid problems accessing the EC2 metadata service:

```
NOZEROCONF=yes
```

6. Un-register the virtual machine so that the resulting image does not contain the same subscription details for every instance cloned based on it:

```
# subscription-manager repos --disable=*
# subscription-manager unregister
# yum clean all
```

7. Power off the instance:

```
# poweroff
```

8. On your Red Hat Enterprise Linux Workstation, log in as root and reset and clean the image using the **virt-sysprep** command so it can be used to create instances without issues:

```
[root@host]# virt-sysprep -d rhel6
```

9. Reduce image size using the **virt-sparsify** command. This command converts any free space within the disk image back to free space within the host:

```
[root@host]# virt-sparsify --compress rhel6.qcow2 rhel6-cloud.qcow2
```

This creates a new **rhel6-cloud.qcow2** file in the location where you enter the command.

NOTE

You must manually resize the partitions of instances based on the image in accordance with the disk space in the flavor that is applied to the instance.

C.10. NEXT STEPS

- Repeat the procedures for every image that you want to provision with Satellite.
- Move the image to the location where you want to store for future use.