Red Hat Satellite 6.12

管理内容

管理来自红帽和自定义源内容的指南

Last Updated: 2024-02-15

Red Hat Satellite 6.12 管理内容

管理来自红帽和自定义源内容的指南

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

法律通告

摘要

使用本指南来理解和管理 Satellite 6 中的内容。这类内容示例包括 RPM 文件和 ISO 镜像。红帽卫星 6 使用跨应用程序生命周期提升的一组内容视图来管理此内容。本指南演示了如何创建适合您组织和内容视图的应用程序生命周期，这些视图能够在生命周期环境中发挥重要作用。这些内容视图最终形成了在红帽卫星 6 环境中置备和更新主机的基础。

目录

向红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。

您可以通过在 Bugzilla 中记录一个 ticket 来提交反馈：

1. 导航到 Bugzilla 网站。
2. 在 **Component** 字段中，使用 **Documentation**。
3. 在 **Description** 字段中，输入您要改进的建议。包括文档相关部分的链接。
4. 点 **Submit Bug**。

第 1 章 内容管理简介

在卫星环境中，*内容* 定义为系统上安装的软件。这包括但不仅限于基础操作系统、中间件服务和最终用户应用程序。使用 Red Hat Satellite，您可以在软件生命周期的每个阶段为 Red Hat Enterprise Linux 系统管理各种内容。

Red Hat Satellite 管理以下内容：

订阅管理

这让组织提供了一种管理其红帽订阅信息的方法。

内容管理

这为组织提供了一种方式来存储红帽内容，并以各种方式组织它。

第 2 章 内容类型概述

使用 Red Hat Satellite，您可以管理以下内容类型：

RPM 软件包

从与您的红帽订阅相关的仓库导入 RPM 软件包。卫星服务器从红帽内容交付网络下载 RPM 文件并在本地存储它们。您可以在内容视图中使用这些软件仓库及其 RPM 文件。

Kickstart 树

为创建系统导入 kickstart 树。新的系统通过网络访问这些 kickstart 树，将其用作安装的基础内容。Red Hat Satellite 还包含一些预定义的 kickstart 模板，并能够创建自己的模板，这些模板可用于配置系统并自定义安装。

您还可以在 Satellite 中管理其他类型的自定义内容。例如：

ISO 和 KVM 镜像

下载并管理用于安装和配置的介质。例如，卫星下载、存储和管理特定红帽企业 Linux 和非红帽操作系统的 ISO 镜像和客户机镜像。

自定义文件类型

您可以针对需要的任何文件管理自定义内容，如 SSL 证书和 OVAL 文件。

第 3 章 管理红帽订阅

Red Hat Satellite 可以从 Red Hat Content Delivery Network(CDN)导入内容。Satellite 需要红帽订阅清单，以便从相应的存储库查找、访问和下载内容。您必须有一个红帽订阅清单，其中包含 Satellite 服务器上每个机构的订阅分配。所有订阅信息可在您的红帽客户门户网站帐户中找到。

在完成本章中的任务前，您必须在客户门户网站中创建红帽订阅清单。

请注意，基于权利的订阅模型已弃用，并将在以后的发行版本中删除。红帽建议您改用简单内容访问的基于访问的订阅服务。

要在客户门户网站中创建、管理和导出红帽订阅清单，请参阅在 *Subscription Central* 中为连接的 Satellite 服务器创建和管理清单。

使用本章导入红帽订阅清单，并在 Satellite Web UI 中管理清单。

订阅分配和机构

如果您有多个订阅分配，您可以管理多个机构。Satellite 需要为卫星服务器中配置的每个组织都有一个分配。这样做的好处是，每个机构都有单独的订阅，以便您通过自己的红帽帐户支持多个组织。

未来D 的订阅

您可以在订阅分配中使用未来日期的订阅。当您在现有订阅到期日前向内容主机添加未来订阅时，您可以对存储库具有不间断访问权限。

在当前订阅过期前，手动将未来日期的订阅附加到您的内容主机。不要依赖自动附加方法，因为此方法是为不同的目的而设计的，且可能无法正常工作。更多信息请参阅 第 3.6 节 “将红帽订阅附加到内容主机”。

当将来的订阅处于活跃状态时，您必须刷新清单以同步存储库内容。如需更多信息，请参阅 第 3.5 节 “更新和刷新红帽订阅清单”。

其他资源

- *在断开连接的环境中安装 Satellite 服务器*中的配置 Satellite 服务器来使用来自一个自定义 CDN 的自定义内容

3.1. 将 RED HAT SUBSCRIPTION MANIFEST导入到 SATELLITE SERVER

使用以下步骤将红帽订阅清单导入到 Satellite 服务器中。

先决条件

- 您必须有一个 从红帽客户门户网站导出的红帽订阅 清单文件。如需更多信息，请参阅 *使用红帽订阅管理* 中的*创建和管理* 清单。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Content** > **Subscriptions**，点 **Manage Manifest**。
3. 在 Manage Manifest 窗口中，单击 **Browse**。
4. 导航到包含红帽订阅清单文件的位置，然后单击 **Open**。如果 Manage Manifest 窗口没有自动关闭，点 **Close** 返回 Subscriptions 窗口。

CLI 过程

1. 将红帽订阅清单文件从您的客户端复制到 Satellite 服务器：

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

2. 以 **root** 用户身份登录 Satellite 服务器，再导入 Red Hat 订阅清单文件：

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

现在，您可以启用软件仓库并导入红帽内容。如需更多信息，请参阅 *管理内容* 中的 导入内容。

3.2. 查找红帽订阅

当您将红帽订阅清单导入到 Satellite 服务器时，清单中的订阅会在 Subscriptions 窗口中列出。如果您有大量订阅，您可以过滤结果以查找特定的订阅。

前提条件

- 您必须有一个红帽订阅清单文件导入到 Satellite 服务器。更多信息请参阅 第 3.1 节 “将 Red Hat Subscription Manifest导入到 Satellite Server”。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Content** > **Subscriptions**。
3. 在 Subscriptions 窗口中，点击 **Search** 字段查看构建搜索查询的搜索条件列表。
4. 选择搜索条件来显示更多选项。
5. 构建搜索查询后，点搜索图标。

例如，如果您将光标移到 **Search** 字段中并选择 **expires**，然后按空格，则会出现一个另一个列表，包括 **>**, **<**, 或 **=** 字符。如果选择 **&** **gt**; 并按空格条，则会出现另一个自动选项列表。您还可以输入自己的条件。

3.3. 将红帽订阅添加到订阅分配中

使用以下步骤将红帽订阅添加到 Satellite Web UI 中的订阅分配中。

前提条件

- 您必须有一个红帽订阅清单文件导入到 Satellite 服务器。更多信息请参阅 第 3.1 节 “将 Red Hat Subscription Manifest导入到 Satellite Server”。

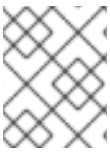
流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Content** > **Subscriptions**。

3. 在 Subscriptions 窗口中，点 **Add Subscriptions**。
4. 在要添加的每个订阅行的行中，输入 bulk to **Allocate to Allocate**列中的数量。
5. 点 **Submit**

3.4. 从订阅分配中删除红帽订阅

使用以下步骤从 Satellite Web UI 中的订阅分配中删除红帽订阅。



注意

不能删除清单。如果您从红帽客户门户网站或 Satellite Web UI 中删除清单，则会删除所有内容主机的所有权利。

前提条件

- 您必须有一个红帽订阅清单文件导入到 Satellite 服务器。更多信息请参阅 第 3.1 节 “将 Red Hat Subscription Manifest导入到 Satellite Server”。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Content** > **Subscriptions**。
3. 在您要删除的每个订阅所在的行中，选择对应的复选框。
4. 单击 **Delete**，然后确认删除。

3.5. 更新和刷新红帽订阅清单

每次更改订阅分配时，您必须刷新清单，以反映这些更改。例如，如果您采取以下任一操作，您必须刷新清单：

- 续订订阅
- 调整订阅量
- 购买额外订阅

您可以在 Satellite Web UI 中直接刷新清单。或者，您可以导入包含更改的更新清单。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Content** > **Subscriptions**。
3. 在 Subscriptions 窗口中，点 **Manage Manifest**。
4. 在 Manage Manifest 窗口中，点 **Refresh**。

3.6. 将红帽订阅附加到内容主机

使用激活密钥是置备过程中将订阅附加到内容主机的主要方法。但是，激活密钥无法更新现有主机。如果需要将新的或附加订阅（如将来的订阅）附加到一个主机，请使用以下步骤。

有关更新多个主机的详情，请参考 第 3.7 节 “在多个主机上更新红帽订阅”。

有关激活码的详情请参考 第 8 章 *管理激活码*。

Satellite 订阅

在 Satellite 中，您必须为您要管理的每个 Red Hat Enterprise Linux 主机维护一个 Red Hat Enterprise Linux Satellite 订阅（以前称为 Red Hat Enterprise Linux 智能管理）。

但是，您不需要将 Satellite 订阅附加到每个内容主机。Satellite 订阅无法自动附加到 Satellite 中的内容主机，因为它们没有与任何产品证书关联。在内容主机中添加 Satellite 订阅不提供任何内容或存储库访问。如果需要，您可以在清单中添加 Satellite 订阅，以满足您自己的记录或跟踪目的。

前提条件

- 您必须有一个红帽订阅清单文件导入到 Satellite 服务器。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Hosts** > **Content Hosts**。
3. 在您要更改的订阅的每个内容主机上的行中，选择对应的复选框。
4. 在 **Select Action** 列表中，选择 **Manage Subscriptions**。
5. （可选）在 **Search** 字段中输入键和值来过滤显示的订阅。
6. 选中您要添加或删除的订阅左侧的复选框，并根据需要点 **Add Selected** 或 **Remove Selected**。
7. 点击 **完成** 保存更改。

CLI 过程

1. 以 root 用户身份连接到卫星服务器，然后列出可用的订阅：

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

2. 为主机附加订阅：

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

3.7. 在多个主机上更新红帽订阅

将这个步骤用于安装后对多个内容主机进行同时的更改。

流程

1. 在卫星 Web UI 中，确保将上下文设置为您要使用的组织。
2. 在 Satellite Web UI 中，导航到 **Hosts** > **Content Hosts**。
3. 在您要更改的订阅的每个内容主机上的行中，选择对应的复选框。
4. 在 **Select Action** 列表中，选择 **Manage Subscriptions**。
5. （可选）在 **Search** 字段中输入键和值来过滤显示的订阅。
6. 选中要添加的订阅左侧的复选框，并根据需要点 **Add Selected** 或 **Remove Selected**。
7. 点击 **完成** 保存更改。

第 4 章 导入内容

本章概述了如何将不同类型的自定义内容导入到 Satellite。如果要向 Satellite 导入 RPM、文件或不同的内容类型，则这一章节中主要有相同的步骤完成。

例如，您可以使用以下章节来获取有关特定类型的自定义内容的信息，但底层步骤相同：

- 第 11 章 *管理 ISO 镜像*
- 第 13 章 *管理自定义文件类型内容*

4.1. SATELLITE 中的产品和存储库

红帽卫星中红帽内容和自定义内容都有相似性：

- 产品及其存储库之间的关系相同，存储库仍然需要同步。
- 自定义产品需要主机的订阅才能访问，类似于红帽产品的订阅。Satellite 为您创建的每个自定义产品创建订阅。

红帽内容已经组织成了产品。例如，红帽企业 Linux 服务器是卫星中 *的产品*。该产品的存储库由不同的版本、架构和附加组件组成。对于红帽存储库，在启用存储库后会自动创建产品。更多信息请参阅 第 4.5 节 "启用红帽软件仓库"。

其它内容可以组织成自定义产品，但您想要。例如，您可以创建一个 EPEL (Extra Packages for Enterprise Linux)产品，并为其添加一个 "EPEL 7 x86_64" 存储库。

有关创建和打包 RPM 的更多信息，请参阅 *Red Hat Enterprise Linux RPM 打包指南*。

4.2. 导入自定义 SSL 证书

在从外部来源同步自定义内容前，可能需要将 SSL 证书导入到自定义产品。这可能包含您要同步的上游软件仓库的客户端 certs 和 key 或 CA 证书。

如果您需要 SSL 证书和密钥下载 RPM，您可以将其添加到 Satellite 中。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Credentials**。在 Content Credentials 窗口中，单击 **Create Content Credential**。
2. 在 **Name** 字段中输入 SSL 证书的名称。
3. 从 **Type** 列表中，选择 **SSL Certificate**。
4. 在 **Content Credentials Content** 字段中，粘贴您的 SSL 证书，或者点击 **Browse** 上传您的 SSL 证书。
5. 点击 **Save**。

CLI 过程

1. 将 SSL 证书复制到 Satellite 服务器中：

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

从在线源将 SSL 证书下载到您的 Satellite 服务器：

```
$ wget -P ~ http://upstream-satellite.example.com/pub/katello-server-ca.crt
```

2. 将 SSL 证书上传到 Satellite:

```
# hammer content-credential create \
--content-type cert \
--name "My_SSL_Certificate" \
--organization "My_Organization" \
--path ~/My_SSL_Certificate
```

4.3. 创建自定义产品

创建自定义产品，以便您可以将存储库添加到自定义产品中。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，点 **Create Product**。
2. 在 **Name** 字段中输入产品的名称。Satellite 根据您为 **Name** 输入的内容自动完成 **Label** 字段。
3. 可选：在 **GPG Key** 列表中，选择产品的 GPG 密钥。
4. 可选：从 **SSL CA Cert** 列表中选择产品的 SSL CA 证书。
5. 可选：从 **SSL Client Cert** 列表中选择产品的 SSL 客户端证书。
6. 可选：从 **SSL Client Key** 列表中，选择产品的 SSL 客户端密钥。
7. 可选：在 **Sync Plan** 列表中，选择现有的同步计划，或者点击 **Create Sync Plan** 并为您的产品要求创建同步计划。
8. 在 **Description** 字段中输入产品的描述。
9. 点击 **Save**。

CLI 过程

运行以下命令来创建产品：

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

4.4. 添加自定义 RPM 软件仓库

使用这个流程在 Satellite 中添加自定义 RPM 软件仓库。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

卫星 Web UI 中的 Products 窗口还提供 **Repo Discovery** 功能，用于从 URL 找到所有存储库，您可以选择要添加到自定义产品中的存储库。例如，您可以使用 **Repo Discovery** 搜索 **http://yum.postgresql.org/9.5/redhat/** 并列出不同 Red Hat Enterprise Linux 版本和架构的所有软件仓库。这有助于帮助用户节省从单一源导入多个存储库的时间。

支持自定义 RPM

红帽不支持直接从第三方站点进行上游 RPM。这些 RPM 用于演示同步过程。有关这些 RPM 的任何问题，请联系第三方开发人员。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，再选择要使用的产品，然后单击 **New Repository**。
2. 在 **Name** 字段中输入存储库的名称。Satellite 根据您为 **Name** 输入的内容自动完成 **Label** 字段。
3. 可选：在 **Description** 字段中输入存储库的描述。
4. 从 **Type** 列表，选择 **yum** 作为存储库类型。
5. 可选：从 **Restrict to** architecture 列表中，选择一个架构。如果您想让存储库可供所有主机使用，请确保选择 **No** limits。
6. 可选：从 **Restrict to OS Version** 列表中，选择 OS 版本。如果您想让存储库可供所有主机使用，无论操作系统版本是什么，请确保选择 **No** limits。
7. 可选：在 **Upstream URL** 字段中输入要用作源的外部存储库的 URL。卫星支持三种协议：**http://**、**https://** 和 **file://**。如果使用 **file://** 存储库，则必须将它放在 **/var/lib/pulp/sync_imports/** 目录中。
   如果没有输入上游 URL，可以手动上传软件包。
8. 可选：检查 **Ignore SRPMs** 复选框，以排除源 RPM 软件包与 Satellite 同步。
9. 如果要验证上游存储库的 SSL 证书是否由可信 CA 签名，请选择 **Verify SSL** 复选框。
10. 可选：在 **Upstream Username** 字段中输入上游存储库的用户名（如果需要）。如果存储库不需要身份验证，请清除此字段。
11. 可选：在 **Upstream Password** 字段中输入上游存储库对应的密码。如果存储库不需要身份验证，请清除此字段。
12. 可选：在 **Upstream Authentication Token** 字段中，提供上游存储库用户的令牌以进行身份验证。如果存储库不需要身份验证，请将此字段留空。
13. 从 **Download Policy** 列表中，选择同步卫星服务器执行的类型。更多信息请参阅 第 4.8 节 “下载策略概述”。
14. 从镜像 **策略** 列表中，选择内容同步卫星服务器执行的类型。更多信息请参阅 第 4.11 节 “镜像策略概述”。
15. 可选：在 **Retain 软件包版本** 字段中，输入每个软件包要保留的版本数量。
16. 可选：在 **HTTP Proxy Policy** 字段中，选择一个 HTTP 代理。
17. 从 **Checksum** 列表中，选择存储库的 checksum 类型。

18. 可选：您可以清除 **未保护** 复选框，以需要一个订阅授权证书来访问这个存储库。默认情况下，存储库通过 HTTP 发布。
19. 可选：在 **GPG Key** 列表中，选择产品的 GPG 密钥。
20. 可选： 在 **SSL CA Cert** 字段中，选择存储库的 SSL CA 证书。
21. 可选： 在 **SSL Client cert** 字段中，选择存储库的 SSL Client Certificate。
22. 可选：在 **SSL Client Key** 字段中，为存储库选择 SSL Client Key。
23. 单击 **Save** 以创建存储库。

CLI 过程

1. 运行以下命令来创建存储库：

```
# hammer repository create \
--arch "My_Architecture" \
--content-type "yum" \
--gpg-key-id My_GPG_Key_ID \
--name "My_Repository" \
--organization "My_Organization" \
--os-version "My_OS_Version" \
--product "My_Product" \
--publish-via-http true \
--url My_Upstream_URL
```

继续 同步存储库。

4.5. 启用红帽软件仓库

如果外部网络访问需要使用 HTTP 代理，请为您的服务器配置默认的 HTTP 代理。如需更多信息，请参阅 向 Satellite 添加默认 HTTP 代理 。

要选择要同步的存储库，您必须首先识别包含存储库的产品，然后根据相关的发行版本版本和基本架构启用该存储库。

Red Hat Enterprise Linux 8 主机

要置备 Red Hat Enterprise Linux 8 主机，您需要 **Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMs)** 和 **Red Hat Enterprise Linux 8 for x86_64 - BaseOS (RPMs)**存储库。

Red Hat Enterprise Linux 7 主机

要配置 Red Hat Enterprise Linux 7 主机，您需要 **Red Hat Enterprise Linux 7 Server (RPMs)**存储库。

将 Red Hat Enterprise Linux 操作系统的发行版本与 **7Server** 仓库或 ***7.X*** 仓库相关联的不同在于：**7Server** 仓库包括所有最新的更新，**Red Hat Enterprise Linux 7*.X*** 仓库会在下一个次版本发行后停止获取更新。请注意，Kickstart 软件仓库只有一个次版本。

流程

1. 在 Satellite Web UI 中，进入 **Content** > **Red Hat Repositories**。

2. 要查找存储库，请输入存储库名称，或者将 **Recommended Repositories** 按钮切换到 on 位置，以查看您所需的存储库列表。

3. 在 Available Repositories 窗格中，单击存储库以展开存储库集。

4. 点基础构架和您想要的发行版本旁的 **Enable** 图标。

CLI 过程

1. 要搜索您的产品，请输入以下命令：

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

2. 列出产品设置的仓库：

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

3. 使用名称或 ID 号启用存储库。包括发行版本，如 **7Server**、基础架构，如 **x86_64**。

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

4.6. 同步软件仓库

您必须同步存储库才能将内容下载到卫星中。您可以使用此流程进行存储库的初始同步，或者根据需要手动同步存储库。

您还可以同步组织中的所有软件仓库。更多信息请参阅 第 4.7 节 “在机构中同步所有存储库”。

创建同步计划，以确保定期进行更新。更多信息请参阅 第 4.19 节 “创建同步计划”。

同步持续时间取决于每个存储库的大小和网络连接速度。下表提供了根据可用的互联网带宽同步内容所需的时长：

	单软件包(10Mb)	次发行版本(750Mb)	主要版本(6Gb)
256 Kbps	5 mins 27 Secs	6 hrs 49 Mins 36 Secs	2 天 7 Hrs 55 Mins
512 Kbps	2 mins 43.84 Secs	3 个 hrs 24 Mins 48 Secs	1 天 3 Hrs 57 Mins
T1 (1.5 Mbps)	54.33 Secs	1 HR 7 Mins 54.78 Secs	9 Hrs 16 Mins 20.57 Secs
10 Mbps	8.39 Secs	10 mins 29.15 Secs	1 HR 25 Mins 53.96 Secs

	单软件包(10Mb)	次发行版本(750Mb)	主要版本(6Gb)
100 Mbps	0.84 Secs	1 分钟 2.91 Secs	8 mins 35.4 Secs
1000 Mbps	0.08 Secs	6.29 Secs	51.54 Secs

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，再选择包含您要同步的存储库的 Product。
2. 选择您要同步的存储库，然后单击 **Sync Now**。
3. 可选： 要在 Satellite Web UI 中查看同步的进度，请导航到 **Content** > **Sync Status**，再展开对应的产品或存储库树。

CLI 过程

- 同步整个产品：

```
# hammer product synchronize \
--name "My_Product" \
--organization "My_Organization"
```

- 同步单个存储库：

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

4.7. 在机构中同步所有存储库

使用此流程同步机构中的所有存储库。

流程

1. 使用 SSH 登录您的卫星服务器。
2. 运行以下 Bash 脚本：

```
ORG="My_Organization"

for i in $(hammer --no-headers --csv repository list --organization $ORG --fields Id)
do
  hammer repository synchronize --id ${i} --organization $ORG --async
done
```

4.8. 下载策略概述

红帽卫星提供了用于同步 RPM 内容的多个下载策略。例如，您可能希望仅下载内容元数据，同时推迟为以后下载的实际内容。

Satellite Server 具有以下策略：

即时

卫星服务器在同步期间下载所有元数据和软件包。

on Demand

卫星服务器仅在同步期间下载元数据。只有胶囊或直接连接的客户端请求它们时，卫星服务器才会获取并存储文件系统中的软件包。如果将胶囊上的对应存储库设置为 **Immediate**，则此设置无效，因为卫星服务器强制下载所有软件包。

On Demand 策略充当 *Lazy Synchronization* 功能，因为它们节省时间同步内容。lazy 同步功能必须只用于 **yum** 软件仓库。您可以将软件包添加到 Content Views 中，并尽可能提升到生命周期环境。

Capsule Server 具有以下策略：

即时

胶囊服务器在同步期间下载所有元数据和软件包。如果卫星服务器上的对应存储库设置为 **On Demand**，则不要使用此设置，因为卫星服务器强制下载所有软件包。

on Demand

胶囊服务器仅在同步期间下载元数据。胶囊服务器仅在直接连接的客户端请求时，仅在文件系统中获取和存储软件包。当您使用 **On Demand** 下载策略时，如果卫星服务器上不可用，则会从卫星服务器下载内容。

继承

胶囊服务器从卫星服务器上的对应存储库继承存储库的下载策略。

流的下载策略

胶囊的流下载策略允许胶囊避免缓存任何内容。从 Capsule 请求内容时，它将充当代理，并直接从 Satellite 请求内容。

4.9. 更改默认下载策略

您可以设置 Satellite 应用到您在所有机构中创建的仓库的默认下载策略。

根据它是红帽还是非红帽自定义软件仓库，Satellite 使用单独的设置。更改默认值不会更改现有设置。

流程

1. 在 Satellite Web UI 中，导航到 **Administer** > **Settings**。
2. 点 **Content** 选项卡。
3. 根据您的要求更改默认下载策略：
   - 要更改红帽存储库的默认下载策略，请更改 **Default Red Hat Repository 下载策略**设置。
   - 要更改自定义存储库的默认下载策略，请更改 **Default Custom Repository download 策略设置的值**。

CLI 过程

- 要将红帽软件仓库的默认下载策略改为 **即时** 或 **on_demand**，请输入以下命令：
  -

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

- 要将非红帽自定义软件仓库的默认下载策略改为 **immediate** 或 **on_demand** 之一，请输入以下命令：

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

4.10. 更改存储库的下载策略

您可以为存储库设置下载策略。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 选择所需的产品名称。
3. 在 **Repositories** 选项卡上，单击所需的存储库名称，找到 **Download Policy** 字段，然后单击 edit 图标。
4. 从列表中选择所需的下载策略，然后单击 **Save**。

CLI 过程

1. 列出机构的存储库：

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

2. 更改存储库的下载策略，使其 **立即** 或 **on_demand** ：

```
# hammer repository update \
--download-policy immediate \
--name "My_Repository" \
--organization-label My_Organization_Label \
--product "My_Product"
```

4.11. 镜像策略概述

镜像使本地存储库与上游存储库完全同步。如果自上一次同步以来，从上游存储库中删除任何内容和下一次同步，也将从本地存储库中删除。

您可以使用镜像策略对存储库同步时对 repodata 和内容进行镜像的镜像控制。例如，如果无法镜像存储库的 repodata，您可以将镜像策略设置为仅镜像这个存储库的内容。

Satellite 服务器具有以下镜像策略：

additive

内容或 repodata 都被镜像(mirror)。因此，只有自最后同步后添加的新内容添加到本地存储库中，不删除任何内容。

仅内容

仅镜像内容而不是 repodata。有些软件仓库不支持元数据镜像，在这种情况下，您可以将镜像策略设置为内容，仅镜像内容。

完整镜像(mirror)

镜像内容和 repodata。这是最快的方法。此镜像策略仅适用于 Yum 内容。

4.12. 更改存储库的镜像策略

您可以为存储库设置镜像策略。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 选择产品名称。
3. 在 **Repositories** 选项卡上，单击存储库名称，找到 **Images Policy** 字段，然后单击编辑图标。
4. 从列表中，选择镜像策略并点 **Save**。

CLI 过程

1. 列出机构的存储库：

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

2. 将一个仓库的镜像（mirroring）策略改为 **additive**, **mirror_complete**, 或 **mirror_content_only**：

```
# hammer repository update \
--id 1 \
--mirroring-policy mirror_complete
```

4.13. 将内容上传到自定义 RPM 存储库

您可以将单独的 RPM 和源 RPM 上传到自定义 RPM 存储库。您可以使用 Satellite Web UI 或 Hammer CLI 上传 RPM。您必须使用 Hammer CLI 上传源 RPM。

流程

1. 在 Satellite Web UI 中，点 **Content** > **Products**。
2. 点自定义产品的名称。
3. 在 **Repositories** 选项卡中，单击自定义 RPM 存储库的名称。
4. 在 **Upload Package** 下，点 **Browse...** 并选择您要上传的 RPM。

5. 点 **Upload**。

要查看此仓库中的所有 RPM，请单击 **Content Counts** 下的 **Packages** 旁边的数字。

CLI 过程

- 输入以下命令上传 RPM:

```
# hammer repository upload-content \
--id Repository_ID \
--path /path/to/example-package.rpm
```

- 输入以下命令上传源 RPM：

```
# hammer repository upload-content \
--content-type srpm \
--id Repository_ID \
--path /path/to/example-package.src.rpm
```

上传完成后，您可以使用命令 **hammer srpm list** 和 **hammer srpm info --id *srpm_ID*** 来查看源 RPM 的信息。

4.14. 在自定义端口上将 SELINUX 配置为 PERMIT CONTENT SYNCHRONIZATION

SELinux 允许在特定端口上访问 Satellite 进行内容同步。默认情况下，允许连接到在以下端口上运行的 Web 服务器： 80、81、443、488、8008、8009、8443 和 9000。

流程

1. 在 Satellite 中，要验证 SELinux 对内容同步允许的端口，请输入以下命令：

```
# semanage port -l | grep ^http_port_t
http_port_t     tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
```

2. 要将 SELinux 配置为允许内容同步端口，如 10011，请输入如下命令：

```
# semanage port -a -t http_port_t -p tcp 10011
```

4.15. 恢复已破坏的存储库

如果存储库崩溃，您可以使用高级同步来恢复它，它有三个选项：

优化同步

同步绕过与上游 RPM 没有检测到的 RPM 的软件仓库。

完成同步

无论所检测到的更改如何同步所有 RPM。如果无法将特定 RPM 下载到本地存储库，则使用此选项，即使它们存在于上游存储库中。

验证内容检查

同步所有 RPM，然后在本地验证所有 RPM 的 checksum。如果 RPM 的 checksum 与上游不同，它会重新下载 RPM。这个选项只与 **yum** 软件仓库相关。如果您有以下错误之一，则使用这个选项：

- 特定的 RPM 在与 **yum** 同步时会导致 **404** 错误。
- **软件包与预期的下载错误不匹配**，这意味着特定的 RPM 被破坏。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 选择包含已损坏存储库的产品。
3. 选择您要同步的存储库的名称。
4. 要执行优化的同步或完整同步，请从 **Select Action** 菜单中选择 **Advanced Sync**。
5. 选择所需选项并点 **Sync**。
6. 可选： 要验证 checksum，从 **Select Action** 菜单中点 **Verify Content Checksum**。

CLI 过程

1. 获取存储库 ID 列表：

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

2. 使用所需选项同步损坏的存储库：
   - 进行优化的同步：

```
# hammer repository synchronize \
--id My_ID
```

   - 对于完整的同步：

```
# hammer repository synchronize \
--id My_ID \
--skip-metadata-check true
```

   - 对于验证内容同步：

```
# hammer repository synchronize \
--id My_ID \
--validate-contents true
```

4.16. 添加 HTTP 代理

使用这个步骤将 HTTP 代理添加到 Satellite。然后，您可以指定哪些 HTTP 代理用于产品、存储库和支持的计算资源。

先决条件

HTTP 代理必须允许访问以下主机：

主机名	端口	协议
subscription.rhsm.redhat.com	443	HTTPS
cdn.redhat.com	443	HTTPS
*.akamaiedge.net	443	HTTPS
cert.console.redhat.com （如果使用 Red Hat Insights）	443	HTTPS
api.access.redhat.com（如果使用 Red Hat Insights）	443	HTTPS
cert-api.access.redhat.com（如果使用 Red Hat Insights）	443	HTTPS

如果 Satellite 服务器使用代理与 subscription.rhsm.redhat.com 和 cdn.redhat.com 通信，则代理不得对这些通信执行 SSL 检查。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Infrastructure** > **HTTP Proxies**，然后选择 **New HTTP Proxy**。
2. 在 **Name** 字段中输入 HTTP 代理的名称。
3. 在 **URL** 字段中输入 HTTP 代理的 URL，包括端口号。
4. 如果您的 HTTP 代理需要身份验证，请输入 **Username** 和 **Password**。
5. 可选：在 **Test URL** 字段中输入 HTTP 代理 URL，然后单击 **Test Connection** 以确保您可以从 Satellite 连接到 HTTP 代理。
6. 点 **Locations** 标签页并添加一个位置。
7. 点 **Organization** 标签页并添加一个机构。
8. 点 **Submit**。

CLI 过程

- 在 Satellite 服务器中，输入以下命令添加 HTTP 代理：

```
# hammer http-proxy create \
--name proxy-name \
--url proxy-URL:port-number
```

如果您的 HTTP 代理需要身份验证，请添加 **--username *name*** 和 **-- *password*密码** 选项。

如需更多信息，请参阅如何在红帽客户门户网站中的防火墙或代理访问 Red Hat Subscription Manager(RHSM)。

4.17. 更改一个产品的 HTTP 代理策略

要精细控制网络流量，您可以为各个产品设置 HTTP 代理策略。产品的 HTTP 代理策略适用于产品中的所有存储库，除非您为各个存储库设置了不同的策略。

要为独立软件仓库设置 HTTP 代理策略，请参阅 第 4.18 节 “更改一个存储库的 HTTP 代理策略”。

流程

1. 在 Satellite web UI 中，导航到 **Content** > **Products**，再选择您要更改的每个产品旁边的复选框。
2. 在 **Select Action** 列表中，选择 **Manage HTTP Proxy**。
3. 从列表中选择一个 **HTTP Proxy Policy** ：
   - **全局默认值** ：使用全局默认代理设置。
   - **没有 HTTP 代理服务器** ：即使配置了全局默认代理，也不使用 HTTP 代理。
   - **使用特定的 HTTP 代理** ：从列表中选择 **HTTP Proxy**。您必须将 HTTP 代理添加到 Satellite，然后才能从这个列表中选择代理。更多信息请参阅 第 4.16 节 “添加 HTTP 代理”。
4. 点 **Update**。

4.18. 更改一个存储库的 HTTP 代理策略

要精细控制网络流量，您可以为每个存储库设置 HTTP 代理策略。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

要为产品中的所有软件仓库设置相同的 HTTP 代理策略，请参阅 第 4.17 节 “更改一个产品的 HTTP 代理策略”。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，并点击包含存储库的产品的名称。
2. 在 **Repositories** 选项卡中，单击存储库的名称。
3. 找到 **HTTP Proxy** 字段并点编辑图标。
4. 从列表中选择一个 **HTTP Proxy Policy** ：
   - **全局默认值** ：使用全局默认代理设置。
   - **没有 HTTP 代理服务器** ：即使配置了全局默认代理，也不使用 HTTP 代理。
   - **使用特定的 HTTP 代理** ：从列表中选择 **HTTP Proxy**。您必须将 HTTP 代理添加到 Satellite，然后才能从这个列表中选择代理。更多信息请参阅 第 4.16 节 “添加 HTTP 代理”。
5. 点击 **Save**。

CLI 过程

- 在 Satellite 服务器中，输入以下命令指定您要使用的 HTTP 代理策略：

```
# hammer repository update \
--http-proxy-policy HTTP_Proxy_Policy \
--id Repository_ID
```

为 **--http-proxy-policy** 指定以下选项之一：

- **无**: 即使配置了全局默认代理，也不使用 HTTP 代理。
- **global_default_http_proxy** ： 使用全局默认代理设置。
- **use_selected_http_proxy**: 使用 **--http-proxy *My_HTTP_Proxy_Name*** 或 **--http-proxy-id *My_HTTP_Proxy_ID*指定 HTTP** 代理。要为 Satellite 添加新的 HTTP 代理，请参阅 第 4.16 节 “添加 HTTP 代理”。

4.19. 创建同步计划

同步计划检查并更新计划的日期和时间中的内容。在 Satellite 中，您可以创建迁移计划，并将产品分配给计划。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Sync Plans**，再点击 **New Sync Plan**。
2. 在 **Name** 字段中输入计划的名称。
3. 可选：在 **Description** 字段中输入计划的描述。
4. 在 **Interval** 列表中，选择要运行计划的时间间隔。
5. 从 **Start Date** 和 **Start Time** 列表中，选择何时开始运行同步计划。
6. 点击 **Save**。

CLI 过程

1. 运行以下命令来创建同步计划：

```
# hammer sync-plan create \
--description "My_Description" \
--enabled true \
--interval daily \
--name "My_Products" \
--organization "My_Organization" \
--sync-date "2023-01-01 01:00:00"
```

2. 查看机构的可用同步计划，以验证是否创建了同步计划：

```
# hammer sync-plan list --organization "My_Organization"
```

4.20. 将 SYNC PLAN 分配给产品

同步计划检查并更新计划的日期和时间中的内容。在 Satellite 中，您可以为产品分配一个同步计划，以定期更新内容。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content > Products**。
2. 选择一个产品。
3. 在 **Details** 标签页中，从下拉菜单中选择 **Sync Plan**。

CLI 过程

1. 为产品分配一个同步计划：

```
# hammer product set-sync-plan \
--name "My_Product_Name" \
--organization "My_Organization" \
--sync-plan "My_Sync_Plan_Name"
```

4.21. 将一个同步计划分配给多个产品

使用这个步骤为机构中至少同步并至少包含一个存储库的产品分配一个同步计划。

流程

1. 运行以下 Bash 脚本：

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

hammer sync-plan create --name $SYNC_PLAN --interval daily --sync-date "2023-04-5
03:00:00" --enabled true --organization $ORG
for i in $(hammer --no-headers --csv --csv-separator="|" product list --organization $ORG --
per-page 999 | grep -vi not_synced | awk -F'|' '$5 != "0" { print $1}')
do
  hammer product set-sync-plan --sync-plan $SYNC_PLAN --organization $ORG --id $i
done
```

2. 执行脚本后，查看分配给同步计划的产品：

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

4.22. 限制同步 CONCURRENCY

默认情况下，每个存储库同步作业一次可最多获取 10 个文件。这可针对每个软件仓库调整。

增加限制可能会提高性能，但可能导致上游服务器过载或开始拒绝请求。如果因为上游服务器拒绝请求而看到存储库同步失败，您可能需要尝试降低限制。

CLI 过程

```
# hammer repository update \
--download-concurrency 5 \
--id Repository_ID \
--organization "My_Organization"
```

4.23. 导入自定义 GPG 密钥

当客户端使用签名的自定义内容时，请确保客户端被配置为使用适当的 GPG 密钥验证软件包安装。这有助于确保只安装来自授权源的软件包。

红帽内容已配置了适当的 GPG 密钥，因此不支持红帽存储库的 GPG 密钥管理。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

先决条件

请确定您有一个 GPG 密钥的副本，用于签署要在 Satellite 中使用和管理的 RPM 内容。大多数 RPM 分发供应商在其网站上提供其 GPG 密钥。您还可以从 RPM 手动提取它：

1. 将版本特定仓库软件包的副本下载到您的客户端系统中：

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

2. 解压 RPM 文件而不安装该文件：

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

GPG 密钥相对于 **etc/pki/rpm-gpg/RPM-GPG-KEY-*EXAMPLE-95*** 的提取。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Credentials**，然后在窗口右上角点击 **Create Content Credential**。
2. 输入存储库的名称，然后从 **Type** 列表中选择 **GPG Key**。
3. 将 GPG 密钥粘贴到 **Content Credential Contents** 字段中，或者点击 **Browse** 并选择您要导入的 GPG 密钥文件。
   如果您的自定义仓库包含多个 GPG 密钥签名的内容，则必须在 **Content Credential Contents** 字段中使用每个密钥之间的新行输入所有所需的 GPG 密钥，例如：

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

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

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

mQINBFw467UBEACmREzDeK/kuScCmfJfHJa0Wgh/2fbJLLt3KSvsgDhORIptf+PP
OTFDlKuLkJx99ZYG5xMnBG47C7ByoMec1j94YeXczuBbynOyyPlvduma/zf8oB9e
```

```
Wl5GnzcLGAnUSRamfqGUWcyMMinHHIKlc1X1P4l=
=WPpI
-----END PGP PUBLIC KEY BLOCK-----
```

4. 点击 **Save**。

CLI 过程

1. 将 GPG 密钥复制到 Satellite 服务器中：

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

2. 将 GPG 密钥上传到 Satellite：

```
# hammer content-credentials create \
--content-type gpg_key \
--name "My_GPG_Key" \
--organization "My_Organization" \
--path ~/RPM-GPG-KEY-EXAMPLE-95
```

4.24. 在 SATELLITE 中将自定义软件仓库限制为 RHEL 9

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 选择您要限制的存储库集合的产品。
3. 在 **Repositories** 选项卡中，点您要限制的存储库。
4. 从 **Restrict to OS** 列表中，选择操作系统以为其对应的 Red Hat Enterprise Linux 软件仓库选择包括 Red Hat Enterprise Linux 操作系统。
5. 点击 **Save** 提交您的更改。

第 5 章 管理应用程序生命周期

本章概述了 Satellite 中的应用程序生命周期，以及如何为 Satellite 和 Capsule 创建和删除应用程序生命周期。

5.1. 应用程序生命周期简介

应用程序生命周期 是卫星内容管理功能的一个概念。应用程序生命周期定义了特定系统及其软件如何进入特定的阶段。例如，应用程序生命周期可能很简单；您可能只有一个开发阶段和生产阶段。在这种情况下，应用程序生命周期可能类似如下：

- 开发
- Production

但是，更复杂的应用程序生命周期可能具有进一步的阶段，如用于测试的阶段或 beta 版本。这会在应用程序生命周期中添加额外的阶段：

- 开发
- 测试
- 测试版本
- Production

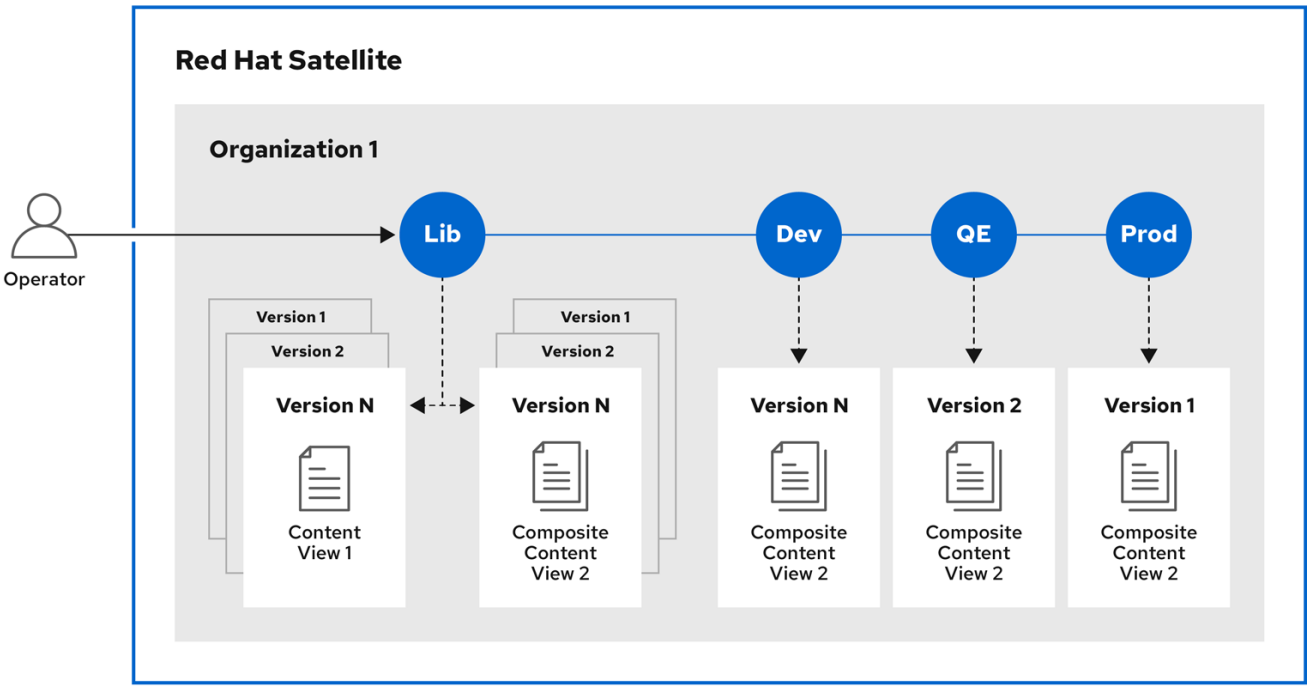
卫星提供了自定义每个应用生命周期阶段的方法，使其符合您的规格。

应用程序生命周期中的每个阶段都在 Satellite 中称为 *环境*。每个环境都使用特定的内容集合。卫星将这些内容集合定义为内容视图。每个内容视图都充当一个过滤器，您可以在什么过滤器中定义要包含在特定环境中的软件包。这为您提供了定义指定到各个环境的特定内容集合的方法。

例如，电子邮件服务器可能只需要一个简单应用程序生命周期，其中有一个生产级服务器用于实际用途，而测试服务器用于尝试最新的邮件服务器软件包。当测试服务器通过初始阶段时，您可以将生产级服务器设置为使用新软件包。

另一个例子是软件产品的开发生命周期。要在开发环境中开发新的软件，请在质量保证环境中进行测试，作为测试版本，作为生产级应用程序发布该软件。

图 5.1. Satellite 应用程序生命周期



5.2. 应用程序生命周期中的内容提升

在应用程序生命周期链中，当内容从一个环境移到下一个环境时，这称为 *提升*。

跨 Satellite 生命周期环境的内容提升示例

每个环境包含注册到 Red Hat Satellite 的一组系统。这些系统只能访问与其环境相关的存储库。当您将软件包从一个环境提升到下一个环境时，目标环境的存储库会接收新的软件包版本。因此，目标环境中的每个系统都可以更新到新软件包版本。

开发	测试	Production
example_software-1.1-0.noarch.rpm	*example_software*-1.0-0.noarch.rpm	*example_software*-1.0-0.noarch.rpm

在补丁上执行开发后，您将把软件包提升到测试环境，以便质量工程团队可以查看补丁。然后，应用程序生命周期会在每个环境中包含以下软件包版本：

开发	测试	Production
example_software-1.1-0.noarch.rpm	***example_software*-1.1-0.noarch.rpm**	*example_software*-1.0-0.noarch.rpm

虽然质量工程团队审查了补丁，但开发团队在 *example_software* 2.0 上开始工作。这会生成以下应用程序生命周期：

开发	测试	Production
example_software-2.0-0.noarch.rpm	example_software-1.1-0.noarch.rpm	example_software-1.0-0.noarch.rpm

质量工程团队完成对补丁的审查。现在，*example_software* 1.1 已准备好发布。将 1.1 提升到 *Production* 环境：

开发	测试	Production
example_software-2.0-0.noarch.rpm	example_software-1.1-0.noarch.rpm	**example_software-1.1-0.noarch.rpm**

开发团队在 *example_software* 2.0 上完成工作，并将其提升到测试环境：

开发	测试	Production
example_software-2.0-0.noarch.rpm	**example_software-2.0-0.noarch.rpm**	example_software-1.1-0.noarch.rpm

最后，质量工程团队会检查软件包。成功检查后，将软件包提升到 *Production* 环境：

开发	测试	Production
example_software-2.0-0.noarch.rpm	example_software-2.0-0.noarch.rpm	**example_software-2.0-0.noarch.rpm**

更多信息请参阅 第 6.3 节 “提升内容视图”。

5.3. 创建生命周期环境路径

要创建用于开发和发布软件的应用程序生命周期，请使用 *Library* 环境作为初始环境，以创建环境路径。然后可选择向环境路径添加其他环境。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Lifecycle Environments**。
2. 点 **New Environment Path** 启动新的应用程序生命周期。
3. 在 **Name** 字段中输入您的环境的名称。
4. 在 **Description** 字段中输入您的环境的描述。
5. 点击 **Save**。
6. 可选：要将环境添加到环境路径中，点 **Add New Environment**、完成 **Name** 和 **Description** 字段，并从 **Prior Environment** 列表中选择前面的环境。

CLI 过程

1. 要创建路径，请输入 **hammer lifecycle-environment create** 命令并使用 **--prior** 选项指定 Library 环境：

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

2. 可选：要在路径中添加环境，请输入 **hammer lifecycle-environment create** 命令并使用 **--prior** 选项指定父环境：

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

3. 要查看生命周期环境的链，请输入以下命令：

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

5.4. 从 SATELLITE 服务器中删除生命周期环境

使用此流程删除生命周期环境。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Life Cycle Environments**。
2. 点您要删除的生命周期环境名称，然后点 **Remove Environment**。
3. 点 **Remove** 删除环境。

CLI 过程

1. 列出您的机构的生命周期环境，并记下您要删除的生命周期环境的名称：

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

2. 使用 **hammer Life-environment delete** 命令删除环境：

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

5.5. 从胶囊服务器中删除生命周期环境

当生命周期环境不再与主机系统或环境不正确添加到胶囊服务器时，您可以从 Capsule Server 中删除生命周期环境。

您可以使用卫星 Web UI 和 Hammer CLI 从 Capsule 删除生命周期环境。

流程

1. 在 Satellite Web UI 中，导航到 **Infrastructure** > **Capsules**，然后选择您要从中删除生命周期的胶囊。
2. 单击 **Edit**，再单击 **Life Cycle Environments** 选项卡。
3. 在右侧菜单中，选择您要从 Capsule 中删除的生命周期环境，然后单击 **Submit**。
4. 若要同步胶囊的内容，可单击 **Overview** 选项卡，然后单击 **Synchronize**。
5. 选择 **Optimized Sync** 或 **Complete Sync**。

CLI 过程

1. 从列表中选择您要的胶囊服务器，并记录其 **id** ：

```
# hammer capsule list
```

2. 要验证胶囊服务器的详情，请输入以下命令：

```
# hammer capsule info \
--id My_Capsule_ID_
```

3. 验证当前附加到胶囊服务器的生命周期环境列表，并记录 **环境 ID** ：

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

4. 从胶囊服务器中删除生命周期环境：

```
# hammer capsule content remove-lifecycle-environment \
--id My_Capsule_ID_
--lifecycle-environment-id My_Lifecycle_Environment_ID
```

对于您要从胶囊服务器中删除的每个生命周期环境，重复此步骤。

5. 将卫星服务器环境的内容同步到胶囊服务器：

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

5.6. 将生命周期环境添加到胶囊服务器

如果您的胶囊服务器启用了内容功能，您必须添加一个环境，以便胶囊可以从卫星服务器同步内容，并向主机系统提供内容。

不要将 *Library* 生命周期环境分配给您的胶囊服务器，因为它在每次 CDN 更新存储库时触发自动胶囊同步。这可能会在胶囊上消耗多个系统资源、卫星和胶囊之间的网络带宽，以及胶囊上的可用磁盘空间。

您可以在卫星服务器或卫星 Web UI 上使用 Hammer CLI。

流程

1. 在 Satellite Web UI 中，导航到 **Infrastructure** > **Capsules**，然后选择您要为其添加生命周期的胶囊。
2. 单击 **Edit**，再单击 **Life Cycle Environments** 选项卡。
3. 在左侧菜单中选择您要添加到胶囊的生命周期环境，然后单击 **Submit**。
4. 要同步胶囊上的内容，请单击 **Overview** 选项卡，再单击 **Synchronize**。
5. 选择 **Optimized Sync** 或 **Complete Sync**。
   有关每种同步类型的定义，请参阅 恢复存储库。

CLI 过程

1. 要显示所有胶囊服务器的列表，请在卫星服务器上输入以下命令：

```
# hammer capsule list
```

   记下您要向其添加生命周期的胶囊 ID。

2. 使用 ID，验证您的胶囊的详细信息：

```
# hammer capsule info \
--id My_capsule_ID
```

3. 要查看您的胶囊服务器可用的生命周期环境，请输入以下命令并记录 ID 和机构名称：

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

4. 将生命周期环境添加到您的胶囊服务器：

```
# hammer capsule content add-lifecycle-environment \
--id My_capsule_ID \
--lifecycle-environment-id My_Lifecycle_Environment_ID
--organization "My_Organization"
```

   对您要添加到胶囊服务器的每个生命周期环境重复。

5. 将卫星中的内容同步到 Capsule。

   - 要将 Satellite 服务器环境中的所有内容同步到胶囊服务器，请输入以下命令：

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

   - 要将 Satellite 服务器中的特定生命周期环境与胶囊服务器同步，请输入以下命令：

```
# hammer capsule content synchronize \
--id My_capsule_ID
--lifecycle-environment-id My_Lifecycle_Environment_ID
```

第 6 章 管理内容视图

Red Hat Satellite 使用内容视图来允许主机访问独立策展的内容子集。要做到这一点，您必须定义要使用的存储库，然后将某些过滤器应用到内容。这些过滤器包括软件包过滤器、软件包组过滤器、勘误表过滤器、模块流过滤器和容器镜像标签过滤器。您可以使用内容视图来定义特定环境使用的软件版本。例如，*生产环境* 可能使用包含较旧软件包版本的内容视图，而 *开发环境* 可能会使用包含较新软件包版本的内容视图。

或者，*默认的组织* 视图是应用程序控制的内容视图，用于与 Satellite 同步的所有内容。如果您想将主机注册到 Satellite 并使用订阅访问内容，而不操作内容视图和生命周期环境，则此类型非常有用。

每个内容视图在每个环境中创建一组存储库，供卫星服务器存储和管理。当您在应用程序生命周期中将内容视图从一个环境提升到下一个环境时，卫星服务器上对应的存储库会更新并发布软件包。

	开发	测试	Production
内容视图和内容	版本 2 - *example_software*-1.1-0.noarch.rpm	版本 1 - *example_software*-1.0-0.noarch.rpm	版本 1 - *example_software*-1.0-0.noarch.rpm

用于 Testing 和 Production 的软件仓库包含 ***example_software*-1.0-0noarch.rpm** 软件包。如果您将 Version 2 的内容视图从 Development 提升到 Testing，则测试会重新生成的存储库，然后包含 ***example_software*-1.1-0.noarch.rpm** 软件包：

	开发	测试	Production
内容视图和内容	版本 2 - *example_software*-1.1-0.noarch.rpm	**版本 2 - *example_software*-1.1-0.noarch.rpm**	版本 1 - *example_software*-1.0-0.noarch.rpm

这样可保证系统被指定到特定的环境，但在该环境使用新版本的内容视图时接收更新。

创建用于过滤和创建快照的内容视图的一般工作流如下：

1. 创建内容视图.
2. 为内容视图添加一个或多个您需要的仓库。
3. 可选：创建一个或多个过滤器来重新定义内容视图的内容。更多信息请参阅 第 6.9 节 “内容过滤器示例”。
4. 可选：解决内容视图的任何软件包依赖项。更多信息请参阅 第 6.7 节 “解决软件包依赖项”。
5. 发布内容视图。
6. 可选：将内容视图提升到另一个环境。更多信息请参阅 第 6.3 节 “提升内容视图”。
7. 将内容主机附加到内容视图。

如果存储库没有与内容视图关联，文件 **/etc/yum.repos.d/redhat.repo** 仍为空，且注册到该系统的系统无法接收更新。

主机只能与单个内容视图关联。要将主机与多个内容视图关联，请创建一个复合内容视图。更多信息请参阅 第 6.5 节 “创建复合内容视图”。

6.1. 创建内容视图

使用此流程创建简单的内容视图。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

先决条件

虽然您可以确定是否要根据内容视图解析内容视图上的任何软件包依赖项，但可能需要更改默认的卫星设置，以为所有内容视图启用或禁用软件包解析。更多信息请参阅 第 6.7 节 “解决软件包依赖项”。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Views**，再点击 **Create content view**。
2. 在 **Name** 字段中输入视图的名称。Satellite 会根据您输入的名称自动完成 **Label** 字段。
3. 在 **Description** 字段中输入视图的描述。
4. 在 **Type** 字段中，选择 **Content view** 或 **Composite 内容视图**。
5. 可选：如果您要在每次发布此内容视图时自动解决依赖项，请选择 Solv **e 依赖项** 复选框。依赖项解决会减慢发布时间，并可能忽略您使用的任何内容视图过滤器。这在解决勘误依赖关系时也会导致错误。
6. 可选： 如果要指定从上游服务器导入此内容视图，请选择 **Import only** 复选框。仅导入的内容视图无法直接发布。
7. 点 **Create 内容视图**。

内容视图步骤

1. 单击 **Create 内容视图** 以创建 Content View。
2. 在 **Repositories** 选项卡中，从您要添加到内容视图的 **Type** 列表中选择存储库，选中您要添加的可用存储库旁边的复选框，然后单击 **添加存储库**。
3. 单击 **Publish new version**，然后在 **Description** 字段中输入有关版本的信息以记录更改。
4. 可选：您可以通过点 **Promote** 到 **Select a lifecycle environment from the available promotion paths to promote new version** 来启用一个提升。
5. 点击 **Next**。
6. 在 **Review** 页面中，您可以查看您要发布的环境。
7. 点 **Finish**。

注意

Remove 和 **Delete** 类似，但 **Delete** 选项会删除整个内容视图以及与那个生命周期环境相关联的版本。**Remove** 选项允许您选择从生命周期环境中删除的版本。

您可以在 Content Views 窗口中查看 Content View。要查看有关内容视图的更多信息，请单击 Content View name。要将一个主机注册到您的内容视图，请参阅*管理主机*中的注册主机。

CLI 过程

1. 获取存储库 ID 列表：

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

2. 创建 Content View 并添加软件仓库：

```
# hammer content-view create \
--description "My_Content_View" \
--name "My_Content_View" \
--organization "My_Organization" \
--repository-ids 1,2
```

对于 **--repository-ids** 选项，您可以在 **hammer 存储库列表** 命令的输出中找到 ID。

3. 发布视图：

```
# hammer content-view publish \
--description "My_Content_View" \
--name "My_Content_View" \
--organization "My_Organization"
```

4. 可选： 要将存储库添加到现有 Content View 中，请输入以下命令：

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

Satellite 服务器会创建新版本的视图，并将它发布到 Library 环境中。

6.2. 查看模块流

在 Satellite 中，您可以在 Content Views 中查看存储库的模块流。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到发布版本的 **Content View** > **Module Streams**，以查看可用于 Content Types 的模块流。
2. 使用 **Search** 字段搜索特定模块。
3. 要查看模块的相关信息，请单击模块及其对应的选项卡，以包含 **详细信息**、**存储库**、**配置文件和 Artifacts**。

CLI 过程

1. 列出所有机构：

```
# hammer organization list
```

2. 查看您的机构的所有模块流：

```
# hammer module-stream list \
--organization-id My_Organization_ID
```

6.3. 提升内容视图

使用此流程在不同生命周期环境中提升内容视图。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

内容视图提升的权限要求

非管理员用户需要两个权限将内容视图提升到环境：

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

promote_or_remove_content_views 权限限制用户可以提升的 Content Views。

promote_or_remove_content_views_to_environment 权限限制用户可以提升内容视图的环境。

通过这些权限，您可以为用户权限将某些内容视图提升到某些环境，但不能分配给其他环境。例如，您可以限制用户，以便允许用户提升测试环境，但不能提升到生产环境。

您必须为用户分配这两个权限，以允许他们提升内容视图。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Views**，再选择您要提升的 Content View。
2. 选择您要提升的版本，单击垂直提示图标，然后单击 **Promote**。
3. 选择您要提升内容视图的环境，然后点 **Promote**。

现在，内容视图的存储库会出现在所有环境中。

CLI 过程

- 为每个生命周期环境使用 Hammer 提升内容视图：

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

现在，所有环境中都提供了数据库内容。

- 或者，您可以使用以下 Bash 脚本在机构中的所有生命周期环境中提升内容视图：

```
ORG="My_Organization"
CVV_ID=My_Content_View_Version_ID

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

验证

- 显示您的 Content View 版本的信息，以验证它是否已提升到所需的生命周期环境：

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

后续步骤

- 要将一个主机注册到您的内容视图，请参阅 *管理主机* 中的注册主机。

6.4. 复合内容视图概述

Composite Content View 组合了几个内容视图中的内容。例如，您可能有单独的内容视图来管理操作系统和一个应用程序。您可以使用 Composite Content View，将两个内容视图的内容合并到新存储库中。原始内容视图的存储库仍然存在，但也会存在用于组合内容的新存储库。

如果要开发支持不同数据库服务器的应用程序。*example_application* 如下所示：

example_software
Application（应用程序）
数据库
操作系统

四个单独的内容视图示例：

- Red Hat Enterprise Linux (Operating System)
- PostgreSQL (Database)
- MariaDB (Database)
- *example_software* （应用程序）

在前面的内容视图中，您可以创建两个 Composite Content Views。

PostgreSQL 数据库的 Composite Content View 示例：

复合内容视图 1 - PostgreSQL 上的 *example_software*
example_software （应用程序）
PostgreSQL (Database)
Red Hat Enterprise Linux (Operating System)

MariaDB 的 Composite Content View 示例：

复合内容视图 2 - MariaDB 上的 *example_software*
example_software （应用程序）
MariaDB (Database)
Red Hat Enterprise Linux (Operating System)

然后，每个内容视图都会被单独管理并单独发布。在创建应用版本时，您将发布新版本的 Composite Content Views。在创建 Composite Content View 时，也可以选择 **Auto Publish** 选项，然后在包含内容视图的内容视图重新发布时，会自动重新发布 Composite Content View。

仓库限制

Docker 存储库不能包含在 Composite Content View 中多次。例如，如果您试图在 Composite Content View 中使用相同的 docker 存储库包含两个内容视图，则 Satellite 服务器会报告错误。

6.5. 创建复合内容视图

使用此流程来创建复合内容视图。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Views**，再点击 **Create content view**。
2. 在 **Create content view** 窗口中，在 **Name** 字段中输入 view 的名称。Red Hat Satellite 会自动从您输入的名称中完成 **Label** 字段。
3. 可选：在 **Description** 字段中输入 view 描述。
4. 在 **Type** 选项卡上，选择 **Composite 内容视图**。
5. 可选：如果您要在内容视图重新发布时，自动发布最新版本的 Composite Content View，请选择 **Auto publish** 复选框。
6. 点 **Create 内容视图**。
7. 在 **内容视图** 选项卡上，选择您要添加到 Composite 内容视图的内容视图，然后单击 **Add 内容视图**。

8. 在 **Add 内容视图** 窗口中，选择每个内容视图的版本。
9. 可选：如果要自动将内容视图更新到最新版本，请选择 **Always update to latest version** 复选框。
10. 点 **Add**，然后点 **Publish new version**。
11. 可选：在 **Description** 字段中输入 Content View 的描述。
12. 在 **Publish** 窗口中，设置 **Promote** 开关，然后选择 生命周期环境。
13. 单击**下一步**，然后单击 **完成**。

CLI 过程

1. 在创建 Composite Content Views 之前，列出您现有内容视图的版本 ID：

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

2. 创建新的 Composite 内容视图。当 **--auto-publish** 选项被设置为 **yes** 时，Composite Content View 会在内容视图包含重新发布时自动重新发布：

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

3. 将内容视图添加到复合内容视图.您可以通过其 ID 或名称来识别命令中的内容视图、内容视图版本和机构。要在 Composite Content View 中添加多个内容视图，请对您要包含的每个内容视图重复此步骤。
   - 如果您为内容视图启用了 **Always update to latest version** 选项：

```
# hammer content-view component add \
--component-content-view-id Content_View_ID \
--composite-content-view "Example_Composite_Content_View" \
--latest \
--organization "My_Organization"
```

   - 如果您为内容视图禁用了 **Always update to latest version** 选项：

```
# hammer content-view component add \
--component-content-view-id Content_View_ID \
--composite-content-view "Example_Composite_Content_View" \
--component-content-view-version-id Content_View_Version_ID \
--organization "My_Organization"
```

4. 发布 Composite Content View：

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

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

5. 在所有环境中提升 Composite Content View:

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

6.6. 内容过滤器概述

内容视图也使用过滤器来包含或限制某些 RPM 内容。如果没有这些过滤器，内容视图包括来自所选存储库的所有内容。

有两种类型的内容过滤器：

表 6.1. 过滤器类型

过滤器类型	描述
include	您从没有内容开始，然后选择要从所选存储库中添加的内容。使用此过滤器合并多个内容项目。
exclude	您从所选仓库开始所有内容，然后选择要删除的内容。当您想使用大多数特定内容存储库但排除某些软件包（如列入黑名单的软件包）时，请使用此过滤器。除您选择的内容外，过滤器会使用存储库中的所有内容。

包含和 Exclude Filter Combinations

如果使用 Include 和 Exclude 过滤器的组合，则首先发布内容视图会触发 include 过滤器，则排除过滤器。在这种情况下，选择要包含哪些内容，然后从所含子集中排除哪些内容。

内容类型

过滤也有五种类型：

表 6.2. 内容类型

内容类型	描述

内容类型	描述
RPM	根据软件包的名称和版本号过滤。**RPM** 选项会过滤非模块化 RPM 软件包和勘误表。源 RPM 不受此过滤器的影响，在内容视图中仍可用。
软件包组	根据软件包组过滤软件包。软件包组的列表基于添加到内容视图中的存储库。
勘误（按 ID）	选择要添加到过滤器中的特定勘误。勘误列表基于添加到内容视图中的存储库。
勘误（按日期和类型）	选择要添加到过滤器的签发或更新日期范围及勘误类型（修复、增强或安全性）。
模块流	选择是否包含或排除特定的模块流。**Module Streams** 选项会过滤模块 RPM 和勘误，但不过滤与所选模块流关联的非修改内容。
容器镜像标签	选择是否包含或排除特定容器镜像标签。

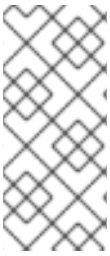
6.7. 解决软件包依赖项

在发布内容视图时，Satellite 可以在内容视图中添加软件包的依赖项到依赖的存储库。要配置此功能，您可以启用 *依赖项解析*。

例如，当您逐步将单个软件包添加到内容视图版本时，依赖项会很有用。您可能需要启用依赖项，以安装该软件包。

然而，在大多数情况下，依赖项无法解决。例如：

- 当逐步添加安全勘误到内容视图时，依赖项解决可能会导致对内容视图发布造成大量延迟，而不会造成主要好处。
- 较新的勘误中的软件包可能会具有与旧内容视图版本中的软件包不兼容的依赖项。使用依赖项解析以递增方式添加勘误可能包括不需要的软件包。作为替代方案，请考虑更新内容视图。



注意

依赖项有助于仅考虑内容视图的存储库中的软件包。它不考虑客户端上安装的软件包。例如，如果内容视图仅包含 AppStream，依赖项需要包括在发布时不包含依赖 BaseOS 内容。

如需更多信息，请参阅管理内容中的 对存储库依赖解析的限制。

依赖项解决可能会导致以下问题：

内容视图发布中的显著延迟

Satellite 会在内容视图中针对依赖项检查每个存储库。因此，发布时间会增加存储库。要缓解这个问题，请使用带有较少存储库的多个内容视图，并将它们合并到复合内容视图中。

忽略依赖软件包的内容视图过滤器

Satellite 优先选择根据过滤器中的规则解析软件包依赖项。
例如，如果您为安全目的创建过滤器，但启用依赖项解析，Satellite 您可以添加可能认为不安全的软件包。

要缓解这个问题，请仔细测试过滤规则以确定所需的依赖项。如果依赖项解决包括不需要的软件包，请手动识别额外的软件包和勘误所需的核心基本依赖项。

> **例 6.1. 将排除过滤器与依赖项解决合并**
>
> 您需要使用 Content View 过滤器重新创建 Red Hat Enterprise Linux 8.3，并包括后续 Red Hat Enterprise Linux 8 次版本中的所选勘误。要做到这一点，您可以在 Red Hat Enterprise Linux 8.3 发行日期后创建过滤器来排除大多数勘误，除了几个您需要的。然后，您可以实现依赖项解决。
>
> 在这种情况下，依赖项解决可能会包括比预期更多的软件包。因此，主机被认为是 Red Hat Enterprise Linux 8.3 机器。
>
> 如果您不需要额外的勘误表和软件包，请不要配置内容视图过滤。反之，在 Satellite Web UI 中的 **Content** > **Red Hat Repositories** 页面中启用并使用 Red Hat Enterprise Linux 8.3 软件仓库。

> **例 6.2. 排除软件包有时会使 DNF 无法进行依赖项解决**
>
> 如果您制作带有一些排除的软件包的 Red Hat Enterprise Linux 8.3 软件仓库，**dnf update** 有时可能会失败。
>
> 不要启用依赖项来解决问题。相反，应调查 **dnf** 中的错误，并调整过滤器以停止排除缺少的依赖项。
>
> 否则，依赖项解决可能会导致存储库从 Red Hat Enterprise Linux 8.3 分离。

6.8. 为内容视图启用依赖解析

使用这个流程启用对内容视图的依赖项。

前提条件

- 依赖项需要仅在有限的上下文中有用。在启用它前，请确定您读和了解 第 6.7 节 "解决软件包依赖项"

流程

1. 在 Satellite Web UI 中，进入到 **Content** > **Content Views**。
2. 从内容视图列表中，选择所需的内容视图。
3. 在 **Details** 标签页中，切换 **Solve 依赖项**。

6.9. 内容过滤器示例

按照下方的步骤使用以下任一示例来构建自定义内容过滤器。

注意

过滤器可能会显著增加发布内容视图的时间。例如，如果内容视图发布任务在短短几分钟内完成，则添加 exclude 或 include 勘误过滤器后可能需要 30 分钟。

示例 1

使用基本 Red Hat Enterprise Linux 软件包创建软件仓库。这个过滤器需要添加到内容视图中的 Red Hat Enterprise Linux 存储库。

filter：

- **包含类型：** Include
- **内容类型：** 软件包组
- **filter：** 只选择 **Base** 软件包组

示例 2

在特定日期之后，创建一个排除所有勘误的存储库，但安全更新除外。如果您要定期执行系统更新，但关键安全更新除外，该更新必须立即应用。这个过滤器需要添加到内容视图中的 Red Hat Enterprise Linux 存储库。

filter：

- **包含类型：** Exclude
- **内容类型：** 勘误（按日期和类型）
- **过滤：** 只选择 **程序 错误修复和功能增强** 勘误类型，并清除 **安全勘误** 类型。将 **Date Type** 设置为 **Updated On**。将 **Start Date** 设置为您要限制勘误的日期。将 **"结束日期"** 留空，以确保过滤任何新的非安全勘误。

示例 3

示例 1 和示例 2，其中您只需要操作系统软件包，并希望排除最新的错误修复和功能增强勘误。这需要两个过滤器附加到同一内容视图。Content View 会首先处理 Include 过滤器，然后是 Exclude 过滤器。

过滤 1：

- **包含类型：** Include
- **内容类型：** 软件包组
- **filter：** 只选择 **Base** 软件包组

过滤 2：

- **包含类型：** Exclude
- **内容类型：** 勘误（按日期和类型）
- **过滤：** 只选择 **程序 错误修复和功能增强** 勘误类型，并清除 **安全勘误** 类型。将 **Date Type** 设置为 **Updated On**。将 **Start Date** 设置为您要限制勘误的日期。将 **"结束日期"** 留空，以确保过滤任何新的非安全勘误。

示例 4

在内容视图中过滤特定的模块流。

过滤 1：

- **包含类型：** Include
- **内容类型：** 模块流
- **filter：** 仅选择您想要用于内容视图的特定模块流，例如 **ant**，然后单击 **Add Module Stream**。

过滤 2：

- **包含类型：** Exclude
- **内容类型：** Package
- **filter：** 添加一个规则，以过滤您要从内容视图中排除的任何非模块化软件包。如果您没有过滤软件包，则 Content View 过滤器包含与模块流 **ant** 关联的所有非单一软件包。添加一个规则来排除所有 * 软件包，或指定您要排除的软件包名称。

有关内容过滤器的工作方式的另一个示例，请参阅以下文章：*"如何将内容过滤器在 Satellite 6 中正常工作"*。

6.10. 为 YUM 内容创建内容过滤器

您可以过滤包含 Yum 内容的 Content Views，以包含或排除特定软件包、软件包组、勘误表或模块流。过滤器基于 名称、版本 和 架构 的组合。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

有关如何构建过滤器的示例，请参阅 第 6.9 节 "内容过滤器示例"。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Views** 并选择 Content View。
2. 在 **Filters** 选项卡中，点 **Create filter**。
3. 输入名称。
4. 从 **Content type** 列表中，选择内容类型。
5. 在 **Inclusion Type** 列表中，选择 **Include 过滤器** 或 **Exclude 过滤器**。
6. 可选：在 **Description** 字段中输入过滤器的描述。
7. 点 **Create filter** 创建您的内容过滤器。
8. 根据您为 **内容类型** 输入的内容，添加规则以创建您想要的过滤器。
9. 如果您希望过滤器应用到 **存储库的子集** 或 **应用到所有存储库**，请选择此项。
10. 单击 **Publish New Version** 以发布过滤的存储库。
11. 可选：在 **Description** 字段中输入更改的描述。
12. 单击 **Create filter**，以发布内容视图的新版本。

您可以在所有环境中提升此内容视图。

CLI 过程

1. 添加过滤器到内容视图。使用 **--inclusion false** 选项将过滤器设置为 Exclude 过滤器：

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

2. 在过滤器中添加规则：

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

3. 发布内容视图：

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

4. 在所有环境中提升视图：

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

第 7 章 同步 SATELLITE 服务器之间的内容

在与多个卫星服务器的卫星设置中，您可以使用卫星服务器间同步(ISS)的内容从一个上游服务器同步到一个或多个下游服务器。

Satellite 有两个可能的 ISS 配置，具体取决于您如何部署基础架构。根据您的用例场景，为您的 Satellite 配置您的 Satellite。如需更多信息，请参阅在断开连接的网络环境中安装 Satellite 服务器 中的如何配置 Inter-Satellite 同步。要更改 pulp 导出路径，请参阅 知识库文章 Hammer 内容导出失败，并带有 "Path '/the/path' is not an allowed export path "。

7.1. 如何使用导出和导入内容同步

使用导出和导入工作流同步内容的方法有多种：

- 您可以使用上游卫星服务器作为内容存储，这意味着您将同步整个库，而不是内容视图版本。此方法提供最简单的导出/导入工作流。在这种情况下，您可以管理下游版本。更多信息请参阅 第 7.1.1 节 “将上游卫星服务器用作内容存储”。
- 您可以使用上游卫星服务器同步内容视图版本。此方法可以提供对卫星服务器之间同步的内容的更多控制。更多信息请参阅 第 7.1.2 节 “使用上游卫星服务器同步内容视图版本”。
- 您同步单个存储库。如果您使用 Content-View 同步方法，但希望在不将其添加到现有的内容视图的情况下同步附加存储库，这会很有用。更多信息请参阅 第 7.1.3 节 “同步单存储库”。

7.1.1. 将上游卫星服务器用作内容存储

在这种情况下，您将使用上游卫星服务器作为更新的内容存储，而不是管理内容。您可以使用下游卫星服务器来管理隔离网络后的所有基础架构的内容。您从上游卫星服务器导出 Library 内容，并将它导入到下游卫星服务器中。

在上游 Satellite 服务器上

1. 确保软件仓库使用以下方法之一使用 **Immediate** 下载策略：
   a. 对于使用 **On Demand** 的现有软件仓库，请在存储库详情页面中将下载策略更改为 **Immediate**。
   b. 对于新的软件仓库，请确保在启用 **Red Hat 软件仓库前将 Default Red Hat Repository** 下载策略设置为 **Immediate**，并且 **Default download 策略** 已设为 **Immediate**。

   更多信息请参阅 第 4.8 节 “下载策略概述”。
2. 启用您要同步的内容。更多信息请参阅 第 4.5 节 “启用红帽软件仓库”。
   如果要同步自定义内容，首先创建一个自定义产品并同步产品存储库。
3. 同步启用的内容：
   a. 在第一个导出上，**执行完整的** Library 导出，以便导出所有同步的内容。这会生成内容存档，您可以稍后导入到一个或多个下游 Satellite 服务器。有关执行完整库导出的更多信息，请参阅 第 7.3 节 “导出库环境”。
   b. 逐步在上游卫星服务器上导出所有未来更新。这会生成包含最新更新集合的更精简内容存档。例如，如果您启用和同步新存储库，下一个导出的内容存档仅包含来自新启用的存储库的内容。有关执行增量库导出的更多信息，请参阅 第 7.5 节 “以正确方式导出库环境”。

在下游卫星服务器上

1. 将上游卫星服务器上导出的内容添加到硬盘。
2. 将它放置在 **/var/lib/pulp/imports** 下的目录中。
3. 使用 第 7.13 节 "导入到库环境中" 中概述的步骤将内容导入到机构。
然后，您可以根据需要通过内容视图或生命周期环境来管理内容。

7.1.2. 使用上游卫星服务器同步内容视图版本

在这种情况下，您将使用上游卫星服务器作为内容存储，还可以同步隔离网络后的所有基础架构的内容。您策展从 CDN 进入 Content Views 和 Lifecycle Environments 的更新。将内容提升到指定的生命周期环境后，您可以从上游卫星服务器导出内容，并将它导入到下游卫星服务器。

在上游 Satellite 服务器上

1. 确保软件仓库使用以下方法之一使用 **Immediate** 下载策略：
   a. 对于使用 **On Demand** 的现有软件仓库，请在存储库详情页面中将下载策略更改为 **Immediate**。
   b. 对于新的软件仓库，请确保在启用 **Red Hat 软件仓库前将 Default Red Hat Repository** 下载策略设置为 **Immediate**，并且 **Default download 策略** 已设为 **Immediate**。

   更多信息请参阅 第 4.8 节 "下载策略概述"。
2. 启用您要同步的内容。更多信息请参阅 第 4.5 节 "启用红帽软件仓库"。
如果要同步自定义内容，首先创建一个自定义产品并同步产品存储库。
3. 同步启用的内容：
   a. 对于第一个导出，请在您要导出的内容视图版本上执行 **完整的** 版本导出。更多信息请参阅 第 7.6 节 "导出内容视图版本"。这会生成您可以导入到一个或多个下游 Satellite 服务器的内容存档。
   b. 以递增方式导出连接的卫星服务器中的所有更新。这会生成更精简的内容存档，其中仅包含来自最新更新集的更改。例如，如果您的内容视图有新存储库，则此导出的内容存档仅包含最新的更改。更多信息请参阅 第 7.8 节 "以方式导出内容视图版本"。
   c. 当您有新内容时，重新发布包含此内容的内容视图，然后再导出递增。更多信息请参阅 第 6 章 管理内容视图。这会创建一个新的内容视图版本，其中包含要导出的内容。

在下游卫星服务器上

1. 将上游卫星服务器上导出的内容添加到硬盘。
2. 将它放置在 **/var/lib/pulp/imports** 下的目录中。
3. 将内容导入到您需要的组织。更多信息请参阅 第 7.14 节 "导入内容视图版本"。这将从导出的内容存档中创建内容视图版本，然后相应地导入内容。

7.1.3. 同步单存储库

在这种情况下，您会导出并导入单个软件仓库。

在上游 Satellite 服务器上

1. 确保软件仓库以以下方式之一使用 **Immediate** 下载策略：

   a. 对于使用 **On Demand** 的现有软件仓库，请在存储库详情页面中将下载策略更改为 **Immediate**。

   b. 对于新的软件仓库，请确保在启用 **Red Hat 软件仓库前将 Default Red Hat Repository** 下载策略设置为 **Immediate**，并且 **Default download 策略** 已设为 **Immediate**。

   更多信息请参阅 第 4.8 节 "下载策略概述"。

2. 启用您要同步的内容。更多信息请参阅 第 4.5 节 "启用红帽软件仓库"。
   如果要同步自定义内容，首先创建一个自定义产品并同步产品存储库。

3. 同步启用的内容：

   a. 在第一个导出上，**执行完整的** 存储库导出，以便导出所有同步的内容。这会生成内容存档，您可以稍后导入到一个或多个下游 Satellite 服务器。有关执行完整的存储库导出的详情，请参考 第 7.9 节 "导出存储库"。

   b. 逐步在上游卫星服务器上导出所有未来更新。这会生成包含最新更新集合的更精简内容存档。有关执行增量存储库导出的详情，请参考 第 7.11 节 "以方式导出存储库"。

在下游卫星服务器上

1. 将上游卫星服务器上导出的内容添加到硬盘。

2. 将它放置在 **/var/lib/pulp/imports** 下的目录中。

3. 将内容导入组织。请参阅 第 7.15 节 "导入存储库"。
   然后，您可以根据需要通过内容视图或生命周期环境来管理内容。

7.2. 同步自定义存储库

在使用 Inter-Satellite Synchronization Network Sync 时，会自动配置红帽存储库，但不会自动配置自定义存储库。使用这个流程，通过 Inter-Satellite Synchronization (ISS) Network Sync 将内容从一个位于已连接的 Satellite 服务器同步到一个断开连接的 Satellite 服务器。

在完成断开连接的 Satellite 服务器的步骤前，请遵循连接的 Satellite 服务器的步骤。

连接的 Satellite 服务器

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。

2. 点自定义产品。

3. 点自定义存储库。

4. 复制 **Published At:** URL。

5. 在断开连接的 Satellite 服务器上继续操作。

断开连接的 Satellite 服务器

1. 从连接的 Satellite 服务器下载 **katello-server-ca.crt** 文件：

```
# curl http://satellite.example.com/pub/katello-server-ca.crt
```

2. 创建一个 SSL 内容凭据，其内容为 **katello-server-ca.crt**。有关创建 SSL 内容凭证的详情，请参考 第 4.2 节 "导入自定义 SSL 证书"。
3. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
4. 使用以下方法创建自定义产品：
   - **Upstream URL** ：粘贴您之前复制的链接。
   - **SSL CA Cert** ：选择从您连接的 Satellite 服务器传输的 SSL 证书。

   有关创建自定义产品的更多信息，请参阅 第 4.3 节 "创建自定义产品"。

完成这些步骤后，断开连接的 Satellite 服务器就正确配置了自定义存储库。

7.3. 导出库环境

您可以将组织的 Library 环境中所有 Yum 存储库的内容导出到卫星服务器的存档文件中，并使用此存档文件在其他卫星服务器或另一个卫星服务器组织中创建相同的存储库。导出的存档文件包含以下数据：

- 包含内容视图版本元数据的 JSON 文件。
- 包含来自组织的 Library 环境的所有存储库的存档文件。

Satellite 服务器仅导出包括在内容视图版本中的 RPM 和 kickstart 文件。Satellite 不会导出以下内容：

- Docker 内容

先决条件

要导出机构的 Library 生命周期环境的内容，请确保要导出的 Satellite 服务器满足以下条件：

- 确保导出目录有可用的存储空间来容纳导出。
- 确保 **/var/lib/pulp/exports** 目录有与导出过程中创建的临时文件大小相等的存储空间。
- 确保为您导出的 Library 生命周期环境中的所有存储库将下载策略设置为 **Immediate**。更多信息请参阅 第 4.8 节 "下载策略概述"。
- 确保您同步您导出到所需日期的产品。

导出机构库内容

1. 使用机构名称或 ID 导出。

```
# hammer content-export complete library --organization="My_Organization"
```

2. 验证包含导出版本的内容视图的存档是否在导出目录中：

```
# ls -lh /var/lib/pulp/exports/My_Organization/Export-Library/1.0/2021-03-02T03-35-24-00-00
total 68M
-rw-r--r--. 1 pulp pulp 68M Mar  2 03:35 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335.tar.gz
```

```
-rw-r--r--. 1 pulp pulp 333 Mar  2 03:35 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335-toc.json
-rw-r--r--. 1 pulp pulp 443 Mar  2 03:35 metadata.json
```

您需要所有三个文件、**tar.gz**、**toc.json** 和 **metadata.json** 文件才能导入。

3. 在 组织中创建一个新的内容视图 **Export-Library**。此内容视图包含属于此机构的所有存储库。此内容视图的新版本会自动发布并导出。

使用块导出

在很多情况下，导出的存档内容可能为几 GB 大小。如果要将其分成较小的大小或块。您可以直接使用导出命令中的 **--chunk-size-gb** 标志来处理此功能。在以下示例中，您可以看到如何指定 **--chunk-size-gb=2** 以以 **2 GB** 块分割存档。

```
# hammer content-export complete library \
--chunk-size-gb=2 \
--organization="My_Organization"

Generated /var/lib/pulp/exports/My_Organization/Export-Library/2.0/2021-03-02T04-01-25-00-00/metadata.json

# ls -lh /var/lib/pulp/exports/My_Organization/Export-Library/2.0/2021-03-02T04-01-25-00-00/
```

7.4. 以可同步格式导出库环境

您可以将机构库环境中所有 yum 存储库、Kickstart 存储库和文件存储库的内容导出到可同步的格式，您可以使用该格式创建自定义 CDN，并通过 HTTP/HTTPS 同步自定义 CDN 的内容。

然后，您可以使用导入的 Satellite Server 或另一个 Satellite Server 组织中的本地 Web 服务器提供生成的内容。

您无法直接导入可同步格式导出。相反，在导入 Satellite 服务器上，您必须：

- 将生成的内容复制到可访问导入卫星服务器的 HTTP/HTTPS Web 服务器。
- 将 CDN 配置更新为 **Custom CDN**。
- 将 CDN URL 设置为指向 Web 服务器。
- 可选：如果 web 服务器需要它，请设置 SSL/TLS CA 凭证。
- 启用存储库。
- 同步存储库。

您可以从 Satellite 服务器以可同步格式导出以下内容：

- yum 软件仓库
- Kickstart 软件仓库
- 文件软件仓库

您无法导出 Ansible、DEB 和 Docker 内容。

导出中包含有软件包的目录，以 Yum 格式 **列出** 存储库的文件和元数据，它们可用于在导入卫星服务器中同步。

先决条件

- 确保将您导出的所有存储库的下载策略设置为 **Immediate**。更多信息请参阅 第 4.8 节 “下载策略概述”。
- 确保您将导出的产品同步到所需的日期。
- 确保导出内容的用户具有 **Content Exporter** 角色。

流程

1. 使用机构名称或 ID 导出：

```
# hammer content-export complete library \
--organization="My_Organization" \
--format=syncable
```

2. 可选：验证导出的内容位于导出目录中：

```
# du -sh /var/lib/pulp/exports/My_Organization/Export-My_Repository/1.0/2021-03-02T03-35-24-00-00
```

7.5. 以正确方式导出库环境

在系统资源方面，导出库内容可能非常昂贵。具有多个 Red Hat Enterprise Linux 树的组织可以在卫星服务器上占用几 GB 空间。

在这种情况下，您可以使用 **Incremental Export** 只导出上一个导出之后更改的内容。增量导出通常会导致存档文件小于完整导出。

以下示例显示组织库中所有存储库的增量导出。

流程

1. 创建一个增量导出：

```
# hammer content-export incremental library --organization="My_Organization"

Generated /var/lib/pulp/exports/My_Organization/Export-Library/3.0/2021-03-02T04-22-14-00-00/metadata.json
```

2. 可选：查看导出的数据：

```
# ls -lh /var/lib/pulp/exports/My_Organization/Export-Library/3.0/2021-03-02T04-22-14-00-00/
total 172K
-rw-r--r--. 1 pulp pulp 161K Mar  2 04:22 export-436882d8-de5a-48e9-a30a-17169318f908-20210302_0422.tar.gz
-rw-r--r--. 1 pulp pulp  333 Mar  2 04:22 export-436882d8-de5a-48e9-a30a-17169318f908-20210302_0422-toc.json
-rw-r--r--. 1 pulp pulp  492 Mar  2 04:22 metadata.json
```

7.6. 导出内容视图版本

您可以从卫星服务器导出内容视图的版本到存档文件，并使用此存档文件在其他卫星服务器或另一台卫星服务器组织上创建相同的内容视图版本。卫星将复合内容视图导出为普通内容视图。复合性质不会被保留。在导入导出的存档时，会在下游卫星服务器上创建或更新常规内容视图。导出的存档文件包含以下数据：

- 包含内容视图版本元数据的 JSON 文件
- 包含在 Content View 版本中包含的所有存储库的存档文件

Satellite 服务器仅导出添加至内容视图版本中的 RPM 和 kickstart 文件。Satellite 不会导出以下内容：

- Docker 内容
- 内容视图定义和元数据，如软件包过滤器。

先决条件

要导出内容视图，请确保要导出的 Satellite 服务器满足以下条件：

- 确保导出目录有可用的存储空间来容纳导出。
- 确保 **/var/lib/pulp/exports** 目录有与导出过程中创建的临时文件大小相等的存储空间。
- 确保将下载策略设置为 **Immediate**，用于您导出的 Content View 中所有存储库。更多信息请参阅 第 4.8 节 “下载策略概述”。
- 确保您同步您导出到所需日期的产品。
- 确保导出内容的用户具有 **Content Exporter** 角色。

要导出内容视图版本

1. 列出可用于导出的 Content View 版本：

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

---|----------|---------|-------------|----------------------
ID | NAME     | VERSION | DESCRIPTION | LIFECYCLE ENVIRONMENTS
---|----------|---------|-------------|----------------------
5  | view 3.0 | 3.0     |             | Library
4  | view 2.0 | 2.0     |             |
3  | view 1.0 | 1.0     |             |
---|----------|---------|-------------|----------------------
```

导出内容视图版本

1. 获取所需版本的版本号。以下示例目标为导出版本 **1.0**。

```
# hammer content-export complete version \
--content-view="Content_View_Name" \
--version=1.0 \
--organization="My_Organization"
```

2. 验证包含导出版本的内容视图的存档是否在导出目录中：

```
# ls -lh /var/lib/pulp/exports/My_Organization/Content_View_Name/1.0/2021-02-25T18-59-26-00-00/
```

您需要所有三个文件，例如 **tar.gz** 存档文件， **toc.json** 和 **metadata.json** 才能成功导入内容。

使用块导出

在很多情况下，导出的存档内容可以是几 GB 大小。您可能想要分割大小较小的大小或块。您可以在 **hammer content-export** 命令中使用 **--chunk-size-gb** 选项来处理此功能。以下示例使用 **--chunk-size-gb=2** 将存档分成 **2 GB** 块。

```
# hammer content-export complete version \
--chunk-size-gb=2 \
--content-view="Content_View_Name" \
--organization="My_Organization" \
--version=1.0
# ls -lh /var/lib/pulp/exports/My_Organization/view/1.0/2021-02-25T21-15-22-00-00/
```

7.7. 以可同步格式导出内容视图版本

您可以将内容视图的版本导出到可同步格式，您可以使用它来创建自定义 CDN，并通过 HTTP/HTTPS 从自定义 CDN 同步内容。

然后，您可以使用导入的 Satellite Server 或另一个 Satellite Server 组织中的本地 Web 服务器提供生成的内容。

您无法直接导入可同步格式导出。相反，在导入 Satellite 服务器上，您必须：

- 将生成的内容复制到可访问导入卫星服务器的 HTTP/HTTPS Web 服务器。
- 将 CDN 配置更新为 **Custom CDN**。
- 将 CDN URL 设置为指向 Web 服务器。
- 可选：如果 web 服务器需要它，请设置 SSL/TLS CA 凭证。
- 启用存储库。
- 同步存储库。

您可以从 Satellite 服务器以可同步格式导出以下内容：

- yum 软件仓库
- Kickstart 软件仓库
- 文件软件仓库

您无法导出 Ansible、DEB 和 Docker 内容。

导出中包含有软件包的目录，以 Yum 格式 **列出** 存储库的文件和元数据，它们可用于在导入卫星服务器中同步。

先决条件

- 确保将您导出的内容视图中所有存储库的下载策略设置为 **Immediate**。更多信息请参阅 第 4.8 节 "下载策略概述"。
- 确保您将导出的产品同步到所需的日期。
- 确保导出内容的用户具有 **Content Exporter** 角色。

要导出内容视图版本

- 列出可用于导出的 Content View 版本：

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

流程

1. 获取所需版本的版本号。以下示例目标版本 **1.0** 用于导出：

```
# hammer content-export complete version \
--content-view="Content_View_Name" \
--version=1.0 \
--organization="My_Organization" \
--format=syncable
```

2. 可选：验证导出的内容位于导出目录中：

```
# ls -lh /var/lib/pulp/exports/My_Organization/My_Content_View_Name/1.0/2021-02-25T18-59-26-00-00/
```

7.8. 以方式导出内容视图版本

对于系统资源而言，导出完整版本会非常昂贵。具有多个 Red Hat Enterprise Linux 树的内容视图版本可占用卫星服务器上的几 GB 空间。

在这种情况下，您可以使用 **Incremental Export** 只导出上一个导出之后更改的内容。增量导出通常会导致存档文件小于完整导出。

在以下示例中，版本 **2.0** 是导出的目标，因为版本 **1.0** 在以前已被导出。

流程

1. 创建一个增量导出：

```
# hammer content-export incremental version \
--content-view="Content_View_Name" \
--organization="My_Organization" \
--version=2.0
```

2. 可选：查看导出的内容视图：

```
# ls -lh /var/lib/pulp/exports/My_Organization/view/2.0/2021-02-25T21-45-34-00-00/
```

7.9. 导出存储库

您可以从卫星服务器在组织的 Library 环境中导出存储库的内容。您可以使用此存档文件在另一个卫星服务器或另一个卫星服务器组织中创建相同的存储库。

您可以从 Satellite 服务器导出以下内容：

- Ansible 软件仓库
- Kickstart 软件仓库
- yum 软件仓库
- 文件软件仓库
- Docker 内容

导出包含以下数据：

- 包含存储库元数据的两个 JSON 文件。
- 一个或多个存档文件，其中包含来自机构的库环境的存储库的内容。

您需要所有文件 **tar.gz**、**toc.json** 和 **metadata.json** 才能导入。

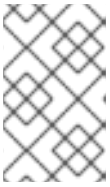
先决条件

- 确保导出目录有足够的可用空间来容纳导出。
- 确保 **/var/lib/pulp/exports** 目录有足够的可用存储空间，相当于您要导出的所有存储库的大小。
- 确保将下载策略设置为 **Immediate**，用于您导出的 Library 生命周期环境中的库。更多信息请参阅 第 4.8 节 “下载策略概述”。
- 确保将您导出的产品同步到所需日期。

流程

1. 使用产品名称和存储库名称导出存储库：

```
# hammer content-export complete repository \
--name="My_Repository" \
--product="My_Product"
```



注意

导出的存档的大小取决于存储库中的软件包的数量和大小。如果要将导出的存档分成块，使用 **--chunk-size-gb** 参数导出您的存储库，以 GB 为单位限制整数值，如 **---chunk-size-gb=2**。

2. 可选：验证导出的存档是否在导出目录中：

```
# ls -lh /var/lib/pulp/exports/My_Organization/Export-My_Repository/1.0/2022-09-02T03-35-24-00-00/
```

7.10. 以可同步格式导出存储库

您可以将组织的 Library 环境中存储库的内容导出到可同步的格式，您可以使用它来创建自定义 CDN 并通过 HTTP/HTTPS 同步自定义 CDN 的内容。

然后，您可以使用导入的 Satellite Server 或另一个 Satellite Server 组织中的本地 Web 服务器提供生成的内容。

您无法直接导入可同步格式导出。相反，在导入 Satellite 服务器上，您必须：

- 将生成的内容复制到可访问导入卫星服务器的 HTTP/HTTPS Web 服务器。
- 将 CDN 配置更新为 **Custom CDN**。
- 将 CDN URL 设置为指向 Web 服务器。
- 可选：如果 web 服务器需要它，请设置 SSL/TLS CA 凭证。
- 启用存储库。
- 同步存储库。

您可以从 Satellite 服务器以可同步格式导出以下内容：

- yum 软件仓库
- Kickstart 软件仓库
- 文件软件仓库

您无法导出 Ansible、DEB 和 Docker 内容。

导出中包含有软件包的目录，以 Yum 格式 **列出** 存储库的文件和元数据，它们可用于在导入卫星服务器中同步。

前提条件

- 确保将您导出的库生命周期环境中的存储库的下载策略设置为 **Immediate**。更多信息请参阅 第 4.8 节 “下载策略概述”。

流程

1. 使用存储库名称或 ID 导出存储库：

```
# hammer content-export complete repository \
--organization="My_Organization" \
--product="My_Product" \
--name="My_Repository" \
--format=syncable
```

2. 可选：验证导出的内容位于导出目录中：

```
# du -sh /var/lib/pulp/exports/My_Organization/Export-My_Repository/1.0/2021-03-02T03-35-24-00-00
```

7.11. 以方式导出存储库

在系统资源方面，导出存储库可能非常昂贵。典型的 Red Hat Enterprise Linux 树可占用卫星服务器上的几 GB 空间。

在这种情况下，您可以使用 **Incremental Export** 只导出上一个导出之后更改的内容。增量导出通常会导致存档文件小于完整导出。

以下示例显示库生命周期环境中存储库的增量导出。

流程

1. 创建一个增量导出：

```
# hammer content-export incremental repository \
 --organization="My_Organization" \
 --product="My_Product" \
 --name="My_Repository"

Generated /var/lib/pulp/exports/My_Organization/Export-My_Repository/3.0/2021-03-02T03-35-24-00-00/metadata.json
```

2. 可选：查看导出的数据：

```
# ls -lh /var/lib/pulp/exports/My_Organization/Export-My_Repository/3.0/2021-03-02T03-35-24-00-00/
total 172K
-rw-r--r--. 1 pulp pulp  20M Mar  2 04:22 export-436882d8-de5a-48e9-a30a-17169318f908-20210302_0422.tar.gz
-rw-r--r--. 1 pulp pulp  333 Mar  2 04:22 export-436882d8-de5a-48e9-a30a-17169318f908-20210302_0422-toc.json
-rw-r--r--. 1 root root  492 Mar  2 04:22 metadata.json
```

7.12. 持续跟踪您的导出

卫星保留所有导出的记录。每次在上游卫星服务器上导出内容时，将记录并维护该导出，以供将来查询。您可以使用记录来组织和管理导出，这在递增导出时特别有用。

当为几个下游 Satellite 服务器导出来自上游 Satellite 服务器的内容时，您还可以跟踪为特定服务器导出的内容。这有助于您跟踪导出哪些内容以及从哪里导出。

在导出期间使用 **--destination-server** 参数来指示目标服务器。这个选项可用于所有内容 **导出** 操作。

跟踪库导出的目的地

- 在导出库时指定目标服务器：

```
# hammer content-export complete library \
--destination-server=My_Downstream_Server_1 \
--organization="My_Organization" \
```

```
--version=1.0
```

跟踪内容视图导出的目的地

- 在导出内容视图版本时指定目标服务器：

```
# hammer content-export complete version \
--content-view="Content_View_Name" \
--destination-server=My_Downstream_Server_1 \
--organization="My_Organization" \
--version=1.0
```

查询出口记录

- 使用以下命令列出内容导出：

```
# hammer content-export list --organization="My_Organization"
```

7.13. 导入到库环境中

您可以将导出的库内容导入到另一个卫星服务器上组织的 Library 生命周期环境中。有关从库环境中导出内容的更多信息，请参阅 第 7.3 节 “导出库环境”。

先决条件

- 导出的文件必须位于 **/var/lib/pulp/imports** 下的目录中。
- 如果导出内容中有红帽存储库，导入机构的清单必须包含导出中包含的产品的订阅。
- 导入内容的用户必须具有 Content Importer 角色。

流程

1. 将导出的文件复制到您要导入的卫星服务器上 **/var/lib/pulp/imports** 的子目录。
2. 将导入目录及其内容的所有权设置为 **pulp:pulp**。

```
# chown -R pulp:pulp /var/lib/pulp/imports/2021-03-02T03-35-24-00-00
# ls -lh /var/lib/pulp/imports/2021-03-02T03-35-24-00-00
total 68M
-rw-r--r--. 1 pulp pulp 68M Mar  2 04:29 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335.tar.gz
-rw-r--r--. 1 pulp pulp 333 Mar  2 04:29 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335-toc.json
-rw-r--r--. 1 pulp pulp 443 Mar  2 04:29 metadata.json
```

3. 确定您要导入的机构。
4. 要将 Library 内容导入到 Satellite Server，请输入以下命令：

```
# hammer content-import library \
--organization="My_Organization" \
--path=/var/lib/pulp/imports/2021-03-02T03-35-24-00-00
```

请注意，您必须输入完整路径 **/var/lib/pulp/imports/My_Exported_Library_Dir**。相对路径不起作用。

5. 要验证您导入了库内容，请检查 Product 和 Repositories 的内容。目标组织中创建名为 **Import-Library** 的新内容视图。此内容视图用于促进库内容导入。
默认情况下，卫星 Web UI 中不会显示此内容视图。**import-Library** 并不旨在直接分配给主机。而是像您正常一样，将主机分配到 **默认组织** 视图或其他内容视图。

7.14. 导入内容视图版本

您可以导入导出的内容视图版本，以便在另一个卫星服务器的组织中创建包含相同内容的版本。有关导出内容视图版本的详情，请参考 第 7.6 节 "导出内容视图版本"。

当您导入内容视图版本时，它具有相同的主版本和次版本号，并包含相同的软件包和勘误表库。如果导入的组织中不存在，则会自动创建自定义存储库和内容视图。

先决条件

- 导出的文件必须位于 **/var/lib/pulp/imports** 下的目录中。
- 如果导出内容中有红帽存储库，导入机构的清单必须包含导出中包含的产品的订阅。
- 导入内容视图版本的用户必须具有 Content Importer 角色。

流程

1. 将导出的文件复制到您要导入的卫星服务器上 **/var/lib/pulp/imports** 的子目录。
2. 将导入目录及其内容的所有权设置为 **pulp:pulp**。

```
# chown -R pulp:pulp /var/lib/pulp/imports/2021-02-25T21-15-22-00-00/
```

3. 验证所有权是否已正确设置：

```
# ls -lh /var/lib/pulp/imports/2021-02-25T21-15-22-00-00/
```

4. 要将 Content View 版本导入到 Satellite 服务器，请输入以下命令：

```
# hammer content-import version \
--organization-id=My_Organization_ID \
--path=/var/lib/pulp/imports/2021-02-25T21-15-22-00-00/
```

请注意，您必须输入完整路径 **/var/lib/pulp/imports/My_Exported_Version_Dir**。相对路径不起作用。

5. 要验证您是否成功导入了 Content View 版本，列出您的机构的 Content View 版本：

```
# hammer content-view version list \
--organization-id=My_Organization_ID
```

7.15. 导入存储库

您可以将导出的存储库导入到另一卫星服务器上的组织中。有关导出存储库内容的更多信息，请参阅第 7.9 节 "导出存储库"。

先决条件

- *导出文件必须位于 **/var/lib/pulp/imports** 下的目录中。*
- *如果导出包含任何红帽存储库，导入机构的清单必须包含导出中包含的产品的订阅。*
- *导入内容的用户必须具有 Content Importer 角色。*

流程

1. *将导出的文件复制到您要导入的卫星服务器上 **/var/lib/pulp/imports** 的子目录。*
2. *将导入目录及其内容的所有权设置为 **pulp:pulp**。*

```
# chown -R pulp:pulp /var/lib/pulp/imports/2021-03-02T03-35-24-00-00
# ls -lh /var/lib/pulp/imports/2021-03-02T03-35-24-00-00
total 68M
-rw-r--r--. 1 pulp pulp 68M Mar  2 04:29 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335.tar.gz
-rw-r--r--. 1 pulp pulp 333 Mar  2 04:29 export-1e25417c-6d09-49d4-b9a5-23df4db3d52a-20210302_0335-toc.json
-rw-r--r--. 1 pulp pulp 443 Mar  2 04:29 metadata.json
```

3. *确定您要导入的机构。*
4. *要将 Library 内容导入到 Satellite Server，请输入以下命令：*

```
# hammer content-import repository \
--organization="My_Organization" \
--path=/var/lib/pulp/imports/2021-03-02T03-35-24-00-00
```

*请注意，您必须输入完整路径 **/var/lib/pulp/imports/My_Exported_Repo_Dir**。相对路径不起作用。*

5. *要验证您导入了存储库，请检查 Product 和 Repository 的内容。*

7.16. 使用 HAMMER CLI 清理和导入内容

表 7.1. Export

意图	命令
完整导出机构库	**Hammer 内容导出完整库 --organization="*My_Organization*"**
逐步导出组织的库（假设您之前已导出内容）	**Hammer 内容导出增量库 --organization="*My_Organization*"**
完整导出内容视图版本	**Hammer content-export complete version --content-view="*My_Content_View*" --version=1.0 --organization="*My_Organization*"**

意图	命令
导出提升到 Dev 环境的内容视图版本	**Hammer content-export 完整版本 --content-view="*My_Content_View*" --organization="*My_Organization*" --lifecycle-environment="Dev"**
在较小的块中导出内容视图 (2-GB slab)	**Hammer content-export complete version --content-view="*My_Content_View*" --version=1.0 --organization="*My_Organization*" --chunk-size-gb=2**
逐步导出内容视图版本（假设之前已导出内容）	**Hammer content-export incremental version --content-view="*My_Content_View*" --version=2.0 --organization="*My_Organization*"**
完全导出存储库	**Hammer content-export complete repository --product="*My_Product*" --name="*My_Repository*" --organization="*My_Organization*"**
逐步导出存储库（假设之前已导出内容）	**Hammer content-export incremental repository --product="*My_Product*" --name="*My_Repository*" --organization="*My_Organization*"**
列出导出	**Hammer content-export list --content-view="*My_Content_View*" --organization="*My_Organization*"**

表 7.2. Import

意图	命令
导入到组织的库	**Hammer content-import library --organization="*My_Organization*" --path="/var/lib/pulp/imports/*My_Exported_ Plus_Dir*"**
导入到内容视图版本	**Hammer content-import version --organization="*My_Organization*" --path="/var/lib/pulp/imports/*My_Exported_Version_Dir*"**
导入存储库	**Hammer content-import repository --organization="*My_Organization*" --path="/var/lib/pulp/imports/*My_Exported_Repo_Dir*"**

第 8 章 管理激活码

激活码提供了一种自动注册系统并订阅附加的方法。您可以创建多个密钥，并将它们与不同的环境和内容视图相关联。例如，您可以使用 Red Hat Enterprise Linux 工作站订阅创建基本激活码，并将其与特定环境中的内容视图相关联。

您可以在内容主机注册过程中使用激活码来提高流程的速度、简单性和一致性。请注意，激活密钥仅在主机注册时使用。如果对激活码进行了更改，它仅适用于将来使用已发送的激活码注册的主机。不会对现有主机进行更改。

激活码可以为内容主机定义以下属性：

- 关联的订阅和订阅附加行为
- 可用产品和存储库
- 生命周期环境和内容视图
- 主机集合成员资格
- 系统目的

主机创建与注册之间的内容视图冲突

置备主机时，Satellite 使用您在主机组或主机设置中设置的内容视图中的置备模板和其他内容。注册主机后，激活密钥的 Content View 将覆盖主机组或主机设置中的原始内容视图。然后，卫星将激活密钥中的 Content View 用于后续任务，例如重建主机。

在重建主机时，请确保设置了您要在激活密钥中使用的内容视图，而不是在主机组或主机设置中使用。

将 Same Activation Key 与多个内容主机搭配使用

如果包含足够订阅，则可以将相同的激活码应用到多个内容主机。但是，激活密钥只为内容主机设置初始配置。将内容主机注册到组织后，可以手动将该组织的内容附加到内容主机。

将多个激活码用于内容主机

内容主机可以关联多个激活密钥，它们组合用于定义主机设置。如果设置有冲突，则最后指定的激活码优先。您可以通过设置主机组参数来指定优先级顺序，如下所示：

```
$ hammer hostgroup set-parameter \
--hostgroup "My_Host_Group" \
--name "My_Activation_Key" \
--value "name_of_first_key", "name_of_second_key", ...
```

8.1. 创建激活码

您可以使用激活码来定义在注册过程中将特定的订阅附加到主机。要添加到激活密钥的订阅必须在关联的内容视图中可用。

订阅管理器会根据以下因素的不同附加订阅：

- 是否有与激活码关联的订阅？
- auto-attach 选项是否已启用？
- 对于 Red Hat Enterprise Linux 8 主机：激活码中是否设置了系统目的？

请注意，Satellite 会自动为主机上安装的产品附加订阅。对于默认情况下没有列出在 Red Hat Enterprise Linux 中安装的产品（如延长更新支持(EUS)订阅）的订阅，请使用指定所需订阅的激活码，并禁用自动附加。

根据前面的因素，有三种可能使用激活码订阅的情况：

1. 自动附加订阅的激活码。
   如果没有指定和自动附加订阅，则使用激活密钥搜索主机，从与激活密钥关联的内容视图提供的内容视图提供的最佳订阅。这与输入 **subscription-manager --auto-attach** 命令类似。对于 Red Hat Enterprise Linux 8 主机，您可以将激活码配置为在注册的过程中在主机上设置系统目的，以增强自动订阅附加。

2. 为自动附加提供自定义订阅集的激活码。
   如果指定订阅并且启用了自动附加，则使用激活密钥的主机从激活密钥中指定的列表中选择最适合的订阅。在激活码中设置系统目的不会影响这种情况。

3. 带有准确的订阅集的激活码。
   如果指定订阅并且禁用了自动附加，则使用该激活密钥的主机将与激活密钥中指定的所有订阅相关联。在激活码中设置系统目的不会影响这种情况。

自定义产品

如果自定义产品（通常包含未由红帽提供的内容）分配给一个激活码，那么无论自动附加设置如何，这个产品始终为注册的内容主机启用。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Activation keys**，再点击 **Create Activation Key**。
2. 在 **Name** 字段中输入激活码的名称。
3. 如果要设置限制，请清除 **无限主机** 复选框，然后在 **Limit** 字段中输入您可以使用激活码注册的最大系统数量。如果您希望没有限制的主机使用激活吗进行注册，确保选中了 **Unlimited Hosts** 复选框。
4. 可选：在 **Description** 字段中输入激活码的描述。
5. 从 **Environment** 列表中，选择要使用的环境。
6. 从 **Content View** 列表中，选择要使用的内容视图。如果要使用已弃用的 **Katello Agent** 而不是 **Remote Execution**，则 Content View 必须包含 Satellite Client 6 存储库，因为它包含 **katello-agent** 软件包。
7. 如果启用了简单内容访问(SCA)：
   a. 在 **Repository Sets** 选项卡中，请确保只启用命名的存储库。
8. 如果没有启用 SCA：
   a. 单击订阅选项卡，然后单击 **添加** 子菜单。
   b. 单击您之前创建的订阅旁边的复选框。
   c. 单击 **Add Selected**。
9. 点击 **Save**。

10. 可选：对于 Red Hat Enterprise Linux 8 主机，在 **System Purpose** 部分中，您可以使用系统目的配置激活码，以便在注册过程中在主机上设置，以增强订阅自动附加。

CLI 过程

1. 创建激活码：

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

2. 可选：对于 Red Hat Enterprise Linux 8 主机，输入以下命令在注册期间使用系统目的在主机上设置激活码，以增强订阅自动附加。

```
# hammer activation-key update \
--organization "My_Organization" \
--name "My_Activation_Key" \
--service-level "Standard" \
--purpose-usage "Development/Test" \
--purpose-role "Red Hat Enterprise Linux Server" \
--purpose-addons "addons"
```

3. 获取您的订阅 ID 列表：

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

4. 将 Red Hat Enterprise Linux 订阅 UUID 附加到激活码中：

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

5. 列出与激活码关联的产品内容：

   a. 如果启用了简单内容访问(SCA)：

```
# hammer activation-key product-content \
--content-access-mode-all true \
--name "My_Activation_Key" \
--organization "My_Organization"
```

   b. 如果没有启用 SCA：

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

6. 覆盖 Satellite Client 6 存储库的默认 auto-enable 状态。默认状态设置为 disabled。要启用，请输入以下命令：

```
# hammer activation-key content-override \
--name "My_Activation_Key" \
--content-label rhel-7-server-satellite-client-6-rpms \
--value 1 \
--organization "My_Organization"
```

8.2. 更新与激活码关联的订阅

使用这个步骤更改与激活码关联的订阅。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

请注意，对激活码的更改仅适用于更改后置备的机器。要更新现有内容主机上的订阅，请参阅 第 3.7 节 "在多个主机上更新红帽订阅"。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Activation keys**，再点击激活码的名称。
2. 点 **Subscriptions** 选项卡。
3. 要删除订阅，请选择 **List/Remove**，然后选择要删除的订阅左侧的复选框，然后单击 **Remove Selected**。
4. 要添加订阅，请选择 **Add**，然后选择要添加的订阅左侧的复选框，然后单击添加所 选项。
5. 点 **Repository Sets** 选项卡，并检查存储库的状态设置。
6. 要启用或禁用存储库，可选择存储库的复选框，然后使用 **Select Action** 列表更改状态。
7. 单击 **Details** 选项卡，选择此激活密钥的 Content View，然后单击 **Save**。

CLI 过程

1. 列出激活码当前包含的订阅：

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

2. 从激活码中删除所需的订阅：

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

对于 **--subscription-id** 选项，您可以使用 UUID 或订阅 ID。

3. 将新订阅附加到激活码中：

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

对于 **--subscription-id** 选项，您可以使用 UUID 或订阅 ID。

4. 列出与激活码关联的产品内容：

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

5. 覆盖所需存储库的默认自动启用状态：

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

对于 **--value** 选项，在启用时输入 **1**，**0** 代表 disable。

8.3. 使用激活码进行主机注册

您可以使用激活码完成以下任务：

- 通过红帽卫星在调配期间注册新主机。Red Hat Satellite 中的 kickstart 虚拟机模板包含用于使用创建主机时定义的激活密钥注册主机的命令。
- 注册现有的 Red Hat Enterprise Linux 主机。将 Subscription Manager 配置为使用 Satellite Server 进行注册，并在运行 **subscription-manager register** 命令时指定激活码。

您可以使用主机注册功能、卫星 API 或 Hammer CLI 使用卫星注册主机。

流程

1. 在 Satellite Web UI 中，导航到 **Hosts > Register Host**。
2. 点击 **Generate** create registration 命令。
3. 点击 文件 图标将命令复制到您的剪贴板中。
4. 登录您要注册的主机并运行之前生成的命令。
5. 检查 **/etc/yum.repos.d/redhat.repo** 文件，并确保启用了适当的软件仓库。

CLI 过程

1. 使用 Hammer CLI 生成主机注册命令：

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

如果您的主机不信任 Satellite 服务器的 SSL 证书，您可以通过在注册命令中添加 **--insecure** 标志来禁用 SSL 验证。

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

2. 登录您要注册的主机并运行之前生成的命令。

3. 检查 **/etc/yum.repos.d/redhat.repo** 文件，并确保启用了适当的软件仓库。

API 流程

1. 使用 Satellite API 生成主机注册命令：

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

如果您的主机不信任 Satellite 服务器的 SSL 证书，您可以通过在注册命令中添加 **--insecure** 标志来禁用 SSL 验证。

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

使用激活码来简化指定环境。如需更多信息，请参阅 管理内容 中的 管理激活密钥。

要以命令行参数输入密码，请使用 **username:password** 语法。请记住，这可以在 shell 历史记录中保存密码。

有关注册的更多信息，请参阅管理主机中的将主机注册到 Red Hat Satellite 。

2. 登录您要注册的主机并运行之前生成的命令。

3. 检查 **/etc/yum.repos.d/redhat.repo** 文件，并确保启用了适当的软件仓库。

多个激活码

在注册内容主机时，您可以使用多个激活码。然后，您可以为特定订阅集合创建激活码，并根据内容主机要求合并它们。例如，以下命令在带有 VDC 和 OpenShift 订阅的机构中注册了内容主机：

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

设置冲突

如果激活码中有冲突设置，则最右侧的键具有优先权。

- 有冲突的设置：**Service Level**, **Release Version**, **Environment**, **Content View**, 和 **Product Content**。
- 没有冲突的设置，主机会发生冲突： **Subscriptions** 和 **Host Collections**。

- 影响密钥本身行为而非主机配置的设置： **内容主机限制和 Auto-Attach**。

8.4. 启用自动附加

当在激活密钥上启用了自动附加并且有与密钥关联的订阅时，订阅管理服务根据当前安装的产品、架构以及服务级别等一组条件选择并附加最匹配的相关订阅。

您可以启用自动附加，并且没有与密钥关联的订阅。当您不希望虚拟机消耗物理订阅时，这种密钥通常用于注册虚拟机，而是从虚拟机监控程序继承基于主机的订阅。如需更多信息，请参阅在 Red Hat Satellite 中配置虚拟机订阅。

auto-attach 会被默认启用。如果要强制附加与激活密钥关联的所有订阅，请禁用该选项。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Activation Keys**。
2. 点您要编辑的激活码名称。
3. 点 **Subscriptions** 选项卡。
4. 点 **Auto-Attach** 旁边的编辑图标。
5. 选择或清除复选框以启用或禁用自动附加。
6. 点击 **Save**。

CLI 过程

- 输入以下命令在激活码中启用自动附加：

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

8.5. 设置服务级别

您可以配置激活码，为使用该激活密钥创建的新主机定义默认服务级别。设置默认服务级别仅选择要附加到主机的匹配订阅。例如，如果激活密钥中的默认服务级别被设置为 Premium(Premium)服务级别，则只会在注册后将具有高级服务级别的订阅附加到主机。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Activation Keys**。
2. 点您要编辑的激活码名称。
3. 点 **Service Level** 旁边的编辑图标。
4. 从列表中选择所需的服务级别。列表中仅包含可用于激活密钥的服务级别。
5. 点击 **Save**。

CLI 过程

- *输入以下命令在激活码中将默认服务级别设置为 Premium：*

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

第 9 章 管理勘误

作为红帽质量控制和发布过程的一部分，我们为客户提供了每个官方红帽 RPM 版本的更新。红帽将相关软件包组编译为一个**勘误（erratum）** 以及一个提供更新描述的公告。公告类型有三种（按重要程度排列）：

安全公告

描述在软件包中找到的固定安全问题。问题的安全影响可以是 Low、Moderate、Important 或 Critical。

程序错误修复建议

描述软件包的程序错误修复。

产品增强公告

描述软件包中添加的功能增强和新功能。

当与红帽的内容交付网络(CDN)同步存储库时，红帽卫星会导入这个勘误信息。红帽卫星还提供检查和过滤勘误表的工具，允许对勘误表进行精确更新管理。这样，您可以选择相关的更新，并通过 Content Views 传播它们到所选内容主机。

勘误根据它们所包含的最重要的公告类型进行标记。因此，标记为产品增强 **公告** 的勘误只能包含增强更新，而 **程序错误修复** 咨询勘误可同时包含程序错误修正和增强，**安全公告** 中可以包含所有三种类型。

在 Red Hat Satellite 中，有两个关键字来描述与可用内容主机的勘误关系：

如果适用

一个适用于一个或多个内容主机的勘误，这意味着它会更新内容主机上存在的软件包。虽然这些勘误会应用到内容主机，在其状态变为 **Installable** 之前，勘误还不能安装。可安装的勘误可自动适用。

可安装

适用于一个或多个内容主机的勘误，并可在内容主机上安装。可安装勘误表可从生命周期环境和相关内容视图提供给内容主机，但尚未安装。

本章介绍了如何管理勘误表并将其应用到一台主机或多个主机。

9.1. 检查可用的勘误

以下流程描述了如何查看和过滤可用的勘误以及如何显示所选公告的元数据。要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Errata** 以查看可用勘误列表。
2. 使用页面顶部的过滤工具来限制显示勘误表的数量：
   - 从列表中选择要检查的存储库。**所有** 软件仓库都被默认选择。
   - **默认选择适用** 复选框，以只查看所选存储库中的适用勘误。选择 **Installable** 复选框可以只查看被标记为 installable 的勘误。
   - 要搜索勘误表，在 **Search** 字段中输入查询，格式为：

     ```
     parameter operator value
     ```

     有关可用于搜索的参数列表，请参阅 第 9.2 节 “可用于勘误搜索的参数”。在 Administering

Red Hat Satellite 中查找 Granular Search 中的适用 Operator 列表。自动建议在您选择的情况下可以正常工作。您还可以将查询与使用 and **和** **或** 运算符合并。例如，只显示与**内核**软件包相关的安全公告，请输入：

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

按 **Enter** 键开始搜索。

3. 点击您要检查 **的勘误 ID** ：
   - **Details** 选项卡包含更新软件包的描述以及更新提供的重要修复和增强文档。
   - 在 **Content Hosts** 选项卡中，您可以将勘误应用到所选内容主机，如 第 9.9 节 “将勘误应用到多个主机” 所述。
   - **Repositories** 选项卡列出已包含勘误的存储库。您可以根据环境和内容视图过滤存储库，并通过存储库名称搜索它们。

您还可以使用新的 Host 页面查看 来检查可用的勘误表并选择要安装的勘误表。

1. 在 Satellite Web UI 中，导航到 **Hosts > All Hosts** 并选择您需要的主机。
2. 如果主机存在勘误，则新主机页面中的 Installable Errata 卡会显示交互式的饼图，显示安全公告、错误修正和增强。
3. 在新的 Host 页面上，选择 **Content** 选项卡。
4. 在 Content 页面上，选择 **Errata** 选项卡。
5. 页面中显示所选主机的可安装勘误表。
6. 单击您要安装的任何勘误的复选框。
7. 通过使用您要添加到主机的勘误旁的垂直 ellipsis 图标，如果您没有使用 SSH 连接到目标主机，请选择 **Apply via Katello 代理**。
8. 选择 **Apply via Remote Execution** 以使用 Remote Execution，或者在要自定义远程执行时使用 **应用**。
9. 点 **Submit**。

CLI 过程

- 要查看所有机构都可用的勘误，请输入以下命令：

```
# hammer erratum list
```

- 要查看特定勘误的详情，请输入以下命令：

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

- 您可以使用 **--search** 选项输入查询来搜索勘误。例如，要查看包含指定错误的所选产品的适用勘误，以便在 top 上显示安全勘误，请输入以下命令：

```
# hammer erratum list \
--product-id 7 \
```

```
--search "bug = 1213000 or bug = 1207972" \
--errata-restrict-applicable 1 \
--order "type desc"
```

9.2. 可用于勘误搜索的参数

参数	描述	示例
错误	按照 Bugzilla 编号搜索。	*bug = 1172165*
CVE	按照 CVE 号搜索。	*cve = CVE-2015-0235*
id	按照勘误表 ID 搜索。Auto-suggest 系统根据类型显示可用 ID 列表。	*id = RHBA-2014:2004*
发布日期	按照问题日期搜索。您可以指定具体日期，如 "Feb16,2015" 或使用关键字，如 "Yesterday" 或 "1 hour ago"。可使用 "<" 和 ">" 运算符指定时间范围。	*发布 < "Jan 12,2015"*
package	按照完整的软件包构建名称搜索。Auto-suggest 系统在键入时显示可用软件包的列表。	*package = glib2-2.22.5-6.el6.i686*
package_name	按照软件包名称搜索。Auto-suggest 系统在键入时显示可用软件包的列表。	*package_name = glib2*
严重性	按照安全更新解决的问题的严重性搜索。指定 *Critical*、*Important* 或 *Moderate* （中度）。	*重要性 = Critical*
title	按照公告标题搜索。	*title ~ openssl*
type	按照公告类型搜索。指定 *安全性*、*错误修正* 或*功能增强*。	*type = bugfix*
已更新	按照最后一次更新的日期搜索。您可以使用与 issued 参数 **相同的格式**。	*updated = "6 days ago"*

9.3. 应用安装勘误表

使用以下步骤查看可安装的勘误列表，并选择要安装的勘误。

流程

1. 在 Satellite Web UI 中，导航到 **Hosts > All Hosts** 并选择您需要的主机。
2. 如果存在与主机关联的勘误，则在新的 Host 页面的 Installable Errata 卡中会显示它们。
3. 在 **内容** 选项卡中，**勘误** 显示所选主机的可安装勘误表。
4. 单击您要安装的任何勘误的复选框。
5. 使用您要添加到主机的勘误旁的垂直 ellipsis 图标，选择 **Apply via Remote Execution** 来使用 Remote Execution。如果要自定义 **远程执行**，请选择 **Apply via** custom remote execution，或者如果您没有使用 SSH 连接到目标主机，选择 **Apply via Katello 代理**。
6. 点 **Submit**。

9.4. 订阅勘误通知

您可以为 Satellite 用户配置电子邮件通知。用户收到适用和可安装的勘误表、内容视图提升通知或同步存储库后的通知。有关更多信息，请参阅管理 Red Hat Satellite 中的配置电子邮件通知首选项。

9.5. 仓库解析的限制

使用 Satellite 时，使用向内容视图增量更新来解决一些存储库依赖关系问题。但是，存储库级别的依赖项解析仍将存在问题。

当存储库更新可用时，如果现有存储库软件包中有旧版本，卫星会检索最新的软件包来解决依赖项。这可在安装软件包时产生其他依赖项解析问题。

示例情境

客户端上的存储库具有带有依赖项 **example_repository-libs-1.0** 的软件包 **example_repository-1.0**。存储库还包含另一个软件包 **example_tools-1.0**。

安全勘误会随软件包 **example_tools-1.1** 一起可用。**example_tools-1.1** 软件包需要 **example_repository-libs-1.1** 软件包作为依赖项。

在增量内容视图更新后，**example_tools-1.1**、**example_tools-1.0** 和 **example_repository-libs-1.1** 现在位于存储库中。存储库也具有软件包 **example_repository-1.0** 和 **example_repository-libs-1.0**。请注意，内容视图的增量更新没有添加软件包 **example_repository-1.1**。因为您可以使用 yum 安装所有这些软件包，所以不会检测到潜在的问题。但是，当客户端安装 **example_tools-1.1** 软件包时，会发生依赖项解析问题，因为无法安装 **example_repository-libs-1.0** 和 **example_repository-libs-1.1**。

当前没有解决此问题的方法。时间段越大，RPM 和所应用勘误表之间的次要 Y 版本越大，依赖项解析问题的可能性就越大。

9.6. 为勘误创建内容视图过滤器

您可以使用内容过滤器来限制勘误表。这种过滤器包括：

- **id** - 选择特定的勘误以允许生成的软件仓库。
- date **Range** - **定义日期范围**，并包含在该日期范围内发布的一组勘误。
- **类型** - 选择要包括程序错误修复、功能增强和安全更新等勘误表类型。

创建内容过滤器，以在特定日期之后排除勘误。这样可保证您在应用程序生命周期中的生产系统保持在一个特定时间点上保持最新状态。然后，您可以修改过滤器的开始日期，以在测试环境中新勘误测试新软件包在应用程序生命周期中的兼容性。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

先决条件

- 将创建包含所需勘误的存储库的内容视图。更多信息请参阅 第 6.1 节 “创建内容视图”。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Content Views**，然后选择您要用于应用勘误表的 Content View。
2. 选择 **Yum Content** > **Filters**，再单击 **New Filter**。
3. 在 **Name** 字段中输入 **Errata Filter**。
4. 从 **内容类型** 列表中，选择 Erratum - **Date 和 Type**。
5. 从 **Inclusion Type** 列表中，选择 **Exclude**。
6. 在 **Description** 字段中输入 **Exclude errata items from YYYY-MM-DD**。
7. 点击 **Save**。
8. 对于 **勘误类型**，选择您要排除的勘误表类型的复选框。例如，选择 **Enhancement** 和 **Bugfix** 复选框并清除 **Security** 复选框来排除特定日期之后的增强和程序错误修复勘误，但包括所有安全勘误。
9. 对于 "**日期类型**"，选择两个复选框之一：
   - **Issued On** 是勘误的发布日期。
   - **更新了 On**，了解勘误最后一次更新的日期。
10. 选择 **Start Date**，以排除在所选日期或之后的所有勘误。
11. 将 "**结束日期**" 字段留空。
12. 点击 **Save**。
13. 单击 **Publish New Version** 以发布生成的存储库。
14. 在 **Description** 字段中输入 **Adding errata filter**。
15. 点击 **Save**。
    当内容视图完成发布时，请注意 **Content** 列中会从初始存储库中报告软件包和勘误表的数量。这意味着，过滤器已成功排除了上一年中的所有非安全勘误。
16. 点 **Versions** 选项卡。
17. 单击 published version 右侧的 **Promote**。
18. 选择您要提升内容视图版本的环境。

19. 在 **Description** 字段中输入提升的描述。
20. 单击 **Promote Version** 以在所需环境中提升此内容视图版本。

CLI 过程

1. 为勘误创建过滤器：

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

2. 创建过滤规则以排除您要设置 的开始日期或之后 的所有勘误：

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

3. 发布内容视图：

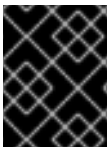
```
# hammer content-view publish --name "CV Name" \
--organization "Default Organization"
```

4. 将内容视图提升到 生命周期环境，以便包括的勘误可供该生命周期环境可用：

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

9.7. 在事件内容视图中添加勘误

如果勘误可用但不可安装，您可以创建一个增量内容视图版本，将勘误表添加到您的内容主机。例如，如果内容视图是 1.0 版本，则将变为 Content View version 1.1，且发布时，它将变为 Content View version 2.0。



重要

如果您的内容视图版本旧，则增量添加增强勘误时可能会遇到不兼容的问题。这是因为增强功能通常为存储库中的最新软件设计。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Errata**。
2. 在 **Errata** 列表中，点击您要应用的勘误表的名称。
3. 选择您要将勘误表应用到的内容主机，然后单击 **Apply to Hosts**。这会创建到内容视图的增量更新。

4. 如果要将勘误表应用到内容主机，请在 发布复选框后立即选中 Apply Errata to Content Hosts。
5. 点 Confirm 以应用勘误。

CLI 过程

1. 列出勘误及其对应的 ID：

```
# hammer erratum list
```

2. 列出不同的内容视图版本以及对应的 ID：

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

3. 将单个勘误应用于内容查看版本。您可以在以逗号分开的列表中添加更多 ID。

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

9.8. 将勘误应用到主机

使用这些流程审核并将勘误表应用到主机。

先决条件

- 将红帽卫星存储库与红帽提供的最新勘误表同步。更多信息请参阅 第 4.6 节 "同步软件仓库"。
- 将主机注册到 Satellite 服务器上的环境和内容视图。如需更多信息，请参阅 管理主机 中的 注册主机。
- 配置主机以进行远程执行。有关运行远程执行作业的更多信息，请参阅管理主机中的配置和设置远程作业。

注意

如果主机已配置为接收弃用的 Katello Agent 的内容更新，请改为迁移到远程执行。如需更多信息，请参阅管理主机中的 从 Katello Agent 迁移到远程执行。

将勘误应用到受管主机的步骤取决于其操作系统。

9.8.1. 将勘误应用到运行 Red Hat Enterprise Linux 9 的主机

使用这个流程查看并将勘误表应用到运行 Red Hat Enterprise Linux 9 的主机。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Hosts** > **Content Hosts** 并选择您要应用勘误的主机。
2. 导航到 **Errata** 选项卡，以查看勘误表列表。
3. 选择要应用的勘误，然后点 **Apply Selected**。在确认窗口中，点 **Apply**。

4. 在任务更新与所选勘误关联的所有软件包完成后，点击 **Details** 选项卡来查看更新的软件包。

CLI 过程

1. 列出主机的所有勘误：

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

2. 查找勘误所属的模块流：

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

3. 在主机上更新模块流：

```
# dnf update Module_Stream_Name
```

9.8.2. 将勘误应用到运行 Red Hat Enterprise Linux 8 的主机

使用这个步骤查看并应用勘误到运行 Red Hat Enterprise Linux 8 的主机。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Hosts** > **Content Hosts** 并选择您要应用勘误的主机。
2. 导航到 **Errata** 选项卡，以查看勘误表列表。
3. 选择要应用的勘误，然后点 **Apply Selected**。在确认窗口中，点 **Apply**。
4. 在任务更新与所选勘误关联的所有软件包完成后，点击 **Details** 选项卡来查看更新的软件包。

CLI 过程

1. 列出主机的所有勘误：

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

2. 查找勘误所属的模块流：

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

3. 在主机上更新模块流：

```
# dnf update Module_Stream_Name
```

9.8.3. 将勘误应用到运行 Red Hat Enterprise Linux 7 的主机

使用这个流程查看并将勘误表应用到运行 Red Hat Enterprise Linux 7 的主机。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Hosts** > **Content Hosts** 并选择您要应用勘误的主机。
2. 导航到 **Errata** 选项卡，以查看勘误表列表。
3. 选择要应用的勘误，然后点 **Apply Selected**。在确认窗口中，点 **Apply**。
4. 在任务更新与所选勘误关联的所有软件包完成后，点击 **Details** 选项卡来查看更新的软件包。

CLI 过程

1. 列出主机的所有勘误：

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

2. 将最新的勘误应用于主机。使用勘误 ID 识别要应用的勘误。
使用远程执行

```
# hammer job-invocation create \
--feature katello_errata_install \
--inputs errata=ERRATUM_ID1,ERRATUM_ID2 \
--search-query "name = client.example.com"
```

使用 **Katello Agent** （已弃用）

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

9.9. 将勘误应用到多个主机

使用这些流程来查看并应用勘误到多个 RHEL 7 主机。

先决条件

- 将红帽卫星存储库与红帽提供的最新勘误表同步。更多信息请参阅 第 4.6 节 “同步软件仓库”。
- 将主机注册到 Satellite 服务器上的环境和内容视图。如需更多信息，请参阅 管理主机 中的 注册主机。
- 配置主机以进行远程执行。有关运行远程执行作业的更多信息，请参阅管理主机中的配置和设置远程作业。

注意

如果主机已配置为接收弃用的 Katello Agent 的内容更新，请改为迁移到远程执行。如需更多信息，请参阅管理主机中的 从 Katello Agent 迁移到远程执行。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Errata**。
2. 点击您要应用的勘误名称。
3. 单击 **Content Hosts** 选项卡。
4. 选择您要应用勘误表的主机，然后单击 **应用至主机**。
5. 单击 **Confirm**。

CLI 过程

1. 列出所有可安装的勘误：

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

2. 将其中一个勘误应用到多个主机：
使用远程执行

```
# hammer job-invocation create \
--feature katello_errata_install \
--inputs errata=ERRATUM_ID \
--search-query "applicable_errata = ERRATUM_ID"
```

使用 **Katello Agent** （已弃用）

确定您要使用的勘误，并列出此勘误适用于的主机：

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

以下 Bash 脚本向有这个勘误的每个主机应用一个勘误：

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

这个命令使用 errata _ID 将所有主机标识为适用勘误，然后将勘误应用到每个主机。

3. 要查看成功应用了勘误，请在以下命令输出中找到对应的任务：

```
# hammer task list
```

4. 查看所选任务的状态：

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

9.10. 将勘误应用到主机集合

使用远程执行

```
# hammer job-invocation create \
--feature katello_errata_install \
--inputs errata=ERRATUM_ID1,ERRATUM_ID2,... \
--search-query "host_collection = HOST_COLLECTION_NAME"
```

使用 Katello Agent （已弃用）

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

第 10 章 管理容器镜像

通过 Satellite，您可以从各种来源导入容器镜像，并使用内容视图将它们分发到外部容器。

有关容器的详情，请参阅 Red Hat Enterprise Linux Atomic Host 7 中的容器入门。

10.1. 导入容器镜像

您可以从 Red Hat Registry 或其他镜像 registry 中导入容器镜像仓库。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

存储库发现过程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，再点击 **Repo Discovery**。
2. 从 **Repository Type** 列表，选择 **Container Images**。
3. 在 **Registry to Discover** 字段中输入要从中导入镜像的 registry URL。
4. 在 **Registry Username** 字段中输入与容器镜像 registry 的用户名对应的名称。
5. 在 **Registry Password** 字段中，输入与您输入的用户名对应的密码。
6. 在 **Registry Search Parameter** 字段中，输入您要用来过滤搜索的任何搜索条件，然后点 **Discover**。
7. 可选：要进一步重新定义 **Discovered Repository** 列表，在 **Filter** 字段中输入您要使用的其他搜索条件。
8. 在 **Discovered Repository** 列表中，选择您要导入的任何存储库，然后单击 **Create Selected**。
9. 可选：要将此容器存储库的下载策略更改为 按需，请参阅 第 4.10 节 "更改存储库的下载策略"。
10. 可选：如果希望创建一个产品，在 **Product** 列表中选择 **New Product**。
11. 在 **Name** 字段中输入产品名称。
12. 可选：在 **Repository Name** 和 **Repository Label** 列中，您可以编辑存储库名称和标签。
13. 单击 **Run Repository Creation**。
14. 当存储库创建完成后，您可以点击每个新存储库来查看更多信息。
15. 可选：要过滤您导入到存储库的内容，请点击存储库，然后导航到 **Limit Sync Tags**。点击以编辑，并添加您要限制与 Satellite 同步的内容的任何标签。
16. 在 Satellite web UI 中，进入 **Content** > **Products** 并选择您的产品的名称。
17. 选择新存储库，然后点 **Sync Now** 以启动同步过程。

手动创建存储库的步骤

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。点所需产品的名称。
2. 单击 **New repository**。

3. 从 Type 列表中，选择 docker。输入存储库的详细信息，然后单击保存。
4. 选择新存储库，然后单击 Sync Now。

后续步骤

- 要查看同步的进度，请导航到 Content > Sync Status 并展开存储库树。
- 同步完成后，您可以点击 Container Image Manifests 来列出可用的清单。在列表中，您还可以删除您不需要的任何清单。

CLI 过程

1. 创建自定义 **Red Hat Container Catalog** 产品：

```
# hammer product create \
--description "My_Description" \
--name "Red Hat Container Catalog" \
--organization "My_Organization" \
--sync-plan "My_Sync_Plan"
```

2. 为容器镜像创建存储库：

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

3. 同步存储库：

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

其他资源

- 有关手动创建产品和存储库的更多信息，请参阅 第 4 章 导入内容。

10.2. 管理容器名称模式

当您使用 Satellite 创建和管理容器时，容器通过 Satellite 生命周期环境的 Content View version 和不同阶段移动时，容器名称在每个阶段更改。例如，如果您将容器镜像与上游存储库的名称 **ssh** 同步，则当将其添加到 Satellite 产品和机构时，作为内容视图的一部分发布，容器镜像可以具有以下名称：**my_organization_production-custom_spin-my_product-custom_ssh**。这在要拉取容器镜像时可能会造成问题，因为容器 registry 只能包含一个容器名称的实例。为了避免 Satellite 命名约定出现问题，您可以设置注册表名称模式来覆盖默认名称，以确保以后使用容器名称是明确的。

限制

如果使用 registry 名称模式来管理容器命名规则，因为 registry 命名模式必须全局生成唯一名称，您可能会遇到命名冲突问题。例如：

- 如果您设置了 **repository.docker_upstream_name** registry 名称模式，则无法发布或将具有相同存储库名称的容器内容提升内容视图到 **Production** 生命周期。
- 如果设置 **lifecycle_environment.name** registry 名称模式，这可以防止创建具有相同名称的第二个容器存储库。

在为容器定义 registry 命名模式时，您必须谨慎。

流程

要使用 registry 名称模式管理容器命名，请完成以下步骤：

1. 在 Satellite Web UI 中，导航到 **Content** > **Lifecycle Environments**，然后创建生命周期环境，或选择要编辑的生命周期环境。
2. 在 **Container Image Registry** 区域中，点 **Registry Name Pattern** 区域右侧的编辑图标。
3. 使用变量和示例列表来决定您需要的 registry 名称模式。
4. 在 **Registry Name Pattern** 字段中输入您要使用的 registry 名称模式。例如，使用 **repository.docker_upstream_name** ：

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

5. 点击 **Save**。

10.3. 管理容器 REGISTRY 身份验证

您可以管理用于从 Satellite 访问容器镜像的身份验证设置。默认情况下，用户必须进行身份验证才能访问卫星中的容器镜像。

您可以指定您是否希望用户进行身份验证以在生命周期环境中访问 Satellite 中的容器镜像。例如，您可能希望允许用户从 **Production** 生命周期访问容器镜像，而无需进行身份验证，并将 **Development** 和 **QA** 环境的访问权限限制为经过身份验证的用户。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Lifecycle Environments**。
2. 选择您要管理验证的生命周期环境。
3. 要允许未经身份验证的访问此生命周期环境中的容器，请选择 **Unauthenticated Pull** 复选框。要限制未经身份验证的访问，请清除 **Unauthenticated Pull** 复选框。
4. 点击 **Save**。

10.4. 配置 PODMAN 和 DOCKER 以信任证书颁发机构

Podman 使用两个路径来定位 CA 文件，即 **/etc/containers/certs.d/** 和 **/etc/docker/certs.d/**。

将 root CA 文件复制到这些位置之一，使用由服务器主机名确定的确切路径，并命名文件 **ca.crt**

在以下示例中，根据您的用例，将 **hostname.example.com** 替换为 **satellite-server.example.com** 或 **capsule-server.example.com**。

- 您可能需要使用以下命令创建相关位置：

```
# mkdir -p /etc/containers/certs.d/hostname.example.com
```

或者

```
# mkdir -p /etc/docker/certs.d/hostname.example.com
```

- 对于 podman，请使用：

```
# cp rootCA.pem /etc/containers/certs.d/hostname.example.com/ca.crt
```

- 或者，如果您使用 Docker，将 root CA 文件复制到对应的 Docker 目录中：

```
# cp rootCA.pem /etc/docker/certs.d/hostname.example.com/ca.crt
```

在登录到 registry 时，不再需要使用 **--tls-verify=false** 选项：

```
$ podman login hostname.example.com

Username: admin
Password:
Login Succeeded!
```

10.5. 使用容器 REGISTRY

Podman 和 Docker 可用于从容器 registry 中获取内容。

Capsules 中的容器 registry

在带有内容的胶囊上，Container Gateway Capsule 插件充当容器 registry。它将从 Katello 中缓存身份验证信息，并将传入的请求代理到 Pulp。容器网关默认在带有内容的胶囊上可用。

流程

登录到容器 registry：

```
# podman login satellite.example.com
```

列出容器镜像：

```
# podman search satellite.example.com/
```

拉取容器镜像：

```
# podman pull satellite.example.com/my-image:<optional_tag>
```

第 11 章 管理 ISO 镜像

您可以使用 Satellite 存储 ISO 镜像，可以是从红帽内容交付网络或其他来源。您还可以上传其他文件，如虚拟机镜像，并将它们发布到存储库中。

11.1. 从红帽导入 ISO 镜像

Red Hat Content Delivery Network 为特定产品提供 ISO 镜像。导入此内容的步骤与为 RPM 内容启用存储库的步骤类似。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，进入 **Content** > **Red Hat Repositories**。
2. 在搜索字段中输入 **镜像名称**，例如： **Red Hat Enterprise Linux 7 Server(ISO)**。
3. 在 Available Repositories 窗口中，展开 **Red Hat Enterprise Linux 7 Server(ISO)**。
4. 对于 **x86_64 7.2** 条目，单击 **Enable** 图标，为镜像启用存储库。
5. 在 Satellite Web UI 中，进入 **Content** > **Products** 并点 **Red Hat Enterprise Linux Server**。
6. 单击 Red Hat Enterprise Linux Server 窗口的 **Repositories** 选项卡，然后点 **Red Hat Enterprise Linux 7 Server ISOs x86_64 7.2**。
7. 在 Red Hat Enterprise Linux 7 Server ISOs x86_64 7.2 窗口右上角，单击 **Select Action**，然后选择 **Sync Now**。

查看同步状态

- 在 Satellite Web UI 中，进入 **Content** > **Sync Status** 并展开 **Red Hat Enterprise Linux Server**。

CLI 过程

1. 找到用于 **文件** 仓库的 Red Hat Enterprise Linux Server 产品：

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

2. 为 Red Hat Enterprise Linux 7.2 Server ISO 启用 **文件** 仓库：

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

3. 在产品中找到存储库：

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

4. 同步产品中的存储库：

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

11.2. 导入独立 ISO 镜像和文件

使用这个步骤手动将 ISO 内容和其他文件导入到卫星服务器。要导入文件，您可以在 Satellite Web UI 或 Hammer CLI 中完成以下步骤。但是，如果要上传的文件大小大于 15 MB，则必须使用 Hammer CLI 将其上传到存储库。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**，然后在 Products 窗口中点击 **Create Product**。
2. 在 **Name** 字段中输入一个名称来标识产品。此名称填充 **Label** 字段。
3. 可选：在 **GPG Key** 字段中，为产品输入 GPG 密钥。
4. 可选：在 **Sync Plan** 列表中，为产品选择一个同步计划。
5. 可选：在 **Description** 字段中输入产品的描述。
6. 点击 **Save**。
7. 在 Products 窗口中，单击新产品，然后单击 **Create Repository**。
8. 在 **Name** 字段中输入存储库的名称。这会自动填充 **Label** 字段。
9. 从 **Type** 列表，选择 **文件**。
10. 在 **Upstream URL** 字段中，输入要用作源的 registry 的 URL。在 **Upstream Username** 和 **Upstream Password** 字段中添加对应的用户名和密码。
11. 点击 **Save**。
12. 选择新存储库。
13. 导航到 **Upload File**，再单击 **Browse**。
14. 选择 **.iso** 文件并单击 **Upload**。

CLI 过程

1. 创建自定义产品：

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

2. *创建存储库：*

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

3. *将 ISO 文件上传到存储库：*

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

第 12 章 管理 ANSIBLE 内容

您可以将 Ansible 集合从多个源导入到 Satellite 服务器。

如需有关 Satellite 中的 Ansible 集成的更多信息，请参阅在 Red Hat Satellite 中使用 Ansible 集成管理配置。

12.1. 同步 ANSIBLE 集合

在 Satellite 上，您可以从 Private Automation Hub、**console.redhat.com** 和其他 Satellite 实例同步 Ansible Collections。在同步后，Ansible Collections 将会出现在 Satellite web UI 菜单的 **Content** 下作为一个新的仓库类型。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 选择所需的产品名称。
3. 在 **Products** 窗口中，选择要为其创建存储库的产品名称。
4. 单击 **Repositories** 选项卡，然后单击 **New Repository**。
5. 在 **Name** 字段中输入存储库的名称。
   Label 字段会根据名称自动填充。
6. 从 **Type** 列表中，选择 **ansible collection**。
7. 在 **Upstream URL** 字段中，输入上游集合存储库的 URL。
   URL 可以是任何 Ansible Galaxy 端点。例如：**https://console.redhat.com/api/automation-hub/**。
8. 可选：在 requirements **.yml** 字段中，您可以指定您要从端点同步的集合列表及其版本。
   如果没有指定集合列表，则会同步来自端点的所有内容。

   ```
   ---
   collections:
   - name: my_namespace.my_collection
     version: 1.2.3
   ```

   如需更多信息，请参阅 Galaxy 用户指南中的安装带有要求文件的多个集合。
9. 身份验证。
   a. 要从 **Private Automation Hub** 同步 Satellite，请在 **Auth Token** 字段中输入您的令牌。
      如需更多信息，请参阅连接到 Hub 中的连接私有 Automation Hub。
   b. 要从 **console.redhat.com** 同步 Satellite，请在 **Auth Token** 字段中输入您的令牌，然后在 **Auth URL** 字段中输入您的 SSO URL。
      如需更多信息，请参阅开始使用自动化中心。
   c. 要从 Satellite 同步 Satellite，请将这两个身份验证字段留空。
10. 点击 **Save**。

11. 导航到 Ansible Collections 存储库。
12. 在 **Select Action** 菜单中，选择 **Sync Now**。

第 13 章 管理自定义文件类型内容

在 Satellite 中，您可能需要管理和分发 SSH 密钥和源代码文件或更大文件（如虚拟机镜像和 ISO 文件）。为实现这一目标，红帽卫星中的自定义产品包括用于自定义文件类型的存储库。这提供了一种通用方法，用于将任意文件纳入一个产品中。

您可以上传文件到存储库，并从上游卫星服务器同步文件。当您将文件添加到自定义文件类型存储库时，您可以使用普通的 Satellite 管理功能，比如将特定版本添加到内容视图，以提供版本控制并在各种胶囊服务器上提供文件存储库。客户端必须使用 **curl -O** 通过 HTTP 或 HTTPS 下载文件。

您可以在卫星服务器中只在自定义产品中创建文件类型存储库，但具有如何创建存储库源的灵活性。您可以在卫星服务器上的目录中创建独立的存储库源，也可以创建远程 HTTP 服务器，然后将该目录的内容同步到卫星中。如果您有多个要添加到 Satellite 仓库的文件，此方法很有用。

13.1. 为自定义文件类型存储库创建本地源

您可以从文件目录（使用 **Pulp Manifest** 安装卫星的基础系统上）创建自定义文件类型存储库源。然后，您可以将文件同步到存储库中，并管理自定义文件内容，如任何其他内容类型。

使用这个步骤在安装 Satellite 的基础系统的目录中配置存储库。要在远程服务器的目录中创建文件类型存储库，请参考 第 13.2 节 "为自定义文件类型存储库创建远程源"。

流程

1. 确定启用了 Utils 存储库：

```
# subscription-manager repos \
--enable=rhel-8-for-x86_64-appstream-rpms \
--enable=rhel-8-for-x86_64-baseos-rpms \
--enable=satellite-utils-6.12-for-rhel-8-x86_64-rpms
```

2. 启用 satellite-utils 模块：

```
# dnf module enable satellite-utils
```

3. 安装 Pulp 清单软件包：

```
# satellite-maintain packages install python39-pulp_manifest
```

请注意，这个命令会停止 Satellite 服务并重新运行 **satellite-installer**。另外，为了避免停止该服务造成的停机，您可以使用以下内容：

```
# satellite-maintain packages unlock
# dnf install python39-pulp_manifest
# satellite-maintain packages lock
```

4. 创建您要用作文件类型存储库的目录，例如：

```
# mkdir -p /var/lib/pulp/local_repos/my_file_repo
```

5. 将父文件夹添加到允许的导入路径中：

```
# satellite-installer --foreman-proxy-content-pulpcore-additional-import-paths
/var/lib/pulp/local_repos
```

6. 将文件添加到目录中或创建测试文件：

```
# touch /var/lib/pulp/local_repos/my_file_repo/test.txt
```

7. 运行 Pulp Manifest 命令创建清单：

```
# pulp-manifest /var/lib/pulp/local_repos/my_file_repo
```

8. 验证已创建了清单：

```
# ls /var/lib/pulp/local_repos/my_file_repo
PULP_MANIFEST test.txt
```

现在，您可以将本地源导入为自定义文件类型存储库。使用 **file://** URL 方案和目录名称指定 **上游 URL**，如 **file:///my_file_repo**。更多信息请参阅 第 13.3 节 "创建自定义文件类型存储库"。

如果更新目录的内容，请重新运行 Pulp 清单并在卫星中同步存储库。更多信息请参阅 第 4.6 节 "同步软件仓库"。

13.2. 为自定义文件类型存储库创建远程源

您可以使用 **Pulp Manifest** 从对卫星服务器外部的文件创建自定义类型存储库源。然后，您可以通过 HTTP 或 HTTPS 将文件同步到卫星服务器上的存储库，并管理自定义文件内容，如任何其他内容类型。

使用这个流程在远程服务器上的目录中配置存储库。要在安装 Satellite Server 的基本系统中创建文件类型存储库，请参考 第 13.1 节 "为自定义文件类型存储库创建本地源"。

先决条件

- 您有运行 Red Hat Enterprise Linux 8 的服务器注册到 Satellite 或 Red Hat CDN。
- 您的服务器有权使用 Red Hat Enterprise Linux 服务器和 Red Hat Satellite Utils 软件仓库。
- 已安装 HTTP 服务器。有关配置 web 服务器的更多信息，请参阅部署不同类型的服务器中的设置 Apache HTTP web 服务器。

流程

1. 在您的服务器中，确保启用了正确的软件仓库：

```
# subscription-manager repos \
--enable=rhel-8-for-x86_64-appstream-rpms \
--enable=rhel-8-for-x86_64-baseos-rpms \
--enable=satellite-utils-6.12-for-rhel-8-x86_64-rpms
```

2. 启用 satellite-utils 模块：

```
# dnf module enable satellite-utils
```

3. 安装 Pulp 清单软件包：

```
# dnf install python39-pulp_manifest
```

4. 创建您要用作 HTTP 服务器公共文件夹中的文件类型存储库的目录：

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

5. 将文件添加到目录中或创建测试文件：

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

6. 运行 Pulp Manifest 命令创建清单：

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

7. 验证已创建了清单：

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

现在，您可以将远程源导入为自定义文件类型存储库。使用到目录的路径指定 **上游 URL**，如 **http://satellite.example.com/my_file_repo**。更多信息请参阅 第 13.3 节 "创建自定义文件类型存储库"。

如果更新目录的内容，请重新运行 Pulp 清单并在卫星中同步存储库。更多信息请参阅 第 4.6 节 "同步软件仓库"。

13.3. 创建自定义文件类型存储库

创建自定义文件类型存储库的步骤与创建任何自定义内容的步骤相同，但创建存储库时除外，您可以选择 **文件类型**。您必须创建一个产品，然后添加自定义存储库。

要使用 CLI 而不是 Satellite Web UI，请参阅 CLI 过程。

流程

1. 在 Satellite Web UI 中，导航到 **Content > Products**。
2. 选择您要为其创建存储库的产品。
3. 在 **Repositories** 选项卡上，单击 **New Repository**。
4. 在 **Name** 字段中输入存储库的名称。Satellite 根据名称自动完成 **Label** 字段。
5. 可选：在 **Description** 字段中输入存储库的描述。
6. 从 **Type** 列表，选择 **file** 作为存储库类型。
7. 可选：在 **Upstream URL** 字段中输入要用作源的上游存储库的 URL。如果没有输入上游 URL，可以手动上传软件包。如需更多信息，请参阅 第 13.4 节 "上传文件到自定义文件类型存储库"。
8. 如果要验证上游存储库的 SSL 证书是否由可信 CA 签名，请选择 **Verify SSL** 复选框。
9. 可选：在 **Upstream Username** 字段中输入上游存储库的用户名（如果需要）。如果存储库不需要身份验证，请清除此字段。

10. 可选：在 **Upstream Password** 字段中输入上游存储库对应的密码。如果存储库不需要身份验证，请清除此字段。
11. 可选：在 **Upstream Authentication Token** 字段中，提供上游存储库用户的令牌以进行身份验证。如果存储库不需要身份验证，请将此字段留空。
12. 从镜像 **策略** 列表中，选择内容同步卫星服务器执行的类型。更多信息请参阅 第 4.11 节 "镜像策略概述"。
13. 可选：在 **HTTP Proxy Policy** 字段中，选择或取消选择使用 HTTP 代理。默认情况下，它使用 **Global Default** HTTP 代理。
14. 可选：您可以清除 **未保护** 复选框，以需要一个订阅授权证书来访问这个存储库。默认情况下，存储库通过 HTTP 发布。
15. 可选： 在 **SSL CA Cert** 字段中，选择存储库的 SSL CA 证书。
16. 可选： 在 **SSL Client Cert** 字段中，选择存储库的 SSL Client Certificate。
17. 可选：在 **SSL Client Key** 字段中，为存储库选择 SSL Client Key。
18. 单击 **Save** 以创建存储库。

CLI 过程

1. 创建自定义产品：

```
# hammer product create \
--description "My_Files" \
--name "My_File_Product" \
--organization "My_Organization" \
--sync-plan "My_Sync_Plan"
```

表 13.1. hammer 产品创建命令 的可选参数

选项	描述
--gpg-key-id *gpg_key_id*	GPG 密钥数字识别符
--sync-plan-id *sync_plan_id*	同步计划数字标识符
--sync-plan *sync_plan_name*	要搜索的同步计划名称

2. 创建 **文件类型** 存储库：

```
# hammer repository create \
--content-type "file" \
--name "My_Files" \
--organization "My_Organization" \
--product "My_File_Product"
```

表 13.2. hammer 仓库的可选参数 创建命令

选项	描述
--checksum-type *sha_version*	软件仓库校验和，支持当前 'sha1' 和 'sha256'
--download-policy *policy_name*	下载 yum 仓库的策略（"immediate"或"on_demand"）。
--gpg-key-id *gpg_key_id*	GPG 密钥数字识别符
--gpg-key *gpg_key_name*	要搜索的密钥名称
--mirror-on-sync *boolean*	同步时，必须把这个存储库从源和过时的 RPM 进行镜像(mirror)？设置为 **true** 或 **false**,**yes** 或 **no**,**1** 或 **0**。
--publish-via-http *boolean*	是否必须使用 HTTP 发布？设置为 **true** 或 **false**,**yes** 或 **no**,**1** 或 **0**。
--upstream-password *repository_password*	上游存储库用户的密码
--upstream-username *repository_username*	身份验证需要上游存储库用户
--url *source_repo_url*	源存储库的 URL
--verify-ssl-on-sync *boolean*	必须为 Katello 验证上游 URL 的 SSL 证书是否由可信 CA 签名？设置为 **true** 或 **false**,**yes** 或 **no**,**1** 或 **0**。

13.4. 上传文件到自定义文件类型存储库

使用此流程将文件上传到自定义文件类型存储库。

流程

1. *在 Satellite Web UI 中，导航到* **Content** > **Products**。
2. *按名称选择自定义产品。*
3. *按名称选择文件类型存储库。*
4. *点击* **Browse** *搜索并选择您要上传的文件。*
5. *单击* **Upload**，*将所选文件上传到卫星服务器。*
6. *访问发布库的 URL，以查看该文件。*

CLI 过程

```
# hammer repository upload-content \
--id repo_ID \
```

```
--organization "My_Organization" \
--path example_file
```

--path 选项可以指明文件、文件目录或文件的通配表达式。通配必须通过单引号或双引号进行转义。

13.5. 将文件下载到自定义文件类型存储库的主机

您可以使用 **curl -O** 通过 HTTPS 将文件下载到客户端，如果选择了存储库的 **Unprotected** 选项，则可以选择通过 HTTP 下载到客户端。

先决条件

- 您有一个自定义文件类型存储库。更多信息请参阅 第 13.3 节 "创建自定义文件类型存储库"。
- 您知道要从文件类型存储库下载至客户端的文件名。
- 要使用 HTTPS，您需要在客户端中需要以下证书：
  1. **katello-server-ca.crt** 有关更多信息，请参阅在 管理红帽卫星 中 导入 Katello Root CA 证书。
  2. 机构调试证书.如需更多信息，请参阅在 管理 Red Hat Satellite 中创建机构 Debug 证书。

流程

1. 在 Satellite Web UI 中，导航到 **Content** > **Products**。
2. 按名称选择自定义产品。
3. 按名称选择文件类型存储库。
4. 确保选中 **Unprotected** 复选框，以访问通过 HTTP 发布的存储库。
5. 复制发布存储库的 URL。

CLI 过程

1. 列出文件类型存储库。

```
# hammer repository list --content-type file
---|------------|------------------|--------------|----
ID | NAME       | PRODUCT          | CONTENT TYPE | URL
---|------------|------------------|--------------|----
7  | My_Files   | My_File_Product  | file         |
---|------------|------------------|--------------|----
```

2. 显示存储库信息。

```
# hammer repository info \
--name "My_Files" \
--organization-id My_Organization_ID \
--product "My_File_Product"
```

如果启用了 **未保护**，输出类似如下：

```
Publish Via HTTP: yes
Published At:
https://satellite.example.com/pulp/content/My_Organization_Label/Library/custom/My_File_Product_Label/My_Files_Label/
```

如果没有启用 未受保护，输出类似如下：

```
Publish Via HTTP: no
Published At:
https://satellite.example.com/pulp/content/My_Organization_Label/Library/custom/My_File_Product_Label/My_Files_Label/
```

3. 在客户端中，为 HTTP 或 HTTPS 输入适当的格式的命令：
对于 HTTP：

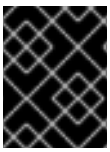
```
# curl -O
http://satellite.example.com/pulp/content/My_Organization_Label/Library/custom/My_File_Product_Label/My_Files_Label/my_file
```

对于 HTTPS：

```
# curl -O --cert ./_My-Organization-key-cert.pem_ --cacert katello-server-ca.crt
https://satellite.example.com/pulp/content/My_Organization_Label/Library/custom/My_File_Product_Label/My_Files_Label/my_file
```

附录 A. 将 NFS 共享用于内容存储

您的环境需要足够的硬盘空间来满足内容存储。在某些情况下，使用 NFS 共享来存储此内容非常有用。本附录演示了如何将 NFS 共享挂载到卫星服务器的内容管理组件上。



重要

为 **/var/lib/pulp** 文件系统使用高带宽、低延迟存储。红帽卫星具有许多 I/O 密集型操作，因此，高延迟和低带宽存储可能会遇到性能下降的问题。

流程

1. 创建 NFS 共享。这个示例在 **nfs.example.com:/Satellite/pulp** 中的一个共享。确保此共享为卫星服务器及其 **apache** 用户提供适当的权限。
2. 在 Satellite 服务器中停止 Satellite 服务：

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

3. 确定安装 **nfs-utils** 软件包：

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

4. 您需要将 **/var/lib/pulp** 的现有内容复制到 NFS 共享。首先，将 NFS 共享挂载到临时位置：

```
# mkdir /mnt/temp
# mount -o rw nfs.example.com:/Satellite/pulp /mnt/temp
```

将 **/var/lib/pulp** 的现有内容复制到临时位置：

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

5. 为共享中的所有文件设置权限，以使用 **pulp** 用户。
6. 卸载临时存储位置：

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

7. 删除 **/var/lib/pulp** 的现有内容：

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

8. 编辑 **/etc/fstab** 文件并添加以下行：

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

这使得挂载在系统重启后会保留。确保包含 SELinux 上下文。

9. 启用挂载：

```
# mount -a
```

10. 确认 NFS 共享挂载至 **var/lib/pulp** ：

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

另外，确认 **var/lib/pulp** 上的挂载中是否存在现有内容：

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

11. 在 Satellite 服务器中启动 Satellite 服务：

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

Satellite 服务器现在使用 NFS 共享来存储内容。运行内容同步以确保 NFS 共享按预期工作。更多信息请参阅 第 4.6 节 “同步软件仓库”。

附录 B. 导入 KICKSTART 存储库

Kickstart 软件仓库不是由内容 ISO 镜像提供。要在断开连接的 Satellite 中使用 Kickstart 软件仓库，您必须下载您要使用的 Red Hat Enterprise Linux 版本的二进制 DVD ISO 文件，并将 Kickstart 文件复制到 Satellite。

要为 Red Hat Enterprise Linux 9 导入 Kickstart 软件仓库，完成 第 B.1 节 "为 Red Hat Enterprise Linux 9 导入 Kickstart 存储库"。

要为 Red Hat Enterprise Linux 8 导入 Kickstart 软件仓库，完成 第 B.2 节 "为 Red Hat Enterprise Linux 8 导入 Kickstart 存储库"。

为 Red Hat Enterprise Linux 7 导入 Kickstart 软件仓库，完成 第 B.3 节 "为 Red Hat Enterprise Linux 7 导入 Kickstart 存储库"。

B.1. 为 RED HAT ENTERPRISE LINUX 9 导入 KICKSTART 存储库

使用这个流程为 Red Hat Enterprise Linux 9 导入 Kickstart 软件仓库。

流程

1. 访问 access.redhat.com/downloads 并登录红帽客户门户网站。
2. 点击 **Red Hat Enterprise Linux**。
3. 从列表中选择一个产品变体和产品版本。例如，产品变体的 **Red Hat Enterprise Linux for x86_64** 和产品版本 **9.0**。
4. 找到完整安装镜像，例如 **Red Hat Enterprise Linux 9.0 Binary DVD**，然后点击 **Download Now**。请注意，您无法使用最小 ISO 置备主机。
5. 下载完成后，将 ISO 镜像复制到卫星服务器。
6. 在卫星服务器上，创建一个挂载点，并在该位置临时挂载 ISO 镜像：

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

将 rhel-binary-dvd.iso 替换为您的 ISO 镜像的名称。

7. 为 Red Hat Enterprise Linux 9 AppStream 和 BaseOS Kickstart 软件仓库创建目录：

```
# mkdir --parents /var/www/html/pub/satellite-
import/content/dist/rhel9/9.0/x86_64/appstream/kickstart
# mkdir --parents /var/www/html/pub/satellite-
import/content/dist/rhel9/9.0/x86_64/baseos/kickstart
```

8. 从 ISO 镜像复制 **kickstart** 文件：

```
# cp -a /mnt/iso/AppStream/* /var/www/html/pub/satellite-
import/content/dist/rhel9/9.0/x86_64/appstream/kickstart
# cp -a /mnt/iso/BaseOS/* /mnt/iso/images/ /var/www/html/pub/satellite-
import/content/dist/rhel9/9.0/x86_64/baseos/kickstart
```

请注意，对于 BaseOS，还必须复制 **/mnt/iso/images/** 目录的内容。

9. 在列出文件中添加以下条目：
   至 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/appstream/listing** 文件，使用新行附加 **kickstart**。

   在 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/baseos/listing** 文件中，使用新行附加 **kickstart**。

   到 **/var/www/html/pub/satellite-import/content/dist/rhel8/listing** 文件，请使用新行附加版本号。例如，对于 Red Hat Enterprise Linux 9.0 二进制 ISO，请附加 **9.0**。

10. 从 ISO 镜像复制 **.treeinfo** 文件：

    ```
    # cp /mnt/iso/.treeinfo /var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/appstream/kickstart/treeinfo
    # cp /mnt/iso/.treeinfo /var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/baseos/kickstart/treeinfo
    ```

11. 打开 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/baseos/kickstart/treeinfo** 文件进行编辑。

12. 在 **[general]** 部分进行以下更改：

    - 将 **packagedir = AppStream/Packages** 改为 **packagedir = Packages**
    - 将 **repository = AppStream** 更改为 **repository = .**
    - 将 **变体 = AppStream** 更改为 **变体 = BaseOS**
    - 将 **变体 = AppStream,BaseOS** 改为 **variants = BaseOS**

13. 在 **[tree]** 部分中，将 **变体 = AppStream,BaseOS** 更改为 **variants = BaseOS**。

14. 在 **[variant-BaseOS]** 部分，进行以下更改：

    - 将 **软件包 = BaseOS/Packages** to **packages = Packages**
    - 将 **repository = BaseOS** 更改为 **repository =**。

15. 删除 **[media]** 和 **[variant-AppStream]** 部分。

16. 保存并关闭该文件。

17. 验证 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/baseos/kickstart/treeinfo** 文件具有以下格式：

    ```
    [checksums]
    images/efiboot.img =
    sha256:c01c18acc6778d6e66c8d0872bac59bfd7219ccf3cfa70a5c605c0fb37f33a83
    images/install.img =
    sha256:ddd08e5a5d92edee150f91ff4f12f39253eae72ff496465cf1b2766fe4a4df49
    images/pxeboot/initrd.img =
    sha256:a09a8ec89d485d71ed1bdad83584d6d816e67448221172d9aad97886cd70adca
    images/pxeboot/vmlinuz =
    sha256:6e523d7c3266e26c695923ab12b2873b16b0c61fb2e48ade608ad8998821584b

    [general]
    ; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
    ```

```
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 9.0.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = BaseOS
variants = BaseOS
version = 9.0.0

[header]
type = productmd.treeinfo
version = 1.2

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

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

[release]
name = Red Hat Enterprise Linux
short = RHEL
version = 9.0.0

[stage2]
mainimage = images/install.img

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

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

18. 打开 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/appstream/kickstart/treeinfo** 文件进行编辑。

19. 在 **[general]** 部分进行以下更改：

    - 将 **packagedir = AppStream/Packages** 改为 **packagedir = Packages**
    - 将 **repository = AppStream** 更改为 **repository = .**

- 将 **变体 = AppStream,BaseOS** 改为 **variants = AppStream**

20. 在 **[tree]** 部分中，将 **变体 = AppStream,BaseOS** 改为 **variants = AppStream**
21. 在 **[variant-AppStream]** 部分进行以下更改：
    - 更改 **软件包 = AppStream/Packages** to **packages = Packages**
    - 将 **repository = AppStream** 更改为 **repository = .**
22. 从文件中删除以下部分： **[checksums]**, **[images-x86_64]**, **[images-xen]**, **[media]**, **[stage2]**, **[variant-BaseOS]**.
23. 保存并关闭该文件。
24. 验证 **/var/www/html/pub/satellite-import/content/dist/rhel9/9.0/x86_64/appstream/kickstart/treeinfo** 文件具有以下格式：

```
[general]
; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 9.0.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = AppStream
variants = AppStream
version = 9.0.0

[header]
type = productmd.treeinfo
version = 1.2

[release]
name = Red Hat Enterprise Linux
short = RHEL
version = 9.0.0

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

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

25. 如果您没有计划使用挂载的二进制 DVD ISO 镜像，请卸载并删除目录：

```
-
```

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

26. 在卫星 Web UI 中，启用 Kickstart 存储库。

B.2. 为 RED HAT ENTERPRISE LINUX 8 导入 KICKSTART 存储库

使用这个流程为 Red Hat Enterprise Linux 8 导入 Kickstart 软件仓库。

流程

1. 访问 access.redhat.com/downloads 并登录红帽客户门户网站。
2. 点击 **Red Hat Enterprise Linux**。
3. 从列表中选择一个产品变体和产品版本。例如，产品变体 **Red Hat Enterprise Linux for x86_64** 和 product version **8.1**。
4. 找到完整安装镜像，例如 **Red Hat Enterprise Linux 8.1 Binary DVD**，并点 **Download Now**。
5. 下载完成后，将 ISO 镜像复制到卫星服务器。
6. 在卫星服务器上，创建一个挂载点，并在该位置临时挂载 ISO 镜像：

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

将 rhel-binary-dvd.iso 替换为您的 ISO 镜像的名称。

7. Create directories for Red Hat Enterprise Linux 8 AppStream and BaseOS Kickstart repositories:

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

8. 从 ISO 镜像复制 **kickstart** 文件：

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

请注意，对于 BaseOS，还必须复制 **/mnt/iso/images/** 目录的内容。

9. 在列出文件中添加以下条目：
到 **/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/appstream/listing** 文件，使用新行附加 **kickstart**。

到 **/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/baseos/listing** 文件，使用新行附加 **kickstart**。

到 **/var/www/html/pub/satellite-import/content/dist/rhel8/listing** 文件，请使用新行附加版本号。例如，对于 Red Hat Enterprise Linux 8.1 二进制 ISO，请附加 **8.1**。

10. *从 ISO 镜像复制* ***.treeinfo*** *文件：*

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

11. *打开* ***/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart/treeinfo*** *文件进行编辑。*

12. *在* ***[general]*** *部分进行以下更改：*

    - *将* ***packagedir = AppStream/Packages*** *改为* ***packagedir = Packages***
    - *将* ***repository = AppStream*** *更改为* ***repository = .***
    - *将* ***变体 = AppStream*** *更改为* ***变体 = BaseOS***
    - *将* ***变体 = AppStream,BaseOS*** *改为* ***variants = BaseOS***

13. *在* ***[tree]*** *部分中，将* ***变体 = AppStream,BaseOS*** *更改为* ***variants = BaseOS。***

14. *在* ***[variant-BaseOS]*** *部分，进行以下更改：*

    - *将* ***软件包 = BaseOS/Packages*** *to* ***packages = Packages***
    - *将* ***repository = BaseOS*** *更改为* ***repository =。***

15. *删除* ***[media]*** *和* ***[variant-AppStream]*** *部分。*

16. *保存并关闭该文件。*

17. *验证* ***/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/baseos/kickstart/treeinfo*** *文件具有以下格式：*

```
[checksums]
images/efiboot.img =
sha256:c01c18acc6778d6e66c8d0872bac59bfd7219ccf3cfa70a5c605c0fb37f33a83
images/install.img =
sha256:ddd08e5a5d92edee150f91ff4f12f39253eae72ff496465cf1b2766fe4a4df49
images/pxeboot/initrd.img =
sha256:a09a8ec89d485d71ed1bdad83584d6d816e67448221172d9aad97886cd70adca
images/pxeboot/vmlinuz =
sha256:6e523d7c3266e26c695923ab12b2873b16b0c61fb2e48ade608ad8998821584b

[general]
; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 8.1.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = BaseOS
variants = BaseOS
```

```
version = 8.1.0

[header]
type = productmd.treeinfo
version = 1.2

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

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

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

[stage2]
mainimage = images/install.img

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

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

18. 打开 **/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart/treeinfo** 文件进行编辑。
19. 在 **[general]** 部分进行以下更改：
    - 将 **packagedir = AppStream/Packages** 改为 **packagedir = Packages**
    - 将 **repository = AppStream** 更改为 **repository = .**
    - 将 **变体 = AppStream,BaseOS** 改为 **variants = AppStream**
20. 在 **[tree]** 部分中，将 **变体 = AppStream,BaseOS** 改为 **variants = AppStream**
21. 在 **[variant-AppStream]** 部分进行以下更改：
    - 更改 **软件包 = AppStream/Packages** to **packages = Packages**
    - 将 **repository = AppStream** 更改为 **repository = .**

22. 从文件中删除以下部分：**[checksums]**, **[images-x86_64]**, **[images-xen]**, **[media]**, **[stage2]**, **[variant-BaseOS]**.

23. 保存并关闭该文件。

24. 验证 **/var/www/html/pub/satellite-import/content/dist/rhel8/8.1/x86_64/appstream/kickstart/treeinfo** 文件具有以下格式：

```
[general]
; WARNING.0 = This section provides compatibility with pre-productmd treeinfos.
; WARNING.1 = Read productmd documentation for details about new format.
arch = x86_64
family = Red Hat Enterprise Linux
name = Red Hat Enterprise Linux 8.1.0
packagedir = Packages
platforms = x86_64,xen
repository = .
timestamp = 1571146127
variant = AppStream
variants = AppStream
version = 8.1.0

[header]
type = productmd.treeinfo
version = 1.2

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

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

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

25. 如果您没有计划使用挂载的二进制 DVD ISO 镜像，请卸载并删除目录：

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

26. 在卫星 Web UI 中，启用 Kickstart 存储库。

B.3. 为 RED HAT ENTERPRISE LINUX 7 导入 KICKSTART 存储库

使用这个流程为 Red Hat Enterprise Linux 7 导入 Kickstart 软件仓库。

流程

1. 访问 access.redhat.com/downloads 并登录红帽客户门户网站。
2. 点击 **Red Hat Enterprise Linux**。
3. 单击 **Switch to version 7，然后在产品变体** 列表中出现。
4. 从列表中选择一个产品变体和产品版本。例如，产品变体 **Red Hat Enterprise Linux for x86_64** 和产品版本 **7.9**。
5. 找到完整安装镜像，例如 **Red Hat Enterprise Linux 7.9 Binary DVD**，然后点 **Download Now**。
6. 下载完成后，将 ISO 镜像复制到卫星服务器。
7. 在卫星服务器上，创建一个挂载点，并在该位置临时挂载 ISO 镜像：

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

   将 rhel-binary-dvd.iso 替换为您的 ISO 镜像的名称。

8. 创建 Kickstart 目录：

```
# mkdir --parents /var/www/html/pub/satellite-
import/content/dist/rhel/server/7/7.9/x86_64/kickstart/
```

9. 从 ISO 镜像复制 **kickstart** 文件：

```
# cp -a /mnt/iso/* /var/www/html/pub/satellite-
import/content/dist/rhel/server/7/7.9/x86_64/kickstart/
```

10. 在列出文件中添加以下条目：
    至 **/var/www/html/pub/satellite-import/content/dist/rhel/server/7/listing** 文件，请使用新行附加版本号。例如，对于 Red Hat Enterprise Linux 7.9 ISO，请附加 **7.9**。

    在 **/var/www/html/pub/satellite-import/content/dist/rhel/server/7/7.9/listing** 文件中，使用新行附加构架。例如，**x86_64**。

    在 **/var/www/html/pub/satellite-import/content/dist/rhel/server/7/7.9/x86_64/listing** 文件中，使用新行附加 **kickstart**。

11. 从 ISO 镜像复制 **.treeinfo** 文件：

```
# cp /mnt/iso/.treeinfo /var/www/html/pub/satellite-
import/content/dist/rhel/server/7/7.9/x86_64/kickstart/treeinfo
```

12. 如果您没有计划使用挂载的二进制 DVD ISO 镜像，请卸载并删除目录：

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

13. 在卫星 Web UI 中，启用 Kickstart 存储库。