Red Hat Software Collections 3

使用 Red Hat Software Collections 容器镜像

Red Hat Software Collections 3.8 容器镜像的基本使用说明

Last Updated: 2024-03-06

Red Hat Software Collections 3 使用 Red Hat Software Collections 容器镜像

Red Hat Software Collections 3.8 容器镜像的基本使用说明

Lenka Špačková
lspackova@redhat.com

Olga Tikhomirova

Robert Krátký

Vladimír Slávik

法律通告

摘要

作为 Red Hat Software Collections 产品的一部分，红帽提供了很多容器镜像，它们基于对应的 Software Collections。Red Hat Software Collections 容器镜像包括应用程序、Web 服务器和数据库镜像。本文档提供了有关获取、配置和使用与 Red Hat Software Collections 分发的容器镜像的信息。

目录

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。有关更多详情，请参阅我们的首席技术官 Chris Wright 提供的消息。

第 1 章 RED HAT SOFTWARE COLLECTIONS 容器镜像

Red Hat Software Collections 容器镜像基于对应的集合和 rhel7 或 ubi7 基础镜像。有关通用基础镜像的更多信息，请参阅 通用基础镜像(UBI)：镜像、存储库、软件包和源代码。

Red Hat Software Collections 容器镜像包括应用程序、守护进程和数据库镜像。运行 Red Hat Software Collections 容器镜像支持：

- Red Hat Enterprise Linux 7 Server
- Red Hat Enterprise Linux 7 Atomic Host
- Red Hat Enterprise Linux 8

有关作为 Red Hat Enterprise Linux 7 Software Collections 的 Software Collections 可用的组件的信息，请参阅 Red Hat Software Collections 和 Red Hat Developer Toolset 文档。

Red Hat Software Collections 容器镜像在表中详述：

- 第 3 章 *基于 Red Hat Software Collections 3.8 的容器镜像*
- 第 4 章 *容器镜像基于 Red Hat Software Collections 3.7*
- 第 5 章 *基于 Red Hat Software Collections 3.6 的容器镜像*
- 第 6 章 *基于 Red Hat Software Collections 3.5 的容器镜像*
- 第 7 章 *容器镜像基于红帽软件集合 3.4*
- 第 8 章 *基于 Red Hat Software Collections 3.3 的容器镜像*
- 第 9 章 *容器镜像基于 Red Hat Software Collections 3.2*
- 第 10 章 *基于 Red Hat Software Collections 3.1 的容器镜像*
- 第 11 章 *基于 Red Hat Software Collections 3.0 的容器镜像*

您还可以在 红帽生态系统目录 中搜索可用的容器镜像。

重要

仅支持红帽提供的每个容器镜像的最新版本。

注意

在将 SELinux 用于控制容器内的进程时，请确保挂载到容器中的卷的所有内容均可读取，并可能根据用例进行写入。如需更多信息，请参阅 podman man page。

其它资源

- 容器入门
- 管理容器
- OpenShift Enterprise 架构的核心概念

- 镜像的 **/help.1** 文件中的 Red Hat Software Collections 容器镜像或上游 GitHub 存储库 中的 README 文件。

1.1. RED HAT SOFTWARE COLLECTIONS CONTAINER IMAGES 作为构建器镜像

您可以使用 Red Hat Software Collections 容器镜像作为构建器镜像，以构建、部署和运行您的应用程序。为了支持常见用例，构建器镜像中包含以下 Source-to-Image (S2I)脚本：

- 运行镜像内的 **/usr/libexec/s2i/assemble** 脚本，以生成包含应用程序工件的新镜像。脚本取给定应用程序的来源，并将它们放置在镜像中的适当目录中。如果应用程序源包含依赖组件的定义（例如，在 Python 项目的情况下，列出 **PyPi** 组件的 **requirements.txt** ），则组件将安装到镜像中。

- 在生成的容器镜像（带有应用程序工件的新镜像）中，**/usr/libexec/s2i/run** 脚本被设置为默认命令。

您可以使用 **podman** 运行生成的应用程序镜像。具体步骤请参阅 使用容器。在 Red Hat Enterprise Linux 7 中，您仍然可以使用相同的命令行语法使用 **docker** 命令而不是 **podman**。

1.2. 扩展现有容器镜像

要扩展红帽提供的容器镜像的功能，您可以以下选项：

- 设置环境变量。请参阅相应容器镜像的文档。

- 使用 OpenShift 机密。

- 构建您的自定义应用程序镜像。具体说明请查看 第 2 章 *使用 Red Hat Software Collections 容器镜像构建应用镜像*。

- 在 OpenShift 中使用 Source-to-Image 构建策略，您可以在支持此功能的守护进程镜像中添加自己的配置文件。按照相关容器镜像的文档进行操作。

- 如果其他守护进程或数据库镜像，请在提供的容器镜像之上构建新容器。编写自定义 Dockerfile，并在 FROM 子句中使用原始容器。请参阅关于相应容器镜像的 *文档中* 名为构建应用的部分，或知识库文章如何扩展 rhscl/mariadb-101-rhel7 容器镜像 的示例部分。

第 2 章 使用 RED HAT SOFTWARE COLLECTIONS 容器镜像构建应用镜像

您可以使用 Red Hat Software Collections 容器镜像构建应用程序镜像的几个选项：

- 使用红帽提供的容器镜像作为基础镜像
- 使用 S2I 脚本的 Dockerfile
- 在 OpenShift 中使用 **Source-to-Image**
- 使用 **source-to-image** 实用程序

2.1. 使用 RED HAT SOFTWARE COLLECTIONS 镜像构建应用程序镜像作为基础镜像

使用红帽提供的容器镜像作为基础镜像：

1. 为您的应用程序镜像创建一个 Dockerfile，并确保该文件包含以下行：

```
FROM registry.redhat.io/rhscl_image_name
```

2. 通过将以下行放在 Dockerfile 中，将应用程序代码添加到镜像中：

```
ADD src /opt/app-root/src
```

3. **使用 podman 构建应用程序镜像：**

```
# podman build -t application_image_name .
```

4. **使用 podman 运行应用程序镜像。例如，要在应用程序镜像中启动交互式 shell，请运行：**

```
# podman run -ti application_image_name /bin/bash -l
```

例 2.1. 使用 rhscl/python-38-rhel7 基础镜像从 Dockerfile 构建的 Django 应用

本例演示了一个 Dockerfile，可用于从 rhscl/python-38-rhel7 容器镜像创建简单 Django 应用。

```
# Set base image
FROM registry.redhat.io/rhscl/python-38-rhel7

# Add application sources
ADD --chown=1001:0 app-src .

# Install the dependencies
```

```
RUN pip install -U "pip>=19.3.1" && \
    pip install -r requirements.txt && \
    python manage.py collectstatic --noinput && \
    python manage.py migrate

# Run the application
CMD python manage.py runserver 0.0.0.0:8080
```

其它资源

- 从 Dockerfile 构建镜像
- Dockerfile 参考文档

2.2. 使用 S2I 脚本从 DOCKERFILE 构建应用程序镜像

您可以将 Red Hat Software Collections 容器镜像用作构建器镜像，并使用 assemble 并运行 构建器镜像中包含的 S2I 脚本从 Dockerfile 构建应用程序镜像。有关 assemble 和 run S2I 脚本的更多信息，请参阅 第 1.1 节 “Red Hat Software Collections Container Images 作为构建器镜像”。

要使用 S2I 脚本从 Dockerfile 创建应用程序镜像，请按照以下步骤操作：

1. 登录到容器 registry：

```
# podman login registry.redhat.io
```

2. 拉取构建器镜像：

```
# podman pull registry.redhat.io/rhscl_image_name
```

3. 准备应用程序代码。

4. 为应用程序镜像创建自定义 Dockerfile 并确保您：

   a. 使用这一行定义构建器镜像：

```
FROM registry.redhat.io/rhscl_image_name
```

b. **将应用程序源放在 src/ 目录中，并确保默认容器用户具有访问源的足够权限：**

```
ADD --chown=1001:0 src /tmp/src
```

c. **使用 /usr/libexec/s2i/assemble 脚本安装依赖项：**

```
RUN /usr/libexec/s2i/assemble
```

d. **使用 /usr/libexec/s2i/run 脚本在生成的镜像中设置默认命令：**

```
CMD /usr/libexec/s2i/run
```

5. **使用 podman 构建应用程序镜像：**

```
# podman build -t application_image_name .
```

6. **使用 podman 运行应用程序镜像。例如，要在应用程序镜像中启动交互式 shell，请运行：**

```
# podman run -ti application_image_name /bin/bash -l
```

例 2.2. 使用 S2I 脚本从 Dockerfile 创建 Python 3.8 应用程序镜像

这个示例演示了如何使用构建器镜像提供的 S2I 脚本从 Dockerfile 构建并运行 Python 3.8 应用程序。

1. **登录到容器 registry：**

```
# podman login registry.redhat.io
```

2. **拉取构建器镜像：**

```
# podman pull registry.redhat.io/rhscl/python-38-rhel7
```

3. 拉取位于 https://github.com/sclorg/django-ex.git 的应用代码：

```
$ git clone https://github.com/sclorg/django-ex.git app-src
```

或者，使用位于 https://github.com/sclorg/s2i-python-container/tree/master/examples 的示例。

4. 使用以下内容创建 Dockerfile：

```
FROM registry.redhat.io/rhscl/python-38-rhel7

# Add application sources to a directory that the assemble script expects them
# and set permissions so that the container runs without root access
USER 0
ADD app-src /tmp/src
RUN chown -R 1001:0 /tmp/src
USER 1001

# Install the dependencies
RUN /usr/libexec/s2i/assemble

# Set the default command for the resulting image
CMD /usr/libexec/s2i/run
```

5. 从上一步中准备的 Dockerfile 构建新镜像：

```
# podman build -t python-app .
```

6. 使用您的 Python 应用程序运行生成的镜像：

```
# podman run -d python-app
```

其它资源

- 从 Dockerfile 构建镜像
- Dockerfile 参考文档
-

相应的构建器镜像 README 文件中的 *Source-to-Image 环境变量* 部分，它们位于镜像内的 /help.1 文件中，或者位于上游 GitHub 存储库中。

- 环境变量也包括了 红帽生态系统目录 中镜像的详细描述。

2.3. 在 OPENSHIFT 中使用 SOURCE-TO-IMAGE 构建应用镜像

OpenShift 中的 Source-to-Image (S2I)是一个框架，它可让您将应用源代码作为输入来编写镜像，使用构建器 Red Hat Software Collections 容器镜像，并生成运行汇编的应用作为输出的新镜像。

在 OpenShift 中使用 S2I 创建应用程序：

1. 使用通过 OpenShift 提供的镜像构建应用：

```
$ oc new-app openshift_image_name~path_to_application_source_code
```

例如，要使用 OpenShift 中的 python:3.8 镜像流标签提供的受支持镜像构建 Python 3.8 应用程序，请运行：

```
$ oc new-app python:3.8~https://github.com/sclorg/django-ex.git
```

2. 列出可用的 pod （实例）：

```
$ oc get pods
```

3. 在 localhost 上执行所选 pod:

```
$ oc exec pod -- curl 127.0.0.1:8080
```

其它资源

- OpenShift Container Platform documentation
- S2I 要求

- GitHub 上的 Source-to-image README 文件
- 相应构建器镜像 README 文件中的 Source-to-Image 部分的环境变量。

2.4. 使用 SOURCE-TO-IMAGE 实用程序构建应用程序镜像

Red Hat Software Collections 提供了 source-to-image 实用程序，您可以在 Red Hat Enterprise Linux 7 服务器中无需 OpenShift 的情况下使用它。

注意

source-to-image 实用程序仅适用于 Red Hat Enterprise Linux 7，且只适用于 docker 拉取的镜像。您不能将 podman 与 source-to-image 实用程序搭配使用。

构建过程包含以下三个基本元素，这些元素组合成最终的容器镜像：

- 应用的源代码，使用编程语言或框架编写。
- 构建器映像，这是 Red Hat Software Collections 容器镜像，它支持使用 source-to-image 实用程序构建镜像。
- 作为构建器镜像一部分的 S2I 脚本。有关这些脚本的详情请参考 第 1.1 节 “Red Hat Software Collections Container Images 作为构建器镜像”。

在构建过程中，source-to-image 实用程序会创建一个 .tar 文件，其中包含源代码和脚本，然后将该文件流传输到构建器镜像中。

在您的系统中使用 source-to-image 工具：

1. 订阅红帽软件集合.具体步骤，请参阅 获取 Red Hat Software Collections。
2. 启用 Red Hat Software Collections Server 存储库，它提供 source-to-image 软件包和

Red Hat Enterprise Linux 7 Server 软件仓库，其中包含 source-to-image 所需的 docker 软件包：

```
# subscription-manager repos --enable rhel-server-rhscl-7-rpms --enable rhel-7-server-extras-rpms
```

3. **安装 source-to-image 软件包：**

```
# yum install source-to-image
```

4. **登录到容器 registry：**

```
# docker login registry.redhat.io
```

拉取构建器镜像：

```
# docker pull registry.redhat.io/rhscl_image_name
```

从应用程序源代码构建应用程序镜像：

```
# s2i build path_to_application_source_code_repository --context-dir=source_code_context_directory application_image_name
```

5. **使用 docker 运行生成的镜像。**

例 2.3. 使用 source-to-image 实用程序从 Git 存储库构建 Python 3.8 应用程序

本例演示了如何使用 rhscl/python-38-rhel7 构建器镜像和 source-to-image 实用程序构建可从公共 Git 存储库中提供的测试应用。

1. **登录到容器 registry：**

```
# docker login registry.redhat.io
```

2. **拉取 rhscl/python-38-rhel7 构建器镜像：**

```
# docker pull registry.redhat.io/rhscl/python-38-rhel7
```

3. **在 3.8/test/setup-test-app/ 目录中从 GitHub s2i-python 存储库构建测试应用：**

```
# s2i build https://github.com/sclorg/s2i-python-container.git --context-dir=3.8/test/setup-test-app/ registry.redhat.io/rhscl/python-38-rhel7 python-38-rhel7-app
```

这会生成新应用镜像 python-38-rhel7-app。

4. **运行生成的 python-38-rhel7-app 镜像：**

```
# docker run -d -p 8080:8080 --name example-app python-38-rhel7-app
```

5. **从 http://localhost:8080/ 获取生成的示例文档：**

```
$ wget http://localhost:8080/
```

6. **停止容器：**

```
# docker stop example-app
```

其它资源

- **S2I 要求**
- **GitHub 上的 Source-to-image README 文件**
- **相应的构建器镜像 README 文件中的 *Source-to-Image 环境变量部分*，它们位于镜像内的 /help.1 文件中，或者位于上游 GitHub 存储库中。**

第 3 章 基于 RED HAT SOFTWARE COLLECTIONS 3.8 的容器镜像

组件	描述	支持的构架
守护进程镜像		
rhscl/nginx-120-rhel7	Nginx 1.20 服务器和反向代理服务器	x86_64, s390x, ppc64le
数据库镜像		
rhscl/redis-6-rhel7	Redis 6 键值存储	x86_64, s390x, ppc64le
Red Hat Developer Toolset Images		
rhscl/devtoolset-12-toolchain-rhel7（自 2022 年 11 月起提供）	Red Hat Developer Toolset toolchain	x86_64, s390x, ppc64le
rhscl/devtoolset-12-perftools-rhel7（自 2022 年 11 月起提供）	Red Hat Developer Toolset perftools	x86_64, s390x, ppc64le
rhscl/devtoolset-11-toolchain-rhel7	Red Hat Developer Toolset 工具链(EOL)	x86_64, s390x, ppc64le
rhscl/devtoolset-11-perftools-rhel7	Red Hat Developer Toolset perftools(EOL)	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.8 提供的组件的详细信息，请参阅 Red Hat Software

Collections 3.8 发行注记。

有关 Red Hat Developer Toolset 11 组件的更多信息，请参阅 Red Hat Developer Toolset 11 用户指南。

有关 Red Hat Developer Toolset 12 组件的详情，请参考 Red Hat Developer Toolset 12 用户指南。

EOL 镜像不再被支持。

第 4 章 容器镜像基于 RED HAT SOFTWARE COLLECTIONS 3.7

组件	描述	支持的构架
应用程序镜像		
rhscl/ruby-30-rhel7	用于构建和运行应用程序的 Ruby 3.0 平台	x86_64, s390x, ppc64le
rhscl/ruby-27-rhel7	用于构建和运行应用程序的 Ruby 2.7 平台 (EOL)	x86_64, s390x, ppc64le
rhscl/ruby-26-rhel7	用于构建和运行应用程序的 Ruby 2.6 平台 (EOL)	x86_64, s390x, ppc64le
数据库镜像		
rhscl/mariadb-105-rhel7	MariaDB 10.5 SQL 数据库服务器	x86_64, s390x, ppc64le
rhscl/postgresql-13-rhel7	PostgreSQL 13 SQL 数据库服务器	x86_64, s390x, ppc64le
Red Hat Developer Toolset Images		
rhscl/devtoolset-10-toolchain-rhel7	Red Hat Developer Toolset 工具链 (EOL)	x86_64, s390x, ppc64le
rhscl/devtoolset-10-perftools-rhel7	Red Hat Developer Toolset perftools (EOL)	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.7 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.7 发行注记。

有关 Red Hat Developer Toolset 10 组件的更多信息，请参阅 Red Hat Developer Toolset 10 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 5 章 基于 RED HAT SOFTWARE COLLECTIONS 3.6 的容器镜像

组件	描述	支持的构架
应用程序镜像		
rhscl/nodejs-14-rhel7	Node.js 14 平台用于构建和运行应用程序	x86_64, s390x, ppc64le
rhscl/perl-530-rhel7	用于构建和运行应用程序的 Perl 5.30 平台	x86_64, s390x, ppc64le
rhscl/php-73-rhel7	用于构建和运行应用程序的 PHP 7.3 平台	x86_64, s390x, ppc64le
rhscl/ruby-25-rhel7	用于构建和运行应用程序的 Ruby 2.5 平台 (EOL)	x86_64
守护进程镜像		
rhscl/httpd-24-rhel7	Apache HTTP 2.4 Server	x86_64, s390x, ppc64le
rhscl/nginx-118-rhel7	nginx 1.18 服务器和反向代理服务器(EOL)	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.6 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.6 发行注记。

有关 Red Hat Developer Toolset 10 组件的更多信息，请参阅 Red Hat Developer Toolset 10 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 6 章 基于 RED HAT SOFTWARE COLLECTIONS 3.5 的容器镜像

组件	描述	支持的构架
应用程序镜像		
rhscl/python-38-rhel7	用于构建和运行应用程序的 Python 3.8 平台	x86_64, s390x, ppc64le
守护进程镜像		
rhscl/varnish-6-rhel7	Varnish Cache 6.0 HTTP 反向代理	x86_64, s390x, ppc64le
Red Hat Developer Toolset Red Hat Developer Toolset Images		
rhscl/devtoolset-9-toolchain-rhel7	Red Hat Developer Toolset 工具链(EOL)	x86_64, s390x, ppc64le
rhscl/devtoolset-9-perftools-rhel7	Red Hat Developer Toolset perftools(EOL)	x86_64, s390x, ppc64le

图例：

- x86_64 - AMD64 和 Intel 64 架构
- s390x - 64 位 IBM Z
- ppc64le - IBM POWER, little endian

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.5 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.5 发行注记。

有关 Red Hat Developer Toolset 9.1 组件的更多信息，请参阅 Red Hat Developer Toolset 9 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 7 章 容器镜像基于红帽软件集合 3.4

组件	描述	支持的构架
应用程序镜像		
rhscl/nodejs-12-rhel7	Node.js 12 平台用于构建和运行应用程序 (EOL)	x86_64, s390x, ppc64le
守护进程镜像		
rhscl/nginx-116-rhel7	Nginx 1.16 服务器和反向代理服务器(EOL)	x86_64, s390x, ppc64le
数据库镜像		
rhscl/postgresql-12-rhel7	PostgreSQL 12 SQL 数据库服务器	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.4 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.4 发行注记。

有关 Red Hat Developer Toolset 9.0 组件的更多信息，请参阅 Red Hat Developer Toolset 9 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software

Collections 2 容器镜像。

EOL 镜像不再被支持。

第 8 章 基于 RED HAT SOFTWARE COLLECTIONS 3.3 的容器镜像

组件	描述	支持的构架
数据库镜像		
rhscl/mariadb-103-rhel7	MariaDB 10.3 SQL 数据库服务器(EOL)	x86_64, s390x, ppc64le
rhscl/redis-5-rhel7	redis 5 键值存储(EOL)	x86_64, s390x, ppc64le
Red Hat Developer Toolset Images		
rhscl/devtoolset-8-toolchain-rhel7	Red Hat Developer Toolset 工具链(EOL)	x86_64, s390x, ppc64le
rhscl/devtoolset-8-perftools-rhel7	Red Hat Developer Toolset perftools(EOL)	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.3 提供的组件的详情，请查看 Red Hat Software Collections 3.3 发行注记。

有关 Red Hat Developer Toolset 8.1 组件的更多信息，请参阅 Red Hat Developer Toolset 8 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 9 章 容器镜像基于 RED HAT SOFTWARE COLLECTIONS 3.2

组件	描述	支持的构架
应用程序镜像		
rhscl/nodejs-10-rhel7	Node.js 10 平台用于构建并运行应用程序 (EOL)	x86_64, s390x, ppc64le
rhscl/php-72-rhel7	用于构建和运行应用程序的 PHP 7.2 平台 (EOL)	x86_64, s390x, ppc64le
守护进程镜像		
rhscl/nginx-114-rhel7	Nginx 1.14 服务器和反向代理服务器(EOL)	x86_64, s390x, ppc64le
数据库镜像		
rhscl/mysql-80-rhel7	MySQL 8.0 SQL 数据库服务器	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.2 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.2 发行注记。

有关 Red Hat Developer Toolset 8.0 组件的更多信息，请参阅 Red Hat Developer Toolset 8 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 10 章 基于 RED HAT SOFTWARE COLLECTIONS 3.1 的容器镜像

组件	描述	支持的构架
应用程序镜像		
rhscl/php-70-rhel7	用于构建和运行应用程序的 PHP 7.0 平台 (EOL)	x86_64
rhscl/perl-526-rhel7	用于构建和运行应用程序的 Perl 5.26 平台 (EOL)	x86_64
守护进程镜像		
rhscl/varnish-5-rhel7	Varnish Cache 5.0 HTTP 反向代理(EOL)	x86_64, s390x, ppc64le
数据库镜像		
rhscl/mongodb-36-rhel7	MongoDB 3.6 NoSQL 数据库服务器(EOL)	x86_64
rhscl/postgresql-10-rhel7	PostgreSQL 10 SQL 数据库服务器	x86_64, s390x, ppc64le
Red Hat Developer Toolset Images		
rhscl/devtoolset-7-toolchain-rhel7	Red Hat Developer Toolset 工具链(EOL)	x86_64, s390x, ppc64le
rhscl/devtoolset-7-perftools-rhel7	Red Hat Developer Toolset perftools(EOL)	x86_64, s390x, ppc64le

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像

可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.1 提供的组件的详细信息，请参阅 Red Hat Software Collections 3.1 发行注记。

有关 Red Hat Developer Toolset 7.1 组件的更多信息，请参阅 Red Hat Developer Toolset 7 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 11 章 基于 RED HAT SOFTWARE COLLECTIONS 3.0 的容器镜像

组件	描述	支持的构架
应用程序镜像		
rhscl/nodejs-8-rhel7	Node.js 8 平台用于构建并运行应用程序(EOL)	x86_64, s390x, ppc64le
rhscl/php-71-rhel7	用于构建和运行应用程序的 PHP 7.1 平台 (EOL)	x86_64
rhscl/python-36-rhel7	用于构建和运行应用程序的 Python 3.6 平台 (EOL)	x86_64, s390x, ppc64le
守护进程镜像		
rhscl/nginx-112-rhel7	Nginx 1.12 服务器和反向代理服务器(EOL)	x86_64, s390x, ppc64le
数据库镜像		
rhscl/mariadb-102-rhel7	MariaDB 10.2 SQL 数据库服务器(EOL)	x86_64
rhscl/mongodb-34-rhel7	MongoDB 3.4 NoSQL 数据库服务器(EOL)	x86_64
rhscl/postgresql-96-rhel7	PostgreSQL 9.6 SQL 数据库服务器(EOL)	x86_64

图例：

- **x86_64 - AMD64 和 Intel 64 架构**
- **s390x - 64 位 IBM Z**
- **ppc64le - IBM POWER, little endian**

所有镜像都基于 Red Hat Software Collections 中的组件。通过 Red Hat Container Registry，镜像可用于 Red Hat Enterprise Linux 7。

有关 Red Hat Software Collections 3.0 提供的组件的详细信息，请参阅 Red Hat Software

Collections 3.0 发行注记。

有关 Red Hat Developer Toolset 7.0 组件的更多信息，请参阅 Red Hat Developer Toolset 7 用户指南。

有关基于 Red Hat Software Collections 2 的容器镜像的信息，请参阅使用 Red Hat Software Collections 2 容器镜像。

EOL 镜像不再被支持。

第 12 章 应用程序镜像

12.1. NODE.JS

12.1.1. 描述

rhscl/nodejs-14-rhel7 镜像提供了一个 Node.js 14 平台，用于构建和运行应用。

12.1.2. 权限

要拉取 rhscl/nodejs-14-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/nodejs-14-rhel7
```

12.1.3. 配置

要设置环境变量，您可以将它们作为键值对放在源代码存储库中的 .s2i/environment 文件中。

变量名称	描述
NODE_ENV	nodejs 运行时模式（默认："生产环境"）
DEV_MODE	当设置为 "true" 时，会使用 **nodemon** 自动重新载入服务器（默认："false"）。将 **DEV_MODE** 设为 "true" 时，会将 **NODE_ENV** 默认更改为"development" （如果没有明确设置）。
NPM_RUN	选择在 **package.json** 文件脚本部分定义的备用 / 自定义运行时模式（默认：npm run "start"）**https://docs.npmjs.com/misc/scripts在使用 DEV_MODE 时，这些用户定义的 run-scripts 不可用。**
HTTP_PROXY	在装配过程中使用 npm 代理
HTTPS_PROXY	在装配过程中使用 npm 代理
NPM_MIRROR	在构建过程中使用自定义 NPM registry 镜像下载软件包

12.2. PHP

12.2.1. 描述

rhscl/php-73-rhel7 镜像提供了一个用于构建和运行应用程序的 PHP 7.3 平台。带有 npm 的 Node.js 预安装在 PHP 镜像中。

12.2.2. 权限

要拉取 rhscl/php-73-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/php-73-rhel7
```

12.2.3. 配置

要设置环境变量，请将它们作为键值对放在源代码存储库中的 .s2i/environment 文件中。

以下环境变量在 php.ini 文件中设置与其对等的属性值：

变量名称	描述	默认
ERROR_REPORTING	告知 PHP 定义要采取哪个错误、警告和通知	**E_ALL & ~E_NOTICE**
DISPLAY_ERRORS	控制 PHP 将输出错误以及什么位置，通知和接收	**ON**
DISPLAY_STARTUP_ERRORS	导致 PHP 启动序列中出现的显示错误与显示错误分开处理	**OFF**
TRACK_ERRORS	将最后的错误/警告消息存储在 **$php_errormsg** (boolean)中。	**OFF**
HTML_ERRORS	将错误链接到与错误相关的文档	**ON**
INCLUDE_PATH	PHP 源文件的路径	**.:/opt/app-root/src:/opt/rh/rh-php73/root/usr/share/pear**
PHP_MEMORY_LIMIT	内存限制	128M
SESSION_NAME	会话的名称	**PHPSESSID**

变量名称	描述	默认
SESSION_HANDLER	保存会话的方法	**files**
SESSION_PATH	会话数据文件的位置	**/tmp/sessions**
SESSION_COOKIE_DOMAIN	Cookie 有效的域	
SESSION_COOKIE_HTTPONLY	是否向 Cookie 中添加 httpOnly 标记	**0**
SESSION_COOKIE_SECURE	指定 Cookie 是否只通过安全连接发送	**OFF**
SHORT_OPEN_TAG	确定 PHP 是否会识别 < 和 ?> 标签之间的代码	**OFF**
DOCUMENTROOT	为您的应用程序定义 DocumentRoot 的路径（例如 /public）	/

根据需要替换 rh-php7* Software Collection 的版本。

以下环境变量在 opcache.ini 文件中设置其对等属性值：

变量名称	描述	默认
OPCACHE_MEMORY_CONSUMPTION	OPcache 共享内存存储大小（以 MB 为单位）	128
OPCACHE_REVALIDATE_FREQ	检查更新的脚本时间戳的频率，以秒为单位。0 将导致 OPcache 检查每个请求的更新。	2
OPCACHE_MAX_FILES	OPcache 哈希表中的键(scripts)的最大数量。只允许 200 到 1000000 间的数字。	4000

您还可以通过设置来覆盖用于加载 PHP 配置的完整目录：

变量名称	描述
PHPRC	设置到 **php.ini** 文件的路径
PHP_INI_SCAN_DIR	扫描额外 ini 配置文件中的路径

您可以覆盖 Apache MPM prefork 设置来提高 PHP 应用的性能。如果您设置了 Cgroup 限制，则镜像将尝试自动设置最佳值。您可以通过自行指定值，随时覆盖此项：

变量名称	描述	默认
HTTPD_START_SERVERS	StartServers 指令设定启动时创建的子服务器进程数目。	8
HTTPD_MAX_REQUEST_WORKERS	MaxRequestWorkers 指令设置要提供的并发请求数的限值。	256 （这可以通过使用这个公式为容器设置 Cgroup 限制来自动调整：**TOTAL_MEMORY / 15MB**。15MB 是单个 **httpd** 进程的平均大小。

您可以使用自定义 composer 存储库镜像 URL 来下载软件包，而不是默认的 packagist.org ：

变量名称	描述
COMPOSER_MIRROR	将自定义 composer 存储库镜像 URL 添加到 composer 配置。注：这只会影响 **composer.json** 中列出的软件包。
COMPOSER_INSTALLER	覆盖下载 https://getcomposer.org/installer Composer 的默认 URL。在断开连接的环境中很有用。
COMPOSER_ARGS	在 **composer install** 命令行添加额外的参数（如 **--no-dev**）。

如果应用程序的 DocumentRoot 位于源目录 /opt/app-root/src 中，用户可以提供自己的 .htaccess 文件。这允许覆盖 Apache 的行为并指定如何处理应用程序请求。.htaccess 文件需要位于应用程序源的根目录下。有关 .htaccess 的详情，请查看 Apache HTTP 服务器教程。

12.2.4. 延长镜像

可以使用 Source-to-image 扩展 PHP 镜像。

例如，要使用 ~/image-configuration/ 目录中的配置构建自定义 PHP 镜像 my-php-rhel7，请运行：

```
$ s2i build ~/image-configuration/ rhscl/php-73-rhel7 my-php-rhel7
```

确保相应地更改源镜像版本。

应用程序的结构与以下示例类似：

目录名称	描述
./httpd-cfg	可以包含其他 Apache 配置文件(***.conf**)
./httpd-ssl	可以包含自己的 SSL 证书（在 **certs/** 子目录中）和密钥（在 **private/** 子目录中）
./php-pre-start	可以包含在 **httpd** 启动前提供的 shell 脚本(***.sh**)
./php-post-assemble	可以包含 **assemble** 脚本末尾提供的 shell 脚本(***.sh**)
./	应用源代码

12.3. PERL

12.3.1. 描述

rhscl/perl-530-rhel7 镜像提供了用于构建和运行应用程序的 Perl 5.30 平台。预安装用于部署 Perl Web 应用程序的 Apache httpd 2.4 和带有 mod_perl 的 Node.js。

这些镜像还支持部署 Perl Web Server Gateway Interface (PSGI)应用程序。

12.3.2. 权限

要拉取 rhscl/perl-530-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/perl-530-rhel7
```

-

12.3.3. 配置

要设置环境变量，您可以将它们作为键值对放在源代码存储库中的 .s2i/environment 文件中。

变量名称	描述	默认
ENABLE_CPAN_TEST	允许安装所有指定的 cpan 软件包及其测试	**false**
CPAN_MIRROR	指定 cpanminus 用来安装依赖项的镜像 URL	默认情况下不指定 URL
PERL_APACHE2_RELOAD	启用自动重新载入修改的 Perl 模块	**false**
HTTPD_START_SERVERS	StartServers 指令设置启动时创建的子服务器进程数	**8**
HTTPD_MAX_REQUEST_WORKERS	Apache 可同时处理的请求数	**256**，但如果内存有限，则会自动降低
PSGI_FILE	指定到 PSGI 应用程序文件的相对路径。使用空值禁用 PSGI 自动配置	顶层目录中单个 *.psgi 文件（如果存在）
PSGI_URI_PATH	指定由 PSGI 应用程序处理的 URI 路径	/

要从完整的 Perl Archive Network (CPAN)安装额外的 Perl 模块，请在应用程序源的根目录中创建 cpanfile。该文件必须符合 Module-CPANFile CPAN 发行版中定义的 cpanfile 格式。有关 cpanfile 格式的详细信息，请参考 cpanfile 文档。

要修改 Apache httpd 行为，请在相应应用程序源树中丢弃 .htaccess 文件。有关 .htaccess 的详情，请查看 Apache HTTP 服务器教程。

12.4. PYTHON

12.4.1. 描述

rhscl/python-38-rhel7 镜像提供了一个 Python 3.8 平台，用于构建和运行应用程序。带有 npm 的 Node.js 预安装。

12.4.2. 权限

要拉取 rhscl/python-38-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/python-38-rhel7
```

12.4.3. 配置

要设置环境变量，您可以将它们作为键值对放在源代码存储库中的 .s2i/environment 文件中。

变量名称	描述
APP_SCRIPT	用于从脚本文件运行应用。这应该是脚本文件的路径（默认为 **app.sh**，除非设置为 null），否则将运行 来启动应用。
APP_FILE	用于从 Python 脚本运行应用程序。这应该是到 Python 文件（默认为 **app.py**）的路径，该路径将传递到 Python 解释器以启动应用程序。
APP_MODULE	用于使用 Gunicorn 运行应用程序，如此处所述。这个变量指定了带有模式 **MODULE_NAME:VARIABLE_NAME** 的 WSGI 调用，其中 **MODULE_NAME** 是模块的完整点路径，**VARIABLE_NAME** 指的是指定模块中的 WSGI 调用。如果未指定，Gusnicorn 将查找名为 application 的 WSGI 调用。如果没有提供 **APP_MODULE**，则 run 脚本将在项目中查找 **wsgi.py** 文件，并在存在时使用它。如果使用 **setup.py** 安装应用程序，则可从那里读取 **MODULE_NAME** 部分。例如，请参阅 **setup-test-app**。
APP_HOME	此变量可用于指定包含要运行的应用程序的子目录。这个变量指向的目录需要包含 **wsgi.py**（如 Gunicorn）或 **manage.py** (for Django)。如果没有提供 **APP_HOME**，则 **assemble** 和 **run** 脚本将使用应用的根目录。
APP_CONFIG	使用 Gunicorn 配置文件的有效 Python 文件的路径。
DISABLE_MIGRATE	将这个变量设置为非空值，禁止在生成的镜像运行时执行 **manage.py migrate**。这只会影响 Django 项目。

变量名称	描述
DISABLE_COLLECTSTATIC	将这个变量设置为非空值，以禁止在构建期间执行 **manage.py collectstatic**。这只会影响 Django 项目。
DISABLE_SETUP_PY_PROCESSING	在 **requirements.txt** 中使用 **-e 触发其处理，或者您不希望应用程序安装到 site-** packages 目录中，将此变量设置为非空值，以跳过对 **setup.py** 脚本的处理。
ENABLE_PIPENV	将这个变量设置为使用 Pipenv （高级 Python 打包工具）来管理应用程序的依赖项。只有在项目包含适当格式化的 **Pipfile** 和 **Pipfile.lock** 时，才应使用此项。
ENABLE_INIT_WRAPPER	将这个变量设置为非空值，以使用 init 打包程序。对于无法获取 Zombie 进程（如 Django 开发服务器或 Tornado）的服务器来说，这非常有用。这个选项可以与 **APP_SCRIPT** 或 **APP_FILE** 变量一起使用。它从不适用于通过 **APP_MODULE** 作为 Gunicorn 获取 Zombie 进程使用的 Gunicorn。
PIP_INDEX_URL	将这个变量设置为使用自定义索引 URL 或镜像 (mirror)在构建过程中下载所需的软件包。这只会影响 requirements.txt 中列出的软件包。
UPGRADE_PIP_TO_LATEST	将这个变量设置为非空值，将 **pip** 程序升级到最新的版本，然后再安装任何 Python 软件包。如果没有设置，它将使用平台为所使用的 Python 版本包含的任何默认版本。
WEB_CONCURRENCY	设置它可更改 worker 数量的默认设置。默认情况下，它被设置为可用内核数 2。

12.5. RUBY

12.5.1. 描述

rhscl/ruby-30-rhel7 镜像提供了一个 Ruby 3.0 平台，用于构建和运行应用，rhscl/ruby-27-rhel7 镜像提供了一个 Ruby 2.7 平台。

带有 npm 的 Node.js 预安装。

12.5.2. 权限

要拉取 rhscl/ruby-30-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/ruby-30-rhel7
```

要拉取 rhscl/ruby-27-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/ruby-27-rhel7
```

12.5.3. 配置

要设置环境变量，您可以将它们作为键值对放在源代码存储库中的 .s2i/environment 文件中。

变量名称	描述
RACK_ENV	此变量指定将部署 Ruby 应用程序的环境（除非被覆盖）- **production**、**development**、**test**。每个级别在日志详细程度、错误页面、Ruby gem 安装和其他方面都有不同的行为。请注意，只有在将 **RACK_ENV** 设置为 **production** 时才会编译应用程序资产。
DISABLE_ASSET_COMPILATION	这个变量设置为 **true** 表示将跳过资产编译过程。由于只有在应用程序在生产环境中运行时才会进行，**所以**只有在资产已经编译后才应使用它。
PUMA_MIN_THREADS,**PUMA_MAX_THREADS**	这些变量表示 Puma 线程池中可用的最小和最大线程。
PUMA_WORKERS	这个变量表示将启动的 worker 进程数量。请参阅有关 Puma 集群模式 的文档。
RUBYGEM_MIRROR	将这个变量设置为使用自定义 RubyGems 镜像 URL 在构建过程中下载所需的 gem 软件包。

要使 S2I 脚本正常工作，您需要在应用程序的 Gemfile 中包含 puma 或 rack gem。

第 13 章 守护进程镜像

13.1. APACHE HTTP 服务器

13.1.1. 描述

rhscl/httpd-24-rhel7 镜像提供 Apache HTTP 2.4 服务器。该镜像可用作基于 Apache HTTP Web 服务器的其他应用的基础镜像。

13.1.2. 权限

要拉取 rhscl/httpd-24-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/httpd-24-rhel7
```

rhscl/httpd-24-rhel7 镜像支持使用 S2I 工具。

13.1.3. 配置和使用

Apache HTTP 服务器容器镜像 支持以下配置变量，这些变量可通过在 podman run 命令中使用 -e 选项进行设置：

变量名称	描述
HTTPD_LOG_TO_VOLUME	默认情况下，**httpd** 登录标准输出，因此可使用 **podman logs** 命令访问日志。当设置了 **HTTPD_LOG_TO_VOLUME** 时，**httpd** 会记录到 **/var/log/httpd24** 中，这可以使用容器卷挂载到主机系统。当容器以 UID **0** 运行时，允许这个选项。
HTTPD_MPM	可设置此变量从软件包默认 MPM 更改默认的 Multi-Processing Module (MPM)。

如果要运行镜像，并将日志文件作为容器卷挂载到主机上的 /wwwlogs 中，请执行以下命令：

```
$ podman run -d -u 0 -e HTTPD_LOG_TO_VOLUME=1 --name httpd -v /wwwlogs:/var/log/httpd24:Z
rhscl/httpd-24-rhel7
```

要使用 事件 MPM （而不是默认的 prefork）运行镜像，请执行以下命令：

```
$ podman run -d -e HTTPD_MPM=event --name httpd rhscl/httpd-24-rhel7
```

您还可以通过将 -v /host:/container 选项传递给 podman run 命令来设置以下挂载点：

卷挂载点	描述
/var/www	Apache HTTP 服务器数据目录
/var/log/httpd24	Apache HTTP 服务器日志目录（仅在以 root 用户身份运行时才可用）

将目录从主机迁移到容器时，请确保挂载的目录具有适当的权限，并且目录的所有者和组与容器中运行的用户 UID 或名称匹配。

注意

rhscl/httpd-24-rhel7 容器镜像现在使用 1001 作为默认 UID，在 OpenShift 中的 source-to-image 策略中正常工作。另外，容器镜像会默认侦听端口 8080。在以前的版本中，rhscl/httpd-24-rhel7 容器镜像默认侦听端口 80，并作为 UID 0 运行。

要将 rhscl/httpd-24-rhel7 容器镜像作为 UID 0 运行，请指定 podman run 命令的 -u 0 选项：

```
podman run -u 0 rhscl/httpd-24-rhel7
```

13.2. NGINX

13.2.1. 描述

rhscl/nginx-120-rhel7 镜像提供 nginx 1.20 服务器和反向代理服务器；镜像可用作基于 nginx 1.20 Web 服务器的基础镜像，即 rhscl/nginx-118-rhel7 镜像提供 nginx 1.18。

13.2.2. 权限

要拉取 rhscl/nginx-120-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/nginx-120-rhel7
```

要拉取 rhscl/nginx-118-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/nginx-118-rhel7
```

13.2.3. 配置

nginx 容器镜像支持以下配置变量，可将 -e 选项用于 podman run 命令进行设置：

变量名称	描述
NGINX_LOG_TO_VOLUME	默认情况下，nginx 会登录到标准输出，因此可使用 podman logs 命令访问日志。当设置了 NGINX_LOG_TO_VOLUME 时，nginx 会登录到 /var/opt/rh-nginx120/log/nginx / 或 /var/opt/rh/rh-nginx120/nginx/nginx/nginx/，它们可以通过容器卷挂载到主机系统。

使用 S2I 工具支持 rhscl/nginx-120-rhel7 和 rhscl/nginx-118-rhel7 镜像。

13.3. VARNISH 缓存

13.3.1. 描述

rhscl/varnish-6-rhel7 镜像提供 Varnish Cache 6.0，它是一个 HTTP 反向代理。

13.3.2. 权限

要拉取 rhscl/varnish-6-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/varnish-6-rhel7
```

13.3.3. 配置

不需要进一步配置。

Red Hat Software Collections Varnish 缓存镜像支持使用 S2I 工具。请注意，S2I 访问的目录中 default.vcl 配置文件需要采用 VCL 格式。

第 14 章 数据库镜像

14.1. MARIADB

14.1.1. 描述

rhscl/mariadb-105-rhel7 镜像提供 MariaDB 10.5 SQL 数据库服务器。

14.1.2. 权限

要拉取 rhscl/mariadb-105-rhel7 镜像，以 root 用户身份运行以下命令：

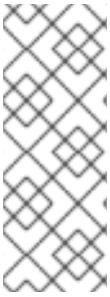
```
# podman pull registry.redhat.io/rhscl/mariadb-105-rhel7
```

14.1.3. 配置和使用

使用和配置与 MySQL 镜像相同。请注意，守护进程的名称为 mysqld，所有环境变量的名称都与 MySQL 中相同。

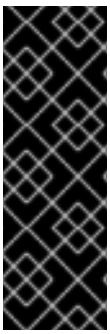
镜像通过将 -e VAR=VALUE 选项传递给 podman run 命令来识别，您可以在初始化过程中设置以下环境变量：

变量名称	描述
MYSQL_USER	要创建的 MySQL 帐户的用户名
MYSQL_PASSWORD	用户帐户的密码
MYSQL_DATABASE	数据库名称
MYSQL_ROOT_PASSWORD	root 用户的密码（可选）
MYSQL_CHARSET	默认字符集（可选）
MYSQL_COLLATION	默认冲突（可选）



注意

root 用户默认不设置密码，仅允许本地连接。您可以在初始化容器时设置 MYSQL_ROOT_PASSWORD 环境变量来设置它。这样，您可以远程登录到 root 帐户。本地连接仍不需要密码。要禁用远程 root 访问权限，只需取消设置 MYSQL_ROOT_PASSWORD 并重新启动容器。



重要

由于密码是镜像配置的一部分，因此唯一支持的为无特权用户(MYSQL_USER)和 root 用户更改密码的方法是分别更改环境变量 MYSQL_PASSWORD 和 MYSQL_ROOT_PASSWORD。通过 SQL 语句更改数据库密码或任何其他方法将导致变量中保存的值与实际密码不匹配。每当数据库容器启动时，它会将密码重置为环境变量中存储的值。

以下环境变量会影响 MySQL 配置文件且是可选的：

变量名称	描述	默认
MYSQL_LOWER_CASE_TABLE_NAMES	设置表名称的存储和比较方式	0
MYSQL_MAX_CONNECTIONS	允许客户端同时连接的最大数量	151
MYSQL_MAX_ALLOWED_PACKET	一个数据包或任何生成的/中间字符串的最大值	200M
MYSQL_FT_MIN_WORD_LEN	FULLTEXT 索引中包含的单词的最小长度	4
MYSQL_FT_MAX_WORD_LEN	FULLTEXT 索引中包含的单词的最大长度	20
MYSQL_AIO	如果原生 AIO 无法正常工作，控制 innodb_use_native_aio 设置值。See http://help.directadmin.com/item.php?id=529	1
MYSQL_TABLE_OPEN_CACHE	所有线程打开的表数	400
MYSQL_KEY_BUFFER_SIZE	用于索引块的缓冲大小	32M（或者 10% 可用内存）

变量名称	描述	默认
MYSQL_SORT_BUFFER_SIZE	用于排序的缓冲区的大小	256K
MYSQL_READ_BUFFER_SIZE	用于后续扫描的缓冲大小	8M（或者 5% 可用内存）
MYSQL_INNODB_BUFFER_POOL_SIZE	InnoDB 缓存表和索引数据的缓冲池的大小	32M（或者 50% 可用内存）
MYSQL_INNODB_LOG_FILE_SIZE	日志组中每个日志文件的大小	8M（或者 15% 可用内存）
MYSQL_INNODB_LOG_BUFFER_SIZE	InnoDB 用来写入磁盘日志文件的缓冲大小	8M（或者 15% 可用内存）
MYSQL_DEFAULTS_FILE	指向其它配置文件	**/etc/my.cnf**
MYSQL_BINLOG_FORMAT	设置 binlog 格式；支持的值是 **row** 和 **statement**	**声明**

当使用 --memory 参数集运行 MariaDB 镜像时，以下参数的值将根据可用内存自动计算，除非指定了参数：

变量名称	默认内存百分比
MYSQL_KEY_BUFFER_SIZE	10%
MYSQL_READ_BUFFER_SIZE	5%
MYSQL_INNODB_BUFFER_POOL_SIZE	50%
MYSQL_INNODB_LOG_FILE_SIZE	15%
MYSQL_INNODB_LOG_BUFFER_SIZE	15%

您还可以通过将 -v /host:/container 选项传递给 podman run 命令来设置以下挂载点：

卷挂载点	描述
/var/lib/mysql/data	MySQL 数据目录

注意

将目录从主机挂载到容器时，请确保挂载的目录具有适当的权限，并且目录的所有者和组与容器中运行的用户 UID 或名称匹配。

14.1.4. 延长镜像

请参阅 **如何扩展 rhscl/mariadb-101-rhel7 容器镜像，它也适用于 rhscl/mariadb-105-rhel7。**

14.2. MYSQL

14.2.1. 描述

rhscl/mysql-80-rhel7 镜像提供 MySQL 8.0 SQL 数据库服务器。

14.2.2. 访问和使用

要拉取 rhscl/mysql-80-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/mysql-80-rhel7
```

要只设置强制环境变量，而不将数据库存储在主机目录中，请执行以下命令：

```
# podman run -d --name mysql_database -e MYSQL_USER=<user> -e
MYSQL_PASSWORD=<pass> \
  -e MYSQL_DATABASE=<db> -p 3306:3306 rhscl/mysql-80-rhel7
```

这将创建一个名为 mysql_database 的容器，它使用数据库 db 和 user 运行 MySQL，其凭据用户为：pass。端口 3306 将公开并映射到主机。如果您希望数据库在容器执行之间持久，还要添加 -v /host/db/path:/var/lib/mysql/data 参数。目录 /host/db/path 将是 MySQL 数据目录。

如果没有初始化数据库目录，则入口点脚本将首先运行 mysql_install_db 并设置必要的数据库用户和密码。初始化数据库后，如果数据库已存在，则执行 mysqld，并将作为 PID 1 运行。您可以通过运行 podman stop mysql_database 命令来停止分离的容器。

14.2.3. 配置

镜像通过将 -e VAR=VALUE 传递给 podman run 命令来识别您可以在初始化过程中设置的以下环境变量：

变量名称	描述
MYSQL_USER	要创建的 MySQL 帐户的用户名
MYSQL_PASSWORD	用户帐户的密码
MYSQL_DATABASE	数据库名称
MYSQL_ROOT_PASSWORD	root 用户的密码（可选）

注意

root 用户默认不设置密码，仅允许本地连接。您可以在初始化容器时设置 MYSQL_ROOT_PASSWORD 环境变量来设置它。这样，您可以远程登录到 root 帐户。本地连接仍不需要密码。要禁用远程 root 访问权限，只需取消设置 MYSQL_ROOT_PASSWORD 并重新启动容器。

重要

由于密码是镜像配置的一部分，因此唯一支持的为无特权用户(MYSQL_USER)和 root 用户更改密码的方法是分别更改环境变量 MYSQL_PASSWORD 和 MYSQL_ROOT_PASSWORD。通过 SQL 语句更改数据库密码或任何其他方法将导致变量中保存的值与实际密码不匹配。每当数据库容器启动时，它会将密码重置为环境变量中存储的值。

以下环境变量会影响 MySQL 配置文件且是可选的：

变量名称	描述	默认
MYSQL_LOWER_CASE_TABLE_NAMES	设置表名称的存储和比较方式	0
MYSQL_MAX_CONNECTIONS	允许客户端同时连接的最大数量	151
MYSQL_MAX_ALLOWED_PACKET	一个数据包或任何生成的/中间字符串的最大值	200M

变量名称	描述	默认
MYSQL_FT_MIN_WORD_LEN	FULLTEXT 索引中包含的单词的最小长度	4
MYSQL_FT_MAX_WORD_LEN	FULLTEXT 索引中包含的单词的最大长度	20
MYSQL_AIO	如果原生 AIO 无法正常工作，控制 **innodb_use_native_aio** 设置值。See http://help.directadmin.com/item.php?id=529	1
MYSQL_TABLE_OPEN_CACHE	所有线程打开的表数	400
MYSQL_KEY_BUFFER_SIZE	用于索引块的缓冲大小	32M（或者 10% 可用内存）
MYSQL_SORT_BUFFER_SIZE	用于排序的缓冲区的大小	256K
MYSQL_READ_BUFFER_SIZE	用于后续扫描的缓冲大小	8M（或者 5% 可用内存）
MYSQL_INNODB_BUFFER_POOL_SIZE	InnoDB 缓存表和索引数据的缓冲池的大小	32M（或者 50% 可用内存）
MYSQL_INNODB_LOG_FILE_SIZE	日志组中每个日志文件的大小	8M（或者 15% 可用内存）
MYSQL_INNODB_LOG_BUFFER_SIZE	InnoDB 用来写入磁盘日志文件的缓冲大小	8M（或者 15% 可用内存）
MYSQL_DEFAULTS_FILE	指向其它配置文件	**/etc/my.cnf**
MYSQL_BINLOG_FORMAT	设置 binlog 格式，支持的值是 **row** 和 **statement**	**声明**
MYSQL_LOG_QUERIES_ENABLED	要启用查询日志记录，将此变量设置为 **1**	**0**

当使用 --memory 参数集运行 MySQL 镜像时，以下参数的值将根据可用内存自动计算，除非指定了参数：

变量名称	默认内存百分比
MYSQL_KEY_BUFFER_SIZE	10%
MYSQL_READ_BUFFER_SIZE	5%
MYSQL_INNODB_BUFFER_POOL_SIZE	50%
MYSQL_INNODB_LOG_FILE_SIZE	15%
MYSQL_INNODB_LOG_BUFFER_SIZE	15%

您还可以通过将 -v /host:/container 选项传递给 podman run 命令来设置以下挂载点：

卷挂载点	描述
/var/lib/mysql/data	MySQL 数据目录

注意

将目录从主机挂载到容器时，请确保挂载的目录具有适当的权限，并且目录的所有者和组与容器中运行的用户 UID 或名称匹配。

14.3. POSTGRESQL

14.3.1. 描述

rhscl/postgresql-13-rhel7 镜像提供 PostgreSQL 13 SQL 数据库服务器； rhscl/postgresql-12-rhel7 镜像提供 PostgreSQL 12 服务器，而 rhscl/postgresql-10-rhel7 镜像提供 PostgreSQL 10 服务器。

14.3.2. 访问和使用

要拉取 rhscl/postgresql-13-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/postgresql-13-rhel7
```

要拉取 rhscl/postgresql-12-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/postgresql-12-rhel7
```

要拉取 **rhscl/postgresql-10-rhel7** 镜像，以 **root** 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/postgresql-10-rhel7
```

要只设置强制环境变量，而不将数据库存储在主机目录中，请执行以下命令：

```
# podman run -d --name postgresql_database -e POSTGRESQL_USER=<user> \
  -e POSTGRESQL_PASSWORD=<pass> -e POSTGRESQL_DATABASE=<db> \
  -p 5432:5432 <image_name>
```

这将创建一个名为 **postgresql_database** 的容器，它使用数据库 **db** 和 **user** 运行 **PostgreSQL**，其凭据用户为 **：pass。端口 5432 将公开并映射到主机。如果您希望数据库在容器执行之间持久，还要添加 -v /host/db/path:/var/lib/pgsql/data 参数。这将是 PostgreSQL** 数据库集群目录。

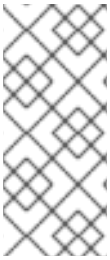
如果没有初始化数据库集群目录，则入口点脚本首先运行 **initdb** 并设置必要的数据库用户和密码。初始化数据库后，或者已经存在，则执行 **postgres**，并将作为 **PID 1** 运行。您可以通过运行 **podman stop postgresql_database** 命令来停止分离的容器。

postgres 守护进程首先将其日志写入标准输出。要检查容器镜像日志，请使用 **podman logs <image_name>** 命令。然后，日志输出被重定向到日志记录收集器进程，并出现在 **pg_log/** 目录中。

14.3.3. 配置

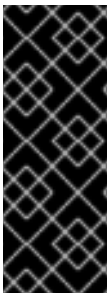
镜像通过将 **-e VAR=VALUE** 传递给 **podman run** 命令来识别您可以在初始化过程中设置的以下环境变量：

变量名称	描述
POSTGRESQL_USER	要创建的 PostgreSQL 帐户的用户名
POSTGRESQL_PASSWORD	用户帐户的密码
POSTGRESQL_DATABASE	数据库名称
POSTGRESQL_ADMIN_PASSWORD	postgres admin 帐户的密码（可选）



注意

postgres 管理员帐户默认没有设置密码，仅允许本地连接。您可以在初始化容器时设置 POSTGRESQL_ADMIN_PASSWORD 环境变量来设置它。这样，您可以远程登录到 postgres 帐户。本地连接仍不需要密码。



重要

由于密码是镜像配置的一部分，因此唯一支持的为数据库用户更改密码的方法是分别更改环境变量 POSTGRESQL_PASSWORD 和 POSTGRESQL_ADMIN_PASSWORD。通过 SQL 语句或者通过环境变量更改数据库密码会导致变量中保存的值与实际密码不匹配。每当数据库容器镜像启动时，它会将密码重置为环境变量中存储的值。

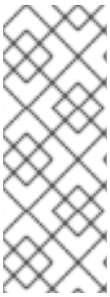
以下选项与迁移相关：

变量名称	描述	默认
POSTGRESQL_MIGRATION_REMOTE_HOST	要从中迁移的主机名/IP	
POSTGRESQL_MIGRATION_ADMIN_PASSWORD	远程 **postgres** admin 用户的密码	
POSTGRESQL_MIGRATION_IGNORE_ERRORS	可选：Ignore sql 导入错误	否

以下环境变量会影响 PostgreSQL 配置文件，并且都是可选的：

变量名称	描述	默认
POSTGRESQL_MAX_CONNECTIONS	允许的最大客户端连接数。这也设置准备的事务的最大数量。	100
POSTGRESQL_MAX_PREPARED_TRANSACTIONS	设置处于 "prepared" 状态的事务的最大数量。如果您使用准备好的事务，可能希望它至少像 max_connections 一样。	0
POSTGRESQL_SHARED_BUFFERS	设置专用于 PostgreSQL 用来缓存数据的内存量	32M

变量名称	描述	默认
POSTGRESQL_EFFECTIVE_CACHE_SIZE	设置为由操作系统和数据库本身提供磁盘缓存量的估算	128M



注意

当使用 --memory 参数集运行 PostgreSQL 镜像时，如果没有为 POSTGRESQL_SHARED_BUFFERS 和 POSTGRESQL_EFFECTIVE_SIZE 提供值，则根据 --memory 参数中提供的值自动计算这些值。该值基于上游公式来计算，并分别设置为 1/4 和 1/2。

您还可以通过将 -v /host:/container 选项传递给 podman run 命令来设置以下挂载点：

卷挂载点	描述
/var/lib/pgsql/data	PostgreSQL 数据库集群目录

注意

将目录从主机挂载到容器时，请确保挂载的目录具有适当的权限，并且目录的所有者和组与容器中运行的用户 UID 或名称匹配。

除非将 -u 选项与 podman run 命令搭配使用，否则容器中的进程通常会在 UID 26 下运行。要更改数据目录权限，请使用以下命令：

```
$ setfacl -m u:26:-wx /your/data/dir
$ podman run <...> -v /your/data/dir:/var/lib/pgsql/data:Z <...>
```

14.3.4. 数据迁移

PostgreSQL 容器镜像支持从远程 PostgreSQL 服务器迁移数据。使用以下命令并更改镜像名称，并在需要时添加可选配置变量：

```
$ podman run -d --name postgresql_database \
    -e POSTGRESQL_MIGRATION_REMOTE_HOST=172.17.0.2 \
    -e POSTGRESQL_MIGRATION_ADMIN_PASSWORD=remoteAdminP@ssword \
    [ OPTIONAL_CONFIGURATION_VARIABLES ]
    rhscl/postgresql-12-rhel7
```

迁移完成转储和恢复方法（针对远程集群运行 pg_dumpall 并通过 psql在本地导入转储）。因为进程已流化(unix 管道)，所以这个过程中没有创建中间转储文件，所以不会浪费额外的存储空间。

如果应用过程中某些 SQL 命令失败，则迁移脚本的默认行为也未能确保脚本化、无人值守迁移的"全部或无"结果。在大多数常见情况下，您期望迁移成功（但不保证），从先前版本的 PostgreSQL 服务器容器迁移（使用相同原则创建），例如：从 rhscl/postgresql-10-rhel7 迁移到 rhscl/postgresql-12-rhel7。从不同类型的 PostgreSQL 容器镜像迁移可能会失败。

如果这个"所有或无"原则不开了，则有一个可选的 POSTGRESQL_MIGRATION_IGNORE_ERRORS 选项，该选项该选项是"尽力迁移"。但是，一些数据可能会丢失，并且用户最多可以检查标准错误输出，并在迁移后手动解决问题。

注意

容器镜像为用户提供迁移帮助，但无法保证完全自动迁移。因此，在开始迁移数据库前，您需要执行手动步骤来获取迁移的所有数据。

在迁移场景中，您可能不使用 POSTGRESQL_USER 等变量。所有数据（包括数据库、角色或密码）的信息都从旧集群中复制。确保您使用与用于初始化旧 PostgreSQL 容器镜像相同的可选配置变量。如果在远程集群中进行一些非默认配置，您可能需要手动复制配置文件。

警告

旧集群和新 PostgreSQL 集群之间的 IP 通信默认没有加密，而是取决于用户在远程集群中配置 SSL，或使用不同方法确保安全性。

14.3.5. 升级数据库

警告

在决定执行数据目录升级前，请确定您已备份所有数据。请注意，如果升级失败，您可能需要手动回滚。

PostreSQL 镜像支持自动升级由上一 rhscl 镜像提供的 PostgreSQL 服务器版本创建的数据目录，例如 rhscl/postgresql-13-rhel7 镜像支持从 rhscl/postgresql-12-rhel7 进行升级。升级过程旨在，您应能够只从镜像 A 切换到镜像 B，并相应地设置 $POSTGRESQL_UPGRADE 变量来显式请求数据库数据转换。

升级过程使用 pg_upgrade 二进制文件进行内部实施，以便容器需要包含两个版本的 PostgreSQL 服务器（请参阅 pg_upgrade man page 了解更多信息）。

对于 pg_upgrade 进程和新服务器版本，需要初始化新的数据目录。这个数据目录由 /var/lib/pgsql/data/ 目录中的容器工具自动创建，它通常是外部 bind-mountpoint。然后，pg_upgrade 执行与转储和恢复方法类似。它同时启动旧的和新的 PostgreSQL 服务器（在容器中使用）和"转储"旧数据目录，同时它会"恢复"到新数据目录中。此操作需要复制许多数据文件。根据您选择的升级类型，相应地设置 $POSTGRESQL_UPGRADE 变量：

复制	数据文件从旧数据目录复制到新目录中。如果升级失败，这个选项的风险较低。
hardlink	数据文件从旧数据到新的数据目录中，从而提高性能。但是，旧目录变得不可用，即使出现失败也是如此。

注意

确保有足够的空间用于复制的数据。因为空间不足而无法升级失败，可能会导致数据丢失。

14.3.6. 延长镜像

PostgreSQL 镜像可以使用 Source-to-image 扩展。

例如，若要使用 ~/image-configuration/ 目录中配置构建自定义 new-postgresql 镜像，请使用以下

命令：

```
$ s2i build ~/image-configuration/ postgresql new-postgresql
```

传递给 **S2I** 构建的目录应包含以下一个或多个目录：

postgresql-pre-start/	在容器的早期启动期间，提供此目录中的所有 ***.sh** 文件。后台没有运行 PostgreSQL 守护进程。
postgresql-cfg/	包含的配置文件(***.conf**)将包含在镜像的 **postgresql.conf** 文件的末尾。
postgresql-init/	当数据库被新初始化时，包含 shell 脚本(***.sh**)会被提供（成功 **initdb** 运行之后，会导致数据目录非空）。在提供这些脚本时，本地 PostgreSQL 服务器正在运行。对于带有持久性数据目录的重新部署场景，脚本不会被提供(no-op)。
postgresql-start/	与 **postgresql-init/** 类似，但这些脚本始终来源（在 **postgresql-init/** 脚本存在后）。

在 S2I 构建期间，提供的所有文件都复制到新镜像的 /opt/app-root/src/ 目录中。只有具有相同名称的文件可以进行自定义，而用户提供的文件优先于 /usr/share/container-scripts/ 目录中的默认文件，因此可以覆盖它们。

14.4. REDIS

14.4.1. 描述

rhscl/redis-6-rhel7 镜像提供 Redis 6，这是高级键值存储。

14.4.2. 权限

要拉取 rhscl/redis-6-rhel7 镜像，以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/redis-6-rhel7
```

14.4.3. 配置和使用

要只设置强制环境变量，而不将数据库存储在主机目录中，请运行：

```
# podman run -d --name redis_database -p 6379:6379 rhscl/redis-6-rhel7
```

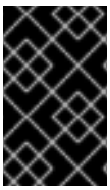
此命令将创建名为 **redis_database** 的容器。端口 **6379** 已公开并映射到主机。

以下环境变量会影响 **Redis** 配置文件且是可选的：

变量名称	描述
REDIS_PASSWORD	服务器访问的密码

要设置密码，请运行：

```
# podman run -d --name redis_database -e REDIS_PASSWORD=strongpassword rhscl/redis-6-rhel7
```

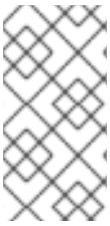


重要

使用一个非常强的密码，因为 **Redis** 是快速的，因此可能会成为暴力攻击的目标。

要使数据库在容器执行过程中持久保留，请在 **podman run** 命令中添加 **-v /host/db/path:/var/lib/redis/data:Z** 选项。

卷挂载点	描述
/var/lib/redis/data	Redis 数据目录



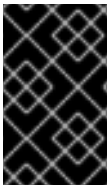
注意

将目录从主机挂载到容器时，请确保挂载的目录具有适当的权限，并且目录的所有者和组与容器中运行的用户 **UID** 或名称匹配。

要检查容器镜像日志，请使用 **podman logs <image_name>** 命令。

第 15 章 RED HAT DEVELOPER TOOLSET IMAGES

Red Hat Developer Toolset 是 Red Hat Enterprise Linux 平台上的开发人员提供的红帽产品。它提供了一套完整的开发和性能分析工具，可在多个 Red Hat Enterprise Linux 版本上安装和使用。然后，Red Hat Developer Toolset 工具链构建的可执行文件也可以在多个 Red Hat Enterprise Linux 版本上部署并运行。有关详细兼容性信息，请参阅 Red Hat Developer Toolset 12 用户指南。



重要

仅支持提供最新版本的 Red Hat Developer Toolset 的容器镜像。

15.1. 从预构建的容器镜像运行红帽开发人员工具集工具

要显示已拉取到本地机器的预构建 Red Hat Developer Toolset 容器镜像的常规使用信息，以 root 用户身份运行以下命令：

```
# podman run image_name usage
```

要在预构建的容器镜像中启动交互式 shell，以 root 用户身份运行以下命令：

```
# podman run -ti image_name /bin/bash -l
```

在以上两个命令中，将 *image_name* 参数替换为您拉取到本地系统的容器镜像名称，现在想要使用。

例如，要使用所选工具链组件在容器镜像中启动交互式 shell，以 root 用户身份运行以下命令：

```
# podman run -ti rhscl/devtoolset-12-toolchain-rhel7 /bin/bash -l
```

例 15.1. 在 Pre-Built Red Hat Developer Toolset Toolchain Image 中使用 GCC

本例演示如何使用 Red Hat Developer Toolset 的所选工具链组件获取并启动预构建的容器镜像，以及如何在该镜像中运行 gcc 编译器。

1. 在使用 podman 的 *管理容器* 文档中的说明，确保在您的系统上正确设置了容器环境。
2. 从官方 Red Hat Container Registry 中拉取预构建的工具链 Red Hat Developer

Toolset 容器镜像：

```
# podman pull rhscl/devtoolset-12-toolchain-rhel7
```

3. **要使用互动 shell 启动容器镜像，请运行以下命令：**

```
# podman run -ti rhscl/devtoolset-12-toolchain-rhel7 /bin/bash -l
```

4. **要以常规（非 root）用户的身份启动容器，请使用 sudo 命令。要将主机系统的目录映射到容器文件系统，请在 podman 命令中包含 -v （或 --volume）选项：**

```
$ sudo podman run -v ~/Source:/src -ti rhscl/devtoolset-12-toolchain-rhel7 /bin/bash -l
```

在以上命令中，主机的 ~/Source/ 目录作为容器内的 /src/ 目录挂载。

5. **旦您在容器的互动 shell 中，您可以如预期运行 Red Hat Developer Toolset 工具。例如，要验证 gcc 编译器的版本，请运行：**

```
bash-4.2$ gcc -v
[...]
gcc version 12.2.1 20221121 (Red Hat 12.2.1-4) (GCC)
```

其它资源

有关 Red Hat Developer Toolset 中可用的组件的更多信息，请参阅以下在线资源：

- Red Hat Developer Toolset 12 用户指南
- Red Hat Developer Toolset 12.1 发行注记
- Red Hat Developer Toolset 12.0 Release Notes

15.2. RED HAT DEVELOPER TOOLSET TOOLCHAIN CONTAINER IMAGE

15.2.1. 描述

Red Hat Developer Toolset Toolchain 镜像提供 GNU Compiler Collection (GCC)和 GNU Debugger (GDB)。

rhscl/devtoolset-12-toolchain-rhel7 镜像包含与以下软件包对应的内容：

组件	版本	软件包
gcc	12.2.1	devtoolset-12-gcc
g++		devtoolset-12-gcc-c++
gfortran		devtoolset-12-gcc-gfortran
gdb	11.2	devtoolset-12-gdb

此外，devtoolset-12-binutils 软件包也作为依赖项包含在内。

15.2.2. 权限

要拉取 rhscl/devtoolset-12-toolchain-rhel7 镜像，请以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/devtoolset-12-toolchain-rhel7
```

15.3. RED HAT DEVELOPER TOOLSET PERFORMANCE TOOLS CONTAINER IMAGE

15.3.1. 描述

Red Hat Developer Toolset Performance Tools 镜像提供了很多性能分析和性能测量工具。

rhscl/devtoolset-12-perftools-rhel7 镜像包括以下组件：

组件	版本	软件包
dwz	0.14	devtoolset-12-dwz
Dyninst	12.1.0	devtoolset-12-dyninst

组件	版本	软件包
elfutils	0.187	devtoolset-12-elfutils
ltrace	0.7.91	devtoolset-12-ltrace
make	4.3	devtoolset-12-make
memstomp	0.1.5	devtoolset-12-memstomp
OProfile	1.4.0	devtoolset-12-oprofile
strace	5.18	devtoolset-12-strace
SystemTap	4.7	devtoolset-12-systemtap
Valgrind	3.19.0	devtoolset-12-valgrind

此外，devtoolset-12-gcc 和 devtoolset-12-binutils 软件包也作为依赖项包含在内。

15.3.2. 权限

要拉取 rhscl/devtoolset-12-perftools-rhel7 镜像，请以 root 用户身份运行以下命令：

```
# podman pull registry.redhat.io/rhscl/devtoolset-12-perftools-rhel7
```

15.3.3. 使用

从容器镜像使用 SystemTap 工具

从容器镜像中使用 SystemTap 工具时，需要额外的配置，并且容器需要使用特殊的命令行选项运行。

需要满足以下条件：

1. **需要使用超级用户权限运行该镜像。要做到这一点，请使用以下命令运行镜像：**

```
~]$ podman run --ti --privileged --ipc=host --net=host --pid=host devtoolset-12-my-perftools
/bin/bash -l
```

要使用预构建的 **perftools** 镜像，请在上述命令中替换 **devtoolset-12-perftools-rhel7** 的镜像名称。

2. 需要在容器中安装以下内核软件包：

- **kernel**
- **kernel-devel**
- **kernel-debuginfo**

以上软件包的版本和发行版本号必须与主机系统上运行的内核的版本和版本号匹配。运行以下命令确定主机系统内核的版本和发行版本号：

```
~]$ uname -r
3.10.0-1160.90.1.el7.x86_64
```

请注意，**kernel-debuginfo** 软件包只能从 ***Debug*** 存储库中获得。启用 **rhel-7-server-debug-rpms** 存储库。有关如何访问 **debuginfo** 软件包的更多信息，请参阅 **如何为 RHEL 系统下载或安装 debuginfo 软件包？**

要使用正确的版本安装所需的软件包，请使用 **yum** 软件包管理器和 **uname** 命令的输出。例如，要安装正确的 **kernel** 软件包版本，以 **root** 用户身份运行以下命令：

```
~]# yum install -y kernel-$(uname -r)
```

3. 通过执行 **podman commit** 命令，将容器保存到可重复使用的镜像。保存自定义构建的 **SystemTap** 容器：

```
~]$ podman commit devtoolset-12-systemtap-$(uname -r)
```

第 16 章 编译器工具集镜像

Red Hat Developer Tools 容器镜像可用于 AMD64 和 Intel 64、64 位 IBM Z 和 IBM POWER，little endian 构架用于以下编译器工具集：

- clang 和 LLVM Toolset
- Rust Toolset
- Go Toolset

详情请查看 Red Hat Developer Tools 文档。

第 17 章 修订历史记录

版本	Date	更改	作者
0.2-7	2023 年 12 月 20 日	删除了过时的链接。	Lenka Špačková
0.2-6	2023 年 7 月 3 日	**rhscl/mariadb-103-rhel7** 容器镜像是 EOL。	Lenka Špačková
0.2-5	2023 年 5 月 23 日	通过 Red Hat Developer Toolset 12.1 发布来更新。	Lenka Špačková
0.2-4	2022 年 11 月 22 日	通过 Red Hat Developer Toolset 12.0 发行版本进行更新。	Lenka Špačková
0.2-3	2021 年 11 月 15 日	使用 Red Hat Software Collections 3.8 容器镜像发行之日。	Lenka Špačková
0.2-2	2021 年 10 月 11 日	使用 Red Hat Software Collections 3.8 Beta 容器镜像发布。	Lenka Špačková
0.2-1	2021 年 6 月 03 日	使用红帽软件集合 3.7 容器镜像发布.	Lenka Špačková
0.2-0	2021 年 5 月 03 日	使用 Red Hat Software Collections 3.7 Beta 容器镜像发布.	Lenka Špačková
0.1-9	2021 年 4 月	改进了支持的构架。	Lenka Špačková
0.1-8	2021 年 1 月 13 日	改进了简介章节，以及有关构建应用程序镜像的扩展信息。	Lenka Špačková
0.1-7	Dec 01 2020	使用红帽软件集合 3.6 容器镜像发布.	Lenka Špačková
0.1-6	2020 年 10 月 29 日	使用 Red Hat Software Collections 3.6 Beta 容器镜像发布.	Lenka Špačková
0.1-5	2020 年 5 月 26 日	使用红帽软件集合 3.5 容器镜像发布.	Lenka Špačková
0.1-4	2020 年 4 月 21 日	使用红帽软件集合 3.5 Beta 容器镜像发布.	Lenka Špačková
0.1-3	2019 年 12 月 10 日	使用红帽软件集合 3.4 容器镜像发布.	Lenka Špačková
0.1-2	2019 年 11 月	使用红帽软件集合 3.4 Beta 容器镜像发布.	Lenka Špačková

版本	Date	更改	作者
0.1-1	2019 年 6 月 11 日	使用红帽软件集合 3.3 容器镜像发布.	Lenka Špačková
0.1-0	2019 年 4 月 16 日	使用红帽软件集合 3.3 Beta 容器镜像发布.	Lenka Špačková
0.0-9	2018 年 11 月 13 日	使用红帽软件集合 3.2 容器镜像发布.	Lenka Špačková
0.0-8	2018 年 10 月 23 日	使用 Red Hat Software Collections 3.2 Beta 的容器镜像发布。	Lenka Špačková
0.0-8	2018 年 8 月 29 日	添加了与 devtoolset-6-perftools 中的 SystemTap 相关的已知问题。	Lenka Špačková
0.0-7	2018 年 5 月 10 日	扩展 MongoDB 镜像文档。	Lenka Špačková
0.0-6	2018 年 5 月 03 日	使用红帽软件集合 3.1 容器镜像发布.	Lenka Špačková
0.0-5	2018 年 4 月 04 2018	使用红帽软件集合 3.1 试用版容器镜像发布.	Lenka Špačková
0.0-3	2017 年 11 月 29 日	添加了扩展现有容器镜像部分。	Lenka Špačková
0.0-2	2017 年 10 月 24 日	使用红帽软件集合 3.0 容器镜像发布.	Lenka Špačková
0.0-1	2017 年 10 月 03	使用红帽软件集合 3.0 试用版容器镜像发布.	Lenka Špačková