



Red Hat Trusted Application Pipeline 1.0

自定义红帽受信任的应用程序管道

了解如何自定义默认软件模板和构建管道配置。

Red Hat Trusted Application Pipeline 1.0 自定义红帽受信任的应用程序管道

了解如何自定义默认软件模板和构建管道配置。

法律通告

Copyright © 2024 Red Hat, Inc.

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

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

摘要

本文档提供了有关集群管理员自定义 RHTAP 的软件模板和构建管道配置的说明，以更好地与 on-prem 环境的唯一要求保持一致。通过自定义这些元素，管理员可以确保开发工作流程简化、安全实践无缝集成，开发人员可专注于创新，而非安全合规和基础架构方面的差别。

目录

前言	3
第 1 章 自定义示例软件模板	4
第 2 章 自定义管道示例	8
自定义示例模板存储库以更新 pac URL*	8
为工作流自定义示例管道存储库	8
第 3 章 为自动管道触发器配置 GITLAB WEBHOOK	10

前言

RHTAP 赋予团队随时可用的软件模板和构建管道配置，旨在将安全实践无缝集成到您的开发过程中。这些工具不仅降低了开发人员的安全考虑因素，还有助于专注于创新。

集群管理员在定制这些资源方面扮演了一个关键角色，以满足其预备环境的唯一要求，包括：

- 自定义软件模板以满足特定机构需求
- 修改构建管道配置，使其与项目目标一致
- 为自动管道触发器配置 GitLab Webhook

这种自定义简化了开发工作流，解决与管道、漏洞和策略合规性相关的常见问题，从而使开发人员能够优先进行编码。

第 1 章 自定义示例软件模板

了解如何为您的 on-prem 环境自定义随时可用的软件模板。集群管理员对此过程有完全的控制权，包括修改元数据和规格。

先决条件

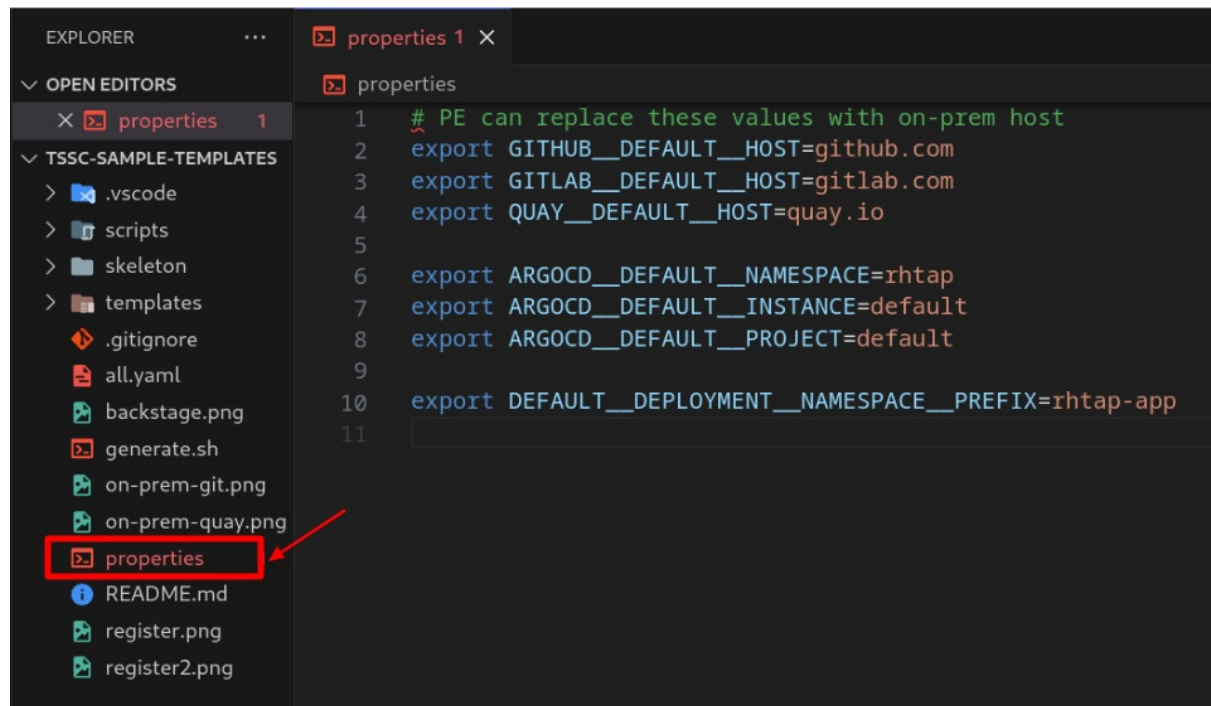
- 您已在 RHTAP 安装过程中使用了 `tssc-sample-templates` 中的 fork 仓库 URL。

步骤

- 克隆已分叉的存储库，然后在您首选的文本编辑器中打开它，如 Visual Studio Code。
- 在项目目录中找到 **属性文件**。此文件存储可自定义的默认值。打开它进行编辑，并根据您的环境更新以下键值对。

键	描述
<code>export GITHUB_DEFAULT_HOST</code>	把它设置为您的 on-prem GitHub 主机完全限定域名。也就是说，没有 HTTP 协议且没有 .git 扩展的 URL。例如 <code>github-github.apps.cluster-ljg9z.sandbox219.opentlc.com</code> 。默认为 github.com 。
<code>export GITLAB_DEFAULT_HOST</code>	把它设置为您的 on-prem GitLab 主机完全限定域名。也就是说，没有 HTTP 协议且没有 .git 扩展的 URL。例如 <code>gitlab-gitlab.apps.cluster-ljg9z.sandbox219.opentlc.com</code> 。默认为 gitlab.com 。
<code>export QUAY_DEFAULT_HOST</code>	默认 Quay URL 对应于没有 HTTP 协议的特定 on-prem 镜像 registry URL。例如： <code>quay-tv2pb.apps.cluster-tv2pb.sandbox1194.opentlc.com</code> 。默认 quay 主机为 quay.io 。
<code>export DEFAULT_DEPLOYMENT_NAMESPACE_PREFIX</code>	<p>RHTAP 中部署的命名空间前缀。默认为 rhtap-app。</p> <div style="display: flex; align-items: center;">  <div> <p>注意</p> <p>如果您在 RHTAP 安装过程中修改了默认的 trusted-application-pipeline 命名空间，请更新它。</p> </div> </div>

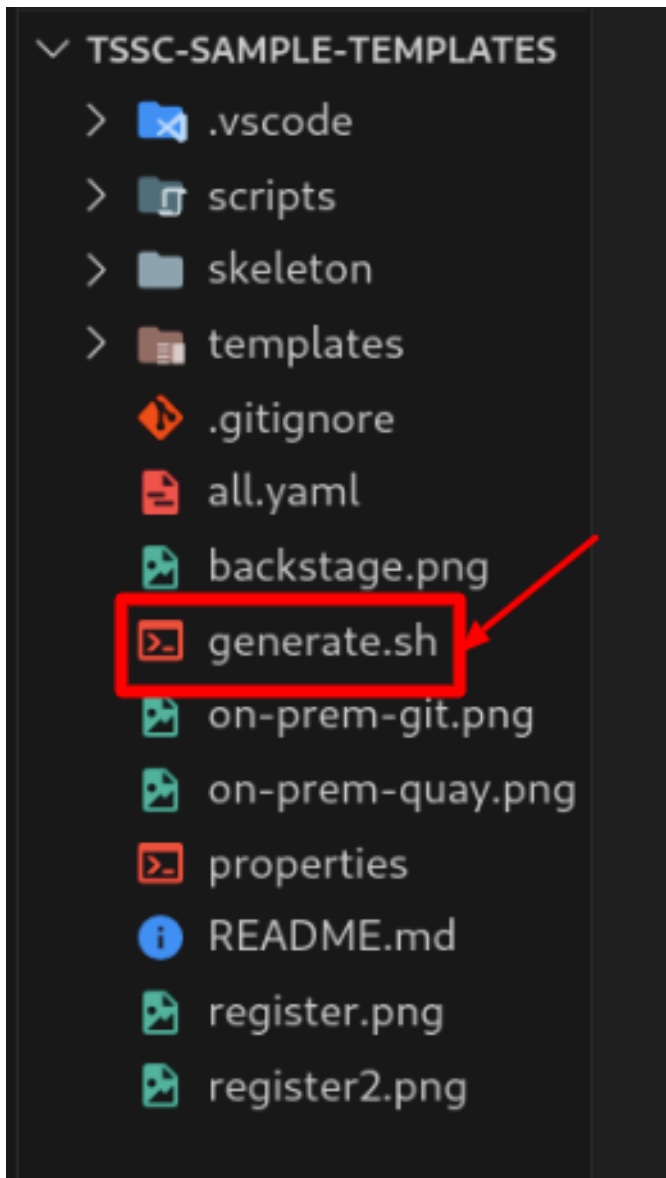
图 1.1. 属性文件



3. 在终端中运行 `generate.sh` 脚本。此操作会调整软件模板，使用您指定的输入替换默认主机值。

```
./generate.sh
```

图 1.2. generate.sh 脚本



4. 提交更改并将其推送到您的存储库。这会更新 RHDH 中的模板。或者，您可以在 RHDH 中直接导入并刷新单个或所有自定义模板。
 - a. 进入 Git 供应商上的 fork 示例模板存储库。
 - b. 对于单个模板，从 **templates** 目录中选择 **template.yaml**。从浏览器地址栏复制其 URL。例如：<https://github.com/<username>/tssc-sample-templates/blob/main/templates/devfile-sample-code-with-quarkus-dance/template.yaml>。否则，对于所有模板，请选择 **all.yaml** 并从浏览器地址栏中复制其 URL。例如：<https://github.com/<username>/tssc-sample-templates/blob/main/all.yaml>。
 - c. 切换回 RHDH 平台。
 - d. 选择 **Create > Register Existing Component**。
 - e. 在 **Select URL** 字段中，粘贴在第 4b 中复制的适当 URL。
 - f. 选择 **analyze e**，然后选择 **Import** 以更新 RHDH 中的模板。

验证

- 考虑创建一个应用程序来探索模板自定义的影响。

其他资源

- 要自定义管道，[请参阅自定义管道模板](#)

第 2 章 自定义管道示例

了解如何在示例模板存储库中更新 Pipeline 作为代码(**pac**) URL，并将示例管道存储库自定义工作流。通过自定义 **pac** URL，组织可以利用根据需求量身定制的特定管道。

先决条件

- 您已在本地分叉并克隆了以下模板：
 - [管道示例](#)
 - [模板示例](#)

自定义示例模板存储库以更新 **pac** URL *

步骤

1. 访问分叉的管道存储库 URL：
 - a. 打开已分叉的示例管道存储库。
 - b. 复制地址栏中的完整 URL。例如，<https://github.com/<username>/tssc-sample-pipelines>。
2. 在示例模板存储库中更新 **pac** URL
 - a. 使用您的终端导航到本地克隆的示例模板存储库。
 - b. 运行以下命令，将 {fork_url} 替换为从第 1 步中复制的 URL，将 {branch_name} 替换为您所需的分支名称（例如 main）：

```
./scripts/update-tekton-definition {fork_url} {branch_name}

# For example, ./scripts/update-tekton-definition https://github.com/<username>/tssc-sample-pipelines main
```

3. 检查、提交和推送更改：
 - a. 查看示例模板存储库中的更新文件。
 - b. 将更改与适当的消息一起提交。
 - c. 将提交的更改推送到您的已分叉的存储库。

为工作流自定义示例管道存储库

示例管道存储库提供了一个基础，您可以构建组织的特定 CI/CD 工作流。管道存储库示例在 **pac** 目录中包含多个关键管道模板：

- **GITOPS-repo**：此目录包含用于验证 GitOps 存储库中的拉取请求的管道定义。它会触发 **gitops-pull-request** 管道（位于 **pipelines** 目录中），验证镜像更新符合机构标准。这个设置对于提升工作流至关重要，其中应用程序的部署状态是高级的，如从开发到暂存或从暂存到生产等。有关 **gitops-repo** 中的管道定义的更多信息，请参阅 [Gitops Pipelines](#)。
- **管道**：此目录包含 **gitops-repo** 和 **source-repo** 中事件处理程序引用的构建和验证管道的实现。通过检查此目录的内容，您可以了解管道执行的特定操作，包括它们如何贡献应用程序的安全提升和部署。

- **source-repo** : 此目录侧重于基于 Dockerfile 的安全供应链软件构建。它包括用于克隆源的管道定义、生成和签名工件（如用于镜像签名的 **.sig**）、**.att** 用于 attestation，以及 **.sbom** 用于软件 Bill 的 Materials，并将它们推送到用户的镜像 registry。如需有关 **source-repo** 中的管道定义的更多信息，请参阅 [共享管道和任务的共享 Git 解析器模型](#)。
- **任务** : 此目录包含可添加或修改的任务集合，与组织需求保持一致。例如，高级 Cluster Security (ACS) 任务可以替换为替代检查，或者完全新的任务可以集成到管道中，以增强其功能和合规性。

验证

- 考虑创建一个应用程序，以探索模板和管道自定义的影响。

其他资源

- 要自定义模板，请参阅 [自定义示例软件模板](#)
- 有关 Pipeline 作为代码的详情，请参考 [Pipelines as Code](#)。

第 3 章 为自动管道触发器配置 GITLAB WEBHOOK

了解如何在 GitLab 中设置 webhook 和 secret，以便在代码更新时自动触发管道运行。

先决条件

- 您有一个现有的 GitLab 项目。
- 在 OpenShift Web 控制台中 [具有管理员特权](#)。

步骤

1. 检索 Webhook URL 和 Secret 令牌：

- a. 使用管理员特权登录 [OpenShift Web 控制台](#)。
- b. 导航到 `rhtap` 项目，展开 **Pipelines**，然后选择 **PipelineRuns**。
- c. 找到 `rhtap-pe-info-<it>` 管道运行，然后选择 **Logs** 选项卡。



注意

这些日志包含 GitLab 配置所需的 webhook URL 和 secret 令牌。

2. 在 GitLab 中配置 Webhook：

- a. 在 GitLab 仓库中，进入到 **Settings > Webhooks**。
- b. 在 **URL** 字段中，输入从 Step 1 复制的 Webhook URL。
- c. 在 **Secret Token** 字段中，输入从 Step 1 复制的 secret 令牌。
- d. 在 **Trigger** 部分：
 - i. 选择 **Push events**。
 - ii. 选择 **Merge 请求事件**。
- e. 单击 **Add Webhook**。

验证

1. 将您的代码更改推送到 GitLab 存储库。
2. 导航到 RHDH 中的 **CI** 选项卡。
3. 验证您的代码推送是否触发了管道运行。

更新于 2024-07-02

