Red Hat OpenShift Pipelines 1.18

发行注记

OpenShift Pipelines 发行版本的主要新功能及变化信息

Last Updated: 2025-11-27

Red Hat OpenShift Pipelines 1.18 发行注记

OpenShift Pipelines 发行版本的主要新功能及变化信息

Legal Notice

Abstract

此发行注记介绍了 OpenShift Pipelines 的新功能、功能增强、重要的技术变化、以及对以前版本中的错误作出的主要修正。另外，还包括在此版本正式发行(GA)时存在的已知问题的信息。

Table of Contents

第 1 章 RED HAT OPENSHIFT PIPELINES 发行注记

注意

如需有关 OpenShift Pipelines 生命周期和支持的平台的更多信息，请参阅 OpenShift Operator 生命周期和 Red Hat OpenShift Container Platform 生命周期政策 。

发行注记包含有关新的和已弃用的功能、破坏更改以及已知问题的信息。以下发行注记适用于 OpenShift Container Platform 的最新 OpenShift Pipelines 版本。

Red Hat OpenShift Pipelines 是基于 Tekton 项目的一个云原生 CI/CD 环境，它提供：

- 标准 Kubernetes 原生管道定义 (CRD)。
- 无需 CI 服务器管理开销的无服务器管道。
- 使用任何 Kubernetes 工具（如 S2I、Buildah、JIB 和 Kaniko）构建镜像。
- 不同 Kubernetes 发布系统间的可移植性。
- 用于与管道交互的强大 CLI。
- 使用 OpenShift Container Platform Web 控制台的 **Developer** 视角集成用户体验。

如需了解 Red Hat OpenShift Pipelines 的概述，请参阅了解 OpenShift Pipelines。

1.1. 兼容性和支持列表

这个版本中的一些功能当前还只是一个技术预览。它们并不适用于在生产环境中使用。

在下表中，被标记为以下状态的功能：

TP	技术预览
GA	公开发行

表 1.1. 兼容性和支持列表

Red Hat OpenShift Pipelines 版本	组件版本	OpenShift 版本	支持状态

Red Hat OpenShift Pipelines 版本	组件版本								OpenShift 版本	支持状态
Operator	Pipelines	触发器	CLI	链	Hub	Pipelines 作为代码（Pipelines as Code）	结果	Manual Approval Gate		
1.18	0.68.x	0.31.x	0.40.x	0.24.x (GA)	1.20.x (TP)	0.33.x (GA)	0.14.x (GA)	0.5.x (TP)	4.15, 4.16, 4.17, 4.18	GA
1.17	0.65.x	0.30.x	0.39.x	0.23.x (GA)	1.19.x (TP)	0.29.x (GA)	0.13.x (TP)	0.4.x (TP)	4.15, 4.16, 4.17, 4.18	GA
1.16	0.62.x	0.29.x	0.38.x	0.22.x (GA)	1.18.x (TP)	0.28.x (GA)	0.12.x (TP)	0.3.x (TP)	4.15, 4.16, 4.17, 4.18	GA
1.15	0.59.x	0.27.x	0.37.x	0.20.x (GA)	1.17.x (TP)	0.27.x (GA)	0.10.x (TP)	0.2.x (TP)	4.14, 4.15, 4.16	GA
1.14	0.56.x	0.26.x	0.35.x	0.20.x (GA)	1.16.x (TP)	0.24.x (GA)	0.9.x (TP)	不适用	4.12, 4.13, 4.14, 4.15, 4.16	GA

如果您有疑问或希望提供反馈信息，请向产品团队发送邮件 pipelines-interest@redhat.com。

1.2. RED HAT OPENSHIFT PIPELINES 1.18 发行注记

在这个版本中，Red Hat OpenShift Pipelines 正式发行(GA) 1.18 包括在 OpenShift Container Platform 4.15 及更新的版本中。

1.2.1. 新功能

除了包括修复和稳定性改进的信息外，以下小节突出介绍了 Red Hat OpenShift Pipelines 1.18 中的新内容：

1.2.1.1. Pipelines

- 在这个版本中，OpenShift Pipelines 控制器的日志消息已被改进，以增强可读性，特别是在存在空变量时。
- 在这个版本中，您可以为管道解析器配置解析超时设置，以便在运行管道时获得更大的灵活性和控制。

  为管道解析器配置超时设置示例

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
# ...
  pipeline:
    options:
      configMaps:
        config-defaults:
          data:
            default-maximum-resolution-timeout: 5m (1)
        bundleresolver-config:
          data:
            fetch-timeout: 1m (2)
# ...
```

(1) 为解析请求指定一个全局最大超时。默认值为 **1m**。

(2) 指定用于捆绑解析请求的超时。

1.2.1.2. Operator

- 在这个版本中，OpenShift Pipelines 支持社区任务。
  在 **openshift-pipelines** 命名空间中默认安装以下社区任务：
  - **argocd-task-sync-and-wait**
  - **git-cli**
  - **helm-upgrade-from-repo**
  - **helm-upgrade-from-source**
  - **jib-maven**
  - **kubeconfig-creator**
  - **pull-request**

- **trigger-jenkins-job**

- 在这个版本中，OpenShift Pipelines 支持通过 **TektonConfig** CR 部署新容器。

使用 TektonConfig CR 部署新的 kube-rbac-proxy 容器的示例

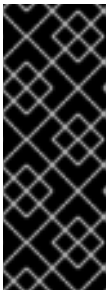
```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
# ...
spec:
  result:
    options:
      deployments:
        tekton-results-watcher:
          spec:
            template:
              spec:
                containers:
                - name: kube-rbac-proxy
                  args:
                    - --secure-listen-address=0.0.0.0:8443
                    - --upstream=http://127.0.0.1:9090/
                    - --logtostderr=true
                  image: registry.redhat.io/openshift4/ose-kube-rbac-proxy:v4.12
# ...
```

- 在这个版本中，OpenShift Pipelines 控制器的高可用性选项会增强 **StatefulSet** 或dinals，作为现有领导选举机制的替代选择。这可让 Operator 使用带有领导选举机制的快速恢复，或与 **StatefulSet** ordinals 一起一致的工作负载分布。
领导选举（默认选项）提供了故障切换功能，但可能会导致热响应。相反，新的 **StatefulSet** 或 dinals 方法可确保密钥在副本中平均分布，以获得更均衡的工作负载分布。

  您可以通过将 **TektonConfig** 自定义资源中的 **statefulset-ordinals** 参数设置为 **true** 来切换到 **StatefulSet** ordinals 方法：

启用 StatefulSet ordinals 功能示例

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
# ...
  pipeline:
    performance:
      disable-ha: false
      buckets: 4
      replicas: 4
      statefulset-ordinals: true
# ...
```



重要

对高可用性使用 **StatefulSet** 或dinals 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

1.2.1.3. 触发器

- 在这个版本中，OpenShift Pipelines 触发器的 **EventListener** 对象包括 **ImagePullSecrets** 字段，以指定监听器用于从私有 registry 中拉取镜像的 secret。

  使用 ImagePullSecrets 字段的示例

```
apiVersion: triggers.tekton.dev/v1beta1
kind: EventListener
metadata:
  name: imagepullsecrets-example
# ...
spec:
  serviceAccountName: triggers-example
  resources:
    kubernetesResource:
      spec:
        template:
          spec:
            imagePullSecrets:
              - name: docker-login
# ...
```

1.2.1.4. CLI

- 在这个版本中，**op** c 命令行工具会随以下组件一起提供：
  - Pipelines as Code 版本 0.33.0
  - CLI 版本 0.40.0
  - 结果版本 0.14.0
  - Manual Approval Gate 版本 0.5.0

1.2.1.5. Pipelines 作为代码（Pipelines as Code）

- 在这个版本中，**PipelineRun** 资源可以根据使用 on-path-change 和 **on-path-change -ignore** 注解更改的文件来触发，这简化了不需要编写复杂的 CEL 表达式的过程。

  使用 on-path-change 和 on-path-change-ignore 注解的示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "on-path-change"
```

```
  annotations:
    pipelinesascode.tekton.dev/on-path-change: ["docs/**"] 1
    pipelinesascode.tekton.dev/on-path-change-ignore: [".github/**"] 2
spec:
# ...
```

1 如果在 **docs** 目录中进行了更改，则匹配。

2 如果 **.github** 目录中没有发生任何更改，则匹配。

- 在这个版本中，可以使用 **on-comment** 注解在 GitHub 上推送提交来触发 **PipelineRun** 资源。此功能可让您在触发 **PipelineRun** 时更好地控制它，使您能够根据特定的注释匹配。
  您必须配置管道运行，以使用 **on-comment** 注解启用触发器：

  通过推送的提交启用管道运行的示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "on-comment-pr"
  annotations:
    pipelinesascode.tekton.dev/on-comment: "^/hello-world" 1
spec:
# ...
```

1 示例配置 **/hello-world** 命令。当您在推送的提交中使用此命令时，例如在 **main** 分支上，将触发 **on-comment-pr** 管道运行。

- 在这个版本中，您可以使用 **tkn pac info globbing [-s | -d] "<pattern >"** 命令测试带有示例输入数据或运行命令的工作目录中的模式。命令测试模式是否在 **PipelineRun** 定义中正常工作。
  例如，您可以测试以下 **on-target-branch** 注解，以确保它与 git 事件有效负载中的 **main** 分支匹配：

  on-target-branch 注解示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "on-target-branch"
  annotations:
    pipelinesascode.tekton.dev/on-target-branch: "[refs/heads/*]"
spec:
# ...
```

  示例命令

```
$ tkn pac info globbing -s "refs/heads/main" "refs/heads/*"
```

- 在这个版本中，Pipelines as Code 支持使用 HTML 实体 **在注解中** 逗号。现在，您可以将用逗号分开的值与值中直接使用的逗号一起使用。
  例如，要匹配名为 **main** 和 **branchWith,comma** 的两个分支，请按如下所示指定注解：

带有逗号的注解示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "comma-annotation"
  annotations:
    pipelinesascode.tekton.dev/on-target-branch: "[main, branchWith&#44;comma]"
spec:
# ...
```

- 在这个版本中，在拉取请求关闭或合并后，所有带有 **cancel-in-progress: true** 注解的关联 **PipelineRun** 资源都会被自动取消。要启用此功能，请在 **PipelineRun** 定义中指定 **cancel-in-progress** 注解：

  cancel-in-progress 注解示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "cancel-in-progress"
  annotations:
    pipelinesascode.tekton.dev/cancel-in-progress: "true"
spec:
# ...
```

重要

Pipelines 中的 cancellation-in-progress 作为代码只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

- 在这个版本中，当 Pipelines as Code 触发名称的新 **PipelineRun** 资源时，带有 **cancel-in-progress: true** 注解的活跃的 **PipelineRun** 资源会自动取消。现在，在引发拉取请求并触发 **PipelineRun** 后，对同一拉取请求的后续提交会触发新的 **PipelineRun**，旧的过时的 **PipelineRun** 会取消保存资源。
  要启用这个功能，在 **PipelineRun** 资源中将 **pipelinesascode.tekton.dev/cancel-in-progress** 注解设置为 **true** ：

  启用取消旧的管道运行的示例

```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: "cancel-in-progress"
  annotations:
    pipelinesascode.tekton.dev/cancel-in-progress: "true"
spec:
# ...
```

- 在此次更新之前，与 Bitbucket Data Center 的 CEL 表达式中使用的 **.pathChanged （）** 函数无法正常工作。在这个版本中，这个功能会被实现。

- 在这个版本中，您可以使用标签与管道运行匹配来拉取请求。要将管道运行与标签匹配，请在 **PipelineRun** 资源中设置 **pipelinesascode.tekton.dev/on-label** 注解。在拉取请求或推送事件中添加标签会立即触发管道运行。如果推送请求或拉取请求仍然具有标签并使用提交更新，则管道运行会再次触发。
  GitHub、GitLab 和 Gitea 存储库供应商支持此功能。Bitbucket 云和 Bitbucket 数据中心不支持它。

1.2.1.6. Tekton Results

- 在这个版本中，Tekton Results 是一个正式发行(GA)功能。

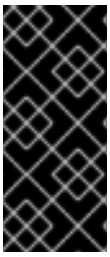
- 在这个版本中，Operator 默认通过 **TektonConfig** CR 安装并添加 Tekton 结果的配置。

- 在这个版本中，Tekton Results API 服务器使用 OpenShift Pipelines 控制台插件中的代理环境变量将授权标头发送到 Tekton Results API。

- 在这个版本中，从 LokiStack 获取的 Tekton Results 日志记录信息包括每个日志条目的容器名称。

1.2.1.7. Tekton Cache

在这个版本中，OpenShift Pipelines 包含新的 **tekton-caches** 工具功能。



重要

Tekton Cache 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

tekton-caches 工具包括以下功能：

- 使用 **cache-upload** 和 **cache-fetch** 步骤操作，保留构建过程保留其依赖项的缓存目录，将其存储在 S3 存储桶、Google Cloud Services (GCS)存储桶或 OCI 存储库。
  cache-fetch 和 **cache-upload** 步骤操作默认安装在 **openshift-pipelines** 命名空间中。

1.2.2. 可能会造成问题的更改

- 在这个版本中，Tekton Results 不再支持由 Tekton Results watcher 将日志转发到持久性卷(PV)、S3 存储桶或 GCS 存储桶中的存储。

- 在此次更新之前，旧版本的任务和步骤操作安装在 **openshift-pipelines** 命名空间中。在这个版本中，这些旧版本会被删除，只安装最新的两个任务和步骤操作。例如，在 OpenShift Pipelines 1.18 中，已安装的版本的任务和步骤操作为 ***-1-17-0** 和 ***-1-18-0**。

1.2.3. 已知问题

- 如果您在 **TektonConfig** CR 的 **result:** 部分添加或更改任何参数，则更改不会自动应用。要应用这些更改，请在 **openshift-pipelines** 命名空间中重启 Results API 服务器的部署或 pod。

1.2.4. 修复的问题

- 在此次更新之前，如果您定义了包含常规参数和 **列表** 参数的任务，**tekton-pipelines-controller** 组件会崩溃并记录分段错误消息。如果没有删除该任务，则组件将继续崩溃，且不会运行任何管道。在这个版本中，控制器不会在这样的情形中崩溃。

- 在此次更新之前，当用户尝试使用 OpenShift Container Platform Web 控制台重新运行基于解析器的 **PipelineRun** 资源时，尝试会出现 **Invalid PipelineRun 配置出错，无法启动 Pipeline。** 在这个版本中，重新运行不再会导致错误，并可以正常工作。

- 在此次更新之前，当 **PipelineRun** 资源失败时，OpenShift Container Platform Web 控制台中的 **PipelineRun** 资源的 **Output** 选项卡会显示一个错误消息，而不是可用的结果。在这个版本中，Web 控制台可以正确地显示结果。

- 在此次更新之前，当 **CONTEXT** 和 **DOCKERFILE** 参数的值指向不同的目录时，**openshift-pipelines** 命名空间中的 **buildah** 任务会失败。在这个版本中，这个问题已被解决。

- 在此次更新之前，在步骤操作中以默认值的形式引用参数会导致 **TaskRun** 资源引用该步骤操作失败。在这个版本中，这个问题已被解决。

  使用参数作为值的示例

  ```
  apiVersion: tekton.dev/v1alpha1
  kind: StepAction
  metadata:
    name: simple-step-action
  # ...
  spec:
    params:
      - name: param1 # This parameter is required
        type: string
      - name: param2 # This parameter uses the value of `param1` as default value
        type: string
        default: $(params.param1)
  # ...
  ```

- 在此次更新之前，当创建一个 **PipelineRun** 资源来引用一个远程 Git 存储库，其中包含该仓库之外的符号链接，**PipelineRun** 将失败。在这个版本中，即使符号链接无效，**PipelineRun** 不再会失败。

- 在此次更新之前，内存和临时存储的资源请求可能会超过定义的限制。在这个版本中，这个问题已被解决。

- 在此次更新之前，在使用注入的 sidecar 的集群中运行的管道有时会不会影响 OpenShift Pipelines 控制器。这是因为有故障的 Kubernetes 版本检查造成的。在这个版本中，这个问题已被解决。

- 在此次更新之前，当结果列表包含带有无效结果的重复密钥时，重复会被删除并替换为每个键的最后结果，这可能会更改结果列表的顺序。在这个版本中，当删除重复的键时，结果的顺序会被保留，确保每个密钥的最后有效结果都保留为最终结果。

- 在此次更新之前，当使用 Pipelines as Code 时，在相对路径下托管的 GitLab 实例（例如 **https://example.servehttp.com/gitlab** ）无法正确更新合并请求的状态。虽然初始事件更新可以正常工作，但后续更新（例如，将管道运行标记为 **Finished** ）没有出现在 GitLab 中，使状态保持在 **Running** 状态。在这个版本中，状态更新可以正常工作。

- 在此次更新之前，当使用 Pipelines as Code 时，如果您在带有空 **[]** 值的 **PipelineRun** 资源中传递了 **onEvent** 或 **onTargetBranch** 注解时，它会阻止与任何 **PipelineRun** 资源匹配。在这个版本中，使用带有空值的注解会返回错误。

- 在此次更新之前，在使用 Pipelines 作为代码时，您可以创建 **Repository** 自定义资源(CR)而无需 URL 或带有无效的 URL。在这个版本中，只有在提供有效的 URL 时，才能创建 **Repository** CR。您仍然可以在 **openshift-pipelines** 命名空间中创建 **Repository** CR，而无需 URL，为所有存储库提供默认设置。

- 在此次更新之前，当使用 Pipelines as Code 时，如果在 GitLab 中的 fork 仓库引发拉取请求，**PipelineRun** 资源被成功创建，但不会在 GitLab UI 中报告其最终状态。在这个版本中，这个问题已被解决。

- 在此次更新之前，当使用 Pipelines as Code 时，GitLab 会显示检查名称作为管道的单一条目运行。这会导致多个管道运行同时运行，其中一个失败。如果其他管道运行成功，它会覆盖失败的状态，管道将显示为成功。在这个版本中，每个检查名称会为每个管道运行单独显示。因此，如果任何管道运行失败，管道的状态会显示正确失败。

- 在此次更新之前，当使用 Pipelines as Code 时，如果您在 **PipelineRun** 资源定义中使用 **{{ trigger_comment }}** 变量，然后在 GitHub 中提交多行注释，Pipelines as Code 可能会重新发布管道运行的 YAML 验证错误。在这个版本中，这个问题已被解决。

- 在此次更新之前，当使用 Pipelines as Code 时，一个 **/test branch:&** lt;branch> 命令，其中分支名称包含可能因为其他命令无法正常工作的部分。例如，在命令 **/test 分支：a/testbranch** 中，分支名称包含 **/test**，此命令将失败。在这个版本中，注释中的第一个 **/test** 子字符串被视为阻止任何故障的命令。

- 在此次更新之前，当使用 Pipelines as Code 时，如果在 GitHub 中推送提交时，如果一个未授权的用户发送 **/test**、**/retest** 或其他 GitOps 命令，则会触发 **PipelineRun** 资源，即使用户没有授权触发它们。在这个版本中，添加了一个额外的用户授权检查，防止在没有正确验证的情况下触发 **PipelineRun** 资源。

- 在此次更新之前，当使用 Pipelines as Code 时，如果未授权用户引发拉取请求，并且存储库管理员发送 **/ok-to-test** 命令，则会创建一个待处理的检查，即使拉取请求事件没有匹配的 **PipelineRun** 资源。在这个版本中，如果没有匹配的 **PipelineRun** 资源，则不会创建待处理的检查。相反，会创建一个中立检查来描述没有匹配的 **PipelineRun**。

- 在此次更新之前，当将 Pipelines as Code 与 Bitbucket Data Center 搭配使用时，在 push 事件中，用户无法引用 **PipelineRun** 资源中事件有效负载的所有字段，如 **body.changes** 对象。在这个版本中解决了这个问题。

  使用 {{ body.changes }} 动态变量的示例

  ```
  apiVersion: tekton.dev/v1
  kind: PipelineRun
  # ...
  spec:
    params:
      - name: ref_id
        value: "{{ body.changes[0].ref.id }}"
  # ...
  ```

- 在此次更新之前，当使用 Pipelines as Code 时，如果您使用 **PipelineRun** 资源中的 **generateName** 字段来把资源与 **传入的** Webhook 匹配，则匹配无法正常工作。在这个版本中，这个问题已被解决。

- 在此次更新之前，在使用 Pipelines 作为代码时，**on-cel-expression** 注解不适用于 Bitbucket 数据中心中的推送事件。在这个版本中，这个问题已被解决。