



# Red Hat OpenShift Pipelines 1.15

## Installing and configuring

Installing and configuring OpenShift Pipelines



# Red Hat OpenShift Pipelines 1.15 Installing and configuring

---

Installing and configuring OpenShift Pipelines

## Legal Notice

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<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

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

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> 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<sup>®</sup> 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.

## Abstract

This document provides information about installing and configuring OpenShift Pipelines.

---

## Table of Contents

<b>CHAPTER 1. INSTALLING OPENSIFT PIPELINES</b> .....	<b>3</b>
Prerequisites	3
1.1. INSTALLING THE RED HAT OPENSIFT PIPELINES OPERATOR IN WEB CONSOLE	3
1.2. INSTALLING THE OPENSIFT PIPELINES OPERATOR USING THE CLI	6
1.3. RED HAT OPENSIFT PIPELINES OPERATOR IN A RESTRICTED ENVIRONMENT	6
1.4. ADDITIONAL RESOURCES	7
<b>CHAPTER 2. UNINSTALLING OPENSIFT PIPELINES</b> .....	<b>8</b>
2.1. DELETING THE OPENSIFT PIPELINES CUSTOM RESOURCES	8
2.2. UNINSTALLING THE RED HAT OPENSIFT PIPELINES OPERATOR	9
2.3. DELETING THE CUSTOM RESOURCE DEFINITIONS OF THE OPERATOR.TEKTON.DEV GROUP	9
<b>CHAPTER 3. CUSTOMIZING CONFIGURATIONS IN THE TEKTONCONFIG CUSTOM RESOURCE</b> .....	<b>11</b>
3.1. PREREQUISITES	11
3.2. PERFORMANCE TUNING USING TEKTONCONFIG CR	11
3.3. CONFIGURING THE RED HAT OPENSIFT PIPELINES CONTROL PLANE	13
3.3.1. Modifiable fields with default values	14
3.3.2. Optional configuration fields	15
3.4. CHANGING THE DEFAULT SERVICE ACCOUNT FOR OPENSIFT PIPELINES	15
3.5. SETTING LABELS AND ANNOTATIONS FOR THE OPENSIFT PIPELINES INSTALLATION NAMESPACE	16
3.6. DISABLING THE SERVICE MONITOR	16
3.7. CONFIGURING PIPELINE RESOLVERS	17
3.8. DISABLING CLUSTER TASKS AND PIPELINE TEMPLATES	17
3.9. DISABLING THE INTEGRATION OF TEKTON HUB	18
3.10. DISABLING THE AUTOMATIC CREATION OF RBAC RESOURCES	18
3.11. DISABLING INLINE SPECIFICATION OF PIPELINES AND TASKS	19
3.12. AUTOMATIC PRUNING OF TASK RUNS AND PIPELINE RUNS	21
3.12.1. Configuring the pruner	21
3.12.2. Annotations for automatically pruning task runs and pipeline runs	23
3.13. ADDITIONAL RESOURCES	23



# CHAPTER 1. INSTALLING OPENSIFT PIPELINES

This guide walks cluster administrators through the process of installing the Red Hat OpenShift Pipelines Operator to an OpenShift Container Platform cluster.

## Prerequisites

- You have access to an OpenShift Container Platform cluster using an account with **cluster-admin** permissions.
- You have installed **oc** CLI.
- You have installed [OpenShift Pipelines \(tkn\) CLI](#) on your local system.
- Your cluster has the [Marketplace capability](#) enabled or the Red Hat Operator catalog source configured manually.



### NOTE

In a cluster with both Windows and Linux nodes, Red Hat OpenShift Pipelines can run on only Linux nodes.

## 1.1. INSTALLING THE RED HAT OPENSIFT PIPELINES OPERATOR IN WEB CONSOLE

You can install Red Hat OpenShift Pipelines using the Operator listed in the OpenShift Container Platform OperatorHub. When you install the Red Hat OpenShift Pipelines Operator, the custom resources (CRs) required for the pipelines configuration are automatically installed along with the Operator.

The default Operator custom resource definition (CRD) **config.operator.tekton.dev** is now replaced by **tektonconfigs.operator.tekton.dev**. In addition, the Operator provides the following additional CRDs to individually manage OpenShift Pipelines components: **tektonpipelines.operator.tekton.dev**, **tektontriggers.operator.tekton.dev** and **tektonaddons.operator.tekton.dev**.

If you have OpenShift Pipelines already installed on your cluster, the existing installation is seamlessly upgraded. The Operator will replace the instance of **config.operator.tekton.dev** on your cluster with an instance of **tektonconfigs.operator.tekton.dev** and additional objects of the other CRDs as necessary.



### WARNING

If you manually changed your existing installation, such as, changing the target namespace in the **config.operator.tekton.dev** CRD instance by making changes to the **resource name - cluster** field, then the upgrade path is not smooth. In such cases, the recommended workflow is to uninstall your installation and reinstall the Red Hat OpenShift Pipelines Operator.

The Red Hat OpenShift Pipelines Operator now provides the option to choose the components that you want to install by specifying profiles as part of the **TektonConfig** custom resource (CR). The **TektonConfig** CR is automatically installed when the Operator is installed. The supported profiles are:

- Lite: This installs only Tekton Pipelines.
- Basic: This installs Tekton Pipelines, Tekton Triggers, and Tekton Chains.
- All: This is the default profile used when the **TektonConfig** CR is installed. This profile installs all of the Tekton components, including Tekton Pipelines, Tekton Triggers, Tekton Chains, Pipelines as Code, and Tekton Addons. Tekton Addons includes the **ClusterTasks**, **ClusterTriggerBindings**, **ConsoleCLIDownload**, **ConsoleQuickStart**, and **ConsoleYAMLSample** resources, as well as the tasks available using the cluster resolver from the **openshift-pipelines** namespace.

## Procedure

1. In the **Administrator** perspective of the web console, navigate to **Operators** → **OperatorHub**.
2. Use the **Filter by keyword** box to search for **Red Hat OpenShift Pipelines** Operator in the catalog. Click the **Red Hat OpenShift Pipelines** Operator tile.
3. Read the brief description about the Operator on the **Red Hat OpenShift Pipelines** Operator page. Click **Install**.
4. On the **Install Operator** page:
  - a. Select **All namespaces on the cluster (default)** for the **Installation Mode**. This mode installs the Operator in the default **openshift-operators** namespace, which enables the Operator to watch and be made available to all namespaces in the cluster.
  - b. Select **Automatic** for the **Approval Strategy**. This ensures that the future upgrades to the Operator are handled automatically by the Operator Lifecycle Manager (OLM). If you select the **Manual** approval strategy, OLM creates an update request. As a cluster administrator, you must then manually approve the OLM update request to update the Operator to the new version.
  - c. Select an **Update Channel**.
    - The **latest** channel enables installation of the most recent stable version of the Red Hat OpenShift Pipelines Operator. Currently, it is the default channel for installing the Red Hat OpenShift Pipelines Operator.
    - To install a specific version of the Red Hat OpenShift Pipelines Operator, cluster administrators can use the corresponding **pipelines-<version>** channel. For example, to install the Red Hat OpenShift Pipelines Operator version **1.8.x**, you can use the **pipelines-1.8** channel.

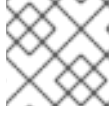


### NOTE

Starting with OpenShift Container Platform 4.11, the **preview** and **stable** channels for installing and upgrading the Red Hat OpenShift Pipelines Operator are not available. However, in OpenShift Container Platform 4.10 and earlier versions, you can use the **preview** and **stable** channels for installing and upgrading the Operator.

5. Click **Install**. You will see the Operator listed on the **Installed Operators** page.



**NOTE**

The Operator is installed automatically into the **openshift-operators** namespace.

- Verify that the **Status** is set to **Succeeded Up to date** to confirm successful installation of Red Hat OpenShift Pipelines Operator.

**WARNING**

The success status may show as **Succeeded Up to date** even if installation of other components is in-progress. Therefore, it is important to verify the installation manually in the terminal.

- Verify that all components of the Red Hat OpenShift Pipelines Operator were installed successfully. Login to the cluster on the terminal, and run the following command:

```
$ oc get tektonconfig config
```

**Example output**

```
NAME      VERSION  READY  REASON
config    1.15.0   True
```

If the **READY** condition is **True**, the Operator and its components have been installed successfully.

Additionally, check the components' versions by running the following command:

```
$ oc get tektonpipeline,tektontrigger,tektonchain,tektonaddon,pac
```

**Example output**

```
NAME                                     VERSION  READY  REASON
tektonpipeline.operator.tekton.dev/pipeline  v0.47.0  True
```

```
NAME                                     VERSION  READY  REASON
tektontrigger.operator.tekton.dev/trigger    v0.23.1  True
```

```
NAME                                     VERSION  READY  REASON
tektonchain.operator.tekton.dev/chain        v0.16.0  True
```

```
NAME                                     VERSION  READY  REASON
tektonaddon.operator.tekton.dev/addon        1.11.0   True
```

```
NAME                                     VERSION  READY  REASON
openshiftpipelinesascode.operator.tekton.dev/pipelines-as-code  v0.19.0  True
```

## 1.2. INSTALLING THE OPENSIFT PIPELINES OPERATOR USING THE CLI

You can install Red Hat OpenShift Pipelines Operator from the OperatorHub using the CLI.

### Procedure

1. Create a Subscription object YAML file to subscribe a namespace to the Red Hat OpenShift Pipelines Operator, for example, **sub.yaml**:

#### Example Subscription

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: openshift-pipelines-operator
  namespace: openshift-operators
spec:
  channel: <channel name> 1
  name: openshift-pipelines-operator-rh 2
  source: redhat-operators 3
  sourceNamespace: openshift-marketplace 4
```

- 1 The channel name of the Operator. The **pipelines-<version>** channel is the default channel. For example, the default channel for Red Hat OpenShift Pipelines Operator version **1.7** is **pipelines-1.7**. The **latest** channel enables installation of the most recent stable version of the Red Hat OpenShift Pipelines Operator.
- 2 Name of the Operator to subscribe to.
- 3 Name of the CatalogSource that provides the Operator.
- 4 Namespace of the CatalogSource. Use **openshift-marketplace** for the default OperatorHub CatalogSources.

2. Create the Subscription object:

```
$ oc apply -f sub.yaml
```

The subscription installs the Red Hat OpenShift Pipelines Operator into the **openshift-operators** namespace. The Operator automatically installs OpenShift Pipelines into the default **openshift-pipelines** target namespace.

## 1.3. RED HAT OPENSIFT PIPELINES OPERATOR IN A RESTRICTED ENVIRONMENT

The Red Hat OpenShift Pipelines Operator enables support for installation of pipelines in a restricted network environment.

The Operator installs a proxy webhook that sets the proxy environment variables in the containers of the pod created by tekton-controllers based on the **cluster** proxy object. It also sets the proxy environment variables in the **TektonPipelines, TektonTriggers, Controllers, Webhooks,** and **Operator Proxy Webhook** resources.

By default, the proxy webhook is disabled for the **openshift-pipelines** namespace. To disable it for any other namespace, you can add the **operator.tekton.dev/disable-proxy: true** label to the **namespace** object.

## 1.4. ADDITIONAL RESOURCES

- You can learn more about installing Operators on OpenShift Container Platform in the [adding Operators to a cluster](#) section.
- To install Tekton Chains using the Red Hat OpenShift Pipelines Operator, see [Using Tekton Chains for Red Hat OpenShift Pipelines supply chain security](#).
- To install and deploy in-cluster Tekton Hub, see [Using Tekton Hub with Red Hat OpenShift Pipelines](#).
- For more information on using pipelines in a restricted environment, see:
  - [Mirroring images to run pipelines in a restricted environment](#)
  - [Configuring Samples Operator for a restricted cluster](#)
  - [Creating a cluster with a mirrored registry](#)

## CHAPTER 2. UNINSTALLING OPENSIFT PIPELINES

Cluster administrators can uninstall the Red Hat OpenShift Pipelines Operator by performing the following steps:

1. Delete the Custom Resources (CRs) for the optional components, **TektonHub** and **TektonResult**, if these CRs exist, and then delete the **TektonConfig** CR.

### CAUTION

If you uninstall the Operator without removing the CRs of optional components, you cannot remove the components later.


2. Uninstall the Red Hat OpenShift Pipelines Operator.
3. Delete the Custom Resource Definitions (CRDs) of the **operator.tekton.dev** group.

Uninstalling only the Operator will not remove the Red Hat OpenShift Pipelines components created by default when the Operator is installed.

### 2.1. DELETING THE OPENSIFT PIPELINES CUSTOM RESOURCES

If the Custom Resources (CRs) for the optional components, **TektonHub** and **TektonResult**, exist, delete these CRs. Then delete the **TektonConfig** CR.

#### Procedure

1. In the **Administrator** perspective of the web console, navigate to **Administration** → **CustomResourceDefinitions**.
2. Type **TektonHub** in the **Filter by name** field to search for the **TektonHub** Custom Resource Definition (CRD).
3. Click the name of the **TektonHub** CRD to display the details page for the CRD.
4. Click the **Instances** tab.
5. If an instance is displayed, click the **Options** menu  for the displayed instance.
6. Select **Delete TektonHub**.
7. Click **Delete** to confirm the deletion of the CR.
8. Repeat these steps, searching for **TektonResult** and then **TektonConfig** in the **Filter by name** box. If any instances are found for these CRDs, delete these instances.



#### NOTE

Deleting the CRs also deletes the Red Hat OpenShift Pipelines components and all the tasks and pipelines on the cluster.



## IMPORTANT

If you uninstall the Operator without removing the **TektonHub** and **TektonResult** CRs, you cannot remove the Tekton Hub and Tekton Results components later.

## 2.2. UNINSTALLING THE RED HAT OPENSIFT PIPELINES OPERATOR

You can uninstall the Red Hat OpenShift Pipelines Operator by using the **Administrator** perspective in the web console.

### Procedure

1. From the **Operators → OperatorHub** page, use the **Filter by keyword** box to search for the **Red Hat OpenShift Pipelines** Operator.
2. Click the **Red Hat OpenShift Pipelines** Operator tile. The Operator tile indicates that the Operator is installed.
3. In the **Red Hat OpenShift Pipelines** Operator description page, click **Uninstall**.
4. In the **Uninstall Operator?** window, select **Delete all operand instances for this operator**, and then click **Uninstall**.



### WARNING


When you uninstall the OpenShift Pipelines Operator, all resources within the **openshift-pipelines** target namespace where OpenShift Pipelines is installed are lost, including the secrets you configured.

## 2.3. DELETING THE CUSTOM RESOURCE DEFINITIONS OF THE OPERATOR.TEKTON.DEV GROUP

Delete the Custom Resource Definitions (CRDs) of the **operator.tekton.dev** group. These CRDs are created by default during the installation of the Red Hat OpenShift Pipelines Operator.

### Procedure

1. In the **Administrator** perspective of the web console, navigate to **Administration → CustomResourceDefinitions**.
2. Type **operator.tekton.dev** in the **Filter by name** box to search for the CRDs in the **operator.tekton.dev** group.
3. To delete each of the displayed CRDs, complete the following steps:

- a. Click the **Options** menu  .
- b. Select **Delete CustomResourceDefinition**.

- c. Click **Delete** to confirm the deletion of the CRD.

#### Additional resources

- You can learn more about uninstalling Operators on OpenShift Container Platform in the [deleting Operators from a cluster](#) section.

## CHAPTER 3. CUSTOMIZING CONFIGURATIONS IN THE TEKTONCONFIG CUSTOM RESOURCE

In Red Hat OpenShift Pipelines, you can customize the following configurations by using the **TektonConfig** custom resource (CR):

- Optimizing OpenShift Pipelines performance, including high-availability mode for the OpenShift Pipelines controller
- Configuring the Red Hat OpenShift Pipelines control plane
- Changing the default service account
- Disabling the service monitor
- Configuring pipeline resolvers
- Disabling cluster tasks and pipeline templates
- Disabling the integration of Tekton Hub
- Disabling the automatic creation of RBAC resources
- Pruning of task runs and pipeline runs

### 3.1. PREREQUISITES

- You have installed the Red Hat OpenShift Pipelines Operator.

### 3.2. PERFORMANCE TUNING USING TEKTONCONFIG CR

You can modify the fields under the **.spec.pipeline.performance** parameter in the **TektonConfig** custom resource (CR) to change high availability (HA) support and performance configuration for the OpenShift Pipelines controller.

#### Example TektonConfig performance fields

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    performance:
      disable-ha: false
      buckets: 7
      replicas: 5
      threads-per-controller: 2
      kube-api-qps: 5.0
      kube-api-burst: 10
```

All fields are optional. If you set them, the Red Hat OpenShift Pipelines Operator includes most of the fields as arguments in the **openshift-pipelines-controller** deployment under the **openshift-pipelines-controller** container. The OpenShift Pipelines Operator also updates the **buckets** field in the **config-**

**leader-election** configuration map under the **openshift-pipelines** namespace.

If you do not specify the values, the OpenShift Pipelines Operator does not update those fields and applies the default values for the OpenShift Pipelines controller.



#### NOTE

If you modify or remove any of the performance fields, the OpenShift Pipelines Operator updates the **openshift-pipelines-controller** deployment and the **config-leader-election** configuration map (if the **buckets** field changed) and re-creates **openshift-pipelines-controller** pods.

High-availability (HA) mode applies to the OpenShift Pipelines controller, which creates and starts pods based on pipeline run and task run definitions. Without HA mode, a single pod executes these operations, potentially creating significant delays under a high load.

In HA mode, OpenShift Pipelines uses several pods (replicas) to execute these operations. Initially, OpenShift Pipelines assigns every controller operation into a bucket. Each replica picks operations from one or more buckets. If two replicas could pick the same operation at the same time, the controller internally determines a *leader* that executes this operation.

HA mode does not affect execution of task runs after the pods are created.

**Table 3.1. Modifiable fields for tuning OpenShift Pipelines performance**

Name	Description	Default value for the OpenShift Pipelines controller
<b>disable-ha</b>	Enable or disable the high availability (HA) mode. By default, the HA mode is enabled.	<b>false</b>
<b>buckets</b>	In HA mode, the number of buckets used to process controller operations. The maximum value is <b>10</b>	<b>1</b>
<b>replicas</b>	In HA mode, the number of pods created to process controller operations. Set this value to the same or lower number than the <b>buckets</b> value.	<b>1</b>
<b>threads-per-controller</b>	The number of threads (workers) to use when the work queue of the OpenShift Pipelines controller is processed.	<b>2</b>
<b>kube-api-qps</b>	The maximum queries per second (QPS) to the cluster master from the REST client.	<b>5.0</b>
<b>kube-api-burst</b>	The maximum burst for a throttle.	<b>10</b>





## NOTE

The OpenShift Pipelines Operator does not control the number of replicas of the OpenShift Pipelines controller. The **replicas** setting of the deployment determines the number of replicas. For example, to change the number of replicas to 3, enter the following command:

```
$ oc --namespace openshift-pipelines scale deployment openshift-pipelines-controller
--replicas=3
```



## IMPORTANT

The **kube-api-qps** and **kube-api-burst** fields are multiplied by 2 in the OpenShift Pipelines controller. For example, if the **kube-api-qps** and **kube-api-burst** values are **10**, the actual QPS and burst values become **20**.

## 3.3. CONFIGURING THE RED HAT OPENSIFT PIPELINES CONTROL PLANE

You can customize the OpenShift Pipelines control plane by editing the configuration fields in the **TektonConfig** custom resource (CR). The Red Hat OpenShift Pipelines Operator automatically adds the configuration fields with their default values so that you can use the OpenShift Pipelines control plane.

### Procedure

1. In the **Administrator** perspective of the web console, navigate to **Administration** → **CustomResourceDefinitions**.
2. Use the **Search by name** box to search for the **tektonconfigs.operator.tekton.dev** custom resource definition (CRD). Click **TektonConfig** to see the CRD details page.
3. Click the **Instances** tab.
4. Click the **config** instance to see the **TektonConfig** CR details.
5. Click the **YAML** tab.
6. Edit the **TektonConfig** YAML file based on your requirements.

### Example of TektonConfig CR with default values

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    running-in-environment-with-injected-sidecars: true
    metrics.taskrun.duration-type: histogram
    metrics.pipelinerun.duration-type: histogram
    await-sidecar-readiness: true
  params:
    - name: enableMetrics
```

```

value: 'true'
default-service-account: pipeline
require-git-ssh-secret-known-hosts: false
enable-tekton-oci-bundles: false
metrics.taskrun.level: task
metrics.pipelinerun.level: pipeline
enable-api-fields: stable
enable-provenance-in-status: false
enable-custom-tasks: true
disable-creds-init: false
disable-affinity-assistant: true

```

### 3.3.1. Modifiable fields with default values

The following list includes all modifiable fields with their default values in the **TektonConfig** CR:

- **running-in-environment-with-injected-sidecars** (default: **true**): Set this field to **false** if pipelines run in a cluster that does not use injected sidecars, such as Istio. Setting it to **false** decreases the time a pipeline takes for a task run to start.



#### NOTE

For clusters that use injected sidecars, setting this field to **false** can lead to an unexpected behavior.

- **await-sidecar-readiness** (default: **true**): Set this field to **false** to stop OpenShift Pipelines from waiting for **TaskRun** sidecar containers to run before it begins to operate. This allows tasks to be run in environments that do not support the **downwardAPI** volume type.
- **default-service-account** (default: **pipeline**): This field contains the default service account name to use for the **TaskRun** and **PipelineRun** resources, if none is specified.
- **require-git-ssh-secret-known-hosts** (default: **false**): Setting this field to **true** requires that any Git SSH secret must include the **known\_hosts** field.
  - For more information about configuring Git SSH secrets, see *Configuring SSH authentication for Git* in the *Additional resources* section.
- **enable-tekton-oci-bundles** (default: **false**): Set this field to **true** to enable the use of an experimental alpha feature named Tekton OCI bundle.
- **enable-api-fields** (default: **stable**): Setting this field determines which features are enabled. Acceptable value is **stable**, **beta**, or **alpha**.



#### NOTE

Red Hat OpenShift Pipelines does not support the **alpha** value.

- **enable-provenance-in-status** (default: **false**): Set this field to **true** to enable populating the **provenance** field in **TaskRun** and **PipelineRun** statuses. The **provenance** field contains metadata about resources used in the task run and pipeline run, such as the source from where a remote task or pipeline definition was fetched.
- **enable-custom-tasks** (default: **true**): Set this field to **false** to disable the use of custom tasks in pipelines.

- **disable-creds-init** (default: **false**): Set this field to **true** to prevent OpenShift Pipelines from scanning attached service accounts and injecting any credentials into your steps.
- **disable-affinity-assistant** (default: **true**): Set this field to **false** to enable affinity assistant for each **TaskRun** resource sharing a persistent volume claim workspace.

### Metrics options

You can modify the default values of the following metrics fields in the **TektonConfig** CR:

- **metrics.taskrun.duration-type** and **metrics.pipelinerun.duration-type** (default: **histogram**): Setting these fields determines the duration type for a task or pipeline run. Acceptable value is **gauge** or **histogram**.
- **metrics.taskrun.level** (default: **task**): This field determines the level of the task run metrics. Acceptable value is **taskrun**, **task**, or **namespace**.
- **metrics.pipelinerun.level** (default: **pipeline**): This field determines the level of the pipeline run metrics. Acceptable value is **pipelinerun**, **pipeline**, or **namespace**.

### 3.3.2. Optional configuration fields

The following fields do not have a default value, and are considered only if you configure them. By default, the Operator does not add and configure these fields in the **TektonConfig** custom resource (CR).

- **default-timeout-minutes**: This field sets the default timeout for the **TaskRun** and **PipelineRun** resources, if none is specified when creating them. If a task run or pipeline run takes more time than the set number of minutes for its execution, then the task run or pipeline run is timed out and cancelled. For example, **default-timeout-minutes: 60** sets 60 minutes as default.
- **default-managed-by-label-value**: This field contains the default value given to the **app.kubernetes.io/managed-by** label that is applied to all **TaskRun** pods, if none is specified. For example, **default-managed-by-label-value: tekton-pipelines**.
- **default-pod-template**: This field sets the default **TaskRun** and **PipelineRun** pod templates, if none is specified.
- **default-cloud-events-sink**: This field sets the default **CloudEvents** sink that is used for the **TaskRun** and **PipelineRun** resources, if none is specified.
- **default-task-run-workspace-binding**: This field contains the default workspace configuration for the workspaces that a **Task** resource declares, but a **TaskRun** resource does not explicitly declare.
- **default-affinity-assistant-pod-template**: This field sets the default **PipelineRun** pod template that is used for affinity assistant pods, if none is specified.
- **default-max-matrix-combinations-count**: This field contains the default maximum number of combinations generated from a matrix, if none is specified.

## 3.4. CHANGING THE DEFAULT SERVICE ACCOUNT FOR OPENSIFT PIPELINES

You can change the default service account for OpenShift Pipelines by editing the **default-service-account** field in the **.spec.pipeline** and **.spec.trigger** specifications. The default service account name is **pipeline**.

### Example

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    default-service-account: pipeline
  trigger:
    default-service-account: pipeline
    enable-api-fields: stable
```

## 3.5. SETTING LABELS AND ANNOTATIONS FOR THE OPENSIFT PIPELINES INSTALLATION NAMESPACE

You can set labels and annotations for the **openshift-pipelines** namespace in which the operator installs OpenShift Pipelines.



### NOTE

Changing the name of the **openshift-pipelines** namespace is not supported.

Specify the labels and annotations by adding them to the **spec.targetNamespaceMetadata** specification in the **TektonConfig** custom resource (CR).

### Example of setting labels and annotations for the **openshift-pipelines** namespace

```
apiVersion: operator.tekton.dev/v1
kind: TektonConfig
metadata:
  name: config
spec:
  targetNamespaceMetadata:
    labels: {"example-label":"example-value"}
    annotations: {"example-annotation":"example-value"}
```

## 3.6. DISABLING THE SERVICE MONITOR

You can disable the service monitor, which is part of OpenShift Pipelines, to expose the telemetry data. To disable the service monitor, set the **enableMetrics** parameter to **false** in the **.spec.pipeline** specification of the **TektonConfig** custom resource (CR):

### Example

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
```

```

name: config
spec:
  pipeline:
    params:
      - name: enableMetrics
        value: 'false'

```

### 3.7. CONFIGURING PIPELINE RESOLVERS

You can configure pipeline resolvers in the **TektonConfig** custom resource (CR). You can enable or disable these pipeline resolvers:

- **enable-bundles-resolver**
- **enable-cluster-resolver**
- **enable-git-resolver**
- **enable-hub-resolver**

#### Example

```

apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    enable-bundles-resolver: true
    enable-cluster-resolver: true
    enable-git-resolver: true
    enable-hub-resolver: true

```

You can also provide resolver specific configurations in the **TektonConfig** CR. For example, define the following fields in the **map[string]string** format to set configurations for each pipeline resolver:

#### Example

```

apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    bundles-resolver-config:
      default-service-account: pipelines
    cluster-resolver-config:
      default-namespace: test
    git-resolver-config:
      server-url: localhost.com
    hub-resolver-config:
      default-tekton-hub-catalog: tekton

```

### 3.8. DISABLING CLUSTER TASKS AND PIPELINE TEMPLATES

By default, the **TektonAddon** custom resource (CR) installs **clusterTasks** and **pipelineTemplates** resources along with OpenShift Pipelines on the cluster.

You can disable installation of the **clusterTasks** and **pipelineTemplates** resources by setting the parameter value to **false** in the **.spec.addon** specification. In addition, you can disable the **communityClusterTasks** parameter.

### Example

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  addon:
    params:
      - name: clusterTasks
        value: 'false'
      - name: pipelineTemplates
        value: 'false'
      - name: communityClusterTasks
        value: 'true'
```



### IMPORTANT

In Red Hat OpenShift Pipelines 1.10, **ClusterTask** functionality is deprecated and is planned to be removed in a future release.

## 3.9. DISABLING THE INTEGRATION OF TEKTON HUB

You can disable the integration of Tekton Hub in the web console **Developer** perspective by setting the **enable-devconsole-integration** parameter to **false** in the **TektonConfig** custom resource (CR).

### Example of disabling Tekton Hub

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  hub:
    params:
      - name: enable-devconsole-integration
        value: false
```

## 3.10. DISABLING THE AUTOMATIC CREATION OF RBAC RESOURCES

The default installation of the Red Hat OpenShift Pipelines Operator creates multiple role-based access control (RBAC) resources for all namespaces in the cluster, except the namespaces matching the **^(openshift|kubernetes)-\*** regular expression pattern. Among these RBAC resources, the **pipelines-scc-rolebinding** security context constraint (SCC) role binding resource is a potential security issue, because the associated **pipelines-scc** SCC has the **RunAsAny** privilege.

To disable the automatic creation of cluster-wide RBAC resources after the Red Hat OpenShift Pipelines Operator is installed, cluster administrators can set the **createRbacResource** parameter to **false** in the cluster-level **TektonConfig** custom resource (CR).

### Example TektonConfig CR

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  params:
  - name: createRbacResource
    value: "false"
  ...
```



#### WARNING

As a cluster administrator or a user with appropriate privileges, when you disable the automatic creation of RBAC resources for all namespaces, the default **ClusterTask** resource does not work. For the **ClusterTask** resource to function, you must create the RBAC resources manually for each intended namespace.

## 3.11. DISABLING INLINE SPECIFICATION OF PIPELINES AND TASKS

By default, OpenShift Pipelines supports inline specification of pipelines and tasks in the following cases:

- You can create a **Pipeline** CR that includes one or more task specifications, as in the following example:

#### Example of an inline specification in a Pipeline CR

```
apiVersion: operator.tekton.dev/v1
kind: Pipeline
metadata:
  name: pipelineInline
spec:
  tasks:
  taskSpec:
  # ...
```

- You can create a **PipelineRun** custom resource (CR) that includes a pipeline specification, as in the following example:

#### Example of an inline specification in a PipelineRun CR

```
apiVersion: operator.tekton.dev/v1
kind: PipelineRun
```

```

metadata:
  name: pipelineRunInline
spec:
  pipelineSpec:
    tasks:
# ...

```

- You can create a **TaskRun** custom resource (CR) that includes a task specification, as in the following example:

### Example of an inline specification in a **TaskRun** CR

```

apiVersion: operator.tekton.dev/v1
kind: TaskRun
metadata:
  name: taskRunInline
spec:
  taskSpec:
    steps:
# ...

```

You can disable inline specification in some or all of these cases. To disable the inline specification, set the **disable-inline-spec** field of the **.spec.pipeline** specification of the **TektonConfig** CR, as in the following example:

### Example configuration that disables inline specification

```

apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
spec:
  pipeline:
    disable-inline-spec: "pipeline,pipelinerun,taskrun"
# ...

```

You can set the **disable-inline-spec** parameter to any single value or to a comma-separated list of multiple values. The following values for the parameter are valid:

**Table 3.2. Supported values for the **disable-inline-spec** parameter**

Value	Description
<b>pipeline</b>	You cannot use a <b>taskSpec:</b> spec to define a task inside a <b>Pipeline</b> CR. Instead, you must use a <b>taskRef:</b> spec to incorporate a task from a <b>Task</b> CR or to specify a task using a resolver.
<b>pipelinerun</b>	You cannot use a <b>pipelineSpec:</b> spec to define a pipeline inside a <b>PipelineRun</b> CR. Instead, you must use a <b>pipelineRef:</b> spec to incorporate a pipeline from a <b>Pipeline</b> CR or to specify a pipeline using a resolver.



Value	Description
<b>taskrun</b>	You cannot use a <b>taskSpec:</b> spec to define a task inside a <b>TaskRun</b> CR. Instead, you must use a <b>taskRef:</b> spec to incorporate a task from a <b>Task</b> CR or to specify a task using a resolver.

## 3.12. AUTOMATIC PRUNING OF TASK RUNS AND PIPELINE RUNS

Stale **TaskRun** and **PipelineRun** objects and their executed instances occupy physical resources that can be used for active runs. For optimal utilization of these resources, Red Hat OpenShift Pipelines provides a pruner component that automatically removes unused objects and their instances in various namespaces.



### NOTE

You can configure the pruner for your entire installation by using the **TektonConfig** custom resource and modify configuration for a namespace by using namespace annotations. However, you cannot selectively auto-prune an individual task run or pipeline run in a namespace.

### 3.12.1. Configuring the pruner

You can use the **TektonConfig** custom resource to configure periodic pruning of resources associated with pipeline runs and task runs.

The following example corresponds to the default configuration:

#### Example of the pruner configuration

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonConfig
metadata:
  name: config
# ...
spec:
  pruner:
    resources:
      - taskrun
      - pipelinerun
    keep: 100
    prune-per-resource: false
    schedule: "* 8 * * *"
    startingDeadlineSeconds: 60
# ...
```

Table 3.3. Supported parameters for pruner configuration

Parameter	Description
-----------	-------------

Parameter	Description
<b>schedule</b>	The Cron schedule for running the pruner process. The default schedule runs the process at 08:00 every day. For more information about the Cron schedule syntax, see <a href="#">Cron schedule syntax</a> in the Kubernetes documentation.
<b>resources</b>	The resource types to which the pruner applies. The available resource types are <b>taskrun</b> and <b>pipelinerun</b>
<b>keep</b>	The number of most recent resources of every type to keep.
<b>prune-per-resource</b>	<p>If set to <b>false</b>, the value for the <b>keep</b> parameter denotes the total number of task runs or pipeline runs. For example, if <b>keep</b> is set to <b>100</b>, then the pruner keeps 100 most recent task runs and 100 most recent pipeline runs and removes all other resources.</p> <p>If set to <b>true</b>, the value for the <b>keep</b> parameter is calculated separately for pipeline runs referencing each pipeline and for task runs referencing each task. For example, if <b>keep</b> is set to <b>100</b>, then the pruner keeps 100 most recent pipeline runs for <b>Pipeline1</b>, 100 most recent pipeline runs for <b>Pipeline2</b>, 100 most recent task runs for <b>Task1</b>, and so on, and removes all other resources.</p>
<b>keep-since</b>	The maximum time for which to keep resources, in minutes. For example, to retain resources which were created not more than five days ago, set <b>keep-since</b> to <b>7200</b> .
<b>startingDeadlineSeconds</b>	This parameter is optional. If the pruner job is not started at the scheduled time for any reason, this setting configures the maximum time, in seconds, in which the job can still be started. If the job is not started within the specified time, OpenShift Pipelines considers this job failed and starts the pruner at the next scheduled time. If you do not specify this parameter and the pruner job does not start at the scheduled time, OpenShift Pipelines attempts to start the job at any later time possible.

**NOTE**

The **keep** and **keep-since** parameters are mutually exclusive. Use only one of them in your configuration.

**3.12.2. Annotations for automatically pruning task runs and pipeline runs**

To modify the configuration for automatic pruning of task runs and pipeline runs in a namespace, you can set annotations in the namespace.

The following namespace annotations have the same meanings as the corresponding keys in the **TektonConfig** custom resource:

- **operator.tekton.dev/prune.schedule**
- **operator.tekton.dev/prune.resources**
- **operator.tekton.dev/prune.keep**
- **operator.tekton.dev/prune.prune-per-resource**
- **operator.tekton.dev/prune.keep-since**

**NOTE**

The **operator.tekton.dev/prune.resources** annotation accepts a comma-separated list. To prune both task runs and pipeline runs, set this annotation to "**taskrun, pipelinerun**".

The following additional namespace annotations are available:

- **operator.tekton.dev/prune.skip**: When set to **true**, the namespace for which the annotation is configured is not pruned.
- **operator.tekton.dev/prune.strategy**: Set the value of this annotation to either **keep** or **keep-since**.

For example, the following annotations retain all task runs and pipeline runs created in the last five days and delete the older resources:

**Example of auto-pruning annotations**

```
kind: Namespace
apiVersion: v1
# ...
spec:
  annotations:
    operator.tekton.dev/prune.resources: "taskrun, pipelinerun"
    operator.tekton.dev/prune.keep-since: 7200
# ...
```

**3.13. ADDITIONAL RESOURCES**

- [Authenticating pipelines with repositories using secrets](#)
- [Managing non-versioned and versioned cluster tasks](#)

- [Creating pipeline templates in the Administrator perspective](#)
- [Pruning objects to reclaim resources](#)