Standalone Deployment

Information about Standalone Deployment of Kubeflow Pipelines

As an alternative to deploying Kubeflow Pipelines (KFP) as part of the Kubeflow deployment, you also have a choice to deploy only Kubeflow Pipelines. Follow the instructions below to deploy Kubeflow Pipelines standalone using the supplied kustomize manifests.

You should be familiar with Kubernetes, kubectl, and kustomize.

> ### Installation options for Kubeflow Pipelines standalone
>
> This guide currently describes how to install Kubeflow Pipelines standalone on Google Cloud Platform (GCP). You can also install Kubeflow Pipelines standalone on other platforms. This guide needs updating. See Issue 1253.

Before you get started

Working with Kubeflow Pipelines Standalone requires a Kubernetes cluster as well as an installation of kubectl.

Download and install kubectl

Download and install kubectl by following the kubectl installation guide.

You need kubectl version 1.14 or higher for native support of kustomize.

Set up your cluster

If you have an existing Kubernetes cluster, continue with the instructions for configuring kubectl to talk to your cluster.

See the GKE guide to creating a cluster for Google Cloud Platform (GCP).

Use the gcloud container clusters create command to create a cluster that can run all Kubeflow Pipelines samples:

```
# The following parameters can be customized based on your needs.

CLUSTER_NAME="kubeflow-pipelines-standalone"
ZONE="us-central1-a"
MACHINE_TYPE="e2-standard-2" # A machine with 2 CPUs and 8GB memory.
SCOPES="cloud-platform" # This scope is needed for running some pipeline samples. Read the w

gcloud container clusters create $CLUSTER_NAME \
     --zone $ZONE \
     --machine-type $MACHINE_TYPE \
     --scopes $SCOPES
```

Note: `e2-standard-2` doesn't support GPU. You can choose machine types that meet your need by referring to guidance in Cloud Machine families.

Warning: Using `SCOPES="cloud-platform"` grants all GCP permissions to the cluster. For a more secure cluster setup, refer to Authenticating Pipelines to GCP.

Note, some legacy pipeline examples may need minor code change to run on clusters with `SCOPES="cloud-platform"`, refer to Authoring Pipelines to use default service account.

References:

- GCP regions and zones documentation
- gcloud command-line tool guide
- gcloud command reference

Configure kubectl to talk to your cluster

See the Google Kubernetes Engine (GKE) guide to configuring cluster access for kubectl.

Deploying Kubeflow Pipelines

1. Deploy the Kubeflow Pipelines:

```
export PIPELINE_VERSION=1.8.1
kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/cluster-scoped-res
kubectl wait --for condition=established --timeout=60s crd/applications.app.k8s.io
kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=$PIPEL
```

The Kubeflow Pipelines deployment requires approximately 3 minutes to complete.

Note: The above commands apply to Kubeflow Pipelines version 0.4.0 and higher.

For Kubeflow Pipelines version 0.2.0 ~ 0.3.0, use:

```
export PIPELINE_VERSION=<kfp-version-between-0.2.0-and-0.3.0>
kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/base/crds?ref=$PIP
kubectl wait --for condition=established --timeout=60s crd/applications.app.k8s.io
kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=$PIPEL
```

For Kubeflow Pipelines version < 0.2.0, use:

```
PIPELINE_VERSION=<kfp-version-0.1.x>
apply -k "github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=$PIPELINE_VERSIO
```

Note: `kubectl apply -k` accepts local paths and paths that are formatted as hashicorp/go-getter URLs. While the paths in the preceding commands look like URLs, the paths are not valid URLs.

> ### Deprecation Notice
>
> Kubeflow Pipelines will change default executor from Docker to Emissary starting KFP backend v1.8, docker executor has been deprecated on Kubernetes 1.20+.
>
> For Kubeflow Pipelines before v1.8, configure to use Emissary executor by referring to Argo Workflow Executors.

2. Get the public URL for the Kubeflow Pipelines UI and use it to access the Kubeflow Pipelines UI:

```
kubectl describe configmap inverse-proxy-config -n kubeflow | grep googleusercontent.c
```

Upgrading Kubeflow Pipelines

1. For release notices and breaking changes, refer to Upgrading Kubeflow Pipelines.

2. Check the Kubeflow Pipelines GitHub repository for available releases.

3. To upgrade to Kubeflow Pipelines 0.4.0 and higher, use the following commands:

   ```
   export PIPELINE_VERSION=<version-you-want-to-upgrade-to>
   kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/cluster-scoped-res
   kubectl wait --for condition=established --timeout=60s crd/applications.app.k8s.io
   kubectl apply -k "github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=$PIPEL
   ```

   To upgrade to Kubeflow Pipelines 0.3.0 and lower, use the deployment instructions to upgrade your Kubeflow Pipelines cluster.

4. Delete obsolete resources manually.

   Depending on the version you are upgrading from and the version you are upgrading to, some Kubeflow Pipelines resources may have become obsolete.

   If you are upgrading from Kubeflow Pipelines < 0.4.0 to 0.4.0 or above, you can remove the following obsolete resources after the upgrade: `metadata-deployment` , `metadata-service` .

   Run the following command to check if these resources exist on your cluster:

   ```
   kubectl -n <KFP_NAMESPACE> get deployments | grep metadata-deployment
   kubectl -n <KFP_NAMESPACE> get service | grep metadata-service
   ```

   If these resources exist on your cluster, run the following commands to delete them:

   ```
   kubectl -n <KFP_NAMESPACE> delete deployment metadata-deployment
   kubectl -n <KFP_NAMESPACE> delete service metadata-service
   ```

   For other versions, you don’t need to do anything.

Customizing Kubeflow Pipelines

Kubeflow Pipelines can be configured through kustomize overlays.

To begin, first clone the Kubeflow Pipelines GitHub repository, and use it as your working directory.

Deploy on GCP with Cloud SQL and Google Cloud Storage

Note: This is recommended for production environments. For more details about customizing your environment for GCP, see the Kubeflow Pipelines GCP manifests.

Change deployment namespace

To deploy Kubeflow Pipelines standalone in namespace `<my-namespace>` :

1. Set the `namespace` field to `<my-namespace>` in dev/kustomization.yaml or gcp/kustomization.yaml.
2. Set the `namespace` field to `<my-namespace>` in cluster-scoped-resources/kustomization.yaml
3. Apply the changes to update the Kubeflow Pipelines deployment:

```
kubectl apply -k manifests/kustomize/cluster-scoped-resources
kubectl apply -k manifests/kustomize/env/dev
```

Note: If using GCP Cloud SQL and Google Cloud Storage, set the proper values in manifests/kustomize/env/gcp/params.env, then apply with this command:

```
kubectl apply -k manifests/kustomize/cluster-scoped-resources
kubectl apply -k manifests/kustomize/env/gcp
```

Disable the public endpoint

By default, the KFP standalone deployment installs an inverting proxy agent that exposes a public URL. If you want to skip the installation of the inverting proxy agent, complete the following:

1. Comment out the proxy components in the base `kustomization.yaml` . For example in manifests/kustomize/env/dev/kustomization.yaml comment out `inverse-proxy` .
2. Apply the changes to update the Kubeflow Pipelines deployment:

```
kubectl apply -k manifests/kustomize/env/dev
```

Note: If using GCP Cloud SQL and Google Cloud Storage, set the proper values in manifests/kustomize/env/gcp/params.env, then apply with this command:

```
kubectl apply -k manifests/kustomize/env/gcp
```

3. Verify that the Kubeflow Pipelines UI is accessible by port-forwarding:

```
kubectl port-forward -n kubeflow svc/ml-pipeline-ui 8080:80
```

4. Open the Kubeflow Pipelines UI at `http://localhost:8080/` .

Uninstalling Kubeflow Pipelines

To uninstall Kubeflow Pipelines, run `kubectl delete -k <manifest-file>` .

For example, to uninstall KFP using manifests from a GitHub repository, run:

```
export PIPELINE_VERSION=1.8.1
kubectl delete -k "github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=$PIPELINE_V
kubectl delete -k "github.com/kubeflow/pipelines/manifests/kustomize/cluster-scoped-resource
```

To uninstall KFP using manifests from your local repository or file system, run:

```
kubectl delete -k manifests/kustomize/env/dev
kubectl delete -k manifests/kustomize/cluster-scoped-resources
```

Note: If you are using GCP Cloud SQL and Google Cloud Storage, run:

```
kubectl delete -k manifests/kustomize/env/gcp
kubectl delete -k manifests/kustomize/cluster-scoped-resources
```

Best practices for maintaining manifests

Similar to source code, configuration files belong in source control. A repository manages the changes to your manifest files and ensures that you can repeatedly deploy, upgrade, and uninstall your components.

Maintain your manifests in source control

After creating or customizing your deployment manifests, save your manifests to a local or remote source control repository. For example, save the following `kustomization.yaml`:

```
# kustomization.yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
# Edit the following to change the deployment to your custom namespace.
namespace: kubeflow
# You can add other customizations here using kustomize.
# Edit ref in the following link to deploy a different version of Kubeflow Pipelines.
bases:
- github.com/kubeflow/pipelines/manifests/kustomize/env/dev?ref=1.8.1
```

Further reading

- To learn about kustomize workflows with off-the-shelf configurations, see the kustomize configuration workflows guide.

Troubleshooting

- If your pipelines are stuck in ContainerCreating state and it has pod events like

```
MountVolume.SetUp failed for volume "gcp-credentials-user-gcp-sa" : secret "user-gcp-sa" not
```

You should remove `use_gcp_secret` usages as documented in Authenticating Pipelines to GCP.

What's next

- Connecting to Kubeflow Pipelines standalone on Google Cloud using the SDK
- Authenticating Pipelines to GCP if you want to use GCP services in Kubeflow Pipelines.

Feedback

Was this page helpful?

Yes No

Last modified March 2, 2022: Update KFP standalone-deployment.md on applying GCP Cloud SQL config (#3167) (3571feb5)