Managing Kubernetes Clusters

The Cisco Container Platform web interface allows you to manage Kubernetes clusters by using the **Kubernetes Dashboard**. Once you set up the **Kubernetes Dashboard**, you can deploy applications on the authorized Kubernetes clusters, and manage the application and the cluster itself.

This chapter contains the following topics:

Setting Up Kubernetes Dashboard to Access Clusters

The steps to set up the Kubernetes dashboard differ based on the method used for cluster creation.

This section contains the following topics:

Setting Up Kubernetes Dashboard for vSphere On-prem Clusters

For a vSphere on-prem cluster, you can access the Kubernetes dashboard using the `Kubeconfig` file or the Kubernetes default token. The steps to use the default token are same as those provided in [Setting up Kubernetes Dashboard for On-prem AWS IAM Enabled Clusters].

Follow these steps to set up the Kubernetes dashboard using the `Kubeconfig` file.

Step 1 To download the Kubeconfig file that provides you access to the Kubernetes cluster, perform these steps on the Cisco Container Platform web interface:

a) From the left pane, click **Clusters**.

b) From the drop-down list displayed under the **ACTIONS** column, choose **Download Token** to get the `Kubeconfig` file of the vSphere cluster.

The `Kubeconfig` file is downloaded to your local system.

Step 2 To set up the Kubernetes dashboard access, perform these steps in the Kubernetes dashboard:

a) Click the **Kubeconfig** radio button.

b) Select the `Kubeconfig` file that you got in Step 1-b.

Setting Up Kubernetes Dashboard for On-prem AWS IAM Enabled

For an on-prem AWS cluster that has IAM enabled, you can access the Kubernetes dashboard only by using the Kubernetes default token.

Step 1 To download the `Kubeconfig` file that provides you access to the Kubernetes cluster, perform these steps on the Cisco Container Platform web interface:

a) From the left pane, click **Clusters**.

b) From the drop-down list displayed under the **ACTIONS** column, choose **Download Token** to get the `Kubeconfig` file of the on-prem AWS cluster.
The `Kubeconfig` file is downloaded to your local system.

Step 2 To get the Kubernetes default token, perform these steps in the kubectl utility:

a) List the Kubernetes secrets in the `kube-system` namespace.

```
kubectl get secrets -n kube-system
```

b) Search for the secret that has the following format:

```
default-token-XXXXX
```

c) Get the default token.

```
kubectl describe secret default-token-XXXXX -n kube-system
```

Step 3 To set up the Kubernetes dashboard access, perform these steps in the Kubernetes dashboard:

a) Click the **Token** radio button.

b) In the **Enter token** field, enter the Kubernetes default token that you got in Step 2-c.

Setting Up Kubernetes Dashboard for AWS EKS Clusters

Step 1 To download the `Kubeconfig` file that provides you access to the AWS EKS cluster, perform these steps on the Cisco Container Platform web interface:

a) From the left pane, click **Clusters**.

b) From the drop-down list displayed under the **ACTIONS** column, choose **Download Token** to get the `Kubeconfig` file of the AWS EKS cluster.
The `Kubeconfig` file is downloaded to your local system.

Step 2 To set up the Kubernetes dashboard access, follow the steps provided on the Dashboard Tutorial page.

Setting Up Kubernetes Dashboard for AKS Clusters

Step 1 To download the Kubeconfig file that provides you access to the AKS cluster, perform these steps on the Cisco Container Platform web interface:

a) From the left pane, click **Clusters** and then click the **Azure** tab.

b) For the cluster whose dashboard you would like to access, from the drop-down list displayed under the **ACTIONS** column, choose **Download Kubeconfig** to get the `kubeconfig` file of the AKS cluster.
The `kubeconfig` file is downloaded to your local system.

Step 2 To set up Kubernetes dashboard access, follow these steps:

a) Create the necessary cluster role binding.

```
kubectl --kubeconfig=[location_of_kubeconfig_downloaded_from_ccp_dashboard] create
clusterrolebinding kubernetes-dashboard --clusterrole=cluster-admin
--serviceaccount=kube-system:kubernetes-dashboard
```

b) Connect to the Kubernetes dashboard using the Azure CLI.

```
az aks browse --resource-group [name_of_your_resource_group] --name [name_of_your_aks_cluster]
```

The Kubernetes dashboard is displayed in a new browser tab.

For more information, refer to the Dashboard Tutorial page.

Note In the Dashboard Tutorial page, you must follow the steps in the RBAC enabled cluster section because currently, in Cisco Container Platform, RBAC is enabled by default for all AKS clusters.

Monitoring Health of Cluster Deployments

It is recommended to continuously monitor the health of your cluster deployment to improve the probability of early detection of failures and avoid any significant impact from a cluster failure.

Cisco Container Platform is deployed with Prometheus and Grafana, which are configured to start monitoring and logging services automatically when a Kubernetes cluster is created.

Prometheus is an open-source systems monitoring and alerting toolkit and Grafana is an open source metric analytics and visualization suite.

Prometheus collects the data from the cluster deployment, and Grafana provides a general purpose dashboard for displaying the collected data. Grafana offers a highly customizable and user-friendly dashboard for monitoring purposes.

Note A user with *Administrator* role can view all the cluster deployments, but a user with *User* role can view only those clusters for which the user has permission to view.

The following example shows a use case in which Grafana is available in the `ccp` namespace.

Step 1 Access the Kubernetes cluster master node using ssh.

```
ssh -l <username> <IP address of master node>
```

Once you create a Kubernetes cluster, it may take a few minutes for the necessary services to start. If ssh to a cluster fails, we recommend that you try again after a few minutes.

Step 2 Follow these steps to access Grafana.

Note On the Control Plane, Grafana is installed in the default namespace, therefore you must skip `-n ccp` in the commands.

```
export INGRESS_IP=$(kubectl get svc nginx-ingress-controller -n ccp
-o=jsonpath='{.spec.loadBalancerIP}')
export GRAFANA_USER=$(kubectl get secret ccp-monitor-grafana -n ccp -o=jsonpath='{.data.admin-user}'
 | base64 --decode)
export GRAFANA_PASSWORD=$(kubectl get secret ccp-monitor-grafana -n ccp
-o=jsonpath='{.data.admin-password}' | base64 --decode)
echo "Login to Grafana at http://${INGRESS_IP}/grafana as user ${GRAFANA_USER} with password
${GRAFANA_PASSWORD}" && unset GRAFANA_PASSWORD GRAFANA_USER
```

Note It is important to either change or retain the original login credentials since the secret that was used to initialize the Grafana login may be lost or changed with future upgrades.

Step 3 Add Prometheus as the data source and configure the Grafana dashboard to monitor the health of your cluster deployments.

Example of Monitoring Multiple Prometheus Instances

To monitor multiple Prometheus instances you must expose Prometheus as an Ingress resource so that you can access it from a Grafana instance that is running in a different cluster.

Note The following example is valid only if Harbor is not installed.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/add-base-url: "true"
    nginx.ingress.kubernetes.io/rewrite-target: /
name: ccp-monitor-prometheus-server
namespace: ccp
spec:
rules:
- http:
    paths:
    - backend:
        serviceName: ccp-monitor-prometheus-server
        servicePort: 9090
        path: /
```

After Promethus is accessible externally from the cluster, you can add it as a new datasource in Grafana.

Monitoring Logs from Cluster Deployments

The Elasticsearch, Fluentd, and Kibana (EFK) stack enables you to collect and monitor log data from containerized applications for troubleshooting or compliance purposes. These components are automatically installed when you install Cisco Container Platform.

Fluentd is an open source data collector. It works at the backend to collect and forward the log data to Elasticsearch.

Kibana is an open source analytics and visualization platform designed to work with Elasticsearch. It allows you to create rich visualizations and dashboards with the aggregated data.

Note A user with the *Administrator* role can view all logs, but a user with *User* role can view logs for only those clusters for which the user has permission to view.

This section contains the following topics:

Viewing EFK Logs Using Kibana (Tenant Cluster)

Before you begin

Ensure that you have installed the `kubectl` utility.

Step 1 Download the Kubeconfig file of the cluster whose logs you want to view, see Downloading Kubeconfig File.

Step 2 Copy the contents of the downloaded Kubeconfig file to:

- Your local host `~/.kube/config`
- A local file and export KUBECONFIG=<*Downloaded Kubeconfig file*>

Step 3 Create a port-forward using kubectl to access Kibana from outside a cluster.

a) Determine the pod.

```
kubectl -n ccp get pods
```

Example

```
ccp-efk-kibana-6d7c97575c-9qxbf
```

b) Open a port-forward.

Example

```
kubectl port-forward -n ccp ccp-efk-kibana-6d7c97575c-9qxbf 5601:5601
```

Step 4 Access the Kibana UI and view the data from the target tenant cluster using a web browser.

http://localhost:5601/app/kibana

For more information on customizing the Kibana UI, refer to the latest Kibana documentation.

Viewing EFK Logs Using Kibana (Control Plane Cluster)

Before you begin

Ensure that you have installed the `kubectl` utility.

Step 1 Access the Kubernetes cluster master node using ssh.

```
ssh ccpuser@control plane master node
sudo cat /etc/kubernetes/admin.conf
```

Step 2 Copy the contents of the downloaded Kubeconfig file to:

- Your local host `~/.kube/config`
- A local file and export KUBECONFIG=<*Full path of the Kubeconfig local file*>

For more information on setting Kubeconfig, see Configure Access to Multiple Clusters.

Step 3 Create a port-forward using kubectl to access Kibana from outside a cluster.

a) Determine the pod.

```
kubectl get pods
```

Example

```
ccp-efk-kibana-6d7c97575c-9qxbf
```

b) Open a port-forward.

Example

```
kubectl port-forward ccp-efk-kibana-6d7c97575c-9qxbf 5601:5601
```

Step 4 Access the Kibana UI and view the data from the target tenant cluster using a web browser.

```
http://localhost:5601/app/kibana
```

For more information on customizing the Kibana UI, refer to the latest Kibana documentation.

Forwarding Logs to External Elasticsearch Server

Use the following Curl commands to configure forwarding of logs to an external Elasticsearch server:

Step 1 Open a terminal that has a curl client installed.

Step 2 Configure Cisco Container Platform login credentials.

```
export MGMT_HOST=https://<Cisco Container Platform IP address>:<Port>
export CCP_USER=<Username>
export CCP_PASSPHRASE=<Passphrase>
```

Step 3 Login to Cisco Container Platform and save the session cookie for future requests into the cookies.txt local file.

```
curl -k -j -c cookies.txt -X POST -H "Content-Type:application/x-www-form-urlencoded" -d
"username=$CCP_USER&password=$CCP_PASSWORD" $MGMT_HOST/2/system/login/
```

Step 4 Get the list of cluster names.

```
curl -s -k -b cookies.txt -H "Content-Type: application/json"
$MGMT_HOST/2/clusters/ | jq -r '.[].name'
```

Step 5 Set the CLUSTER_NAME environment variable to the cluster that you are working on.

```
export CLUSTER_NAME="<A cluster name from Step 2>"
```

Step 6 Configure the cluster UUID.

```
export CLUSTER_UUID=$(curl -s -k -b cookies.txt -H "Content-Type: application/json"
$MGMT_HOST/2/clusters/$CLUSTER_NAME | jq -r '.uuid')
```

Step 7 Configure the Elasticsearch server IP address and port number.

```
export EFK_SERVER=<IP address of Elasticsearch server>
export EFK_PORT=<Port number of Elasticsearch server>
```

Step 8 Install the helm chart to configure the custom Elasticsearch server.

```
curl -s -k -b cookies.txt -X POST --header 'Content-Type: application/json' --header 'Accept:
application/json' -d '{"chart_url": "/opt/ccp/charts/ccp-agent.tgz", "name": "ccpagent", "options":

"ccp-efk.localLogForwarding.enabled=False,ccp-efk.localLogForwarding.elasticsearchHost='$EFK_SERVER',ccp-efk.localLogForwarding.elasticsearchPort='$EFK_PORT'"}'
 $MGMT_HOST/2/clusters/$CLUSTER_UUID/helmcharts
```