Quick Start Guide For Kubernetes And App Service Monitoring

Exported on 08/14/2024

This quick start guide describes how to install and deploy Kubernetes and App Service Monitoring. This installation flow is supported for Amazon Elastic Kubernetes Service (EKS), Azure Kubernetes Service (AKS), Google Kubernetes Engine (GKE), and Red Hat OpenShift.

These are the high-level steps:

1. Sign up and Access the Web Interface
2. Install Kubernetes and App Service Monitoring Using Helm Charts
3. Monitor Your Kubernetes Infrastructure
4. Set up a Health Rule

1. Sign up and Access the Web Interface

1. Contact our team to get started with a free trial or become a customer.
2. You'll receive two emails:
   a. The first email provides a link to set your password. Create a secure password.
   b. The second email includes your tenant URL and license details. Navigate to your tenant URL and sign in with your password.

2. Install Kubernetes and App Service Monitoring Using Helm Charts

1. From the Cisco Cloud Observability UI, navigate to **Configure > Kubernetes and App Services**. Follow the on-screen instructions to download the `operators-values.yaml` and `collectors-values.yaml` files.
2. Create the Cisco AppDynamics `namespace` where Kubernetes and App Service Monitoring is installed:

```
kubectl create namespace appdynamics
```

3. Install cert-manager, which is required by the OpenTelemetry Operator:

```
kubectl apply -f https://github.com/cert-manager/cert-manager/releases/download/v1.8.0/
cert-manager.yaml
```

4. Add the Helm repo:

```
helm repo add appdynamics-cloud-helmcharts https://appdynamics.jfrog.io/artifactory/
appdynamics-cloud-helmcharts/
```

5. Install the Cisco AppDynamics Operators using the `operators-values.yaml` file:

```
helm install appdynamics-operators appdynamics-cloud-helmcharts/appdynamics-operators -n
appdynamics -f operators-values.yaml --wait
```

> If you have already installed your own OpenTelemetry Operator, you must disable the one bundled in Cisco AppDynamics Operators. See Disable the OpenTelemetry Operator.

6. Install the Cisco AppDynamics Collectors using the `collectors-values.yaml` file:

```
helm install appdynamics-collectors appdynamics-cloud-helmcharts/appdynamics-collectors
 -n appdynamics -f collectors-values.yaml
```

For the full list of configuration options, see Cisco AppDynamics Collectors Settings.
To configure the Windows node when using GKE, see Deploy Windows Exporter Pods on Google Kubernetes Engine.

7. Validate the installation.
Check the Kubernetes® pods in the `appdynamics` namespace using the following command:

```
kubectl get all -n appdynamics
```

This is a sample output with the validation:

```
ME                                                                READY   STATUS
RESTARTS   AGE
pod/appdynamics-collectors-appdynamics-clustermon-68756cb9d6-hhjwn    1/1     Running   0
          3m17s
pod/appdynamics-collectors-appdynamics-inframon-fkfh6                 1/1     Running   0
          3m17s
pod/appdynamics-collectors-appdynamics-otel-co-collector-9j6k7        1/1     Running   0
          54s
pod/appdynamics-operators-appdynamics-cloud-operator-669f8f6d4b5b6q   2/2     Running   0
          3m42s
pod/opentelemetry-operator-controller-manager-5cb47c7666-dfplr        2/2     Running   0
          3m42s
pod/appdynamics-operators-appdynamics-smartagent-7888448b58-vz4fm     1/1     Running   0
          3m42s
NAME                                                                      TYPE
CLUSTER-IP       EXTERNAL-IP   PORT(S)                        AGE
service/appdynamics-cloud-operator-metrics-service                        ClusterIP
10.109.232.54    <none>        8443/TCP                       3m42s
service/appdynamics-collectors-appdynamics-otel-co-collector              ClusterIP
10.97.70.222     <none>        4318/TCP,4317/TCP,55679/TCP    54s
service/appdynamics-collectors-appdynamics-otel-co-collector-headless     ClusterIP
None             <none>        4318/TCP,4317/TCP,55679/TCP    54s
service/appdynamics-collectors-appdynamics-otel-co-collector-monitoring   ClusterIP
10.106.157.16    <none>        8888/TCP                       54s
service/appdynamics-otel-collector-service                                ClusterIP
10.100.41.15     <none>        4318/TCP,4317/TCP,55679/TCP    3m20s
```

```
service/opentelemetry-operator-controller-manager-metrics-service         ClusterIP
10.102.222.204    <none>        8443/TCP,8080/TCP             3m42s
service/opentelemetry-operator-webhook-service                            ClusterIP
10.99.26.174      <none>        443/TCP                       3m42s
NAME                                                                 DESIRED   CURRENT
READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/appdynamics-collectors-appdynamics-inframon           1         1
1       1            1           <none>          3m17s
daemonset.apps/appdynamics-collectors-appdynamics-otel-co-collector  1         1
1       1            1           <none>          54s
NAME                                                              READY   UP-TO-DATE
AVAILABLE   AGE
deployment.apps/appdynamics-collectors-appdynamics-clustermon     1/1     1            1
          3m17s
deployment.apps/appdynamics-operators-appdynamics-cloud-operator  1/1     1            1
          3m42s
deployment.apps/opentelemetry-operator-controller-manager         1/1     1            1
          3m42s
deployment.apps/appdynamics-operators-appdynamics-smartagent      1/1     1            1
          3m42s
NAME                                                                        DESIRED
CURRENT   READY   AGE
replicaset.apps/appdynamics-collectors-appdynamics-clustermon-68756cb9d6    1         1
        1       3m17s
replicaset.apps/appdynamics-operators-appdynamics-cloud-operator-669f8f6d49 1         1
        1       3m42s
replicaset.apps/opentelemetry-operator-controller-manager-5cb47c7666        1         1
        1       3m42s
replicaset.apps/appdynamics-operators-appdynamics-smartagent-7888448b58     1         1
        1       3m42s
```

For a comprehensive installation guide, see Install Kubernetes and App Service Monitoring Using Helm Charts.

3. Monitor Your Kubernetes Infrastructure

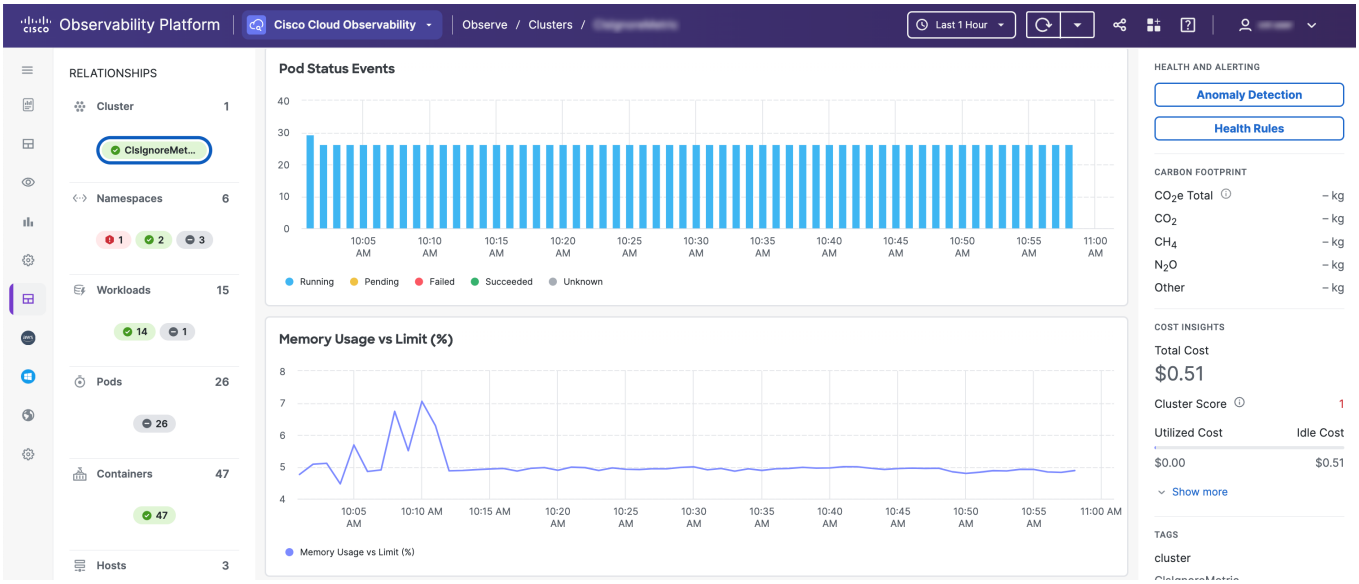
After 5-10 minutes, Cisco Cloud Observability will populate the **Observe** page with a **Kubernetes** domain.

This domain contains links to entity-centric pages (ECPs), which are UI pages that can be used to monitor your services. ECPs display everything of relevance (e.g., metrics, metadata, health status, events, logs, relationships) for a given entity.

To navigate to your Kubernetes ECPs:

1. Sign in to Cisco Cloud Observability.
2. On the **Observe** page, navigate to the **Kubernetes** domain.
3. Click an entity to navigate to the list view, which displays a list of all of the entities of that type.
4. From the list view, click an entity name to navigate the detail view for that instance.
   The detail view displays the metrics, key performance indicators, properties (attributes), and other data related to the instance you selected. For the full list of data that can be monitored, see

Kubernetes Entities.



4. Set up a Health Rule

To monitor one entity or a group of entities:

1. In the Cisco Cloud Observability UI, click **Configure > Health Rules**.
2. Click **Create Health Rule**.
3. Follow the on-screen instructions to create the health rule. For more information on health rules, see Configure a Health Rule.

You can view the health violation details for a selected entity or any health rule in the health violation timeline. To access the health violation timeline:

1. On the **Observe** page, select a required time period and then select an entity type from the available domains. For example, in the Application Performance Monitoring domain, you can select the entity type **Services**.
2. From the **List** view, click an entity name. The details of health violations, endpoints, metrics, and logs corresponding to the selected entity appears.
3. Click **Entity Health Timeline** to view a list of all alerts associated with the selected entity. An alert triggered by a health rule is displayed with the type **Alert**. The health rules violate when there are any violating metrics, violating events, or violating logs based on the health rule conditions.