Quest™

Foglight® for Container Management 3.0.0

User and Administration Guide

Legend

■ **WARNING: A WARNING icon indicates a potential for property damage, personal injury, or death.**

! **CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.**

i **IMPORTANT NOTE**, **NOTE**, **TIP**, **MOBILE**, or **VIDEO:** An information icon indicates supporting information.

Foglight for Container Management User and Administration Guide
Updated - March 2020
Foglight Version - 5.9.7
Software Version - 3.0.0

Contents

1

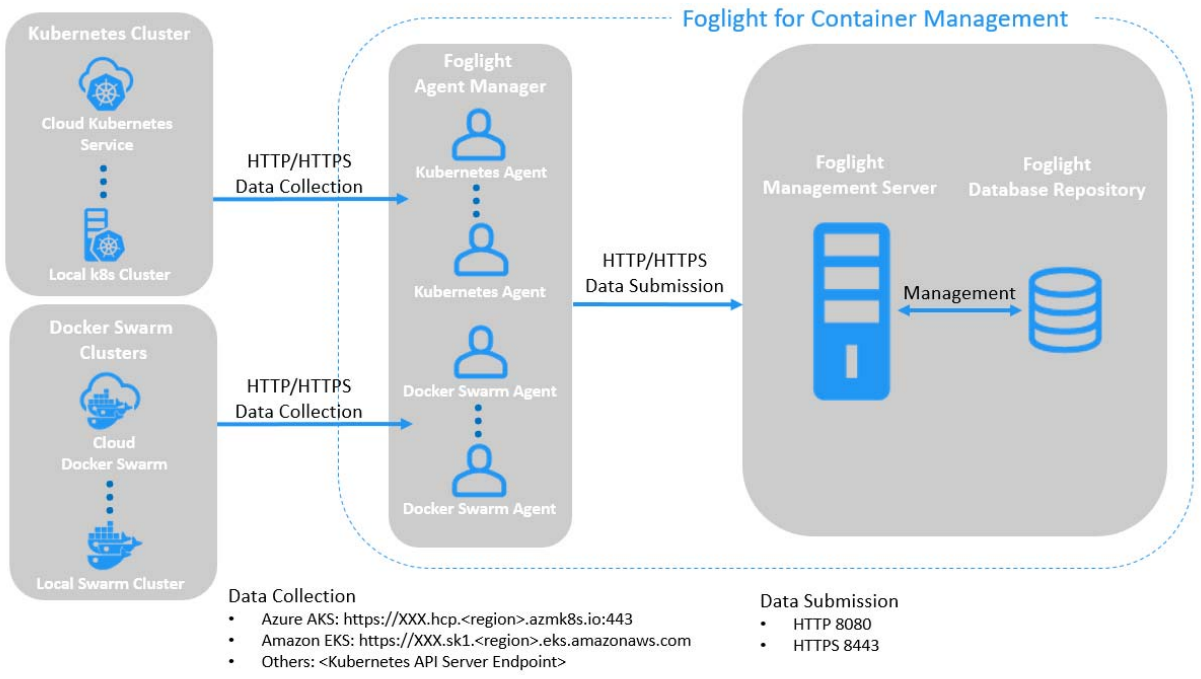
Understanding Foglight for Container Management

About Foglight for Container Management

Containers are a method of operating system virtualization that allow you to run an application and its dependencies in resource-isolated processes. Foglight® for Container Management simplifies this process by tracking each container, the resources it consumes, and the remaining compute of the container host, as well as providing you with the cluster information and pre-configured rules with notifications identifying the problem of your clusters.

Architecture

Figure 1. Components of Foglight for Container Management


Foglight for Container Management
Kubernetes Cluster
Cloud Kubernetes Service
Local k8s Cluster
Docker Swarm Clusters
Cloud Docker Swarm
Local Swarm Cluster
HTTP/HTTPS Data Collection
HTTP/HTTPS Data Collection
Foglight Agent Manager
Kubernetes Agent
Kubernetes Agent
Docker Swarm Agent
Docker Swarm Agent
HTTP/HTTPS Data Submission
Foglight Management Server
Foglight Database Repository
Management
Data Collection
Azure AKS: https://XXX.hcp.<region>.azmk8s.io:443
Amazon EKS: https://XXX.sk1.<region>.eks.amazonaws.com
Others: <Kubernetes API Server Endpoint>
Data Submission
HTTP 8080
HTTPS 8443

Foglight for Container Management consists of three main components:

- Foglight Management Server and Foglight Database Repository — Responsible for managing, alerting, and viewing the collected data. Both components can be set to run on the same machine or reside on separate machines.
- Agent Manager — Hosts the monitoring Kubernetes agents.
- Docker Swarm clusters — Manages containerized applications in a clustered environment.
- Kubernetes clusters — Manages containerized applications in a clustered environment.

Sizing Your Monitored Environment

Consider the possibility of a great amount of objects being collected, analyzed, and maintained by the application, several aspects of the underlying server must be taken into account. The sizing of the supporting clusters and containers depends on the complexity of the underlying environment. Sufficient processing power and CPU memory are required to support effective collection, server data handling, and analytics.

NOTE: Currently Quest validates the environment with up to 10000 containers. If your environment beyonds this scale, contact Quest Support.

Foglight Management Server Requirements

The minimum system requirements of the Foglight Management Server vary from the scale of clusters. The scale of clusters is determined by running containers.

Table 1. Foglight Management Server requirements

Operating System	Maximum Containers	Foglight		Agent Manager	
		JVM Settings	**# of CPUs**	**JVM Settings**	**# of CPUs**
Windows 64-bit	1000	Xms|Xmx=4G	2	Xms|Xmx=4G	2
	5000	Xms|Xmx=8G	4	Xms|Xmx=8G	4
	10000	Xms|Xmx=12G	6	Xms|Xmx=12G	6
Linux 64-bit	1000	Xms|Xmx=4G	2	Xms|Xmx=4G	2
	5000	Xms|Xmx=8G	4	Xms|Xmx=8G	4
	10000	Xms|Xmx=12G	6	Xms|Xmx=12G	6

If you are using an embedded Agent Manager, make sure to use the sum resources of both Foglight and Agent Manager.

Kubernetes Agent Requirements

Kubernetes Agent collects inventory and metrics every 5 minutes by default. Refer to Configuring data collection interval for details about how to change the collection interval.

Table 2. Kubernetes Agent requirements

Maximum Containers	Kubernetes Agent Collection Interval (minutes)	
	Inventory	**Metrics**
500	5	5
1000	10	10
5000	30	30
10000	60	60

Table 2 is the recommendations for local Kubernetes clusters. If you deploy Kubernetes clusters on the Cloud Provider Kubernetes Service, consider your network rate and change your configurations based on different Cloud Provider and different region/zone of your cluster.

Docker Swarm Agent Requirements

Docker Swarm Agent collects inventory and metrics every 5 minutes by default. Refer to Configuring data collection interval for details about how to change the collection interval.

Table 3. Docker Swarm Agent requirements

Maximum Containers	Docker Swarm Agent Collection Interval (minutes)	
	Inventory	**Metrics**
500	5	5
1000	10	10
5000	30	30

Table 3 is the recommendations for local Docker Swarm clusters. For cloud environment, consider network rate and change configurations based on different Cloud Provider and different region/zone.

Getting Started

Prerequisite

Kubernetes Agent

Each Kubernetes Agent monitors the assets inside the selected Kubernetes Service Providers. To enable the data collection, complete the following prerequisites before create agent.

- Preparing the Kubernetes credential
- Enabling Heapster service in monitored environment

Preparing the Kubernetes credential

The Kubernetes configuration file named *KubeConfig* is a standard configuration of Kubernetes and is required for Kubernetes agents to access the cluster. Foglight for Container Management verifies and supports the local Kubernetes and the following Cloud Kubernetes Service Providers. Based upon your environment, select either of approaches to get your *KubeConfig* file:

NOTE: Data from different Kubernetes Agents with the same cluster name will be merged into one cluster.

- Local Kubernetes
- Azure Kubernetes Service (AKS)
- Amazon Elastic Container Service for Kubernetes (EKS)
- Google Cloud Platform Container Engine (GKE)
- IBM Cloud Kubernetes Service
- Openshift Origin

Local Kubernetes

If you build a Kubernetes cluster locally, find this *KubeConfig* file under the `/etc/kubernetes/admin.kubeconfig` on your master node.

Azure Kubernetes Service (AKS)

Before generating the Kubernetes credentials, record the following information:

- Azure Username
- Azure Password
- Azure Subscription Number
- The name of your AKS Cluster Resource Group
- The name of your AKS cluster

Download the Azure Command Line Interface and install it in your local platform, and then follow steps below to generate your Kubernetes credential:

1. Run the command `az login`.

   Then a browser shows up, directing you to the Azure Portal where you should enter your Azure Username and Password to complete the authentication.

2. Run the command: `az account set --subscription <azure subscription number>`
3. Run the command: `az aks get-credentials --resource-group <azure resource group name> --name <azure cluster name>`
4. Find the Kubernetes configuration file under `<USER_HOME>/.kube/config` on your local platform.

   NOTE: The token in this Kubernetes configuration file will get expired after two years. If you don't want the credential gets expired, refer to Foglight Container Tools for detail.

Amazon Elastic Container Service for Kubernetes (EKS)

Follow the Amazon EKS offical guide Getting Started with Amazon EKS. Follow the guide and complete Create a kubeconfig for Amazon EKS. in the end of the guide.

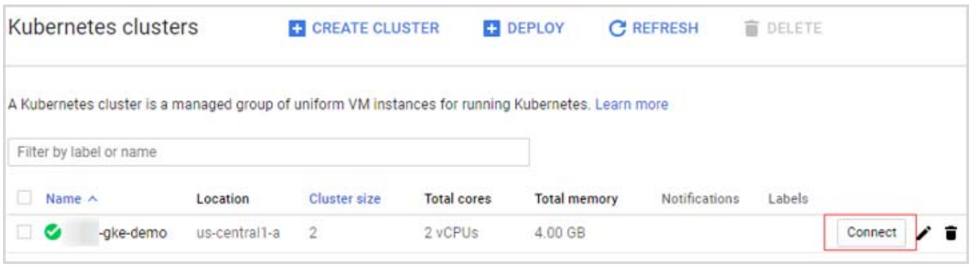
NOTE: If you don't want the credential gets expired, refer to Foglight Container Tools for detail.

Google Cloud Platform Container Engine (GKE)

Download the Google Cloud Client tool and install it in your local platform, and then follow steps below to generate your Kubernetes credential:

1. Generate the intermediate Kubernetes credential for your cluster.

   a. Log in to your Kubernetes cluster, click **Connect** next to your cluster name.

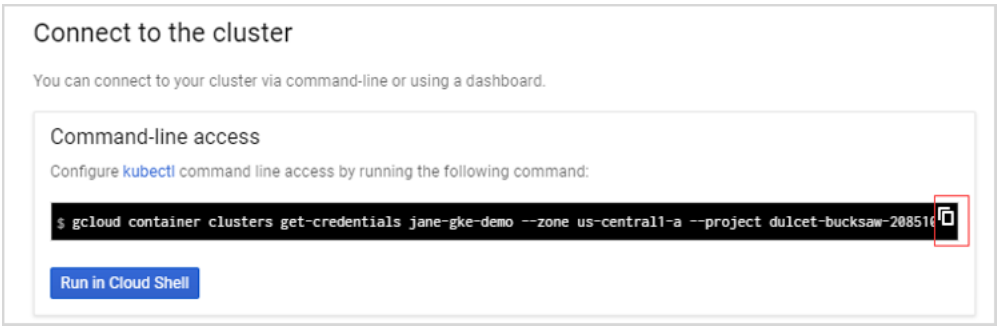


Kubernetes clusters

A Kubernetes cluster is a managed group of uniform VM instances for running Kubernetes. Learn more

Name	Location	Cluster size	Total cores	Total memory	Notifications	Labels
-gke-demo	us-central1-a	2	2 vCPUs	4.00 GB		

   b. Click to copy the command below, and then run this command.


Connect to the cluster

You can connect to your cluster via command-line or using a dashboard.

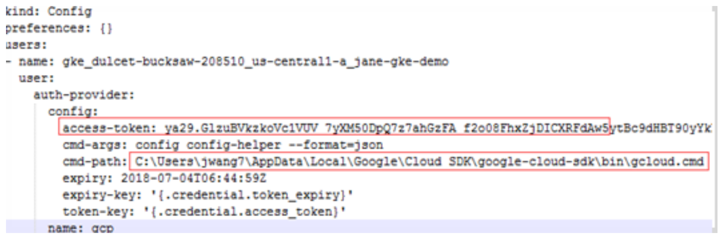
Command-line access

Configure kubectl command line access by running the following command:

```
$ gcloud container clusters get-credentials jane-gke-demo --zone us-central1-a --project dulcet-bucksaw-20851
```

c Find the intermediate Kubernetes configuration file under `<USER_HOME>/.kube/config` on your local platform. The following is the example of this intermediate Kubernetes configuration file.

> **NOTE:** This Kubernetes configuration file cannot be used as the agent credential because the token in this file will get expired soon and "*cmd-path*" of the token directs to your local platform.



```
kind: Config
preferences: {}
users:
- name: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
  user:
    auth-provider:
      config:
        access-token: ya29.GlzuBVkzkoVc1VUV 7yXM50DpQ7z7ahGzFA f2o08FhxZjDICXRFdAw5ytBc9dHBT90yYk
        cmd-args: config config-helper --format=json
        cmd-path: C:\Users\jwang7\AppData\Local\Google\Cloud SDK\google-cloud-sdk\bin\gcloud.cmd
        expiry: 2018-07-04T06:44:59Z
        expiry-key: '{.credential.token_expiry}'
        token-key: '{.credential.access_token}'
      name: gcp
```

d Open Google Cloud Client tool and run the following commands to create a Kubernetes service account that grants with the *cluster-admin* role and the access to your Google Kubernetes Engine (GKE) cluster.

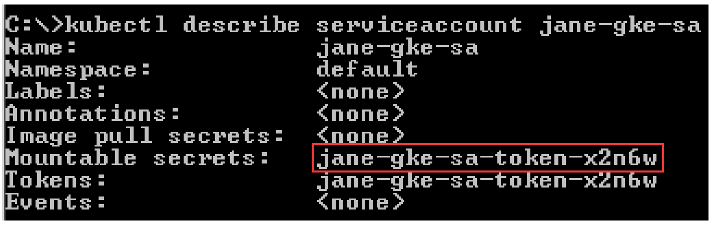
  a *kubectl create serviceaccount <service account name>*

  b *kubectl create clusterrolebinding <cluster role binding name> --clusterrole=cluster-admin -serviceaccount=default:<service account name>*

  "default" in the above command is the namespace name of this service account name. The name space name will be "default" if you do not change it. You can also change to other namespace names, as needed.

  c *kubectl describe serviceaccount <service account name>*

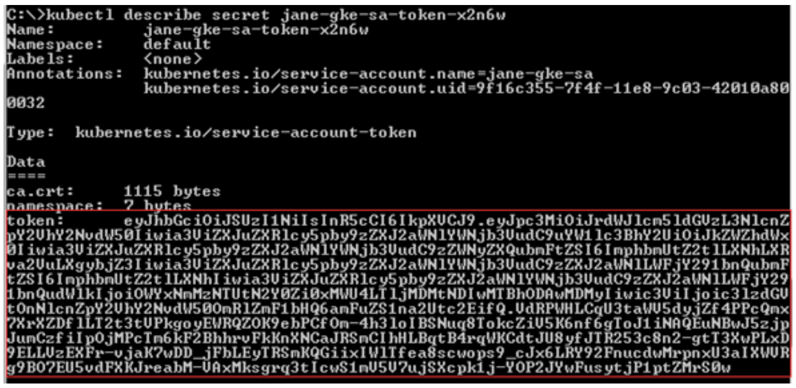
  You will get the response similar as below. Record the <secret name> for later use.



```
C:\>kubectl describe serviceaccount jane-gke-sa
Name:                jane-gke-sa
Namespace:           default
Labels:              <none>
Annotations:         <none>
Image pull secrets:  <none>
Mountable secrets:   jane-gke-sa-token-x2n6w
Tokens:              jane-gke-sa-token-x2n6w
Events:              <none>
```

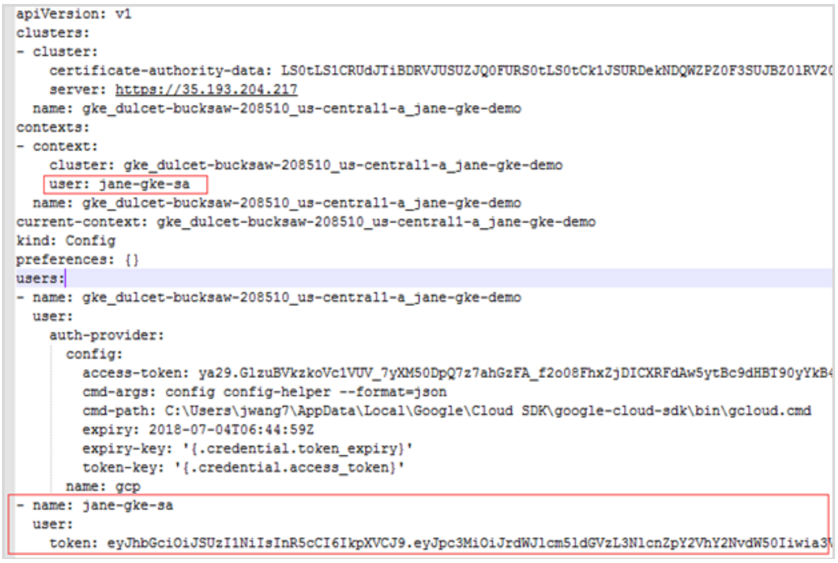
  d *kubectl describe secret <secret name>*

  You will get response similar as below. Record the token value (exclude "token:") for later use.



```
C:\>kubectl describe secret jane-gke-sa-token-x2n6w
Name:         jane-gke-sa-token-x2n6w
Namespace:    default
Labels:       <none>
Annotations:  kubernetes.io/service-account.name=jane-gke-sa
              kubernetes.io/service-account.uid=9f16c355-7f4f-11e8-9c03-42010a80
0032

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1115 bytes
namespace:  7 bytes
token:      eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZ
pY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx
0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6ImphbmUtZ2tlLXNhLXR
va2VuLXgybjZ3Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmF
tZSI6ImphbmUtZ2tlLXNhIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY29
1bnQudWlkIjoiOWYxNmMzNTUtN2Y0Zi0xMWU4LTljMDMtNDIwMTBhODAwMDMyIiwic3ViIjoic3lzdGV
tOnNlcnZpY2VhY2NvdW50OmRlZmF1bHQ6amFuZS1na2Utc2EifQ.VdRPWHLCqU3taWU5dyjZf4PPcQmx
7XrXZDf1LT2t3tVPkgoyEWRQZOK9ebPCfOm-4h3loIBSNuq8TokcZiV5K6nf6gToJ1iNAQEuNBwJ5zjp
JumCzfiIpOjMPcTm6kF2BhhrvFkKnXNCaJRSmCIhHLBqtB4rqWKCdtJU8yfJTR253c8n2-gtT3XwPLxD
9ELLVzEXFr-vjaK7wDD_jFbLEyTRSmKQGiixIWlTfea8scwops9_cJx6LRY92FnucdwMrpnxU3aIXWVR
g9BO7EU5vdFXKJreabM-VAxMksgrq3tIcwS1mV5V7ujSXcpk1j-YOP2JYwFusytjP1ptZMrS0w
```

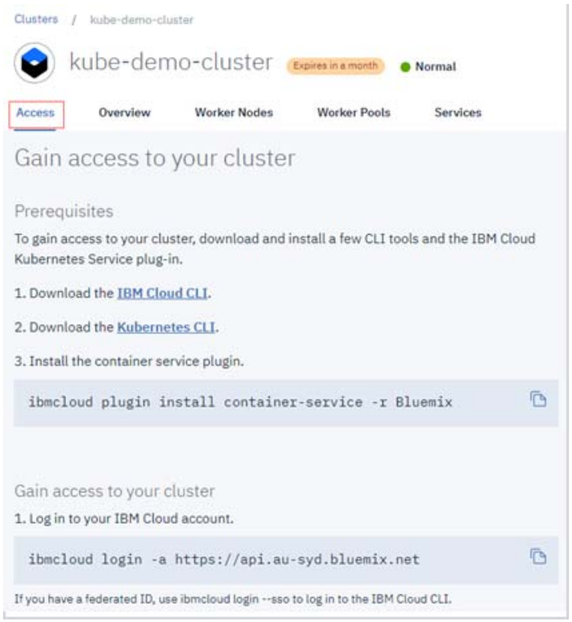
e   Open the intermediate Kubernetes configuration file under <USER_HOME>/.kube/config, and then add the user and change the token to the new one.



```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURDekNDQWZPZ0F3SUJBZ01RV2(
    server: https://35.193.204.217
  name: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
contexts:
- context:
    cluster: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
    user: jane-gke-sa
  name: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
current-context: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
kind: Config
preferences: {}
users:
- name: gke_dulcet-bucksaw-208510_us-central1-a_jane-gke-demo
  user:
    auth-provider:
      config:
        access-token: ya29.GlzuBVkzkoVc1VUV_7yXM50DpQ7z7ahGzFA_f2o08FhxZjDICXRFdAw5ytBc9dHBT90yYkB(
        cmd-args: config config-helper --format=json
        cmd-path: C:\Users\jwang7\AppData\Local\Google\Cloud SDK\google-cloud-sdk\bin\gcloud.cmd
        expiry: 2018-07-04T06:44:59Z
        expiry-key: '{.credential.token_expiry}'
        token-key: '{.credential.access_token}'
      name: gcp
- name: jane-gke-sa
  user:
    token: eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3\
```

IBM Cloud Kubernetes Service

If you have created your cluster on IBM Cloud Kubernetes Service, get the access from the console as described on the cluster’s *Access* view. You will get a .pem file and a .yml file after you performing the steps.



Gain access to your cluster

Prerequisites

To gain access to your cluster, download and install a few CLI tools and the IBM Cloud Kubernetes Service plug-in.

1. Download the IBM Cloud CLI.

2. Download the Kubernetes CLI.

3. Install the container service plugin.

```
ibmcloud plugin install container-service -r Bluemix
```

Gain access to your cluster

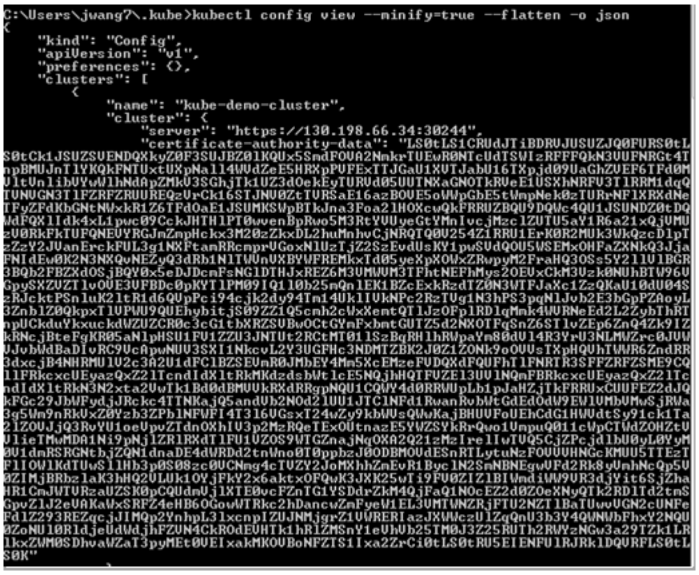
1. Log in to your IBM Cloud account.

```
ibmcloud login -a https://api.au-syd.bluemix.net
```

If you have a federated ID, use ibmcloud login --sso to log in to the IBM Cloud CLI.

By default IBM Cloud Kubernetes Service uses certificate authority file and token/refresh token. However, certificate authority data and service account token should be used in the Kubernetes Agent credential. After you successfully test your connection through "kubectl get nodes", follow the steps below to generate the Kubernetes Agent credential.

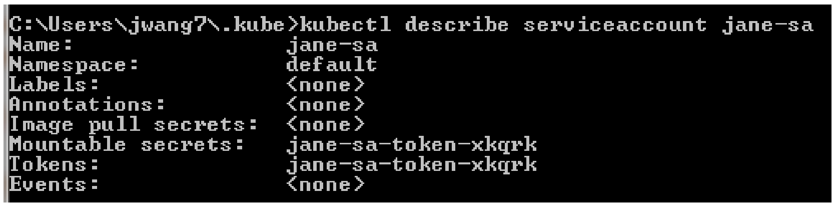
1 Run the command *kubectl config view -minify=true -flatten -o json*. You will get an output similar as below, then record the *<certificate authority data>* for later use.



```
C:\Users\jwang7\.kube>kubectl config view --minify=true --flatten -o json
{
    "kind": "Config",
    "apiVersion": "v1",
    "preferences": {},
    "clusters": [
        {
            "name": "kube-demo-cluster",
            "cluster": {
                "server": "https://130.198.66.34:30244",
                "certificate-authority-data": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tL
S0tCk1JSUZSVENDQXkyZ0F3SUJBZ01KQUx5SmdFOVA2NmkrTUEwR0NTcUdTSWIzRFFFQkN3VUFNRGt4T
npBMUJnTl1YKQkFNTUxtUXpNall4WVdZeE5HRXpPVFExTTJGaU1XVTJabU16TXpjd09UaGhZVEF6TFd0M
UltVnlibVYwWlhNdApZMkU3SGhjTk1UZ3dOekEyTURVd05UUTNXaGNOTkRVeE1USXhNRFU3T1RRM1dqQ
TVNVGN3T1FZRFZRUUREQzVrCk16STJNV0ZtTVRSaE16azBOVE5oWWpGbE5tWmpNek0zTURrNF1XRXdNe
TFyZFdKbGNtNWxkR1Z6TFd0aE1JSUMKSWpBTkJna3Foa2lHOXcwQkFRRUZBQU9DQWc4QU1JSUNDZ0tDQ
WdFQX1Idk4xL1pwc09CckJHTH1PT0wvenBpRwo5M3RtYVUyeGtYMnIvcjMzc1ZUTU5aY1R6a21xQjVMU
zV0RkFkTUFQNEVYRGJmZmpHckx3M20zZkxDL2huMnhvCjNRQTQ0V254Z1RRU1ErK0R2MUk3WkQzcDlpT
zZzY2JVanErckFUL3g1NXFtamRRcmprVGoxNlUzTjZ2SzEvdUsKY1pwSVdQOU5WSEMxOHFaZXNkQ3Jja
FNIdEw0K2N3NXQvNEZyQ3dRb1N1TWUnVXBYWFREMkxTd05yeXpXOWxZRwpyM2FraHQ3OSs5Y2llV1BGR
3BQb2FBZXdOSjBQY0x5eDJDcmFsNGlDTHJxREZ6M3VMWVM3TFhtNEFhMys2OEVxCkM3Vzk0NUhBTW96V
GpySXZUZTlvOVE3VFBDc0pKYTlPM09IQ1l0b25mQnlEK1BZcExkRzdTZ0N3WTFJaXc1ZzQKaU10dU04S
zRJcktPSnluK2ltR1d6QVpPci94cjk2dy94Tm14UklIVkNPc2RzTVg1N3hPS3pqNlJvb2E3bGpPZAoyL
3Znb1Z0QkpxTlVPWU9QUEhybitjS09ZZ1Q5cmh2cWxXemtQTlJzOFplRDlqMmk4WURNeEd2L2ZybThRT
npUCkduYkxuckdWZUZCR0c3cG1tbXRZSVBwOCtGYmFxbmtGUTZ5d2NXOTFqSnZ6STlvZEp6ZnQ4Zk9IZ
kRNcjBteFgKR05aNlpHSU1FV1ZZU3JNTUt2RCtMT01lSzBqRHlhRWpaYm80dVl4R3YrU3NLMWZrc0JUW
UJvbWdBaDIvRC9VcApwNUV3SXI1NkcvL2Y3UGFHc3NDMTZBK2J0Z1ZONk9oOVVsTXpHQVhTWWR6ZndRR
3dxcjB4NHRMU1V2c3A2U1dFClBZSEVmR0JMbEY4Mm5XcEMzeFVDQXdFQUFhT1FNRTR3SFFZRFZSME9CQ
l1FRkcxcUEyazQxZ21TcndIdX1tRkMKdzdsbWtlcE5NQjhHQTFVZE13UV1NQmFBRkcxcUEyazQxZ21Tc
ndIdX1tRkN3N2xta2VwTk1Bd0dBMVVkRXdRRgpNQU1CQWY4d0RRWUpLb1pJaHZjTkFRRUxCUUFEZ2dJQ
kFGc29JbWFydjJRckc4TTNKajQ5andVb2NOd21UU1JTC1NFd1RwanRvbWtGdEdOdW9EW1VMbVMwSjRWa
3g5Wm9nRkVxZ0Yzb3ZPblNFWFI4T316VGsxT24wZy9kbWVsQWwKajBHUVFoUEhCdG1HWVdtSy91ck1Ta
21ZOVJjQ3RvYU1oeVpvZTdnOXhIV3p2MzRQeTExOUtnazE5YWZSYkRrQwo1VmpuQ011cWpCTWdZOHZtV
UlieTMwMDA1Ni9pNjlZR1RXdT1FU1VZOS9WTGZnajNqOXA2Q21zMzIrellIwTVQ5CjZPcjdlbU0yL0YyM
0V1dmRSRGNtbjZQN1dnaDE4dWRDd2tnWno0T0ppbzJ0ODBMOVdESnRTLytuNzFOVVVHNGcKMUU5TTEzT
F1IOW1KdTUwS11Hb3p0S08zc0VCNmg4cTVZY2JoMXhhZmEvR1BycUN2SmNBNEgwVFd2Rk8yUmhNcQp5U
0ZIMjBRbzlaK3hHQ2VLUk1OYjFkY2x6aktxOFQwK3JXK25wTi9FV0ZIZ1BIWmdiWW9VR3djYit6SjZha
HR1CmJWTVRzaUZSK0pCQUdmVj1XTE0vcFZnTG1YSDdrZkM4QjFaQ1NOcEZ2d0ZOeXNyQTk2RD1Td2tmS
GpvZ1J2eUAKaWxSRFZ4eHB6OGowWTRkc2hDancwZmFyeW1EL3UMTWNZRjFTU2NZT1BaTUwvUGN2cUNFe
Fd1Z293REZqcjJIMQp2YnhpL31xcnpIZUJNMjgrZ1VWRERIazJXWWczU1ZqQnU3b3Y4QWNWbFhxY2NQU
0ZoNU10R1djeUdWdjhFZUN4CkROdEVHTk1hR1ZMSnY1eVhVb25TM0J3Z25RUTh2RWYzNGw3a29TZk1LR
1kxZWM0SDhvaWZaT3pyMEt0VEIxakMKOVBoNFZTS1Ixa2ZrCi0tLS0tRU5EIENFU1RJRk1DQVRFLS0tL
S0K"
```

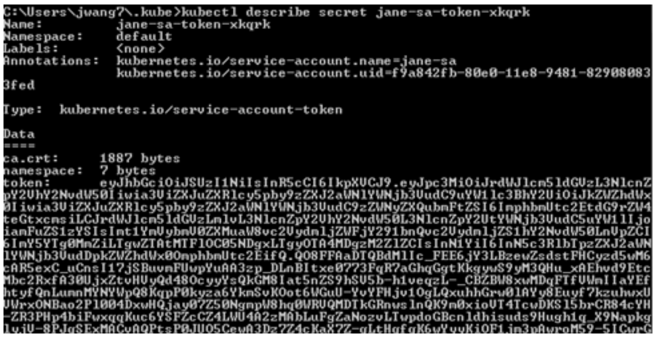
2 Run the command *kubectl create serviceaccount <service account>*.

3 Run the command *kubectl describe serviceaccount <service account>*. You will get a response similar as below, then record <service account secret> (in this sample, it is jane-sa-token-xkqrk) for later use.



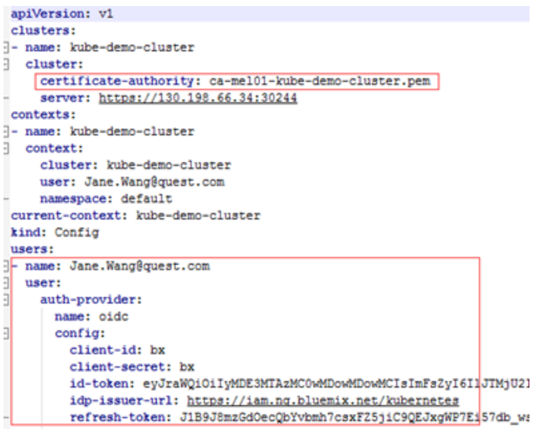
```
C:\Users\jwang7\.kube>kubectl describe serviceaccount jane-sa
Name:                    jane-sa
Namespace:               default
Labels:                  <none>
Annotations:             <none>
Image pull secrets:      <none>
Mountable secrets:       jane-sa-token-xkqrk
Tokens:                  jane-sa-token-xkqrk
Events:                  <none>
```

4 Run the command *kubectl describe secret <service account secret>*. You will get a response similar as below, then record <service account token> for later use.



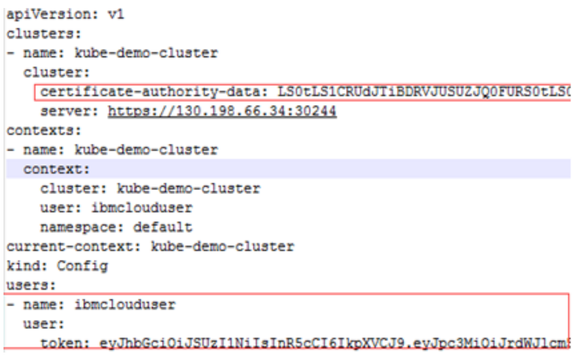
```
C:\Users\jwang7\.kube>kubectl describe secret jane-sa-token-xkqrk
Name:         jane-sa-token-xkqrk
Namespace:    default
Labels:       <none>
Annotations:  kubernetes.io/service-account.name=jane-sa
              kubernetes.io/service-account.uid=f9a842fb-80e0-11e8-9481-82908083
3fed

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1887 bytes
namespace:  7 bytes
token:      eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZ
pY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx
0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6ImphbmUtc2EtdG9rZW4
teGtxcmsiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC5uYW1lIjo
iamFuZS1zYSIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VydmljZS1hY2NvdW50LnVpZCI
6ImY5YTg0MmZiLTgwZTAtMTFlOC05NDgxLTgyOTA4MDgzM2ZlZCIsInN1YiI6InN5c3RlbTpzZXJ2aWN
lYWNjb3VudDpkZWZhdWx0OmphbmUtc2EifQ.QO8FFAaDTQBdMlIc_FEE6jY3LBzewZsdstFHCyzd5wM6
cAR5exC_uCnsI17jSBuvmFUwpYuAA3zp_DLnBItxe0773FqR7aGhqGgtKkgywS9yM3QHu_xAEhvd9Etc
Mbc2RxfA30UjxZtvHUyQd48OcyyYsQkGM8Iat5nZS9hSV5b-h1veqzL-_CBZBW8xwMDqFTfUWmIIaYEf
htyfQnLumnMYNYWpQ8KqpP0kyza6YkmSvKOot6WGuU-YvYFHjv1OgLQxuhhGrw0lAYy8Euyf7kzuhwxU
UWrxONBao2P1004DxwHQjay07Z50NgmpW8hq0WRUQMDTkGRnws1nQK9m0xioUT4TcwDKS15brCR84cYH
-ZR3PHp4biFwxqqKuc6YSFZcCZ4LWU4A2zMAbLuFgZaNozvLTwpdoGBcn1dhisuds9Hugh1q_X9Napkg
1vjU-8PJqSExMACvAQPtsP0JUO5CewA3Dz7Z4cKaX7Z-gLtHgfgK6wYvvKiOF1jm3pAwroM59-5ICwrG
```

5 Open the .yml file generated previously, which looks like below.



```
apiVersion: v1
clusters:
- name: kube-demo-cluster
  cluster:
    certificate-authority: ca-mel01-kube-demo-cluster.pem
    server: https://130.198.66.34:30244
contexts:
- name: kube-demo-cluster
  context:
    cluster: kube-demo-cluster
    user: Jane.Wang@quest.com
    namespace: default
current-context: kube-demo-cluster
kind: Config
users:
- name: Jane.Wang@quest.com
  user:
    auth-provider:
      name: oidc
      config:
        client-id: bx
        client-secret: bx
        id-token: eyJraWQiOiIyMDE3MTAzMC0wMDowMDowMCIsImFsZyI6IlJTMjU2I
        idp-issuer-url: https://iam.ng.bluemix.net/kubernetes
        refresh-token: J1B9J8mzGdOecQbYvbmh7csxFZ5jiC9QEJxgWP7Ei57db_ws
```

6 Change the certificate authority to the data <certificate authority data> of this authority and change the users section to use <service account token>. Save your changes, and then you will get a credential file like below. This file will be used as the Kubernetes Agent credential to connect to your IBM cloud Kubernetes service cluster.



```
apiVersion: v1
clusters:
- name: kube-demo-cluster
  cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS
    server: https://130.198.66.34:30244
contexts:
- name: kube-demo-cluster
  context:
    cluster: kube-demo-cluster
    user: ibmclouduser
    namespace: default
current-context: kube-demo-cluster
kind: Config
users:
- name: ibmclouduser
  user:
    token: eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm
```

Openshift Origin

If you could access the `/etc/origin/master/admin.kubeconfig` on the master node, download this file which can be used as the Kubernetes Agent credential.

If you could not access the `/etc/origin/master/admin.kubeconfig` on the master node, follow instructions below to generate a permanent credential file.

Before generating the permanent Kubernetes credentials, record the following information and ensure you have granted the privilege for accessing the cluster-wide resources:

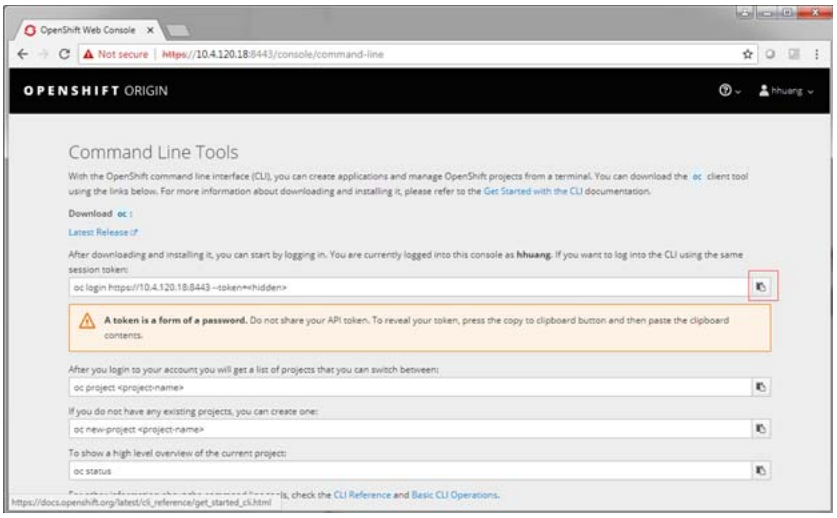
- Openshift Username
- Openshift Password

Download the Openshift Command Line Interface and install it in your local platform, and then follow steps below to generate your Kubernetes credential:

1 Log into Openshift and generate an intermediate Kubernetes configuration file.

    1 After logging into Openshift, click **Command Line Tools** on the upper right.

    2 Click the button next to the *Session token* field, copy the command, and then paste it in your local Command Line Tool. Make sure to find the intermediate Kubernetes configuration file under *`<USER_HOME>/.kube/config`* on your local platform.



OpenShift Web Console

Not secure | https://10.4.120.18:8443/console/command-line

OPENSHIFT ORIGIN

Command Line Tools

With the OpenShift command line interface (CLI), you can create applications and manage OpenShift projects from a terminal. You can download the oc client tool using the links below. For more information about downloading and installing it, please refer to the Get Started with the CLI documentation.

Download oc :

Latest Release

After downloading and installing it, you can start by logging in. You are currently logged into this console as **hhuang**. If you want to log into the CLI using the same session token:

```
oc login https://10.4.120.18:8443 --token=<hidden>
```

A token is a form of a password. Do not share your API token. To reveal your token, press the copy to clipboard button and then paste the clipboard contents.

After you login to your account you will get a list of projects that you can switch between:

```
oc project <project-name>
```

If you do not have any existing projects, you can create one:

```
oc new-project <project-name>
```

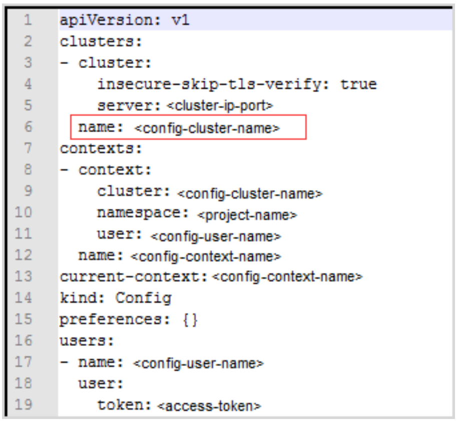
To show a high level overview of the current project:

```
oc status
```

…ls, check the CLI Reference and Basic CLI Operations.

https://docs.openshift.org/latest/cli_reference/get_started_cli.html

    3 On your local platform, browse to open this configuration file. You may see the context similar to the following. Record ***<config-cluster-name>*** for later use.



```
apiVersion: v1
clusters:
- cluster:
    insecure-skip-tls-verify: true
    server: <cluster-ip-port>
  name: <config-cluster-name>
contexts:
- context:
    cluster: <config-cluster-name>
    namespace: <project-name>
    user: <config-user-name>
  name: <config-context-name>
current-context: <config-context-name>
kind: Config
preferences: {}
users:
- name: <config-user-name>
  user:
    token: <access-token>
```

2 The token generated in step 1 will be expired after 4 hours, however Foglight for Container Management needs a permanent Kubernetes credential. So you need to create a service account with "**cluster-admin**" role, and then get the authorization code (not expired) of this service account to generate our permanent Kubernetes credential.

    1 Run the command *`oc project`* ***`<project-name>`***.

    2 Run the command *`oc create serviceaccount`* ***`<service-account-name>`***.

    You can check if your service account has been created successfully using the command: *`kubectl get serviceaccounts`*

    3 Run the command *`oc serviceaccounts get-token`* ***`<service-account-name>`***. Then you will get a token <*service-account-token*> like below. Record this token for later use.

    "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6Im9zLWFkb

```
WluLXRva2VuLWY0a2ZsIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLW
FjY291bnQubmFtZSI6Im9zLWFkbWluIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9
zZXJ2aWNlLWFjY291bnQudWlkIjoiODMzNGU0NTQtNzQ1Yy0xMWU4LWFmNmEtMDA1MDU2YjY3
NDFhIiwic3ViIjoic3lzdGVtOnNlcnZpY2VhY2NvdW50OmRlZmF1bHQ6b3MtYWRtaW4ifQ.RW
H_AoXy2U1elkHN_Bs9IR1xo0zNCJlwcY0h3zuQnrkOFi8gVpX1I77uhAPp7oIjPqDSWkUAN9F
6mP_tNdGwJsqRmHYEMOtCLnnIM61BYxIcABvwr66aOZ3Gn0D7EM5M_7XgKDCl6ON3W5NaH0D8
DpVTYqxkQ49u3qt4gqrcjVCaSsDNWlgGxY4KOIDrUbKkdgaRKzeD9o4Bv9VbYICqyxwoUebku
JAcHiXGIcSe-ozS_zroPi1tT5HW-RY0Pn3Fp3zBnydiokna0-mXot5lqoYc-
R6E1U9YSrAOhWm9Q8ipiut6OczXbmLPM4DYve6dmHi_j5FquCqhod-QlA7aPw"
```

4. Run the following command to grant your service account with the “**cluster-admin**” privilege:
   kubectl create clusterrolebinding ***<cluster-role-binding-name>*** *-- clusterrole=cluster-admin --serviceaccount=default:****<service-account-name>***.

3. Generate a permanent Kubernetes configuration file and save it under *<USER_HOME>/.kube/config file/credential*.
   1. Open and edit the intermediate configuration file.
   2. Use kubectl to add user credentials, create new context, in the end change the existing contexts to the ones that you added in step 2. For example,

      ```
      kubectl config set-credentials <credential-name> --token=<service-
      account-token>
      ```

      ```
      kubectl config set-context <new-context-name> --cluster=<config-cluster-
      name> --user=<credential-name> --namespace=<project-name>
      ```

      ```
      kubectl config use-context <new-context-name>
      ```
   3. Save the current Kubernetes configuration file.

Enabling Heapster service in monitored environment

There are various approaches to enable Heapster on your Kubernetes cluster. Visit Heapster official website to determine the approach that you are going to deploy your Heapster service, or you can follow instructions in https://github.com/foglight/container to deploy your service.

Some of the cloud platform Kubernetes service has enabled Heapster service for the cluster. If you have connected to the cluster, run the following command to check: `kubectl cluster-info`

Enabling Prometheus service in monitored environment

There are various approaches to enable Prometheus on your Kubernetes cluster. Visit Prometheus Official Website to determine the approach that you are going to deploy your Prometheus service. Or you can refer to configuration management systems, such as helm (https://github.com/helm/charts/tree/master/stable/prometheus) or ansible (https://github.com/cloudalchemy/ansible-prometheus) to simplify your installation process.

In the Foglight Container **Administration** dashboard, a simple template is provided for you to deploy Prometheus service onto your cluster. For more information, see the Deploy/Migrate section in Metrics Collector on page 47.

Docker Swarm Agent

Each Docker Swarm Agent monitors the assets in one docker host. Docker Remote API needs to be enabled for the Docker Swarm Agent collecting data from the docker host. If TLS is enabled to secure the Docker Remote API, credential for Docker Swarm Agent needs to be prepared. Complete the following prerequisites before create agent.

- Preparing Docker Swarm Agent credentials
- Enabling Docker Remote API for monitored docker host
- Uploading Docker Swarm Agent credentials

Preparing Docker Swarm Agent credentials

If TLS enabled to secure Docker Remote API, then complete the following guide to get the credentials for Docker Swarm Agent for the docker host. Otherwise, continue with Enabling Docker Remote API for monitored docker host on page 17

Refer to the official guide to generate the keys. Be aware that, during generating the keys, the Foglight Agent Manager host address should be in the allow access list.

Docker Swarm Agent needs following credentials, you can get them when you finish the official guide.

- CA Public Key (ca.pem in official guide)
- Client Public Key (cert.pem in official guide)
- Client Private Key (key.pem in official guide)

Enabling Docker Remote API for monitored docker host

Change `ExecStart` in docker service startup script as below.

Non-TLS secured

```
ExecStart=/usr/bin/dockerd -H tcp://0.0.0.0:2375 -H
unix:///var/run/docker.sock
```

NOTE: Access should be allowed to the TCP port 2375

TLS secured

If TLS enabled, complete Preparing Docker Swarm Agent credentials on page 17 first, then you will get the ca.pem, server-cert.pem and server-key.pem mentioned in the official guide.

```
ExecStart=/usr/bin/dockerd --tlsverify --tlscacert=ca.pem --tlscert=server-
cert.pem --tlskey=server-key.pem -H tcp://0.0.0.0:2375 -H
unix:///var/run/docker.sock
```

NOTE: Access should be allowed to the TCP port 2375

Then restart docker service.

Uploading Docker Swarm Agent credentials

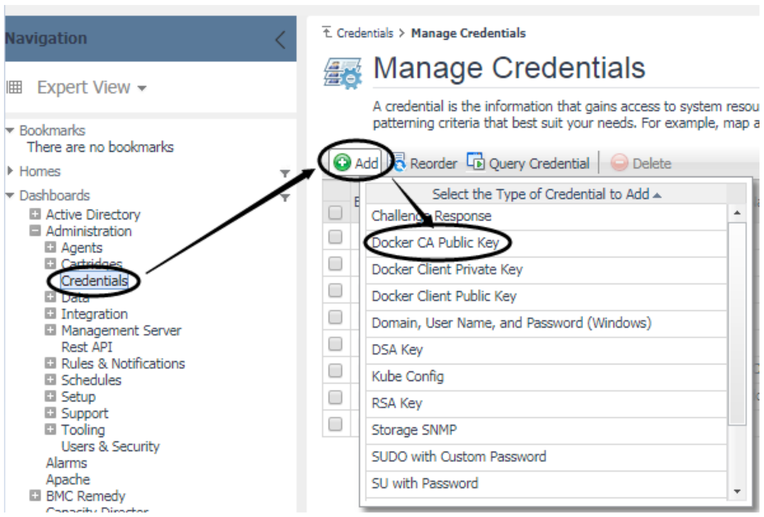
If TLS is enabled to secure Docker Remote API, go through this section to upload the credential for Docker Swarm Agent. Otherwise, skip this section.

When complete Preparing Docker Swarm Agent credentials on page 17, following credentials should be generated.

- CA Public Key
- Client Public Key
- Client Private Key

These are the credentials for Docker Swarm Agent, complete the following steps to upload the credentials.

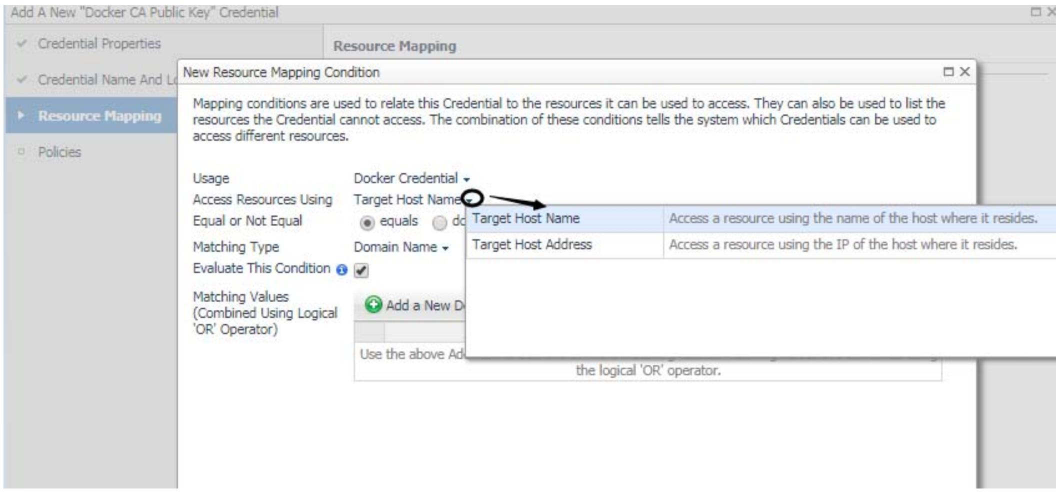
On the **Administration > Credentials > Manage Credentials** dashboard, click **Add**, and then select Docker CA Public Key or Docker Client Public Key or Docker Client Private Key to upload related credentials. Take Docker CA Public Key as an example.



A credential is the information that gains access to system resou
patterning criteria that best suit your needs. For example, map a

In the **Add a New "Docker CA Public Key" Credential** dialog box, specify the following values:

- Credential Properties: Click **Load from file** to import Docker CA Public Key, and then click **Next**.
- Credential Name And Lockbox: Specify a unique name for this credential, and then click **Next**.
- Resource Mapping: Click **Add**. In the **New Resource Mapping Condition** dialog box, choose Target Host Name or Target Host Address for the monitored docker host.

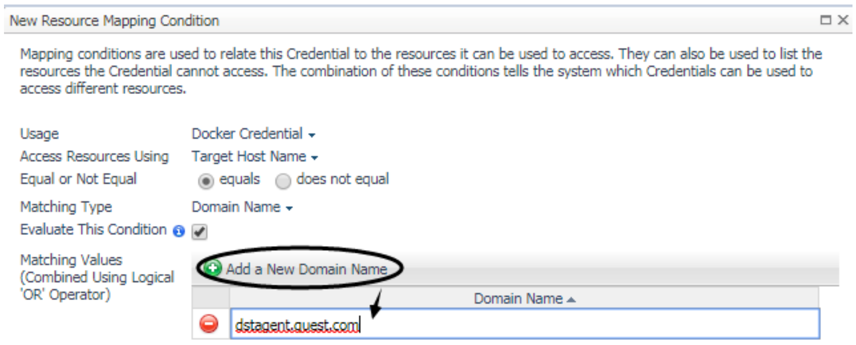


Mapping conditions are used to relate this Credential to the resources it can be used to access. They can also be used to list the resources the Credential cannot access. The combination of these conditions tells the system which Credentials can be used to access different resources.

Usage	Docker Credential
Access Resources Using	Target Host Name
Equal or Not Equal	● equals ○ do
Matching Type	Domain Name
Evaluate This Condition	✓
Matching Values (Combined Using Logical 'OR' Operator)	

Target Host Name	Access a resource using the name of the host where it resides.
Target Host Address	Access a resource using the IP of the host where it resides.

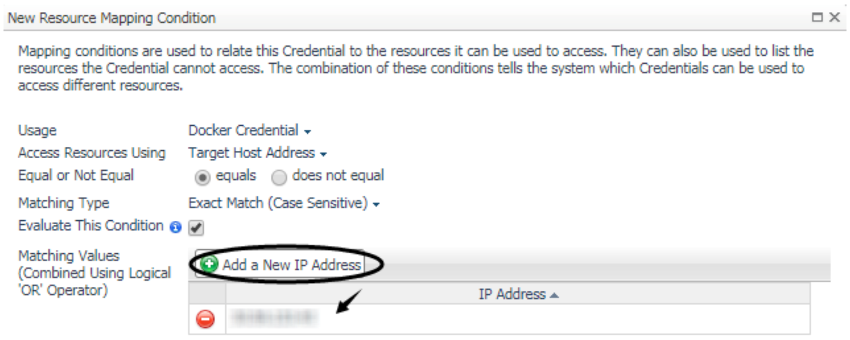
If choose **Target Host Name**, then enter the host name of the monitored docker host.



Mapping conditions are used to relate this Credential to the resources it can be used to access. They can also be used to list the resources the Credential cannot access. The combination of these conditions tells the system which Credentials can be used to access different resources.

Usage	Docker Credential
Access Resources Using	Target Host Name
Equal or Not Equal	● equals ○ does not equal
Matching Type	Domain Name
Evaluate This Condition	✓
Matching Values (Combined Using Logical 'OR' Operator)	dstagent.quest.com

If choose Target Host Address, then enter the IP address of the monitored docker host.



New Resource Mapping Condition

Mapping conditions are used to relate this Credential to the resources it can be used to access. They can also be used to list the resources the Credential cannot access. The combination of these conditions tells the system which Credentials can be used to access different resources.

Usage	Docker Credential
Access Resources Using	Target Host Address
Equal or Not Equal	[✓ equals] does not equal
Matching Type	Exact Match (Case Sensitive)
Evaluate This Condition	✓
Matching Values (Combined Using Logical 'OR' Operator)	Add a New IP Address

IP Address
[redacted]

Then click **Add** to finish editing **New Resource Mapping Condition** and back to **Resource Mapping**. Then click **Finish**.

Then **Docker CA Public Key** has been uploaded and mapped to the docker host. To monitor this docker host, **Docker Client Public Key** and **Docker Client Private Key** also need to be uploaded following the above steps.

Creating and Activating Agent

Foglight for Container Management supports Kubernetes Agent and Docker Swarm Agent.

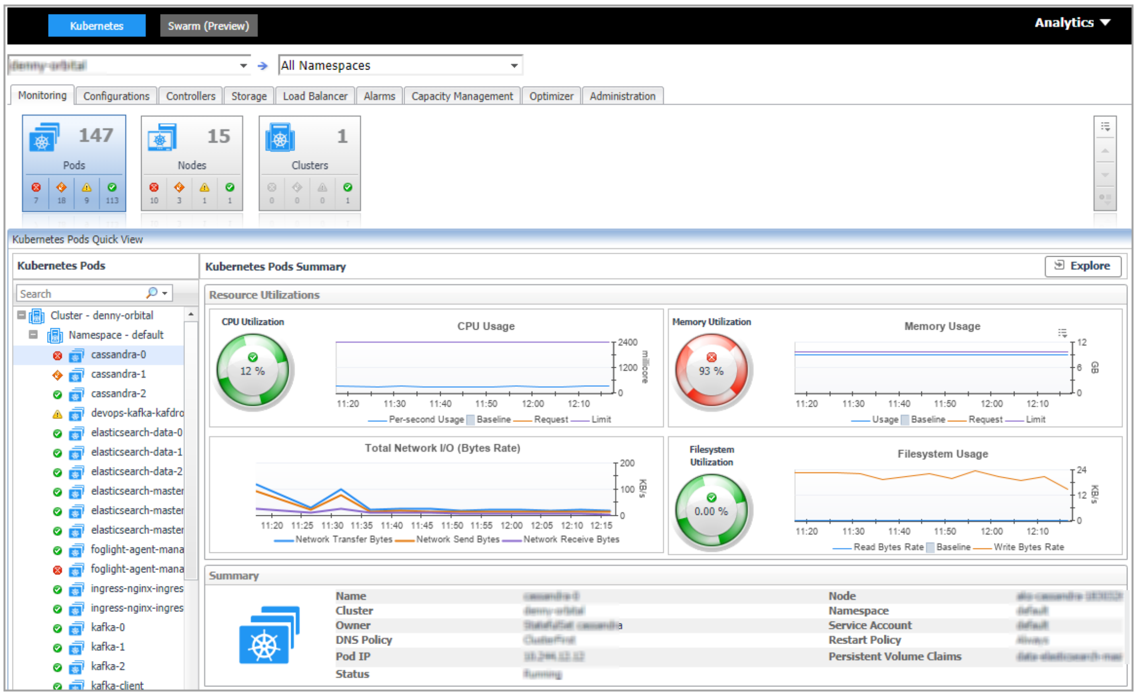
- Creating and Activating a Kubernetes Agent
- Creating and Activating a Docker Swarm Agent

Creating and Activating a Kubernetes Agent

To create a Kubernetes agent on a monitored host:

1 Log in to the Foglight browser interface and make sure the left Navigation panel is open.

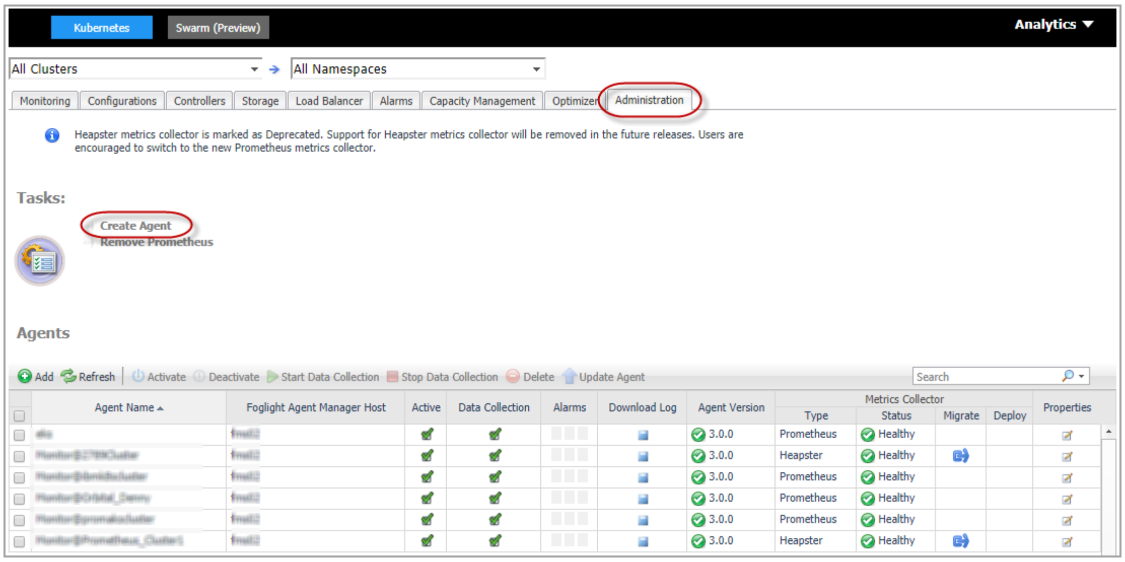
2 On the navigation panel, from **Standard View** click **Container Environment** or from **Expert View** click **Dashboards > Container**. Then the Container dashboard will display as below.



Cluster - denny-orbital
Namespace - default
cassandra-0
cassandra-1
cassandra-2
devops-kafka-kafdro
elasticsearch-data-0
elasticsearch-data-1
elasticsearch-data-2
elasticsearch-master
elasticsearch-master
elasticsearch-master
foglight-agent-mana
foglight-agent-mana
ingress-nginx-ingres
ingress-nginx-ingres
kafka-0
kafka-1
kafka-2
kafka-client

Name	[redacted]	Node	[redacted]
Cluster	[redacted]	Namespace	[redacted]
Owner	[redacted]	Service Account	[redacted]
DNS Policy	[redacted]	Restart Policy	[redacted]
Pod IP	[redacted]	Persistent Volume Claims	[redacted]
Status	[redacted]		

3 In the Container dashboard, click **Administration** tab, and then click **Create Agent**. The **Create Docker Agent** wizard opens.



Heapster metrics collector is marked as Deprecated. Support for Heapster metrics collector will be removed in the future releases. Users are encouraged to switch to the new Prometheus metrics collector.

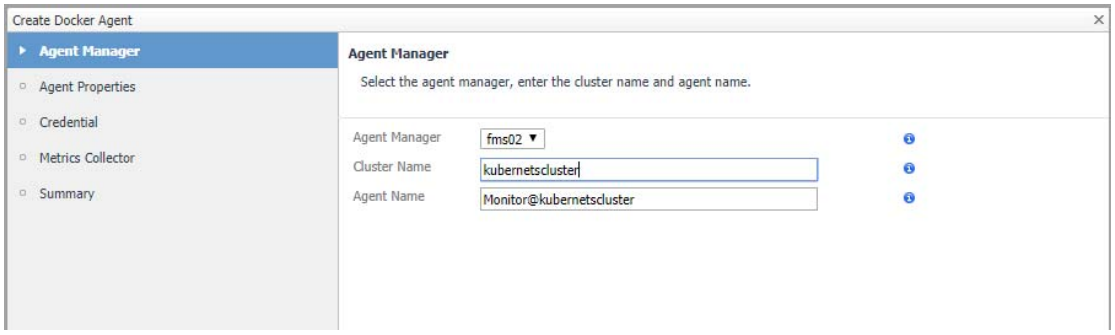
Tasks:

Create Agent
Remove Prometheus

Agents

	Agent Name	Foglight Agent Manager Host	Active	Data Collection	Alarms	Download Log	Agent Version	Metrics Collector Type	Metrics Collector Status	Migrate	Deploy	Properties
☐	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
☐	[redacted]	[redacted]	✓	✓			3.0.0	Heapster	Healthy			
☐	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
☐	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
☐	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
☐	[redacted]	[redacted]	✓	✓			3.0.0	Heapster	Healthy			

4 *Agent Manager*: specify the following values, and then click **Next**.



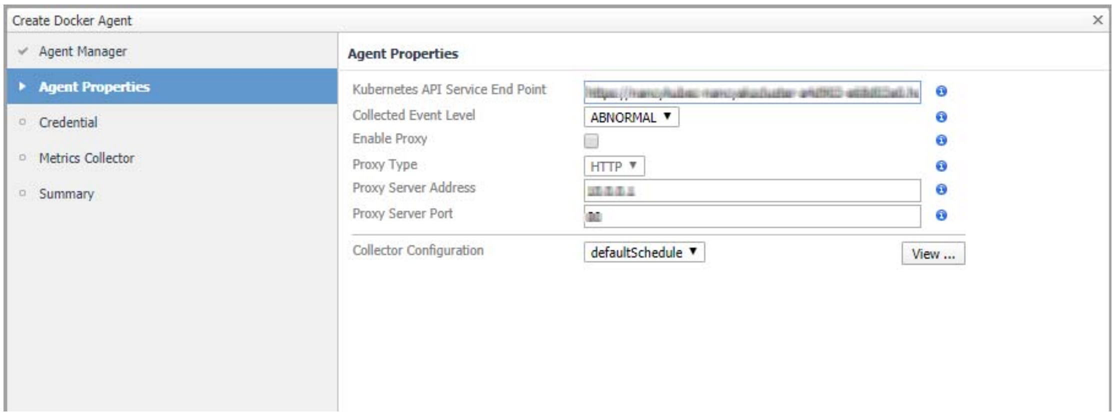
Agent Manager

Select the agent manager, enter the cluster name and agent name.

Agent Manager	fms02
Cluster Name	kubernetscluster
Agent Name	Monitor@kubernetscluster

- *Agent Manager*: The Kubernetes Agent will be create in the selected Agent Manager.
- *Cluster Name*: Customized cluster name which identifies a Kubernetes cluster.
- *Agent Name:* Auto-generated agent name. You can change the name according to your requirement. It should be a unique name.

5 *Agent Properties*



Agent Properties

Kubernetes API Service End Point	[redacted]
Collected Event Level	ABNORMAL
Enable Proxy	☐
Proxy Type	HTTP
Proxy Server Address	[redacted]
Proxy Server Port	[redacted]
Collector Configuration	defaultSchedule

- *Kubernetes API Service End Point*: The format is: http(s)://<url:port>. If you have a Kubeconfig file, retrieve this endpoint from the file (path: clusters -> cluster -> server). If there are multiple clusters, find the current context related cluster server URL.

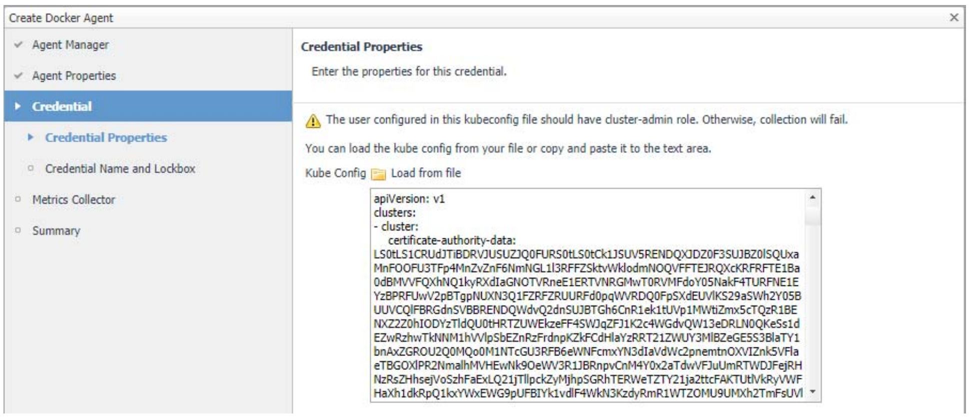
- *Collected Event Level:* Set the collected event level, including *ABNORMAL* and *ALL*. *ALL* will collect both abnormal and normal events while *Abnormal* only collects abnormal events.
- *Enable Proxy*: To enable the proxy, select the checkbox. Enter the *Proxy Server Address* and *Proxy Server Port* information.
- *Collector Configuration*: Used to configure collection interval for inventory and metrics. You can change the collector intervals of defaultSchedule, however, this will affect all the Kubernetes Agents. Or you can create a new scheduler, configure the collector intervals, and then assign this scheduler to this agent.

  Collector intervals to configure:

  - inventory interval (inventory data collection interval).
  - metrics interval (performance metrics data collection interval).

6 *Credential*

- *If no credentials were found for the provided resource, configure credentials:*
  - Credential Properties: Click **Load from file** to upload the credential and click **Next**.



Credential Properties

Enter the properties for this credential.

The user configured in this kubeconfig file should have cluster-admin role. Otherwise, collection will fail.

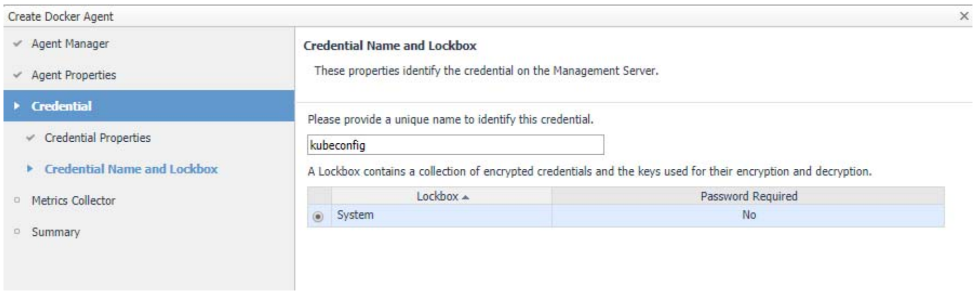
You can load the kube config from your file or copy and paste it to the text area.

Kube Config Load from file

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
```

LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUV5RENDQXJDZ0F3SUJBZ0lSQUxa
MnFOOFU3TFp4MnZvZnF6NmNGL1l3RFFZSktvWklodmNOQVFFTEJRQXcKRFRFTE1Ba
0dBMVVFQXhNQ1kyRXdIaGNOTVRneE1ERTVNRGMwT0RVMFdoY05NakF4TURFNE1E
YzBPRFUwV2pBTgpNUXN3Q1FZRFZRUURFd0pqWVRDQ0FpSXdEUVlKS29aSWh2Y05B
UUVCQlFBRGdnSVBBRENDQWdvQ2dnSUJBTGh6CnR1ek1tUVp1MWtiZmx5cTQzR1BE
NXZ2Z0hIODYzTldQU0tHRTZUWEkzeFF4SWJqZFJ1K2c4WGdvQW13eDRLN0QKeSs1d
EZwRzhwTkNNM1hVVlpSbEZnRzFrdnpKZkFCdHlaYzRRT21ZWUY3MlBZeGE5S3BlaTY1
bnAxZGROU2Q0MQo0M1NTcGU3RFB6eWNFcmxYN3dIaVdWc2pnemtnOXVIZnk5VFla
eTBGOXlPR2NmalhMVHEwNk9OeWV3R1JBRnpvCnM4Y0x2aTdwVFJuUmRTWDJFejRH
NzRsZHhsejVoSzhFaExQ21jTllpckZyMjhpSGRhTERWeTZTY21ja2ttcFAKTUtlVkRyVWF
HaXh1dkRpQ1kxYWxEWG9pUFBIYk1vdlF4WkN3KzdyRmR1WTZOMU9UMXh2TmFsUVl

  - Credential Name and Lockbox: give a name for the credential, choose a lockbox, then click **Next**.



Credential Name and Lockbox

These properties identify the credential on the Management Server.

Please provide a unique name to identify this credential.

kubeconfig

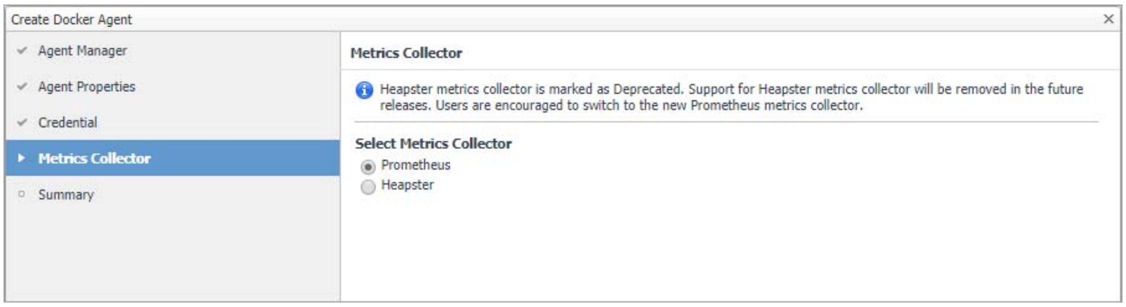
A Lockbox contains a collection of encrypted credentials and the keys used for their encryption and decryption.

	Lockbox	Password Required
●	System	No

- If an existing credential is detected, go to *Metrics Collector* directly.

7 *Metrics Collector*

Currently both Heapster and Prometheus are supported for metrics collector. However, users are encouraged to use the Prometheus metrics collector.


Create Docker Agent

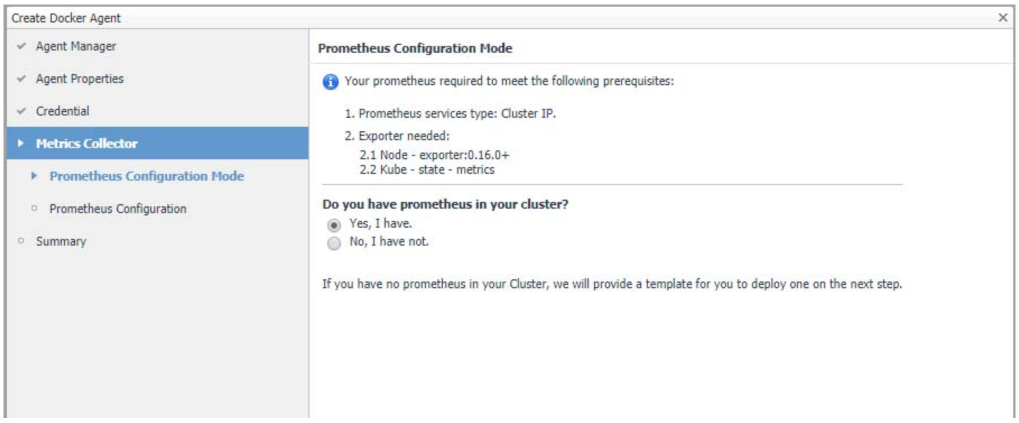
Metrics Collector

Heapster metrics collector is marked as Deprecated. Support for Heapster metrics collector will be removed in the future releases. Users are encouraged to switch to the new Prometheus metrics collector.

Select Metrics Collector

[✓ Prometheus] Heapster

- Prometheus metrics collector


Create Docker Agent

Prometheus Configuration Mode

Your prometheus required to meet the following prerequisites:

1. Prometheus services type: Cluster IP.
2. Exporter needed:
   2.1 Node - exporter:0.16.0+
   2.2 Kube - state - metrics

Do you have prometheus in your cluster?

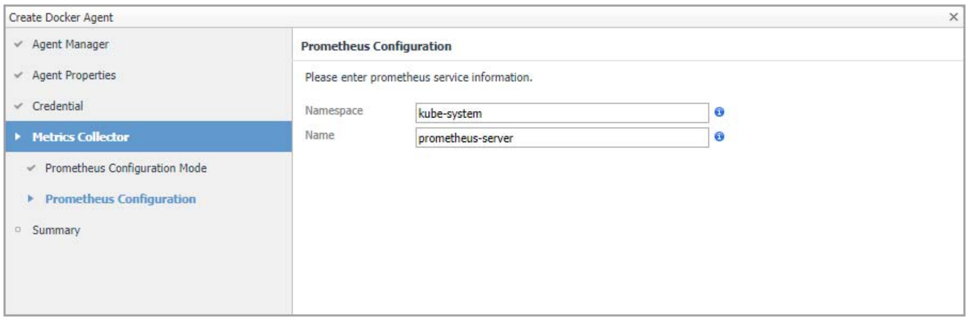
[✓ Yes, I have.] No, I have not.

If you have no prometheus in your Cluster, we will provide a template for you to deploy one on the next step.

  - Have existing Prometheus in your cluster.

    Enter the existing Prometheus service namespace and name to configure the metrics collector.

NOTE: Ensure that you have Prometheus in your cluster before this step. We will check the service existence and health status after you click Next.


Create Docker Agent

Prometheus Configuration

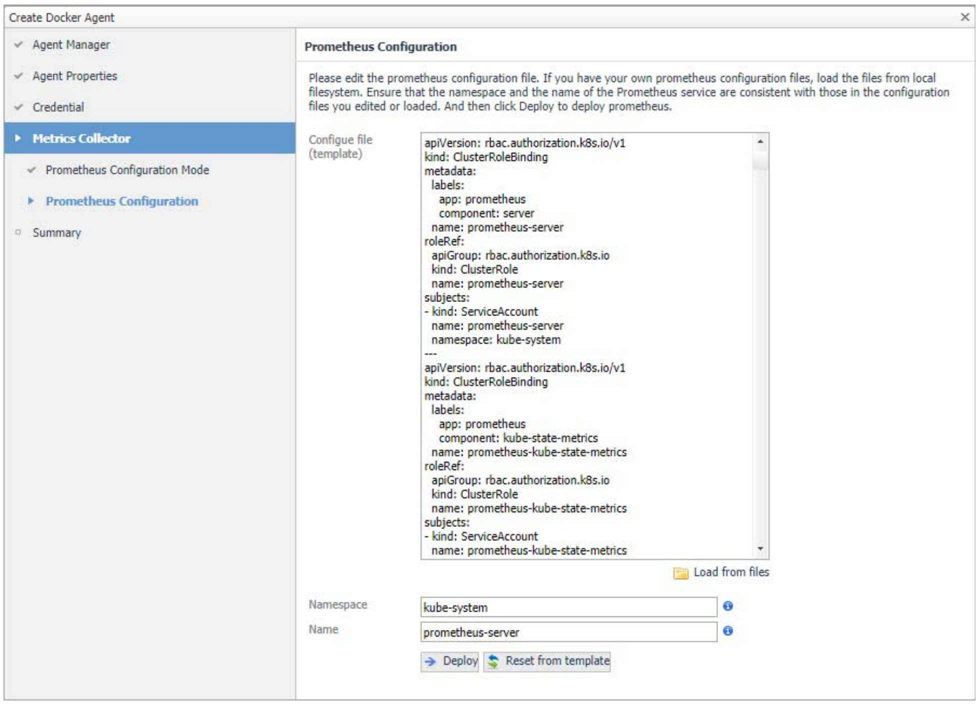
Please enter prometheus service information.

Namespace	kube-system
Name	prometheus-server

  - Do not have existing Prometheus in your cluster.

    A *Prometheus Configuration* wizard will appear. To configure the metrics collector, either use the default Prometheus template or upload your own Prometheus deployment .yml files through *Load from files*.

NOTE: Whenever you change the content in Configure file (template), ensure that the *Namespace* and *Name* fields are consistent with the Prometheus service configurations in your Configure file (template).
If you are using the default template, there is need to change anything.
For a full version of the default template, see
https://github.com/Foglight/Container/tree/master/prometheus



Please edit the prometheus configuration file. If you have your own prometheus configuration files, load the files from local filesystem. Ensure that the namespace and the name of the Prometheus service are consistent with those in the configuration files you edited or loaded. And then click Deploy to deploy prometheus.

Configue file (template)

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    app: prometheus
    component: server
  name: prometheus-server
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus-server
subjects:
- kind: ServiceAccount
  name: prometheus-server
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    app: prometheus
    component: kube-state-metrics
  name: prometheus-kube-state-metrics
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus-kube-state-metrics
subjects:
- kind: ServiceAccount
  name: prometheus-kube-state-metrics
```

Namespace: kube-system
Name: prometheus-server

Reset from template: Helps you to reset the Configure file (template) content to the default template.

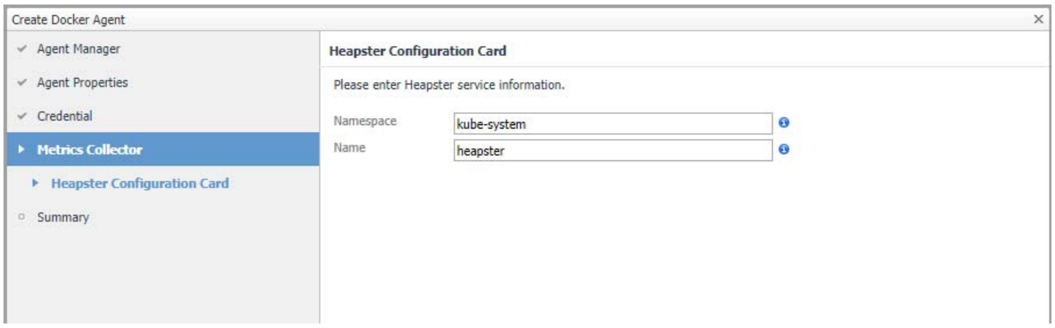
Deploy: Creates Prometheus components to your cluster with the Configure file.
After clicking *Deploy*, a progress message will be displayed.

- If deployed successfully, a succeeded message will be displayed. Close the *Validation Result* page and then click **Next** to finish your agent creation process.
- If failed to deploy the Prometheus configuration, a *Validation Result* page will be displayed with possible solutions.

- Heapster metric collector

  Enter the existing Heapster service namespace and name to configure the metrics collector.

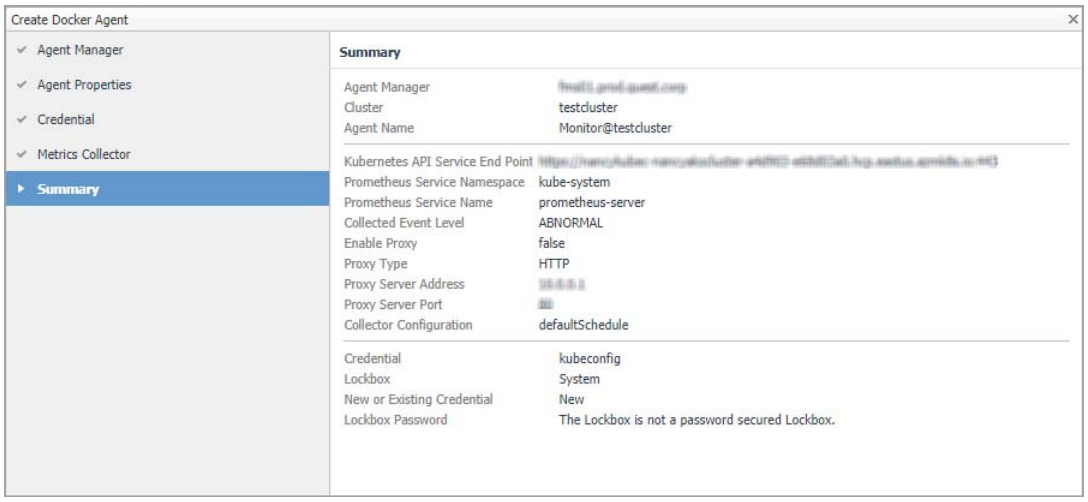
  IMPORTANT: Deploy the Heapster service to your cluster manually before creating an agent with Heapster as metric collector. Otherwise, connection test to your Heapster service will fail, and you cannot proceed to the next step.



Please enter Heapster service information.

Namespace: kube-system
Name: heapster

8 *Summary*: click **Finish**.


Create Docker Agent

- ✓ Agent Manager
- ✓ Agent Properties
- ✓ Credential
- ✓ Metrics Collector
- ▸ Summary

Summary

Agent Manager	[redacted]
Cluster	testcluster
Agent Name	Monitor@testcluster
Kubernetes API Service End Point	[redacted]
Prometheus Service Namespace	kube-system
Prometheus Service Name	prometheus-server
Collected Event Level	ABNORMAL
Enable Proxy	false
Proxy Type	HTTP
Proxy Server Address	[redacted]
Proxy Server Port	[redacted]
Collector Configuration	defaultSchedule
Credential	kubeconfig
Lockbox	System
New or Existing Credential	New
Lockbox Password	The Lockbox is not a password secured Lockbox.

9 Then, the agent will be created and activated automatically.

Creating and Activating a Docker Swarm Agent

Each Docker Swarm Agent monitored one docker host. If the docker host belongs to a Docker Swarm cluster, it will be considered as a manager/worker node. Otherwise, it will be considered to be a standalone docker host.

NOTE: For a Docker Swarm cluster, you should create one Docker Swarm Agent for one host in the cluster, and if you want to monitor the whole cluster environment, you need to create all the Docker Swarm Agents for all the hosts in the cluster.

To create a Docker Swarm agent on a monitored host:

1 Login in to the Foglight browser interface and make sure the left navigation panel is open.

2 On the navigation panel, under **Dashboards**, click **Administration > Agents > Agent Status**.

The **Agent Status** dashboard opens.

3 In the **Agent Status** dashboard, click **Create Agent**.

The **Create Agent** wizard opens.

4 *Host Selector*: Select the monitored host that you want to monitor with the Docker Swarm agent instance that you are about to create, and then click **Next**.

NOTE: In order to select the host, the Foglight Agent Manager must be installed and running on the monitored host.

5 *Agent Type and Instance Name*: Specify the following values, and then click **Next**.

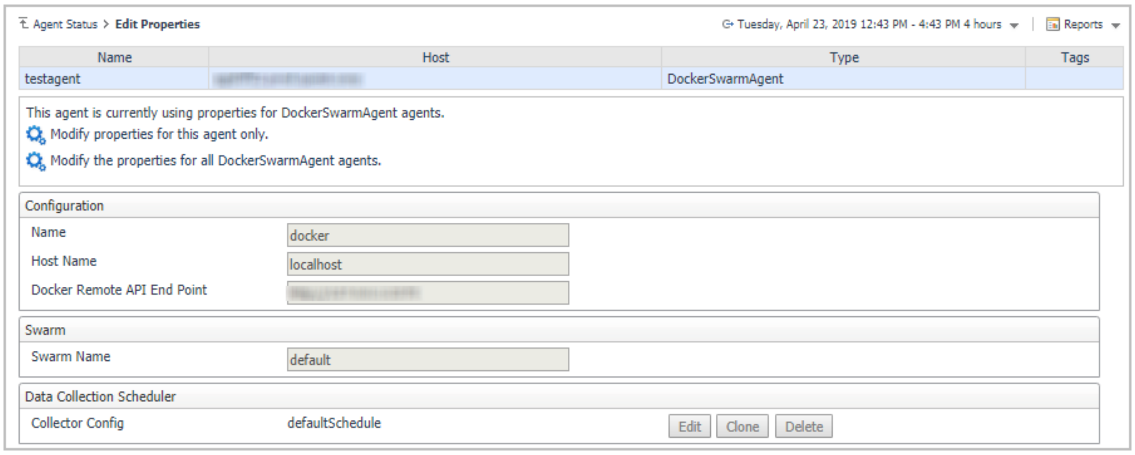
- *Agent Type*: Select DockerSwarmAgent from the agent type list.
- *Agent Name*: Specify the name of the agent instance that you are about to create using either of the following approaches:
  - *Generic Name*: This option is selected by default. A generic name is a combination of the host name and the agent type and uses the following syntax: `agent_type@host_name`.
  - *Specify Name*: Type that name in the *Name* field. For example, `MyAgent`.

6 On the **Summary** page, review the choices you have made, and then click **Finish**.

The *Agents* table refreshes automatically, showing the new Docker Swarm Agent.

7 On the *Agents* table, select the Docker Swarm Agent that you create, click **Edit Properties**, and then click **Modify the private properties for this agent**.

8 In the *Agents* properties view, check if the following values have been configured based upon your environment:



Agent Status > **Edit Properties**

Tuesday, April 23, 2019 12:43 PM - 4:43 PM 4 hours | Reports

Name	Host	Type	Tags
testagent	[redacted]	DockerSwarmAgent	

This agent is currently using properties for DockerSwarmAgent agents.
Modify properties for this agent only.
Modify the properties for all DockerSwarmAgent agents.

Configuration

Name	docker
Host Name	localhost
Docker Remote API End Point	[redacted]

Swarm

Swarm Name	default

Data Collection Scheduler

Collector Config	defaultSchedule	Edit Clone Delete

- *Name*: give a name to the monitored docker host, it should be unique.
- *Host Name*: IP address or host name of the monitored docker host.
- *Docker Remote API End Point*: Docker Remote API endpoint of the monitored docker host. For more information, see Enabling Docker Remote API for monitored docker host on page 17.
- *Swarm Name*: specify the swarm cluster name for display. If the swarm name is kept as “default”, then the cluster name will be displayed as “default (cluster ID)” on the dashboard. If a customized name is input here, then the customized cluster name will be displayed on the dashboard.

NOTE: Ensure that the docker host inside the same cluster has the same configuration for Swarm Name.

9 Return back to the *Agents* table, select the above property changed Docker Swarm Agent, and then click **Activate**.

The new Docker Swarm Agent is created and data will be shown on the **Monitoring** tab after a few minutes.

Configuring data collection interval

The default data collection interval of agents is set to 5 minutes by default. Foglight for Container Management enables you to change this collection interval as needed.

NOTE: Changing the data collection interval will take effect for all Kubernetes agents and Docker Swarm agents.

To configure the data collection interval:

1 On the navigation panel, under **Dashboards**, select **Administration > Agents > Agent Status**.

2 On the *Agent Status* dashboard, select the Kubernetes agent that you use to monitoring the container environment, and then click **Edit Properties**.

3 In the *Edit Properties* dashboard, click **Edit** next to the *Collector Config* field.

4 In the KubernetesAgent or DockerSwarmAgent Collector Config dialog box, change the following values, as needed:

- *Inventory Collector*: Specifies the interval for collecting components.
- *Metrics Collector*: Specifies the interval for collecting metrics.

5 Click **Save**.

2

Using Foglight for Container Management

- Customization

Kubernetes

Monitoring Kubernetes Pods

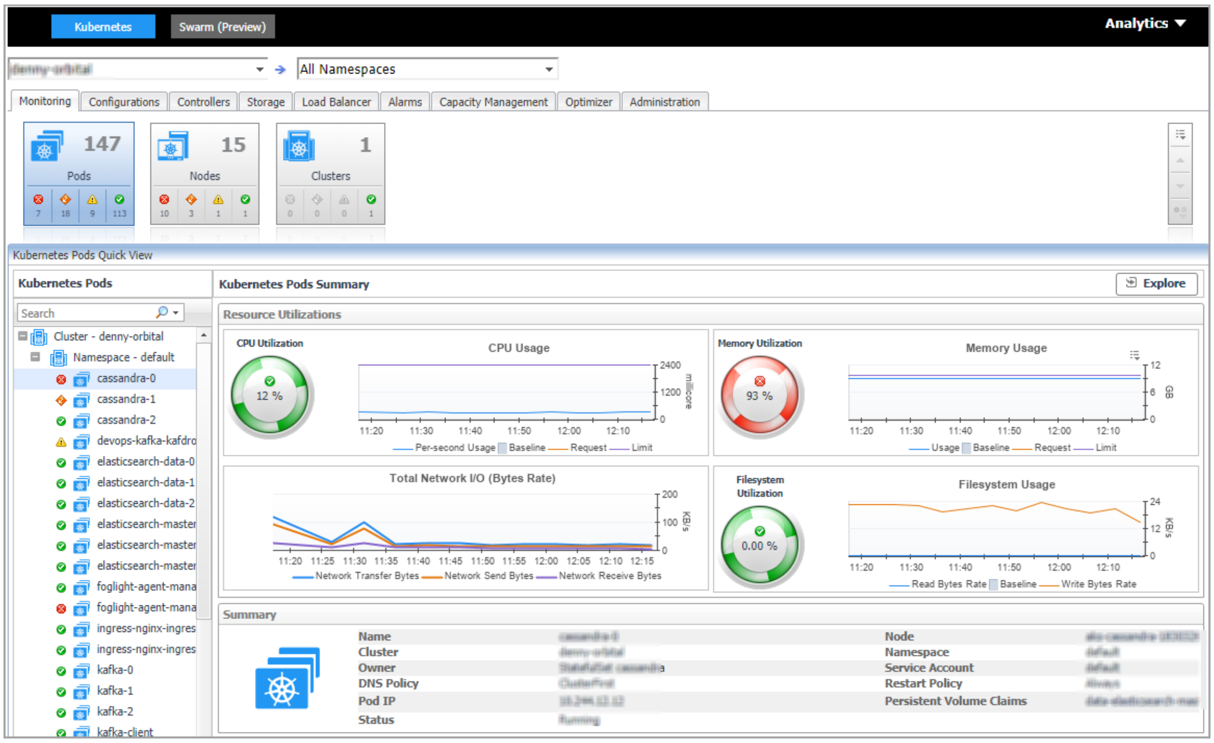
A pod contains one or multiple containers, such as Docker containers, which contains storage/network and the specification about how to run the containers. The *Kubernetes Pods Quick View*, which appears after clicking **Monitoring > Pods**, shows the data collected about the selected clusters and namespaces. This view consists of the following two panes:

- The **Kubernetes Pods** tree view, which appears on the left of *Kubernetes Pods Quick View*, lists the pods existing in the monitored Kubernetes environment.
- The Kubernetes Pods Summary view, which appears on the right after you select an individual pod in the **Kubernetes Pods** tree view.

Kubernetes Pods Summary view

The **Kubernetes Pods Summary** view appears on the right when you select a cluster in the **Kubernetes Pods** tree view.

Figure 2. Kubernetes Pods Summary view



Name	[redacted]	Node	[redacted]
Cluster	[redacted]	Namespace	[redacted]
Owner	[redacted]	Service Account	[redacted]
DNS Policy	[redacted]	Restart Policy	[redacted]
Pod IP	[redacted]	Persistent Volume Claims	[redacted]
Status	[redacted]		

The **Kubernetes Pods Summary** view displays the following data:

- *Resource Utilizations*: The resource utilization for the selected Kubernetes Pod over a selected period of time, which includes the following:
  - *CPU Utilization:* shows the percentage of CPU usage divides CPU limit, if the limit is configured for this Pod.

CPU Usage: shows CPU usage rate, request, and limit, the trend of usage rate, as well as request and limit configuration.

- *Memory Utilization:* shows *the* percentage of Memory usage divides Memory limit, if the limit is configured for this Pod.

  Memory Usage: shows Memory usage, request, and limit, the trend of usage rate, as well as request and limit configuration.
- *Total Network I/O:* shows the sending, receiving, and transferring rate.
- *Filesystem Utilization:* shows the Filesystem utilization.

  Filesystem Usage: shows the read and write rate in byte/s.

- *Summary*: Displays the detailed information about the selected Kubernetes Pod, including *Name, Node, Cluster, Namespace, Owner, Pod IP, Service Account, DNS Policy, Restart Policy,* and *Status*.

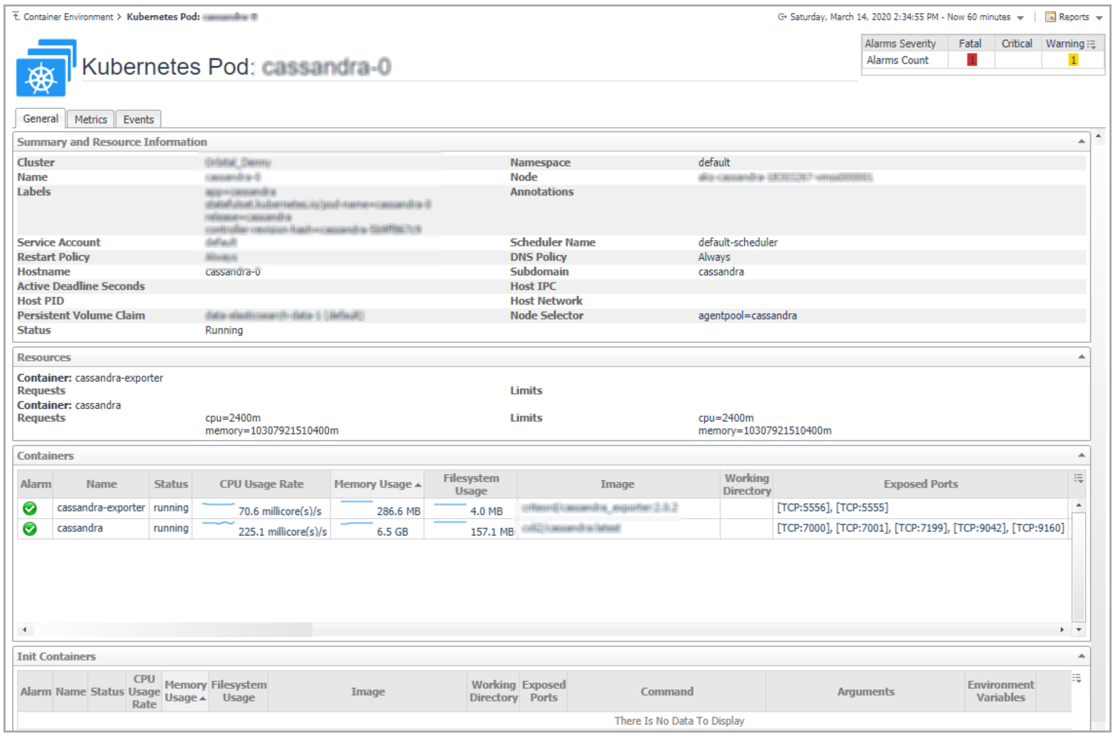
Click **Explore** on the upper right of the **Kubernetes Pods Summary** view to open the Pods Explorer view, which shows more detailed information about this Kubernetes cluster.

Pods Explorer view

The *Pods Explorer* view opens when you click **Explore** in the Kubernetes Pods Summary view, which includes the following tabs:

- *General tab:* The *General* tab displays the overall information of the selected Kubernetes Pod over a selected period of time, including the *Summary and Resource Information* table, the *Containers* table, and the *Init Containers* table. For more information, see Pod metrics *on page* 65.

Figure 3. Kubernetes Pods Explorer view General Tab



Container Environment > **Kubernetes Pod:** [redacted]

Saturday, March 14, 2020 2:34:55 PM - Now 60 minutes | Reports

Alarms Severity	Fatal	Critical	Warning
Alarms Count	1		1

Kubernetes Pod: cassandra-0

General | Metrics | Events

Summary and Resource Information

Cluster	[redacted]	**Namespace**	default
Name	[redacted]	**Node**	[redacted]
Labels	[redacted]	**Annotations**	
Service Account	[redacted]	**Scheduler Name**	default-scheduler
Restart Policy	[redacted]	**DNS Policy**	Always
Hostname	cassandra-0	**Subdomain**	cassandra
Active Deadline Seconds		**Host IPC**	
Host PID		**Host Network**	
Persistent Volume Claim	[redacted]	**Node Selector**	agentpool=cassandra
Status	Running		

Resources

Container: cassandra-exporter

Requests		**Limits**	

Container: cassandra

Requests	cpu=2400m memory=10307921510400m	**Limits**	cpu=2400m memory=10307921510400m

Containers

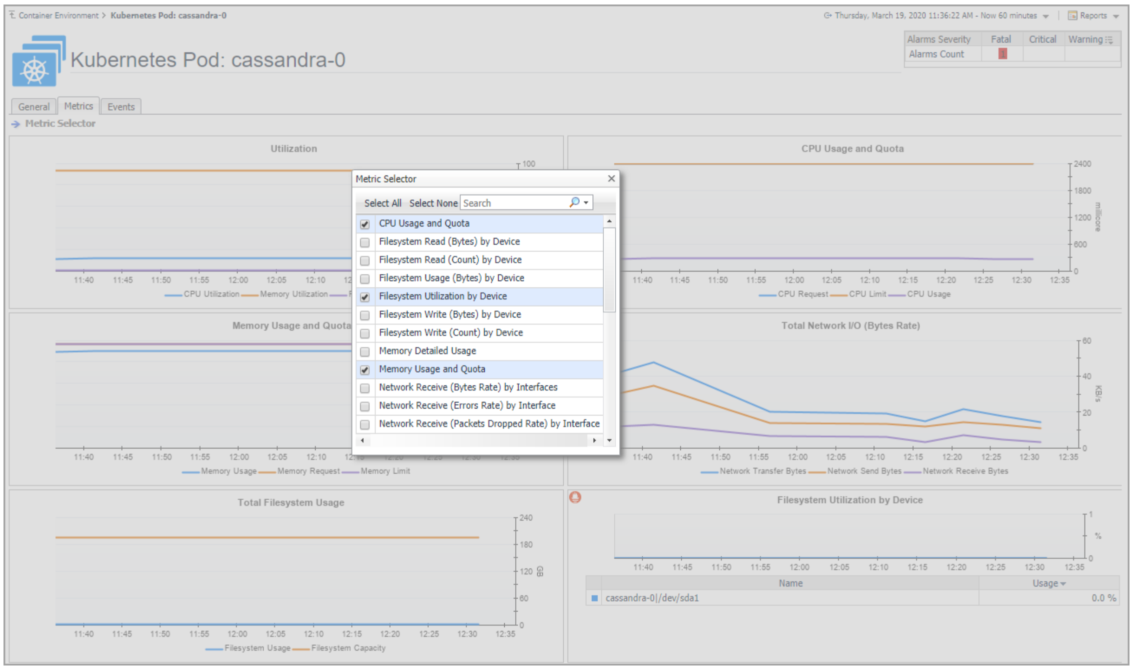
Alarm	Name	Status	CPU Usage Rate	Memory Usage ▲	Filesystem Usage	Image	Working Directory	Exposed Ports
✓	cassandra-exporter	running	70.6 millicore(s)/s	286.6 MB	4.0 MB	[redacted]		[TCP:5556], [TCP:5555]
✓	cassandra	running	225.1 millicore(s)/s	6.5 GB	157.1 MB	[redacted]		[TCP:7000], [TCP:7001], [TCP:7199], [TCP:9042], [TCP:9160]

Init Containers

Alarm	Name	Status	CPU Usage Rate	Memory Usage ▲	Filesystem Usage	Image	Working Directory	Exposed Ports	Command	Arguments	Environment Variables
There Is No Data To Display											

- *Metrics tab:* The *Metrics* tab displays a *Metric Selector* allowing you to choose the metrics to be plotted on this dashboard. Charts of *CPU Usage, Memory Usage,* and *Network I/O* are presented by default.

Figure 4. Kubernetes Pods Explorer view Metrics Tab



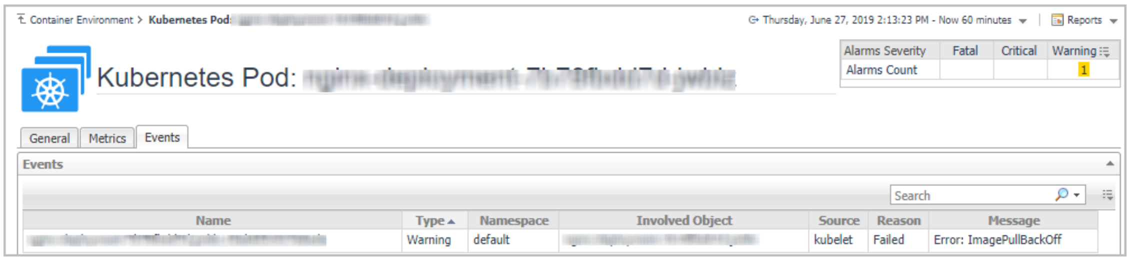
Metric Selector

- ☑ CPU Usage and Quota
- ☐ Filesystem Read (Bytes) by Device
- ☐ Filesystem Read (Count) by Device
- ☐ Filesystem Usage (Bytes) by Device
- ☑ Filesystem Utilization by Device
- ☐ Filesystem Write (Bytes) by Device
- ☐ Filesystem Write (Count) by Device
- ☐ Memory Detailed Usage
- ☑ Memory Usage and Quota
- ☐ Network Receive (Bytes Rate) by Interfaces
- ☐ Network Receive (Errors Rate) by Interface
- ☐ Network Receive (Packets Dropped Rate) by Interface

Name	Usage
cassandra-0|/dev/sda1	0.0 %

- *Event* tab: The *Event* tab lists all the events occur on the pods.
  - Name: name of the event.
  - Type: type of the event, Warning or Normal.
  - Namespace: namespace of where this event happens.
  - Involved Object: name of the Kubernetes component on which this event occurs.
  - Source: where this event has been triggered from.
  - Reason: reason of this event.
  - Message: detailed message of this event.

Figure 5. Kubernetes Pods Explorer view Events tab


Container Environment > Kubernetes Pod: [redacted]
Thursday, June 27, 2019 2:13:23 PM - Now 60 minutes
Reports
Alarms Severity
Fatal
Critical
Warning
Alarms Count
1
Kubernetes Pod: [redacted]
General
Metrics
Events
Events
Search

Name	Type	Namespace	Involved Object	Source	Reason	Message
[redacted]	Warning	default	[redacted]	kubelet	Failed	Error: ImagePullBackOff

Monitoring Kubernetes Nodes

A node, previously known as a minion, is a worker machine in Kubernetes. A node may be a VM or physical machine, depending on the cluster. Each node has the services necessary to run pods and is managed by the master components. The *Kubernetes Nodes Quick View*, which appears after clicking **Monitoring > Nodes**, shows the data collected about the selected clusters and namespaces. This view consists of the following two panes:

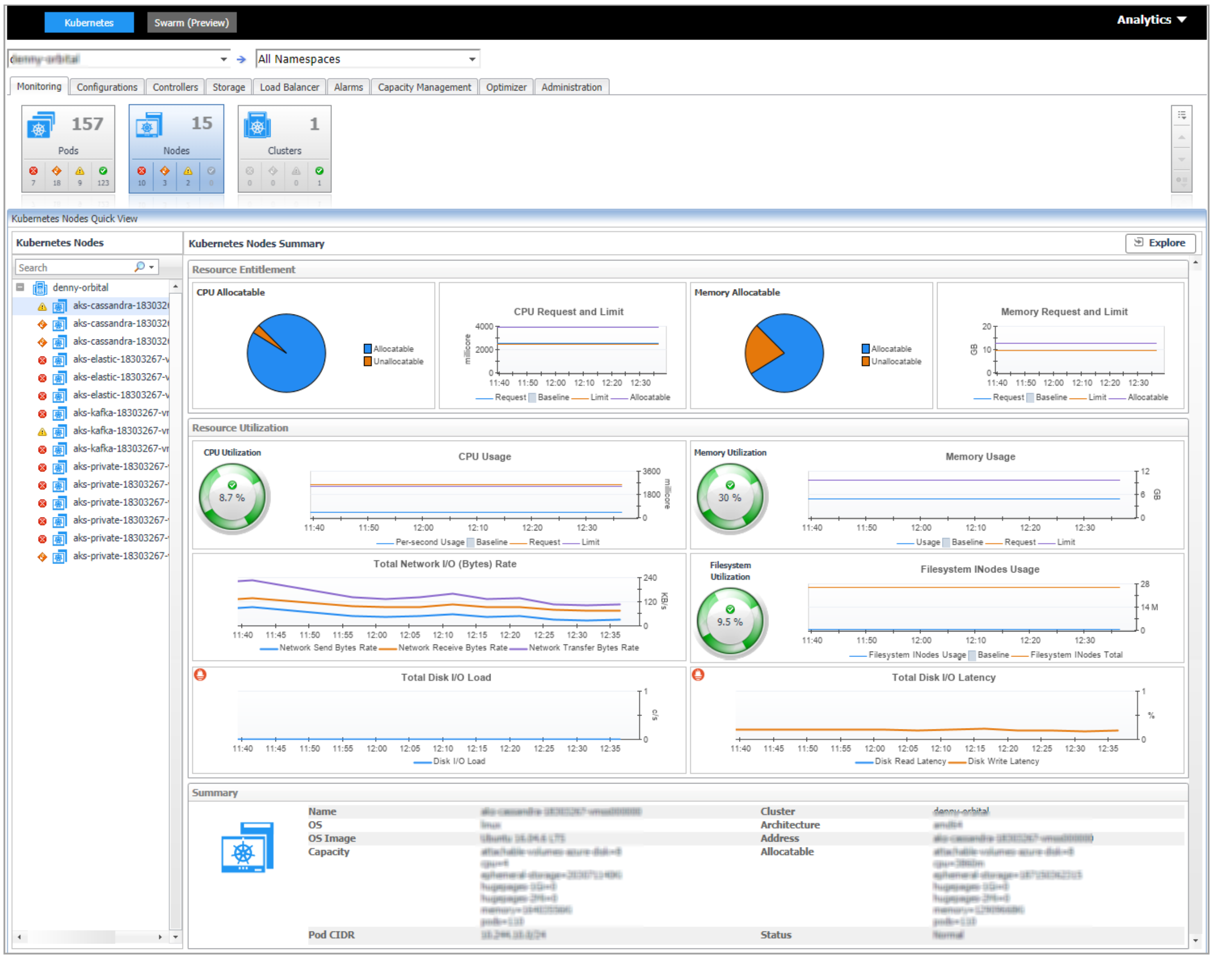
- The **Kubernetes Nodes** tree view, which appears on the left of *Kubernetes Nodes Quick View*, lists the nodes existing in the monitored Kubernetes environment.

- The Kubernetes Nodes Summary view, which appears on the right after you select an individual node in the **Kubernetes Nodes** tree view.

Kubernetes Nodes Summary view

The **Kubernetes Nodes Summary** view appears on the right when you select a node in the **Kubernetes Nodes** tree view.

Figure 6. Kubernetes Nodes Summary view



The **Kubernetes Nodes Summary** view displays the following data:

- *Resource Entitlement*: The resource allocation for the selected Kubernetes node over a selected period of time, which includes the following:
  - *CPU Allocatable*: Shows the current allocatable CPU resources of this node.
  - *Memory Allocatable*: Shows the current allocatable Memory resources of this node.
  - *CPU Request and Limit*: Shows the trend of CPU request, limit, and capacity of this node.
  - *Memory Request and Limit*: Shows the trend of Memory request, limit, and capacity of this node.
- *Resource Utilizations*: The resource utilization for the selected Kubernetes node over a selected period of time, which includes the following:
  - *CPU Utilization*: shows the percentage of CPU usage divide CPU capacity.

    CPU Usage: shows the usage, request, and limit of CPU.
  - *Memory Utilization*: shows the percentage of Memory usage divide Memory capacity.

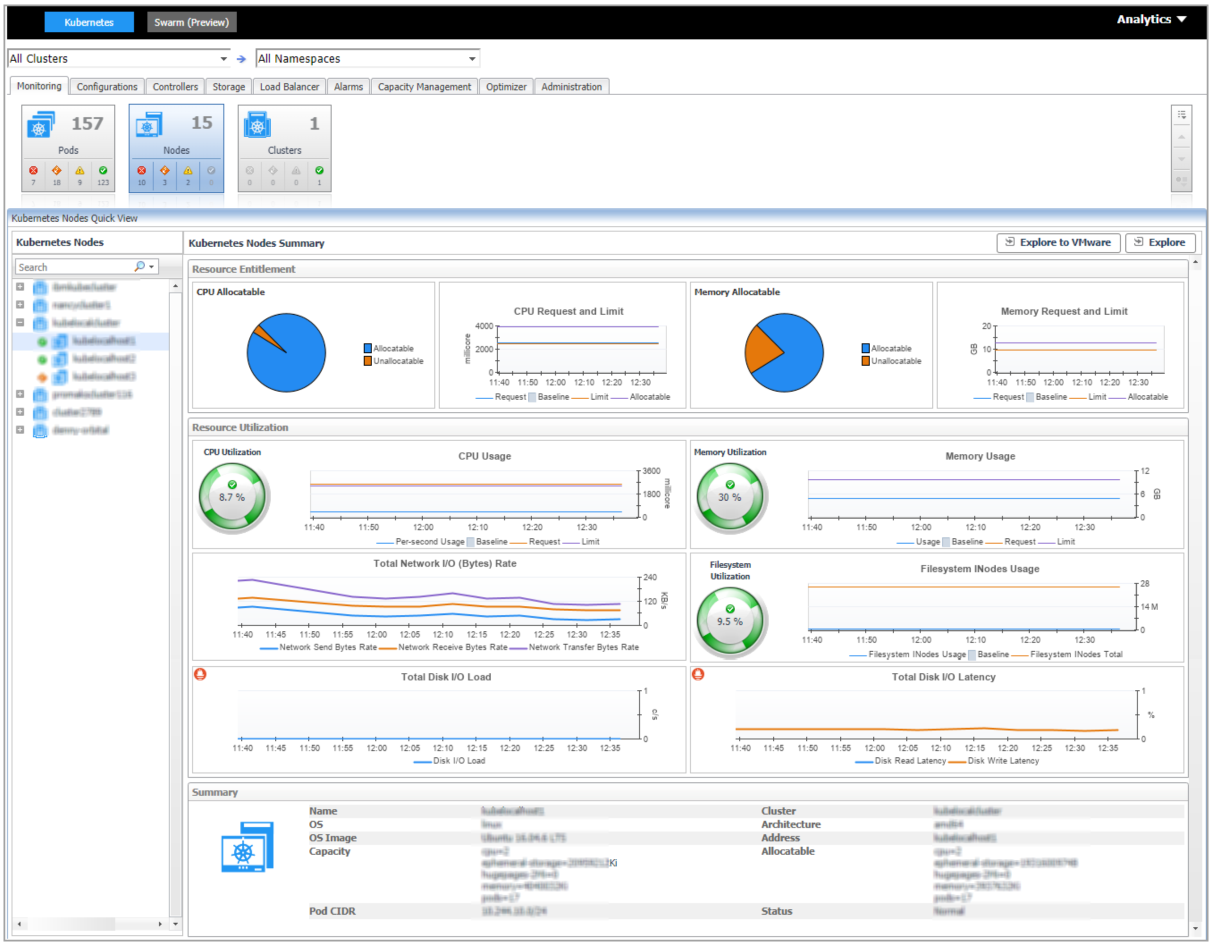
    Memory Usage: shows the usage, request and limit of Memory.

  - *Total Network I/O*: shows the sending, receiving, and transferring rate in byte/s, aggregated from all interfaces.
  - *Filesystem Utilization*: shows the filesystem utilization.

    Filesystem Inodes Usage: shows the inodes usage and total inodes.
  - *Total Disk I/O Load*: shows the number of IOs in progress per second, aggregated from all disk devices.
  - *Total Disk I/O Latency*: shows the read and write latency in percentage.
  - : The icon indicates this metrics is collected by Prometheus metrics collector.
- *Summary*: Displays the detailed information about the selected Kubernetes node, including *Name, Pod CIDR, OS, Architecture, OS Image, Address, Capacity, Allocatable,* and *Status*.

Click **Explore** on the upper right of the **Kubernetes Nodes Summary** view to open the Nodes Explorer view, which shows more detailed information about this Kubernetes node.

Figure 7. Kubernetes Nodes Summary view for VMware



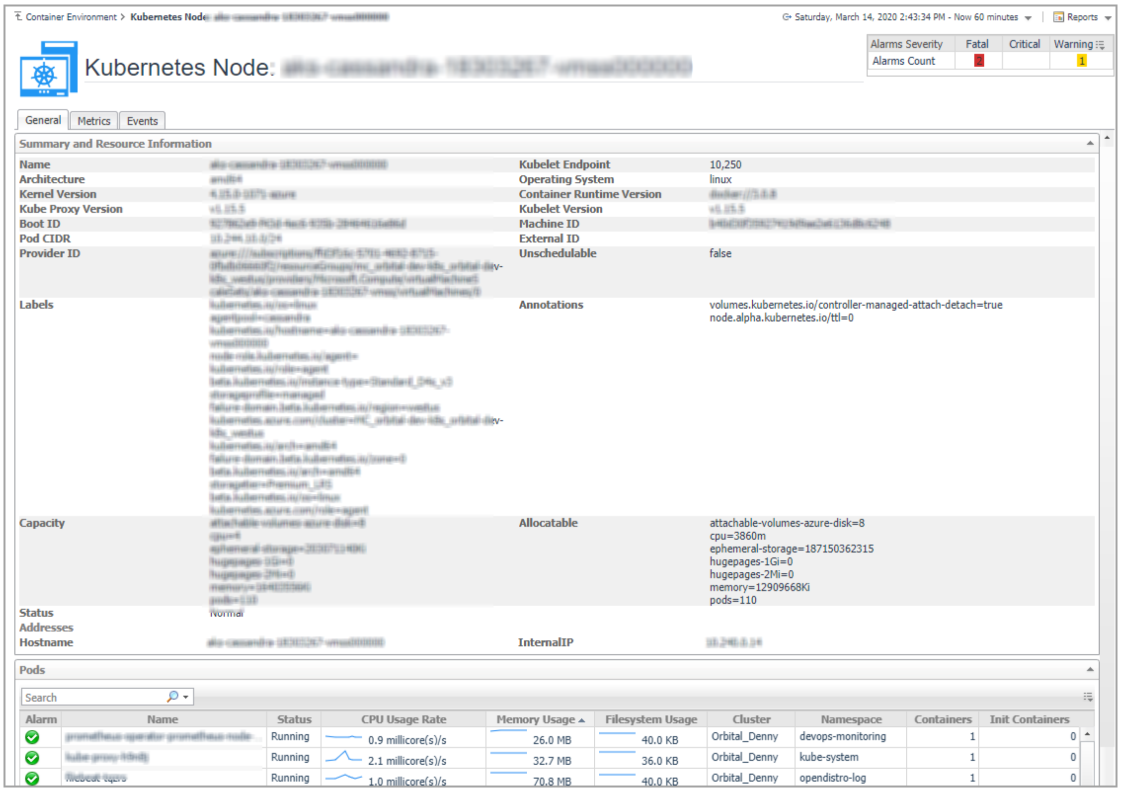
- *Explore to VMware VM*: Click the button to open the *VMware Explorer* view, which is the same view from VMware cartridge. The *Explore to xx* button varies from the cartridge that is monitoring the machines. Currently, the supported cartridges include: VMware, Infrastructure, AWS, and Azure.

Nodes Explorer view

The *Nodes Explorer* view opens when you click **Explore** in the Kubernetes Nodes Summary view, which includes the following tabs:

- *General tab:* The *General* tab displays the overall information of the selected Kubernetes node over a selected period of time, including the *Summary and Resource Information* table and the *Pods* table. For more information, see Node metrics *on page 67*.

Figure 8. Kubernetes Nodes Explorer view General Tab



Container Environment > Kubernetes Node: [redacted]

Saturday, March 14, 2020 2:43:34 PM - Now 60 minutes | Reports

Kubernetes Node: [redacted]

Alarms Severity	Fatal	Critical	Warning
Alarms Count	🟥		1

General | Metrics | Events

Summary and Resource Information

Name	[redacted]	**Kubelet Endpoint**	10,250
Architecture	[redacted]	**Operating System**	linux
Kernel Version	[redacted]	**Container Runtime Version**	[redacted]
Kube Proxy Version	[redacted]	**Kubelet Version**	[redacted]
Boot ID	[redacted]	**Machine ID**	[redacted]
Pod CIDR	[redacted]	**External ID**	
Provider ID	[redacted]	**Unschedulable**	false
Labels	[redacted]	**Annotations**	volumes.kubernetes.io/controller-managed-attach-detach=true node.alpha.kubernetes.io/ttl=0

Capacity	[redacted]	**Allocatable**	attachable-volumes-azure-disk=8
			cpu=3860m
			ephemeral-storage=187150362315
			hugepages-1Gi=0
			hugepages-2Mi=0
			memory=12909668Ki
			pods=110

Status	Normal		
Addresses			
Hostname	[redacted]	**InternalIP**	[redacted]

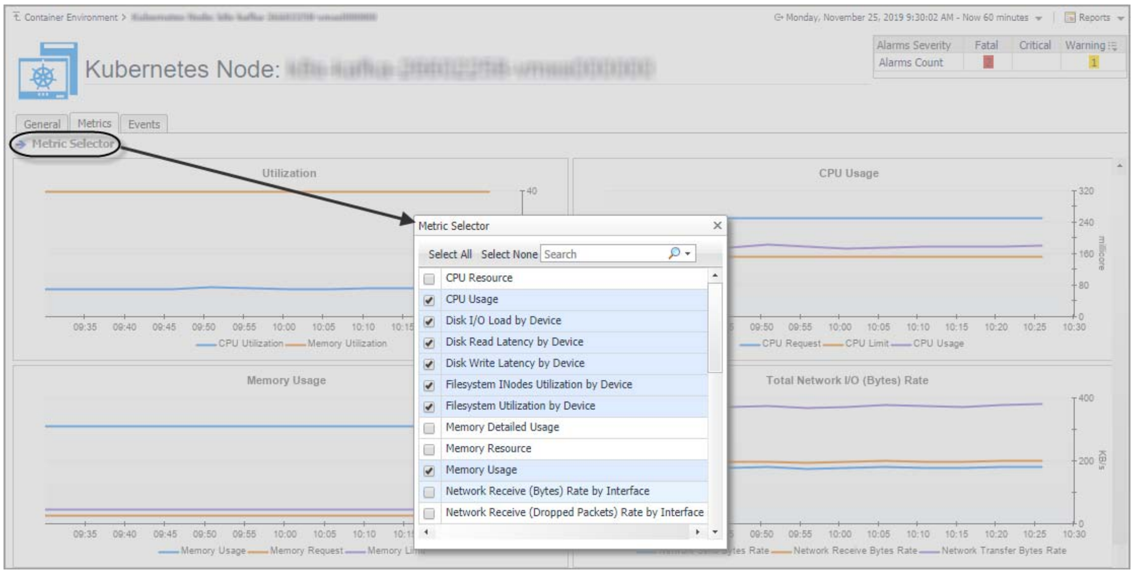
Pods

Search

Alarm	Name	Status	CPU Usage Rate	Memory Usage ▲	Filesystem Usage	Cluster	Namespace	Containers	Init Containers
✓	[redacted]	Running	0.9 millicore(s)/s	26.0 MB	40.0 KB	Orbital_Denny	devops-monitoring	1	0
✓	[redacted]	Running	2.1 millicore(s)/s	32.7 MB	36.0 KB	Orbital_Denny	kube-system	1	0
✓	[redacted]	Running	1.0 millicore(s)/s	70.8 MB	40.0 KB	Orbital_Denny	opendistro-log	1	0

- *Metrics tab:* The *Metrics* tab displays a *Metric Selector* allowing you to choose the metrics to be plotted on this dashboard. Charts of *CPU Usage, Utilization, Memory Usage,* and *Network I/O* are presented by default.

Figure 9. Kubernetes Nodes Explorer view Metrics Tab


Container Environment > [redacted]
Monday, November 25, 2019 9:30:02 AM - Now 60 minutes
Reports
Kubernetes Node: [redacted]
Alarms Severity | Fatal | Critical | Warning
Alarms Count
General
Metrics
Events
Metric Selector
Utilization
CPU Utilization
Memory Utilization
CPU Usage
CPU Request
CPU Limit
CPU Usage
millicore
Memory Usage
Memory Usage
Memory Request
Memory Limit
Total Network I/O (Bytes) Rate
Network Receive Bytes Rate
Network Transfer Bytes Rate
KB/s

Metric Selector

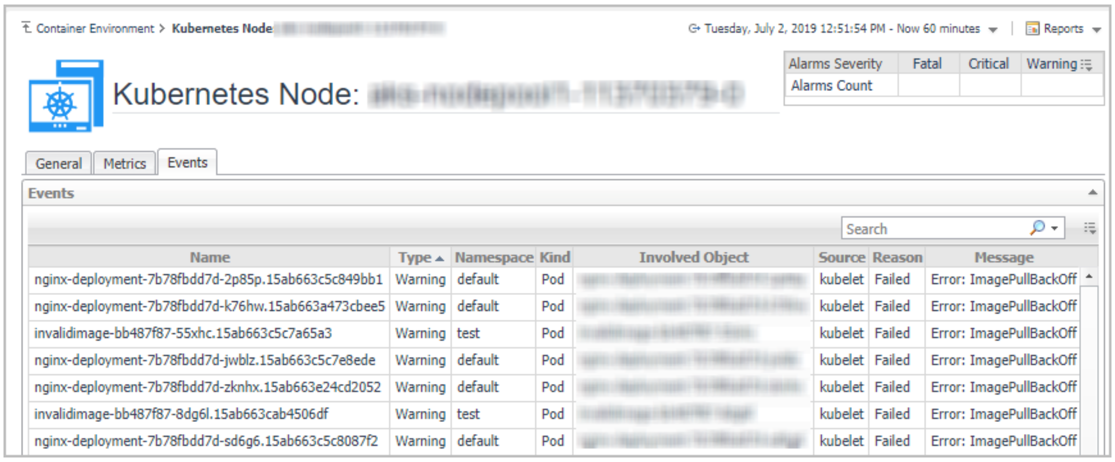
Select All | Select None | Search

- [ ] CPU Resource
- [x] CPU Usage
- [x] Disk I/O Load by Device
- [x] Disk Read Latency by Device
- [x] Disk Write Latency by Device
- [x] Filesystem INodes Utilization by Device
- [x] Filesystem Utilization by Device
- [ ] Memory Detailed Usage
- [ ] Memory Resource
- [x] Memory Usage
- [ ] Network Receive (Bytes) Rate by Interface
- [ ] Network Receive (Dropped Packets) Rate by Interface

- *Event* tab: The *Event* tab lists all the events occur on the nodes.
  - Name: name of the event.

- Type: type of the event, Warning or Normal.
- Namespace: namespace of where this event happens.
- Kind: type of the Kubernetes component on which this event occurs.
- Involved Object: name of the Kubernetes component on which this event occurs.
- Source: where this event has been triggered from.
- Reason: reason of this event.
- Message: detailed message of this event.

Figure 10. Kubernetes Nodes Explorer view Events tab


Container Environment > Kubernetes Node [redacted]
Tuesday, July 2, 2019 12:51:54 PM - Now 60 minutes
Reports
Kubernetes Node: [redacted]
General
Metrics
Events
Search

Alarms Severity	Fatal	Critical	Warning
Alarms Count			

Events

Name	Type	Namespace	Kind	Involved Object	Source	Reason	Message
nginx-deployment-7b78fbdd7d-2p85p.15ab663c5c849bb1	Warning	default	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
nginx-deployment-7b78fbdd7d-k76hw.15ab663a473cbee5	Warning	default	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
invalidimage-bb487f87-55xhc.15ab663c5c7a65a3	Warning	test	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
nginx-deployment-7b78fbdd7d-jwblz.15ab663c5c7e8ede	Warning	default	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
nginx-deployment-7b78fbdd7d-zknhx.15ab663e24cd2052	Warning	default	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
invalidimage-bb487f87-8dg6l.15ab663cab4506df	Warning	test	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff
nginx-deployment-7b78fbdd7d-sd6g6.15ab663c5c8087f2	Warning	default	Pod	[redacted]	kubelet	Failed	Error: ImagePullBackOff

Monitoring Kubernetes Clusters

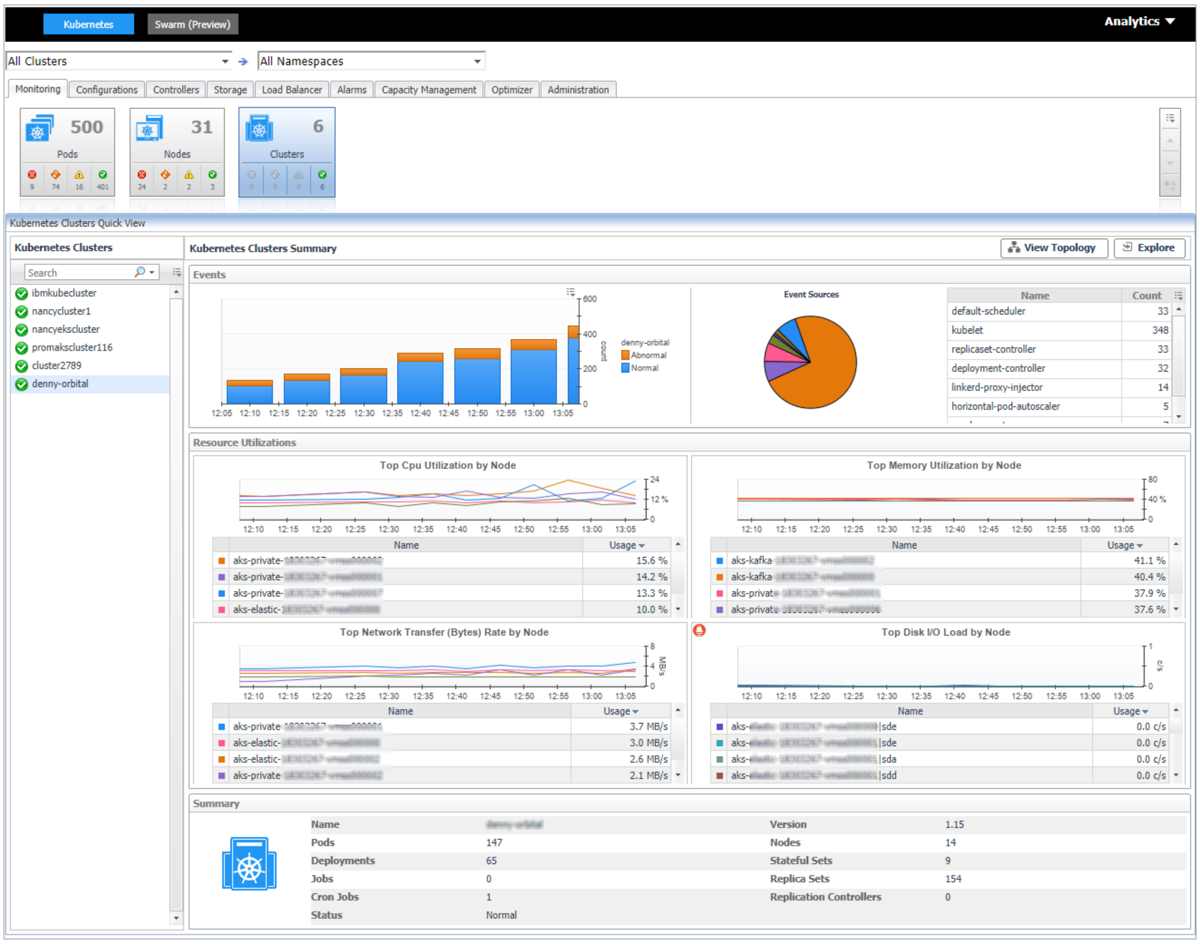
Kubernetes cluster is a group of kubernetes resources. There are two kinds of nodes inside a cluster, Kubernetes master and Kubernetes nodes. Kubernetes master is responsible for maintaining the desired state of your cluster which Kubernetes node is responsible to run your application and cloud workflows.The *Kubernetes Cluster Quick View*, which appears after clicking **Monitoring > Clusters**, shows the data collected about the selected clusters and namespaces. This view consists of the following two panes:

- The **Kubernetes Clusters** tree view, which appears on the left of *Kubernetes Clusters Quick View*, lists the clusters existing in the monitored Kubernetes environment.
- The Kubernetes Clusters Summary view, which appears on the right after you select an individual cluster in the **Kubernetes Clusters** tree view.

Kubernetes Clusters Summary view

The **Kubernetes Clusters Summary** view appears on the right when you select a node in the **Kubernetes Clusters** tree view.

Figure 11. Kubernetes Clusters Summary view



Kubernetes Clusters

- ibmkubecluster
- nancycluster1
- nancyekscluster
- promakscluster116
- cluster2789
- denny-orbital

Kubernetes Clusters Summary

Events

Name	Count
default-scheduler	33
kubelet	348
replicaset-controller	33
deployment-controller	32
linkerd-proxy-injector	14
horizontal-pod-autoscaler	5

Resource Utilizations

Top Cpu Utilization by Node

Name	Usage
aks-private-[redacted]	15.6 %
aks-private-[redacted]	14.2 %
aks-private-[redacted]	13.3 %
aks-elastic-[redacted]	10.0 %

Top Memory Utilization by Node

Name	Usage
aks-kafka-[redacted]	41.1 %
aks-kafka-[redacted]	40.4 %
aks-private-[redacted]	37.9 %
aks-private-[redacted]	37.6 %

Top Network Transfer (Bytes) Rate by Node

Name	Usage
aks-private-[redacted]	3.7 MB/s
aks-elastic-[redacted]	3.0 MB/s
aks-elastic-[redacted]	2.6 MB/s
aks-private-[redacted]	2.1 MB/s

Top Disk I/O Load by Node

Name	Usage
aks-[redacted]|sde	0.0 c/s
aks-[redacted]|sde	0.0 c/s
aks-[redacted]|sda	0.0 c/s
aks-[redacted]|sdd	0.0 c/s

Summary

Name	[redacted]	Version	1.15
Pods	147	Nodes	14
Deployments	65	Stateful Sets	9
Jobs	0	Replica Sets	154
Cron Jobs	1	Replication Controllers	0
Status	Normal		

The **Kubernetes Clusters Summary** view displays the following data:

- *Events*: The events occur on the selected Kubernetes cluster over a selected period of time, which includes:
  - The column chart on the left: Shows the timeline of the occurred events, which indicates at what time and how many events have occurred.
  - The pie chart on the right- Event Sources: Shows the events distribution for different event source.
- *Resource Utilizations*: The resource utilization for the selected Kubernetes cluster over a selected period of time, which includes the following:
  - *Top CPU Utilization by Node*: shows the nodes that CPU Utilization are top N highest.
  - *Top Memory Utilization by Node*: shows the nodes that Memory Utilization are top N highest.
  - *Top Network Transfer (Bytes) Rate by Node*: shows the nodes that Network Transfer (Bytes) Rate are top N highest.
  - *Top Disk I/O Load by Node*: shows the nodes that Disk I/O Load are top N highest.
- *Summary*: Displays the detailed information about the selected Kubernetes cluster, including *Name, Version, Pods, Nodes, Deployments, Stateful Sets, Jobs,* and *Replica Sets*.

Click **Explore** on the upper right of the **Kubernetes Clusters Summary** view to open the Cluster Explorer view, which shows more detailed information about this Kubernetes cluster.

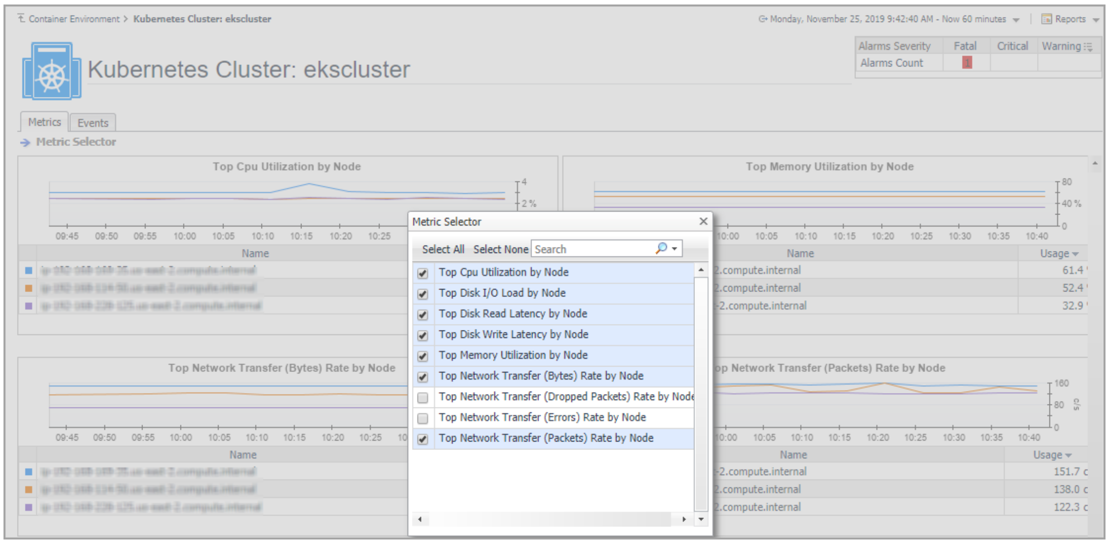
Click **View Topology** on the upper right of the **Kubernetes Clusters Summary** view to open the Cluster Topology view, which shows the topology graph from the application accessible aspect.

Cluster Explorer view

The *Cluster Explorer* view opens when you click **Explore** in the Kubernetes Clusters Summary view, which includes the following tabs:

- *Metrics tab:* The *Metrics* tab displays a *Metric Selector* allowing you to choose the metrics to be plotted on this dashboard. Charts of *CPU Usage* and *Memory Usage* are presented by default.

Figure 12. Kubernetes Clusters Explorer view Metrics tab



Container Environment > Kubernetes Cluster: ekscluster
Monday, November 25, 2019 9:42:40 AM - Now 60 minutes
Reports
Alarms Severity
Fatal
Critical
Warning
Alarms Count
Kubernetes Cluster: ekscluster
Metrics
Events
Metric Selector
Top Cpu Utilization by Node
Top Memory Utilization by Node
Top Network Transfer (Bytes) Rate by Node
Top Network Transfer (Packets) Rate by Node
Name
Usage

Metric Selector

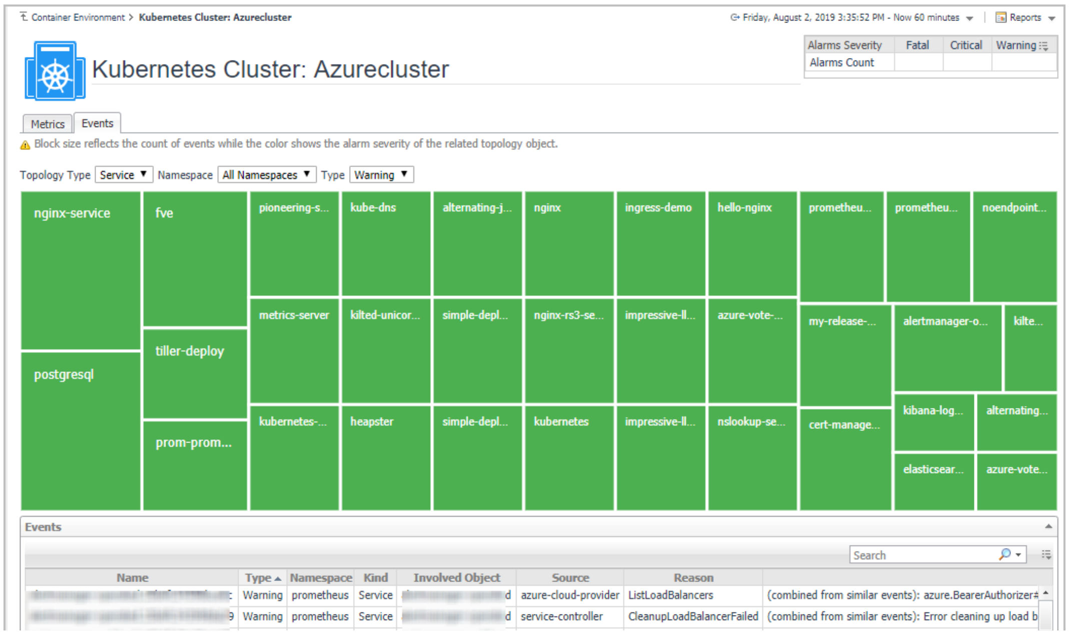
Select All | Select None | Search

- ✓ Top Cpu Utilization by Node
- ✓ Top Disk I/O Load by Node
- ✓ Top Disk Read Latency by Node
- ✓ Top Disk Write Latency by Node
- ✓ Top Memory Utilization by Node
- ✓ Top Network Transfer (Bytes) Rate by Node
- ☐ Top Network Transfer (Dropped Packets) Rate by Node
- ☐ Top Network Transfer (Errors) Rate by Node
- ✓ Top Network Transfer (Packets) Rate by Node

- *Events tab*: The Events tab shows a Heat Map of the events occur in this cluster. Heat maps will be refreshed automatically when you change either of the following fields:
  - *Topology Type*: Indicates the Kubernetes components on which the event occurs, including Pod, Node, and Service.
  - *Namespace*: Use the namespaces to filter the events.
  - *Type*: Indicates the severity of the event, including warning and normal.

NOTE: The color in the heatmap indicates the severity of component alarms.
Green: indicates normal. Yellow: indicates warning. Orange: indicates critical. Red: indicates fatal.

Figure 13. Kubernetes Clusters Explorer view Events tab



Alarms Severity	Fatal	Critical	Warning
Alarms Count			

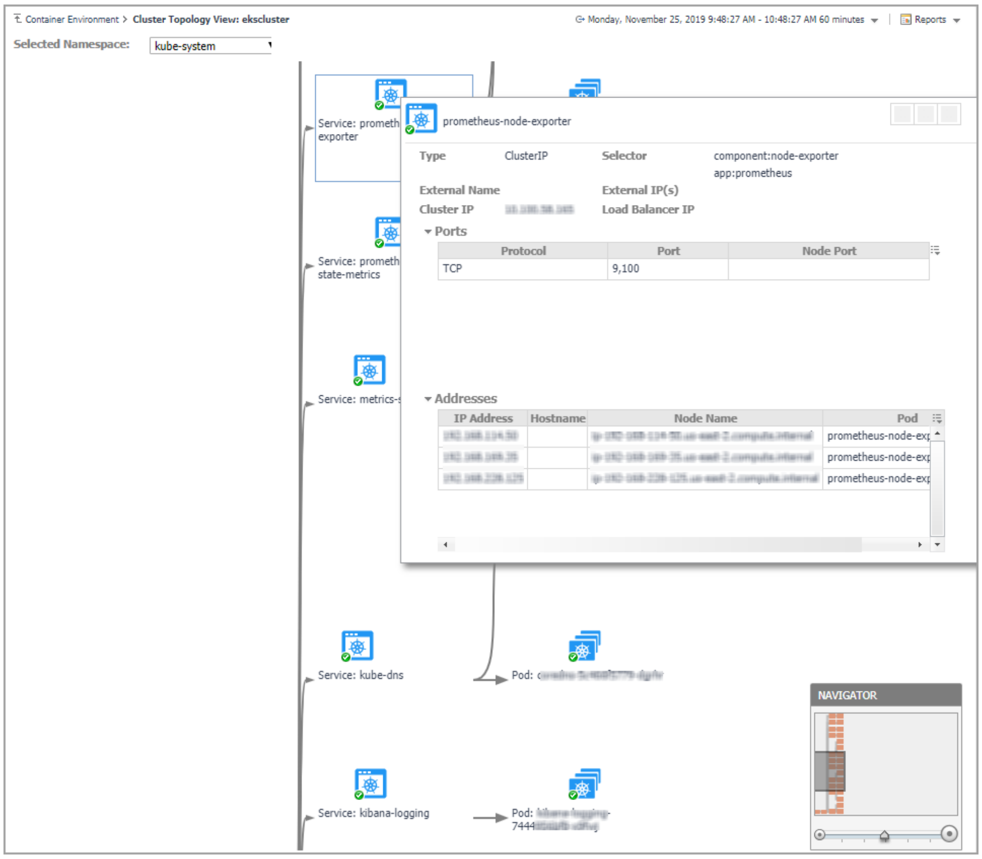
⚠ Block size reflects the count of events while the color shows the alarm severity of the related topology object.

Topology Type: Service ▼ Namespace: All Namespaces ▼ Type: Warning ▼

Name	Type ▲	Namespace	Kind	Involved Object	Source	Reason	
[redacted]:	Warning	prometheus	Service	[redacted]d	azure-cloud-provider	ListLoadBalancers	(combined from similar events): azure.BearerAuthorizer#
[redacted]9	Warning	prometheus	Service	[redacted]d	service-controller	CleanupLoadBalancerFailed	(combined from similar events): Error cleaning up load b

Cluster Topology view

Figure 14. Kubernetes Clusters Topology view



Type	ClusterIP	Selector	component:node-exporter app:prometheus
External Name		External IP(s)	
Cluster IP	[redacted]	Load Balancer IP	

Protocol	Port	Node Port
TCP	9,100	

IP Address	Hostname	Node Name	Pod
[redacted]		[redacted]	prometheus-node-exp
[redacted]		[redacted]	prometheus-node-exp
[redacted]		[redacted]	prometheus-node-exp

The *topology* view visualizes the relationships between the objects from the pods accessible aspect in your environment through an interactive dependency map. The map illustrates how different components relate to each other, and the levels of the available resources available to them. Click on Pod, another sub topology view will popup to show the relationship from pods controller to storage for the selected Pod. Click other components or click the Pod inside the sub topology view, an information view will popup to show alarms, basic information, some metrics. From the information popup view of Pod, Node, and Cluster, click the Explore button will navigate to the explorer view of the selected Pod/Node/Cluster.

The **NAVIGATOR** in the bottom-right corner allows you to easily set the zoom level by dragging the slider into the appropriate position.

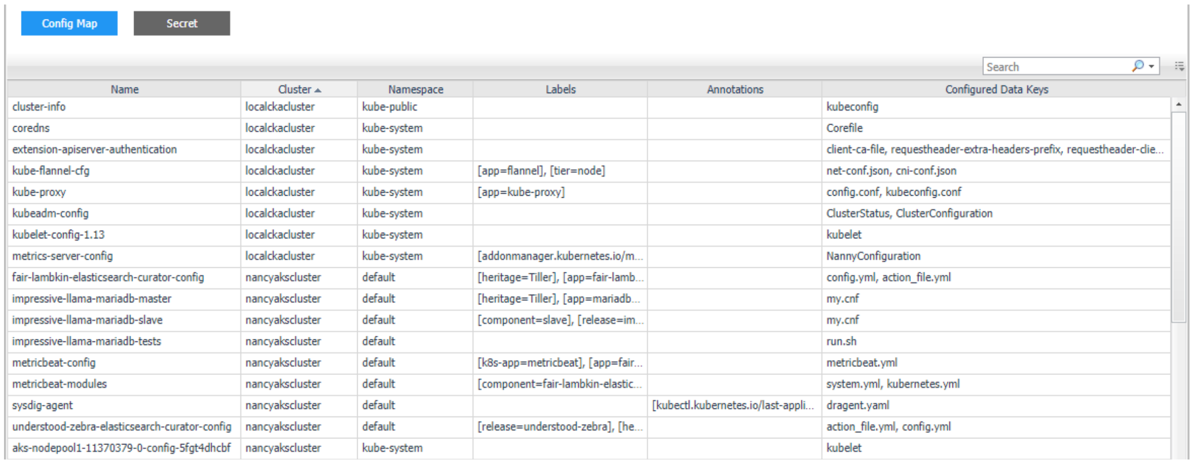
Monitoring Kubernetes Other Components

Kubernetes other components here including pods controllers, services, ingresses, persistent volumes, secrets and so on. All these components are grouped and displayed in tabs.

- Configurations
- Controllers
- Storage
- Load Balancer

Configurations

Figure 15. Kubernetes Configuration Dashboard



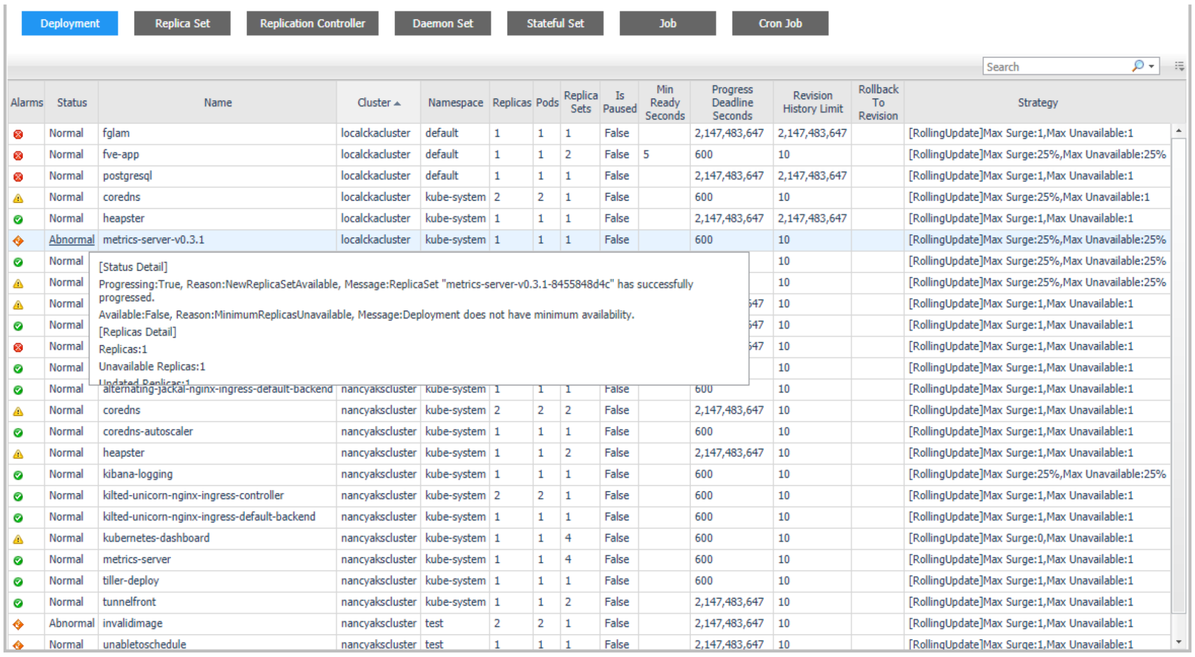
Name	Cluster	Namespace	Labels	Annotations	Configured Data Keys
cluster-info	localckacluster	kube-public			kubeconfig
coredns	localckacluster	kube-system			Corefile
extension-apiserver-authentication	localckacluster	kube-system			client-ca-file, requestheader-extra-headers-prefix, requestheader-clie...
kube-flannel-cfg	localckacluster	kube-system	[app=flannel], [tier=node]		net-conf.json, cni-conf.json
kube-proxy	localckacluster	kube-system	[app=kube-proxy]		config.conf, kubeconfig.conf
kubeadm-config	localckacluster	kube-system			ClusterStatus, ClusterConfiguration
kubelet-config-1.13	localckacluster	kube-system			kubelet
metrics-server-config	localckacluster	kube-system	[addonmanager.kubernetes.io/m...		NannyConfiguration
fair-lambkin-elasticsearch-curator-config	nancyakscluster	default	[heritage=Tiller], [app=fair-lamb...		config.yml, action_file.yml
impressive-llama-mariadb-master	nancyakscluster	default	[heritage=Tiller], [app=mariadb...		my.cnf
impressive-llama-mariadb-slave	nancyakscluster	default	[component=slave], [release=im...		my.cnf
impressive-llama-mariadb-tests	nancyakscluster	default			run.sh
metricbeat-config	nancyakscluster	default	[k8s-app=metricbeat], [app=fair...		metricbeat.yml
metricbeat-modules	nancyakscluster	default	[component=fair-lambkin-elastic...		system.yml, kubernetes.yml
sysdig-agent	nancyakscluster	default		[kubectl.kubernetes.io/last-appli...	dragent.yaml
understood-zebra-elasticsearch-curator-config	nancyakscluster	default	[release=understood-zebra], [he...		action_file.yml, config.yml
aks-nodepool1-11370379-0-config-5fgt4dhcbf	nancyakscluster	kube-system			kubelet

The *Configurations* dashboard includes Kubernetes Secret and Config Map.

- A Kubernetes Secret is an object that contains a small amount of sensitive data, such as a password, a token, or a key. Such information might otherwise be put in a Pod specification or in an image; putting it in a Secret object allows for more control over how it is used, and reduces the risk of accidental exposure.
- A Kubernetes Config Map binds configuration files, command-line arguments, environment variables, port numbers, and other configuration artifacts to your Pods' containers and system components at runtime. Config maps allow you to separate your configurations from your Pods and components, which helps keep your workloads portable, makes their configurations easier to change and manage, and prevents hardcoding configuration data to Pod specifications.

Controllers

Figure 16. Kubernetes Controllers Dashboard



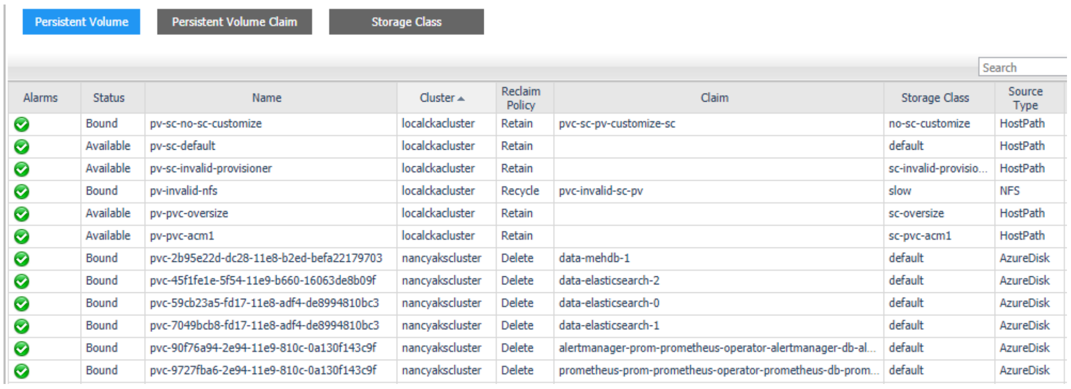
Alarms	Status	Name	Cluster ▲	Namespace	Replicas	Pods	Replica Sets	Is Paused	Min Ready Seconds	Progress Deadline Seconds	Revision History Limit	Rollback To Revision	Strategy
✖	Normal	fglam	localckacluster	default	1	1	1	False		2,147,483,647	2,147,483,647		[RollingUpdate]Max Surge:1,Max Unavailable:1
✖	Normal	fve-app	localckacluster	default	1	1	2	False	5	600	10		[RollingUpdate]Max Surge:25%,Max Unavailable:25%
✖	Normal	postgresql	localckacluster	default	1	1	1	False		2,147,483,647	2,147,483,647		[RollingUpdate]Max Surge:1,Max Unavailable:1
⚠	Normal	coredns	localckacluster	kube-system	2	2	1	False		600	10		[RollingUpdate]Max Surge:25%,Max Unavailable:1
✓	Normal	heapster	localckacluster	kube-system	1	1	1	False		2,147,483,647	2,147,483,647		[RollingUpdate]Max Surge:1,Max Unavailable:1
◆	Abnormal	metrics-server-v0.3.1	localckacluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:25%,Max Unavailable:25%
✓	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:25%,Max Unavailable:25%
⚠	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:25%,Max Unavailable:25%
⚠	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✖	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	alternating-jackal-nginx-ingress-default-backend	nancyakscluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
⚠	Normal	coredns	nancyakscluster	kube-system	2	2	2	False		2,147,483,647	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	coredns-autoscaler	nancyakscluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
⚠	Normal	heapster	nancyakscluster	kube-system	1	1	2	False		2,147,483,647	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	kibana-logging	nancyakscluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:25%,Max Unavailable:25%
✓	Normal	kilted-unicorn-nginx-ingress-controller	nancyakscluster	kube-system	2	2	1	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	kilted-unicorn-nginx-ingress-default-backend	nancyakscluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
⚠	Normal	kubernetes-dashboard	nancyakscluster	kube-system	1	1	4	False		600	10		[RollingUpdate]Max Surge:0,Max Unavailable:1
✓	Normal	metrics-server	nancyakscluster	kube-system	1	1	4	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	tiller-deploy	nancyakscluster	kube-system	1	1	1	False		600	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
✓	Normal	tunnelfront	nancyakscluster	kube-system	1	1	2	False		2,147,483,647	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
◆	Abnormal	invalidimage	nancyakscluster	test	2	2	1	False		2,147,483,647	10		[RollingUpdate]Max Surge:1,Max Unavailable:1
◆	Normal	unabletoschedule	nancyakscluster	test	1	1	1	False		2,147,483,647	10		[RollingUpdate]Max Surge:1,Max Unavailable:1

[Status Detail]
Progressing:True, Reason:NewReplicaSetAvailable, Message:ReplicaSet "metrics-server-v0.3.1-8455848d4c" has successfully progressed.
Available:False, Reason:MinimumReplicasUnavailable, Message:Deployment does not have minimum availability.
[Replicas Detail]
Replicas:1
Unavailable Replicas:1
Updated Replicas:1

A controller manages a set of pods and ensures that the cluster is in the specified state. Instead of manually creating a pod, controllers can be used to create pods and to manage the pods. For example, the pods maintained by a replication controller are automatically replaced if they fail, get deleted, or are terminated. The *Controllers* dashboard presents the information related to the following controller types: *Deployment, Replica Set, Replication Controller, Daemon Set, Stateful Set, Job,* and *Cron Job*.

Storage

Figure 17. Kubernetes Storage Dashboard



Alarms	Status	Name	Cluster ▲	Reclaim Policy	Claim	Storage Class	Source Type
✓	Bound	pv-sc-no-sc-customize	localckacluster	Retain	pvc-sc-pv-customize-sc	no-sc-customize	HostPath
✓	Available	pv-sc-default	localckacluster	Retain		default	HostPath
✓	Available	pv-sc-invalid-provisioner	localckacluster	Retain		sc-invalid-provisio...	HostPath
✓	Bound	pv-invalid-nfs	localckacluster	Recycle	pvc-invalid-sc-pv	slow	NFS
✓	Available	pv-pvc-oversize	localckacluster	Retain		sc-oversize	HostPath
✓	Available	pv-pvc-acm1	localckacluster	Retain		sc-pvc-acm1	HostPath
✓	Bound	pvc-2b95e22d-dc28-11e8-b2ed-befa22179703	nancyakscluster	Delete	data-mehdb-1	default	AzureDisk
✓	Bound	pvc-45f1fe1e-5f54-11e9-b660-16063de8b09f	nancyakscluster	Delete	data-elasticsearch-2	default	AzureDisk
✓	Bound	pvc-59cb23a5-fd17-11e8-adf4-de8994810bc3	nancyakscluster	Delete	data-elasticsearch-0	default	AzureDisk
✓	Bound	pvc-7049bcb8-fd17-11e8-adf4-de8994810bc3	nancyakscluster	Delete	data-elasticsearch-1	default	AzureDisk
✓	Bound	pvc-90f76a94-2e94-11e9-810c-0a130f143c9f	nancyakscluster	Delete	alertmanager-prom-prometheus-operator-alertmanager-db-al...	default	AzureDisk
✓	Bound	pvc-9727fba6-2e94-11e9-810c-0a130f143c9f	nancyakscluster	Delete	prometheus-prom-prometheus-operator-prometheus-db-prom...	default	AzureDisk

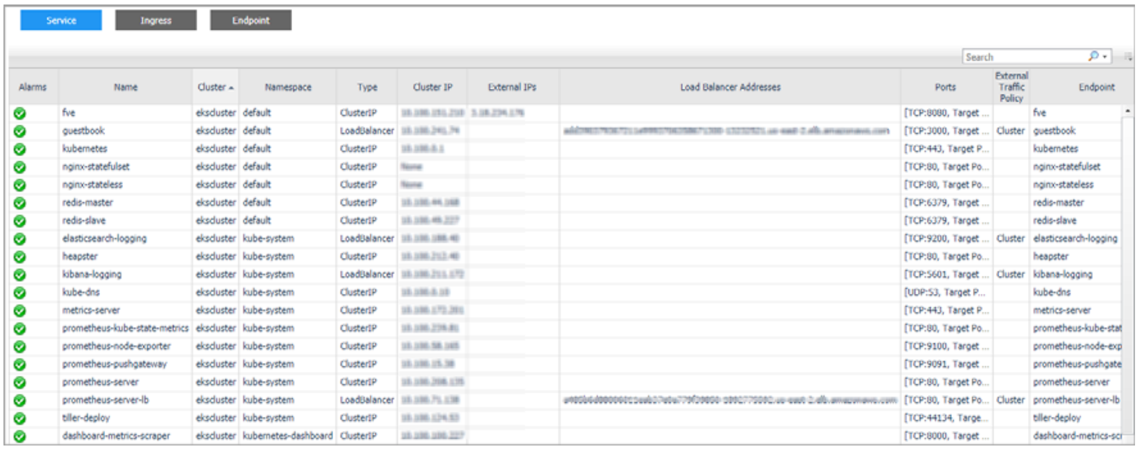
The Kubernetes storage contains volumes, storage class, persistent volume, and persistent volume claim. Volumes are on-disk files used by the containers for persistent their data as well as sharing with other containers.The *Storage* dashboard shows the information about the following storage classes:

- *Storage Class* provides a way for the administrator to describe the "class" of storage they offer.
- *Persistent Volume* subsystem provides an API for users and administrators that abstracts details of how storage is provided from how it is consumed.

- *Persistent Volume Claim* is used for dynamic volume provisioning which allow storage volumes to be created on-demand.

Load Balancer

Figure 18. Kubernetes Load Balancer Dashboard

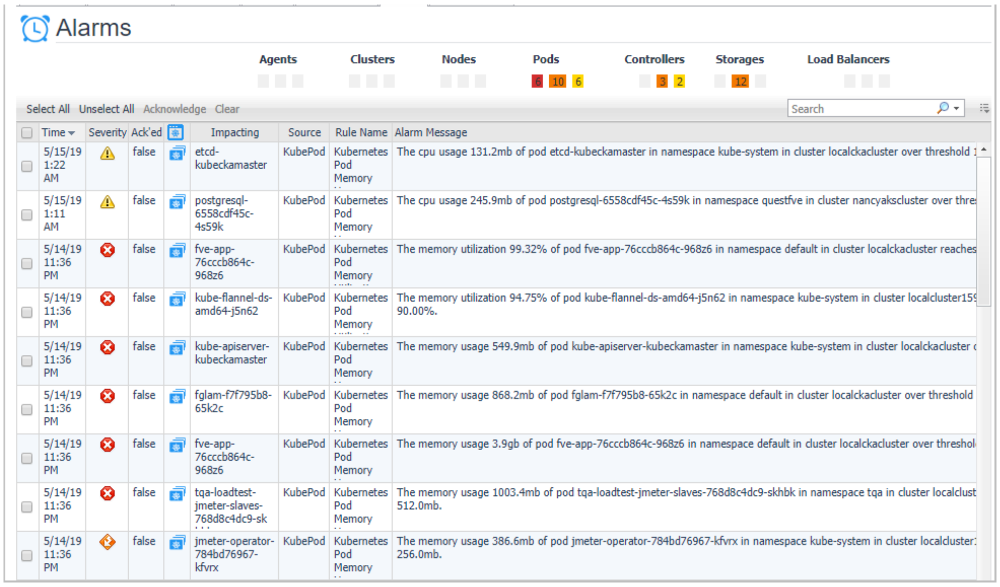


Alarms	Name	Cluster	Namespace	Type	Cluster IP	External IPs	Load Balancer Addresses	Ports	External Traffic Policy	Endpoint
✓	fve	eksduster	default	ClusterIP	[redacted]	[redacted]		[TCP:8080, Target ...		fve
✓	guestbook	eksduster	default	LoadBalancer	[redacted]		[redacted]	[TCP:3000, Target ...	Cluster	guestbook
✓	kubernetes	eksduster	default	ClusterIP	[redacted]			[TCP:443, Target P...		kubernetes
✓	nginx-statefulset	eksduster	default	ClusterIP	[redacted]			[TCP:80, Target Po...		nginx-statefulset
✓	nginx-stateless	eksduster	default	ClusterIP	[redacted]			[TCP:80, Target Po...		nginx-stateless
✓	redis-master	eksduster	default	ClusterIP	[redacted]			[TCP:6379, Target ...		redis-master
✓	redis-slave	eksduster	default	ClusterIP	[redacted]			[TCP:6379, Target ...		redis-slave
✓	elasticsearch-logging	eksduster	kube-system	LoadBalancer	[redacted]			[TCP:9200, Target ...	Cluster	elasticsearch-logging
✓	heapster	eksduster	kube-system	ClusterIP	[redacted]			[TCP:80, Target Po...		heapster
✓	kibana-logging	eksduster	kube-system	LoadBalancer	[redacted]			[TCP:5601, Target ...	Cluster	kibana-logging
✓	kube-dns	eksduster	kube-system	ClusterIP	[redacted]			[UDP:53, Target P...		kube-dns
✓	metrics-server	eksduster	kube-system	ClusterIP	[redacted]			[TCP:443, Target P...		metrics-server
✓	prometheus-kube-state-metrics	eksduster	kube-system	ClusterIP	[redacted]			[TCP:80, Target Po...		prometheus-kube-stat
✓	prometheus-node-exporter	eksduster	kube-system	ClusterIP	[redacted]			[TCP:9100, Target ...		prometheus-node-exp
✓	prometheus-pushgateway	eksduster	kube-system	ClusterIP	[redacted]			[TCP:9091, Target ...		prometheus-pushgate
✓	prometheus-server	eksduster	kube-system	ClusterIP	[redacted]			[TCP:80, Target Po...		prometheus-server
✓	prometheus-server-lb	eksduster	kube-system	LoadBalancer	[redacted]		[redacted]	[TCP:80, Target Po...	Cluster	prometheus-server-lb
✓	tiller-deploy	eksduster	kube-system	ClusterIP	[redacted]			[TCP:44134, Targe...		tiller-deploy
✓	dashboard-metrics-scraper	eksduster	kubernetes-dashboard	ClusterIP	[redacted]			[TCP:8000, Target ...		dashboard-metrics-scr

The *Load Balancer* dashboard includes information about Kubernetes service, endpoint, and ingress. A Kubernetes ingress can provide load balancing, SSL termination, and name-based virtual hosting. A Kubernetes service is an abstraction which defines a logical set of pods and a policy by which to access them - sometime called micro-services. Kubernetes will update the endpoint whenever the set of pods in a service changes.

Alarms

Figure 19. Kubernetes Alarms Dashboard



	Time	Severity	Ack'ed		Impacting	Source	Rule Name	Alarm Message
☐	5/15/19 1:22 AM	⚠	false		etcd-kubeckamaster	KubePod	Kubernetes Pod Memory ...	The cpu usage 131.2mb of pod etcd-kubeckamaster in namespace kube-system in cluster localckacluster over threshold 1
☐	5/15/19 1:11 AM	⚠	false		postgresql-6558cdf45c-4s59k	KubePod	Kubernetes Pod Memory ...	The cpu usage 245.9mb of pod postgresql-6558cdf45c-4s59k in namespace questfve in cluster nancyakscluster over thre
☐	5/14/19 11:36 PM	✖	false		fve-app-76cccb864c-968z6	KubePod	Kubernetes Pod Memory ...	The memory utilization 99.32% of pod fve-app-76cccb864c-968z6 in namespace default in cluster localckacluster reaches
☐	5/14/19 11:36 PM	✖	false		kube-flannel-ds-amd64-j5n62	KubePod	Kubernetes Pod Memory ...	The memory utilization 94.75% of pod kube-flannel-ds-amd64-j5n62 in namespace kube-system in cluster localcluster15 90.00%.
☐	5/14/19 11:36 PM	✖	false		kube-apiserver-kubeckamaster	KubePod	Kubernetes Pod Memory ...	The memory usage 549.9mb of pod kube-apiserver-kubeckamaster in namespace kube-system in cluster localckacluster
☐	5/14/19 11:36 PM	✖	false		fglam-f7f795b8-65k2c	KubePod	Kubernetes Pod Memory ...	The memory usage 868.2mb of pod fglam-f7f795b8-65k2c in namespace default in cluster localckacluster over threshold
☐	5/14/19 11:36 PM	✖	false		fve-app-76cccb864c-968z6	KubePod	Kubernetes Pod Memory ...	The memory usage 3.9gb of pod fve-app-76cccb864c-968z6 in namespace default in cluster localckacluster over threshol
☐	5/14/19 11:36 PM	✖	false		tqa-loadtest-jmeter-slaves-768d8c4dc9-sk ...	KubePod	Kubernetes Pod Memory ...	The memory usage 1003.4mb of pod tqa-loadtest-jmeter-slaves-768d8c4dc9-skhbk in namespace tqa in cluster localclust 512.0mb.
☐	5/14/19 11:36 PM	◆	false		jmeter-operator-784bd76967-kfvrx	KubePod	Kubernetes Pod Memory ...	The memory usage 386.6mb of pod jmeter-operator-784bd76967-kfvrx in namespace kube-system in cluster localcluster 256.0mb.

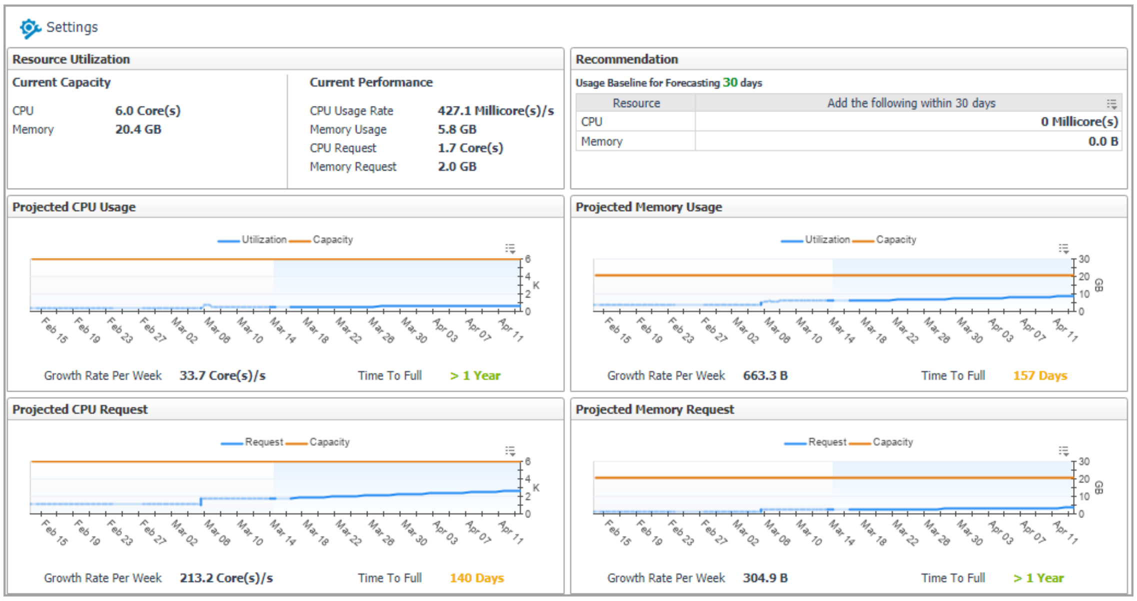
The *Alarms* dashboard displays a list of alarms generated against the monitored Kubernetes environment. Use this view to quickly identify any potential problems related to a specific Kubernetes component.

Capacity Management

Foglight for Container Management provide capacity management feature for Kubernetes. This feature uses historical data to predict the trend and usage within a specific future period.

NOTE: If the Capacity Management tab is not displayed, ensure the following:
1. You have purchased a license for Capacity Management. If not, contact Quest Support to purchase a license.
2. You have the Container Administrator role.

Figure 20. Capacity Management for Kubernetes



Resource Utilization

Current Capacity		Current Performance	
CPU	6.0 Core(s)	CPU Usage Rate	427.1 Millicore(s)/s
Memory	20.4 GB	Memory Usage	5.8 GB
		CPU Request	1.7 Core(s)
		Memory Request	2.0 GB

Recommendation

Usage Baseline for Forecasting 30 days

Resource	Add the following within 30 days
CPU	0 Millicore(s)
Memory	0.0 B

Projected CPU Usage

Growth Rate Per Week 33.7 Core(s)/s Time To Full > 1 Year

Projected Memory Usage

Growth Rate Per Week 663.3 B Time To Full 157 Days

Projected CPU Request

Growth Rate Per Week 213.2 Core(s)/s Time To Full 140 Days

Projected Memory Request

Growth Rate Per Week 304.9 B Time To Full > 1 Year

The Capacity Management dashboard contains the following fields:

- *Setting*: Click to change the following values:
  - *Baseline for Forecasting*: Defines the historical period used for the calculations of metric views, current capacity, and recommended resources in the Resource Utilization view. The default value is *60 Days Trending*.
  - *Time Frame*: Defines the predicted period for calculating metric views, current capacity, and recommended resources in the Resource Utilization view. The default value is *Next 30 Days*.
- *Resource Utilization*:
  - *Current Capacity*: current resource capacity.
  - *Current Performance*: current resource usage.
- *Recommendation*:

  In this section, it shows how many resources are recommended to be added in the current trend, so as to meet the predicated usage.
- *Projected CPU/Memory Usage*: Shows the historical data and the predicted usage trend within the configured future period.
- *Projected CPU/Memory Request*: Shows the historical data and the predicted request trend within the configured future period.
  - *Utilization*: usage.
  - *Capacity*: upper bound which the usage might reach.

- *Growth Rate per Week*: growth amount of the resource.
- *Time to Full*: how many days the resource usage/request will reach the capacity.

NOTE: If a value *Never* is displayed at *Time to Full*, which means the usage/request trend is declining and the usage/request will never reach the capacity.

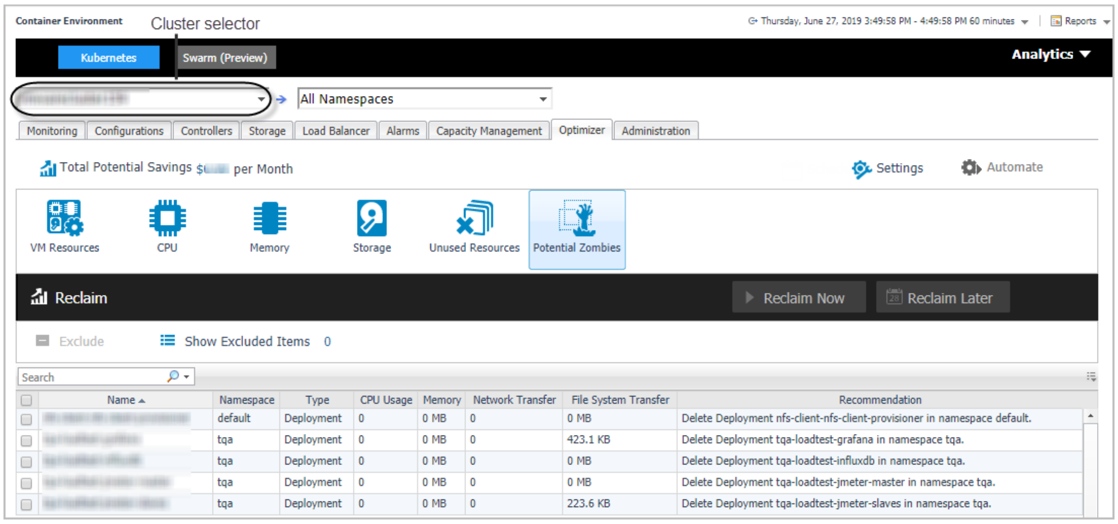
Optimizer

The *Optimizer* view appears after clicking **Container** > **Kubernetes** > **Optimizer**.

NOTE: If the Optimizer tab is not displayed, ensure the following:
1. You have purchased a license for Optimizer. If not, contact Quest Support to purchase a license.
2. You have the Container Administrator role.

NOTE: The displayed views are varied if the cluster hosts are monitored by the agents of VMware, AWS, or Azure.

Figure 21. Kubernetes Optimizer Dashboard



Name	Namespace	Type	CPU Usage	Memory	Network Transfer	File System Transfer	Recommendation
[redacted]	default	Deployment	0	0 MB	0	0 MB	Delete Deployment nfs-client-nfs-client-provisioner in namespace default.
[redacted]	tqa	Deployment	0	0 MB	0	423.1 KB	Delete Deployment tqa-loadtest-grafana in namespace tqa.
[redacted]	tqa	Deployment	0	0 MB	0	0 MB	Delete Deployment tqa-loadtest-influxdb in namespace tqa.
[redacted]	tqa	Deployment	0	0 MB	0	0 MB	Delete Deployment tqa-loadtest-jmeter-master in namespace tqa.
[redacted]	tqa	Deployment	0	0 MB	0	223.6 KB	Delete Deployment tqa-loadtest-jmeter-slaves in namespace tqa.

The Optimizer view includes the following elements:

- **Cluster Selector**. The cluster selector is located at the top of the Optimizer view and allows you to select the environment that you want to optimize.

  NOTE: The Namespace selector doesn't work for Optimizer dashboard.

- **Settings**. The Settings dialog box is used to change the time period and properties that are used for calculation. For more information, see Settings on page 42.
- **Automate**. Use the Automate menu to set the criteria for automatically sending recommendations for improvements. Currently, this button only functions for **CPU** and **Memory** when a VMware cluster is selected.
- **Reclaim Now** and **Reclaim Later** buttons. System administrator can select a VM from the list and review the Reclaiming Savings bar for information about how many resources can be reclaimed.

NOTE: The **Reclaim Now** and **Reclaim Later** buttons are enabled only after selecting a checkbox from the table. Currently, the two buttons only function for **VM Resources**, **CPU**, and **Memory** when a VMware cluster is selected.
The **Automate**, **Reclaim Now**, and **Reclaim Later** buttons are displayed only when a VMware cluster is selected.

- **Exclude** button. Select an object you want to exclude from the table to enable the Exclude button, and click Exclude. Then, this object is added to the list of excluded objects under a specific category.

- **Show Excluded Items** button. Click the Show Excluded Items button to view the excluded objects. The Settings dialog box appears. For more information, see Excluded tab on page 44.
- **VM Resources/VM Resizing**. Shows instance or virtual machine name, utilization, recommendations for both CPU and memory resources, and savings.
- **CPU**. Shows instance or virtual machine name, utilization, recommendations for CPU resource, and estimated savings.
- **Memory**. Shows instance or virtual machine name, utilization, recommendations for memory resource, and estimated savings.
- **Storage**. Shows virtual machine name, utilization, storage and modify recommendations, and savings.

NOTE: **VM Resizing** will be displayed when a cloud cluster is selected. **VM Resources**, **CPU**, **Memory**, and **Storage** will be displayed when a VMware cluster is selected.

- **Unused Resources** table. Detects and shows those unused resources in container environment.

  For example, persistent volume stays unused for more than 3 months. persistent volume stays in unbound status. This is due to the Unused Resources configuration in Settings.
- **Potential Zombies** table. Detects and shows the potential pod controllers in container environment, including Deployment, Daemon Set, Stateful Set, Replication Controller, as well as Pod that is not managed by any Pod Controller.

  For example, if all pods managed by a pod controller are zombies, then we might suggest you to delete the whole pod controller.

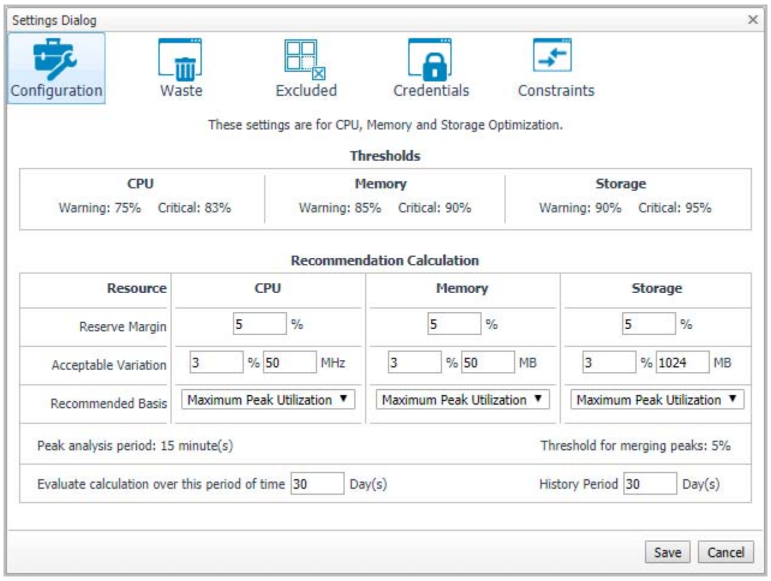
Settings

Use the Settings menu to define the default optimization settings for your environment. The Settings Dialog box provides information about the following components:

- Configuration tab
- Waste tab
- Excluded tab
- Credentials tab
- Constraints tab

Configuration tab

Figure 22. Configuration tab



These settings are for CPU, Memory and Storage Optimization.

Thresholds

CPU	Memory	Storage
Warning: 75% Critical: 83%	Warning: 85% Critical: 90%	Warning: 90% Critical: 95%

Recommendation Calculation

Resource	CPU	Memory	Storage
Reserve Margin	5 %	5 %	5 %
Acceptable Variation	3 % 50 MHz	3 % 50 MB	3 % 1024 MB
Recommended Basis	Maximum Peak Utilization	Maximum Peak Utilization	Maximum Peak Utilization

Peak analysis period: 15 minute(s) Threshold for merging peaks: 5%

Evaluate calculation over this period of time 30 Day(s) History Period 30 Day(s)

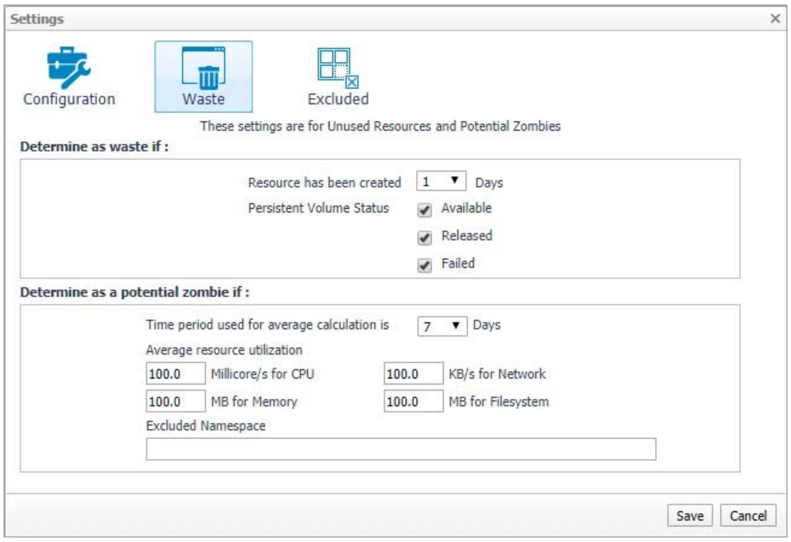
The **Configuration** tab provides the recommended settings for CPU, memory, and storage optimization.

- **Thresholds**. Provides the values of a resource metric that define the Warning and Critical levels (for CPU, memory, and storage).
- **Recommendation Calculation** area. Allows you to define the following parameters for optimizing the CPU, memory resources in your environment, Storage resources not supported at current version:

To save any changes made to the **Configuration** settings, click **Save** at the bottom of the tab.

Waste tab

Figure 23. Waste tab



These settings are for Unused Resources and Potential Zombies

Determine as waste if :

Resource has been created 1 Days

Persistent Volume Status ✓ Available ✓ Released ✓ Failed

Determine as a potential zombie if :

Time period used for average calculation is 7 Days

Average resource utilization

100.0 Millicore/s for CPU 100.0 KB/s for Network

100.0 MB for Memory 100.0 MB for Filesystem

Excluded Namespace

The **Waste** tab allows you to configure the settings for determining resources wasted in your environment. These include unused resources and potential zombie Pod controllers.

- *Determine as waste if:* used to filter Unused Resources.

- *Resource has been created [time] days*: Resources that has been created more than the set days will be considered here.
- *Persistent Volume Status*: By default, select *Available* and *Failed*. For detailed information, go to https://kubernetes.io/docs/concepts/storage/persistent-volumes/#phase.

- *Determine as a potential zombie if*: used to filter Potential Zombies.

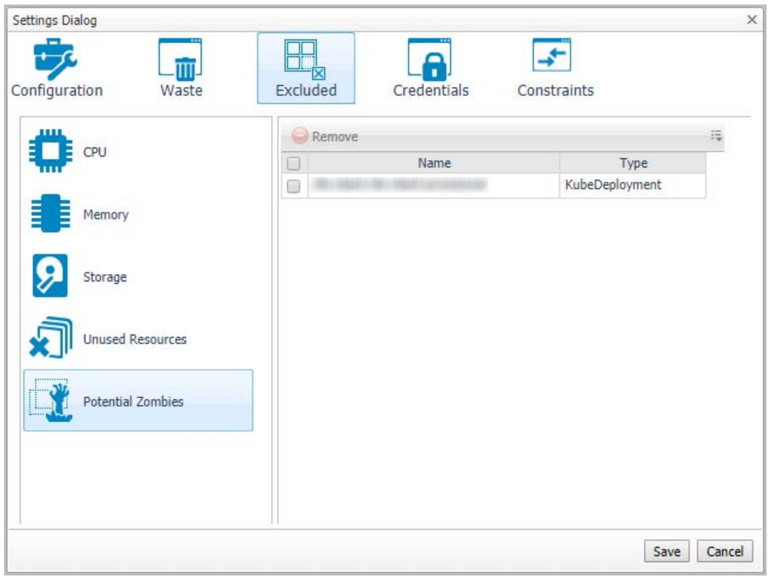
  In container environment, Potential Zombie Pod Controller is considered here, including Deployment, Daemon Set, Stateful Set, Replication Controller, and Pod that is not managed by any Pod Controller. Settings work for single pod managed by Pod Controller. If all pods or partial pods of a Pod Controller are considered as zombies, different recommendations will be generated.

  - *Time period used for average calculation is [time] days*: The average metrics for the pods are calculated, so a time range should be set to calculate the average value.
  - *Average resource utilization*: only if a pod's metrics satisfy all the conditions, it will be considered to be a potential zombie pod.
  - *Excluded Namespace*: pods in the namespace can be excluded in the Potential Zombies check.

To save any changes made to the **Waste** settings, click **Save** at the bottom of the tab.

Excluded tab

Figure 24. Excluded tab



	Name	Type
		KubeDeployment

The **Excluded** tab allows you to remove a resource from the list of excluded objects. The **Excluded** tab includes the following information:

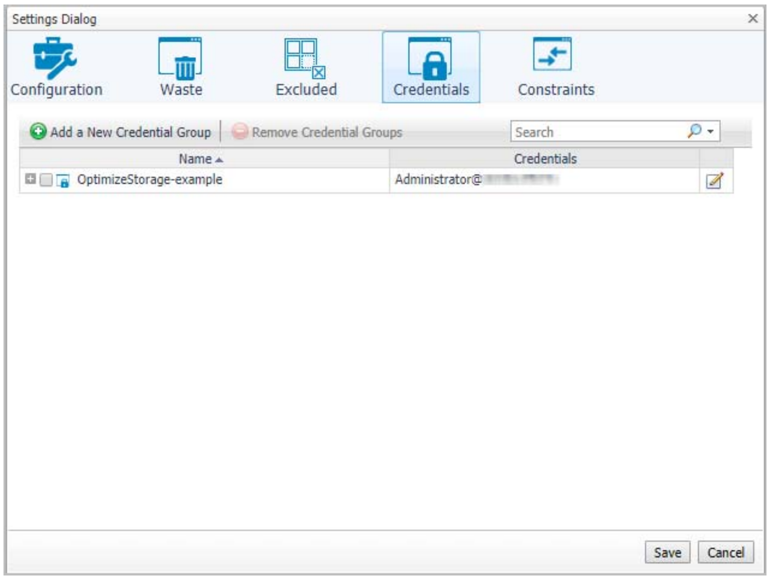
- On the left side, a navigation tree, that allows you to select the resource category.
- On the right side, the list of resources excluded from the selected category.

To remove resources from the list of **Excluded** objects, select the check boxes for these resources and click **Remove**. To save any changes made to the **Excluded** settings, click **Save** at the bottom of the tab.

The **Excluded** tab can also be accesses by clicking **Show Excluded Items** on the **Optimizer** tab.

Credentials tab

Figure 25. Credentials tab

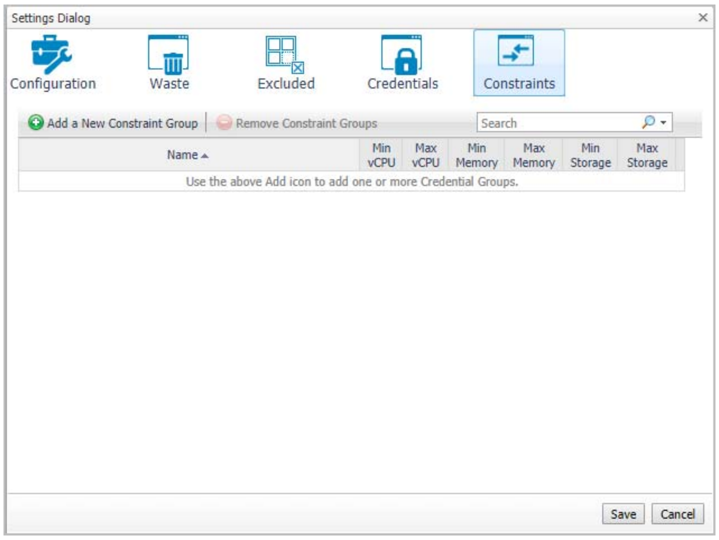


Name	Credentials
OptimizeStorage-example	Administrator@[redacted]

This tab is available in VMware environment. The **Credentials** tab allows you to add, edit, and remove credentials groups. This tab is only for the Storage rightsizer.

Constraints tab

Figure 26. Constraints tab



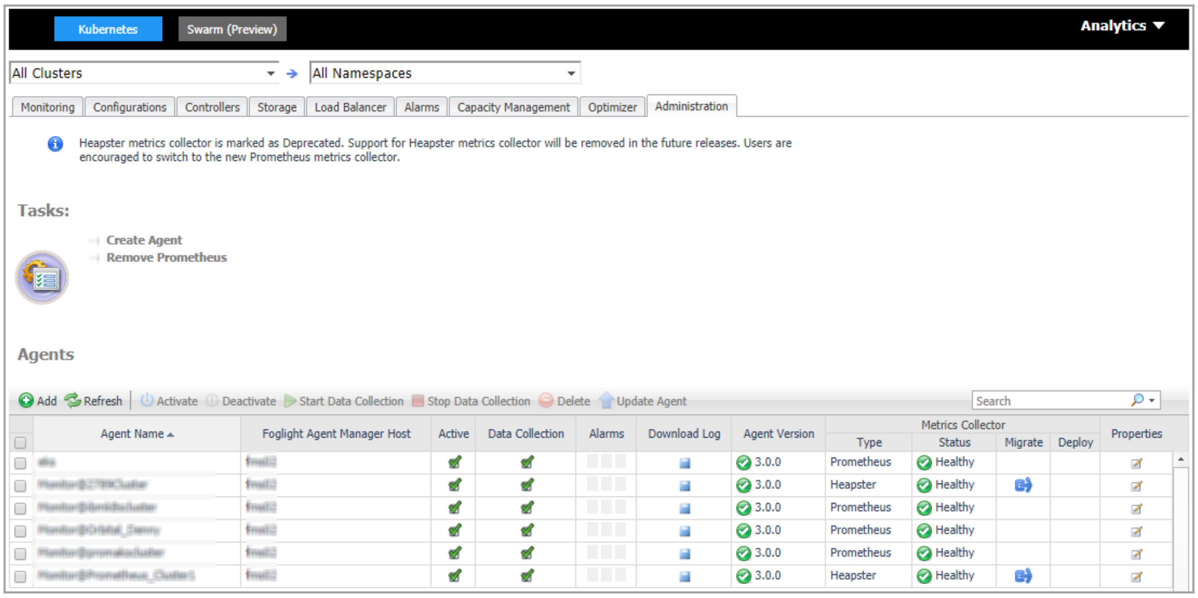
Name	Min vCPU	Max vCPU	Min Memory	Max Memory	Min Storage	Max Storage
Use the above Add icon to add one or more Credential Groups.						

This tab is available in VMware environment. The **Constraints** tab allows you to set custom thresholds for select objects in the environment. These recommendations are displayed in the **Optimizer** tab > **VM Configuration/ CPU/ Memory/ Storage** views > **Modify Recommendation** column. Use this tab to add, edit, and remove constraints groups.

IMPORTANT: A virtual machine may have several partitions. VM environment makes recommendations for each partition separately, but the custom constraints can be set only for the entire VM (not for individual partitions). Therefore, the custom constraint for storage are applied to all partitions on the selected VM.

Administration

Figure 27. Kubernetes Administration Dashboard



Heapster metrics collector is marked as Deprecated. Support for Heapster metrics collector will be removed in the future releases. Users are encouraged to switch to the new Prometheus metrics collector.

Tasks:

Create Agent
Remove Prometheus

Agents

	Agent Name	Foglight Agent Manager Host	Active	Data Collection	Alarms	Download Log	Agent Version	Metrics Collector				Properties
								Type	Status	Migrate	Deploy	
	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
	[redacted]	[redacted]	✓	✓			3.0.0	Heapster	Healthy			
	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
	[redacted]	[redacted]	✓	✓			3.0.0	Prometheus	Healthy			
	[redacted]	[redacted]	✓	✓			3.0.0	Heapster	Healthy			

NOTE: The *Administration* dashboard can be accessed only when the users have both the Administrator role and the Container Administrator role. To grant the users with the Container Administrator role, go to **Administration** > **Users & Security** management under *Administer Server* > **Manage Users, Groups, Roles** > **Roles** tab.

IMPORTANT: Heapster metrics collector is marked as Deprecated. Support for Heapster metrics collector will be removed in the future release. Users are encouraged to switch to the new Prometheus metrics collector.

The *Administration* dashboard supports new agent creation and existing agents management. It contains the following features:

- Create Agent task
- Remove Prometheus task
- Agents table

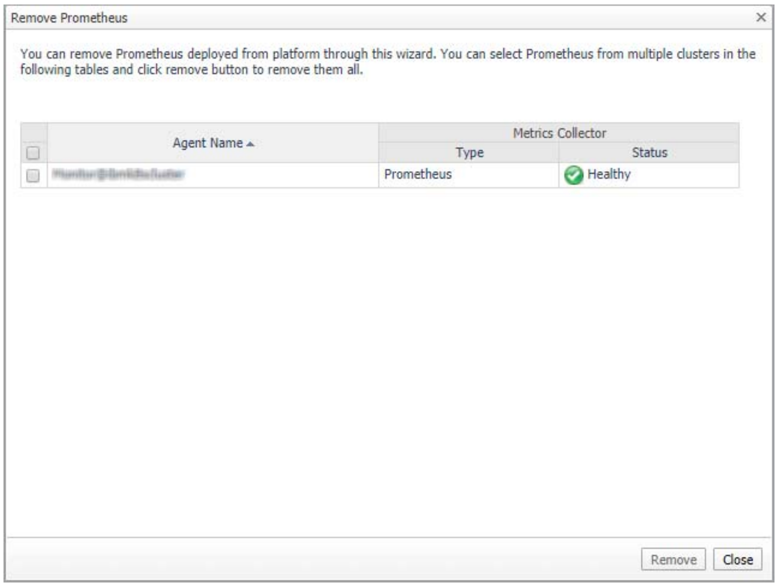
Create Agent task

Use *Create Agent* or the *Add* button to start a wizard to create a new agent. See Creating and Activating a Kubernetes Agent on page 19 for more information.

Remove Prometheus task

Remove Prometheus will start a wizard to remove the Prometheus successfully deployed through our platform.

NOTE: Prometheus that are not deployed through our platform are not listed here.


Remove Prometheus

You can remove Prometheus deployed from platform through this wizard. You can select Prometheus from multiple clusters in the following tables and click remove button to remove them all.

	Agent Name ▲	Metrics Collector	
		Type	Status
☐	[redacted]	Prometheus	✓ Healthy

Select the Prometheus you want to remove and click **Remove**.

> **NOTE:** All the related Kubernetes components will also be removed according to the template that you used to deploy the Prometheus through our platform.

If the Prometheus service is deleted successfully, a result message will be display. Otherwise, the *Remove Prometheus* wizard will appear again with error messages to guide you with further operations. Follow the messages and delete them again later.

Agents table

Use *Activate*, *Deactivate*, *Start Data Collection*, *Stop Data Collection, Delete*, and *Update Agent* to manage the agent.

Metrics Collector

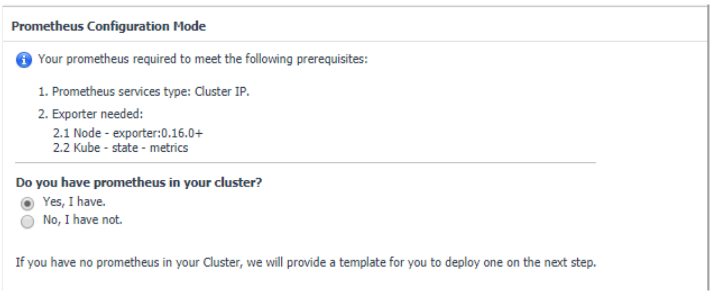
- *Type*: the type of Metrics Collector, supports Heapster or Prometheus.
- *Status*:
  - Healthy: discovered the Kubernetes service of the Metrics Collector and the result for the health check of the service is successful.
  - Unhealthy: discovered the Kubernetes service of the Metrics Collector, however, the result for the health check of the service is failed.
  - Discover failed: failed to discover the Kubernetes service in your cluster. The *Deploy* icon is enabled.
  - Waiting for data update: waiting for status update after performing some operations for Metrics Collector.
  - Failed to update status: connection failed or other known issues. Contact the support.
- *Migrate* : start a wizard to migrate Heapster to Prometheus Metrics Collector.

  If the agent version is 3.0.0 and is using Heapster as the metric collector, you can Migrate from Heapster to Prometheus metrics collector.
- *Deploy* : start a wizard to configure or guide to deploy Prometheus to your cluster.

  If our platform failed to discover Prometheus in your cluster and the *Deploy* icon is enabled.

Migrate or Deploy process

Clicking *Migrate* or *Deploy* will launch the *Configure Prometheus Metrics Collector* Wizard.


Prometheus Configuration Mode

Your prometheus required to meet the following prerequisites:

1. Prometheus services type: Cluster IP.
2. Exporter needed:
   2.1 Node - exporter:0.16.0+
   2.2 Kube - state - metrics

Do you have prometheus in your cluster?

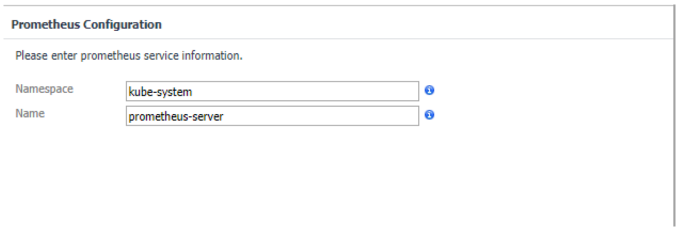
[✓ Yes, I have.]
No, I have not.

If you have no prometheus in your Cluster, we will provide a template for you to deploy one on the next step.

- Have existing Prometheus in your cluster.

  Enter the existing Prometheus service namespace and name to configure the metrics collector.

  NOTE: Ensure that you have Prometheus in your cluster before this step. We will check the service existence and health status after you click Next.


Prometheus Configuration

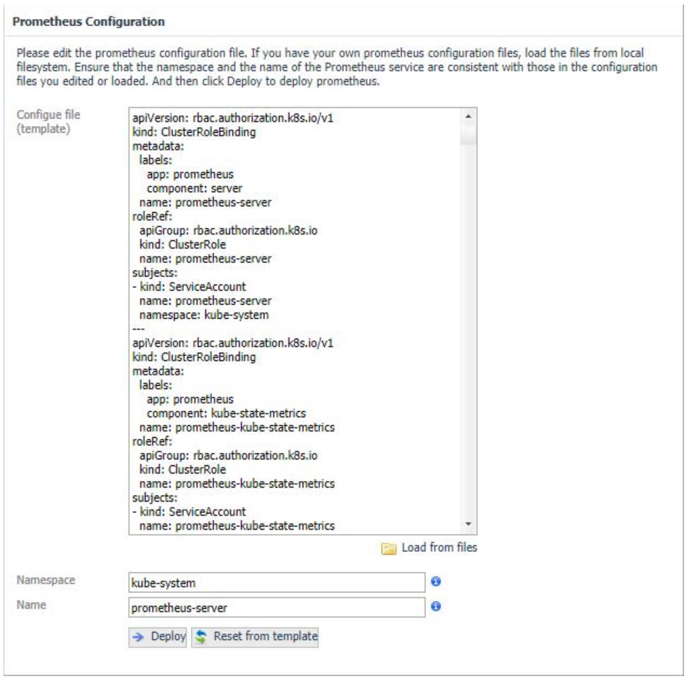
Please enter prometheus service information.

Namespace	kube-system
Name	prometheus-server

- Do not have existing Prometheus in your cluster.

  A *Prometheus Configuration* wizard will appear. To configure the metrics collector, either use the default Prometheus template or upload your own Prometheus deployment .yml files through *Load from files*.

  NOTE: Whenever you change the content in Configure file (template), ensure that the *Namespace* and *Name* fields are consistent with the Prometheus service configurations in your Configure file (template). If you are using the default template, there is need to change anything.


Prometheus Configuration

Please edit the prometheus configuration file. If you have your own prometheus configuration files, load the files from local filesystem. Ensure that the namespace and the name of the Prometheus service are consistent with those in the configuration files you edited or loaded. And then click Deploy to deploy prometheus.

Configue file (template)

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    app: prometheus
    component: server
  name: prometheus-server
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus-server
subjects:
- kind: ServiceAccount
  name: prometheus-server
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    app: prometheus
    component: kube-state-metrics
  name: prometheus-kube-state-metrics
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus-kube-state-metrics
subjects:
- kind: ServiceAccount
  name: prometheus-kube-state-metrics
```

Namespace	kube-system
Name	prometheus-server

Reset from template: Helps you to reset the Configure file (template) content to the default template.

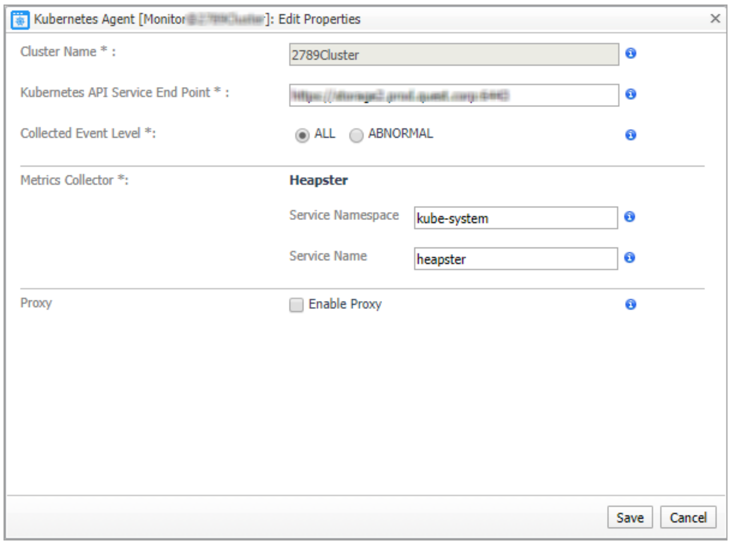
Deploy: Creates Prometheus components to your cluster with the Configure file.
After clicking *Deploy*, a progress message will be displayed.

- If deployed successfully, a succeeded message will be displayed. Close the *Validation Result* page and then click **Next** to finish your agent creation process.
- If failed to deploy the Prometheus configuration, a *Validation Result* page will be displayed with possible solutions.

Agent Edit properties

Click in Properties column of the Agents table to edit the property of the agent.

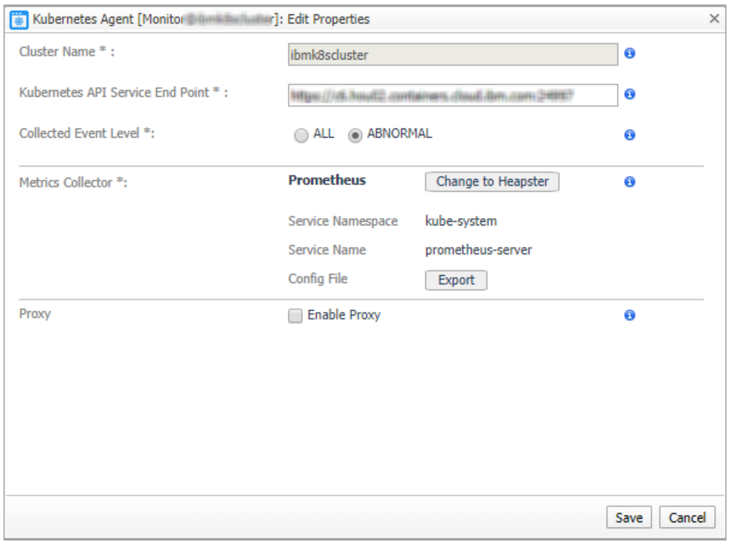
- For agent using Heapster as metric collector, a wizard similar as below will be displayed. Update the agent properties and save the changes. A new data collection process will be initiated.



Kubernetes Agent [Monitor[redacted]]: Edit Properties

Cluster Name * :	2789Cluster
Kubernetes API Service End Point * :	[redacted]
Collected Event Level *:	[✓ ALL] ABNORMAL
Metrics Collector *:	**Heapster**
Service Namespace	kube-system
Service Name	heapster
Proxy	☐ Enable Proxy

- For agent using Prometheus as metric collector and the Prometheus is successfully deployed through our platform, a wizard similar as below will be displayed.



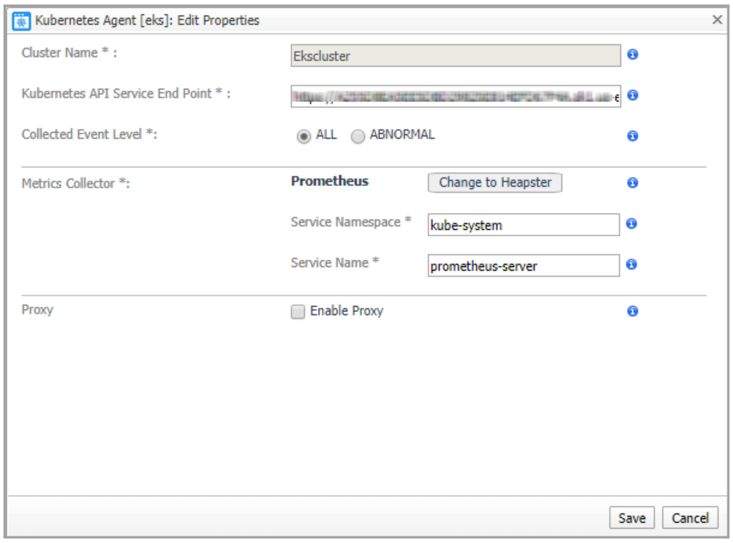
Kubernetes Agent [Monitor[redacted]]: Edit Properties

Cluster Name * :	ibmk8scluster
Kubernetes API Service End Point * :	[redacted]
Collected Event Level *:	ALL [✓ ABNORMAL]
Metrics Collector *:	**Prometheus**
Service Namespace	kube-system
Service Name	prometheus-server
Config File	Export
Proxy	☐ Enable Proxy

NOTE: If the Prometheus is deployed through our platform, the Service Namespace and Service Name cannot be updated through *Kubernetes Agent Edit Properties wizard.* You can use *Remove Prometheus* to remove Prometheus and deploy a new one.

Export: export the deployed .yml file of the Prometheus.
Change to Heapster: change from Prometheus Metrics Collector to Heapster. However, it is not recommended.

- For agent using existing Prometheus as Metrics Collector, a wizard similar as below will be displayed.



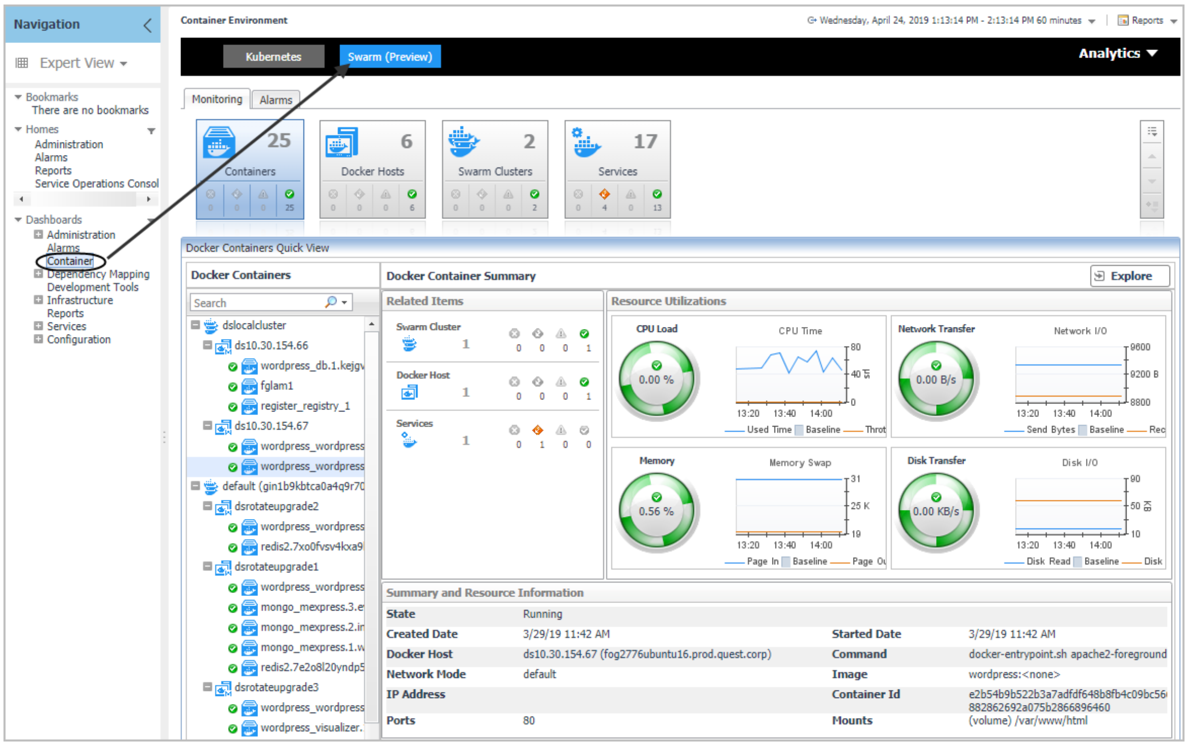
Cluster Name * : Ekscluster
Kubernetes API Service End Point * : [illegible]
Collected Event Level *: [✓ ALL] ABNORMAL
Metrics Collector *: Prometheus
Service Namespace * kube-system
Service Name * prometheus-server
Proxy: Enable Proxy

NOTE: If you are using your existing Prometheus as Metrics Collector, we will not help you to manage your Prometheus. Both the Service Namespace and Service Name can be updated in *Kubernetes Agent Edit Properties* wizard.

Docker Swarm

The *Docker Container Quick View* appears after clicking **Monitoring > Containers**. Click **Swarm (Preview)** from the header on top to switch to Docker Swarm dashboard.

Figure 28. Docker Swarm Dashboard



Docker Containers

- dslocalcluster
  - ds10.30.154.66
    - wordpress_db.1.kejgv
    - fglam1
    - register_registry_1
  - ds10.30.154.67
    - wordpress_wordpress
    - wordpress_wordpress
- default (gin1b9kbtca0a4q9r70
  - dsrotateupgrade2
    - wordpress_wordpress
    - redis2.7xo0fvsv4kxa9
  - dsrotateupgrade1
    - wordpress_wordpress
    - mongo_mexpress.3.e
    - mongo_mexpress.2.in
    - mongo_mexpress.1.w
    - redis2.7e2o8l20yndp5
  - dsrotateupgrade3
    - wordpress_wordpress
    - wordpress_visualizer.

Summary and Resource Information

State	Running		
Created Date	3/29/19 11:42 AM	**Started Date**	3/29/19 11:42 AM
Docker Host	ds10.30.154.67 (fog2776ubuntu16.prod.quest.corp)	**Command**	docker-entrypoint.sh apache2-foreground
Network Mode	default	**Image**	wordpress:<none>
IP Address		**Container Id**	e2b54b9b522b3a7adfdf648b8fb4c09bc56 882862692a075b2866896460
Ports	80	**Mounts**	(volume) /var/www/html

Monitoring Docker Containers

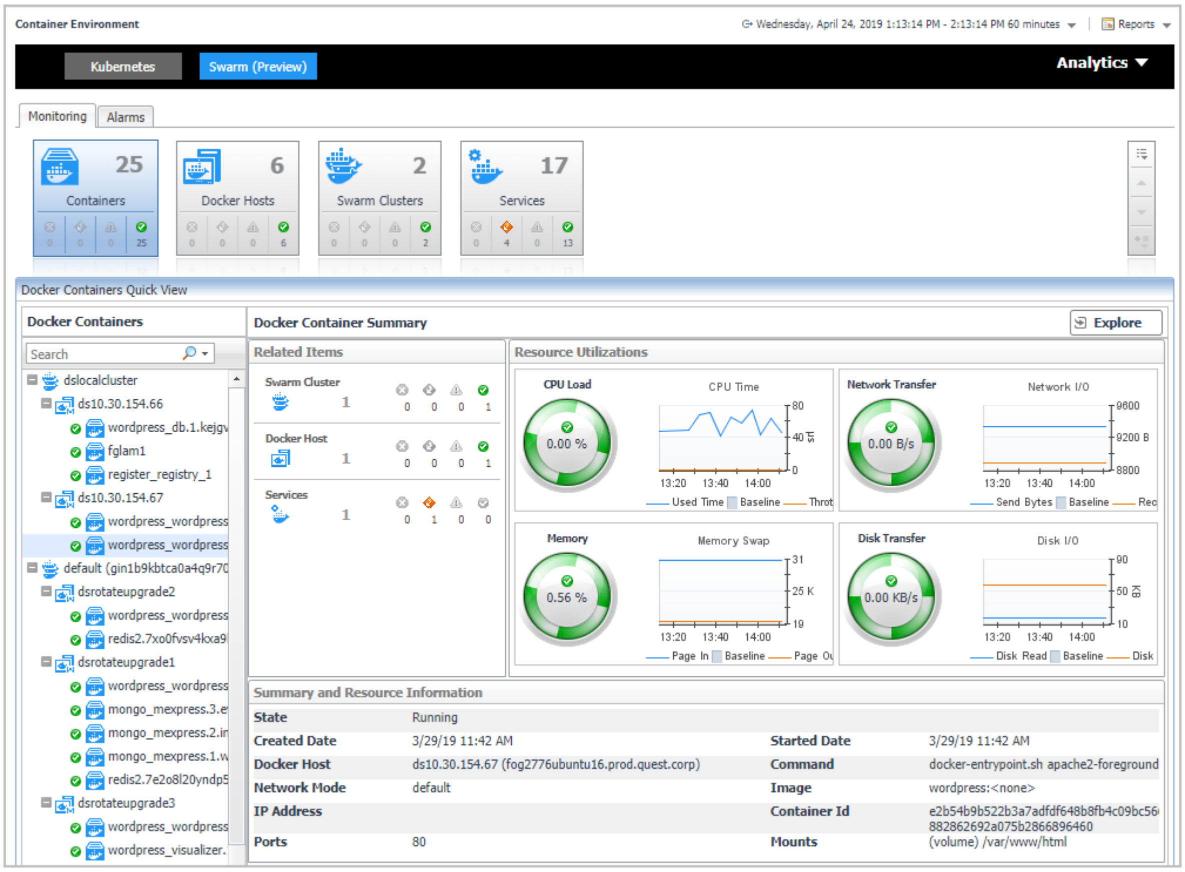
This view consists of the following two panes:

- The **Docker Containers** tree view, which appears on the left of *Docker Containers Quick View*, lists the containers existing in the monitored *Docker* environment. The containers in the tree view are grouped by **cluster > docker host > container**.
- The Docker Container Summary view, which appears on the right after you select an individual container in the **Docker Containers** tree view.

Docker Container Summary view

The **Docker Container Summary** view appears on the right when you select a container in the **Docker Containers** tree view.

Figure 29. Docker Container Summary view



Summary and Resource Information

State	Running		
Created Date	3/29/19 11:42 AM	Started Date	3/29/19 11:42 AM
Docker Host	ds10.30.154.67 (fog2776ubuntu16.prod.quest.corp)	Command	docker-entrypoint.sh apache2-foreground
Network Mode	default	Image	wordpress:<none>
IP Address		Container Id	e2b54b9b522b3a7adfdf648b8fb4c09bc56 882862692a075b2866896460
Ports	80	Mounts	(volume) /var/www/html

The **Docker Container Summary** view displays the following data:

- *Related Items: Shows the related Docker components grouped by type as well as the associated alarms.*
- *Resource Utilizations*: The resource utilization for the selected Docker Container over a selected period of time, which includes the following:
  - *CPU Load*: Shows the CPU utilization of the selected container.
  - CPU Time: Shows the used time and throttled time of the selected container.
  - *Network Transfer*: Shows the transfer bytes rate of the selected container over a selected period of time.
  - Network I/O: Shows the total send/receive bytes of the selected container.
  - *Memory*: Shows the memory utilization of the selected container.
  - Memory Swap: Shows the mounts of memory pages that are swapped to disk.
  - *Disk Transfer*: Shows the disk transfer bytes rate of the selected container over a selected period of time.
  - Disk I/O: Shows the disk read/write bytes of the selected container.
- *Summary and Resource Information*: Displays the detailed information about the selected Container, including *State, Command, Created Time, Started Time, Image,* and *so on*.

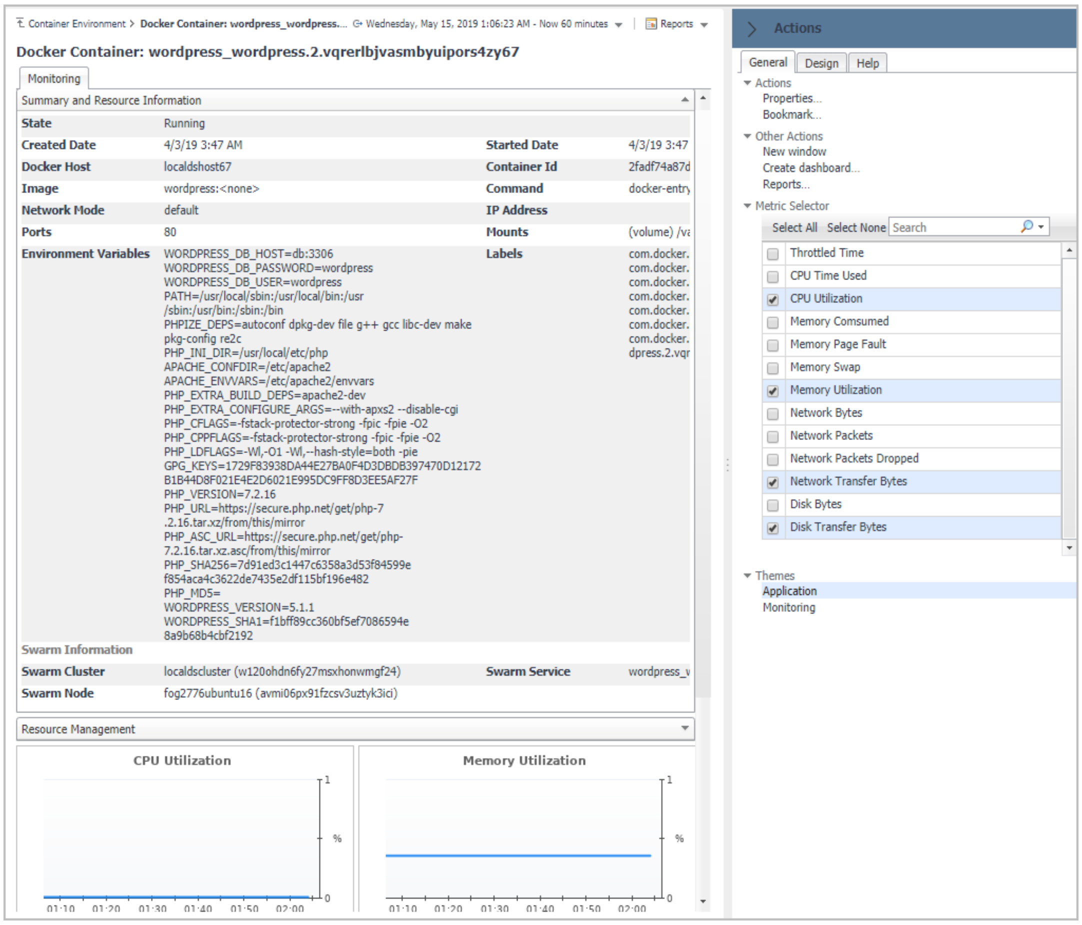
Click **Explore** on the upper right of the **Docker Container Summary** view to open the Container Explorer view, which shows more detailed information about this container.

Container Explorer view

The *Container Explorer* view opens when you click **Explore** in the Docker Container Summary view, which includes the following tabs:

> *Monitoring tab:* The *Monitoring* tab displays the overall information of the selected container over a selected period of time, including the *Summary and Resource Information* table, Resource Management table as well as the Metrics list. To set the Metrics list displayed, go to **Action** > **General** > **Metric Selector**. For more information, see Container metrics on page 68.

Figure 30. Docker Container Explorer view Monitoring Tab



Container Environment > Docker Container: wordpress_wordpress.... Wednesday, May 15, 2019 1:06:23 AM - Now 60 minutes | Reports

Docker Container: wordpress_wordpress.2.vqrerlbjvasmbyuipors4zy67

Monitoring

Summary and Resource Information

State	Running		
Created Date	4/3/19 3:47 AM	**Started Date**	4/3/19 3:47
Docker Host	localdshost67	**Container Id**	2fadf74a87d
Image	wordpress:<none>	**Command**	docker-entry
Network Mode	default	**IP Address**	
Ports	80	**Mounts**	(volume) /va

Environment Variables	WORDPRESS_DB_HOST=db:3306
	WORDPRESS_DB_PASSWORD=wordpress
	WORDPRESS_DB_USER=wordpress
	PATH=/usr/local/sbin:/usr/local/bin:/usr
	/sbin:/usr/bin:/sbin:/bin
	PHPIZE_DEPS=autoconf dpkg-dev file g++ gcc libc-dev make
	pkg-config re2c
	PHP_INI_DIR=/usr/local/etc/php
	APACHE_CONFDIR=/etc/apache2
	APACHE_ENVVARS=/etc/apache2/envvars
	PHP_EXTRA_BUILD_DEPS=apache2-dev
	PHP_EXTRA_CONFIGURE_ARGS=--with-apxs2 --disable-cgi
	PHP_CFLAGS=-fstack-protector-strong -fpic -fpie -O2
	PHP_CPPFLAGS=-fstack-protector-strong -fpic -fpie -O2
	PHP_LDFLAGS=-Wl,-O1 -Wl,--hash-style=both -pie
	GPG_KEYS=1729F83938DA44E27BA0F4D3DBDB397470D12172
	B1B44D8F021E4E2D6021E995DC9FF8D3EE5AF27F
	PHP_VERSION=7.2.16
	PHP_URL=https://secure.php.net/get/php-7
	.2.16.tar.xz/from/this/mirror
	PHP_ASC_URL=https://secure.php.net/get/php-
	7.2.16.tar.xz.asc/from/this/mirror
	PHP_SHA256=7d91ed3c1447c6358a3d53f84599e
	f854aca4c3622de7435e2df115bf196e482
	PHP_MD5=
	WORDPRESS_VERSION=5.1.1
	WORDPRESS_SHA1=f1bff89cc360bf5ef7086594e
	8a9b68b4cbf2192
		Labels	com.docker.
			com.docker.
			com.docker.
			com.docker.
			com.docker.
			com.docker.
			com.docker.
			dpress.2.vqr

Swarm Information

Swarm Cluster	localdscluster (w120ohdn6fy27msxhonwmgf24)	**Swarm Service**	wordpress_v
Swarm Node	fog2776ubuntu16 (avmi06px91fzcsv3uztyk3ici)		

Resource Management

Actions

General | Design | Help

- Actions
  - Properties...
  - Bookmark...
- Other Actions
  - New window
  - Create dashboard...
  - Reports...
- Metric Selector
  - Select All | Select None | Search
  - ☐ Throttled Time
  - ☐ CPU Time Used
  - ✓ CPU Utilization
  - ☐ Memory Comsumed
  - ☐ Memory Page Fault
  - ☐ Memory Swap
  - ✓ Memory Utilization
  - ☐ Network Bytes
  - ☐ Network Packets
  - ☐ Network Packets Dropped
  - ✓ Network Transfer Bytes
  - ☐ Disk Bytes
  - ✓ Disk Transfer Bytes
- Themes
  - Application
  - Monitoring

Monitoring Docker Hosts

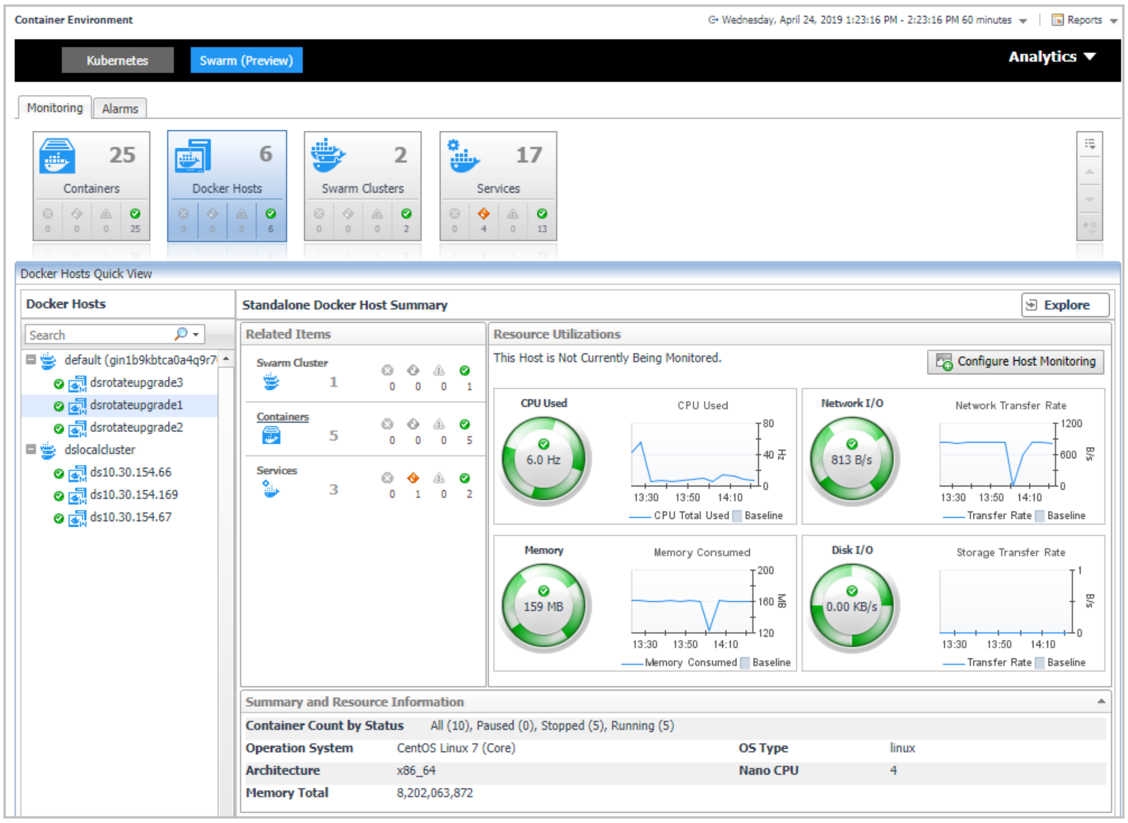
This view consists of the following two panes:

- The **Docker Hosts** tree view, which appears on the left of *Docker Hosts Quick View*, lists the docker hosts existing in the monitored *Docker* environment. The docker hosts in the tree view are grouped by **cluster > docker host**.
- The Docker Host Summary view appears on the right after you select an individual docker host in the **Docker Hosts** tree view.

Docker Host Summary view

The **Docker Host Summary** view appears on the right when you select a docker host in the **Docker Hosts** tree view.

Figure 31. Docker Host Summary view



Summary and Resource Information

Container Count by Status	All (10), Paused (0), Stopped (5), Running (5)		
Operation System	CentOS Linux 7 (Core)	**OS Type**	linux
Architecture	x86_64	**Nano CPU**	4
Memory Total	8,202,063,872		

The **Docker Host Summary** view displays the following data:

- *Related Items: Shows the related Docker components grouped by type as well as the associated alarms.*
- *Resource Utilizations*: The resource utilization for the selected docker host over a selected period of time, which includes the following:
  - *CPU Load*: Shows the CPU utilization of the selected docker host.
  - *CPU Used*: Shows the used CPU resources aggregated from the containers running on the docker host.
  - *Network I/O* and Network Transfer Rate: Shows the transfer bytes rate of the selected docker host aggregated from the containers running on the docker host over a selected period of time.
  - *Memory* and Memory Consumed: Shows the memory consumed bytes aggregated from the containers running on the docker host.
  - *Disk I/O* and Disk Transfer: Shows the disk transfer bytes rate of the selected docker host aggregated from the containers running on the docker host over a selected period of time.
- *Summary and Resource Information*: Displays the detailed information about the selected docker host, including *Container Count by Status, Operating System, Memory Total,* and *so on*.

Click **Explore** on the upper right of the **Docker Host Summary** view to open the Docker Host Explorer view, which shows more detailed information about this container.

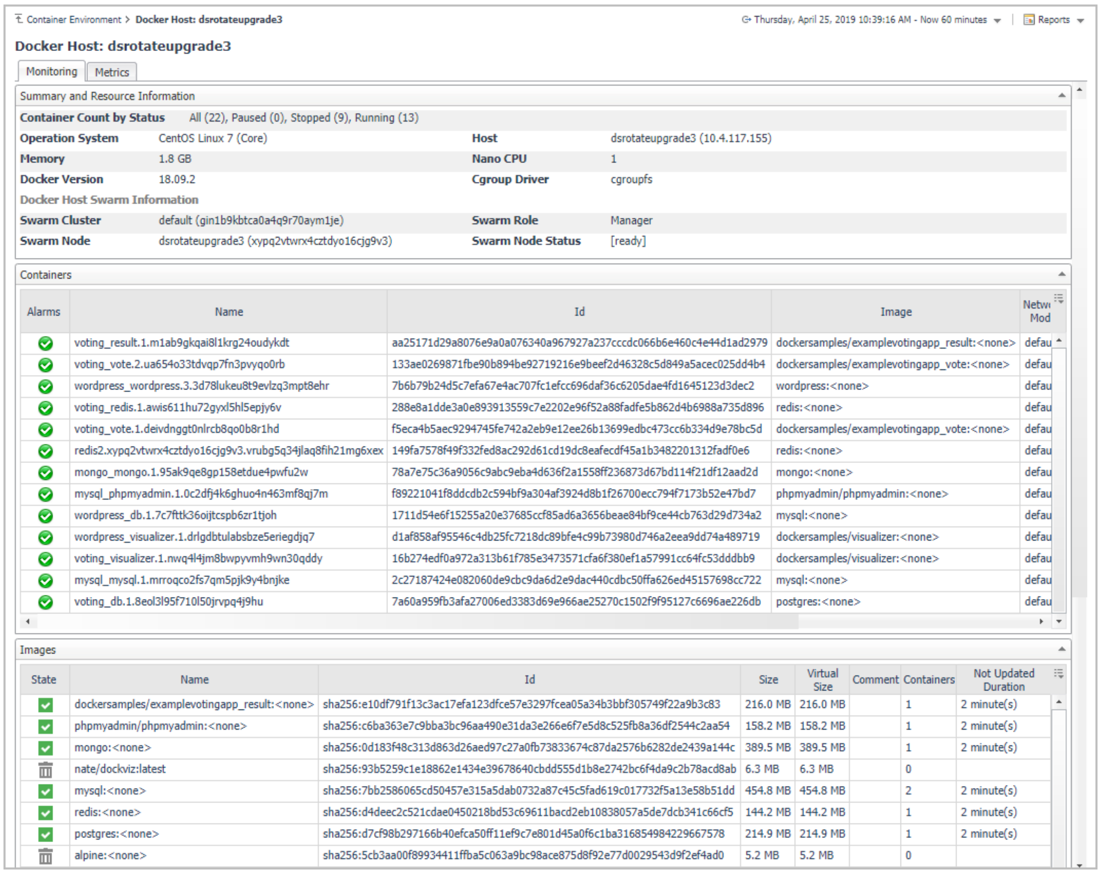
Docker Host Explorer view

The *Docker Host Explorer* view opens when you click **Explore** in the Docker Host Summary view, which includes the following tabs:

- *Monitoring* tab*:* The *Monitoring* tab displays the overall information of the selected docker host over a selected period of time, including the *Summary and Resource Information* table, *Containers* table, *Images* table, and *Volumes* table.

  NOTE: All the docker host metrics are calculated from the aggregated metrics of the containing containers on the docker host.

Figure 32. Docker Host Explorer view Monitoring Tab



Container Environment > **Docker Host: dsrotateupgrade3**   Thursday, April 25, 2019 10:39:16 AM - Now 60 minutes   Reports

Docker Host: dsrotateupgrade3

Monitoring | Metrics

Summary and Resource Information

Container Count by Status	All (22), Paused (0), Stopped (9), Running (13)		
Operation System	CentOS Linux 7 (Core)	**Host**	dsrotateupgrade3 (10.4.117.155)
Memory	1.8 GB	**Nano CPU**	1
Docker Version	18.09.2	**Cgroup Driver**	cgroupfs
Docker Host Swarm Information			
Swarm Cluster	default (gin1b9kbtca0a4q9r70aym1je)	**Swarm Role**	Manager
Swarm Node	dsrotateupgrade3 (xypq2vtwrx4cztdyo16cjg9v3)	**Swarm Node Status**	[ready]

Containers

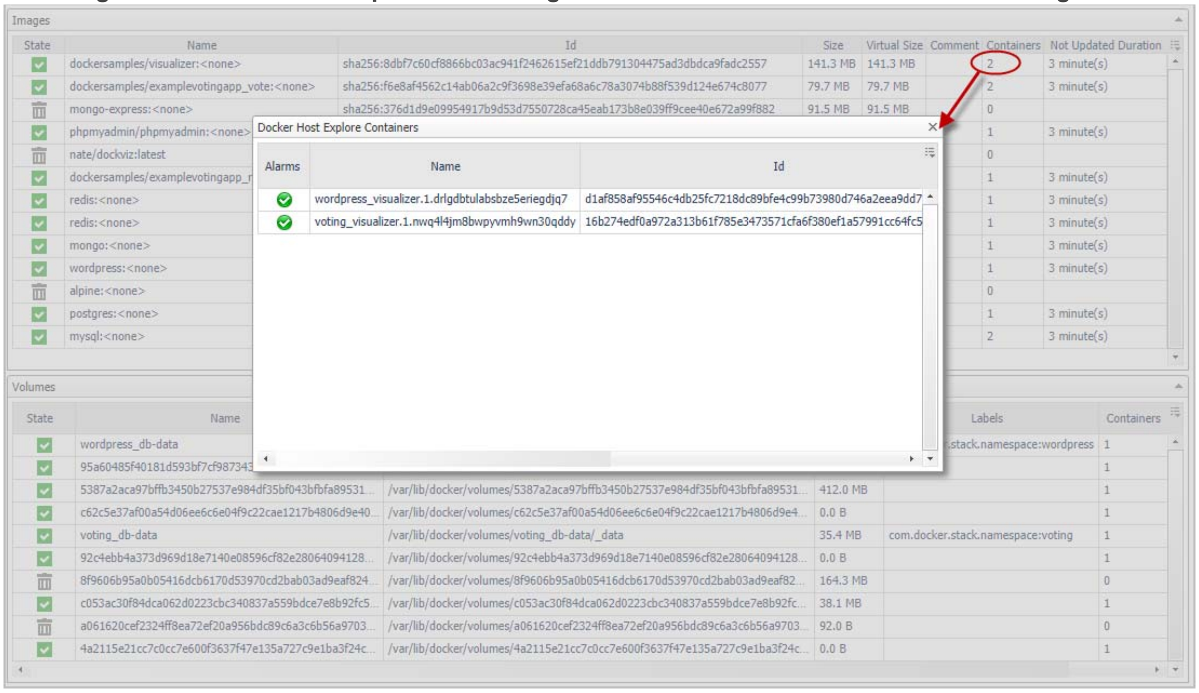
Alarms	Name	Id	Image	Netw Mod
✓	voting_result.1.m1ab9gkqai8l1krg24oudykdt	aa25171d29a8076e9a0a076340a967927a237cccdc066b6e460c4e44d1ad2979	dockersamples/examplevotingapp_result:<none>	defau
✓	voting_vote.2.ua654o33tdvqp7fn3pvyqo0rb	133ae0269871fbe90b894be92719216e9beef2d46328c5d849a5acec025dd4b4	dockersamples/examplevotingapp_vote:<none>	defau
✓	wordpress_wordpress.3.3d78lukeu8t9evlzq3mpt8ehr	7b6b79b24d5c7efa67e4ac707fc1efcc696daf36c6205dae4fd1645123d3dec2	wordpress:<none>	defau
✓	voting_redis.1.awis611hu72gyxl5hl5epjy6v	288e8a1dde3a0e893913559c7e2202e96f52a88fadfe5b862d4b6988a735d896	redis:<none>	defau
✓	voting_vote.1.deivdnggt0nlrcb8qo0b8r1hd	f5eca4b5aec9294745fe742a2eb9e12ee26b13699edbc473cc6b334d9e78bc5d	dockersamples/examplevotingapp_vote:<none>	defau
✓	redis2.xypq2vtwrx4cztdyo16cjg9v3.vrubg5q34jlaq8fih21mg6xex	149fa7578f49f332fed8ac292d61cd19dc8eafecdf45a1b3482201312fadf0e6	redis:<none>	defau
✓	mongo_mongo.1.95ak9qe8gp158etdue4pwfu2w	78a7e75c36a9056c9abc9eba4d636f2a1558ff236873d67bd114f21df12aad2d	mongo:<none>	defau
✓	mysql_phpmyadmin.1.0c2dfj4k6ghuo4n463mf8qj7m	f89221041f8ddcdb2c594bf9a304af3924d8b1f26700ecc794f7173b52e47bd7	phpmyadmin/phpmyadmin:<none>	defau
✓	wordpress_db.1.7c7fttk36oijtcspb6zr1tjoh	1711d54e6f15255a20e37685ccf85ad6a3656beae84bf9ce44cb763d29d734a2	mysql:<none>	defau
✓	wordpress_visualizer.1.drlgdbtulabsbze5eriegdjq7	d1af858af95546c4db25fc7218dc89bfe4c99b73980d746a2eea9dd74a489719	dockersamples/visualizer:<none>	defau
✓	voting_visualizer.1.nwq4l4jm8bwpyvmh9wn30qddy	16b274edf0a972a313b61f785e3473571cfa6f380ef1a57991cc64fc53dddbb9	dockersamples/visualizer:<none>	defau
✓	mysql_mysql.1.mrroqco2fs7qm5pjk9y4bnjke	2c27187424e082060de9cbc9da6d2e9dac440cdbc50ffa626ed45157698cc722	mysql:<none>	defau
✓	voting_db.1.8eol3l95f710l50jrvpq4j9hu	7a60a959fb3afa27006ed3383d69e966ae25270c1502f9f95127c6696ae226db	postgres:<none>	defau

Images

State	Name	Id	Size	Virtual Size	Comment	Containers	Not Updated Duration
✓	dockersamples/examplevotingapp_result:<none>	sha256:e10df791f13c3ac17efa123dfce57e3297fcea05a34b3bbf305749f22a9b3c83	216.0 MB	216.0 MB		1	2 minute(s)
✓	phpmyadmin/phpmyadmin:<none>	sha256:c6ba363e7c9bba3bc96aa490e31da3e266e6f7e5d8c525fb8a36df2544c2aa54	158.2 MB	158.2 MB		1	2 minute(s)
✓	mongo:<none>	sha256:0d183f48c313d863d26aed97c27a0fb73833674c87da2576b6282de2439a144c	389.5 MB	389.5 MB		1	2 minute(s)
🗑	nate/dockviz:latest	sha256:93b5259c1e18862e1434e39678640cbdd555d1b8e2742bc6f4da9c2b78acd8ab	6.3 MB	6.3 MB		0	
✓	mysql:<none>	sha256:7bb2586065cd50457e315a5dab0732a87c45c5fad619c017732f5a13e58b51dd	454.8 MB	454.8 MB		2	2 minute(s)
✓	redis:<none>	sha256:d4deec2c521cdae0450218bd53c69611bacd2eb10838057a5de7dcb341c66cf5	144.2 MB	144.2 MB		1	2 minute(s)
✓	postgres:<none>	sha256:d7cf98b297166b40efca50ff11ef9c7e801d45a0f6c1ba316854984229667578	214.9 MB	214.9 MB		1	2 minute(s)
🗑	alpine:<none>	sha256:5cb3aa00f89934411ffba5c063a9bc98ace875d8f92e77d0029543d9f2ef4ad0	5.2 MB	5.2 MB		0	

- *Containers* table: Includes the containers on this docker host.
- *Images* table: Includes the images pulled onto this docker host.
  - ✓: Indicates this image is using by a container.
  - 🗑: Indicates no container is using this image and the image can be recycled.
- *Volumes* table: Includes the volumes created on this docker host.
  - ✓: Indicates this volume is using by a container.
  - 🗑: Indicates no container is using this volume and the volume can be recycled.

Figure 33. Docker Host Explorer view Images table and Volumes table under Monitoring tab



Images

State	Name	Id	Size	Virtual Size	Comment	Containers	Not Updated Duration
✓	dockersamples/visualizer:<none>	sha256:8dbf7c60cf8866bc03ac941f2462615ef21ddb791304475ad3dbdca9fadc2557	141.3 MB	141.3 MB		2	3 minute(s)
✓	dockersamples/examplevotingapp_vote:<none>	sha256:f6e8af4562c14ab06a2c9f3698e39efa68a6c78a3074b88f539d124e674c8077	79.7 MB	79.7 MB		2	3 minute(s)
🗑	mongo-express:<none>	sha256:376d1d9e09954917b9d53d7550728ca45eab173b8e039ff9cee40e672a99f882	91.5 MB	91.5 MB		0	
✓	phpmyadmin/phpmyadmin:<none>					1	3 minute(s)
🗑	nate/dockviz:latest					0	
✓	dockersamples/examplevotingapp_r					1	3 minute(s)
✓	redis:<none>					1	3 minute(s)
✓	redis:<none>					1	3 minute(s)
✓	mongo:<none>					1	3 minute(s)
✓	wordpress:<none>					1	3 minute(s)
🗑	alpine:<none>					0	
✓	postgres:<none>					1	3 minute(s)
✓	mysql:<none>					2	3 minute(s)

Docker Host Explore Containers

Alarms	Name	Id
✓	wordpress_visualizer.1.drlgdbtulabsbze5eriegdjq7	d1af858af95546c4db25fc7218dc89bfe4c99b73980d746a2eea9dd7
✓	voting_visualizer.1.nwq4l4jm8bwpyvmh9wn30qddy	16b274edf0a972a313b61f785e3473571cfa6f380ef1a57991cc64fc5

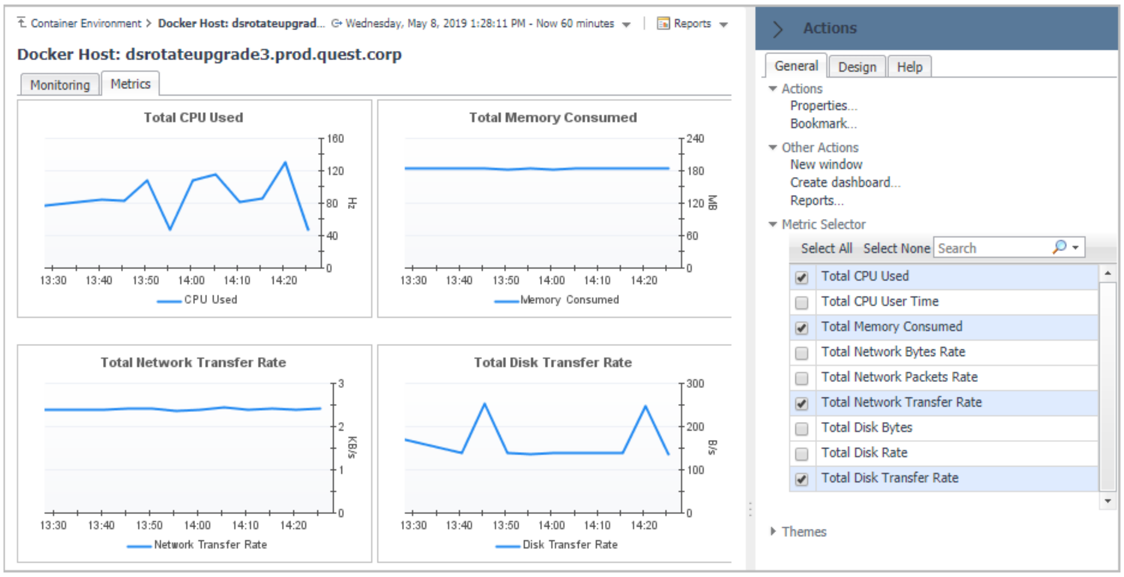
Volumes

State	Name			Labels	Containers
✓	wordpress_db-data			.stack.namespace:wordpress	1
✓	95a60485f40181d593bf7cf987343				1
✓	5387a2aca97bffb3450b27537e984df35bf043bfbfa89531...	/var/lib/docker/volumes/5387a2aca97bffb3450b27537e984df35bf043bfbfa89531...	412.0 MB		1
✓	c62c5e37af00a54d06ee6c6e04f9c22cae1217b4806d9e40...	/var/lib/docker/volumes/c62c5e37af00a54d06ee6c6e04f9c22cae1217b4806d9e4...	0.0 B		1
✓	voting_db-data	/var/lib/docker/volumes/voting_db-data/_data	35.4 MB	com.docker.stack.namespace:voting	1
✓	92c4ebb4a373d969d18e7140e08596cf82e28064094128...	/var/lib/docker/volumes/92c4ebb4a373d969d18e7140e08596cf82e28064094128...	0.0 B		1
🗑	8f9606b95a0b05416dcb6170d53970cd2bab03ad9eaf824...	/var/lib/docker/volumes/8f9606b95a0b05416dcb6170d53970cd2bab03ad9eaf82...	164.3 MB		0
✓	c053ac30f84dca062d0223cbc340837a559bdce7e8b92fc5...	/var/lib/docker/volumes/c053ac30f84dca062d0223cbc340837a559bdce7e8b92fc...	38.1 MB		1
🗑	a061620cef2324ff8ea72ef20a956bdc89c6a3c6b56a9703...	/var/lib/docker/volumes/a061620cef2324ff8ea72ef20a956bdc89c6a3c6b56a9703...	92.0 B		0
✓	4a2115e21cc7c0cc7e600f3637f47e135a727c9e1ba3f24c...	/var/lib/docker/volumes/4a2115e21cc7c0cc7e600f3637f47e135a727c9e1ba3f24c...	0.0 B		1

By clicking the number in the *Containers* column, a *Docker Host Explore Containers* view will open, which lists the containers using this image or this volume. Click the Name or ID of the container and an explore page of the container will appear.

- *Metrics* tab: The *Metrics* tab displays the Metrics list. To set the Metrics list displayed, go to **Action** > **General** > **Metric Selector**. For more information about the description of the metrics, see Container metrics on page 68.

Figure 34. Docker Host Explorer view Metrics Tab



Monitoring Docker Swarm Clusters

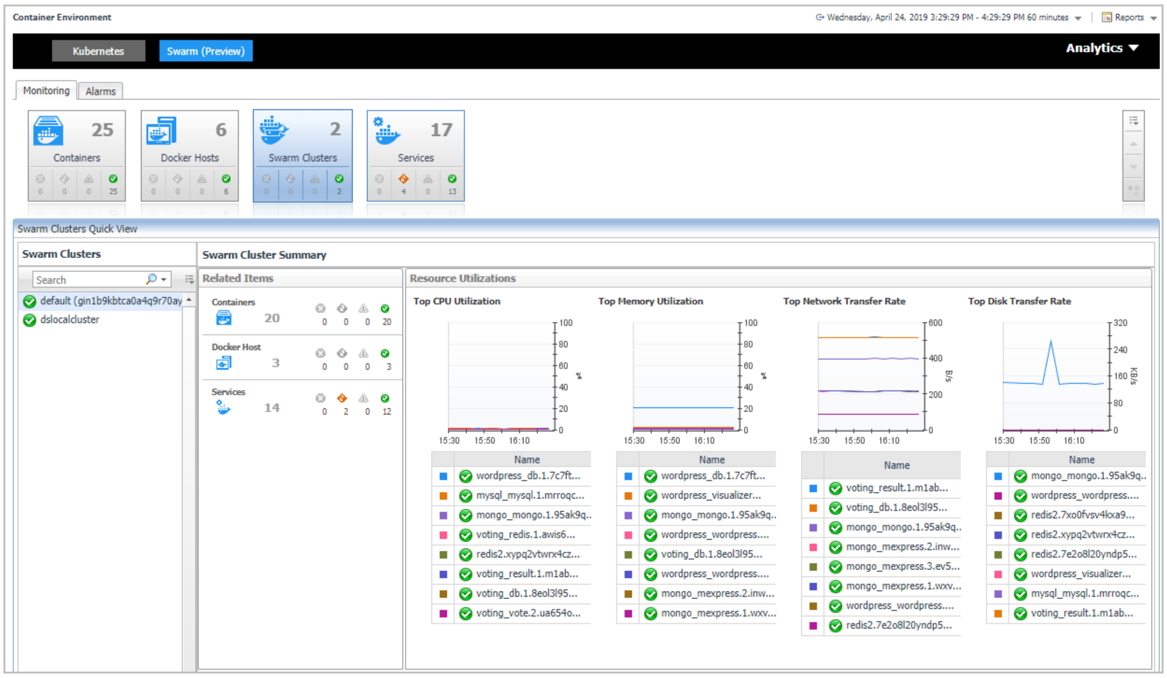
This view consists of the following two panes:

- The **Swarm Clusters** tree view, which appears on the left of *Swarm Clusters Quick View*, lists the docker swarm clusters existing in the monitored *Docker* environment.
- The Docker Swarm Cluster Summary view, which appears on the right after you select an individual docker swarm cluster in the **Swarm Clusters** tree view.

Docker Swarm Cluster Summary view

The **Docker Swarm Cluster Summary** view appears on the right when you select a docker swarm cluster in the **Swarm Clusters** tree view.

Figure 35. Docker Swarm Cluster Summary view



	25 Containers	6 Docker Hosts	2 Swarm Clusters	17 Services
Fatal	0	0	0	0
Critical	0	0	0	4
Warning	0	0	0	0
Normal	25	6	2	13

Related Items

	Total				
Containers	20	0	0	0	20
Docker Host	3	0	0	0	3
Services	14	0	2	0	12

Top CPU Utilization

	Name
🟦	✓ wordpress_db.1.7c7ft...
🟧	✓ mysql_mysql.1.mrroqc...
🟪	✓ mongo_mongo.1.95ak9q..
🟥	✓ voting_redis.1.awis6...
🟩	✓ redis2.xypq2vtwrx4cz...
🟦	✓ voting_result.1.m1ab...
🟫	✓ voting_db.1.8eol3l95...
🟪	✓ voting_vote.2.ua654o...

Top Memory Utilization

	Name
🟦	✓ wordpress_db.1.7c7ft...
🟧	✓ wordpress_visualizer...
🟪	✓ mongo_mongo.1.95ak9q..
🟥	✓ wordpress_wordpress....
🟩	✓ voting_db.1.8eol3l95...
🟦	✓ wordpress_wordpress....
🟫	✓ mongo_mexpress.2.inw...
🟪	✓ mongo_mexpress.1.wxv...

Top Network Transfer Rate

	Name
🟦	✓ voting_result.1.m1ab...
🟧	✓ voting_db.1.8eol3l95...
🟪	✓ mongo_mongo.1.95ak9q..
🟥	✓ mongo_mexpress.2.inw...
🟩	✓ mongo_mexpress.3.ev5...
🟦	✓ mongo_mexpress.1.wxv...
🟫	✓ wordpress_wordpress....
🟪	✓ redis2.7e2o8l20yndp5...

Top Disk Transfer Rate

	Name
🟦	✓ mongo_mongo.1.95ak9q..
🟪	✓ wordpress_wordpress....
🟫	✓ redis2.7xo0fvsv4kxa9...
🟦	✓ redis2.xypq2vtwrx4cz...
🟩	✓ redis2.7e2o8l20yndp5...
🟥	✓ wordpress_visualizer...
🟪	✓ mysql_mysql.1.mrroqc...
🟧	✓ voting_result.1.m1ab...

The **Docker Swarm Cluster Summary** view displays the following data:

- *Related Items*: Shows the related Docker components grouped by type as well as the associated alarms.
- *Resource Utilizations*: Shows CPU Utilization, Memory Utilization, Network Transfer Rate, Disk Transfer Rate metrics of the containers running in this docker swarm cluster in descending order.

Monitoring Docker Swarm Services

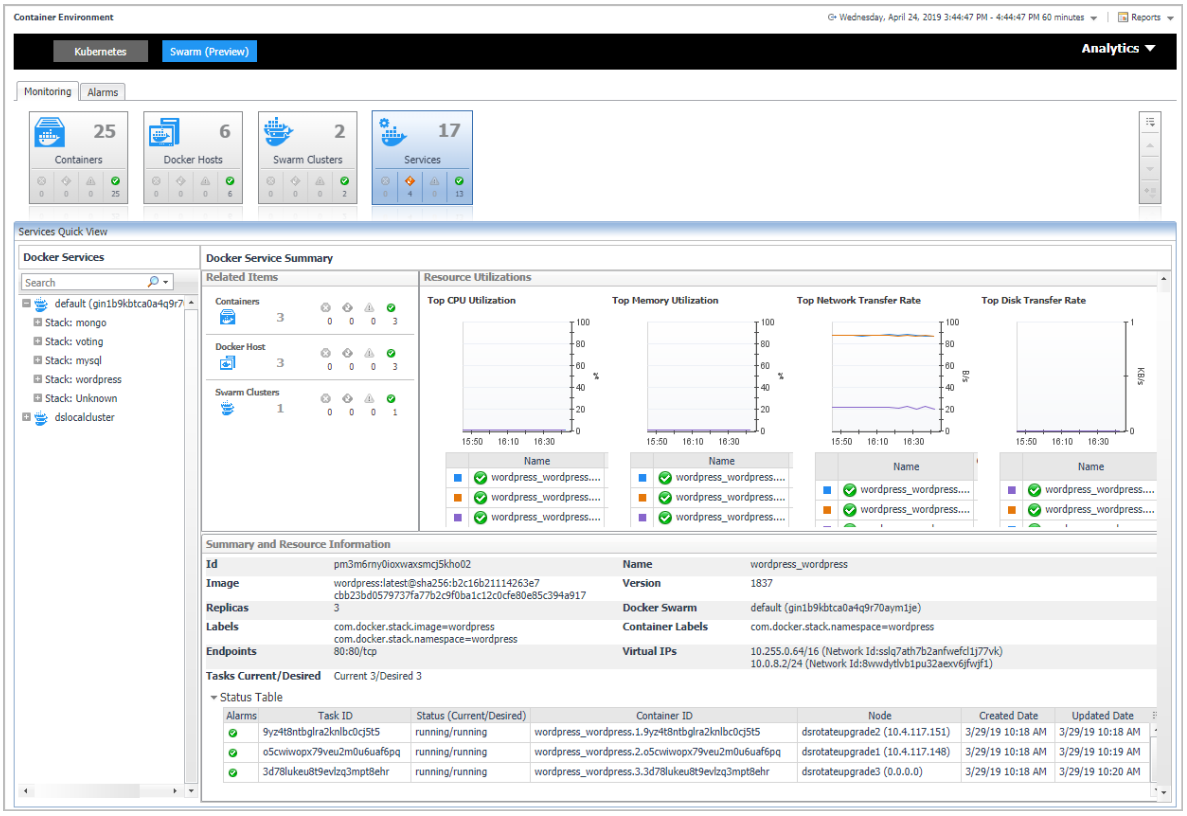
This view consists of the following two panes:

- The **Swarm Services** tree view, which appears on the left of *Swarm Services Quick View*, lists the docker swarm services existing in the monitored *Docker* environment.
- The Docker Swarm Service Summary view, which appears on the right after you select an individual docker swarm service in the **Swarm Services** tree view.

Docker Swarm Service Summary view

The **Docker Service Summary** view appears on the right when you select a docker swarm service in the **Docker Services** tree view.

Figure 36. Docker Service Summary view



Summary and Resource Information

Id	pm3m6rny0ioxwaxsmcj5kho02	**Name**	wordpress_wordpress
Image	wordpress:latest@sha256:b2c16b21114263e7 cbb23bd0579737fa77b2c9f0ba1c12c0cfe80e85c394a917	**Version**	1837
Replicas	3	**Docker Swarm**	default (gin1b9kbtca0a4q9r70aym1je)
Labels	com.docker.stack.image=wordpress com.docker.stack.namespace=wordpress	**Container Labels**	com.docker.stack.namespace=wordpress
Endpoints	80:80/tcp	**Virtual IPs**	10.255.0.64/16 (Network Id:sslq7ath7b2anfwefcl1j77vk) 10.0.8.2/24 (Network Id:8wwdytlvb1pu32aexv6jfwjf1)
Tasks Current/Desired	Current 3/Desired 3		

Status Table

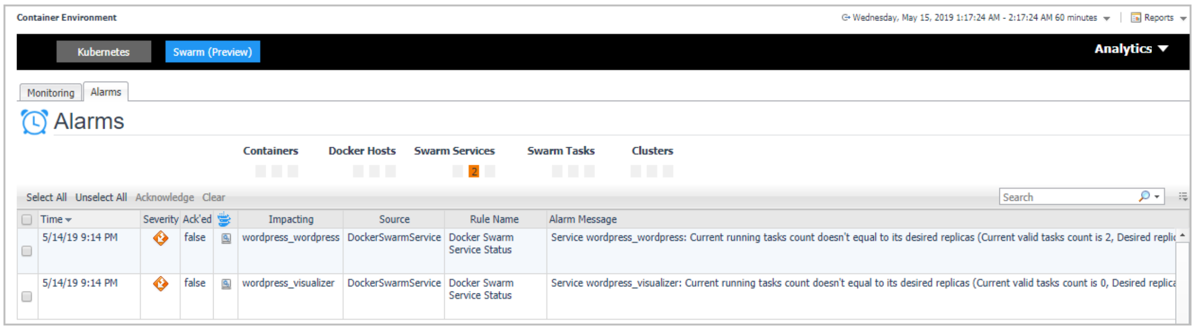
Alarms	Task ID	Status (Current/Desired)	Container ID	Node	Created Date	Updated Date
✓	9yz4t8ntbglra2knlbc0cj5t5	running/running	wordpress_wordpress.1.9yz4t8ntbglra2knlbc0cj5t5	dsrotateupgrade2 (10.4.117.151)	3/29/19 10:18 AM	3/29/19 10:18 AM
✓	o5cwiwopx79veu2m0u6uaf6pq	running/running	wordpress_wordpress.2.o5cwiwopx79veu2m0u6uaf6pq	dsrotateupgrade1 (10.4.117.148)	3/29/19 10:18 AM	3/29/19 10:19 AM
✓	3d78lukeu8t9evlzq3mpt8ehr	running/running	wordpress_wordpress.3.3d78lukeu8t9evlzq3mpt8ehr	dsrotateupgrade3 (0.0.0.0)	3/29/19 10:18 AM	3/29/19 10:20 AM

The **Docker Service Summary** view displays the following data:

- *Related Items*: Shows the related Docker components grouped by type as well as the associated alarms.
- *Resource Utilizations*: Shows CPU Utilization, Memory Utilization, Network Transfer Rate, Disk Transfer Rate metrics of the containers running in this docker swarm service in descending order.
- *Summary and Resource Information*: Shows the summary information of the docker swarm service, including Labels, Image, Mount Volumes, Ports, Container Status and so on.

Alarms

Figure 37. Docker Swarm Alarms Dashboard



Time	Severity	Ack'ed		Impacting	Source	Rule Name	Alarm Message
5/14/19 9:14 PM		false		wordpress_wordpress	DockerSwarmService	Docker Swarm Service Status	Service wordpress_wordpress: Current running tasks count doesn't equal to its desired replicas (Current valid tasks count is 2, Desired replic
5/14/19 9:14 PM		false		wordpress_visualizer	DockerSwarmService	Docker Swarm Service Status	Service wordpress_visualizer: Current running tasks count doesn't equal to its desired replicas (Current valid tasks count is 0, Desired replica

The *Alarms* dashboard displays a list of alarms generated against the monitored Docker environment. Use this view to quickly identify any potential problems related to a specific Docker component.

Analytics

Foglight for Container Management provide analytics feature for Kubernetes and Docker Swarm.

Heat Map is a two-dimensional representation of data in which values are represented by colors. Showing collected metrics with elaborate heat maps allows you to understand complex data sets and the monitored cluster environment well.

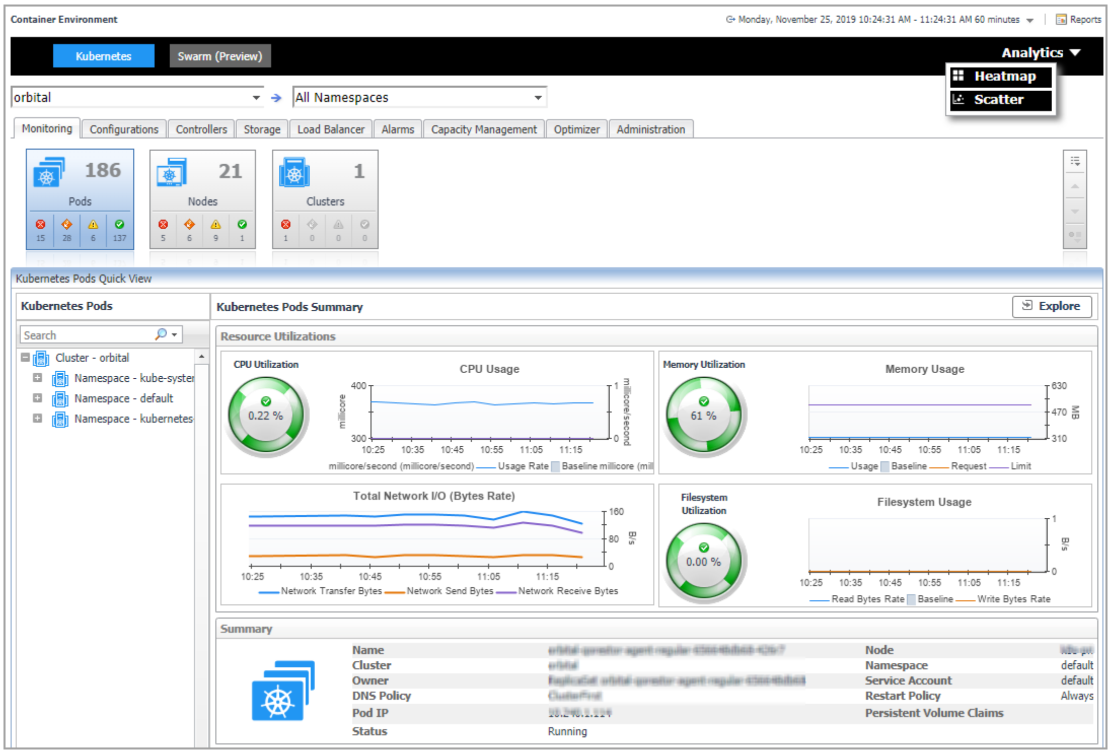
Scatter Plot is used to display values in points using two variables for a set of data. The points is color-coded also, Color Metric can be used to display one additional variable.

- Kubernetes analytics
    - Heatmap analytics
    - Scatter Plot analytics
- Docker Swarm analytics
    - Heatmap analytics
    - Scatter Plot analytics

Kubernetes analytics

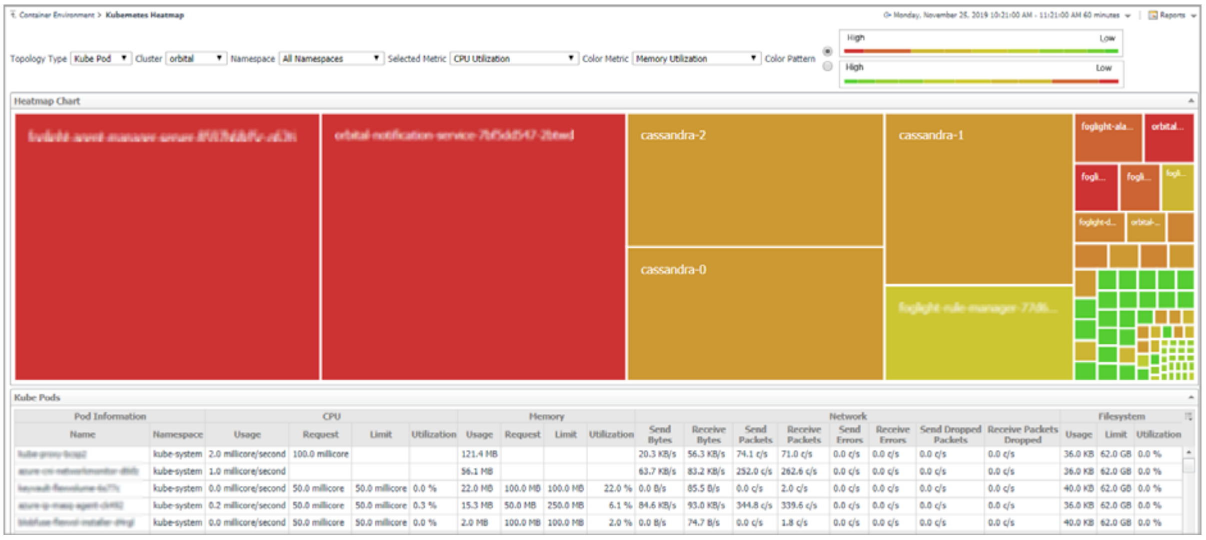
In the Container dashboard, choose **Kubernetes** from the header. Then click **Analytics** from the header, a drop down view will display with **Heatmap** and **Scatter** on it. Click **Heatmap** will navigate to the Kubernetes **Heatmap Analytics** dashboard, while click **Scatter** will navigate to the Kubernetes **Scatter Plot Analytics** dashboard.

Figure 38. Kubernetes analytics Navigation



Heatmap analytics

Figure 39. Kubernetes Heatmap Analytics Dashboard



Container Environment > Kubernetes Heatmap
Monday, November 25, 2019 10:21:00 AM - 11:21:00 AM 60 minutes
Reports
High
Low
High
Low
Heatmap Chart
orbital-notification-service-7bf5dd547-2btwd
cassandra-2
cassandra-1
cassandra-0
foglight-ala...
orbital...
fogli...
fogli...
fogli...
foglight-d...
orbital-...
foglight-rule-manager-7766...
Kube Pods

Topology Type: Kube Pod; Cluster: orbital; Namespace: All Namespaces; Selected Metric: CPU Utilization; Color Metric: Memory Utilization; Color Pattern

Pod Information		CPU				Memory				Network								Filesystem		
Name	Namespace	Usage	Request	Limit	Utilization	Usage	Request	Limit	Utilization	Send Bytes	Receive Bytes	Send Packets	Receive Packets	Send Errors	Receive Errors	Send Dropped Packets	Receive Packets Dropped	Usage	Limit	Utilization
[redacted]	kube-system	2.0 millicore/second	100.0 millicore			121.4 MB				20.3 KB/s	56.3 KB/s	74.1 c/s	71.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	36.0 KB	62.0 GB	0.0 %
[redacted]	kube-system	1.0 millicore/second				56.1 MB				63.7 KB/s	83.2 KB/s	252.0 c/s	262.6 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	36.0 KB	62.0 GB	0.0 %
[redacted]	kube-system	0.0 millicore/second	50.0 millicore	50.0 millicore	0.0 %	22.0 MB	100.0 MB	100.0 MB	22.0 %	0.0 B/s	85.5 B/s	0.0 c/s	2.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	40.0 KB	62.0 GB	0.0 %
[redacted]	kube-system	0.2 millicore/second	50.0 millicore	50.0 millicore	0.3 %	15.3 MB	50.0 MB	250.0 MB	6.1 %	84.6 KB/s	93.0 KB/s	344.8 c/s	339.6 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	36.0 KB	62.0 GB	0.0 %
[redacted]	kube-system	0.0 millicore/second	50.0 millicore	50.0 millicore	0.0 %	2.0 MB	100.0 MB	100.0 MB	2.0 %	0.0 B/s	74.7 B/s	0.0 c/s	1.8 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s	40.0 KB	62.0 GB	0.0 %

Heat maps will be refreshed automatically when you change either of the following fields:

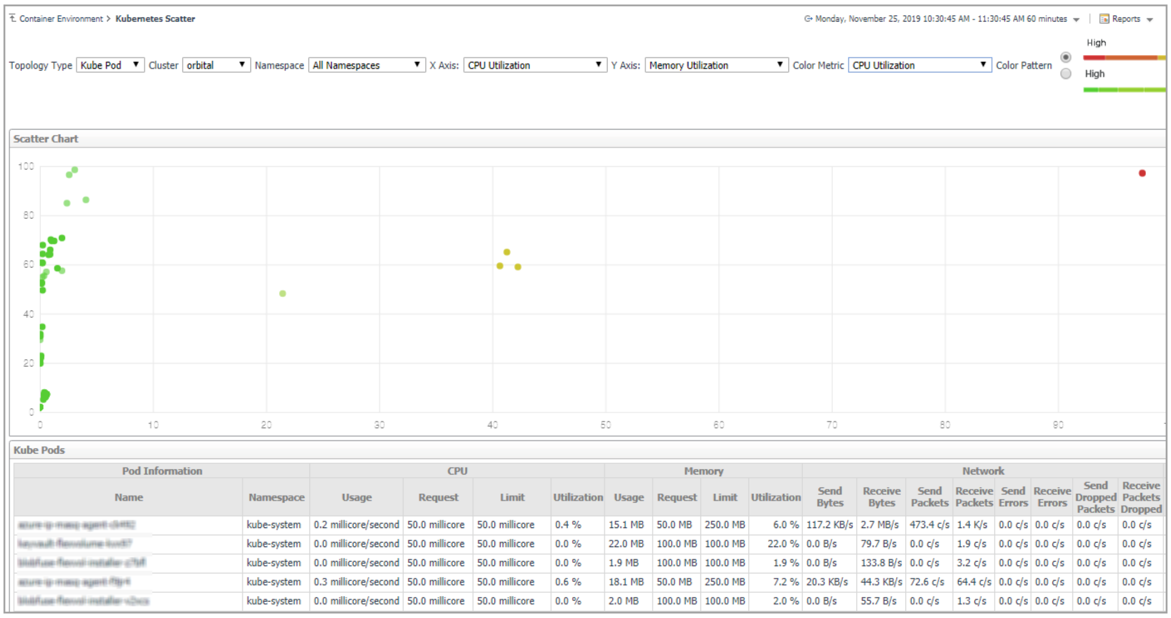
- *Topology Type:* Indicates the monitored topology object, including Kubernetes Pod and Kubernetes Node.
- *Cluster:* Lists all clusters available in the monitored Kubernetes environment.
- *Namespace:* Lists all namespaces available in the monitored Kubernetes environment.
- *Selected Metric:* Populates a rectangle based upon the selected metrics. For example, if you select *Memory Usage* from the *Selected Metric* drop-down list, the rectangle area will be populated based on the used memory for the selected topology object. For more information about metrics, refer to Kubernetes metrics *on page* 65.
- Rendering related metrics: For example, if you select *CPU Usage Rate* and Red to Green, the rectangle of the topology object that has larger value of CPU Usage Rate will be rendered in red.
  - *Color Metric*: Renders the color of rectangle based upon the selected color metric.
  - *Color Pattern*: Offers two patterns, Red to Green (larger value shows in red) or Green to Red (larger value shows in green).

Figure 39 shows an example of heat map. Clicking the object name on the heat map directs you to the relevant object *Explorer* dashboard. For more information, see:

- Pods Explorer view on page 28
- Pod metrics on page 65
- Nodes Explorer view on page 31
- Node metrics on page 67
- Cluster Explorer view on page 35
- Docker Swarm metrics on page 68

Scatter Plot analytics

Figure 40. Kubernetes Scatter Plot Analytics Dashboard


Container Environment > Kubernetes Scatter
Monday, November 25, 2019 10:30:45 AM - 11:30:45 AM 60 minutes
Reports
High
High
Scatter Chart
100
80
60
40
20
0
0
10
20
30
40
50
60
70
80
90
Kube Pods

Topology Type: Kube Pod | Cluster: orbital | Namespace: All Namespaces | X Axis: CPU Utilization | Y Axis: Memory Utilization | Color Metric: CPU Utilization | Color Pattern

Pod Information		CPU				Memory				Network							
Name	Namespace	Usage	Request	Limit	Utilization	Usage	Request	Limit	Utilization	Send Bytes	Receive Bytes	Send Packets	Receive Packets	Send Errors	Receive Errors	Send Dropped Packets	Receive Packets Dropped
[redacted]	kube-system	0.2 millicore/second	50.0 millicore	50.0 millicore	0.4 %	15.1 MB	50.0 MB	250.0 MB	6.0 %	117.2 KB/s	2.7 MB/s	473.4 c/s	1.4 K/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s
[redacted]	kube-system	0.0 millicore/second	50.0 millicore	50.0 millicore	0.0 %	22.0 MB	100.0 MB	100.0 MB	22.0 %	0.0 B/s	79.7 B/s	0.0 c/s	1.9 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s
[redacted]	kube-system	0.0 millicore/second	50.0 millicore	50.0 millicore	0.0 %	1.9 MB	100.0 MB	100.0 MB	1.9 %	0.0 B/s	133.8 B/s	0.0 c/s	3.2 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s
[redacted]	kube-system	0.3 millicore/second	50.0 millicore	50.0 millicore	0.6 %	18.1 MB	50.0 MB	250.0 MB	7.2 %	20.3 KB/s	44.3 KB/s	72.6 c/s	64.4 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s
[redacted]	kube-system	0.0 millicore/second	50.0 millicore	50.0 millicore	0.0 %	2.0 MB	100.0 MB	100.0 MB	2.0 %	0.0 B/s	55.7 B/s	0.0 c/s	1.3 c/s	0.0 c/s	0.0 c/s	0.0 c/s	0.0 c/s

The points on the chart will be refreshed automatically when you change either of the following fields:

- *Topology Type:* Indicates the monitored topology object, including Kubernetes Pod and Kubernetes Node.
- *Cluster:* Lists all clusters available in the monitored Kubernetes environment.
- *Namespace:* Lists all namespaces available in the monitored Kubernetes environment.
- *X Axis:* Indicates which metrics will be plotted on X axis.
- *Y Axis:* Indicates which metrics will be plotted on Y axis.
- Rendering related metrics:
  - Color Metric: Renders the color of circle based upon the selected metrics.
  - *Color Pattern*: Offers two patterns, Red to Green (larger value shows in red) or Green to Red (larger value shows in green).

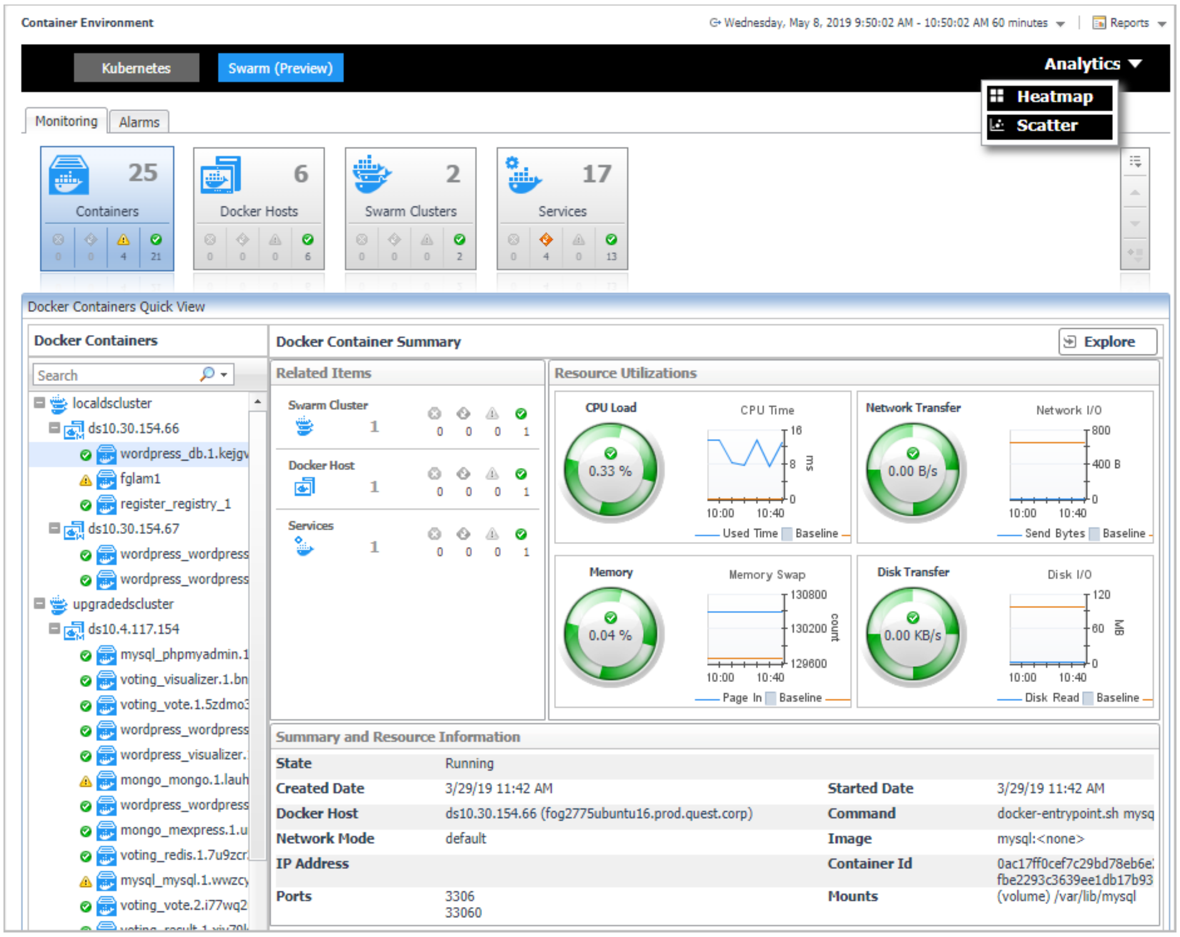
Figure 40 shows an example of Scatter Plot analytics. For more information, see:

- Pods Explorer view on page 28
- Pod metrics on page 65
- Nodes Explorer view on page 31
- Node metrics on page 67
- Cluster Explorer view on page 35
- Docker Swarm metrics on page 68

Docker Swarm analytics

In the Container dashboard, choose **Docker Swarm** from the header. Then click **Analytics** from the header, a drop down view will display with **Heatmap** and **Scatter** on it. Click **Heatmap** will navigate to the Docker Swarm **Heatmap Analytics** dashboard, while click **Scatter** will navigate to the Docker Swarm **Scatter Plot Analytics** dashboard.

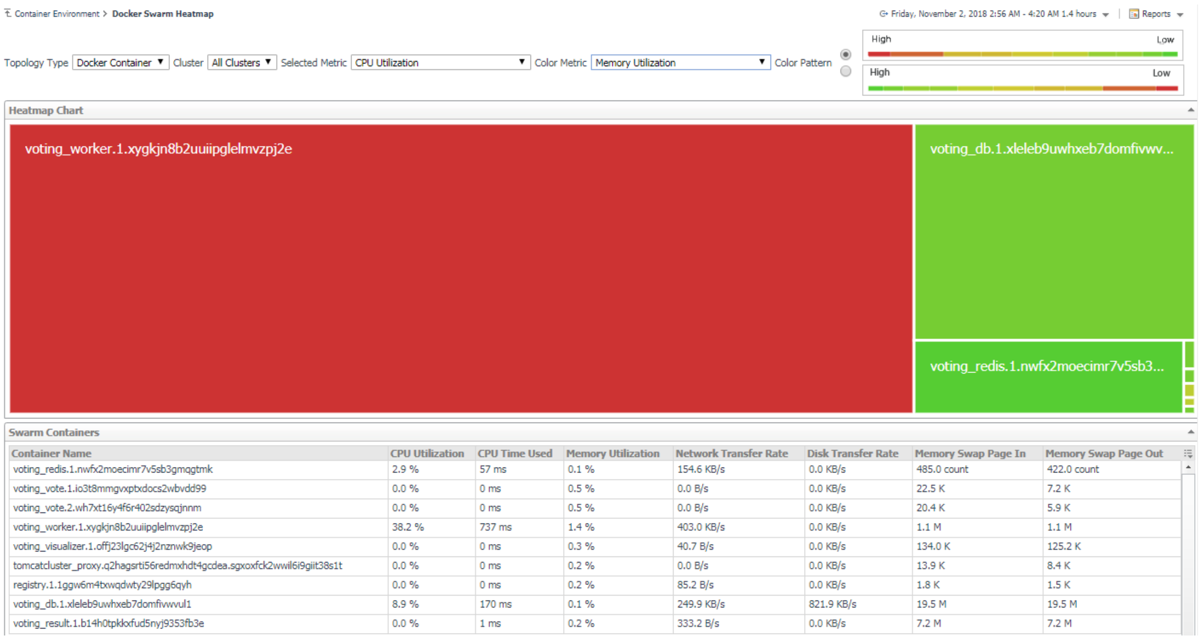
Figure 41. Docker Swarm Analytics Navigation



State	Running		
Created Date	3/29/19 11:42 AM	**Started Date**	3/29/19 11:42 AM
Docker Host	ds10.30.154.66 (fog2775ubuntu16.prod.quest.corp)	**Command**	docker-entrypoint.sh mysq
Network Mode	default	**Image**	mysql:<none>
IP Address		**Container Id**	0ac17ff0cef7c29bd78eb6e fbe2293c3639ee1db17b93
Ports	3306 33060	**Mounts**	(volume) /var/lib/mysql

Heatmap analytics

Figure 42. Docker Swarm Heatmap Analytics Dashboard


Container Environment > Docker Swarm Heatmap
Friday, November 2, 2018 2:56 AM - 4:20 AM 1.4 hours
Reports
High
Low
High
Low
Heatmap Chart
voting_worker.1.xygkjn8b2uuiipglelmvzpj2e
voting_db.1.xleleb9uwhxeb7domfivwv...
voting_redis.1.nwfx2moecimr7v5sb3...
Swarm Containers

Topology Type: Docker Container; Cluster: All Clusters; Selected Metric: CPU Utilization; Color Metric: Memory Utilization; Color Pattern

Container Name	CPU Utilization	CPU Time Used	Memory Utilization	Network Transfer Rate	Disk Transfer Rate	Memory Swap Page In	Memory Swap Page Out
voting_redis.1.nwfx2moecimr7v5sb3gmqgtmk	2.9 %	57 ms	0.1 %	154.6 KB/s	0.0 KB/s	485.0 count	422.0 count
voting_vote.1.io3t8mmgvxptxdocs2wbvdd99	0.0 %	0 ms	0.5 %	0.0 B/s	0.0 KB/s	22.5 K	7.2 K
voting_vote.2.wh7xt16y4f6r402sdzysqjnnm	0.0 %	0 ms	0.5 %	0.0 B/s	0.0 KB/s	20.4 K	5.9 K
voting_worker.1.xygkjn8b2uuiipglelmvzpj2e	38.2 %	737 ms	1.4 %	403.0 KB/s	0.0 KB/s	1.1 M	1.1 M
voting_visualizer.1.offj23lgc62j4j2nznwk9jeop	0.0 %	0 ms	0.3 %	40.7 B/s	0.0 KB/s	134.0 K	125.2 K
tomcatcluster_proxy.q2hagsrti56redmxhdt4gcdea.sgxoxfck2wwil6i9giit38s1t	0.0 %	0 ms	0.2 %	0.0 B/s	0.0 KB/s	13.9 K	8.4 K
registry.1.1ggw6m4txwqdwty29lpgg6qyh	0.0 %	0 ms	0.2 %	85.2 B/s	0.0 KB/s	1.8 K	1.5 K
voting_db.1.xleleb9uwhxeb7domfivwvul1	8.9 %	170 ms	0.1 %	249.9 KB/s	821.9 KB/s	19.5 M	19.5 M
voting_result.1.b14h0tpkkxfud5nyj9353fb3e	0.0 %	1 ms	0.2 %	333.2 B/s	0.0 KB/s	7.2 M	7.2 M

Heat maps will be refreshed automatically when you change either of the following fields:

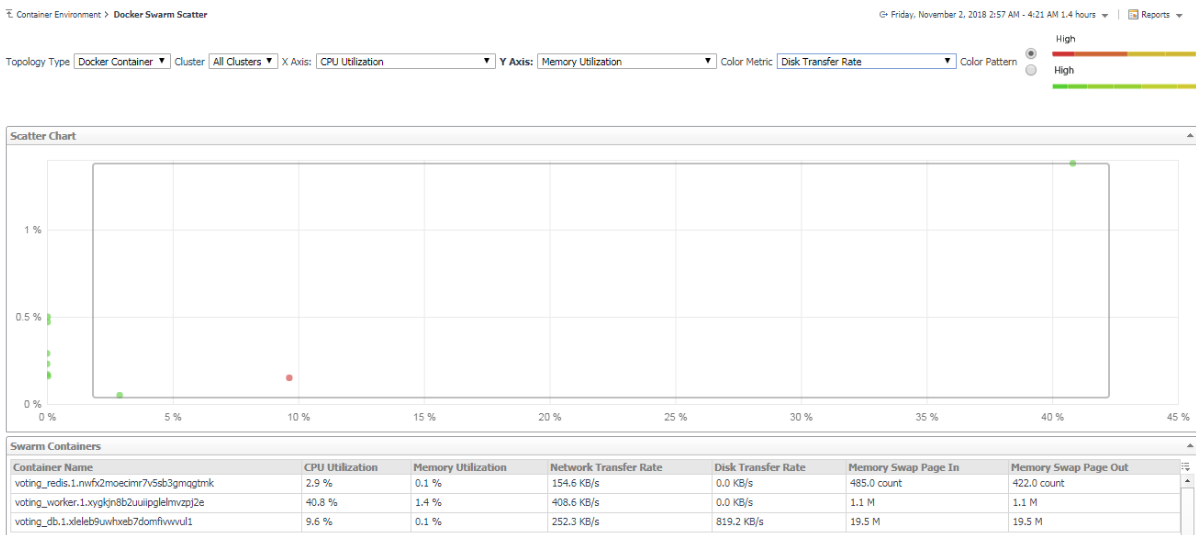
- *Topology Type:* Indicates the monitored topology object, including Docker Container and Docker Host.
- *Cluster:* Lists all clusters available in the monitored Docker Swarm environment.
- *Selected Metric:* Populates a rectangle based upon the selected metrics. For example, if you select *Memory Time Used* from the *Selected Metric* drop-down list, the rectangle area will be populated based on the used CPU time for the selected topology object. For more information about metrics, refer to Docker Swarm metrics *on page* 68.
- Rendering related metrics: For example, if you select *CPU Utilization* and Red to Green, the rectangle of the topology object that has larger value of CPU Utilization will be rendered in red.
  - *Color Metric*: Renders the color of rectangle based upon the selected color metric.
  - *Color Pattern*: Offers two patterns, Red to Green (larger value shows in red) or Green to Red (larger value shows in green).

Figure 42 shows an example of heat map. This sample diagram represents the "voting_redis.1.nwfx2moecimr7v5sb3gmqgtmk" has the maximum amounts of CPU Utilization which is the largest in size, and also it has the higher Memory Utilization since it is in Red. If you switch the Color Pattern, then "voting_redis.1.nwfx2moecimr7v5sb3gmqgtmk" will turn to green. Clicking the object name on the heat map directs you to the relevant object *Explorer* dashboard. For more information, see:

- Container Explorer view on page 53
- Docker Host Explorer view on page 54
- Container metrics on page 68

Scatter Plot analytics

Figure 43. Docker Swarm Scatter Plot Analytics Dashboard


Container Environment > Docker Swarm Scatter
Friday, November 2, 2018 2:57 AM - 4:21 AM 1.4 hours
Reports
Topology Type: Docker Container
Cluster: All Clusters
X Axis: CPU Utilization
Y Axis: Memory Utilization
Color Metric: Disk Transfer Rate
Color Pattern
High
High
Scatter Chart
1 %
0.5 %
0 %
0 %
5 %
10 %
15 %
20 %
25 %
30 %
35 %
40 %
45 %
Swarm Containers

Container Name	CPU Utilization	Memory Utilization	Network Transfer Rate	Disk Transfer Rate	Memory Swap Page In	Memory Swap Page Out
voting_redis.1.nwfx2moecimr7v5sb3gmqgtmk	2.9 %	0.1 %	154.6 KB/s	0.0 KB/s	485.0 count	422.0 count
voting_worker.1.xygkjn8b2uuiipglelmvzpj2e	40.8 %	1.4 %	408.6 KB/s	0.0 KB/s	1.1 M	1.1 M
voting_db.1.xleleb9uwhxeb7domfivwvul1	9.6 %	0.1 %	252.3 KB/s	819.2 KB/s	19.5 M	19.5 M

The points on the chart will be refreshed automatically when you change either of the following fields:

- *Topology Type:* Indicates the monitored topology object, including Docker Container and Docker Host.
- *Cluster:* Lists all clusters available in the monitored Docker Swarm environment.
- *X Axis:* Indicates which metrics will be plotted on X axis.
- *Y Axis:* Indicates which metrics will be plotted on Y axis.
- Rendering related metrics:
  - Color Metric: Renders the color of circle based upon the selected metrics.

- *Color Pattern*: Offers two patterns, Red to Green (larger value shows in red) or Green to Red (larger value shows in green).

Figure 43 shows an example of Scatter Plot analytics. The purple circle in the middle represents the following: “voting_redis.1.nwfx2moecimr7v5sb3gmqgtmk” CPU Utilization is 2.9%, its Memory Usage is 0.1%, and its value of Network Transfer Bytes is not high. For more information, see:

- Container Explorer view on page 53
- Docker Host Explorer view on page 54
- Container metrics on page 68

Domains and Object Groups

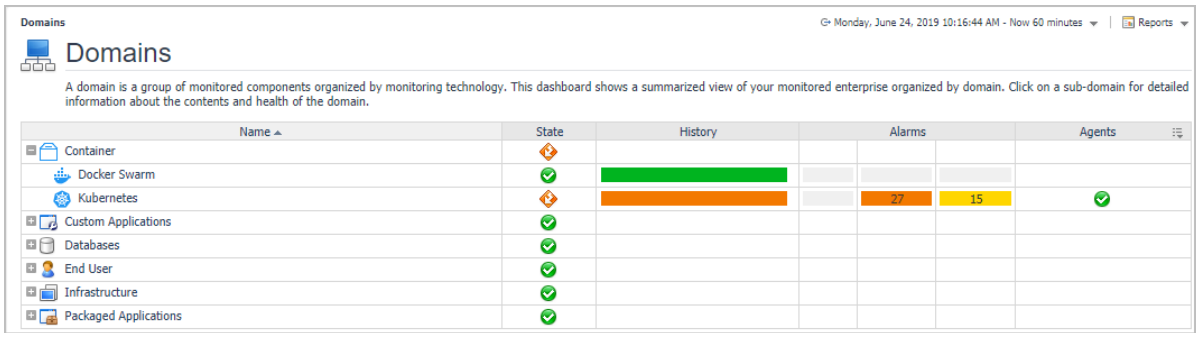
Domains

A domain is a group of monitored components organized by monitoring technology. This dashboard shows a summarized view of your monitored enterprise organized by domain. Click on a sub-domain for detailed information about the contents and health of the domain.

To access the Domains dashboard, on the Navigation panel, click **Dashboards** > **Services** > **Domains**.

Click the + icon to display the components under Container.

Figure 44. Container Components in Domains dashboard



Domains

Monday, June 24, 2019 10:16:44 AM - Now 60 minutes | Reports

Domains

A domain is a group of monitored components organized by monitoring technology. This dashboard shows a summarized view of your monitored enterprise organized by domain. Click on a sub-domain for detailed information about the contents and health of the domain.

Name	State	History	Alarms			Agents
Container	◆					
Docker Swarm	✓	🟩				
Kubernetes	◆	🟧		27	15	✓
Custom Applications	✓					
Databases	✓					
End User	✓					
Infrastructure	✓					
Packaged Applications	✓					

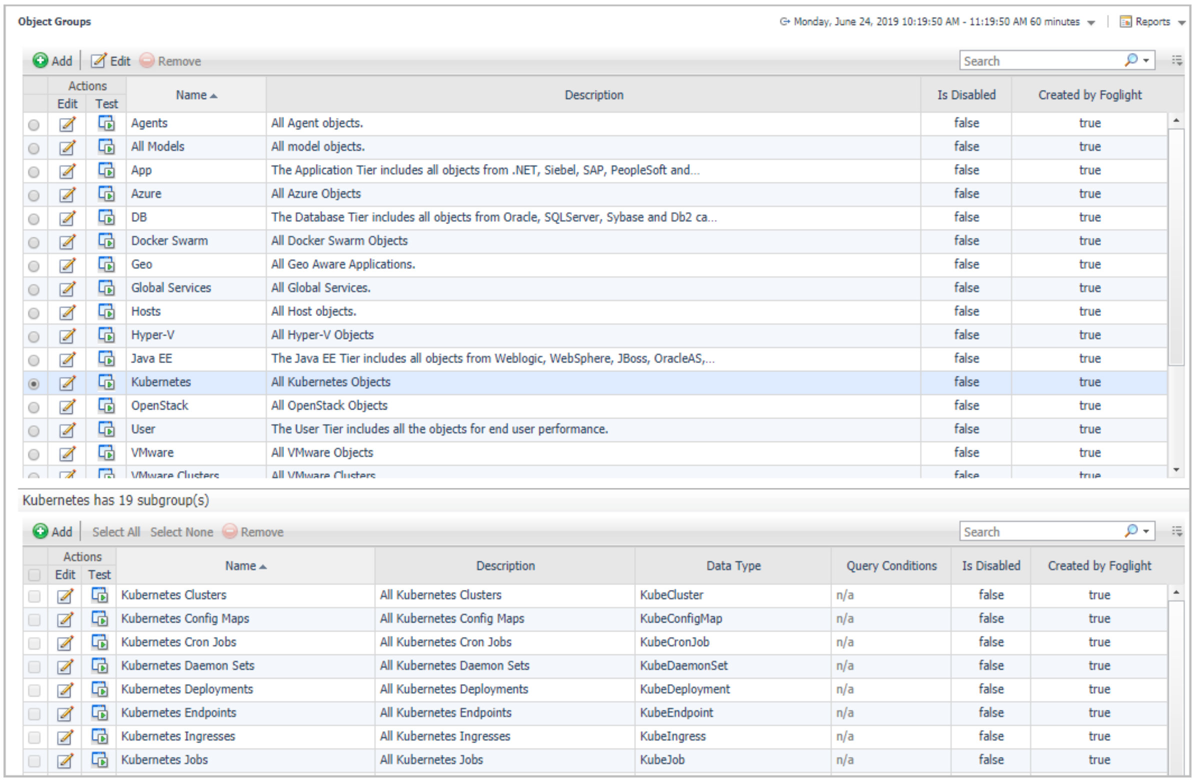
Click the State, History, Alarms, and Agents column, for detailed alarms and health information.

Object Groups

An object group is a mapping to a certain set of data types of the objects you are interested in.

To access the Object Groups dashboard, on the Navigation panel, click **Dashboards** > **Services** > **Object Groups**.

Figure 45. Object Groups for Container



Object Groups

Monday, June 24, 2019 10:19:50 AM - 11:19:50 AM 60 minutes | Reports

Add | Edit | Remove | Search

	Actions		Name	Description	Is Disabled	Created by Foglight
	Edit	Test				
○	✎	▶	Agents	All Agent objects.	false	true
○	✎	▶	All Models	All model objects.	false	true
○	✎	▶	App	The Application Tier includes all objects from .NET, Siebel, SAP, PeopleSoft and...	false	true
○	✎	▶	Azure	All Azure Objects	false	true
○	✎	▶	DB	The Database Tier includes all objects from Oracle, SQLServer, Sybase and Db2 ca...	false	true
○	✎	▶	Docker Swarm	All Docker Swarm Objects	false	true
○	✎	▶	Geo	All Geo Aware Applications.	false	true
○	✎	▶	Global Services	All Global Services.	false	true
○	✎	▶	Hosts	All Host objects.	false	true
○	✎	▶	Hyper-V	All Hyper-V Objects	false	true
○	✎	▶	Java EE	The Java EE Tier includes all objects from Weblogic, WebSphere, JBoss, OracleAS,...	false	true
◉	✎	▶	Kubernetes	All Kubernetes Objects	false	true
○	✎	▶	OpenStack	All OpenStack Objects	false	true
○	✎	▶	User	The User Tier includes all the objects for end user performance.	false	true
○	✎	▶	VMware	All VMware Objects	false	true
○	✎	▶	VMware Clusters	All VMware Clusters	false	true

Kubernetes has 19 subgroup(s)

Add | Select All | Select None | Remove | Search

	Actions		Name	Description	Data Type	Query Conditions	Is Disabled	Created by Foglight
☐	Edit	Test						
☐	✎	▶	Kubernetes Clusters	All Kubernetes Clusters	KubeCluster	n/a	false	true
☐	✎	▶	Kubernetes Config Maps	All Kubernetes Config Maps	KubeConfigMap	n/a	false	true
☐	✎	▶	Kubernetes Cron Jobs	All Kubernetes Cron Jobs	KubeCronJob	n/a	false	true
☐	✎	▶	Kubernetes Daemon Sets	All Kubernetes Daemon Sets	KubeDaemonSet	n/a	false	true
☐	✎	▶	Kubernetes Deployments	All Kubernetes Deployments	KubeDeployment	n/a	false	true
☐	✎	▶	Kubernetes Endpoints	All Kubernetes Endpoints	KubeEndpoint	n/a	false	true
☐	✎	▶	Kubernetes Ingresses	All Kubernetes Ingresses	KubeIngress	n/a	false	true
☐	✎	▶	Kubernetes Jobs	All Kubernetes Jobs	KubeJob	n/a	false	true

Select *Docker Swarm* or *Kubernetes* to display the subgroups.

Metrics

Kubernetes metrics

Pod metrics

Table 4. Pod metrics

Metric name	Description
CPU Usage Rate	CPU usage rate on all cores in millicores/second.
CPU Request	CPU request (the guaranteed amount of resources) in millicores.
CPU Limit	CPU hard limit in millicores.
CPU Utilization	Percentage of CPU usage / CPU limit if user configured CPU limit for this pod.
Memory Usage	Total memory usage in bytes.
Memory Working Set	Total working set usage. Working set is the memory being used and not easily dropped by the kernel.
Memory RSS	RSS memory usage.
Memory Cache	Number of bytes of page cache memory.

Table 4. Pod metrics

Metric name	Description
Memory Swap	Container swap usage in bytes.
Memory Request	Memory request (the guaranteed amount of resources) in bytes.
Memory Limit	Memory hard limit in bytes.
Memory Utilization	Percentage of Memory usage / Memory limit if user configured Memory limit for this pod.
Network Send Bytes Rate	Network send bytes per second.
Network Receive Bytes Rate	Network receive bytes per second.
Network Transfer Bytes Rate	Network send and receive bytes per second.
Network Send Errors Rate	Network send errors count per second.
Network Receive Errors Rate	Network receive errors count per second.
Network Transfer Errors Rate	Network send and receive errors count per second.
Network Send Packets Rate	Network send packets count per second.
Network Receive Packets Rate	Network receive packets count per second.
Network Transfer Packets Rate	Network send and receive packets count per second.
Network Send Dropped Packets Rate	Network send dropped packets count per second.
Network Receive Dropped Packets Rate	Network receive dropped packets count per second.
Network Transfer Dropped Packets Rate	Network send and receive dropped packets count per second.
Filesystem Usage	Number of bytes that are consumed by the container on this filesystem.
Filesystem Capacity	Number of bytes that can be consumed by the container on this filesystem.
Filesystem Utilization	Percentage of Filesystem Usage / Filesystem Capacity.
Filesystem Read Bytes Rate	Filesystem read bytes per second.
Filesystem Write Bytes Rate	Filesystem write bytes per second.
Filesystem Read Rate	Filesystem read counts per second.
Filesystem Write Rate	Filesystem write counts per second.

Node metrics

Table 5. Node metrics

Metric name	Description
CPU Usage Rate	CPU usage rate on all cores in millicores/second.
CPU Request	CPU request (the guaranteed amount of resources) in millicores.
CPU Limit	CPU hard limit in millicores.
CPU Utilization	CPU utilization as a share of node allocatable.
CPU Allocatable	Available CPU to allocate to workloads.
CPU Capacity	Hard CPU capacity of node.
Memory Usage	Total memory usage in bytes.
Memory Working Set	Total working set usage. Working set is the memory being used and not easily dropped by the kernel.
Memory RSS	RSS memory usage.
Memory Cache	Number of bytes of page cache memory.
Memory Swap	Container swap usage in bytes.
Memory Request	Memory request (the guaranteed amount of resources) in bytes.
Memory Capacity	Hard memory capacity of node.
Memory Limit	Memory hard limit in bytes.
Memory Allocatable	Available Memory to allocate to workloads.
Memory Utilization	Memory utilization as a share of memory allocatable.
Network Send Bytes Rate	Network send bytes per second.
Network Receive Bytes Rate	Network receive bytes per second.
Network Transfer Bytes Rate	Network send and receive bytes per second.
Network Send Errors Rate	Network send errors count per second.
Network Receive Errors Rate	Network receive errors count per second.
Network Transfer Errors Rate	Network send and receive errors count per second.
Network Send Packets Rate	Network send packets count per second.
Network Receive Packets Rate	Network receive packets count per second.
Network Transfer Packets Rate	Network send and receive packets count per second.
Network Send Dropped Packets Rate	Network send dropped packets count per second.
Network Receive Dropped Packets Rate	Network receive dropped packets count per second.
Network Transfer Dropped Packets Rate	Network send and receive dropped packets count per second.
Filesystem Usage	Number of bytes that are consumed by the container on this filesystem.
Filesystem Capacity	Number of bytes that can be consumed by the container on this filesystem.
Filesystem Utilization	Percentage of Filesystem Usage / Filesystem Capacity.
Filesystem Inodes Usage	Number of Inodes that are consumed by the container on this filesystem.
Filesystem Inodes Total	Number of Inodes that can be consumed by the container on this filesystem.
Filesystem Inodes Utilization	Percentage of Filesystem Inodes Usage / Filesystem Inodes Capacity.
Filesystem Read Bytes Rate	Filesystem read bytes per second.
Filesystem Write Bytes Rate	Filesystem write bytes per second.

Table 5. Node metrics

Metric name	Description
Filesystem Read Rate	Filesystem read counts per second.
Filesystem Write Rate	Filesystem write counts per second.

Docker Swarm metrics

Container metrics

Table 6. Container metrics

Metric name	Description
CPU Utilization	CPU utilization.
CPU Time Used	Total CPU time that a container used.
CPU Throttled Time	Total time that a container's CPU usage was throttled.
Memory Page Fault	Total page fault count of a container's Memory.
Memory Consumed	Total memory consumed of a container in bytes.
Memory Utilization	Memory utilization.
Memory PageIn Rate	Total page in count of a container's Memory.
Memory PageOut Rate	Total page out count of a container's Memory.
Disk Read Bytes	Total disk read bytes.
Disk Write Bytes	Total disk write bytes.
Disk Transfer Rate	Sum of total disk read and write bytes.
Network Send Packets	Total network send packets count.
Network Receive Packets	Total network receive packets count.
Network Send Bytes	Total network send bytes.
Network Receive Bytes	Total network receive bytes.
Network Inbound Dropped Packets	Total dropped packet count of all the packets coming into the container.
Network Outbound Dropped Packets	Total dropped packet count of all the packets going out from the container.
Network Transfer Rate	Sum of network send bytes and receive bytes per seconds during a specific period.

Rules

Foglight for Container Management allows you to create flexible rules that can be applied to complex interrelated data from multiple sources within your clusters. You can associate several different actions with a rule, configure a rule so that it does not fire repeatedly, and associate a rule with schedules to define when it should be evaluated or not.

Different types of data can be used in rules, including registry variables, raw metrics, derived metrics, and topology object properties.

There are two types of rules: simple rules and multiple-severity rules. A simple rule has a single condition, and can be in one of three states: *Fire*, *Undefined*, or *Normal*. A multiple-severity rule can have up to five severity levels: *Undefined*, *Fatal*, *Critical*, *Warning*, and *Normal*.

Rule conditions are regularly evaluated against monitoring data (metrics and topology object properties collected from your monitored environment and transformed into a standard format). Therefore, the state of the rule can change if the data changes. For example, if a set of monitoring data matches a simple rule's condition, the rule enters the *Fire* state. If the next set does not match the condition, the rule exits the *Fire* state and enters the *Normal* state.

Rules can be configured to send emails, pager messages, or perform other actions you define. Performance data can be viewed and analyzed using Foglight for Container Management.

Foglight for Container Management includes a number of predefined rules used to monitor the health of your container clusters. You are allowed to modify these rules to satisfy your different requirements. Many of these rules listed and described in this section have thresholds defined within them. Those thresholds include standard deviations, utilization percentages, and so on, are default values predefined in the registry.

Kubernetes

All rules are controlled by registry variable Kubernetes:AlertSensitivity. If the value is 0, then no alarm can be fired. If the value is 1, warning level alarm can be fired. If the value is above 1, then all level alarm can be fired.

Kubernetes Administrator email address can be configured in Registry Variable KubernetesAdmin.

Health Check

Kubernetes Pod Health Check

Purpose

This rule detects abnormal Pod health status and fires alarm for different severity abnormal health status.

Scope

KubePod

Conditions and Severities

Conditions	Severity	Action
Pods that is in Failed or Unknown status. Or the node which is running the pod gets disconnected.	Critical	Send email to Kubernetes Administrator.
Pods that is in CrashLoopBackOff status.	Warning	None

Kubernetes Pod Health Check (Pending Phase)

Purpose

This rule detects Pods that stays in pending phase for an abnormal long time.

Scope

KubePod

Conditions and Severities

Conditions	Severity	Action
Pods that is pending for two continuous data submission periods because of Failed to schedule to Node.	Critical	Send email to Kubernetes Administrator.
Pods that is pending for two continuous data submission periods because container is not ready.	Warning	None

Kubernetes Container Health Check

Purpose

This rule detects abnormal Container health status and fires alarm for different severity abnormal health status.

Scope

KubeContainer

Conditions and Severities

Conditions	Severity	Action
Container that is terminated for abnormal reasons.	Critical	Send email to Kubernetes Administrator.

Kubernetes Node Health Check

Purpose

This rule detects abnormal Node health status and fires alarm for different severity abnormal health status.

Scope

KubeNode

Conditions and Severities

Conditions	Severity	Action
Nodes that is not Ready or out of disk or network unavailable.	Critical	Send email to Kubernetes Administrator.
Nodes whose memory or disk is under pressure.	Warning	None

Kubernetes Deployment Health Check

Purpose

This rule detects abnormal Deployment health status and fires alarm for different severity abnormal health status.

Scope

KubeDeployment

Conditions and Severities

Conditions	Severity	Action
Deployment is not available.	Critical	Send email to Kubernetes Administrator.
Deployment has failed to create some of the replicated pods.	Warning	None

Kubernetes Daemon Set Health Check

Purpose

This rule detects abnormal Daemon Set health status and fires alarm for different severity abnormal health status.

Scope

KubeDaemonSet

Conditions and Severities

Conditions	Severity	Action
Some of the pods created by the Daemon Set is not available or mis-scheduled.	Critical	Send email to Kubernetes Administrator.
The daemon set doesn’t have enough replicated pods running that meets its desired replicated pods count.	Warning	None

Kubernetes Job Health Check

Purpose

This rule detects abnormal Job health status and fires alarm for different severity abnormal health status.

Scope

KubeJob

Conditions and Severities

Conditions	Severity	Action
Job that is failed.	Warning	None

Kubernetes Persistent Volume Health Check

Purpose

This rule detects abnormal Persistent Volume health status and fires alarm for different severity abnormal health status.

Scope

KubePersistentVolume

Conditions and Severities

Conditions	Severity	Action
Persistent Volume that is in failed status.	Warning	None

Kubernetes Persistent Volume Claim Health Check

Purpose

This rule detects abnormal Persistent Volume Claim health status and fires alarm for different severity abnormal health status.

Scope

KubePersistentVolumeClaim

Conditions and Severities

Conditions	Severity	Action
Persistent Volume Claim that is in failed status.	Warning	None

Kubernetes Persistent Volume Claim Health Check (Long Pending)

Purpose

This rule detects abnormal long pending Persistent Volume Claim and fires alarm for different severities.

Scope

KubePersistentVolumeClaim

Conditions and Severities

Conditions	Severity	Action
Persistent Volume Claim that is pending for two continuous data submission periods.	Critical	None

Usage

NOTE: All the Pod usage related rules are disabled by default, these rules are used as default values or examples for the customers to customize their different kinds of workloads. For how to customize the rules, refer to Customization on page 83.

Kubernetes Pod CPU Utilization

Purpose

This rule detects abnormal CPU Utilization for Pods, and fires alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities. This rule only works for those Pods that configures CPU limit.

Scope

KubePodCpu

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodCpu UtilizationFatal	Send email to Kubernetes Administrator
Pods whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes:PodCpu UtilizationCritical	None
Pods whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes:PodCpu UtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Pod Memory Utilization

Purpose

This rule detects abnormal Memory Utilization for Pods, and fires alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities. This rule only works for those Pods that configures Memory limit.

Scope

KubePodMemory

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodMemory UtilizationFatal	Send email to Kubernetes Administrator
Pods whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes:PodMemory UtilizationCritical	None
Pods whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes:PodMemory UtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Pod CPU Usage

Purpose

This rule detects abnormal CPU Usage for Pods, and fires alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubePodCpu

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodCpu UsageFatal	Send email to Kubernetes Administrator
Pods whose usage is above the value configured in critical Threshold.	Critical	Kubernetes:PodCpu UsageCritical	None
Pods whose usage is above the value configured in warning Threshold.	Warning	Kubernetes:PodCpu UsageWarning	None

*Note: the unit is millicores/second.

Kubernetes Pod Memory Usage

Purpose

This rule detects abnormal Memory Usage for Pods, and fires alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubePodMemory

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodMemory UsageFatal	Send email to Kubernetes Administrator
Pods whose usage is above the value configured in critical Threshold.	Critical	Kubernetes:PodMemory UsageCritical	None
Pods whose usage is above the value configured in warning Threshold.	Warning	Kubernetes:PodMemory UsageWarning	None

*Note: the unit is bytes.

Kubernetes Pod Network Receive

Purpose

This rule detects abnormal Network Receive Rate in bytes/second for Pods, and fire alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubePodNetwork

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodNetwork ReceiveFatal	Send email to Kubernetes Administrator
Pods whose usage is above the value configured in critical Threshold.	Critical	Kubernetes:PodNetwork ReceiveCritical	None
Pods whose usage is above the value configured in warning Threshold.	Warning	Kubernetes:PodNetwork ReceiveWarning	None

*Note: the unit is bytes/second.

Kubernetes Pod Network Send

Purpose

This rule detects abnormal Network Send Rate in bytes/second for Pods, and fire alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubePodNetwork

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is above the value configured in fatal Threshold.	Fatal	Kubernetes:PodNetwork SendFatal	Send email to Kubernetes Administrator
Pods whose usage is above the value configured in critical Threshold.	Critical	Kubernetes:PodNetwork SendCritical	None
Pods whose usage is above the value configured in warning Threshold.	Warning	Kubernetes:PodNetwork SendWarning	None

*Note: the unit is bytes/second.

Kubernetes Pod Filesystem Utilization

Purpose

This rule checks kubernetes pod filesystem utilization to see if it reaches the defined threshold. The filesystem utilization rule only works for pod that configures filesystem limit.

Scope

KubePodStorage

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Pods whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes: PodFilesystemUtilization Fatal	Send email to Kubernetes Administrator
Pods whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes: PodFilesystemUtilization Critical	None
Pods whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes: PodFilesystemUtilization Warning	None

*Note: the unit is percentage.

Kubernetes Node CPU Utilization

Purpose

This rule detects abnormal CPU Utilization for Nodes, and fire alarm on different severities. It is enabled by default. You can change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubeNodeCpu

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose utilization is above the value configured in fatal Threshold.	Fatal	Kubernetes:NodeCpu UtilizationFatal	Send email to Kubernetes Administrator
Nodes whose utilization is above the value configured in critical Threshold.	Critical	Kubernetes:NodeCpu UtilizationCritical	None
Nodes whose utilization is above the value configured in warning Threshold.	Warning	Kubernetes:NodeCpu UtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Node Memory Utilization

Purpose

This rule detects abnormal Memory Utilization for Nodes, and fire alarm on different severities. It is enabled by default. You can change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubeNodeMemory

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose utilization is above the value configured in fatal Threshold.	Fatal	Kubernetes:NodeMemory UtilizationFatal	Send email to Kubernetes Administrator
Nodes whose utilization is above the value configured in critical Threshold.	Critical	Kubernetes:NodeMemory UtilizationCritical	None
Nodes whose utilization is above the value configured in warning Threshold.	Warning	Kubernetes:NodeMemory UtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Node Network Receive

Purpose

This rule detects abnormal Network Receive Rate in bytes/second for Nodes, and fire alarm on different severities. It is enabled by default. You can change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubeNodeNetwork

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose utilization is above the value configured in fatal Threshold.	Fatal	Kubernetes:NodeNetwork ReceiveFatal	Send email to Kubernetes Administrator
Nodes whose utilization is above the value configured in critical Threshold.	Critical	Kubernetes:NodeNetwork ReceiveCritical	None
Nodes whose utilization is above the value configured in warning Threshold.	Warning	Kubernetes:NodeNetwork ReceiveWarning	None

*Note: the unit is bytes/second.

Kubernetes Node Network Send

Purpose

This rule detects abnormal Network Send Rate in bytes/second for Nodes, and fire alarm on different severities. It is enabled by default. You can change value of the registry variables or use your own value to change the threshold of each severities.

Scope

KubeNodeNetwork

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose utilization is above the value configured in fatal Threshold.	Fatal	Kubernetes:NodeNetwork SendFatal	Send email to Kubernetes Administrator
Nodes whose utilization is above the value configured in critical Threshold.	Critical	Kubernetes:NodeNetwork SendCritical	None
Nodes whose utilization is above the value configured in warning Threshold.	Warning	Kubernetes:NodeNetwork SendWarning	None

*Note: the unit is bytes/second.

Kubernetes Node Network Transfer

Purpose

Periodically check Kubernetes node Network Transfer Rate in bytes/second, if the value is too high and changes too much, then an alarm will be triggered.

Scope

KubeNodeNetwork

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes: NodeNetworkTransferFatal	Send email to Kubernetes Administrator
Nodes whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes: NodeNetworkTransferCritical	None
Nodes whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes: NodeNetworkTransferWarning	None

*Note: the unit is bytes/second.

Kubernetes Node Filesystem Utilization

Purpose

Periodically check Kubernetes node Filesystem Utilization, if the value is too high and changes too much, then an alarm will be triggered.

Scope

KubeNodeFilesystem

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Nodes whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	KubeNodeFilesystemUtilizationFatal	Send email to Kubernetes Administrator
Nodes whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	KubeNodeFilesystemUtilizationCritical	None
Nodes whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	KubeNodeFilesystemUtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Container CPU Utilization

Purpose

This rule checks Kubernetes container CPU utilization to see if it reaches the defined threshold. The CPU utilization rule only works for container that configures CPU limit.

Scope

KubeContainerCpu

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Containers whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes:ContainerCpuUtilizationFatal	Send email to Kubernetes Administrator
Containers whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes:ContainerCpuUtilizationCritical	None
Containers whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes:ContainerCpuUtilizationWarning	None

*Note: the unit is percentage.

Kubernetes Container Memory Utilization

Purpose

This rule checks Kubernetes container Memory utilization to see if it reaches the defined threshold. The Memory utilization rule only works for container that configures Memory limit.

Scope

KubeContainerMemory

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Containers whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Kubernetes: ContainerMemoryUtilizationFatal	Send email to Kubernetes Administrator
Containers whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Kubernetes: ContainerMemoryUtilizationCritical	None
Containers whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Kubernetes: ContainerMemoryUtilizationWarning	None

*Note: the unit is percentage.

Docker Swarm

All rules are controlled by registry variable Docker:AlertSensitivity. If the value is 0, then no alarm can be fired. If the value is 1, warning level alarm can be fired. If the value is above 1, then all level alarm can be fired.

Docker Swarm Administrator email address can be configured in Registry Variable Docker:DockerAdmin.

Health Check

Docker Container Status

Purpose

This rule detects abnormal Container health status and fires alarm for different severity abnormal health status.

Scope

DockerContainer

Conditions and Severities

Conditions	Severity	Action
Container that is already stopped for abnormal reason.	Critical	Send email to Docker Swarm Administrator

Docker Container Status - Paused

Purpose

This rule detects abnormal long-time paused Container and fires alarm for different severity abnormal health status.

Scope

DockerContainer

Conditions and Severities

Conditions	Severity	Action
Container paused for two continuous data submission periods.	Warning	None

Docker Service Status

Purpose

This rule detects abnormal Docker Swarm Service health status and fires alarm for different severity abnormal health status.

Scope

DockerService

Conditions and Severities

Conditions	Severity	Action
Missing some of the replicated task running for this service.	Critical	Send email to Docker Swarm Administrator

Docker Task Status

Purpose

This rule detects abnormal Docker Swarm Task health status and fires alarm for different severity abnormal health status.

Scope

DockerTask

Conditions and Severities

Conditions	Severity	Action
Task that is in failed, orphaned or remove status.	Critical	Send email to Docker Swarm Administrator

Docker Task Status -- pending

Purpose

This rule detects abnormal long-time pending Docker Swarm Task and fires alarm for different severity abnormal health status.

Scope

DockerTask

Conditions and Severities

Conditions	Severity	Action
Task that is in pending status for two continuous data submission periods.	Warning	None

Usage

Docker Swarm Container CPU Utilization

Purpose

This rule detects abnormal CPU Utilization for Docker Swarm Containers, and fire alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities. This rule only works for those Containers that configures CPU limit.

Scope

DockerContainerCPU

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Container whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Docker:ContainerCpuUtilizationFatal	Send email to Kubernetes Administrator
Pods whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Docker:ContainerCpuUtilizationCritical	None
Pods whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Docker:ContainerCpuUtilizationWarning	None

*Note: the unit is percentage.

Docker Swarm Container Memory Utilization

Purpose

This rule detects abnormal Memory Utilization for Docker Swarm Containers, and fire alarm on different severities. It is disabled by default. You can customize it and enable it based on your different requirements. For more details about customization, refer to Customization on page 83. You can also change value of the registry variables or use your own value to change the threshold of each severities. This rule only works for those Containers that configures Memory limit.

Scope

DockerContainerMemory

Conditions and Severities

Conditions	Severity	Threshold (Registry Variable)*	Action
Container whose usage is about to reach the limit, the ration is above the value configured in fatal Threshold.	Fatal	Docker:ContainerMemory UtilizationFatal	Send email to Kubernetes Administrator
Pods whose usage is about to reach the limit, the ration is above the value configured in critical Threshold.	Critical	Docker:ContainerMemory UtilizationCritical	None
Pods whose usage is about to reach the limit, the ration is above the value configured in warning Threshold.	Warning	Docker:ContainerMemory UtilizationWarning	None

*Note: the unit is percentage.

Customization

To customize a rule, *Rule Scope* and *Condition* will be used frequently.

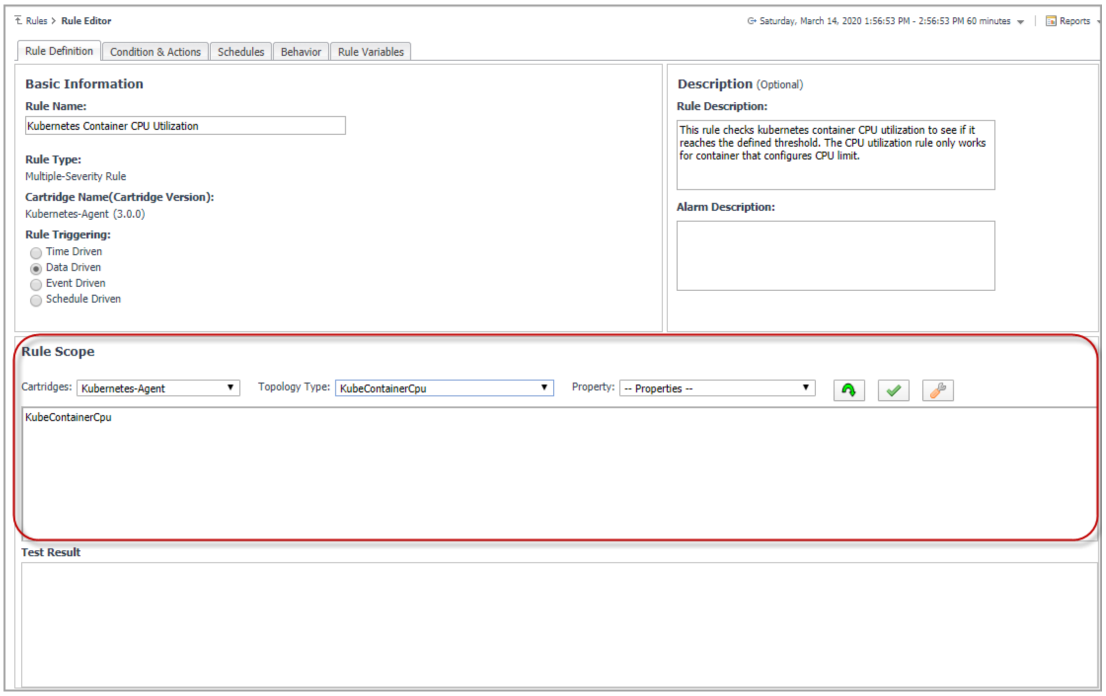
To access Rule Scope and Condition, do the following:

1 Under **Dashboards**, click **Administration** > **Rules & Notifications** > **Rules**, then click on the rule and select *View and Edit*.

2 Click **Rule Editor** on the *Rule Detail* popup dialog box. Then click **Continue** on the *Confirm Edit Rule* popup dialog box.

3 On the **Rule Editor** dashboard, *Rule Scope* can be located on the **Rule Definition** tab and Condition can be located on the **Condition & Actions** tab.

Figure 46. Rule Scope



Rules > Rule Editor

Saturday, March 14, 2020 1:56:53 PM - 2:56:53 PM 60 minutes | Reports

Rule Definition | Condition & Actions | Schedules | Behavior | Rule Variables

Basic Information

Rule Name: Kubernetes Container CPU Utilization

Rule Type: Multiple-Severity Rule

Cartridge Name(Cartridge Version): Kubernetes-Agent (3.0.0)

Rule Triggering:
- ○ Time Driven
- ● Data Driven
- ○ Event Driven
- ○ Schedule Driven

Description (Optional)

Rule Description: This rule checks kubernetes container CPU utilization to see if it reaches the defined threshold. The CPU utilization rule only works for container that configures CPU limit.

Alarm Description:

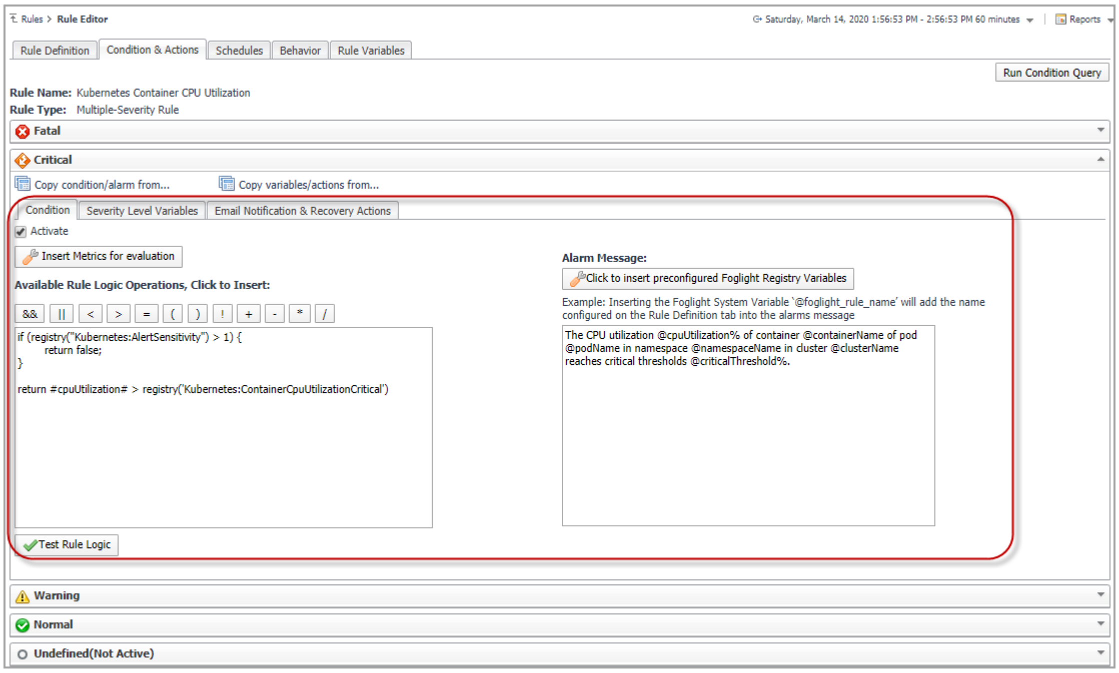
Rule Scope

Cartridges: Kubernetes-Agent | Topology Type: KubeContainerCpu | Property: -- Properties --

```
KubeContainerCpu
```

Test Result

Figure 47. Condition



Rules > Rule Editor

Saturday, March 14, 2020 1:56:53 PM - 2:56:53 PM 60 minutes | Reports

Rule Definition | Condition & Actions | Schedules | Behavior | Rule Variables

Run Condition Query

Rule Name: Kubernetes Container CPU Utilization
Rule Type: Multiple-Severity Rule

Fatal

Critical

Copy condition/alarm from... | Copy variables/actions from...

Condition | Severity Level Variables | Email Notification & Recovery Actions

✓ Activate

Insert Metrics for evaluation

Available Rule Logic Operations, Click to Insert:

&& | || | < | > | = | (|) | ! | + | - | * | /

```
if (registry("Kubernetes:AlertSensitivity") > 1) {
      return false;
}

return #cpuUtilization# > registry('Kubernetes:ContainerCpuUtilizationCritical')
```

Test Rule Logic

Alarm Message:

Click to insert preconfigured Foglight Registry Variables

Example: Inserting the Foglight System Variable `@foglight_rule_name' will add the name configured on the Rule Definition tab into the alarms message

The CPU utilization @cpuUtilization% of container @containerName of pod @podName in namespace @namespaceName in cluster @clusterName reaches critical thresholds @criticalThreshold%.

Warning

Normal

Undefined(Not Active)

Kubernetes

Filter Pods by Cluster

Finding Pods inside cluster “kubecluster”, enter following statement in the Scope of a rule, and choose KubePod as the Topology Type in the Rule Scope.

```
KubePod where namespace.cluster.name='kubecluster'
```

Filter Pods by Namespace

Finding Pods inside namespace “default” of Cluster “kubecluster”, enter following statement in the Scope of a rule, and choose KubePod as the Topology Type in the Rule Scope.

```
KubePod where namespace.cluster.name='kubecluster' and namespace.name='test'
```

Filter Nodes by Cluster

Finding Nodes inside cluster “kubecluster”, enter following statement in the Scope of a rule, and choose KubeNode as the Topology Type in the Rule Scope.

```
KubeNode where cluster.name='nancyakscluster'
```

Filter Pod by Labels

Find Pods with labels “run=nginx” and “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose KubePod as the Topology Type in the Rule Scope.

```
KubePod where labels.key='run' and labels.value='nginx-rollingupdate' and
labels.key=’env’ and labels.value=’prod’
```

If you want to find Pods by labels in namespace “test” of cluster “kubecluster”, you can append `and namespace.cluster.name='kubecluster' and namespace.name='test'` to the end of above statement.

Filter Node by Labels

Find Nodes with labels “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose KubeNode as the Topology Type in the Rule Scope.

```
KubeNode where labels.key='env' and labels.value='prod'
```

If you want to find Nodes by labels in cluster “kubecluster”, you can append `and cluster.name='kubecluster'` to the end of above statement.

Filter Pod Metrics by Pod Labels

Find Pods Metrics with labels “run=nginx” and “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose KubeHeapsterMetrics as the Topology Type in the Rule Scope.

```
KubePod.metrics where object.labels.key='run' and object.labels.value=’nginx’ and
object.labels.key='env' and object.labels.value=’prod’
```

If you want to find Pods by labels in namespace “test” of cluster “kubecluster”, you can append `and namespace.cluster.name='kubecluster' and namespace.name='test'` to the end of above statement.

Filter Nodes Metrics by Node Labels

Find Node Metrics with labels “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose KubeHeapsterMetrics as the Topology Type in the Rule Scope.

```
KubeNode.metrics where object.labels.key='env' and object.labels.value=’prod’
```

If you want to find Nodes by labels in cluster “kubecluster”, you can append `and cluster.name='kubecluster'` to the end of above statement.

Docker Swarm

Filter Container by Swarm Cluster

Find Containers in cluster “dockercluster”, enter following statement in the Scope of a rule, and choose DockerContainer as the Topology Type in the Rule Scope.

```
DockerContainer where dockerSwarm.service.cluster.name='kicakdscluster'
```

Filter Container by Labels

Find Containers with labels “com.docker.stack.namespace=nginx” and “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose DockerContainer as the Topology Type in the Rule Scope.

```
DockerContainer where labels.key='com.docker.stack.namespace' and
labels.value='nginx' and labels.key='env' and labels.value='prod'
```

If you want to find Containers by labels in cluster “swarmcluster”, you can append `and dockerSwarm.service.cluster.name='kicakdscluster'` to the end of above statement.

Filter Docker Host by Swarm Cluster

Find Docker Hosts in cluster “dockercluster”, enter following statement in the Scope of a rule, and choose DockerHost as the Topology Type in the Rule Scope.

```
DockerHost where dockerSwarmNodeInfo.node.cluster.name='kicakdscluster'
```

Filter Container CPU Usage by Container Labels

Find Container CPU Usage by container labels “com.docker.stack.namespace=nginx” and “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose DockerContainerCPU as the Topology Type in the Rule Scope.

```
DockerContainerCPU where container.labels.key='com.docker.stack.namespace' and
container.labels.value='nginx' and container.labels.key='env' and
container.labels.value='prod'
```

If you want to find Containers by labels in cluster “swarmcluster”, you can append `and container.dockerSwarm.service.cluster.name='kicakdscluster'` to the end of above statement.

Filter Container Memory Usage by Container Labels

Find Container CPU Usage by container labels “com.docker.stack.namespace=nginx” and “env=prod” among all clusters, enter following statement in the Scope of a rule, and choose DockerContainerMemory as the Topology Type in the Rule Scope.

```
DockerContainerMemory where container.labels.key='com.docker.stack.namespace' and
container.labels.value='nginx' and container.labels.key='env' and
container.labels.value='prod'
```

If you want to find Containers by labels in cluster “swarmcluster”, you can append `and container.dockerSwarm.service.cluster.name='kicakdscluster'` to the end of above statement.

About Us

We are more than just a name

We are on a quest to make your information technology work harder for you. That is why we build community-driven software solutions that help you spend less time on IT administration and more time on business innovation. We help you modernize your data center, get you to the cloud quicker and provide the expertise, security and accessibility you need to grow your data-driven business. Combined with Quest's invitation to the global community to be a part of its innovation, and our firm commitment to ensuring customer satisfaction, we continue to deliver solutions that have a real impact on our customers today and leave a legacy we are proud of. We are challenging the status quo by transforming into a new software company. And as your partner, we work tirelessly to make sure your information technology is designed for you and by you. This is our mission, and we are in this together. Welcome to a new Quest. You are invited to Join the Innovation™.

Our brand, our vision. Together.

Our logo reflects our story: innovation, community and support. An important part of this story begins with the letter Q. It is a perfect circle, representing our commitment to technological precision and strength. The space in the Q itself symbolizes our need to add the missing piece—you—to the community, to the new Quest.

Contacting Quest

For sales or other inquiries, visit https://www.quest.com/company/contact-us.aspx/.

Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at https://support.quest.com.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request.
- View Knowledge Base articles.
- Sign up for product notifications.
- Download software and technical documentation.
- View how-to-videos.
- Engage in community discussions.
- Chat with support engineers online.
- View services to assist you with your product.