VMware vRealize Operations Management Pack for Container Monitoring

Management Packs for vRealize Operations Manager 1.4.1
You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

If you have comments about this documentation, submit your feedback to
docfeedback@vmware.com
# Contents

vRealize Operations Management Pack for Container Monitoring  4

1 Introduction  5

2 Install the vRealize Operations Management Pack for Container Monitoring  6
   cAdvisor YAML Definition  6

3 Configuring vRealize Operations Management Pack for Container Monitoring  11
   Configure PKS Adapter  11
   Configure Kubernetes Adapter  12

4 Dashboards in vRealize Operations Management Pack for Container Monitoring  15
   Kubernetes Overview  15
      Kubernetes Overview - Environment  15
      Kubernetes Overview - Nodes  16
      Kubernetes Overview - Pods and Container  17

5 Metrics in vRealize Operations Management Pack for Container Monitoring  19
   Super Metrics  22

6 Alerts in vRealize Operations Management Pack for Container Monitoring  23

7 Reports in vRealize Operations Management Pack for Container Monitoring  28
vRealize Operations Management Pack for Container Monitoring

The vRealize Operations Management Pack for Container Monitoring provides information for automating deployment and scaling operations of application containers across clusters of hosts providing container-centric infrastructure. The current version of the management pack supports monitoring Kubernetes clusters and containers deployed using the same.

Intended Audience

This information is intended for anyone who wants to install, and use vRealize Operations Management Pack for Container Monitoring for monitoring their Kubernetes Clusters and containers deployed through Kubernetes.
Introduction

With VMware vRealize® Operations Management Pack™ for Container Monitoring, Virtual Infrastructure Administrators can get complete Kubernetes topology of Namespaces, Clusters, Replica Sets, Nodes, Pods, and Containers for monitoring Kubernetes clusters. The OOTB dashboard not only provides an overview of Kubernetes eco-system but also helps in troubleshooting by highlighting the Key Performance Index and alerts for various objects pertaining to Kubernetes clusters that are monitored. This management pack extends the monitoring capability of vRealize Operations Manager to provide insights to the Kubernetes clusters to the Virtual Infrastructure administrator.
Install the vRealize Operations Management Pack for Container Monitoring


**Prerequisites**
- Verify that you have the latest version of the management pack with administrator privileges.
- Install the cAdvisor DaemonSet on the cluster. Based on the Kubernetes settings, you must create a cAdvisor YAML definition. For more information, see cAdvisor YAML Definition.

**Procedure**
1. Log in to the vRealize Operations Manager with administrator privileges.
2. From the main menu of vRealize Operations Manager, click Administration, and then in the left pane, click Solutions.
3. From the Solutions page, click the plus sign.
4. Browse to locate the temporary folder and select the PAK file.
5. Click Upload. The upload might take several minutes.
6. Read and accept the EULA, and click Next.
7. When the vRealize Operations Management Pack for Container Monitoring is installed, click Finish.

**cAdvisor YAML Definition**

Before you install vRealize Operations Management Pack for Container Monitoring, you must deploy the cAdvisor DaemonSet on the cluster. Based on the Kubernetes settings, you must create a cAdvisor YAML definition.
Here are a few points to consider when you create a cAdvisor YAML definition:

- Containers running on `hostPort` must be accessible on your cluster. For example, the sample YAML definition on `hostPort` given below has port 31194 as the `hostPort`. So, the cluster must allow a connection on port 31194.

  If the containers running on `hostPort` are not accessible, verify with `hostNetwork`. A sample YAML definition on `hostNetwork` specific to PKS is provided below.

- The docker path configured in the volume must be correct.

  **Note** The docker path can be different based on your settings.

- All the nodes must have sufficient CPU and memory to run DaemonSets.

- You must use the `hostPort` defined in the YAML definition as the cAdvisor port when you create an adapter instance.

### Sample cAdvisor YAML Definition on HostPort

```yaml
apiVersion: apps/v1 # apps/v1beta2 in Kube 1.8, extensions/v1beta1 in Kube < 1.8
kind: DaemonSet
metadata:
  name: cadvisor
  namespace: kube-system
  labels:
    app: cadvisor
  annotations:
    seccomp.security.alpha.kubernetes.io/pod: 'docker/default'
spec:
  selector:
    matchLabels:
      app: cadvisor
  template:
    metadata:
      labels:
        app: cadvisor
        version: v0.31.0
    spec:
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
      - name: cadvisor
        image: google/cadvisor:v0.31.0
        resources:
          requests:
            memory: 250Mi
            cpu: 250m
          limits:
            cpu: 400m
        volumeMounts:
        - name: rootfs
          mountPath: /rootfs
```
Sample cAdvisor YAML Definition on HostNetwork

```yaml
apiVersion: apps/v1beta2  # apps/v1beta2 in Kube 1.8, extensions/v1beta1 in Kube < 1.8
kind: DaemonSet
metadata:
  name: vrops-cadvisor
  namespace: kube-system
labels:
  app: vrops-cadvisor
spec:
  selector:
    matchLabels:
      name: vrops-cadvisor
template:
    metadata:
      labels:
        name: vrops-cadvisor
    spec:
      containers:
      - name: var-run
        mountPath: /var/run
        readOnly: true
      - name: sys
        mountPath: /sys
        readOnly: true
      - name: docker
        mountPath: /var/lib/docker  #Mounting Docker volume
        readOnly: true
      - name: disk
        mountPath: /dev/disk
        readOnly: true
      ports:
      - name: http
        containerPort: 8080  #Port exposed
        hostPort: 31194  #Host's port - Port to expose your cAdvisor DaemonSet on each node
        protocol: TCP
        automountServiceAccountToken: false
      terminationGracePeriodSeconds: 30
      volumes:
      - name: rootfs
        hostPath:
          path: /
      - name: var-run
        hostPath:
          path: /var/run
      - name: sys
        hostPath:
          path: /sys
      - name: docker
        hostPath:
          path: /var/lib/docker  #Docker path in Host System
      - name: disk
        hostPath:
          path: /dev/disk
```

VMware vRealize Operations Management Pack for Container Monitoring

VMware, Inc.
version: v0.31.0

spec:
tolerations:
  - key: node-role.kubernetes.io/master
effect: NoSchedule
hostNetwork: true
containers:
  - name: vrops-cadvisor
    image: google/cadvisor:v0.31.0
    imagePullPolicy: Always
    volumeMounts:
      - name: rootfs
        mountPath: /rootfs
        readOnly: true
      - name: var-run
        mountPath: /var/run
        readOnly: false
      - name: sys
        mountPath: /sys
        readOnly: true
      - name: docker
        mountPath: /var/lib/docker #Mounting Docker volume
        readOnly: true
      - name: docker-sock
        mountPath: /var/run/docker.sock
        readOnly: true
      - name: containerd-sock
        mountPath: /var/run/containerd.sock
        readOnly: true
      - name: disk
        mountPath: /dev/disk
        readOnly: true
    ports:
      - name: http
        containerPort: 31194 #Port exposed
        hostPort: 31194  #Host's port - Port to expose your cAdvisor DaemonSet on each node
        protocol: TCP
    securityContext:
      capabilities:
        drop:
        - ALL
        add:
        - NET_BIND_SERVICE
      args:
        - --port=31194
        - --profiling
        - --housekeeping_interval=1s
      terminationGracePeriodSeconds: 30
    volumes:
      - name: rootfs
        hostPath:
          path: /
      - name: var-run
        hostPath:
          path: /var/run
- name: sys
  hostPath:
    path: /sys
- name: docker
  hostPath:
    path: /var/vcap/store/docker/docker #Docker path in Host System
- name: docker-sock
  hostPath:
    path: /var/vcap/sys/run/docker/docker.sock
- name: containerd-sock
  hostPath:
    path: /var/run/docker/containerd/docker--containerd.sock
- name: disk
  hostPath:
    path: /dev/disk
Configuring vRealize Operations Management Pack for Container Monitoring

You can configure vRealize Operations Management Pack for Container Monitoring on vRealize Operations Manager after you install the solution.

This chapter includes the following topics:

- Configure PKS Adapter
- Configure Kubernetes Adapter

### Configure PKS Adapter

Configure the PKS adapter to monitor the Kubernetes clusters created using PKS. The Kubernetes adapter instance is automatically created once you configure the PKS adapter. If you are deploying the Kubernetes cluster through PKS, do not configure the Kubernetes adapter instance.

#### Prerequisites

- Verify that you have installed vRealize Operations Management Pack for Container Monitoring.
- Verify that the PKS API Hostname (FQDN) is accessible and resolvable.
- Verify that the Kubernetes Clusters available in the PKS Environment have the cAdvisor DaemonSet configured on port 31194.

#### Procedure

1. From the main menu of vRealize Operations Manager, click Administration, and then in the left pane, click Solutions.
2. From the Solutions list, select VMware vRealize Operations Management Pack for Container Monitoring.
3. Click the Configure icon to edit an object.
4. Select PKS Adapter from the Adapter list and configure the adapter instance.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Name</td>
<td>Enter the display name of the adapter.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) Enter a description for the instance.</td>
</tr>
</tbody>
</table>
**Field Name** | **Action**
---|---
PKS API Hostname (FQDN) | Enter the API URL for the PKS instance.
PKS Instance Alias | Enter the alias name for the adapter instance.
Credential | Select the credential you want to use to sign on to the environment from the drop-down menu. To add new credentials, click the plus sign.
  - Credential Name. The name by which you are identifying the configured credentials.
  - PKS Username. The user name to access the PKS API.
  - PKS UAA Management Admin Client's secret. The PKS UAA Management Admin client secret.
Provide proxy details in the following fields if accessing the PKS API requires proxy authentication.
  - Proxy HostName
  - Proxy Port
  - Proxy Username
  - Proxy Password
Test Connection | Click **Test Connection** to validate the connection.
Advanced Settings | Use **Advance Setting** to define the following:
  - Collectors/Groups. Select the collector or collector group on which you want to run the adapter instance.
    This option is set to the optimal collector by default.
  - Auto Configure Kubernetes Adapter Instance. Select the **Enabled** option to discover the Kubernetes cluster in a PKS instance and create Kubernetes adapter instances automatically. Select the **Disabled** option to manually create the Kubernetes adapter instance.
  - Auto-accept Kubernetes Cluster SSL Certificate. Select the **Enabled** option to accept the untrusted certificates presented by the K8s adapter instances by default. Select the **Disabled** option to manually accept the untrusted certificates for the auto-configured K8s adapter instances.

5 Click **Save Settings**.

**Note** By default, the PKS Adapter instance auto-discovers the Kubernetes clusters available in the PKS Environment. It creates an appropriate Kubernetes Cluster Resource and a Kubernetes Adapter instance against each cluster.

### Configure Kubernetes Adapter

Configure the Kubernetes adapter only when PKS is not used to deploy the K8s Clusters.

**Prerequisites**

Verify that you have installed vRealize Operations Management Pack for Container Monitoring.
Procedure

1. From the main menu of vRealize Operations Manager, click **Administration**, and then in the left pane, click **Solutions**.

2. From the Solutions list, select **VMware vRealize Operations Management Pack for Container Monitoring**.

3. Click the **Configure** icon to edit an object.

4. Enter the display name of the adapter.

5. Enter the http URL of the Kubernetes master node in the **Master URL** text box.

6. Select either **Kubelet** or **DaemonSet** as the cAdvisor Service. You can select a cAdvisor service running inside the Kubelet or the one deployed externally as a DaemonSet.

   **Note** By default, some Kubernetes deployments might have the cAdvisor service disabled on Kubelet. In such a situation, the cAdvisor service must be enabled on Kubelet or a standalone cAdvisor service must be deployed as a DaemonSet.

7. Enter the port number if cAdvisor is running as a **DaemonSet**.

8. Enter the **Credential** details of the Master URL.
   
   a. Click the **Add New** icon.
   
   b. Select the authentication to connect to the Kubernetes API Server. vRealize Operations Management Pack for Container Monitoring currently supports basic, client certificate, and token authentication.

   **Table 3-1. Authentication Types**

<table>
<thead>
<tr>
<th>Authentication</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Auth</td>
<td>Uses HTTP basic authentication to authenticate API requests through authentication plugins.</td>
</tr>
<tr>
<td>Client Certification Auth</td>
<td>Uses client certificates to authenticate API requests through authentication plugins.</td>
</tr>
<tr>
<td>Token Auth</td>
<td>Uses bearer tokens to authenticate API requests through authentication plugins.</td>
</tr>
</tbody>
</table>

   For more information, see **Kubernetes Authentication**.

9. Under **Advanced Settings**
   
   a. Select the collector that is used to manage the adapter processes.
   
   b. If the Kubernetes cluster is running on vCenter Server and the same server is monitored by the vCenter Adapter instance, you can view a link from the Kubernetes node to the vSphere Virtual Machine. To view the link, enter the IP address of the vCenter Server instance.
   
   c. If you want to monitor Java Process, then enable this option.
d  If you want to delete the non-existent objects for a defined period, then select the time frame from the drop-down menu.

**Note**  The object deletion schedule is applicable to the Container Monitoring management pack only, and is over and above the global setting object deletion policy.

e  Click *Save Settings*.

10  Click *Close*. 
Dashboards in vRealize Operations Management Pack for Container Monitoring

You can use the dashboards to view and troubleshoot objects in your Kubernetes cluster eco-system that are monitored by vRealize Operations Management Pack for Container Monitoring.

Access Dashboards

Procedure

1. To access the dashboards, from the main menu of VMware vRealize Operations Manager, click **Dashboards**.
2. From the dashboard list, select **Kubernetes Overview**.

**Kubernetes Overview**

The overview dashboard provides an overall representation of Kubernetes environment, nodes, pods, and containers. The overview provides information of the overall health status of clusters, nodes, and pods with their respective historical trend and metric chart.

**Kubernetes Overview - Environment**

The Kubernetes overview environment widget provides an overall view of Kubernetes adapter instances, its associated objects information, alerts, and health status of objects.
Figure 4-1. Kubernetes Overview - Environment

<table>
<thead>
<tr>
<th>Widget Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search for a Kubernetes Cluster</td>
<td>This widget displays only Kubernetes instances but not all objects types. You can retrieve the total metrics from the instances that are listed under this widget.</td>
</tr>
<tr>
<td>Summary of the Selected Cluster</td>
<td>This widget displays the total number of nodes, namespaces, pods, containers and services within the Kubernetes cluster.</td>
</tr>
<tr>
<td>Any Alerts on the Nodes, Namespaces, Pods or Containers</td>
<td>This widget displays all the immediate and critical alerts within a cluster of nodes, namespaces, pods or containers. When you select a object type from the <strong>Search for a Kubernetes Cluster</strong> widget, the corresponding alerts that are only immediate and critical gets populated.</td>
</tr>
<tr>
<td>Are the cluster members healthy?</td>
<td>This widget provides a hierarchical view of object relationship of a Kubernetes cluster.</td>
</tr>
</tbody>
</table>

**Note** The **Total Objects** column in the **Search for a Kubernetes Cluster** widget does not match with the sum of the objects in the **Summary of the Selected Cluster** widget. This is because the value of the total objects includes the total count of replica sets and Java container processes.

**Kubernetes Overview - Nodes**

The Kubernetes overview nodes widget provides detailed set of information of nodes, node properties, health status, metrics, and hierarchical representation of pod relationship.
**Figure 4-2. Kubernetes Overview - Nodes**

<table>
<thead>
<tr>
<th>Widget Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 5 Least Healthy Nodes in the Selected Cluster</td>
<td>This widget displays the top 5 least healthy nodes in the selected cluster.</td>
</tr>
<tr>
<td>Node Properties</td>
<td>This widget displays respective node properties of the node that is selected in the Top 5 Least Healthy Nodes in the <strong>Selected Cluster</strong> widget.</td>
</tr>
<tr>
<td>Pods running on this Node</td>
<td>This widget provides a hierarchical view of pods and its relationship on a selected cluster.</td>
</tr>
<tr>
<td>How is the trend of Key Node Metrics</td>
<td>This widget displays the important metrics key metrics and its usage trend.</td>
</tr>
<tr>
<td>Pick another Metric or Property if needed</td>
<td>This widget lists all the metrics and properties for a selected node. This widget populates metrics and property of a node when it is selected from the Top 5 Least Healthy Nodes in the <strong>Selected Cluster</strong> widget.</td>
</tr>
<tr>
<td>Node Metric Chart</td>
<td>This widget provides a chart with the metric information of a metric or a property that is selected from the previous widget.</td>
</tr>
</tbody>
</table>

**Kubernetes Overview - Pods and Container**

The Kubernetes Overview pods and container widget provides detailed set of information of pods health status, hierarchical representation of pod relationship, metrics and so on.
**Figure 4-3. Kubernetes Overview - Pods and Containers**

<table>
<thead>
<tr>
<th>Widget Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 25 Least Healthy Pods in the Selected Cluster</td>
<td>This widget displays the Top 5 least healthy pods in the selected cluster.</td>
</tr>
<tr>
<td>How is the Pod associated with other components</td>
<td>This widget provides a hierarchical view of pods and its relationship with other components on a selected cluster.</td>
</tr>
<tr>
<td>How is the trend of Key Pod metrics?</td>
<td>This widget displays the important key metrics and its usage trend.</td>
</tr>
<tr>
<td>Pick any Metric from the selected component</td>
<td>This widget lists all the metrics and properties for a selected pod. This widget populates metrics and property of a pod when it is selected from the <strong>How is this Pod Associated with other components</strong> widget.</td>
</tr>
<tr>
<td>Metric Chart</td>
<td>This widget provides a chart with the metric information of a metric or a property that is selected from the previous widget.</td>
</tr>
</tbody>
</table>
The Kubernetes adapter instance collects metrics for the objects that are available in the kubernetes container.

### Table 5-1. Metrics in vRealize Operations Management Pack for Container Monitoring

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Group</th>
<th>Name</th>
<th>Unit</th>
<th>Type</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes Minion</td>
<td>CPU</td>
<td>Usage</td>
<td>System</td>
<td>cores</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>CPU</td>
<td>Usage</td>
<td>Total</td>
<td>cores</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>CPU</td>
<td>Usage</td>
<td>User</td>
<td>cores</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Filesystem</td>
<td>Free</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Filesystem</td>
<td>Total</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Filesystem</td>
<td>Used</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Memory</td>
<td>Cache</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Memory</td>
<td>Usage</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Memory</td>
<td>Working Set</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Read</td>
<td>KB</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Read Dropped</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Read Errors</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Read Packets</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Write</td>
<td>KB</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Write Dropped</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Write Errors</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Network</td>
<td>Interface</td>
<td>Write Packets</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Capacity</td>
<td>Memory</td>
<td>KB</td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Capacity</td>
<td>No of CPUs</td>
<td>Count</td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Capacity</td>
<td>No of Pods</td>
<td>Count</td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Object Type</td>
<td>Group</td>
<td>Name</td>
<td>Unit</td>
<td>Type</td>
<td>Remarks</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>---------------------</td>
<td>---------------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Node Info</td>
<td>Container Runtime</td>
<td></td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Node Info</td>
<td>Kubelet Version</td>
<td></td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Node Info</td>
<td>OS Image</td>
<td></td>
<td>Property</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Allocatable</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Minion</td>
<td>Allocatable</td>
<td>Memory</td>
<td>GB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Service</td>
<td>Summary</td>
<td>Availability</td>
<td></td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Summary</td>
<td>Availability</td>
<td></td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Summary</td>
<td>No of Containers</td>
<td>Count</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Node Limits</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Node Limits</td>
<td>Memory</td>
<td>GB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Resource Limits</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Resource Limits</td>
<td>Memory</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Pod-Minion Limit Factor</td>
<td>CPU</td>
<td></td>
<td>Metric</td>
<td>Multiplier value used to find out how much %age of Node's CPU is used by a Pod</td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Pod-Minion Limit Factor</td>
<td>Memory</td>
<td></td>
<td>Metric</td>
<td>Multiplier value used to find out how much %age of Node's Memory is used by a Pod</td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Parent Node Name</td>
<td></td>
<td></td>
<td>Property</td>
<td>Name of a pod’s parent node</td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Labels</td>
<td></td>
<td></td>
<td>Property</td>
<td>List of labels added to a pod which are dynamic</td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Resource Requests</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Kubernetes Pod</td>
<td>Resource Requests</td>
<td>Memory</td>
<td>KB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU Usage</td>
<td>System</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU Usage</td>
<td>Total</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU Usage</td>
<td>User</td>
<td>cores</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU</td>
<td>CPU contention</td>
<td>%</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Object Type</td>
<td>Group</td>
<td>Name</td>
<td>Unit</td>
<td>Type</td>
<td>Remarks</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>-----------------------</td>
<td>------</td>
<td>--------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Container</td>
<td>CPU</td>
<td>Demand</td>
<td>Mhz</td>
<td>Metric</td>
<td>Unable to get these metrics from Cadvisor.</td>
</tr>
<tr>
<td>Container</td>
<td>CPU</td>
<td>I/O Wait</td>
<td>%</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU</td>
<td>Swap wait</td>
<td>%</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>CPU</td>
<td>Co-stop</td>
<td>%</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Disk</td>
<td>Aggregate all instances</td>
<td>Read Latency</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Filesystem</td>
<td>Base Usage</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Filesystem</td>
<td>Total Usage</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Memory</td>
<td>Cache</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Memory</td>
<td>Rss</td>
<td>MB</td>
<td>Metric</td>
<td>Resident Set Size --&gt; Resident Memory</td>
</tr>
<tr>
<td>Container</td>
<td>Memory</td>
<td>Usage</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Memory</td>
<td>Working Set</td>
<td>MB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Memory</td>
<td>Contention</td>
<td>%</td>
<td>Metric</td>
<td>Memory usage % added for those containers that have limits specified</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Read</td>
<td>KB</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Read Dropped</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Read Errors</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Read Packets</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Write</td>
<td>KB</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Write Dropped</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Write Errors</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>Interface</td>
<td>Write Packets</td>
<td>Count</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Network</td>
<td>I/O</td>
<td>Workload</td>
<td>%</td>
<td>Metric</td>
</tr>
<tr>
<td>Container</td>
<td>Disk IO</td>
<td>Async</td>
<td>KB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Disk IO</td>
<td>Sync</td>
<td>KB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Disk IO</td>
<td>Read</td>
<td>KB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Disk IO</td>
<td>Write</td>
<td>KB</td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Summary</td>
<td>Restart Count</td>
<td></td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Summary</td>
<td>Availability</td>
<td></td>
<td>Metric</td>
<td></td>
</tr>
<tr>
<td>Container</td>
<td>Node Limits</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td>CPU limit set for the node on which a container is running</td>
</tr>
<tr>
<td>Container</td>
<td>Node Limits</td>
<td>Memory</td>
<td>GB</td>
<td>Metric</td>
<td>Memory limit set for the node on which a container is running</td>
</tr>
<tr>
<td>Container</td>
<td>Resource Limits</td>
<td>CPU</td>
<td>cores</td>
<td>Metric</td>
<td>CPU limit of a container set by the admin - 0 if not set</td>
</tr>
</tbody>
</table>
This chapter includes the following topics:

- **Super Metrics**

**Super Metrics**

The Kubernetes adapter instance also collects super metrics for the objects that are available in the Kubernetes container.

<table>
<thead>
<tr>
<th>Super Metric</th>
<th>Applied on Object</th>
<th>Enabled by Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory Usage %</td>
<td>Pod</td>
<td>Yes</td>
</tr>
<tr>
<td>Memory Usage (MB)</td>
<td>Pod</td>
<td>No</td>
</tr>
<tr>
<td>CPU Usage %</td>
<td>Pod</td>
<td>Yes</td>
</tr>
<tr>
<td>CPU Usage (cores)</td>
<td>Pod</td>
<td>No</td>
</tr>
<tr>
<td>Memory Requests</td>
<td>Node</td>
<td>Yes</td>
</tr>
<tr>
<td>CPU Requests</td>
<td>Node</td>
<td>Yes</td>
</tr>
<tr>
<td>Max Pod Memory Usage %</td>
<td>Namespace, Service</td>
<td>Yes</td>
</tr>
<tr>
<td>Max Pod CPU Usage %</td>
<td>Namespace, Service</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The Kubernetes objects raise alerts and alert symptom for instances that are available in the cluster.

### Table 6-1. Alerts in vRealize Operations Management Pack for Container Monitoring

<table>
<thead>
<tr>
<th>Alert Definition</th>
<th>Symptoms</th>
<th>Severity</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container CPU limit is set to unlimited</td>
<td>Container CPU limit is not defined</td>
<td>Info</td>
<td>A container running without CPU limit may claim all of Node's resources. Modify your Pod configuration with a CPU limit on the affected container. A quick glimpse at the CPU usage trend can help you set the limit.</td>
</tr>
</tbody>
</table>
| Container CPU usage is high | Container CPU usage is higher than 90%  
Container CPU usage is higher than 80%  
Container CPU usage is higher than 70% | Critical  
Immediate  
Warning | Consider the option of increasing CPU limit on the affected container if Node's resources permit. Else, you may have to add a new Node to the cluster to ease out the CPU crunch. |
| Container Memory limit is set to unlimited | Container Memory limit is not defined | Info | A container running without Memory limit may claim all of Node's resources. Modify your Pod configuration with a Memory limit on the affected container. A quick glimpse at the Memory usage trend can help you set the limit. |
| Container is not available | Container is not available | Immediate | Redeploy the Pod and make sure it goes to Ready state |
| Container Memory usage is high | Container Process CPU usage is higher than 90%  
Container Process CPU usage is higher than 80%  
Container Process CPU usage is higher than 70% | Critical  
Immediate  
Warning | Consider the option of increasing Memory limit on the affected container if Node's resources permit. Else, you may have to add a new Node to the cluster to ease out the Memory crunch. |
<table>
<thead>
<tr>
<th>Alert Definition</th>
<th>Symptoms</th>
<th>Severity</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Container Process has high Memory Usage</td>
<td>Container Process Memory usage is higher than 90% Container Process Memory usage is higher than 80% Container Process Memory usage is higher than 70%</td>
<td>Critical Immediate Warning</td>
<td>Consider increasing CPU limit of the container.</td>
</tr>
<tr>
<td>Container Process has high Memory Usage</td>
<td>Node Process Memory usage is higher than 90% Node Process Memory usage is higher than 80% Node Process Memory usage is higher than 70%</td>
<td>Critical Immediate Warning</td>
<td>Consider increasing Memory limit of the container.</td>
</tr>
<tr>
<td>Master Node is not available</td>
<td>Master Node is not available</td>
<td>Immediate</td>
<td>Ensure that the Master Node is reachable and API server is up and running.</td>
</tr>
<tr>
<td>Namespace is not Available</td>
<td>Namespace is not Available</td>
<td>Immediate</td>
<td>Check if the namespace has been deleted</td>
</tr>
<tr>
<td>Node has high CPU Usage</td>
<td>Node CPU usage is higher than 90% Node CPU Memory usage is higher than 80% Node CPU Memory usage is higher than 70% Node CPU Memory usage is higher than 60%</td>
<td>Critical Immediate Warning Info</td>
<td>Consider increasing CPU resource of the Node OR add a new Node to the cluster</td>
</tr>
<tr>
<td>Node is not available</td>
<td>Node is not available</td>
<td>Immediate</td>
<td>Verify if the Node is reachable and in Ready state</td>
</tr>
<tr>
<td>Node has high Memory Usage</td>
<td>Node Memory usage is higher than 90% NodeMemory usage is higher than 80% NodeMemory usage is higher than 70% NodeMemory usage is higher than 60%</td>
<td>Critical Immediate Warning Info</td>
<td>Consider increasing Memory resource of the Node OR add a new new Node to the cluster</td>
</tr>
<tr>
<td>Alert Definition</td>
<td>Symptoms</td>
<td>Severity</td>
<td>Recommendation</td>
</tr>
<tr>
<td>------------------</td>
<td>----------</td>
<td>----------</td>
<td>----------------</td>
</tr>
</tbody>
</table>
| One of the Pods has highest CPU usage on Namespace | Pod with highest utilization on namespace has CPU usage higher than 90%  
Pod with highest utilization on namespace has CPU usage higher than 80%  
Pod with highest utilization on namespace has CPU usage higher than 70%  
Descendant pod object (OR Operation)  
Pod memory usage is higher than 70%  
Pod memory usage is higher than 80%  
Pod memory usage is higher than 90% | Critical  
Immediate  
Warning  
Info | Consider modifying the affected Pod configurations to increase CPU limits |
| One of the pods has high Memory usage on Namespace | Pod with highest utilization on namespace has memory usage higher than 90%  
Pod with highest utilization on namespace has memory usage higher than 80%  
Pod with highest utilization on namespace has memory usage higher than 70%  
Descendant pod object (OR Operation)  
Pod CPU Usage is higher than 70%  
Pod CPU Usage is higher than 80%  
Pod CPU Usage is higher than 90% | Critical  
Immediate  
Warning  
Info | Consider modifying the affected Pod configurations to increase Memory limits |
<table>
<thead>
<tr>
<th>Alert Definition</th>
<th>Symptoms</th>
<th>Severity</th>
<th>Recommendation</th>
</tr>
</thead>
</table>
| One of the pods has high CPU usage on Service | Pod with highest utilization on Service has CPU usage higher than 70%  
Pod with highest utilization on Service has CPU usage higher than 80%  
Pod with highest utilization on Service has CPU usage higher than 90%  
Descendant pod object (OR Operation)  
Pod CPU Usage is higher than 70%  
Pod CPU Usage is higher than 80%  
Pod CPU Usage is higher than 90% | Critical  
Immediate  
Warning | Consider modifying the affected Pod configurations to increase CPU limits |
| One of the pods has high Memory usage on Service | Pod with highest utilization on Service has memory usage higher than 70%  
Pod with highest utilization on Service has memory usage higher than 80%  
Pod with highest utilization on Service has memory usage higher than 90%  
Descendant pod object (OR Operation)  
Pod memory usage is higher than 70%  
Pod memory usage is higher than 80%  
Pod memory usage is higher than 90% | Critical  
Immediate  
Warning | Consider modifying the affected Pod configurations to increase Memory limits |
| Pod has high CPU Usage            | Pod CPU Usage is higher than 90%  
Pod CPU Usage is higher than 80%  
Pod CPU Usage is higher than 70%  
Pod CPU Usage is higher than 60% | Critical  
Immediate  
Warning | Go through the individual usage of the affected Pod’s containers and balance their CPU limits. |
<table>
<thead>
<tr>
<th>Alert Definition</th>
<th>Symptoms</th>
<th>Severity</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pod has high Memory Usage</td>
<td>Pod memory usage is higher than 90%</td>
<td>Critical</td>
<td>Go through the individual usage of the affected Pod's containers and balance their Memory limits.</td>
</tr>
<tr>
<td></td>
<td>Pod memory usage is higher than 80%</td>
<td>Immediate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pod memory usage is higher than 70%</td>
<td>Warning</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pod memory usage is higher than 60%</td>
<td>Info</td>
<td></td>
</tr>
<tr>
<td>Pod is not available</td>
<td>Pod is not available</td>
<td>Critical</td>
<td>Redeploy the Pod and make sure it goes to Ready state</td>
</tr>
<tr>
<td>ReplicaSet is not available</td>
<td>ReplicaSet is not available</td>
<td>Immediate</td>
<td>Make sure that the Replica Set is present.</td>
</tr>
<tr>
<td>Service is not available</td>
<td>Service is not available</td>
<td>Immediate</td>
<td>Make sure that the Service is present.</td>
</tr>
<tr>
<td>Sum of Resource Requests of Pods exceed Node Capacity</td>
<td>CPU Requests greater than node capacity</td>
<td>Critical</td>
<td>Minimum CPU/Memory resources required to run the Pods of the affected node has exceeded Node capacity. Consider increasing Node resources OR add more Nodes to distribute the workload.</td>
</tr>
<tr>
<td></td>
<td>Memory Requests greater than node capacity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

VMware vRealize Operations Management Pack for Container Monitoring
A report is a snapshot of views. The reports provide a view of Kubernetes adapter instance objects in XSL and PDF format. The report in the Container Monitoring management pack is called as Kubernetes Adapter Instance Summary.

The report is based on the following views:

- Kubernetes adapter instance objects
- Containers with no memory limit
- Container with no CPU limit