



# Red Hat Advanced Cluster Management for Kubernetes 2.10

## Health metrics

Health metrics



# Red Hat Advanced Cluster Management for Kubernetes 2.10 Health metrics

---

Health metrics

## Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

XFS<sup>®</sup> is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack<sup>®</sup> Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

Read more to learn about metrics and monitoring across your clusters and components.

---

## Table of Contents

<b>CHAPTER 1. HEALTH METRICS</b> .....	<b>3</b>
1.1. USING THE METRICS SERVICE	3
1.1.1. Accessing the hub cluster metrics service	3
1.1.2. Scraping with Prometheus	3
1.1.2.1. Scraping the hub cluster	3
1.1.2.2. Scraping the managed cluster	5
1.1.3. Scraping the standalone cluster	6



# CHAPTER 1. HEALTH METRICS

You can use metrics to monitor the health of your components.

See the following documentation:

- [Using the metrics service](#)

## 1.1. USING THE METRICS SERVICE

You can use metrics to monitor component health across Red Hat Advanced Cluster Management for Kubernetes. Many custom metrics are documented in the [Metrics chronicle overview](#).

### 1.1.1. Accessing the hub cluster metrics service

To view the collected metrics, you must expose the metrics service on the hub cluster. If your metrics are already exposed in the Grafana dashboard, this procedure is optional.

From the OpenShift Container Platform console, find the metrics service. Click **Observe > Metrics**.

If you do not see the metrics in the Grafana dashboard, Grafana Explorer, or in the OpenShift Container Platform console, Prometheus might not be configured to scrape metrics. Continue with *Scraping with Prometheus* to expose your metrics.

### 1.1.2. Scraping with Prometheus

You can use Prometheus to expose metrics that are not exposed from the product console. See the procedures for both the hub and managed cluster metrics.

#### 1.1.2.1. Scraping the hub cluster

See the following procedure to expose metrics for the hub cluster. These files are within the **openshift-monitoring** namespace:

1. Create a **ServiceMonitor** for collecting services and exposing metrics. See the following YAML example:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: hub-subscription-metrics
  namespace: openshift-monitoring
spec:
  endpoints:
    - port: metrics
  namespaceSelector:
    matchNames:
      - open-cluster-management
  selector:
    matchLabels:
      app: hub-subscription-metrics
```

2. Run the following command to apply the file:

```
oc apply -f
```

3. Create a **Role** for setting the permissions for monitoring. See the following YAML file:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: prometheus-k8s-monitoring
  namespace: open-cluster-management
rules:
- apiGroups:
  - ""
  resources:
  - services
  - endpoints
  - pods
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - extensions
  resources:
  - ingresses
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - networking.k8s.io
  resources:
  - ingresses
  verbs:
  - get
  - list
  - watch
```

4. Run the following command to apply the file:

```
oc apply -f
```

5. Create a **RoleBinding** for binding the role to the Prometheus monitoring **ServiceAccount**, as it is in the following example:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: prometheus-k8s-monitoring-binding
  namespace: open-cluster-management
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: prometheus-k8s-monitoring
subjects:
```



```
- kind: ServiceAccount
  name: prometheus-k8s
  namespace: monitoring
```

6. Run the following command to apply the file:

```
oc apply -f
```

7. To verify, run the following query in the dashboard to find metrics that are reported by the Subscription Operator Metrics Service:

```
{service="hub-subscription-metrics"}
```

### 1.1.2.2. Scraping the managed cluster

See the following procedure to expose metrics for managed clusters. These files are within the **openshift-monitoring** namespace:

1. Create a **ServiceMonitor** for collecting services exposing metrics. See the following YAML file example:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: mc-subscription-metrics
  namespace: openshift-monitoring
spec:
  endpoints:
    - port: metrics
  namespaceSelector:
    matchNames:
      - open-cluster-management-agent-addon
  selector:
    matchLabels:
      app: mc-subscription-metrics
```

2. Run the following command to apply your file:

```
oc apply -f
```

3. Create a **Role** for setting the permissions for monitoring. See the following YAML file example:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: prometheus-k8s-monitoring
  namespace: open-cluster-management-agent-addon
rules:
  - apiGroups:
      - ""
    resources:
      - services
      - endpoints
      - pods
```

```

verbs:
- get
- list
- watch
- apiGroups:
- extensions
resources:
- ingresses
verbs:
- get
- list
- watch
- apiGroups:
- networking.k8s.io
resources:
- ingresses
verbs:
- get
- list
- watch

```

4. Run the following command to apply the file:

```
oc apply -f
```

5. Create a **RoleBinding** for binding the **Role** to the Prometheus monitoring **ServiceAccount**:

```

apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: prometheus-k8s-monitoring-binding
  namespace: open-cluster-management-agent-addon
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: prometheus-k8s-monitoring
subjects:
- kind: ServiceAccount
  name: prometheus-k8s
  namespace: monitoring

```

6. Run the following command to apply the file:

```
oc apply -f
```

7. Verify in the **Prometheus** dashboard by running the following query to find metrics that are reported by the Subscription Operator Metrics Service:

```
{service="mc-subscription-metrics"}
```

### 1.1.3. Scraping the standalone cluster

1. Create a **ServiceMonitor** for collecting services exposing metrics:

```

apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: standalone-subscription-metrics
  namespace: openshift-monitoring
spec:
  endpoints:
  - port: metrics
  namespaceSelector:
    matchNames:
    - open-cluster-management
  selector:
    matchLabels:
      app: standalone-subscription-metrics

```

2. Create a **Role** for setting the permissions for monitoring:

```
oc apply -f
```

```

apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: prometheus-k8s-monitoring
  namespace: open-cluster-management
rules:
- apiGroups:
  - ""
  resources:
  - services
  - endpoints
  - pods
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - extensions
  resources:
  - ingresses
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - networking.k8s.io
  resources:
  - ingresses
  verbs:
  - get
  - list
  - watch

```

3. Create a **RoleBinding** for binding the **Role** to the Prometheus monitoring **ServiceAccount**. See the following YAML file example:

■

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: prometheus-k8s-monitoring-binding
  namespace: open-cluster-management
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: prometheus-k8s-monitoring
subjects:
- kind: ServiceAccount
  name: prometheus-k8s
  namespace: monitoring
```

4. Run the following command to apply the file:

```
oc apply -f
```

5. Verify in the **Prometheus** dashboard by running the following query to find metrics that are reported by the Subscription Operator Metrics Service:

```
{service="standalone-subscription-metrics"}
```