Red Hat OpenShift GitOps 1.10

Observability

Using observability features to view Argo CD logs and monitor the performance and health of Argo CD and application resources
Using observability features to view Argo CD logs and monitor the performance and health of Argo CD and application resources
Abstract

This document details how to use OpenShift Logging with OpenShift GitOps and monitor the performance of Argo CD instances, application health status, and Argo CD custom resource workloads.
# Table of Contents

## CHAPTER 1. LOGGING

1.1. VIEWING ARGO CD LOGS ................................................................. 3  
   1.1.1. Storing and retrieving Argo CD logs 3  
   1.1.2. Additional resources 3

## CHAPTER 2. MONITORING

2.1. MONITORING ARGO CD INSTANCES .............................................. 5  
  2.1.1. Prerequisites 5  
  2.1.2. Monitoring Argo CD health using Prometheus metrics 5  
2.2. MONITORING HEALTH INFORMATION FOR APPLICATION RESOURCES AND DEPLOYMENTS 5  
  2.2.1. Settings for environment labels and annotations 6  
     Environment labels 6  
     Environment annotations 6  
  2.2.2. Checking health information 7  
2.3. MONITORING ARGO CD CUSTOM RESOURCE WORKLOADS 8  
  2.3.1. Prerequisites 8  
  2.3.2. Enabling Monitoring for Argo CD custom resource workloads 8  
  2.3.3. Disabling Monitoring for Argo CD custom resource workloads 9  
  2.3.4. Additional resources 10
CHAPTER 1. LOGGING

1.1. VIEWING ARGO CD LOGS

You can view the Argo CD logs with the logging subsystem for Red Hat OpenShift. The logging subsystem visualizes the logs on a Kibana dashboard. The OpenShift Logging Operator enables logging with Argo CD by default.

1.1.1. Storing and retrieving Argo CD logs

You can use the Kibana dashboard to store and retrieve Argo CD logs.

Prerequisites

- The Red Hat OpenShift GitOps Operator is installed in your cluster.
- The logging subsystem for Red Hat OpenShift is installed with default configuration in your cluster.

Procedure

1. In the OpenShift Container Platform web console, go to the menu → Observability → Logging to view the Kibana dashboard.

2. Create an index pattern.
   a. To display all the indices, define the index pattern as *, and click Next step.
   b. Select @timestamp for Time Filter field name.
   c. Click Create index pattern

3. In the navigation panel of the Kibana dashboard, click the Discover tab.

4. Create a filter to retrieve logs for Argo CD. The following steps create a filter that retrieves logs for all the pods in the openshift-gitops namespace:
   a. Click Add a filter +.
   b. Select the kubernetes.namespace_name field.
   c. Select the is operator.
   d. Select the openshift-gitops value.
   e. Click Save.

5. Optional: Add additional filters to narrow the search. For example, to retrieve logs for a particular pod, you can create another filter with kubernetes.pod_name as the field.

6. View the filtered Argo CD logs in the Kibana dashboard.

1.1.2. Additional resources
- Installing the logging subsystem for Red Hat OpenShift using the web console
CHAPTER 2. MONITORING

2.1. MONITORING ARGO CD INSTANCES

By default, the Red Hat OpenShift GitOps Operator automatically detects an installed Argo CD instance in your defined namespace, for example, openshift-gitops, and connects it to the monitoring stack of the cluster to provide alerts for out-of-sync applications.

2.1.1. Prerequisites

- You have access to the cluster with cluster-admin privileges.
- You have access to the OpenShift Container Platform web console.
- You have installed the Red Hat OpenShift GitOps Operator in your cluster.
- You have installed an Argo CD application in your defined namespace, for example, openshift-gitops.

2.1.2. Monitoring Argo CD health using Prometheus metrics

You can monitor the health status of an Argo CD application by running Prometheus metrics queries against it.

Procedure

1. In the Developer perspective of the web console, select the namespace where your Argo CD application is installed, and navigate to Observe → Metrics.

2. From the Select query drop-down list, select Custom query.

3. To check the health status of your Argo CD application, enter the Prometheus Query Language (PromQL) query similar to the following example in the Expression field:

   Example

   ```promql
   sum(argocd_app_info{dest_namespace=~"<your_defined_namespace>",health_status!=""})
   by (health_status)
   ```

   Replace the `<your_defined_namespace>` variable with the actual name of your defined namespace, for example openshift-gitops.

2.2. MONITORING HEALTH INFORMATION FOR APPLICATION RESOURCES AND DEPLOYMENTS

The Red Hat OpenShift GitOps Environments page in the Developer perspective of the OpenShift Container Platform web console shows a list of the successful deployments of the application environments, along with links to the revision for each deployment.

The Application environments page in the Developer perspective of the OpenShift Container Platform web console displays the health status of the application resources, such as routes, synchronization status, deployment configuration, and deployment history.
The environments pages in the Developer perspective of the OpenShift Container Platform web console are decoupled from the Red Hat OpenShift GitOps Application Manager command-line interface (CLI), kam. You do not have to use kam to generate Application Environment manifests for the environments to show up in the Developer perspective of the OpenShift Container Platform web console. You can use your own manifests, but the environments must still be represented by namespaces. In addition, specific labels and annotations are still needed.

2.2.1. Settings for environment labels and annotations

This section provides reference settings for environment labels and annotations required to display an environment application in the Environments page, in the Developer perspective of the Red Hat OpenShift GitOps web console.

Environment labels
The environment application manifest must contain labels.openshift.gitops/environment and destination.namespace fields. You must set identical values for the <environment_name> variable and the name of the environment application manifest.

Specification of the environment application manifest

```yaml
spec:
  labels:
    openshift.gitops/environment: <environment_name>
    destination:
      namespace: <environment_name>
# ...
```

Example of an environment application manifest

```yaml
apiVersion: argoproj.io/v1beta1
kind: Application
metadata:
  name: dev-env
  namespace: openshift-gitops
spec:
  labels:
    openshift.gitops/environment: dev-env
    destination:
      namespace: dev-env
# ...
```

The name of the environment application manifest. The value set is the same as the value of the <environment_name> variable.

Environment annotations
The environment namespace manifest must contain the annotations.app.openshift.io/vcs-uri and annotations.app.openshift.io/vcs-ref fields to specify the version controller code source of the application. You must set identical values for the <environment_name> variable and the name of the environment namespace manifest.

Specification of the environment namespace manifest

```yaml
apiVersion: v1
```
The name of the environment namespace manifest. The value set is the same as the value of the `<environment_name>` variable.

Example of an environment namespace manifest

```yaml
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    app.openshift.io/vcs-uri: https://example.com/<your_domain>/<your_gitops.git>
    app.openshift.io/vcs-ref: main
labels:
  argocd.argoproj.io/managed-by: openshift-gitops
name: dev-env
# ...
```

2.2.2. Checking health information

The Red Hat OpenShift GitOps Operator will install the GitOps backend service in the `openshift-gitops` namespace.

Prerequisites

- The Red Hat OpenShift GitOps Operator is installed from OperatorHub.
- Ensure that your applications are synchronized by Argo CD.

Procedure

1. Click **Environments** under the **Developer** perspective. The **Environments** page shows the list of applications along with their **Environment status**.

2. Hover over the icons under the **Environment status** column to see the synchronization status of all the environments.

3. Click the application name from the list to view the details of a specific application.

4. In the **Application environments** page, if the **Resources** section under the **Overview** tab displays icons, hover over the icons to get status details.
   - A broken heart indicates that resource issues have degraded the application’s performance.
   - A yellow yield sign indicates that resource issues have delayed data about the application’s health.

```yaml
kind: Namespace
metadata:
  annotations:
    app.openshift.io/vcs-uri: <application_source_url>
    app.openshift.io/vcs-ref: <branch_reference>
name: <environment_name>  
# ...
```
5. To view the deployment history of an application, click the Deployment History tab. The page includes details such as the Last deployment, Description (commit message), Environment, Author, and Revision.

2.3. MONITORING ARGO CD CUSTOM RESOURCE WORKLOADS

With Red Hat OpenShift GitOps, you can monitor the availability of Argo CD custom resource workloads for specific Argo CD instances. By monitoring Argo CD custom resource workloads, you have the latest information about the state of your Argo CD instances by enabling alerts for them. When the component workload pods such as application-controller, repo-server, or server of the corresponding Argo CD instance are unable to come up for certain reasons and there is a drift between the number of ready replicas and the number of desired replicas for a certain period of time, the Operator then triggers the alerts.

You can enable and disable the setting for monitoring Argo CD custom resource workloads.

2.3.1. Prerequisites

- You have access to the cluster as a user with the cluster-admin role.
- Red Hat OpenShift GitOps is installed in your cluster.
- The monitoring stack is configured in your cluster in the openshift-monitoring project. In addition, the Argo CD instance is in a namespace that you can monitor through Prometheus.
- The kube-state-metrics service is running in your cluster.
- Optional: If you are enabling monitoring for an Argo CD instance already present in a user-defined project, ensure that the monitoring is enabled for user-defined projects in your cluster.

NOTE

If you want to enable monitoring for an Argo CD instance in a namespace that is not watched by the default openshift-monitoring stack, for example, any namespace that does not start with openshift-*, then you must enable user workload monitoring in your cluster. This action enables the monitoring stack to pick up the created PrometheusRule.

2.3.2. Enabling Monitoring for Argo CD custom resource workloads

By default, the monitoring configuration for Argo CD custom resource workloads is set to false.

With Red Hat OpenShift GitOps, you can enable workload monitoring for specific Argo CD instances. As a result, the Operator creates a PrometheusRule object that contains alert rules for all the workloads managed by the specific Argo CD instances. These alert rules trigger the firing of an alert when the replica count of the corresponding component has drifted from the desired state for a certain amount of time. The Operator will not overwrite the changes made to the PrometheusRule object by the users.

Procedure

1. Set the .spec.monitoring.enabled field value to true on a given Argo CD instance:

Example Argo CD custom resource

```yaml
apiVersion: argoproj.io/v1beta1
```
2. Verify whether an alert rule is included in the PrometheusRule created by the Operator:

**Example alert rule**

```yaml
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: argocd-component-status-alert
namespace: openshift-gitops
spec:
groups:
  - name: ArgoCDComponentStatus
    rules:
      - alert: ApplicationSetControllerNotReady
        annotations:
          message: >-
            applicationSet controller deployment for Argo CD instance in namespace "default" is not running
          expr: >-
            kube_statefulset_status_replicas{statefulset="openshift-gitops-application-controller statefulset", namespace="openshift-gitops"} != kube_statefulset_status_replicas_ready{statefulset="openshift-gitops-application-controller statefulset", namespace="openshift-gitops"}
          for: 1m
        labels:
          severity: critical
```

1 Alert rule in the PrometheusRule that checks whether the workloads created by the Argo CD instances are running as expected.

2.3.3. Disabling Monitoring for Argo CD custom resource workloads

You can disable workload monitoring for specific Argo CD instances. Disabling workload monitoring deletes the created PrometheusRule.

**Procedure**

- Set the `.spec.monitoring.enabled` field value to `false` on a given Argo CD instance:

**Example Argo CD custom resource**
2.3.4. Additional resources

- Enabling monitoring for user-defined projects