

Logging, Monitoring, and Alerting

This section touches on several topics related to logging, monitoring, and alerting.

- [Telemetry data collection and display](#)
- [Log data collection and display](#)
- [Grafana™ dashboards](#) (Appliance mode only)
- [Configuring custom resources for data collection](#)
 - [Configuring and installing the Prometheus™ stack](#)
 - [Configuring and installing Loki™ \(Appliance Mode only\)](#)
 - [Configuring and installing Fluentd™ \(Appliance Mode only\)](#)

Telemetry Data Collection and Display

Telemetry streams are collected from several sources including:

- Kubernetes telemetry data (pod telemetry),
- KumoScale application telemetry, and
- Other applications running in the KumoScale software cluster.

See the section [Telemetry](#) for complete information on how telemetry data is collected and how it can be managed.

Log Data Collection and Display

Log data is collected from multiple sources such as system logs and KumoScale (Appliance mode only) applications using **Fluentd**, an open-source data collector. It is a cloud-native logging solution to unify data collection and consumption. It collects the following logs:

- kubernetes /var/log/containers/*.log.
- Syslog logs.
- /var/log/messages.
- ks-engine /data/ks/logs/*.log.

The log streams are stored in **Grafana Loki**, a multi-tenant log aggregation system created at Grafana Labs. The volume capacity for storing the data is a configurable parameter that may be changed by the user. Open-source Grafana dashboards are used to visualize log data. This is documented further in the [KumoScale Dashboard Guide](#).

The figure below illustrates the data flow and components in log collection and display.



Overview of Log Collection and Display

Grafana Dashboards (Appliance Mode only)

An example Grafana dashboard is included with KumoScale for Appliance mode to demonstrate the kinds of visualization that can be created from the Prometheus data feed.

To configure the Prometheus data feed for Grafana, see [Configuring Custom Resources for Data Collection](#).

Details on how to customize the dashboard layout is documented further in the [KumoScale Dashboard Guide](#).

Configuring Custom Resources for Data Collection

This section documents the custom resource files available in KumoScale to configure and create components used in data collection and display. All these files are available in `KumoScale_Operator/ks-install-operator/deploy/crds`. There are dependencies between them and they must be installed in this order:

1. [Prometheus Stack](#) (Appliance and Managed Mode)
2. [Loki](#) (Appliance Mode only)
3. [Fluentd](#) (Appliance Mode only)

To install a component:

```
kubect1 create -f <resource CR file>
```

To upgrade a component:

```
kubect1 apply -f <resource CR file>
```

To uninstall a component:

```
kubect1 delete -f <resource CR file>
```

Configuring and Installing the Prometheus Stack

The Prometheus Stack includes the following:

- **Prometheus Operator** provides Kubernetes native deployment and management of Prometheus and related monitoring components.
- **Prometheus Service** provides a web user interface (UI) that enables you to view simple graphs, Prometheus configurations and rules, and the state of the monitoring endpoints.
- **Alert Manager** handles alerts sent by client applications such as the Prometheus server.
- **Prometheus Node Exporter** reads system-level statistics about bare-metal nodes or virtual machines and exports them to Prometheus.
- **Kube State Metrics** is a simple service that listens to the Kubernetes API server and generates metrics about the state of the objects. It focuses on the health of the various objects, such as deployments, nodes and pods.
- **Grafana** supports querying Prometheus. It is an interactive visualization web application that provides charts, graphs, and alerts. (The initial Grafana credentials are: admin/ksAdmin and you will need to log into the UI after setting up the stack.)

The following services are sampled by the default ServiceMonitor configuration every 30 seconds:

- alertmanager
- grafana
- kubeApiServer
- kubelet
- kubeControllerManager
- coreDns
- kubeDns
- kubeEtdcd
- kubeScheduler
- kubeProxy
- kubeStateMetrics
- nodeExporter
- prometheusOperator
- prometheus (instance)

To configure the Prometheus stack, KumoScale software provides a **Prometheus Stack CRD**, a template of which is in **KumoScale_Operator/ks-install-operator/deploy/crds/prometheusstack..kumoscale.kioxia.com_v1_prometheusstack_cr.yaml**.

1. Make a copy of **prometheusstack..kumoscale.kioxia.com_v1_prometheusstack_cr.yaml** for customizing, (e.g., **myapp_prometheusstack_cr.yaml**)
2. Configure the Prometheus stack by specifying the appropriate parameter values in **myapp_prometheusstack_cr.yaml**.The table below shows the parameters supported in the Prometheus Stack CRD.
3. After specifying your custom values, install the Prometheus Stack with **kubect1 create**. In this case we use our example file myapp_prometheusstack_cr.yaml:

```
kubect1 create - f myapp_prometheusstack_cr.yaml
```

Notes:

(1) The initial Grafana credentials are: admin/ksAdmin. You should log into the UI after installing the CRD.

(2) Many of the services used for data collection cannot be configured in Highly Available (HA) mode so that data is replicated. Prometheus is one of the few services that does support replication using the storage class parameter **replicas**. We recommend that **replicas** be set as follows:

- When using one master node, use replicas= 1.
- In other deployments, use replicas= 3.

The applications will scale as you add nodes.

prometheus Stack Parameter	Description	Optional/Required
static_configs: targets	Target associated with the KumoScale Provisioner service expressed as an IP address or FQDN. If FQDN is used it must be on port 8443. For example, kumoscale-provisioner-service.kumo-services:8443.	Required

prometheus Stack Parameter	Description	Optional/Required
type nodePort	If a FQDN is specified for the target, then type must be equal to NodePort and nodePort must be 30190: type: NodePort nodePort: 30190	Required when target is a FQDN.
retention	The amount of time to retain metrics. Possible values are [0-9]+(ms s m h d w y). Default value is 1y.	Optional
replicas	The number of replicas for data collection. For single node clusters, replicas should be 1. In other cases, replicas should be 3. (see note above.) Default value is 3.	Optional
storageClassName	The storage class of the volume. Default value is kumoscale-local-storage. Keep in mind that if you specify a name, you must use that name throughout other operations.	Optional
storage	The size of storage for Prometheus service. Default value is 40Gi.	Optional
alertManager: enabled	Whether the alertManager is enabled. Possible values are true (enabled) or false (not enabled). Default value is true.	Optional
alertManager: retention	The amount of time to retain data. Possible values are [0-9]+(ms s m h). Default value is 8760h.	Optional
alertManager: storageClassName	The storage class of the volume. Default value is kumoscale-local-storage.	Optional
alertManager: storage	The size of storage for the alertManager service. Default value is 40Gi.	Optional
prometheus-node-exporter: enabled	Whether the Prometheus node exporter is enabled. Possible values are true (enabled) or false (not enabled). Default value is true.	Optional
kube-state-metrics	Whether the Kubernetes kube-state-metrics service is enabled. Possible values are true (enabled) or false (not enabled). Default value is true.	Optional
grafana: externalIPs	The IP address for the Grafana web interface. Required when target is a VIP.	Required when target is an IP address.
grafana: type nodePort	If a FQDN is specified for the target, then type must be equal to NodePort and nodePort must be 30191: type: NodePort nodePort: 30191	Required when target is a FQDN.
grafana: persistence.enabled	Enable grafana persistence for persistent password and data sources. Default value is true.	Optional
grafana: storageClassName	The storage class of the volume. Default value is kumoscale-local-storage.	Optional
grafana: storage	The volume size of storage for grafana service. Default value is 1Gi.	Optional

Uninstalling Prometheus

To uninstall Prometheus use kubectl delete as in:

```
kubectl delete -f myapp_prometheusstack_cr.yaml
```

Modifying Prometheus

To update Prometheus with new values use kubectl apply as in:

```
kubectl apply -f myapp_prometheusstack_cr.yaml
```

Configuring and Installing Loki (Appliance Mode only)

To configure Loki, KumoScale software provides a **Loki CRD**, a template of which is in **KumoScale_Operator/ks-install-operator/deploy/crds/loki.kumoscale.kioxia.com_v1__loki.yaml**.

1. Make a copy of **loki.kumoscale.kioxia.com_v1_loki.yaml** for customizing, (e.g. **myapp_loki_cr.yaml**)
2. Configure Loki by specifying the appropriate parameter values in **myapp_loki_cr.yaml**. Below are details on parameters used in configuration.

Loki Parameter Name	Description	Optional/Required
size	The size of the volume that saves the logs.	Optional

	Default value is 100Gi.	
storageClassName	The storage class of the volume. Default value is kumoscale-local-storage This has the protocol:Local and provisioningType:"thin".	Optional

3. Install Loki using kubectl create. Using our example CR file:

```
kubectl create -f myapp_loki_cr.yaml
```

Uninstalling Loki

To uninstall Loki use kubectl delete as in:

```
kubectl delete -f myapp_loki_cr.yaml
```

Modifying Loki

To update Loki with new values, use kubectl apply as in:

```
kubectl apply -f myapp_loki_cr.yaml
```

Configuring and Installing Fluentd (Appliance Mode only)

To configure Fluentd, KumoScale software provides a **Fluentd CRD**, a template of which is in **fluentd.kumoscale.kioxia.com_v1_fluentd.yaml**.

1. Make a copy of **fluentd.kumoscale.kioxia.com_v1_fluentd.yaml** for customizing, (e.g. **myapp_fluentd_cr.yaml**)
2. Configure Fluentd by specifying the appropriate parameter values in the Fluentd storage class file, myapp_ fluentd _cr.yaml . Below are details on parameters used in configuration.

Note: Fluentd does not support TCP.

fluentd Parameter Name	Description	Optional/ Required
clusterIP	The Cluster IP. To assign a cluster IP for a service such as Fluentd, you need to know the address range from which to select the IP. You can get the range by entering kubectl cluster-info dump grep -m 1 service-cluster-ip-range An example of output returned includes --service-cluster-ip-range=192.0.0.0/12	Required
name	Port name	Optional
Protocol	Syslog protocol	Optional
containerPort	Container port	Optional

3. Install Fluentd with kubectl create. Using our example above:

```
kubectl create -f fluentd.kumoscale.kioxia.com_v1_fluentd_cr.yam
```

Uninstalling Fluentd

To uninstall Fluentd use kubectl delete as in:

```
kubectl delete -f myapp_fluentd_cr.yaml
```

Modifying Fluentd

To update Fluentd with new values use kubectl apply as in:

```
kubectl apply -f myapp_fluentd_cr.yaml
```

Next: [Volume Management](#)

