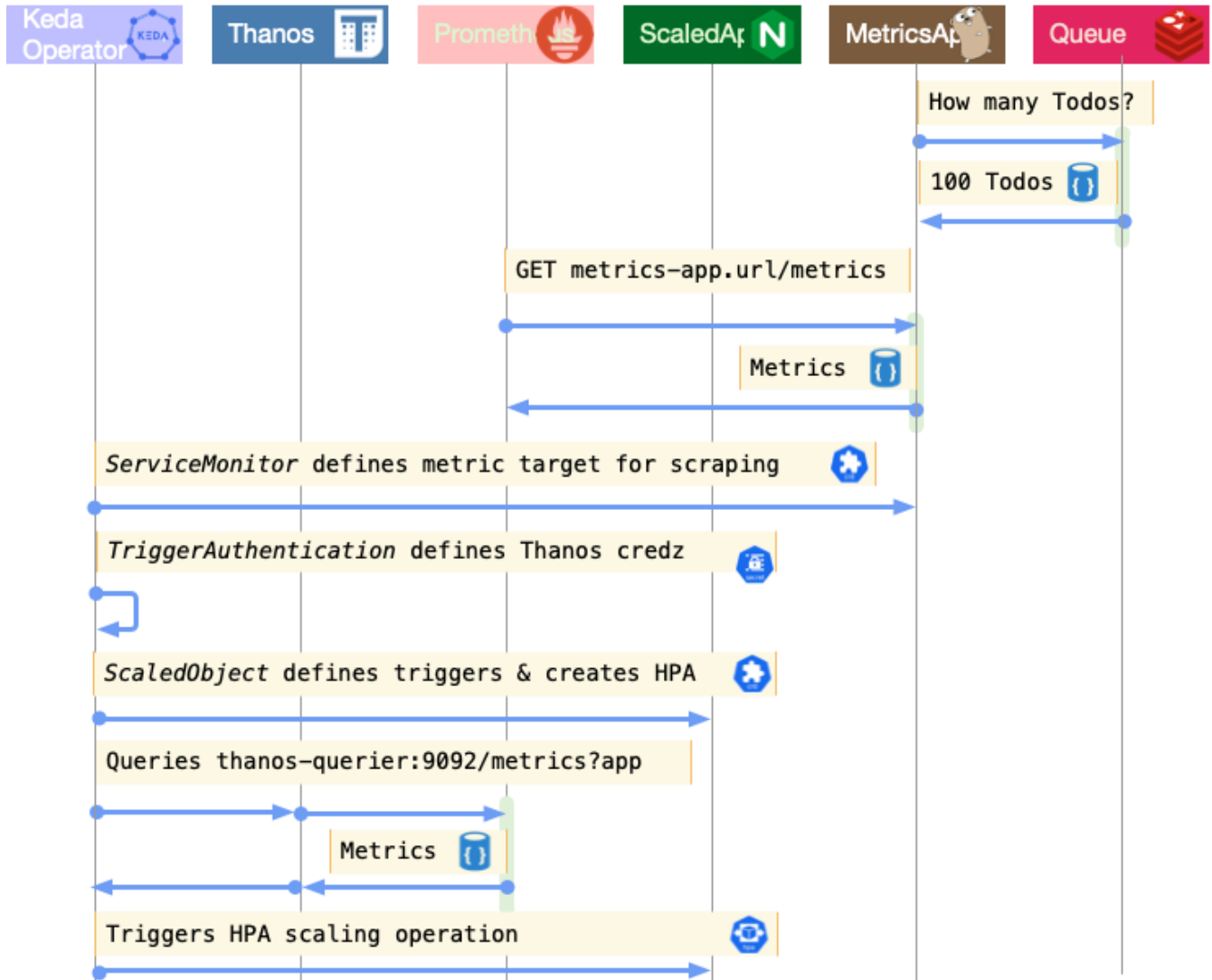


Demo Using Custom Prometheus Metrics for OpenShift Autoscaling with KEDA



Updated: 2022-08-15

Outline of Steps

- Enable user workload monitoring
- Install "OpenShift Custom Metrics Autoscaler" operator and deploy KedaController
- Deploy application with custom metrics target and associated ServiceMonitor
- Create Thanos serviceaccount, token, role, rolebinding
- Configure authentication from Keda to Thanos
- Deploy sample application to autoscale
- Define deployment to scale and the metrics to trigger scaling with ScaledResource

Docs and Refs

- [Enabling user workload monitoring](#)
- <https://docs.openshift.com/container-platform/4.11/nodes/pods/nodes-pods-autoscaling-custom.html>
- [Autoscaling for RGW in ODF via HPA using KEDA](#)
- <https://docs.openshift.com/container-platform/4.11/monitoring/managing-metrics.html>
- <https://github.com/rhobs/prometheus-example-app>
- <https://keda.sh/docs/2.8/scalers/prometheus/>
- [Enabling TLS in ServiceMonitor](#)

Prerequisites

Enable User Workload Monitoring

- [OpenShift user workload monitoring](#)

```
oc extract configmap/cluster-monitoring-config \
  -n openshift-monitoring --to=- \
  | yq eval '.enableUserWorkload = true' - > config.yaml
oc set data configmap/cluster-monitoring-config \
  --from-file=config.yaml -n openshift-monitoring
```

Granting non-admin users permission to monitor user-defined projects

Roles

- `monitoring-rules-view` grants read access to PrometheusRule custom resources for a project.
- `monitoring-rules-edit` grants create, modify, and deleting PrometheusRule custom resources for a project.
- `monitoring-edit` grants `monitoring-rules-edit` plus create new scrape targets for services or pods. With this role, you can also create, modify, and delete ServiceMonitor and PodMonitor resources.

Grant `monitoring-edit` role.

Deploy Custom Metrics Autoscaler Operator

Deploys the OpenShift [Custom Metric Autoscaler operator](#) which is downstream from [KEDA.sh](https://keda.sh).

```
oc apply -k operator
namespace/openshift-keda created
kedacontroller.keda.sh/keda created
operatorgroup.operators.coreos.com/openshift-keda created
subscription.operators.coreos.com/openshift-custom-metrics-autoscaler-operator
created
oc logs -f -n openshift-keda -l app=keda-operator
```

Deploy Custom Metric Producer

This application exists to expose a metrics endpoint that proffers custom metrics used for scaling a [second application](#).

This metric may be tangentially related to the load of the scaled app. For instance it may gather and report on a queue depth or work backlog. However, the current [app referenced here](#) does not.

```
oc apply -k custom-metric-app
namespace/keda-test created
service/prometheus-example-app created
deployment.apps/prometheus-example-app created
servicemonitor.monitoring.coreos.com/prometheus-example-monitor created
route.route.openshift.io/prometheus-example-app created
```

View Example metrics

```
ROUTE=$(oc get route prometheus-example-app -o jsonpath='{.spec.host}')
curl $ROUTE/metrics
# HELP version Version information about this binary
# TYPE version gauge
version{version="v0.4.1"} 1
# create some traffic stats then retry above
curl $ROUTE
```

TODO

TIP Create a new app with more pertinent metric example. Like returning the number of Todos from a work queue.

Deploy Scaled-App

This is a sample application that is scaled by virtue of custom metrics exported from [custom metric app](#).

```
oc apply -k scaled-app
namespace/keda-test unchanged
serviceaccount/thanos created
role.rbac.authorization.k8s.io/thanos-metrics-reader created
rolebinding.rbac.authorization.k8s.io/thanos-metrics-reader created
secret/thanos-token created
service/static-app created
deployment.apps/static-app created
buildconfig.build.openshift.io/static-app created
imagestream.image.openshift.io/static-app created
scaledobject.keda.sh/static-app created
triggerauthentication.keda.sh/keda-trigger-auth-prometheus created
route.route.openshift.io/static-app created
```

```
oc get deployment,scaledobject,hpa -n keda-test
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/prometheus-example-app	1/1	1	1	37m
deployment.apps/static-app	1/1	1	1	36m

NAME	SCALETARGETKIND	SCALETARGETNAME	MIN	MAX
scaledobject.keda.sh/static-app	apps/v1.Deployment	static-app	1	10
prometheus	keda-trigger-auth-prometheus	True	False	False

NAME	REFERENCE
horizontalpodautoscaler.autoscaling/keda-hpa-static-app	Deployment/static-app
TARGETS	MINPODS
MAXPODS	REPLICAS
AGE	
horizontalpodautoscaler.autoscaling/keda-hpa-static-app	Deployment/static-app
(avg)	0/5

Managing metrics targets

Go to Observer → Metrics → promql query and search for "version" to see metric from [custom metric app](#).

<https://docs.openshift.com/container-platform/4.11/monitoring/managing-metrics-targets.html>

As admin Observe → Targets → filter to keda-test namespace.

Target <http://10.128.5.43:8080/metrics> corresponds to endpoint of service prometheus-example-app