Red Hat OpenShift

Creating a ping source

A ping source is an event source that can be used to periodically send ping events with a constant payload to an event consumer. A ping source can be used to schedule sending events, similar to a timer.

Creating a ping source by using the web console

After Knative Eventing is installed on your cluster, you can create a ping source by using the web console. Using the OpenShift Container Platform web console provides a streamlined and intuitive user interface to create an event source.

Prerequisites

- You have logged in to the OpenShift Container Platform web console.
- The OpenShift Serverless Operator, Knative Serving and Knative Eventing are installed on the cluster.
- You have created a project or have access to a project with the appropriate roles and permissions to create applications and other workloads in OpenShift Container Platform.

Procedure

- To verify that the ping source is working, create a simple Knative service that dumps incoming messages to the logs of the service.
  - In the **Developer** perspective, navigate to **+Add** → **YAML**.
  - Copy the example YAML:

```
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
  name: event-display
spec:
  template:
    spec:
      containers:
        - image: quay.io/openshift-knative/showcase
```

- Click **Create**.

- Create a ping source in the same namespace as the service created in the previous step, or any other sink that you want to send events to.
  - In the **Developer** perspective, navigate to **+Add** → **Event Source**. The **Event Sources** page is displayed.
  - Optional: If you have multiple providers for your event sources, select the required provider from the **Providers** list to filter the available event sources from the provider.
  - Select **Ping Source** and then click **Create Event Source**. The **Create Event Source** page is displayed.

    You can configure the **PingSource** settings by using the **Form view** or **YAML view** and can switch between the views. The data is persisted when switching between the views.

  - Enter a value for **Schedule**. In this example, the value is `*/2 * * * *`, which creates a PingSource that sends a message every two minutes.
  - Optional: You can enter a value for **Data**, which is the message payload.
  - In the **Target** section, select your event sink. This can be either a **Resource** or a **URI**:
    - Select **Resource** to use a channel, broker, or service as an event sink for the event source. In this example, the `event-display` service created in the previous step is used as the target **Resource**.

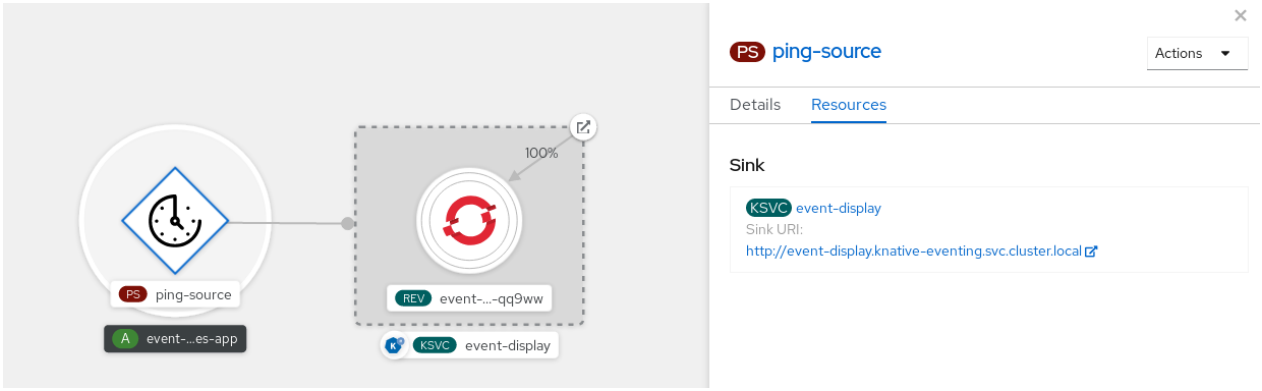
- Select **URI** to specify a Uniform Resource Identifier (URI) where the events are routed to.

- Click **Create**.

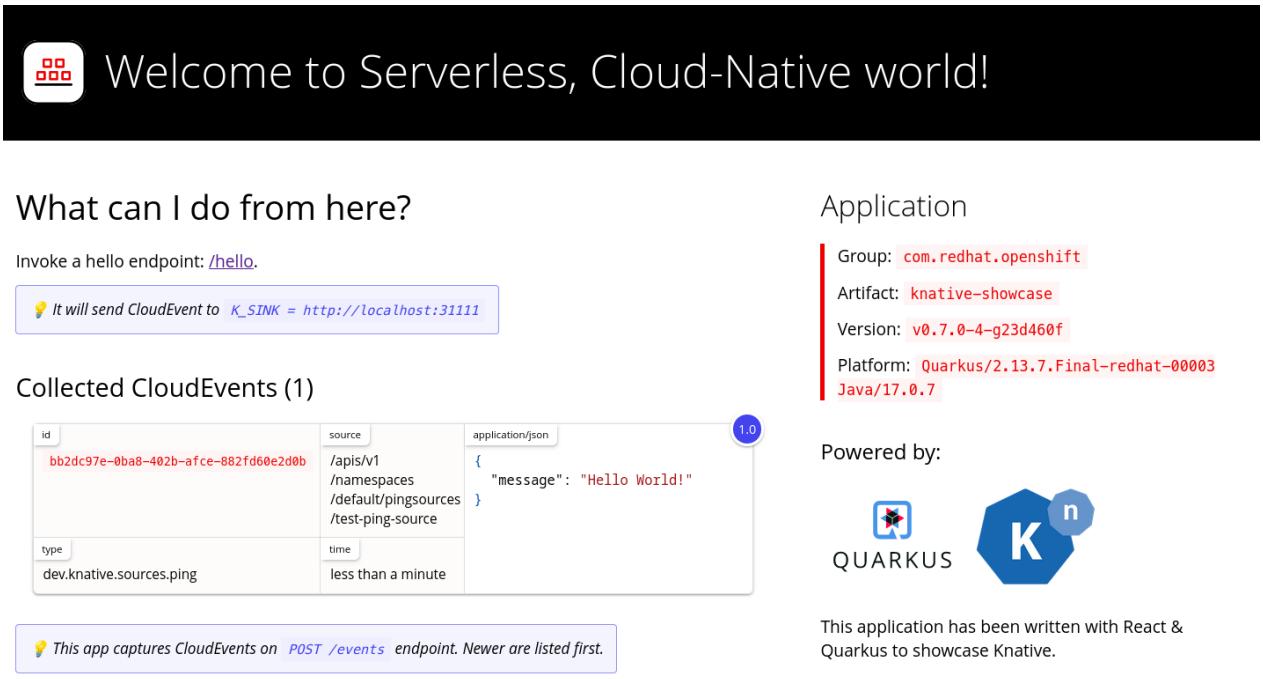
Verification

You can verify that the ping source was created and is connected to the sink by viewing the **Topology** page.

- In the **Developer** perspective, navigate to **Topology**.
- View the ping source and sink.



- View the event-display service in the web browser. You should see the ping source events in the web UI.



Deleting the ping source

- Navigate to the **Topology** view.

- Right-click the API server source and select **Delete Ping Source**.

Creating a ping source by using the Knative CLI

You can use the `kn source ping create` command to create a ping source by using the Knative (`kn`) CLI. Using the Knative CLI to create event sources provides a more streamlined and intuitive user interface than modifying YAML files directly.

Prerequisites

- The OpenShift Serverless Operator, Knative Serving and Knative Eventing are installed on the cluster.
- You have installed the Knative (`kn`) CLI.
- You have created a project or have access to a project with the appropriate roles and permissions to create applications and other workloads in OpenShift Container Platform.
- Optional: If you want to use the verification steps for this procedure, install the OpenShift CLI (`oc`).

Procedure

- To verify that the ping source is working, create a simple Knative service that dumps incoming messages to the service logs:

```
$ kn service create event-display \
    --image quay.io/openshift-knative/showcase
```

- For each set of ping events that you want to request, create a ping source in the same namespace as the event consumer:

```
$ kn source ping create test-ping-source \
    --schedule "*/2 * * * *" \
    --data '{"message": "Hello world!"}' \
    --sink ksvc:event-display
```

- Check that the controller is mapped correctly by entering the following command and inspecting the output:

```
$ kn source ping describe test-ping-source
```

Example output

```
Name:         test-ping-source
Namespace:    default
Annotations:  sources.knative.dev/creator=developer,
sources.knative.dev/lastModifier=developer
Age:          15s
Schedule:     */2 * * * *
Data:         {"message": "Hello world!"}

Sink:
  Name:       event-display
  Namespace:  default
  Resource:   Service (serving.knative.dev/v1)

Conditions:
  OK TYPE                 AGE REASON
  ++ Ready                 8s
  ++ Deployed              8s
  ++ SinkProvided         15s
  ++ ValidSchedule        15s
  ++ EventTypeProvided    15s
  ++ ResourcesCorrect     15s
```

Verification

You can verify that the Kubernetes events were sent to the Knative event sink by looking at the logs of the sink pod.

By default, Knative services terminate their pods if no traffic is received within a 60 second period. The example shown in this guide creates a ping source that sends a message every 2 minutes, so each message should be observed in a newly created pod.

- Watch for new pods created:

```
$ watch oc get pods
```

- Cancel watching the pods using Ctrl+C, then look at the logs of the created pod:

```
$ oc logs $(oc get pod -o name | grep event-display) -c user-
container
```

Example output

```
☁  cloudevents.Event
Validation: valid
Context Attributes,
  specversion: 1.0
  type: dev.knative.sources.ping
  source: /apis/v1/namespaces/default/pingsources/test-ping-
source
  id: 99e4f4f6-08ff-4bff-acf1-47f61ded68c9
  time: 2020-04-07T16:16:00.000601161Z
  datacontenttype: application/json
Data,
  {
    "message": "Hello world!"
  }
```

Deleting the ping source

- Delete the ping source:

```
$ kn delete pingsources.sources.knative.dev <ping_source_name>
```

Knative CLI sink flag

When you create an event source by using the Knative (`kn`) CLI, you can specify a sink where events are sent to from that resource by using the `--sink` flag. The sink can be any addressable or callable resource that can receive incoming events from other resources.

The following example creates a sink binding that uses a service, `http://event-display.svc.cluster.local` , as the sink:

Example command using the sink flag

```
$ kn source binding create bind-heartbeat \
  --namespace sinkbinding-example \
  --subject "Job:batch/v1:app=heartbeat-cron" \
  --sink http://event-display.svc.cluster.local \  (1)
  --ce-override "sink=bound"
```

1 `svc` in `http://event-display.svc.cluster.local` determines that the sink is a Knative service. Other default sink prefixes include `channel` , and `broker` .

Creating a ping source by using YAML

Creating Knative resources by using YAML files uses a declarative API, which enables you to describe event sources declaratively and in a reproducible manner. To create a serverless ping source by using YAML, you must create a YAML file that defines a `PingSource` object, then apply it by using `oc apply` .

Example `PingSource` object

```
apiVersion: sources.knative.dev/v1
kind: PingSource
metadata:
  name: test-ping-source
spec:
  schedule: "*/2 * * * *" (1)
  data: '{"message": "Hello world!"}' (2)
  sink: (3)
    ref:
      apiVersion: serving.knative.dev/v1
      kind: Service
      name: event-display
```

1. The schedule of the event specified using CRON expression.
2. The event message body expressed as a JSON encoded data string.
3. These are the details of the event consumer. In this example, we are using a Knative service named `event-display` .

Prerequisites

- The OpenShift Serverless Operator, Knative Serving and Knative Eventing are installed on the cluster.
- Install the OpenShift CLI (`oc`).
- You have created a project or have access to a project with the appropriate roles and permissions to create applications and other workloads in OpenShift Container Platform.

Procedure

- To verify that the ping source is working, create a simple Knative service that dumps incoming messages to the service's logs.
    - Create a service YAML file:

```
apiVersion: serving.knative.dev/v1
kind: Service
metadata:
  name: event-display
spec:
  template:
    spec:
      containers:
        - image: quay.io/openshift-knative/showcase
```

- Create the service:

```
$ oc apply -f <filename>
```

- For each set of ping events that you want to request, create a ping source in the same namespace as the event consumer.
  - Create a YAML file for the ping source:

```
apiVersion: sources.knative.dev/v1
kind: PingSource
metadata:
  name: test-ping-source
spec:
  schedule: "*/2 * * * *"
  data: '{"message": "Hello world!"}'
  sink:
    ref:
      apiVersion: serving.knative.dev/v1
      kind: Service
      name: event-display
```

  - Create the ping source:

```
$ oc apply -f <filename>
```

- Check that the controller is mapped correctly by entering the following command:

```
$ oc get pingsource.sources.knative.dev <ping_source_name> -
oyaml
```

Example output

```
apiVersion: sources.knative.dev/v1
kind: PingSource
metadata:
  annotations:
    sources.knative.dev/creator: developer
    sources.knative.dev/lastModifier: developer
  creationTimestamp: "2020-04-07T16:11:14Z"
  generation: 1
  name: test-ping-source
  namespace: default
  resourceVersion: "55257"
  selfLink:
/apis/sources.knative.dev/v1/namespaces/default/pingsources/tes
t-ping-source
  uid: 3d80d50b-f8c7-4c1b-99f7-3ec00e0a8164
spec:
  data: '{ value: "hello" }'
  schedule: '*/2 * * * *'
  sink:
    ref:
      apiVersion: serving.knative.dev/v1
      kind: Service
      name: event-display
      namespace: default
```

Verification

You can verify that the Kubernetes events were sent to the Knative event sink by looking at the sink pod's logs.

By default, Knative services terminate their pods if no traffic is received within a 60 second period. The example shown in this guide creates a PingSource that sends a message every 2 minutes, so each message should be observed in a newly created pod.

- Watch for new pods created:

```
$ watch oc get pods
```

- Cancel watching the pods using Ctrl+C, then look at the logs of the created pod:

```
$ oc logs $(oc get pod -o name | grep event-display) -c user-
container
```

Example output

```
☁  cloudevents.Event
Validation: valid
Context Attributes,
  specversion: 1.0
  type: dev.knative.sources.ping
  source: /apis/v1/namespaces/default/pingsources/test-ping-
source
  id: 042ff529-240e-45ee-b40c-3a908129853e
  time: 2020-04-07T16:22:00.000791674Z
  datacontenttype: application/json
Data,
  {
    "message": "Hello world!"
  }
```

Deleting the ping source

- Delete the ping source:

```
$ oc delete -f <filename>
```

Example command

```
$ oc delete -f ping-source.yaml
```