



Serverless with Knative

Kubernetes-based platform for modern event-driven serverless workloads

JCON Online, October 28, 2020

Dr. Roland Huß @ro14nd

Principal Software Engineer, Red Hat



Wait ... wat ?



Serverless

"Serverless computing refers to the concept of building and running **applications** that **do not require server management**. It describes a finer-grained **deployment model** where applications, bundled as one or more functions are uploaded to a platform and then **executed**, **scaled**, and **billed** in response to the exact **demand** needed at the moment"

-- CNCF Definition, <https://www.cncf.io/blog/2018/02/14/cncf-takes-first-step-towards-serverless-computing/>

Serverless vs. FaaS

Serverless is a **Deployment Model** that abstracts away the driving machine infrastructure.

- No server management required
- Executed, scaled and billed according to demand
- Defines a deployment packaging, but otherwise agnostic to the application

FaaS (Function-as-a-Service) is a **Programming Model** that mandates developing your application with fine grained function that match a given signature.

- Deployed as Serverless application
- Typically used as *glue code* to connect services

A photograph of a wooden boat's interior, looking towards the bow. The boat is made of weathered wood and has several coils of rope on the deck. In the background, the ocean stretches to a horizon with several large, rocky islands under a blue sky with scattered white clouds. The water is a deep blue with some white foam visible near the boat's hull.

Knative

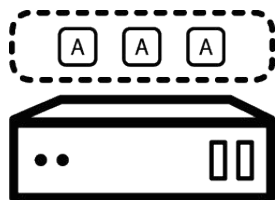
Kubernetes-based platform to
deploy and **manage** modern
serverless workloads.

<https://knative.dev>

Components

Serving

A request-driven model that serves the container with your application and can "scale to zero".



Eventing

Common infrastructure for consuming and producing events that will stimulate applications.



Background Information

- Started as an **Open Source** Project mid-2018 by Google
- Community driven with a lot of vendor backing
 - <https://github.com/knative>
 - <https://knative.dev>
 - Support by Google, Red Hat, IBM, VMware, Triggermesh, SAP and more
 - Organized in multiple Working Groups with weekly meetings
- Releases
 - Current: **v0.18**
 - 6 week release cadence

Try Knative !

- Install from resource descriptors on Kubernetes Cluster
 - <https://knative.dev/docs/install/>
- Google **Cloud Run** (managed and on GKE)
 - <https://cloud.google.com/run/>
 - Not all Knative features implemented
 - see <https://ahmet.im/blog/cloud-run-is-a-knative> ?
- Red Hat **OpenShift Serverless**
 - <https://www.openshift.com/learn/topics/serverless>
 - Supports all Knative features
 - Serving GA with full support, Eventing in Technical Preview (GA in November 2020)



Serving

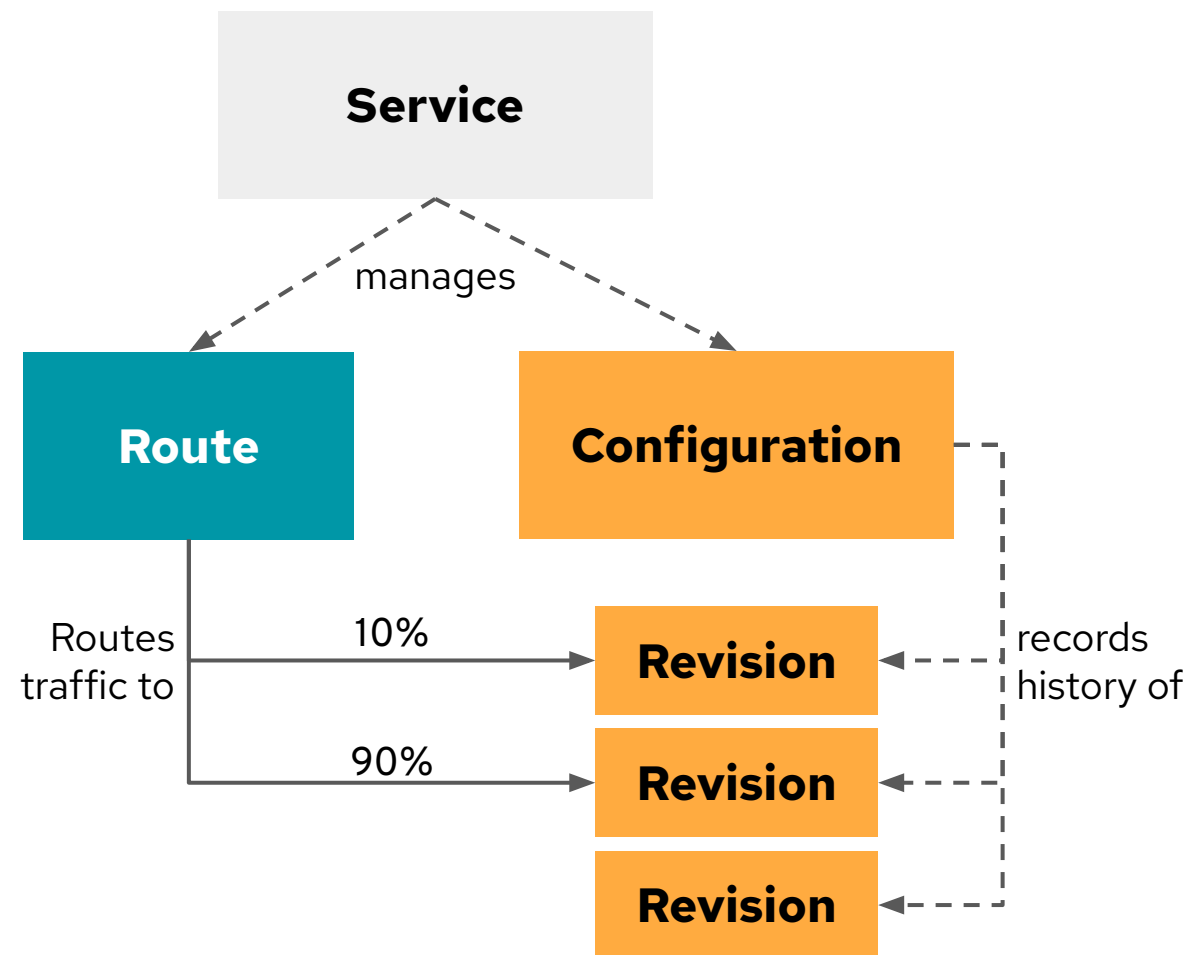
Route, scale-to-zero and track application **revisions** with ease.

Concepts

- **Demand-based autoscaling**, including scale-to-zero
- Separation of code and configuration
- Opinionated deployment model catered for **stateless applications**
 - Single Port
 - No PersistentVolumes
 - Single Container (about to change)
- Rich **traffic split capabilities** to enable custom rollout strategies of new versions

Resources

- **Configuration** represent the *floating HEAD* of a history of **Revisions**
- **Revision** represents an immutable snapshot of code and configuration
- **Route** configure ingress over a collection of Revisions
- **Service** (not K8s services !) is a top-level entity that manage a set of Routes and Configurations



From Deployment to KService

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: random
spec:
  replicas: 1
  selector:
    matchLabels:
      app: random
  template:
    metadata:
      labels:
        app: random
    spec:
      containers:
        - image: rhuss/random
          name: random
          ports:
            - containerPort: 8080
```

```
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: random
spec:
  replicas: 1
  selector:
    matchLabels:
      app: random
  template:
    metadata:
      labels:
        app: random
    spec:
      containers:
        - image: rhuss/random
          name: random
          ports:
            - containerPort: 8080
```

No more K8s
Service or
Ingress/Route
required!

Demo



Eventing

Demo

**Universal subscription, delivery,
and management of CloudEvents.**

Eventing

- Based on CloudEvents (CNCF Standard)
- Pluggable event transport via **Channels**
 - In-Memory
 - Apache Kafka
 - Google Pub-Sub
- Flexible routing of events from Sources to Sinks
 - **Source:** Adapter for integrating 3rd party systems and emitting CloudEvents
 - **Sink:** Addressable endpoint for CloudEvents (like a Knative Service)



Event Sources

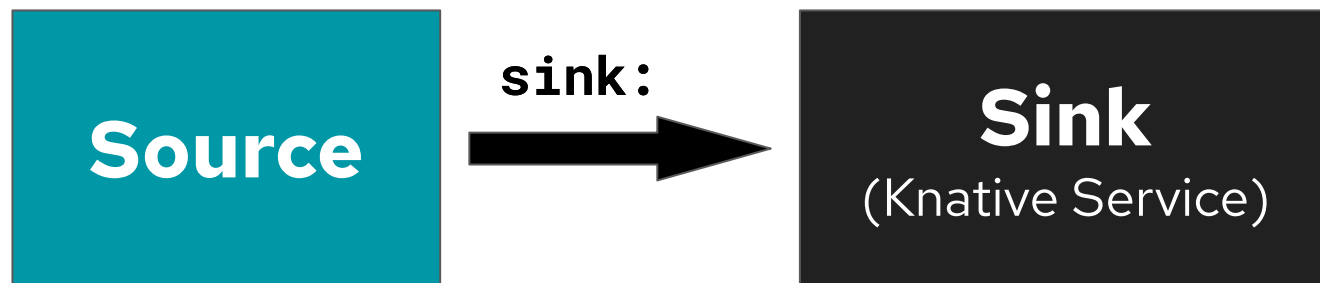
- Integrating 3rd party systems with Knative
- More often “**Adapter**” than an original event source
- Declared with a **Custom Resource**
- Evaluated by an Operator
- Push or Pull based
- Converting custom event formats to **CloudEvents**

Sources

Builtin Sources	
PingSource	Emitting static CloudEvents periodically
ApiServerSource	Kubernetes API Server events as CloudEvents
SinkBinding	Binds an arbitrary Pod specification to a Sink
ContainerSource	Meta-Source combining SinkBinding & Deployment
Contributed Sources	
GitHubSource	Converts GitHub webhooks events to CloudEvents
KafkaSource	Apache Kafka messages as CloudEvents
CamelKSource	Apache Camel components as sources

and many more: <https://knative.dev/docs/eventing/sources/>

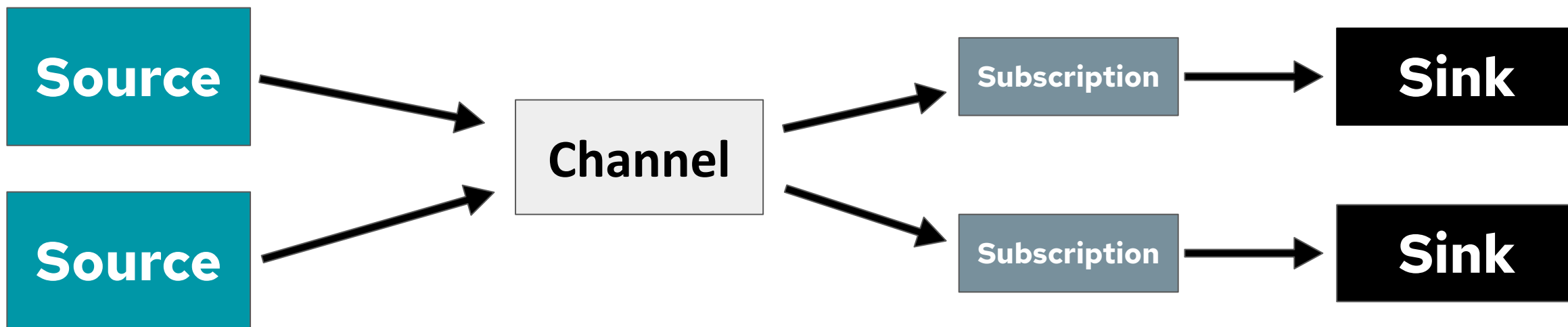
Source → Service : Direct Connection



- Simplest way to get CloudEvents to a service
- Drawbacks:
 - No queuing support when service is unavailable
 - No back pressure support
 - Only one Service can consume events
 - No filtering, Service gets always all events

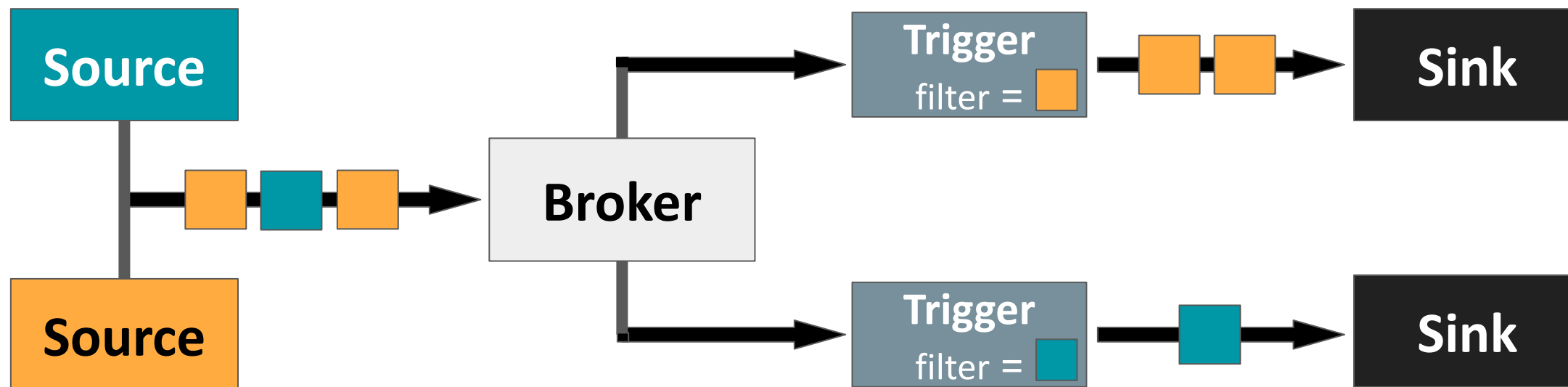
Demo

Source → Service : Channel & Subscription



- Multiple Services can consume the same event
- Subscription can point to a reply channel (not shown here)
- Various Channel Backends available
 - In-Memory, Kafka, GCP PubSub, (write your own)
- Drawbacks:
 - Channel Infrastructure needs to be set up manually
 - No filtering, Service gets always all events

Source → Service: Broker & Trigger



Broker

- Eventing Mesh for distributing Events
- Addressed by sources as sink

Trigger

- Filter on CloudEvent attributes (e.g. type)
- Connects a Sink with Broker

Source → Service: Broker & Trigger

- **Broker**

- Eventing Mesh (or Event Delivery System)
- Connects Sources with Sinks
- Uses Channels internally, creating on the fly
- Multi-tenant

- **Trigger**

- Filter events (e.g. type and/or source)
- Can produce new events (returned to Broker)
- Delivered as CloudEvents

More Knative Eventing

- **EventRegistry**
 - EventType CRD
 - Discoverability of Events
- **Sequence**
 - Chaining multiple Services
 - Sinking to an “Addressable” (Service, Channel, Sequence, Broker ...)
- **Parallel**
 - Branching of events with filters
 - Allows to implement conditional processing

A photograph of a wooden boat on the ocean. The boat is made of weathered wood and has a small cabin structure at the front. A red rope is tied to the front of the boat. The ocean is a deep blue color, and there are several small, rocky islands in the distance under a blue sky with white clouds. The word "Summary" is written in large white letters across the middle of the image.

Summary

Summary

Knative Serving

- Simplified Deployment for stateless workloads
- Traffic based autoscaling including Scale-to-Zero
- Traffic splitting for custom rollout / rollback scenarios

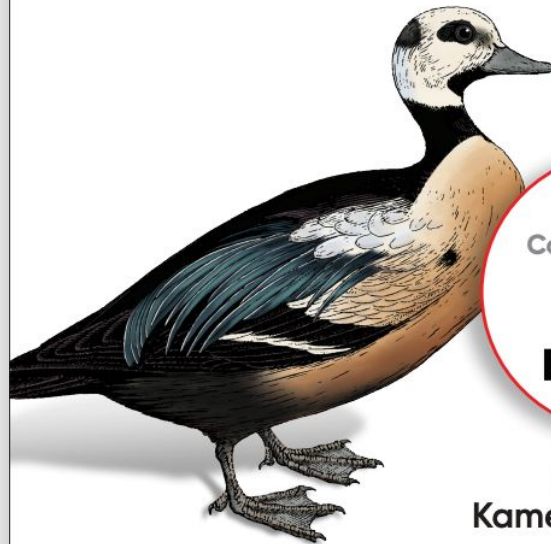
Knative Eventing

- External Triggers for feeding Knative Services
- Based on CloudEvents
- Backed by proven messaging systems
- Flexible messaging setup

O'REILLY®

Knative Cookbook

Building Effective Serverless Applications
with Kubernetes and OpenShift

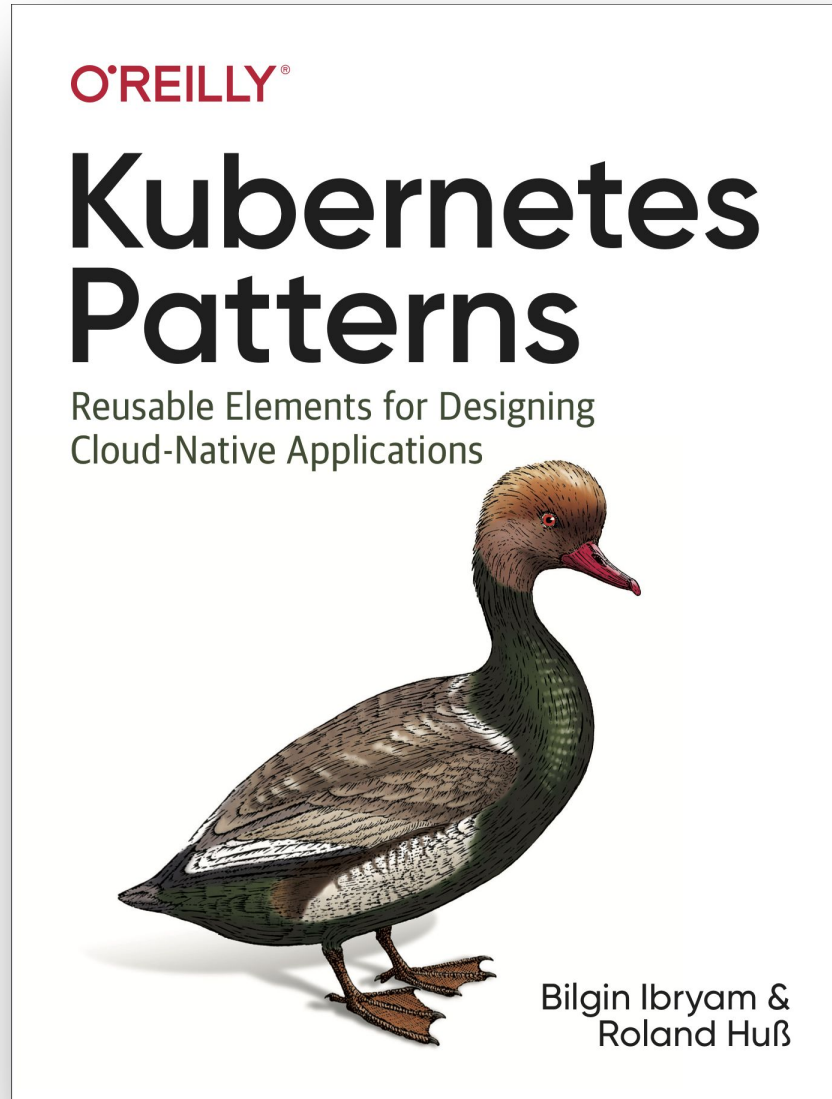


Compliments of



Red Hat

Burr Sutter &
Kamesh Sampath



<https://k8spatterns.io>

Thank you

 @ro14nd

Picture Credits

<https://www.pexels.com/photo/boat-island-ocean-sea-218999/>

<https://unsplash.com/photos/t6t2-gXKxXM>

<https://unsplash.com/photos/UGMf30W28qc>

<https://pixabay.com/photos/hamburg-speicherstadt-channel-2976711/>

<https://pixabay.com/photos/beer-machine-alcohol-brewery-1513436/>

<https://unsplash.com/photos/9SWHlgu8A8k>

<https://me.me/i/aws-lambda-is-just-glorified-cgi-bin-imgflip-com-change-my-mind-d0b715592ba34b08b79452ad02783ca2>

https://unsplash.com/photos/dodn_OTESN0