



Knative

Kubernetes-based platform to manage modern serverless workloads.

Roland Huß @ro14nd

Principal Software Engineer, Red Hat



What is serverless again ?

“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/>

Wait... wat ?



A photograph of a wooden boat, likely a traditional Thai longtail boat, viewed from the front. The boat is made of weathered wood and has a pointed prow. On the prow, there are some colorful cloths (green, red, and blue) and a coiled rope. The boat is on a body of water with a deep blue-green hue. In the background, there are several large, rocky islands or reefs under a blue sky with scattered white clouds. The word "Knative" is overlaid in the center of the image in a large, white, sans-serif font.

Knative

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

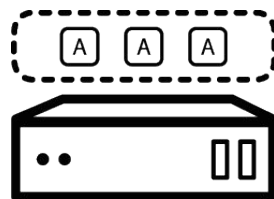
<https://knative.dev>

Knative 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.



O'REILLY®

Kubernetes Patterns

Reusable Elements for Designing
Cloud-Native Applications



Bilgin Ibryam &
Roland Huß

<https://k8spatterns.io>

Background Information

- Started as Open Source Project mid-2018 by Google
- Community + Company Driven
 - <https://github.com/knative>
 - <https://knative.dev>
 - Support by Google, Red Hat, IBM, Pivotal, Triggermesh..
 - Organized in 7 Working Groups with weekly meetings
- Releases
 - Current: Serving v0.9.0, Eventing v0.9.0
 - GA:
 - Serving: ETA this year
 - Eventing: A “bit” later

Try Knative

- Install from resource descriptors on Kubernetes Cluster
 - <https://knative.dev/docs/install/>
- Google Cloud Run (managed and on GKE) - Beta
 - <https://cloud.google.com/run/>
 - Free tier: (180k CPU-s, 360k GB-s, 2M Requests) per month
- Red Hat OpenShift Serverless - Technical Preview
 - <https://www.openshift.com/learn/topics/serverless>



Serving

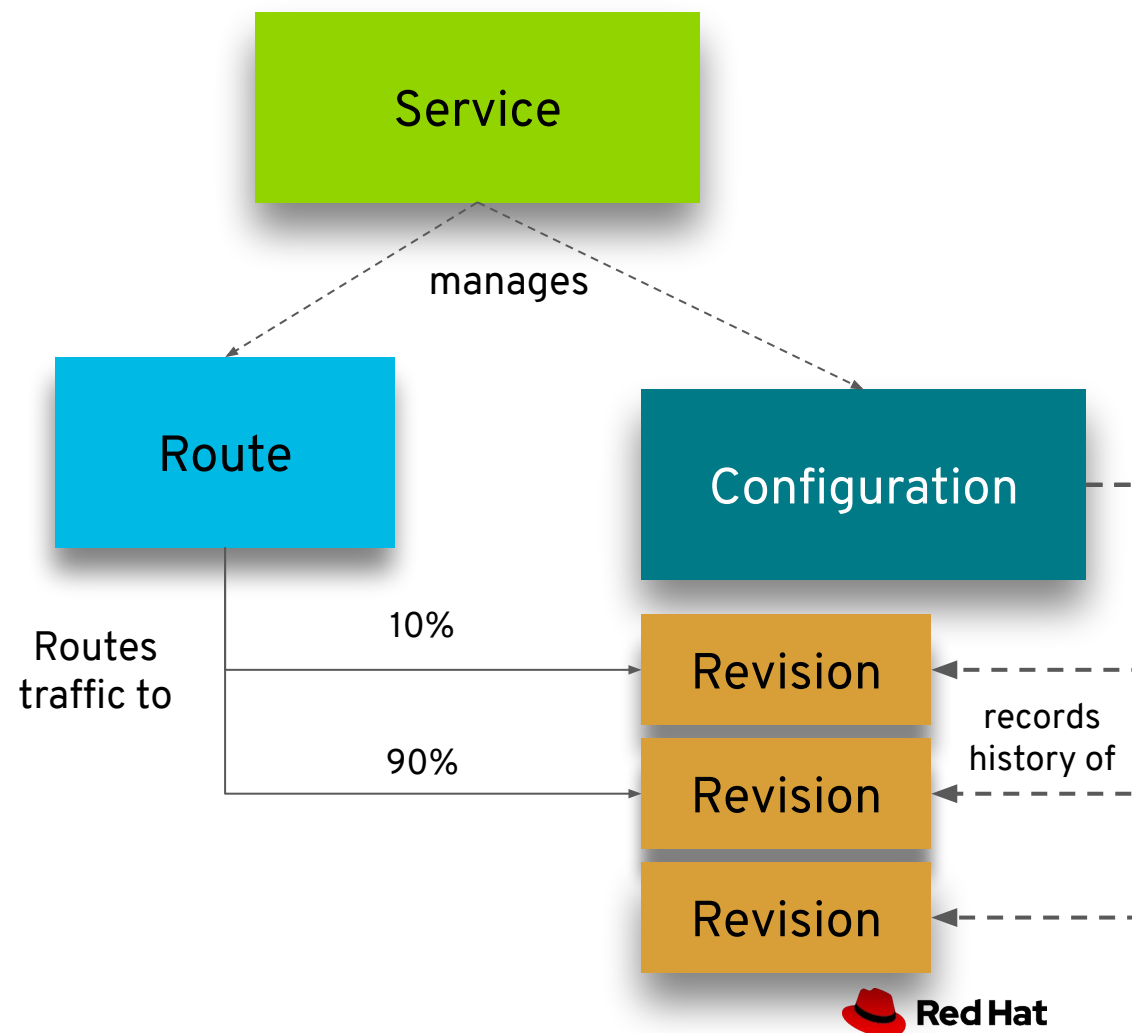
Route, scale-to-zero and
track application revisions
with ease.

Knative Serving Concepts

- Separation of code and configuration
- Immutable Revisions
- Autoscaling including scale-to-zero
- Traffic Splitting
- Simplified and opinionated deployment model
 - Single Port
 - No PersistentVolumes
 - Single Container
- Callable by Knative eventing

Knative Serving 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**
- **Services** (not K8s services !) are top-level entities that manage a set of **Routes** and **Configurations**



Service Peeling

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: lotto
spec:
  replicas: 1
  selector:
    matchLabels:
      app: lotto
  template:
    metadata:
      labels:
        app: lotto
    spec:
      containers:
        - image: cds19/lotto
          name: lotto
          ports:
            - containerPort: 8080

```

```

apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: lotto
spec:
  replicas: 1
  selector:
    matchLabels:
      app: lotto
  template:
    metadata:
      labels:
        app: lotto
    spec:
      containers:
        - image: cds19/lotto
          name: lotto
          ports:
            - containerPort: 8080

```

No more K8s
Service or
Ingress/Route
required!

Demo



Eventing

Universal subscription, delivery,
and management of events.

Knative Eventing Key Features

Benefits

- Event orchestration
 - Declaratively API to distribute events
- Scales from just few events to live data-streaming pipelines
- Powered by



cloudevents

Extendable

- Multitude of Eventing Sources
- Pluggable internal transport (channel)
 - In-Memory(default)
 - Apache Kafka
 - Google Pub-Sub

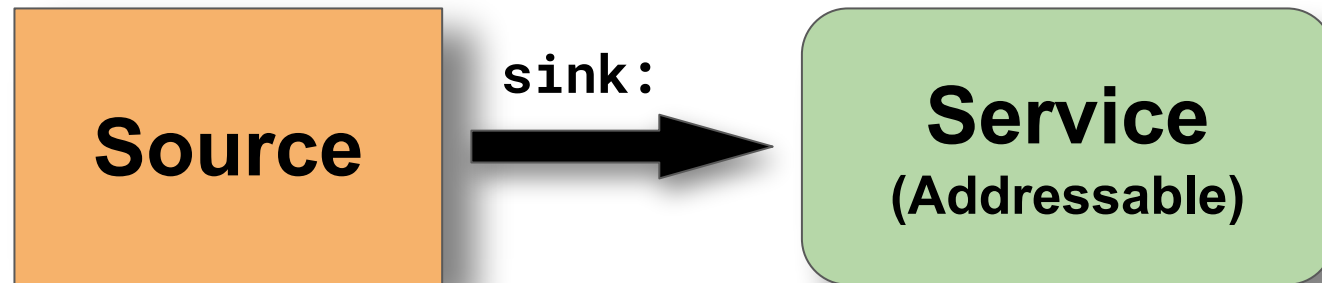
Knative Event Sources

- 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

Knative Eventing Sources

- Existing
 - **CronJob:** Constant events created at a fixed schedule
 - **Apiserver:** Watching for Kubernetes resource lifecycle events
 - **Container:** Arbitrary container instantiated
 - **GitHub:** Listen on GitHub API calls
 - **Kafka:** Connect to Kafka topics of a given Kafka Cluster
 - **Camel-K:** Camel components for connecting to external systems
 - and more: <https://knative.dev/docs/eventing/sources/>
- Custom
 - Write your own Source (CRD + Operator)
 - Use ContainerSource with custom image
 - Talk to Broker directly via HTTP

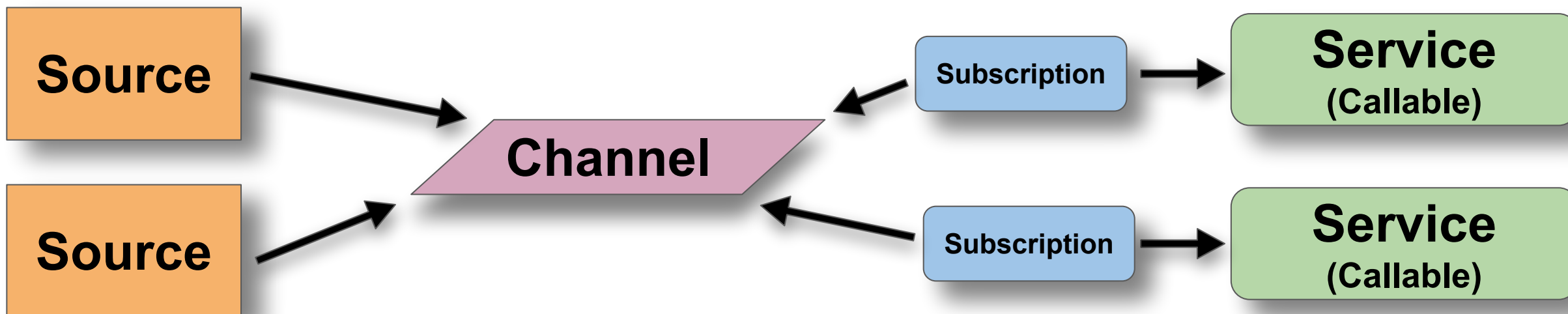
Source → Service : Direct Connection



- 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
 - No filtering, Service gets always all events

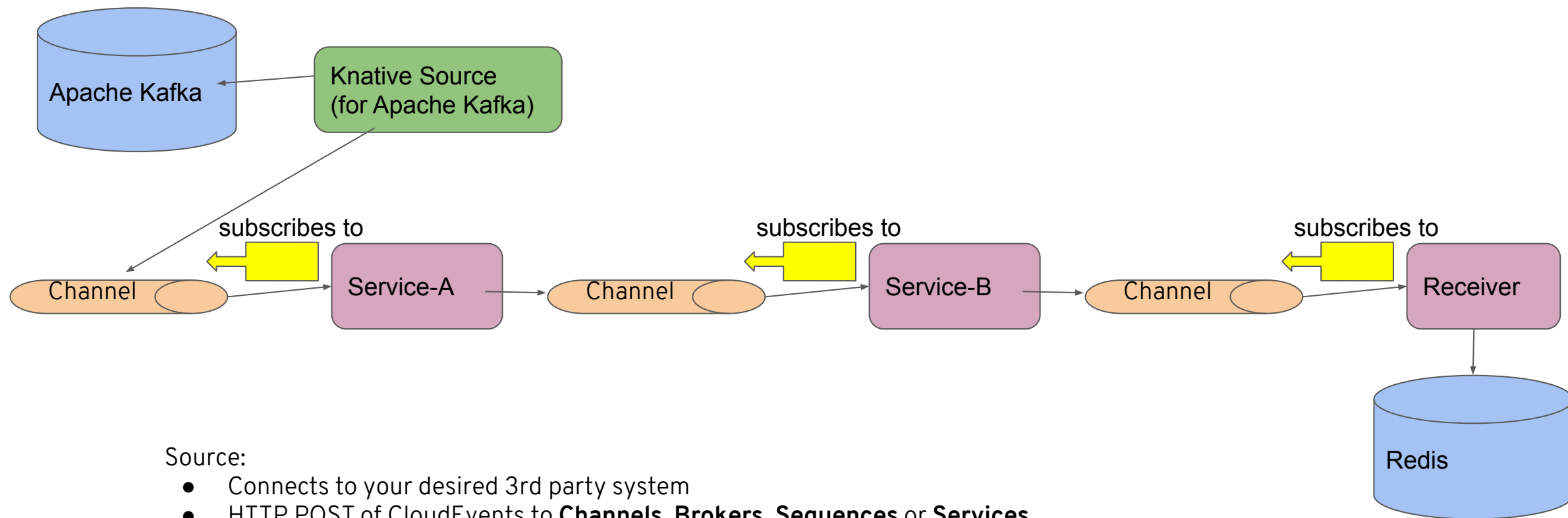
Demo

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

Event Driven Flow



Source:

- Connects to your desired 3rd party system
- HTTP POST of CloudEvents to **Channels, Brokers, Sequences** or **Services**

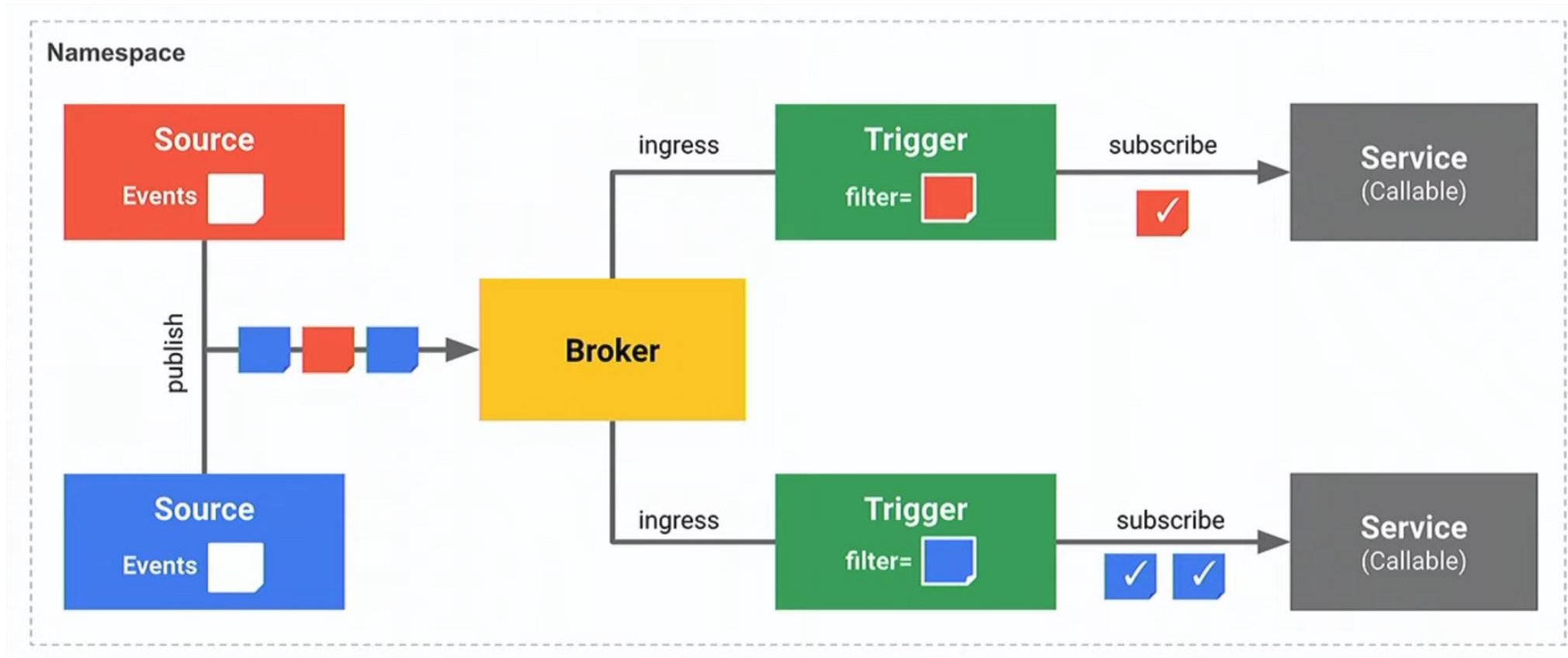
Channel:

- Has n subscribers (of Knative Services)
- “Persisting” messages for consumption by Subscribers

Service:

- Receives the HTTP POST of the (CloudEvent) message
 - Optionally returns processed data (replyChannel)

Source → Service: Broker & Trigger



Source → Service: Broker & Trigger

- **Broker**
 - Eventing Mesh (or Event Delivery System)
 - Connects Sources
 - Uses Channels internally
- **Trigger**
 - Filter events (e.g. type and/or source)
 - Can produce new events (returned to “Broker”)
 - Delivered as CloudEvents

Demo

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. A red rope is tied to the top of the cabin. The boat is moving through the water, creating a wake. In the background, there are several small, rocky islands 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

Thank you



<https://k8spatterns.io>



<https://twitter.com/ro14nd>



<https://twitter.com/mwessendorf>



<https://twitter.com/k8spatterns>

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_0TESN0