

OpenShift Developer

Distance Learning

Optional Learnings and homework assignment



[linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)



[facebook.com/redhatinc](https://www.facebook.com/redhatinc)



[youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)



twitter.com/RedHat

Self introduction

Name: Wanja Pernath

Email: wpernath@redhat.com

Base: Germany (very close to the Alps)

Role: EMEA Technical Partner Development

Manager - OpenShift and MW

Experience: Years of Consulting, Training,

PreSales at Red Hat and before



Agenda

Agenda

- What you've done so far
- Optional Learnings - Introduction
 - Serverless / knative
 - ServiceMesh / Istio
- Homework
 - Description
 - Where to upload?
- Summary & Thank you

What you've done so far...

Week #1

- You've started to become familiar with development on OpenShift
 - Basic understanding
 - Using the CLI and the UI
 - First apps created
- More advanced basic development

Week #2

- You've learned what and how to package your app to do
 - Release management
 - CI/CD
 - You've learned to differentiate between the need of tools for both
- You might come to the conclusion that you will use
 - Kustomize for CI/CD and releases
 - Helm Charts for simple app deployment
 - Operators for more complex apps which require special handling of Day 2 operations

Week #3

- You've learned how to do CI/CD with OpenShift
- You've learned and have an understanding of GitOps and how to use it
- You've a good understanding of Tekton / OpenShift Pipelines
- You understand that you still need BOTH tools for your projects
 - ArgoCD (or similar things) for CD
 - And something like Tekton to do CI

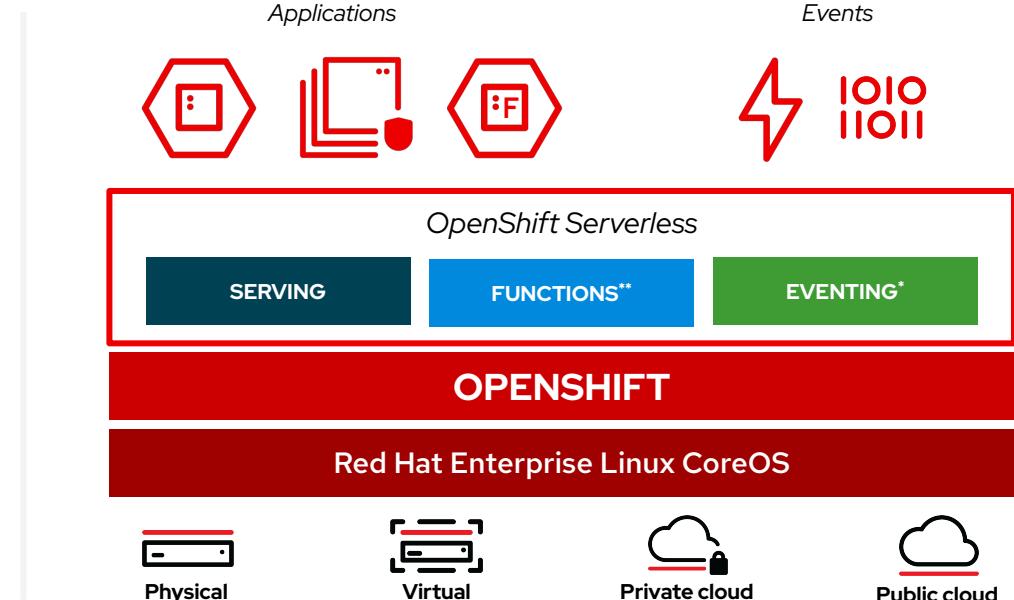
Optional - Serverless / knative



OpenShift Serverless

Event-driven serverless containers and functions

- Deploy and run **serverless containers**
- Use any programming language or runtime
- Modernize existing applications to run serverless
- Powered by a rich ecosystem of event sources
- Manage serverless apps natively in Kubernetes
- Based on open source project **Knative**
- Run anywhere OpenShift runs



* Eventing is currently in Technology Preview

** Functions are currently a work in progress initiative

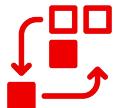
OpenShift Serverless

Key Features



Containers made easy

Simplified developer experience to deploy applications/code on serverless containers abstracting infrastructure & focusing on what matters.



Immutable revisions

Deploy new features: performing canary, A/B or blue-green testing with gradual traffic rollout with no sweat and following best practices.



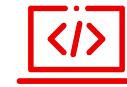
Automatic scaling

No need to configure number of replicas, or idling. Scale to zero when not in use, auto scale to thousands during peak, with built-in reliability and fault-tolerance.



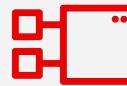
Ready for the Hybrid

Portable serverless running anywhere OpenShift runs, that is on-premises or on any public cloud. Leverage data locality and SaaS when needed.



Any programming language

Use any programming language or runtime of choice. From Java, Python, Go and JavaScript to Quarkus, SpringBoot or Node.js.



Event Driven

Architectures coupled & distributed apps connecting with a variety of built-in or third-party event sources or connectors powered by Operators.

Installation experience

"Easy day 1 and even better for day 2"

- Click Install experience
- Developer & admin experience in Console
- Built-in event sources
- No external dependencies.

 OpenShift Serverless Operator

1.7.0 provided by Red Hat, Inc.

[Install](#)

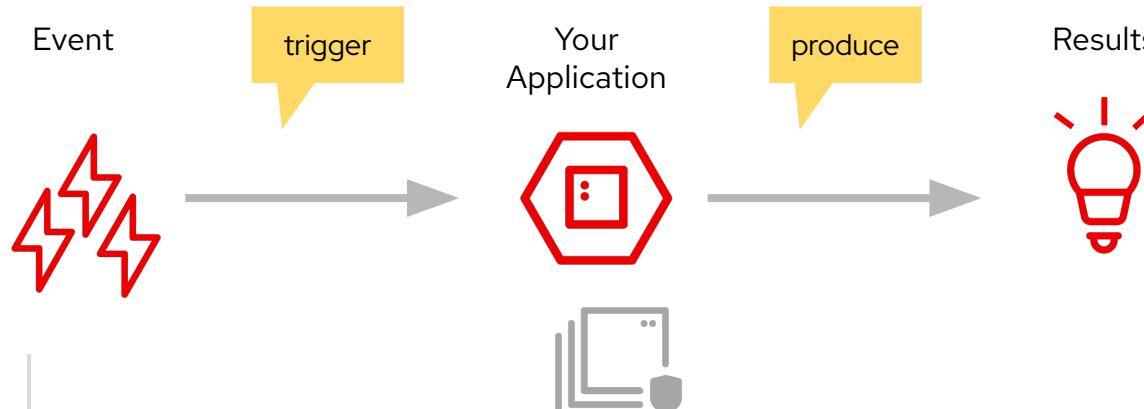
OPERATOR VERSION
1.7.0

PROVIDER TYPE
Red Hat

PROVIDER
Red Hat, Inc.

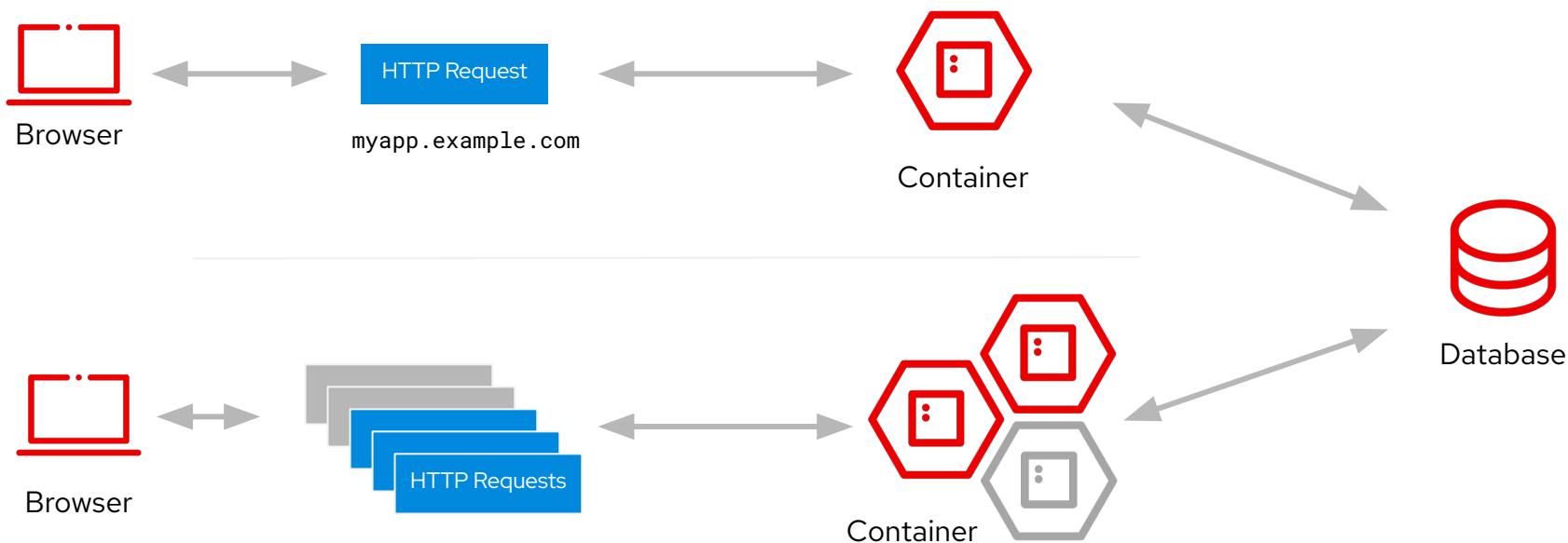
The Red Hat OpenShift Serverless operator provides a collection of APIs that enables containers, microservices and functions to run "serverless". Serverless applications can scale up and down (to zero) on demand and be triggered by a number of event sources. OpenShift Serverless integrates with a number of platform services, such as Metering and Monitoring and it is based on the open source project Knative.

The "Serverless Pattern"



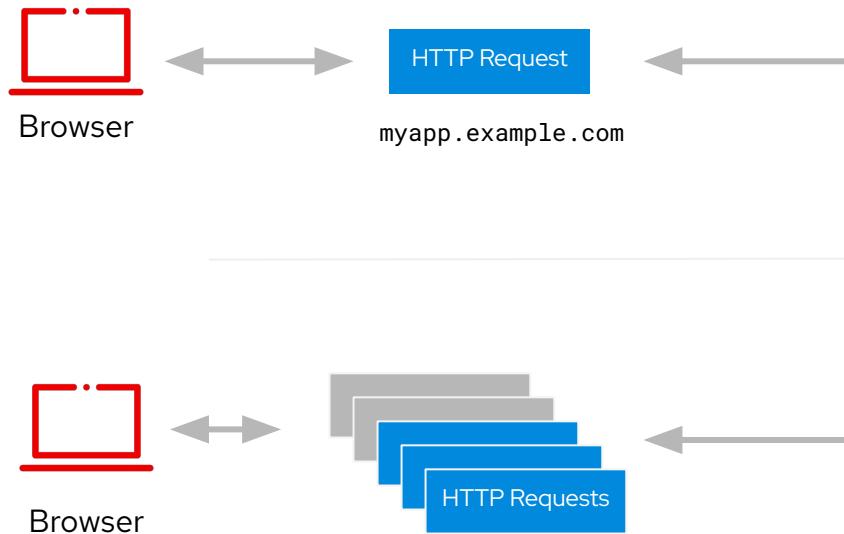
The "Serverless Pattern"

A serverless web application



The "Serverless Pattern"

A serverless web application



Benefits of this model:

- No need to setup auto-scaling and load balancers
 - Scale down and save resources when needed.
 - Scale up to meet the demand.
- No tickets to configure SSL for applications
- Enable Event Driven Architectures (EDA) patterns
- Enable teams to associate cost with IT
- Modernize existing applications to run as serverless containers

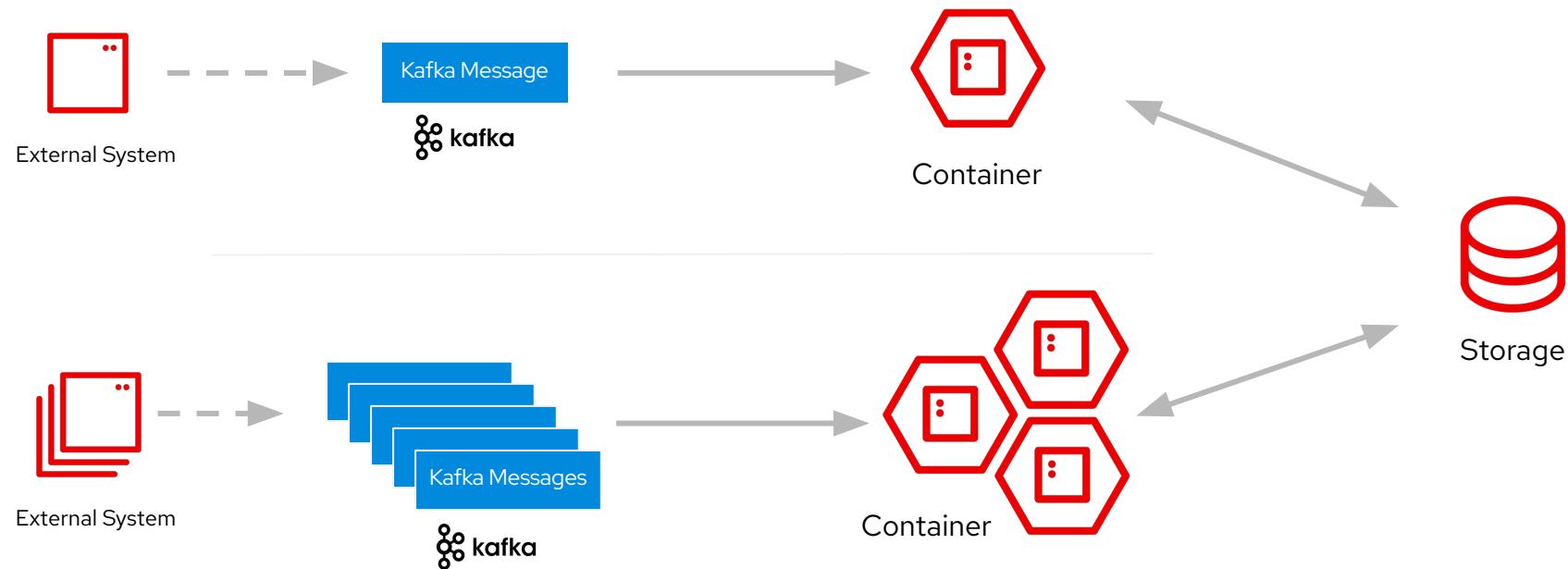


Database

Container

The "Serverless Pattern"

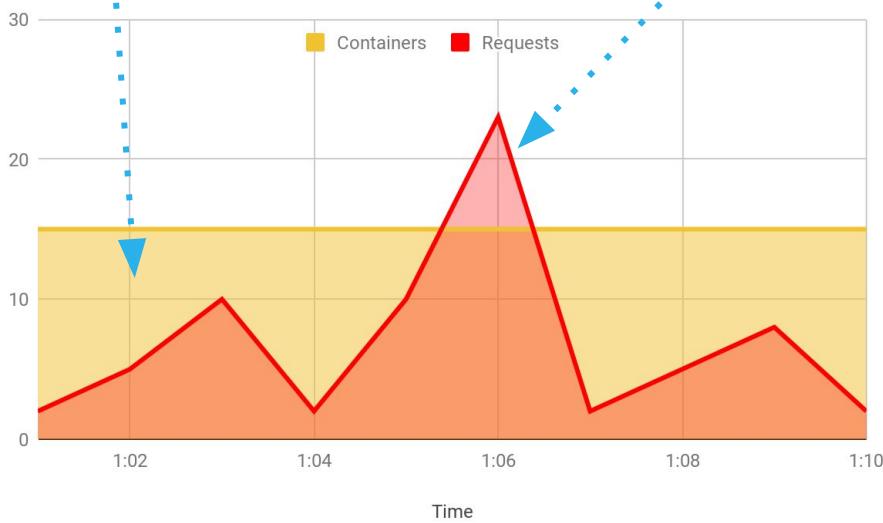
Processing a Kafka message



Serverless Operational Benefits

Over provisioning

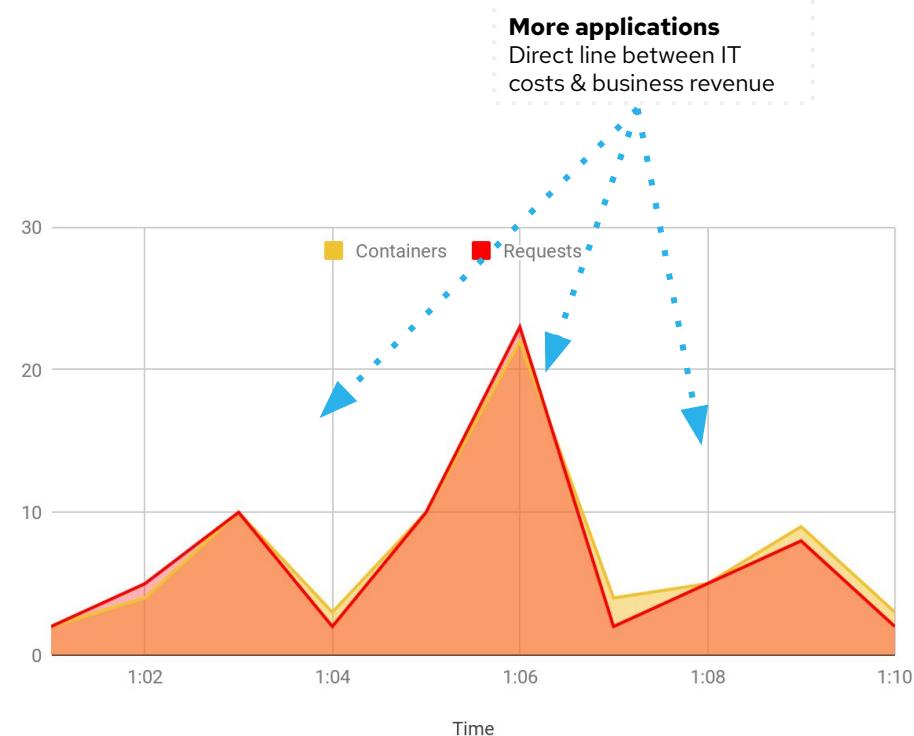
Time in capacity planning
IT cost of idle resources



NOT Serverless

Under provisioning

Lost business revenue
Poor quality of service



with Serverless

Choosing the Right Tool

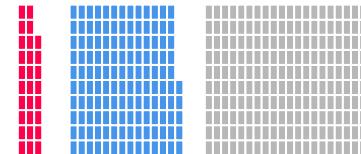
Your Team



The Ecosystem



Performance





Developer experience

</> Java
</> Node
</> Python
</> Go
</> Ruby
</> Ruby on Rails
</> PHP
</> Perl
</> .NET Core

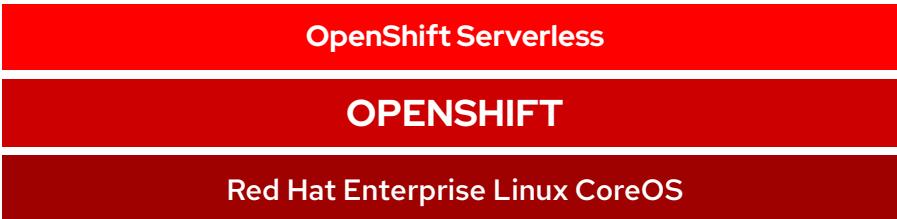
```
$ kn service create --image=
```



CLI



UI



Physical



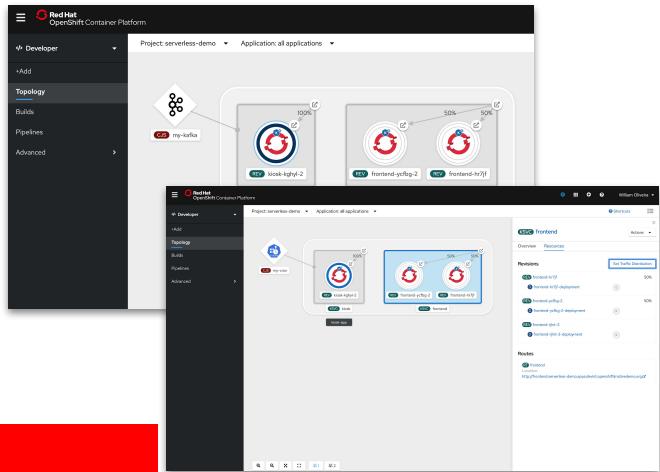
Virtual



Private cloud



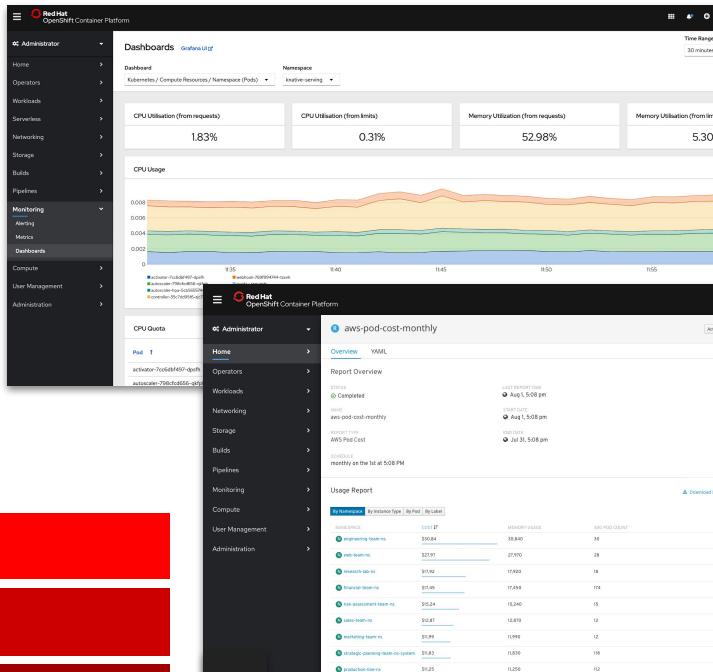
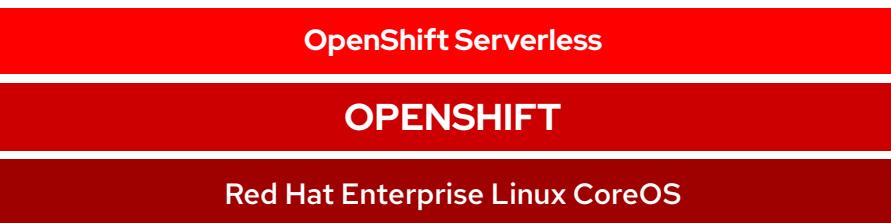
Public cloud





Admin experience

- Monitoring, Metering and Logging
- Disconnected install support (air-gapped)
- Egress proxy with TLS support
- Over the air updates and patches



Physical



Virtual



Private cloud



Public cloud



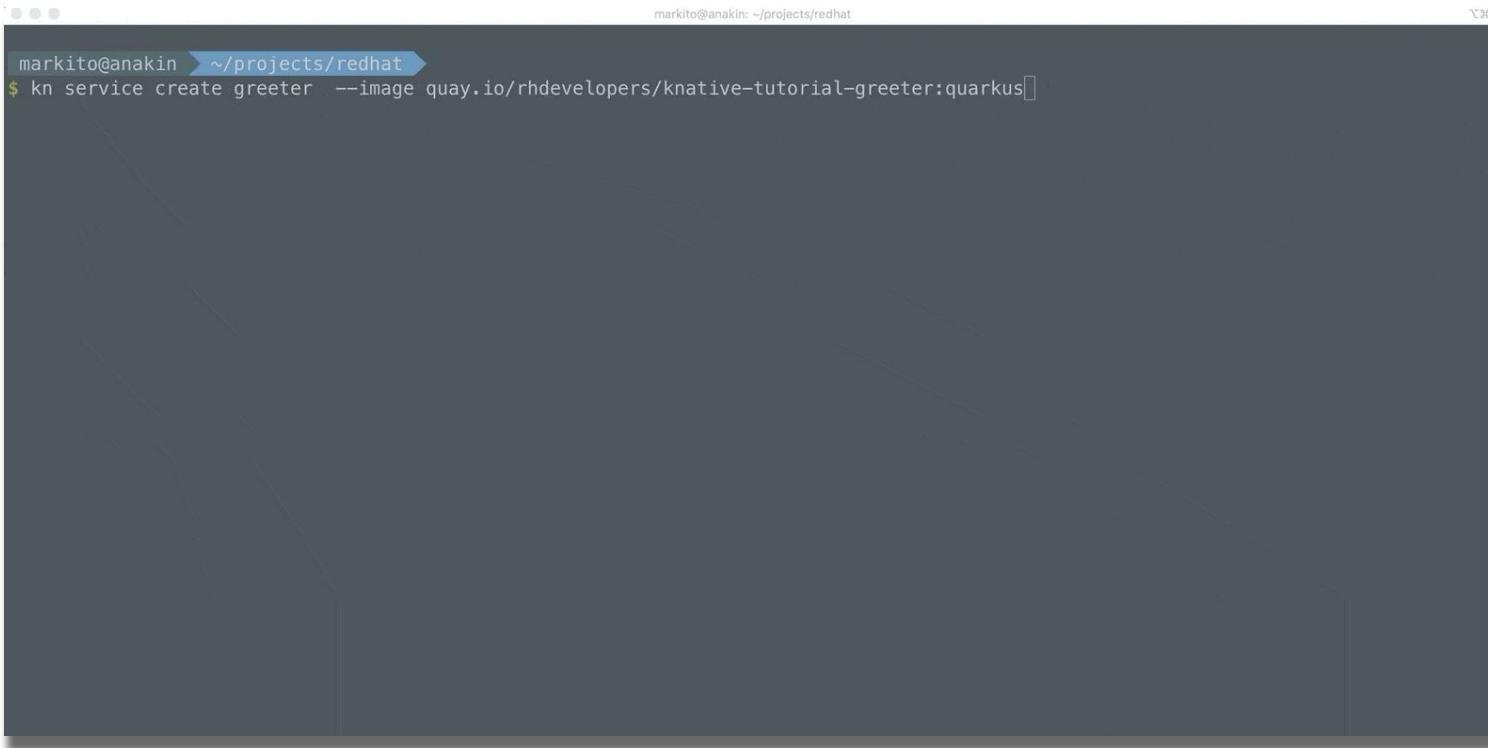
Command Line Experience

```
$ kn service create myService --image=[registry/mycontainer:v1] --min-scale=1 --max-scale=100
```

```
$ kn service update myService --traffic myService-rev1=50,myService-rev2=50
```

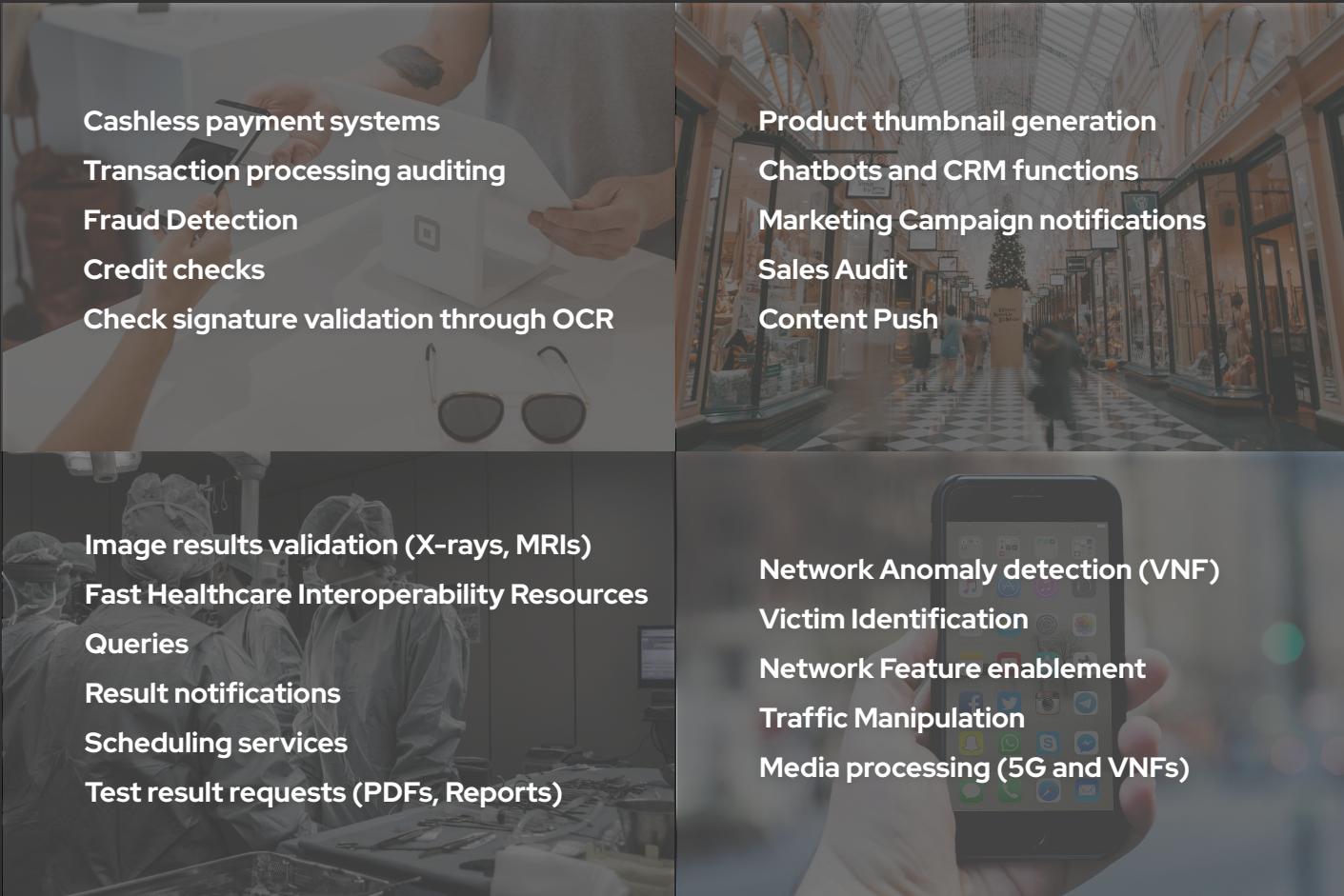
```
$ kn source cronjob create my-cron --schedule "* * * * */1" --data "ping" --sink svc:myService
```

Hello World with Quarkus!



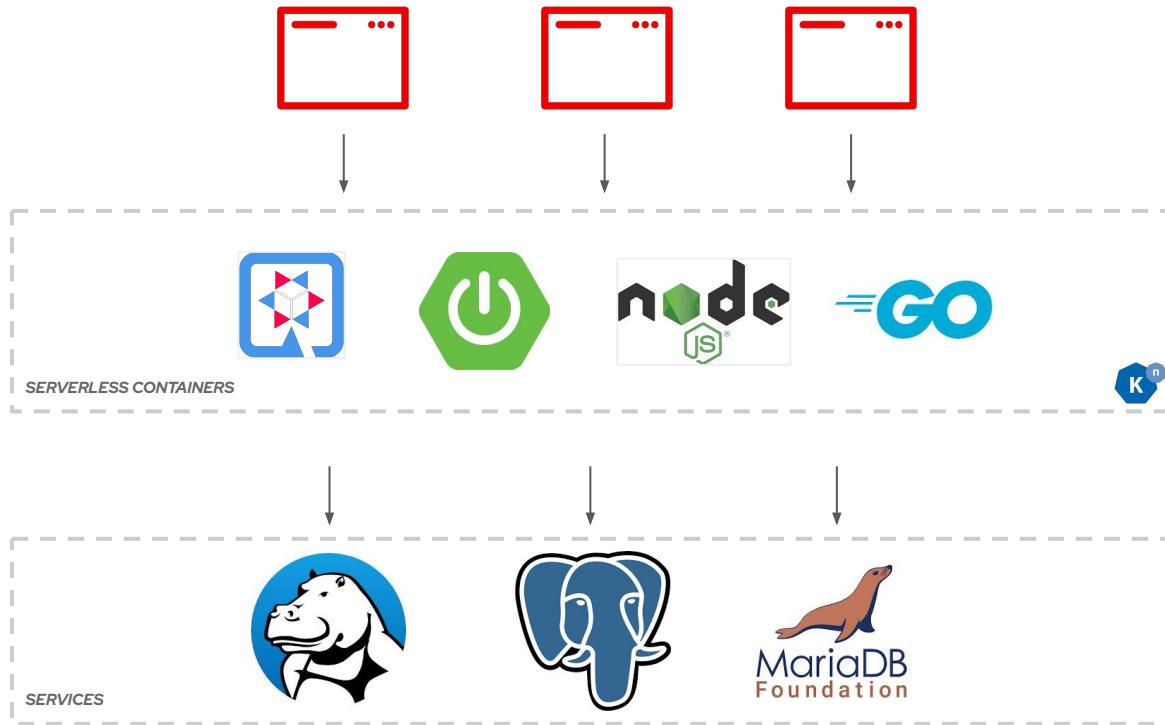
A dark terminal window with a light gray border. The title bar says "markito@anakin ~/projects/redhat". The command line shows the user typing "\$ kn service create greeter --image quay.io/rhdevelopers/knative-tutorial-greeter:quarkus". The background of the terminal is dark, and the text is white.

```
markito@anakin ~/projects/redhat
$ kn service create greeter --image quay.io/rhdevelopers/knative-tutorial-greeter:quarkus
```





Web Applications and APIs



Language or runtime of your choice:

OpenJDK



python™



django



VERT.X



RAILS

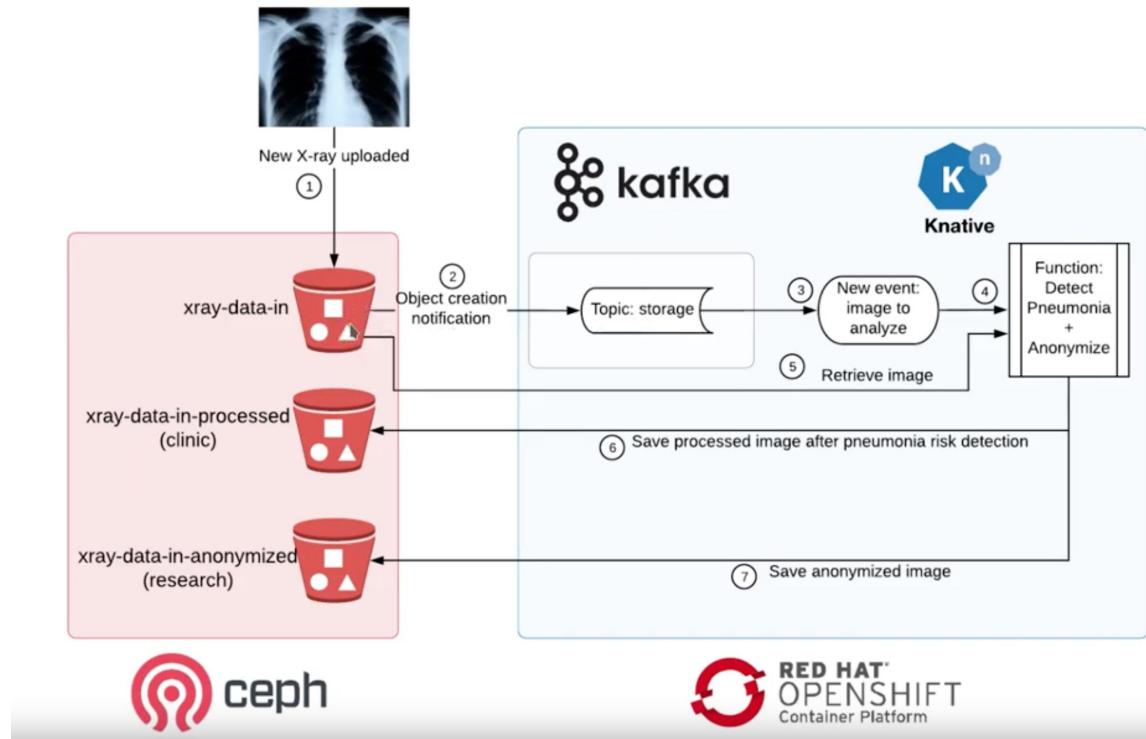




Data Transformation

Common Use cases

- Image analysis* (medical, AI/ML, media)
- Video format transcoding
- File type conversion (financial, medical)
- Data extraction
- Lightweight Data Transformation
- Invoice Generation



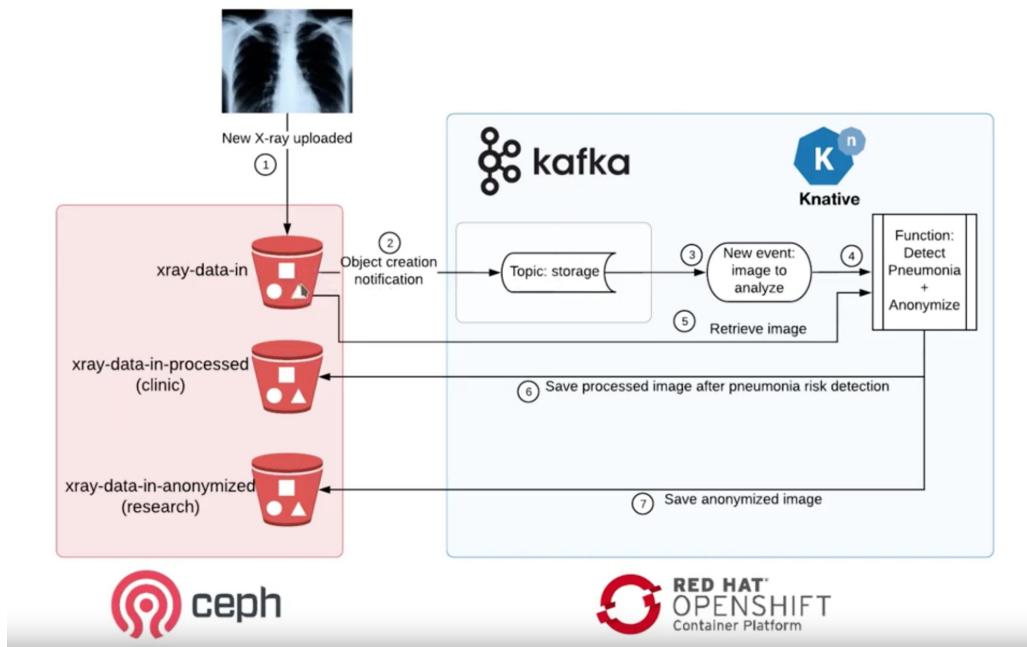


Ceph + Kafka + Serverless

"Automated AI/ML Data Pipelines"

Common Use cases

- Image analysis*
(medical, AI/ML, media)
- Video format transcoding
- File type conversion
(financial, medical)
- Data extraction
- Data Transformation
- Invoice Generation





Building on OpenShift Serverless with Red Hat Services

Connected Services

How Knative services interact with the outside world.



Service Orchestrator

Composing multiple services together into an application.



Event Streaming

All modern architectures need some Kafka.



API Gateway

Next gen APIs still require management.



Implementing Services

Functions, languages, and the vagaries of cold starts.



The Dirty Word in Serverless

Yep, you still need state to handle long-lived orchestration.

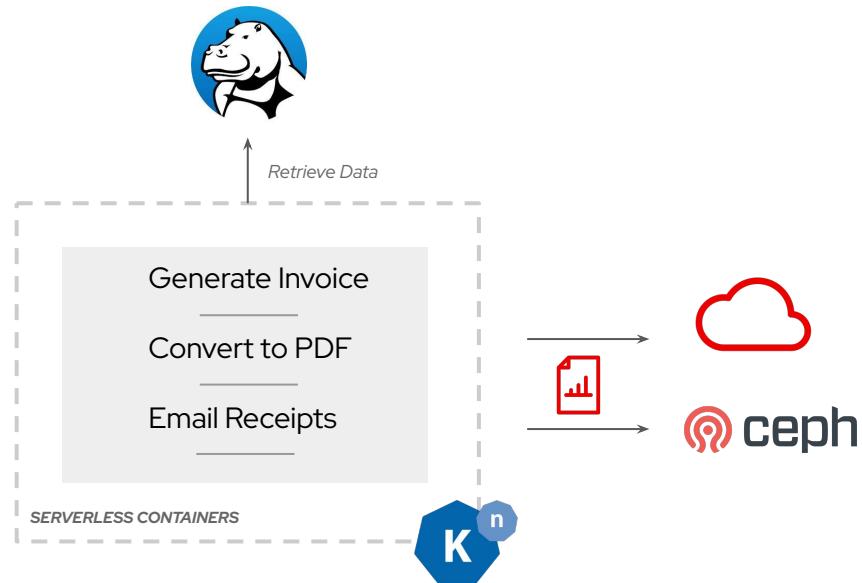




Scheduled Apps Cron Jobs



- Every Hour →
- Every Day →
- Every Week →
- Every Month →

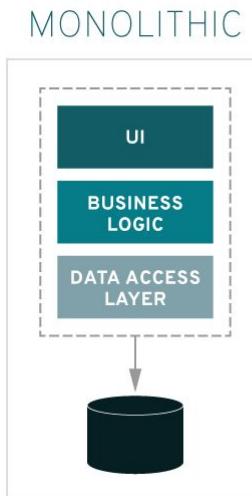


Optional - ServiceMesh / Istio

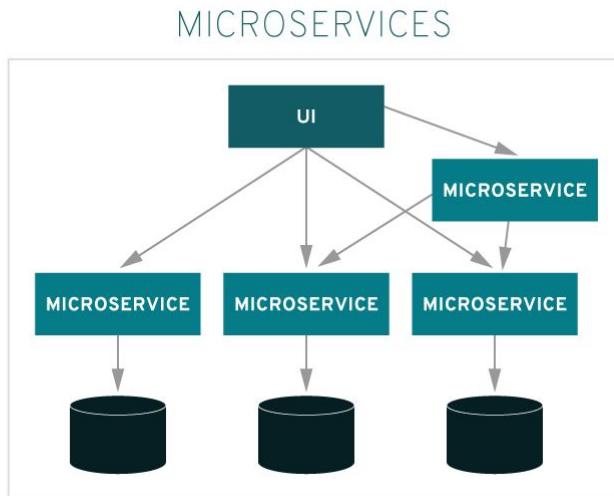
What are Microservices?

an architectural style that structures an application as a collection of services

- Single purpose
- Independently deployable
- Have their context bound to a biz domain
- Owned by a small team
- Often stateless

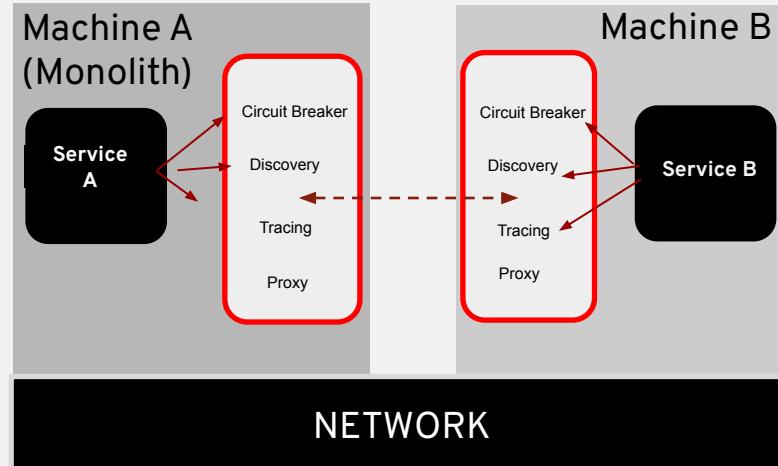


VS.

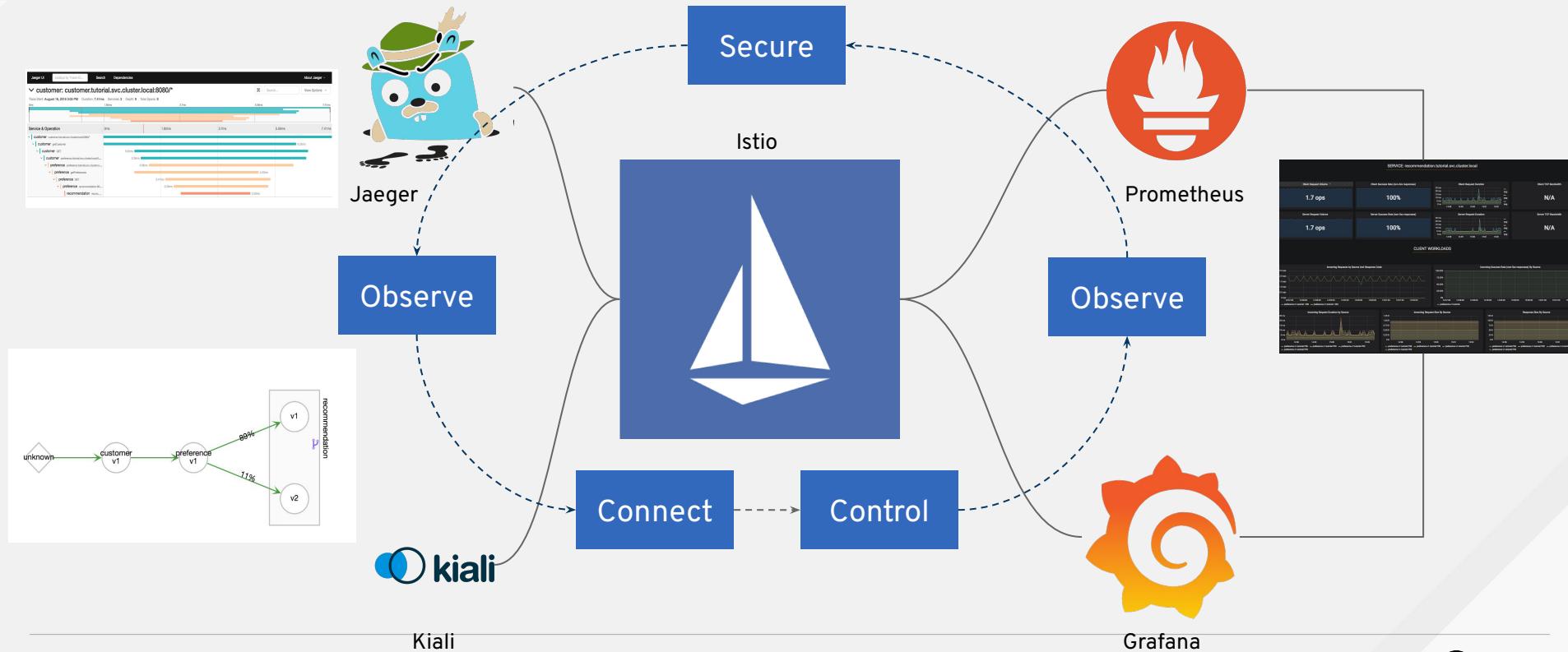


OVERVIEW

WHAT IS A SERVICE MESH ?

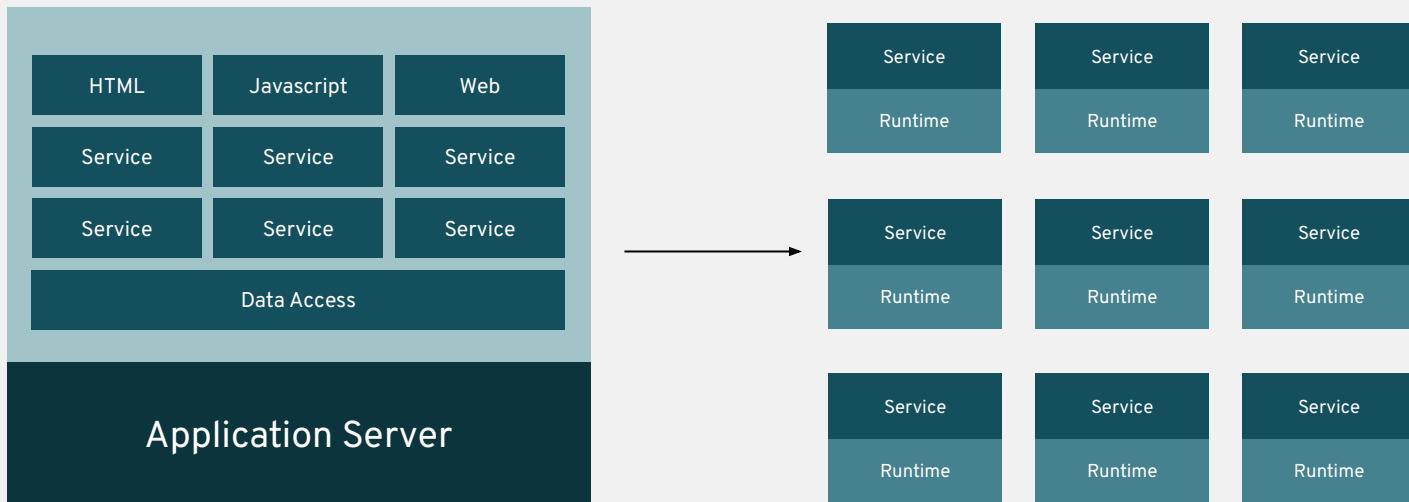


SERVICE MESH ECOSYSTEM



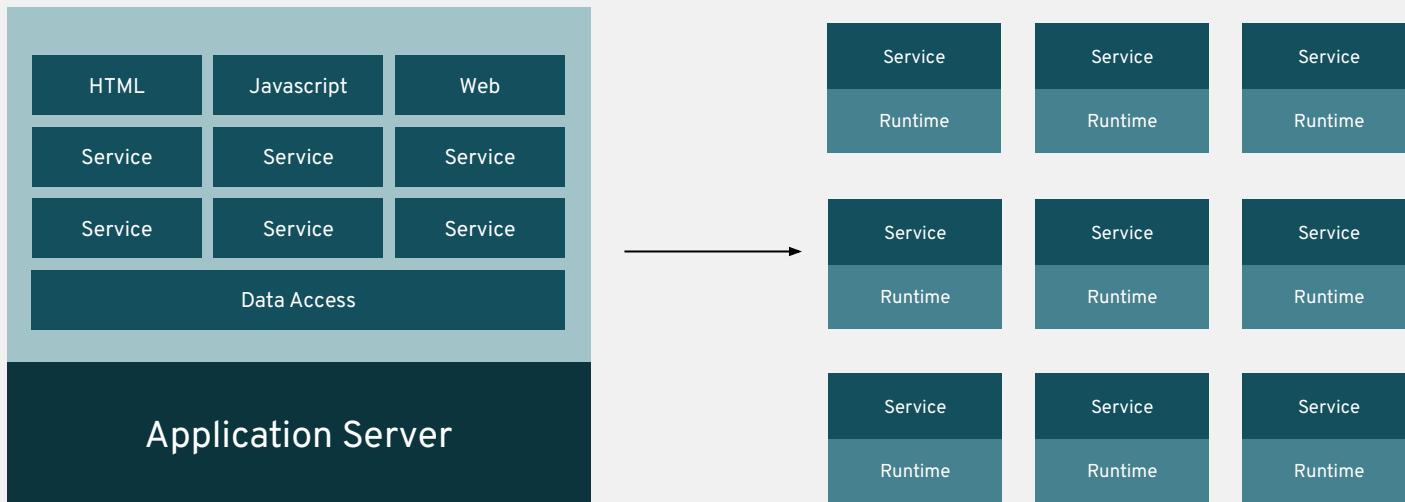
UNDER THE HOOD

MICROSERVICES ARCHITECTURE

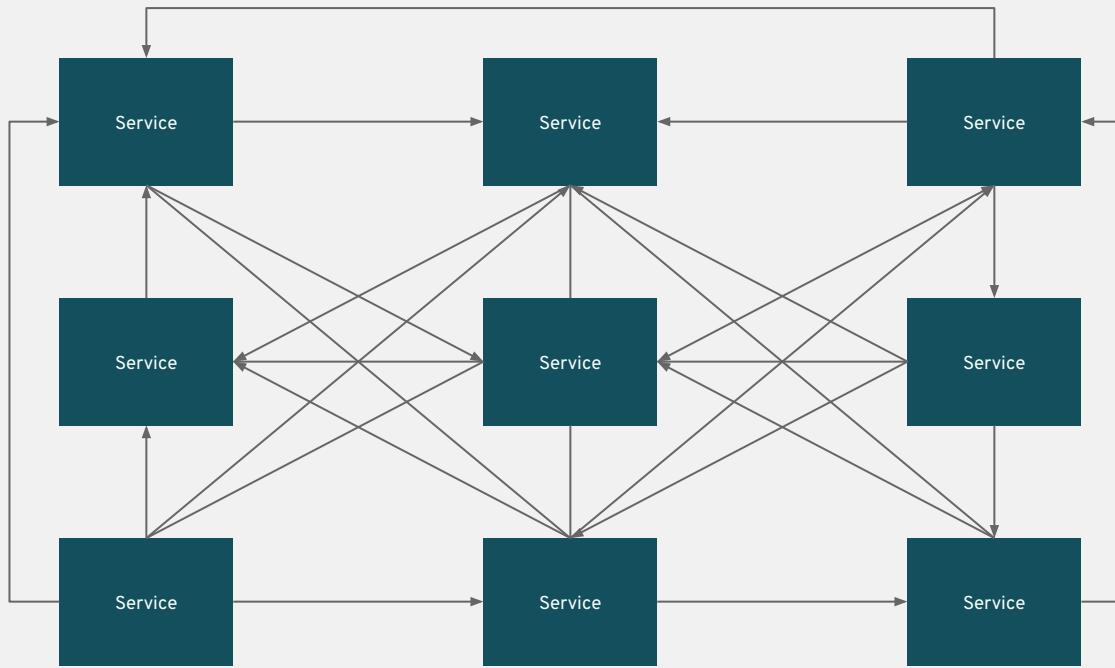


MICROSERVICES ARCHITECTURE

DISTRIBUTED



DISTRIBUTED ARCHITECTURE





HOW TO DEAL WITH THE COMPLEXITY?

DEPLOYMENT

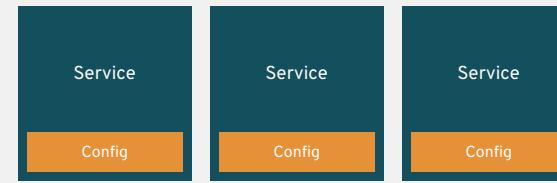
Service
Container

Service
Container

Service
Container

INFRASTRUCTURE

CONFIGURATION



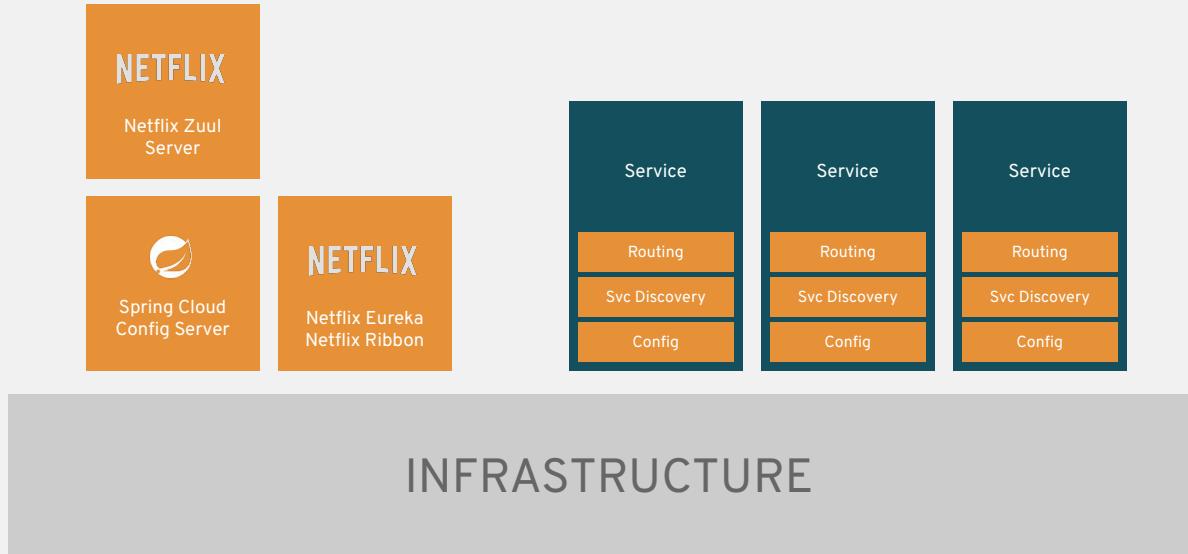
INFRASTRUCTURE

SERVICE DISCOVERY



INFRASTRUCTURE

DYNAMIC ROUTING

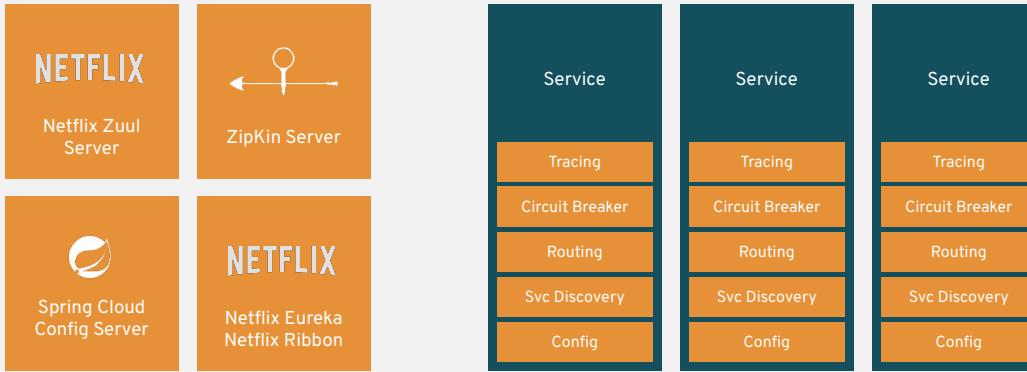


FAULT TOLERANCE



INFRASTRUCTURE

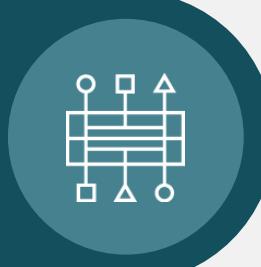
TRACING AND VISIBILITY



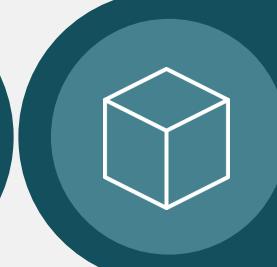
INFRASTRUCTURE

WHAT ABOUT...?

POLYGLOT
APPS

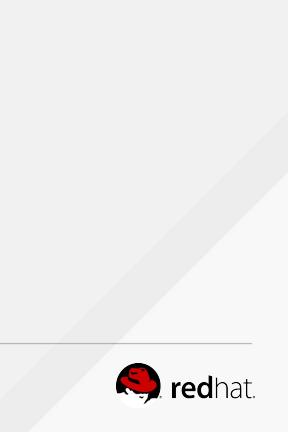


EXISTING
APPS





**THERE SHOULD BE A
BETTER WAY**



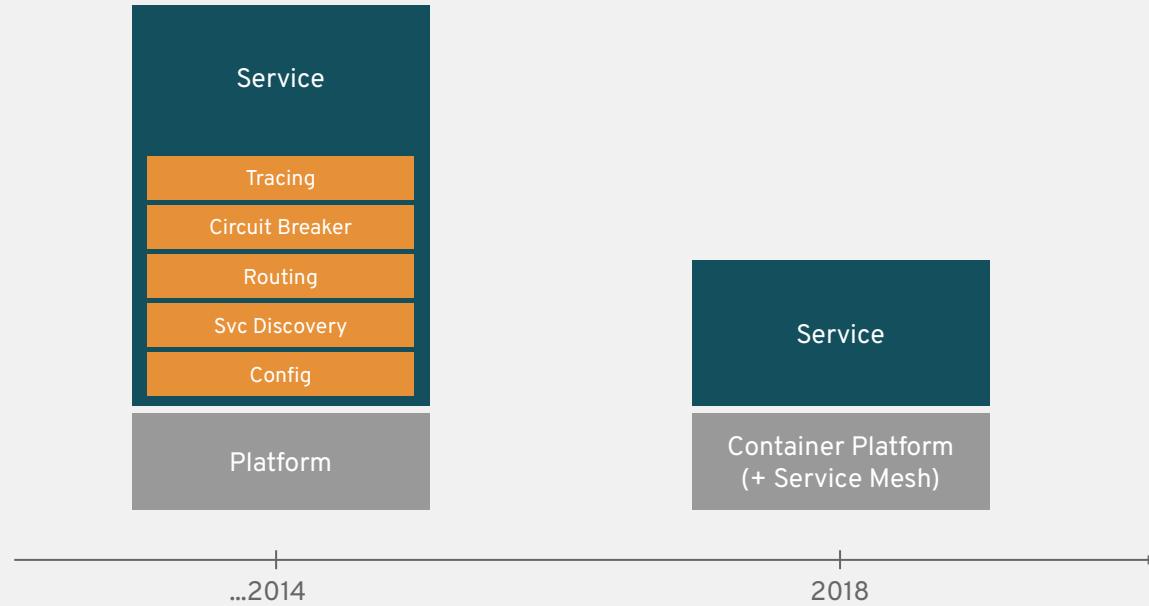


ADDRESS THE COMPLEXITY IN THE INFRASTRUCTURE

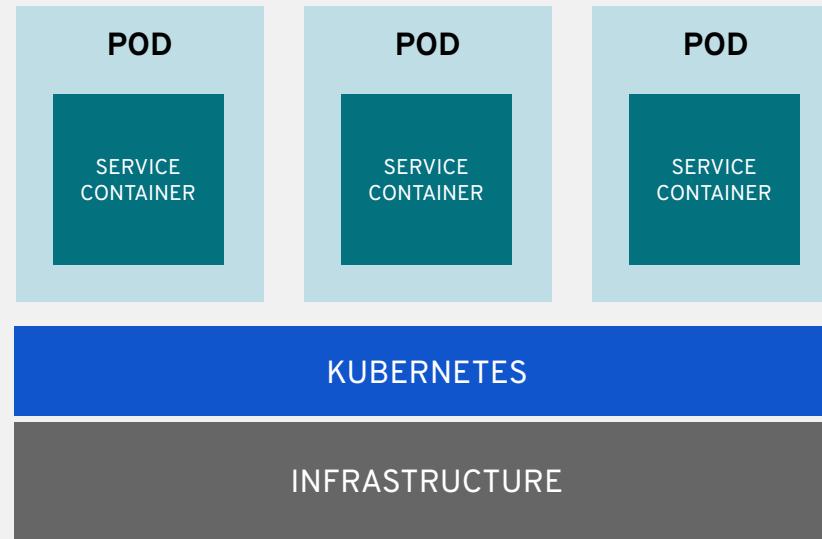
SERVICE MESH

A dedicated infrastructure layer for
service-to-service communications

MICROSERVICES EVOLUTION



AUTOMATING CONTAINER DEPLOYMENT



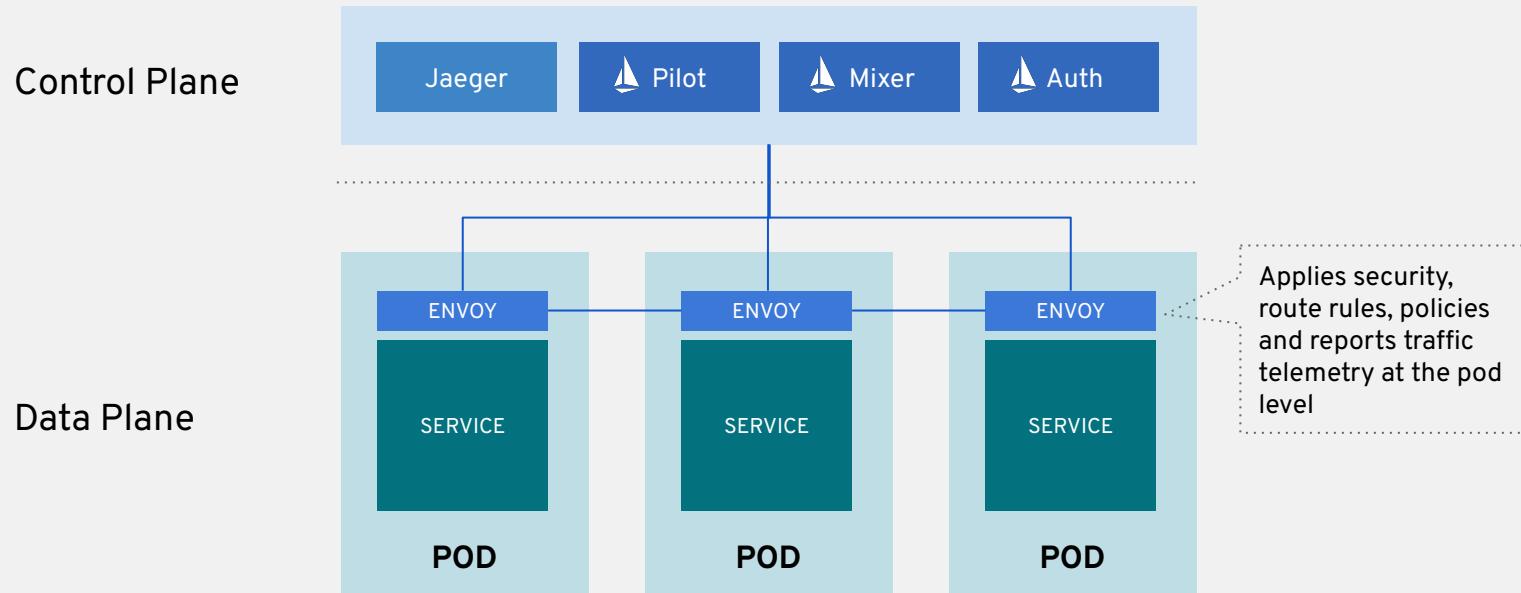
SIDECARS



- Two or more containers deployed to same pod
- Share
 - Same
 - Namespace
 - Pod IP
 - Shared lifecycle
- Used to enhance the co-located containers
- Istio Proxy (L7 Proxy)
 - Proxy all network traffic in and out of the app container

Source: <http://blog.kubernetes.io/2015/06/the-distributed-system-toolkit-patterns.html>

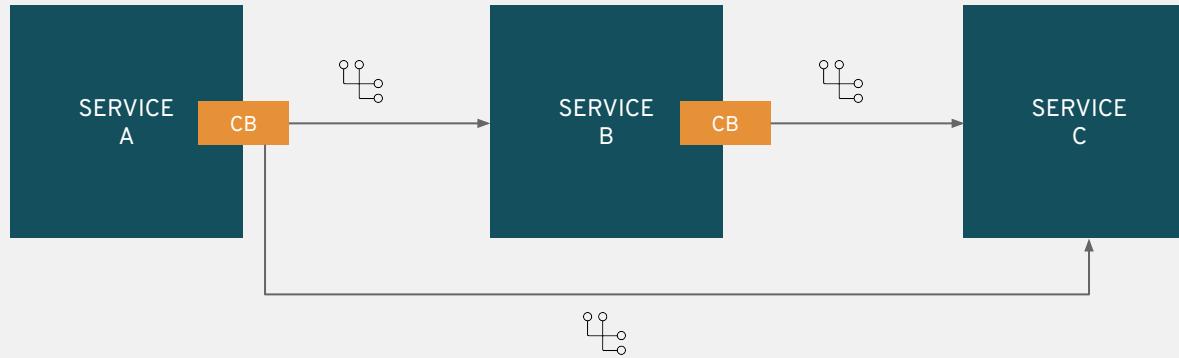
SERVICE MESH ARCHITECTURE



MAJOR FUNCTIONALITY

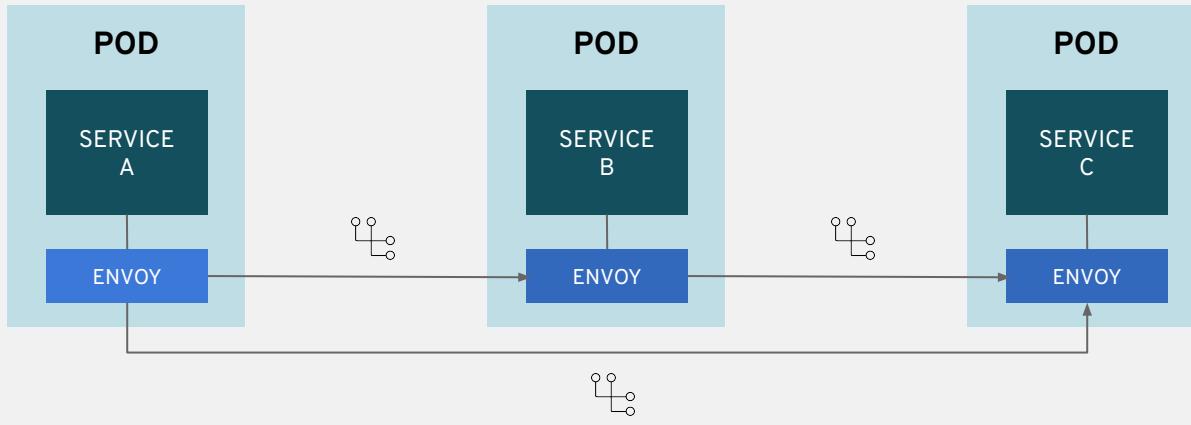
FAULT TOLERANCE

CIRCUIT BREAKERS WITHOUT ISTIO



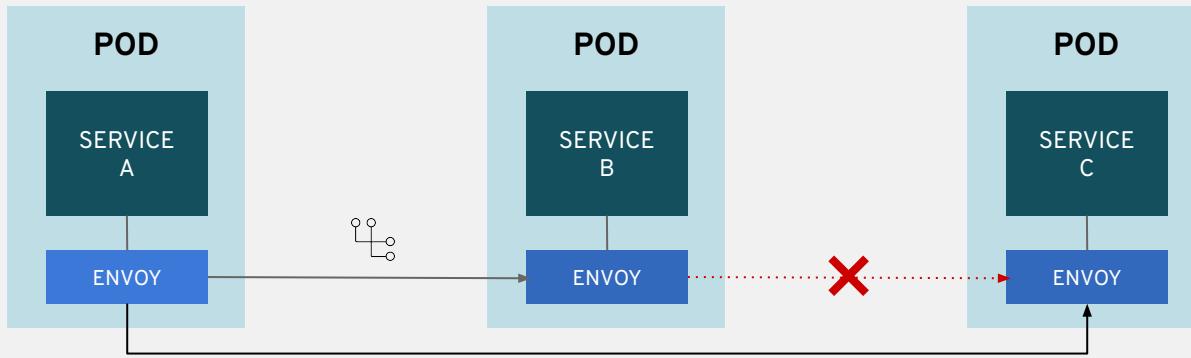
coupled to the service code

CIRCUIT BREAKERS WITH ISTIO



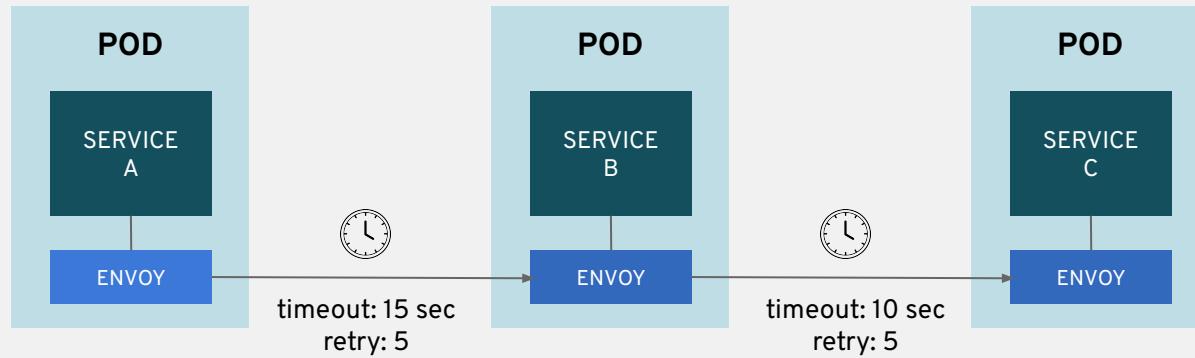
transparent to the services

CIRCUIT BREAKERS WITH ISTIO



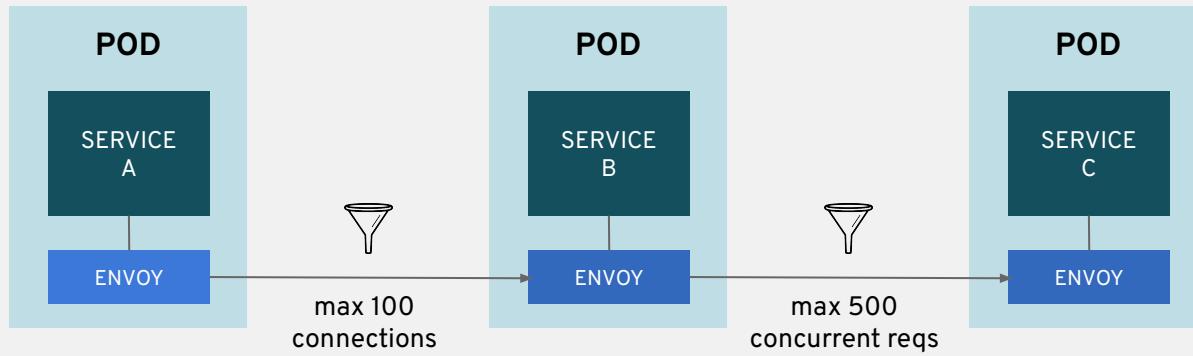
improved response time with global circuit status

TIMEOUTS AND RETRIES WITH ISTIO



configure timeouts and retries, transparent to the services

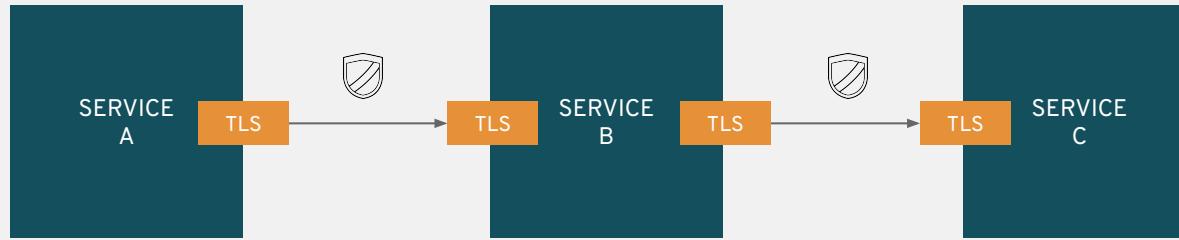
RATE LIMITING WITH ISTIO



limit invocation rates, transparent to the services

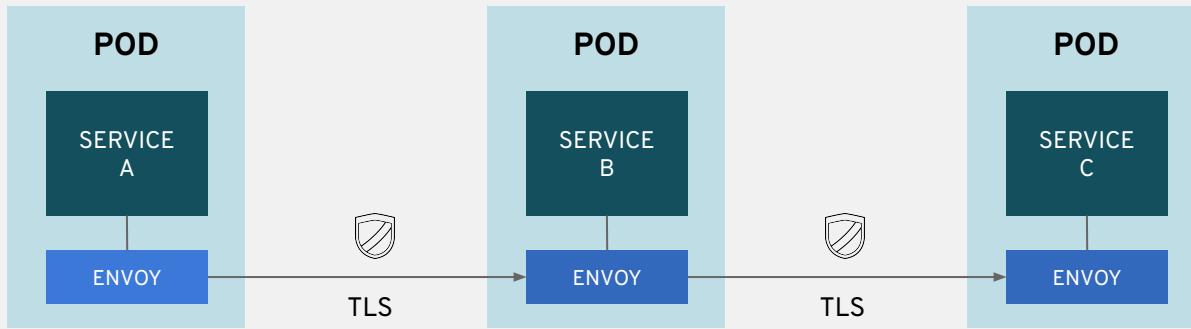
SERVICE SECURITY

SECURE COMMUNICATION WITHOUT ISTIO



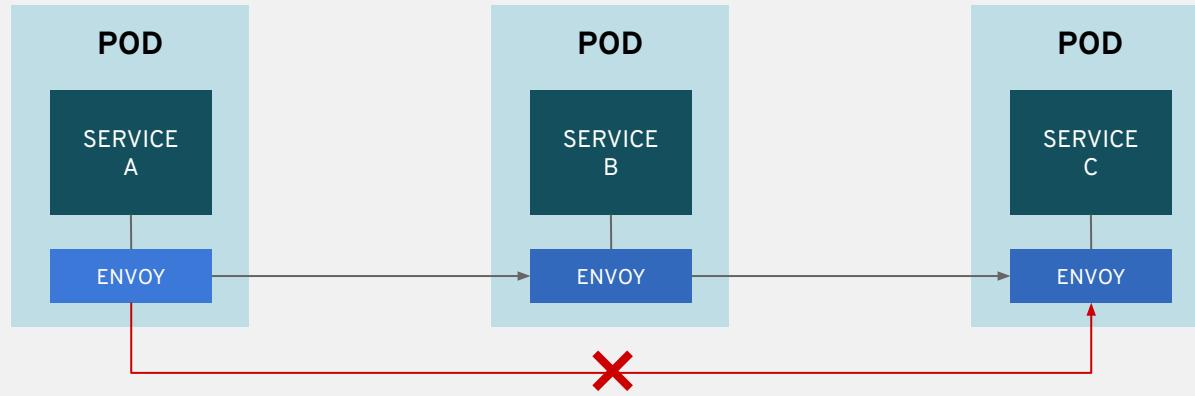
coupled to the service code

SECURE COMMUNICATION WITH ISTIO



mutual TLS authentication, transparent to the services

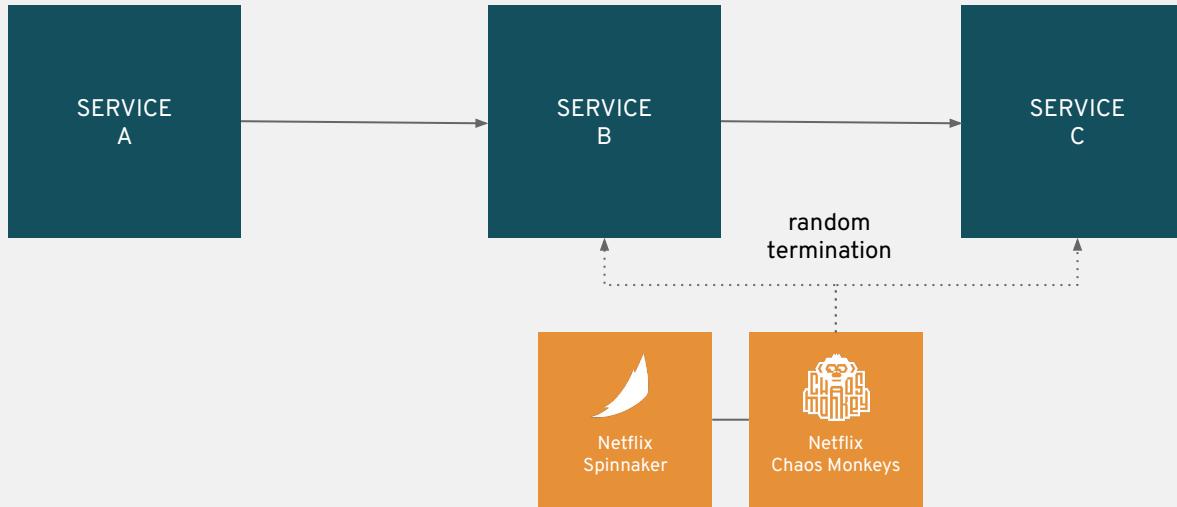
CONTROL SERVICE ACCESS WITH ISTIO



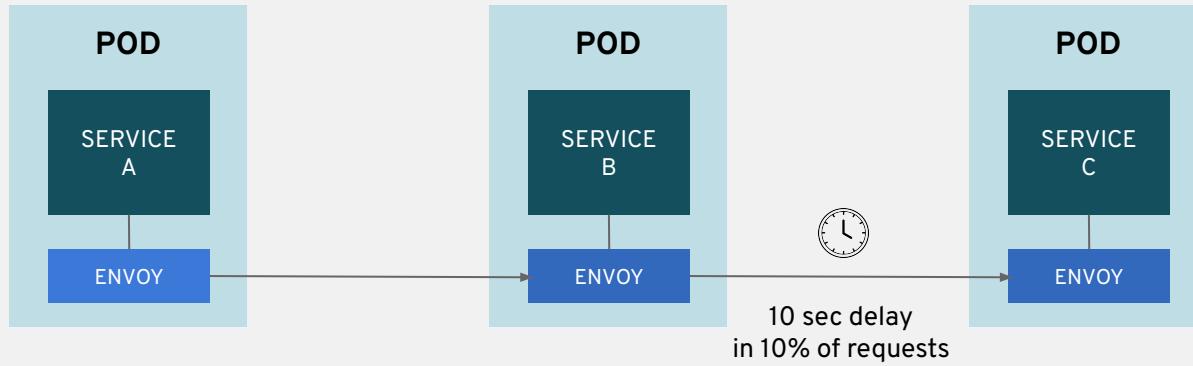
control the service access flow, transparent to the services

CHAOS ENGINEERING

CHAOS ENGINEERING WITHOUT ISTIO

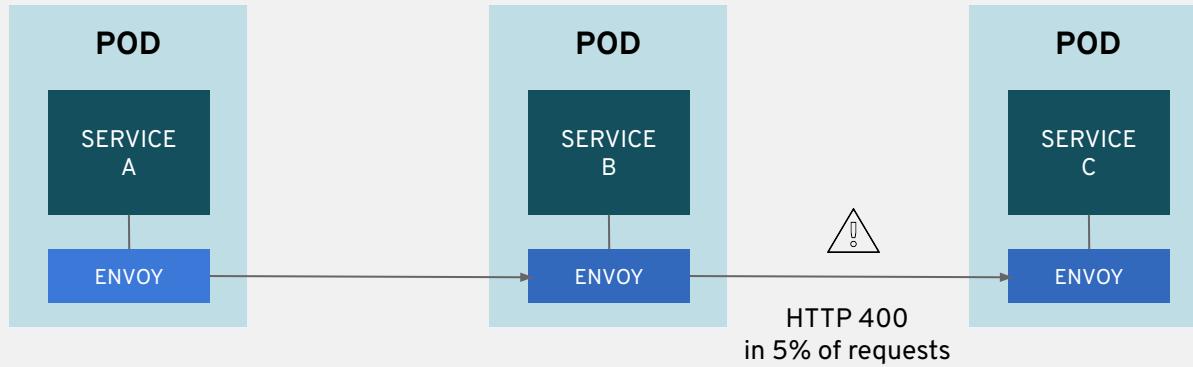


CHAOS ENGINEERING WITH ISTIO



inject delays, transparent to the services

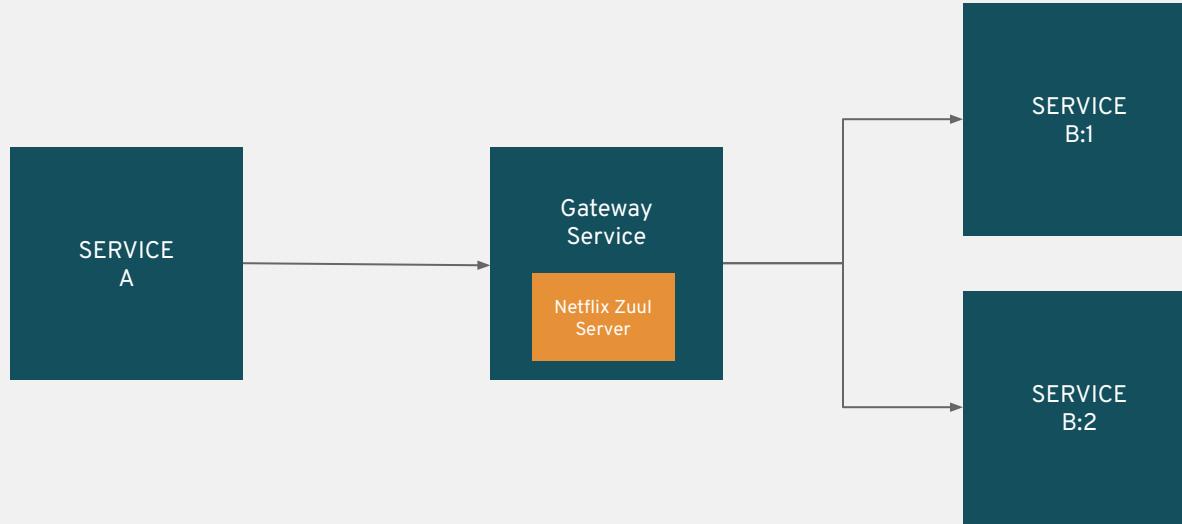
CHAOS ENGINEERING WITH ISTIO



inject protocol-specific errors, transparent to the services

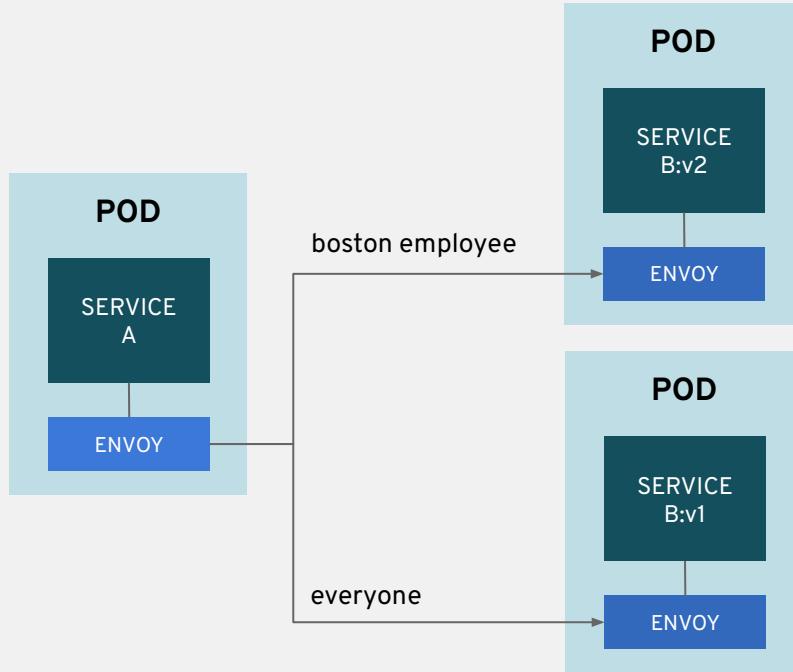
DYNAMIC ROUTING

DYNAMIC ROUTING WITHOUT ISTIO

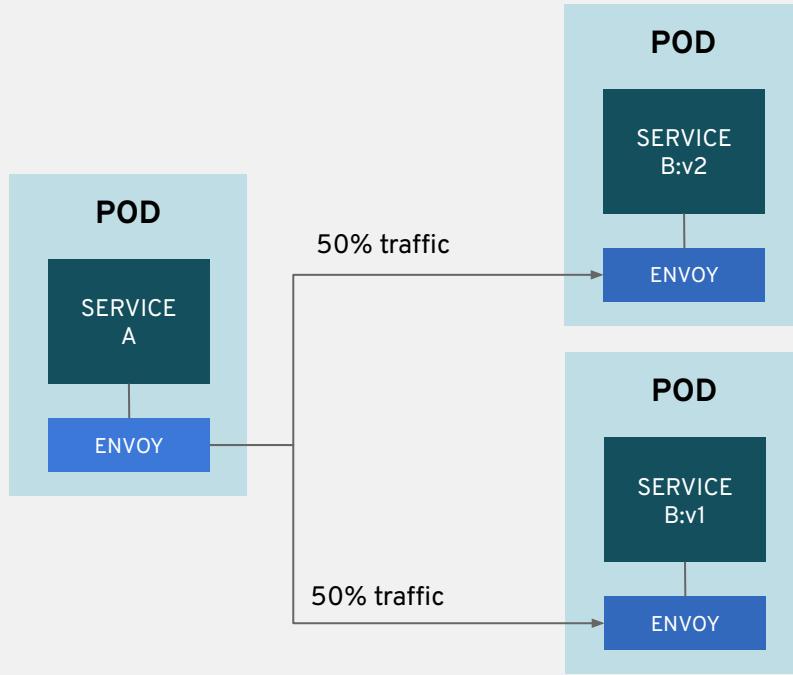


custom code to enable dynamic routing

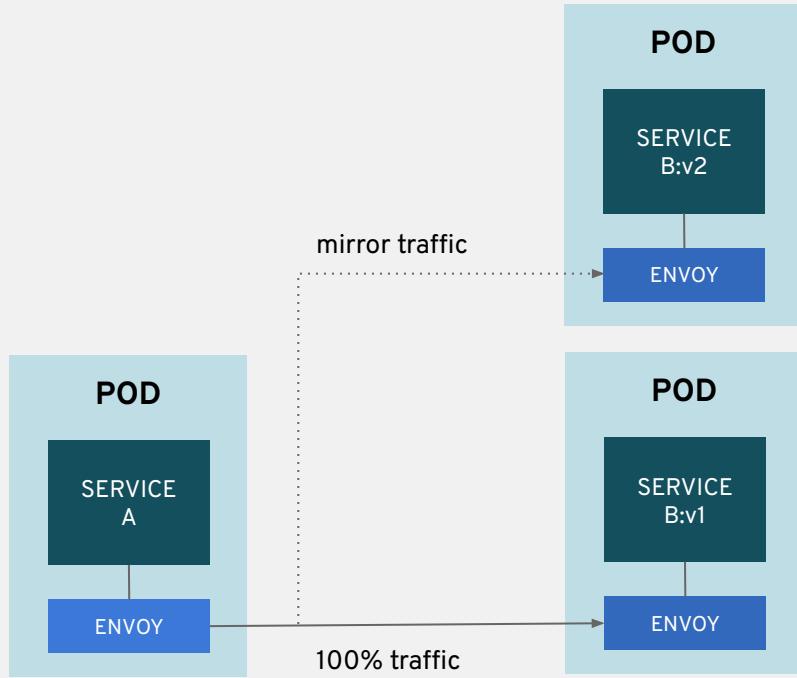
CANARY DEPLOYMENT WITH ISTIO



A/B DEPLOYMENT WITH ISTIO



DARK LAUNCHES WITH ISTIO



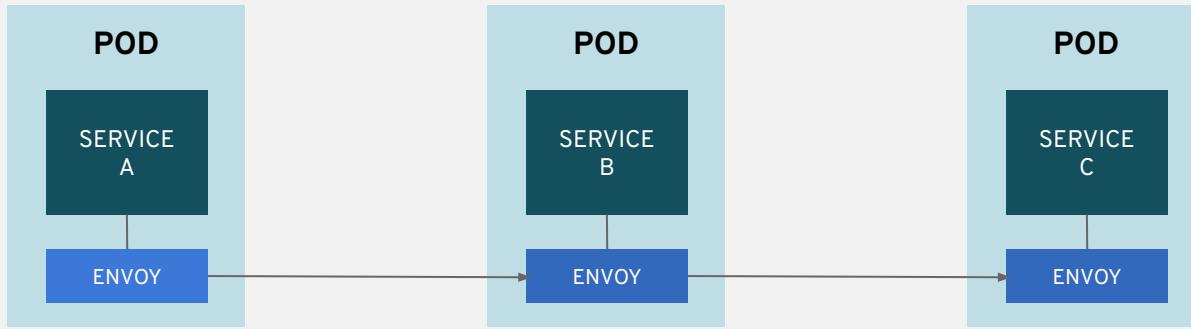
DISTRIBUTED TRACING (JAEGER)

DISTRIBUTED TRACING WITHOUT ISTIO



code to enable dynamic tracing

DISTRIBUTED TRACING WITH ISTIO & JAEGER



discovers service relationships and process times,
transparent to the services



SERVICE MESH OBSERVABILITY (KIALI)

kiali

Overview

Graph

Applications

Workloads

Services

Istio Config

Distributed Trac...

Namespace: bookinfo

Graph

Display Edge Labels Graph Type Versioned app Find... Hide... Fetching Last min Every 15 sec

Feb 18, 16:07:37 ... Feb 18, 16:08:37

Current Graph:
9 apps
5 services
14 edges

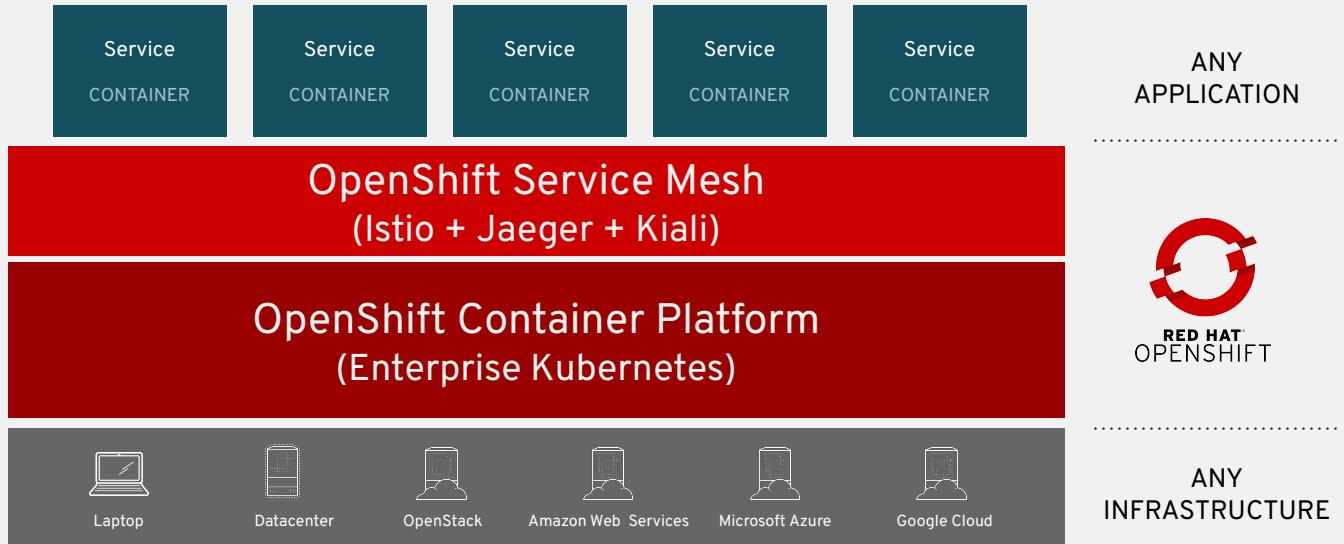
HTTP Traffic (requests per second):
Total %Success %Error
3.68 100.00 0.00

HTTP - Total Request Traffic min / max:
RPS: 3.60 / 3.60 , %Error 0.00 / 0.00

TCP - Total Traffic - min / max:
Sent: 143.00 / 143.00 B/s
Received: 115.67 / 115.67 B/s

+ - X 1 X 2 Legend

DISTRIBUTED SERVICES PLATFORM



Istio - Caveats

- Caveats
 - Be careful, istio is NOT a ServiceBus
 - Be careful, istio is NOT an API Manager
 - Be careful, istio is NOT meant to be a router
 - Be careful, istio is NOT a BPM tool
- Istio is meant to help solving some challenges
 - Centrally encryption / management of internal service communication
 - Distributed networking / communication of services
 - Increase Fault tolerance
 - Fault simulation

**... and now it's
time to prove
your knowledge!**

Week #4 - Homework

- You have 2 different options to complete this course
 - **Either** create a presentation and upload it **OR**
 - Create a demo, make a screencast and upload it

Please upload your work until end of this week (Friday, 15th of October 2021)

(if you have urgent other important things to do this week, please get in touch with me)

Homework - The presentation

- Think about a use case / a customer scenario
- Describe the use case in 1-2 slides
- Describe your solution in 1-2 slides
- Create an architecture diagram of the solution and describe it in 1-2 slides
- Describe the benefits of using your solution in 1-2 slides
- Upload it to OPEN as PDF file

Homework - The Demo

- Create your own demo script, which should contain
 - A scenario you'd like to address
 - Take one of the technologies you've learned over the course
 - Odo, Helm, Kustomize, Tekton, GitOps, Serverless, ServiceMesh...
 - Describe, why you have chosen this technology and why you think this would help in the scenario
 - Use a screen recorder tool to record your demo with your voice
 - The whole recording should not be longer than 5 minutes
- Upload it to OPEN

Homework - Upload your work

The screenshot shows the Red Hat Partner Connect interface. At the top, there's a navigation bar with links for 'Partner Home', 'Catalog', 'My Learning', and 'Reporting'. A user profile for 'Wanja Pernath' is visible on the right. Below the navigation, the main content area displays the 'OpenShift Developer Distance Learning Program - Sales Engineer (EMEA)' course. The course title is bolded, and the subtitle 'Sales Engineer (EMEA)' is in parentheses. A progress bar indicates 73% completion. Below the title, there's a brief description of the program, mentioning it's a new OpenShift learning experience for Developers. It highlights that Red Hat has put together a comprehensive learning program for developers to learn everything they need to understand the concepts, architectural principles, and components of Red Hat products. The course aims to successfully discuss and develop OpenShift at a customer site from a developer perspective. A list of learning objectives follows, including topics like the benefits of OpenShift over plain Kubernetes, how to effectively and efficiently demo OpenShift, and how to start coding with OpenShift in various languages. A note states that with this Distance Learning Program, you will get an overview of the whole OpenShift ecosystem for developers by gaining knowledge around these topics. The learning path is designed for the EMEA region and audience. A 'Detailed Overview' link is provided. The 'WEEK 1: Introduction to OpenShift for Developers' section is expanded, showing five mandatory modules: 'Introduction to Red Hat OpenShift for Developers' (2 hours, Not Completed), 'Red Hat Foundations' (2 hours, Incomplete), 'Getting Started with Red Hat OpenShift for Developers' (15 minutes, Completed Feb-14-21), 'Using the CLI to Manage Resource Objects' (15 minutes, In Progress), and 'Deploying Applications From Source' (15 minutes, Completed Mar-03-21). Each module has a 'Launch' button next to it. At the bottom of the page, there are 'RED HAT' and 'GUIDED HELP' buttons.

- Open your OPEN progress
- Scroll down to Week #4

Homework - Upload your work

The screenshot shows a web-based learning platform interface for Red Hat OPEN - WPERNATH. The main content area displays a list of learning modules categorized by week:

- WEEK 3: CI/CD** (All Mandatory, 4 items):
 - Operator SDK with Helm (30 minutes, Completed Mar-05-21)
 - Operator SDK with Go (30 minutes, Completed Mar-05-21)
 - CI/CD with Red Hat OpenShift (2 hours, Not Enrolled)
 - GITOPS Integration with Red Hat OpenShift (30 minutes, Completed Mar-05-21)
- WEEK 4: Summary and Wrap Up** (All Mandatory, 2 items):
 - Summary and Wrap up and Next Steps (2 hours, Not Enrolled)
 - OpenShift Developer Distance Learning Program (EMEA) - Homework Assignment** (In Progress, View button highlighted with a red box and arrow)
- OPTIONAL LEARNING** (All Optional, 2 items):
 - Getting Started with OpenShift Serverless (30 minutes)
 - Red Hat OpenShift ServiceMesh Interactive Learning Scenarios (9 hours)

At the bottom of the interface, there are two buttons: "RED HAT" and "GUIDED HELP".

- Click on “View” of the entry “Homework Assignment”

Homework - Upload your work

The screenshot shows the Red Hat Partner Connect interface for the OpenShift Developer Distance Learning Program (EMEA). The page title is "OpenShift Developer Distance Learning Program (EMEA) - Homework Assignment". On the left, there are sections for "Submission Files" (with an "Upload" button highlighted by a red box and arrow), "Submission Comments" (with an "Add Comment" button), and "Learning Path" (listing "OpenShift Developer Distance Learning Program - Sales Engineer (EMEA)" and "My Personal Learning Path"). On the right, there is a "Details" sidebar with the following information:

Identifier	44043053
Date Enrolled	Mar-01-21 17:29 CET
Language	English (en)
Submissions From	Jan-11-21
Status	In Progress
Grading Type	Simple - No Score

- Click on Upload
- Choose your File (presentation or video)

Homework - Upload your work

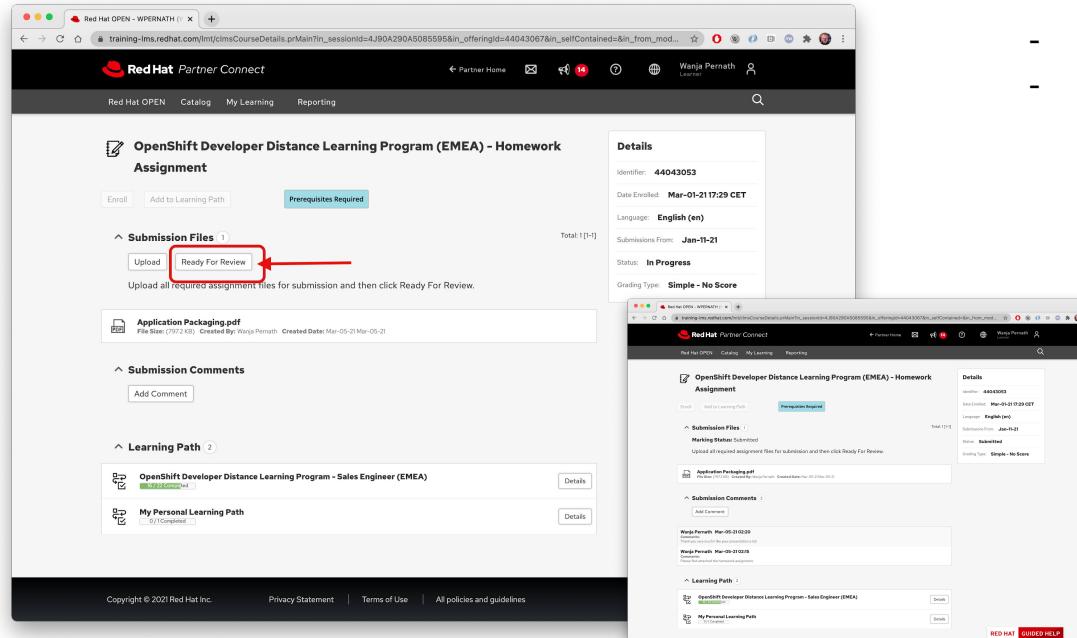
The screenshot shows a web browser window for 'Red Hat OPEN - WPERNATH'. The URL is https://training-lms.redhat.com/lmt/clmsCourseDetails.prMain?in_sessionId=4J90A290A5085595&in_offeringId=44043067&in_selfContained=&in_from_mod.... The page title is 'Red Hat Partner Connect'.

The main content area displays a 'Homework Assignment' for the 'OpenShift Developer Distance Learning Program (EMEA)'. It includes sections for 'Submission Files' (with a PDF file attached) and 'Submission Comments' (with a message from 'Wanja Pernath'). A red box highlights these two sections, with red arrows labeled 1 and 2 pointing to them respectively.

At the bottom of the page, there are links for 'Copyright © 2021 Red Hat Inc.', 'Privacy Statement', 'Terms of Use', 'All policies and guidelines', 'RED HAT', and 'GUIDED HELP'.

- Upload your file
- You should see it here
- Optionally, add comments to explain what your homework is about

Homework - Upload your work



The screenshot shows the Red Hat Partner Connect interface for the OpenShift Developer Distance Learning Program (EMEA). The main page displays the assignment details and submission files. A red box highlights the 'Ready For Review' button in the 'Submission Files' section.

Assignment Details:

- Identifier: 44043053
- Date Enrolled: Mar-01-21 17:29 CET
- Language: English (en)
- Submissions From: Jan-11-21
- Status: In Progress
- Grading Type: Simple - No Score

Submission Files:

- Upload (button)
- Ready For Review (button, highlighted with a red box)

Upload all required assignment files for submission and then click Ready For Review.

Submission Comments:

- Add Comment (button)

Learning Path:

- OpenShift Developer Distance Learning Program - Sales Engineer (EMEA)
- My Personal Learning Path

- Upload your file
- Click on “Ready for Review” once you’re done

Homework - Let me have a look at your work

Now I need to have a look at your work. If I need to know anything, have questions or something, I am going to comment it. → You'll be getting an email

If I am accepting it, you're also going to get an email

This should be done by end of the week (depending on how many of you are going to submit your homework)

Homework - Upload your work

The screenshot shows a web browser window titled "Red Hat OPEN - WPERNATH" displaying the "Optional Learning" section of the Red Hat OpenShift Distance Learning Platform.

Kubernetes API Fundamentals
30 minutes. Completed Mar-05-21
Before diving into the Operator Framework, this section will give an overview of Kubernetes API fundamentals.

Operator Lifecycle Manager
30 minutes. Completed Mar-05-21
The Operator Lifecycle Manager project is a component of the Operator Framework, an open source toolkit to manage Kuber...

ETCD Operator
30 minutes. Completed Mar-05-21
The etcd operator manages etcd clusters deployed to Kubernetes and automates tasks related to operating an etcd cluster.

Red Hat Ansible Operator Overview
30 minutes. Completed Mar-05-21
This section will give a brief overview of the Ansible Operator with a step-by-step example of developing an Ansible Operator u...

Operator SDK with Helm
30 minutes. Completed Mar-05-21
We will now focus on the easiest way to get started developing an Operator: Helm.

Operator SDK with Go
30 minutes. Completed Mar-05-21
Operators make it easy to manage complex stateful applications on top of Kubernetes.

WEEK 3: CI/CD
All Mandatory (4)

CI/CD with Red Hat OpenShift
2 hours. Not Enrolled
This webinar will help attendees to learn and understand how to successfully do CI/CD with OpenShift. The base for this is Ope...

GitOps Introduction with Red Hat OpenShift
30 minutes. Completed Mar-05-21
Learn how to apply GitOps principles to build and deploy an application on OpenShift.

Multi-Cluster GitOps with Red Hat OpenShift
30 minutes. Completed Mar-05-21
Learn how to use GitOps practices on OpenShift to deploy an application on multiple clusters, configure the application for ea...

Getting Started with Red Hat OpenShift Pipelines
30 minutes. Completed Mar-05-21
In this self-paced tutorial, you will learn how to use OpenShift Pipelines to automate the deployment of your applications.

WEEK 4: Summary and Wrap Up
All Mandatory (2)

Summary and Wrap up and Next Steps
2 hours. Not Enrolled
This webinar will wrap up the Distance Learning Program, recap what attendees have learned over the last weeks and introduce...

OpenShift Developer Distance Learning Program (EMEA) - Homework Assignment
Created - Published Mar-05-21

OPTIONAL LEARNING

RED HAT GUIDED HELP

Summary

Summary

- You've learned a lot over the past weeks
- Goal was to help you understanding the huge ecosystem of OpenShift from a Developer / User perspective

Optional section marker or title

93



Thank you



[linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)



[facebook.com/redhatinc](https://www.facebook.com/redhatinc)



[youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)



twitter.com/RedHat