ISTIO: Service mesh e microservizi

Giuseppe Bonocore - Solution Architect

Ugo Landini - Solution Architect
SHORT HISTORY OF MICROSERVICES

- **NETFLIX to AWS** 2010
- **AWS EC2** 2006
- **Java EE6** 2009
- **DropWizard** 2011
- **Hystricx** 2012
- **Eureka** July 2012
- **Spring Boot** 2013
- **Continuous Integration via XP** 1999
- **Agile Manifesto** 2001
- **DevOps** 2009
- **Ribbon** 2012
- **Microservices marked Assess Thoughtworks Radar** 2012
- **Docker** 2013
- **Microservices Defined Thoughtworks Fowler, Lewis** 2014

- **Microservices** 2014

- **DevOps** 2009
- **Ribbon** 2012
- **Microservices marked Assess Thoughtworks Radar** 2012
- **Docker** 2013
- **Microservices Defined Thoughtworks Fowler, Lewis** 2014

- **Microservices** 2014
EIGHT FALLACIES OF DISTRIBUTED COMPUTING

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is zero
8. The network is homogeneous

Photo: Icon made by Freepik from www.flaticon.com
Java Microservices Platform circa 2014

- Netflix Ribbon
- Eureka OSS
- Config Server
- Zipkin
- Netflix OSS
- Zuul
- Hystrix
Why these components?

- **Eureka** is the Service Registry where the clients lookup for service locations a.k.a Service Discovery
- **Config Server** externalized the Configuration
- **Ribbon** is the client side Load Balancer
- **Hystrix** is the Circuit Breaker
- **Zipkin** is the Distributed Tracer
- **Zuul** is the smart proxy purely based on Java
AN OPERATION NIGHTMARE!

Infra capabilities are tightly coupled with applications and services

- Incompatible across languages and frameworks
- Existing apps require refactoring
- Upgrades needs tight coordinations libraries
Infrastructure cluttering your code?

<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-eureka</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-zuul</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-hystrix</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-sleuth</artifactId>
</dependency>
SHORT HISTORY OF MICROSERVICES

- **Java EE6** 2009
- **AWS EC2** 2006
- **DropWizard** 2011
- **Hystricx** 2012
- **Eureka** July 2012
- **Spring Boot** 2013
- **Kubernetes** 2014
- **Continuous Integration via XP** 1999
- **Agile Manifesto** 2001
- **DevOps** 2009
- **NETFLIX to AWS** 2010
- **Ribbon** 2012
- **Microservices marked Assess Thoughtworks Radar** 2012
- **Docker** 2013
- **Microservices Defined Thoughtworks Fowler, Lewis** 2014
- **OpenShift 3** 2015

#RedHatOSD
MICROSERVICES'ILITIES + KUBERNETES

MyService

API
Tracing
Discovery
Invocation
Elasticity
Resilience
Pipeline
Authentication
Logging
Monitoring
MICROSERVICES’ILITIES + OPENSSHIFT
Better Microservices Platform circa 2016
Istio - Sail
(Kubernetes - Helmsman or ship’s pilot)
EVOLUTION OF SERVICES

- Service
  - Tracing
  - Circuit Breaker
  - Routing
  - Svc Discovery
  - Config

- Platform
  - Container Platform (+ Service Mesh)

- ...2014
- 2018
DISTRIBUTED SERVICES WITH ISTIO SERVICE MESH

ISTIO SERVICE MESH
- Load Balancing
- Fault Tolerance
- Traceability
- Observability
- Service Security
- Infra Security
- Chaos Engineering
- Traffic Control

OPENSİFT
THE ENTERPRISE KUBERNETES
- Build Automation
- Logs
- Monitoring
- Infra Security
- CI/CD
- Load Balancing
- Deployment Resiliency
- Service Discovery
- Config
- Resource Management
- Elasticity

INFRA
- Physical
- Virtual
- Cloud

SERVICE

SERVICE OPS

INFRA OPS

#RedHatOSD
Sidecars

#RedHatOSD
### SIDECAR PATTERN

- A utility container in the same pod to enhance the main container’s functionality
- Share the same network and lifecycle
- Istio uses an Istio Proxy (L7 Proxy) sidecar to proxy all network traffic between apps

Source: [http://blog.kubernetes.io/2015/06/the-distributed-system-toolkit-patterns.html](http://blog.kubernetes.io/2015/06/the-distributed-system-toolkit-patterns.html)
MICROSERVICES'ILITIES + ISTIO

MyService

API
Discovery
Invocation
Elasticity
Resilience
Pipeline
Authentication
Logging
Monitoring
Tracing

#RedHatOSD
Polyglot Microservices Platform circa 2018
### OpenShift Container Platform

**Overview**

- **customer**
  - **customer-v1**, #1
- **preference**
  - **preference-v1**, #1
- **recommendation**
  - **recommendation-v1**, #1
  - **recommendation-v2**, #1

**Applications**

- **customer**

**Builds**

- [Build](#)

**Resources**

- [Resources](#)

**Storage**

- [Storage](#)

**Monitoring**

- [Monitoring](#)

**Networking**

- [Networking](#)

**Logging**

- [Logging](#)
OPENSHIFT SERVICE MESH

- Supported distribution of
  - Istio
  - Jaeger
  - Kiali
  - Prometheus
  - Grafana

- Upstream project called Maistra
  [https://github.com/Maistra](https://github.com/Maistra)

- Integrated with Red Hat OpenShift Application Runtimes (RHOAR)
Istio Service Mesh

Data Plane
- HTTP1.1, HTTP2, gRPC, TCP w/TLS
- istioctl, API, config
- Quota, Telemetry, Rate Limiting, ACL
- CA, SPIFFE

Control Plane
- Pilot
- Mixer
- Citadel

POD
- SERVICE A
  - Istio Proxy
- SERVICE B
  - Istio Proxy
- SERVICE C
  - Istio Proxy
ISTIO MONITORING

POD

SERVICE A

Istio Proxy

POD

SERVICE A

Istio Proxy

POD

SERVICE A

Istio Proxy

Grafana

Prometheus Adapter

Backend

Adapter API

Mixer

Pilot

Citadel

Control Plane

Check & Report

Logging
Telemetry
Auth
Quota

#RedHatOSD
TECH PREVIEW 1
- Available for OCP 3.10
- Renders the whole cluster as tech preview
- Installation with Service Mesh Operator

LIMITATIONS
- Works on a single cluster where all deployments are on OpenShift
- Works with OpenShift SDN with no external providers

HOW TO GET ACCESS
- Install Istio Operator
- Create an Istio Installation object
- Scheduled to be available on CDK at the end of the year
Homeworks

Self-paced Labs
learn.openshift.com/servicemesh

GitHub
bit.ly/istio-tutorial
GRAZIE PER L’ATTENZIONE

Giuseppe Bonocore - Solution Architect
Ugo Landini - Solution Architect

#RedHatOSD