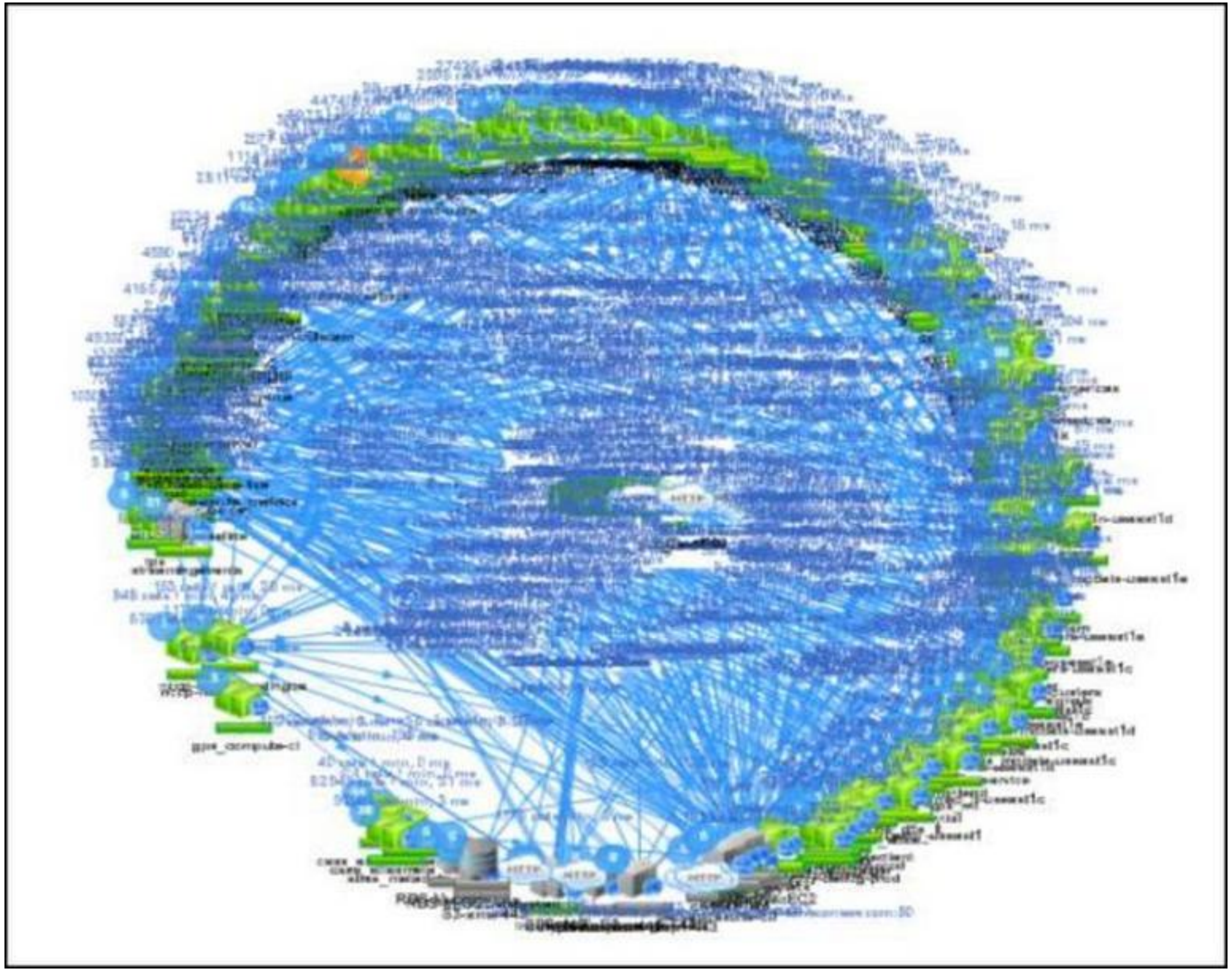
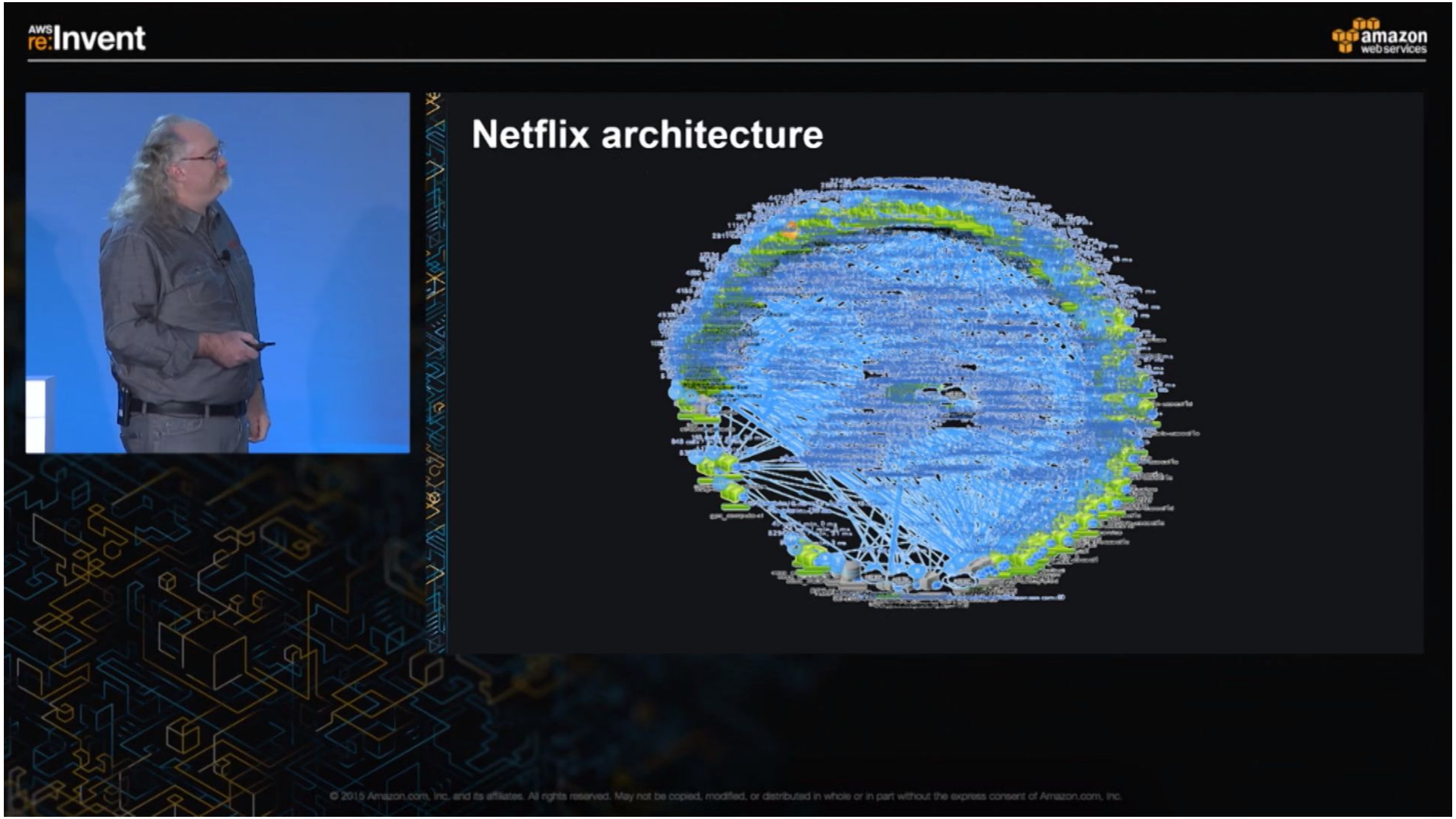
Docker and Kubernetes

docker

kubernetes



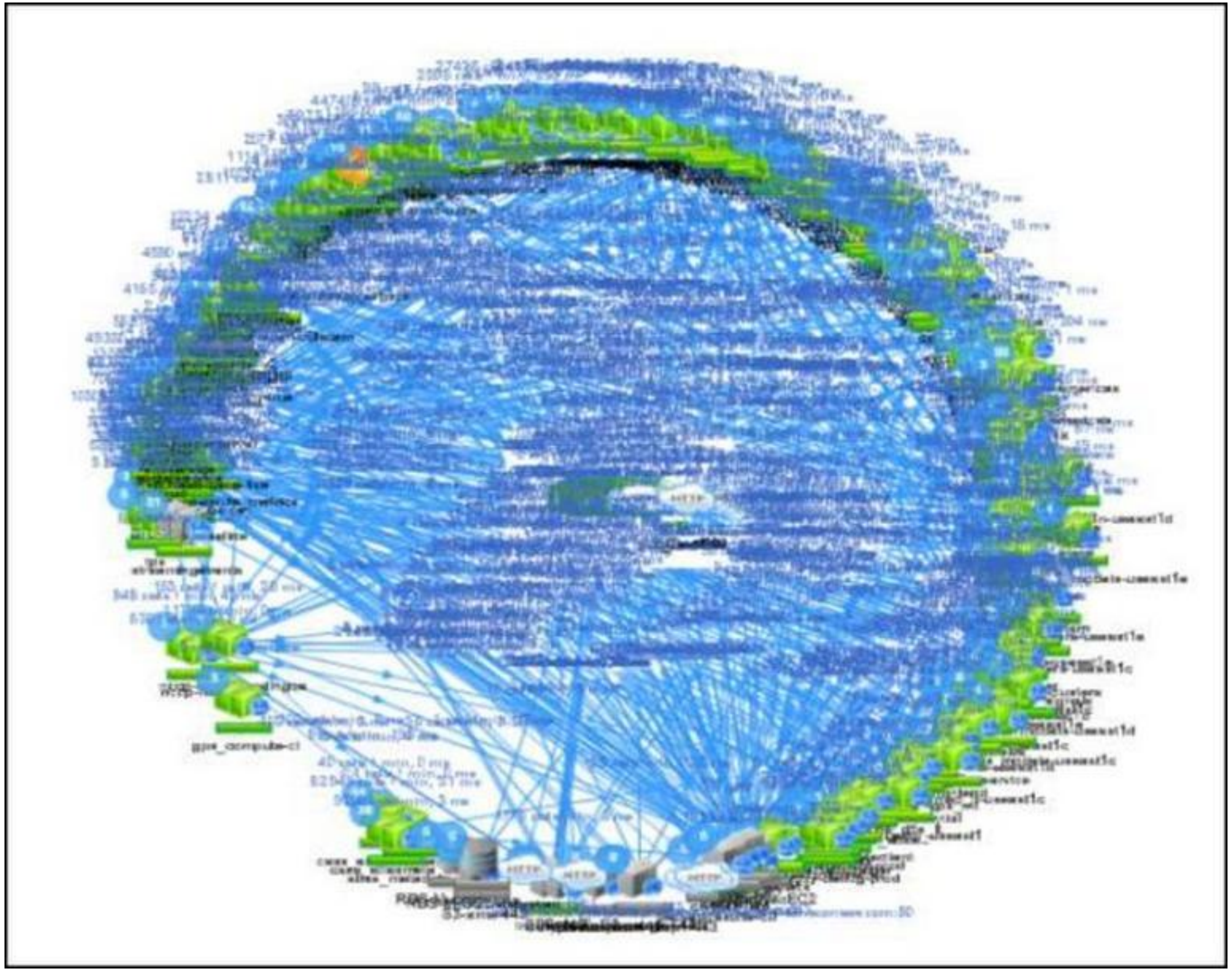


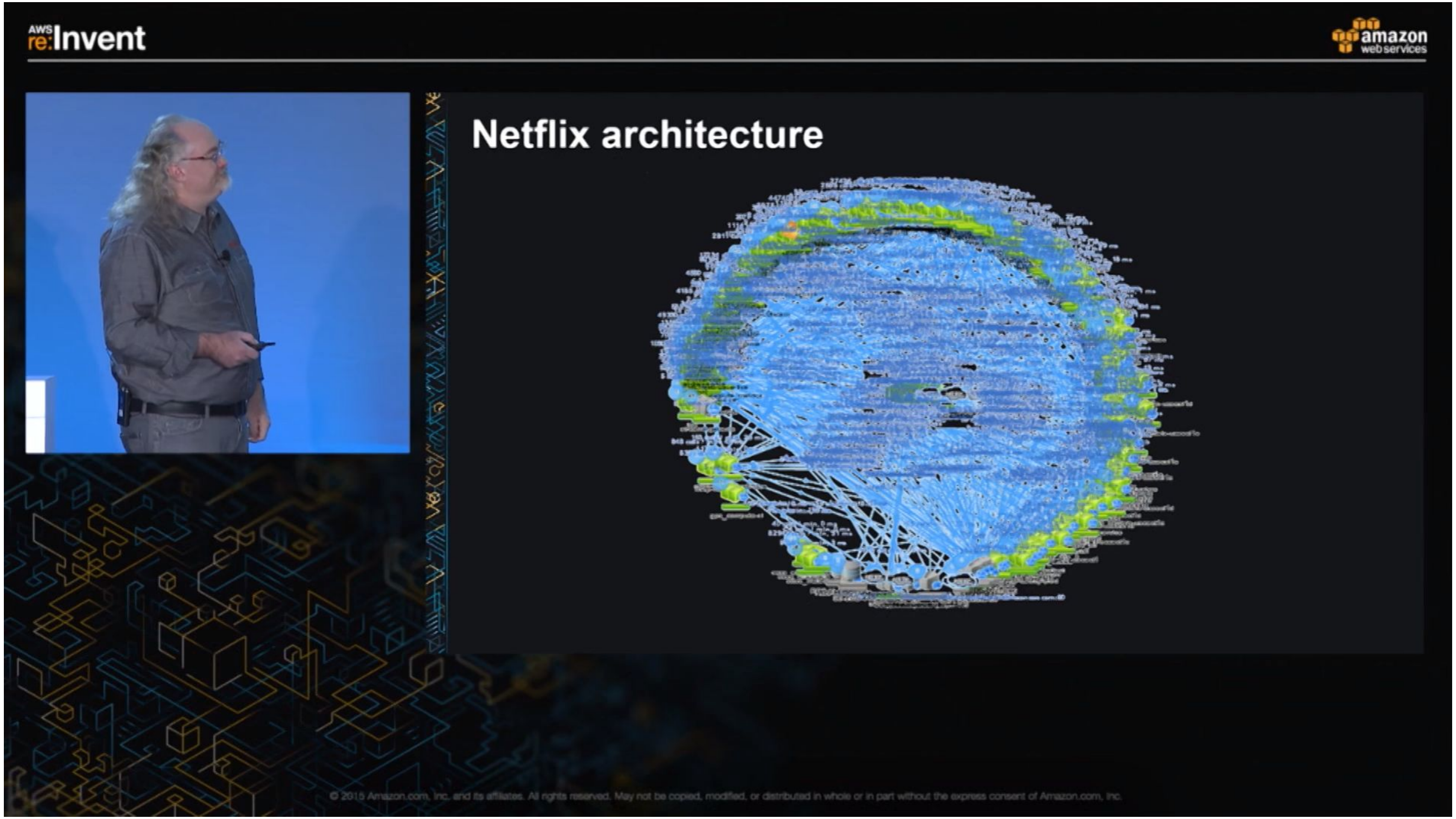
re:Invent

Netflix architecture

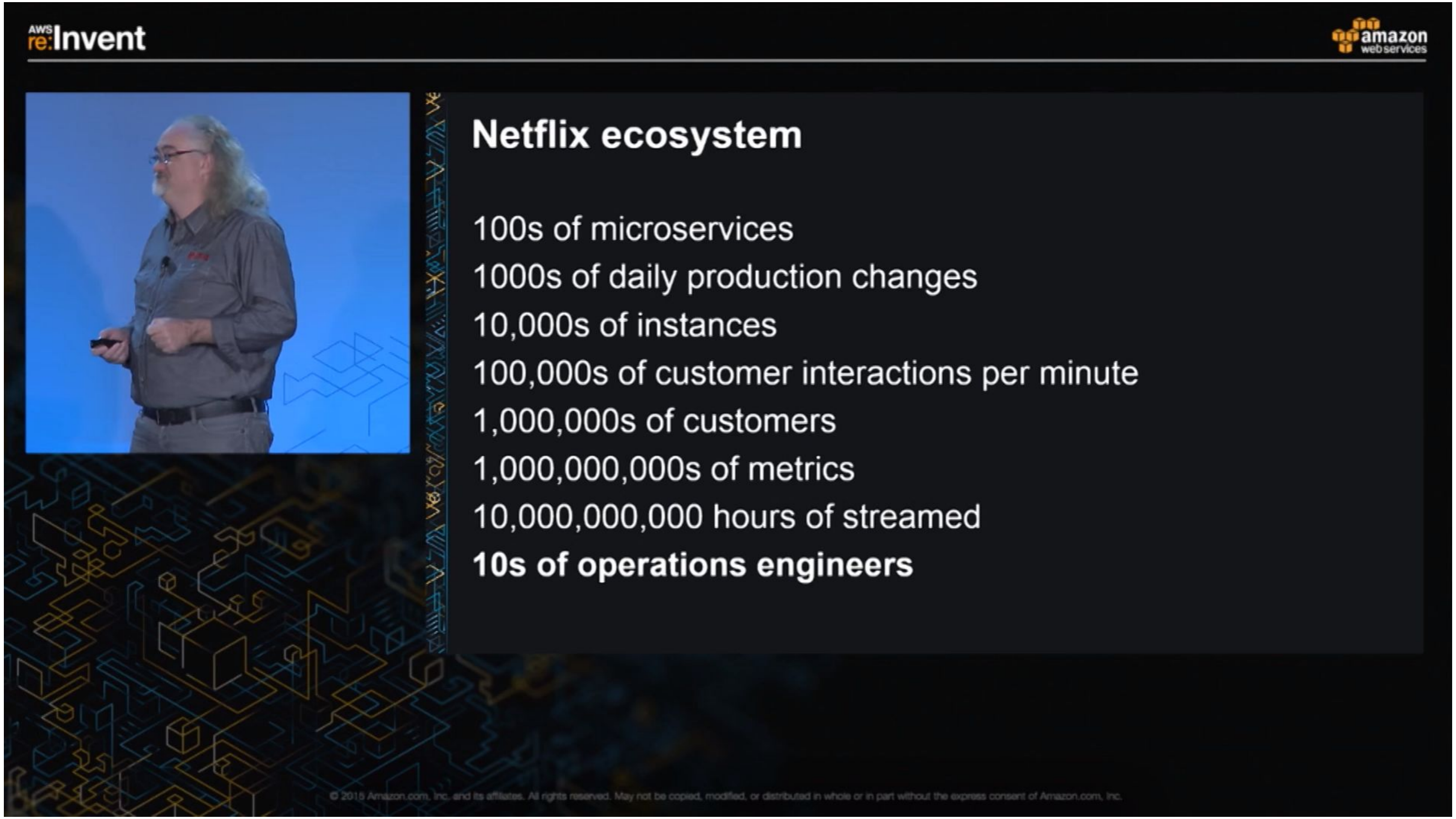
# whoami

- Martin Štefanko
- Software engineer, Red Hat
- MicroProfile committer
- Microservices enthusiast
- @xstefank





Netflix architecture



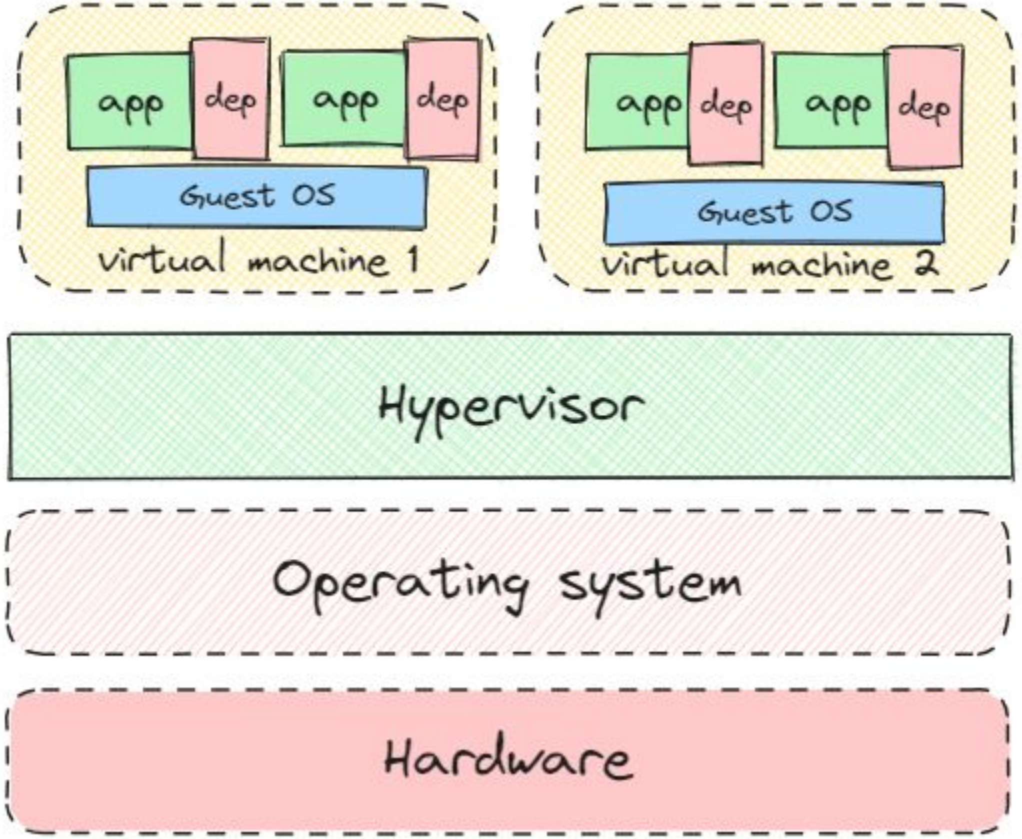
Netflix ecosystem

100s of microservices
1000s of daily production changes
10,000s of instances
100,000s of customer interactions per minute
1,000,000s of customers
1,000,000,000s of metrics
10,000,000,000 hours of streamed
10s of operations engineers

Technologies

Past deployment model

- **Virtual machines**
  - One or more virtual “guest” OS on a physical “host” machine
  - Owns its resources independently
  - Functions separately
  - Essentially what was understood as cloud


app
dep
app
dep
Guest OS
virtual machine 1
app
dep
app
dep
Guest OS
virtual machine 2
Hypervisor
Operating system
Hardware

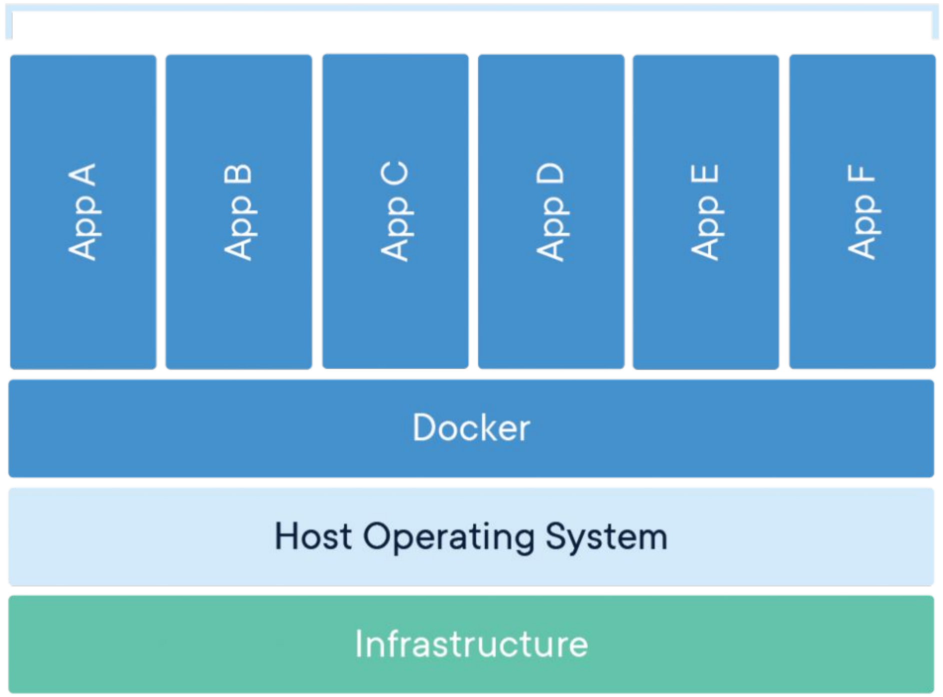
Docker (containers)

- **Container** - a standardized unit of software
- Packages code and its dependencies, runtime, system tools, system libraries, settings
- Users build **Docker images** - lightweight, standalone, executable package of software
- Images can be run anywhere where Docker is installed
- **Docker hub** (hub.docker.com)

Docker containers

- Container is runtime representation of the image
- Containers run on Docker Engine
- It doesn’t matter on which platform (Linux, Mac, Windows) you run
- Containers isolate software from its environment
- Uniform behavior everywhere

Containerized Applications

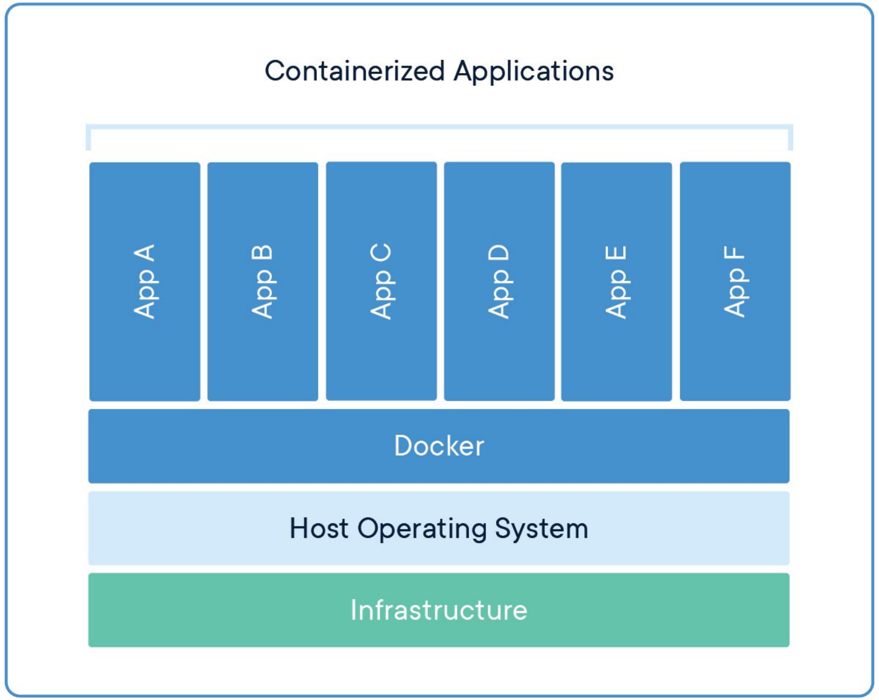


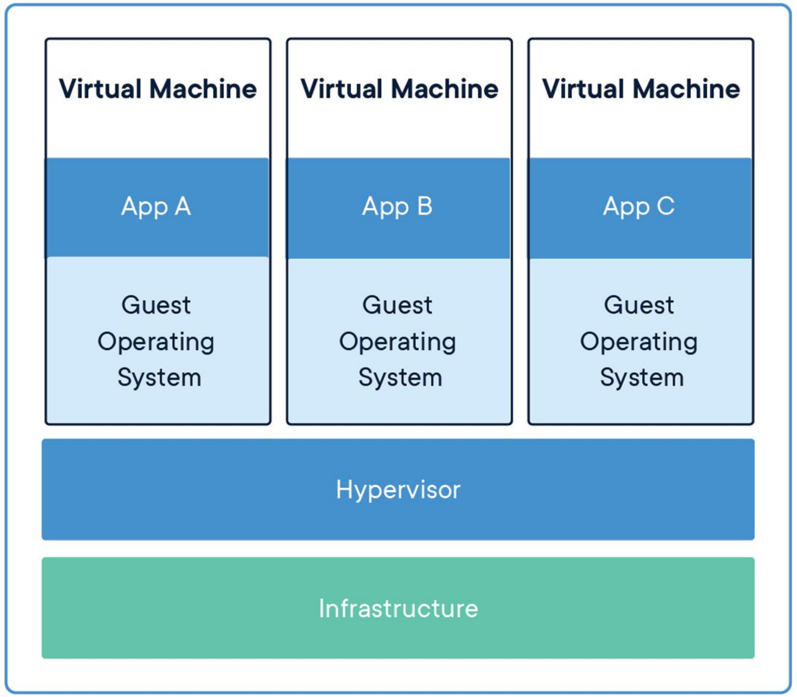
Docker Engine

- **Standard** - Docker created the industry standard for containers, so they could be portable anywhere
- **Lightweight** - Containers share the machine's OS system kernel and therefore do not require an OS per application
- **Secure** - Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry

Containers vs Virtual Machines

Containers and virtual machines have similar resource isolation and allocation benefits, but function differently because **containers virtualize the operating system instead of hardware**. Containers are more portable and efficient.


Containerized Applications
App A
App B
App C
App D
App E
App F
Docker
Host Operating System
Infrastructure


Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure

Docker – standardization

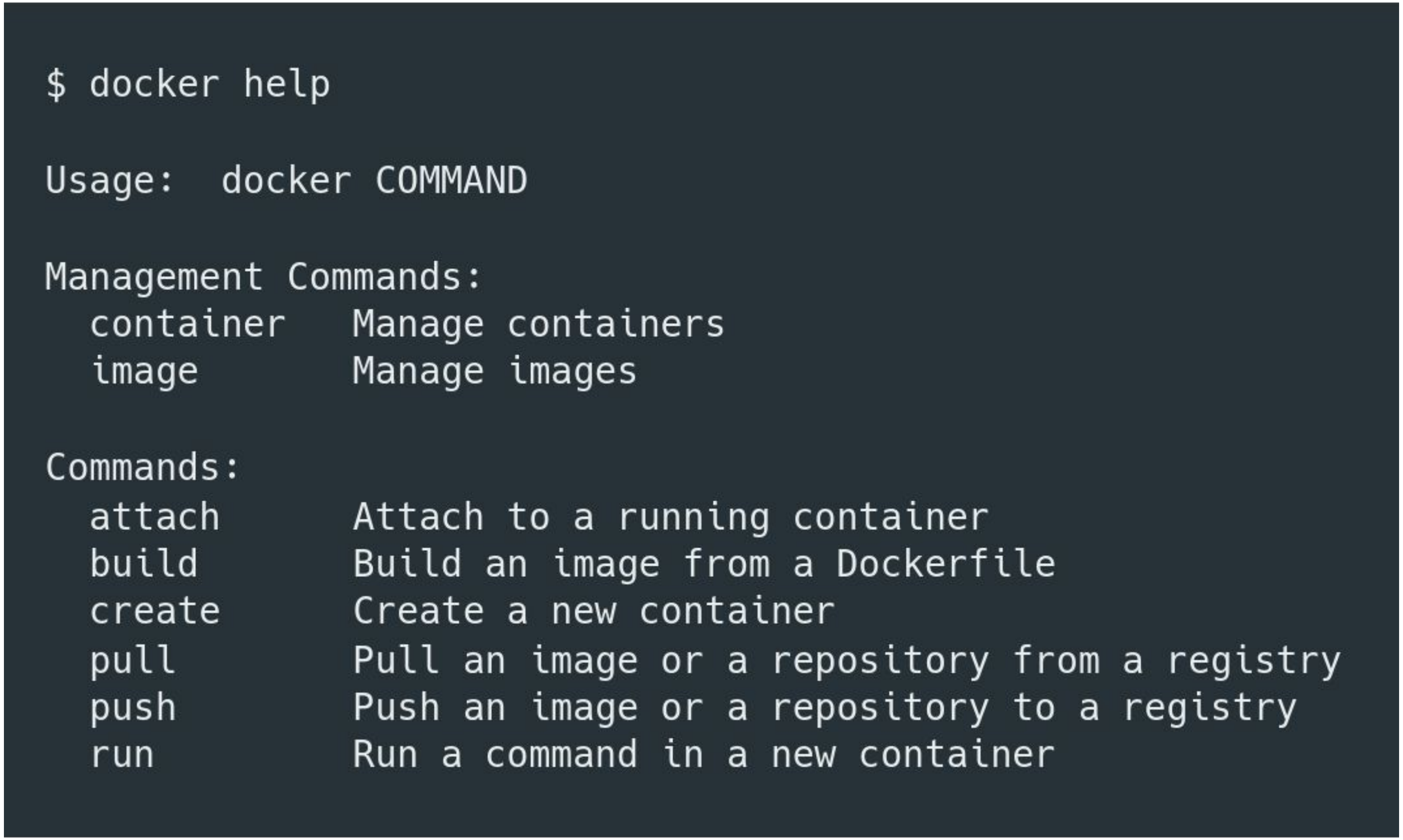
- Docker launched in 2013 – revolution in application development
- In June 2015, Docker donated the container image specification and runtime code now known as runc, to the **Open Container Initiative (OCI)**
- Other alternatives - Podman, Buildah

Dockerfile

```
FROM registry.access.redhat.com/ubi8/ubi-minimal
WORKDIR /work/
COPY target/*-runner /work/application
RUN chmod 775 /work
EXPOSE 8080
CMD ["./application", "-Dquarkus.http.host=0.0.0.0"]
```

Dockerfile

- FROM base-image
- LABEL key=value
- RUN command
- COPY src dest
- ADD src-file dest
- WORKDIR dir
- ENV ENV_VAR=value
- VOLUME /data
- ENTRYPOINT cmd
- CMD args (possible to override)



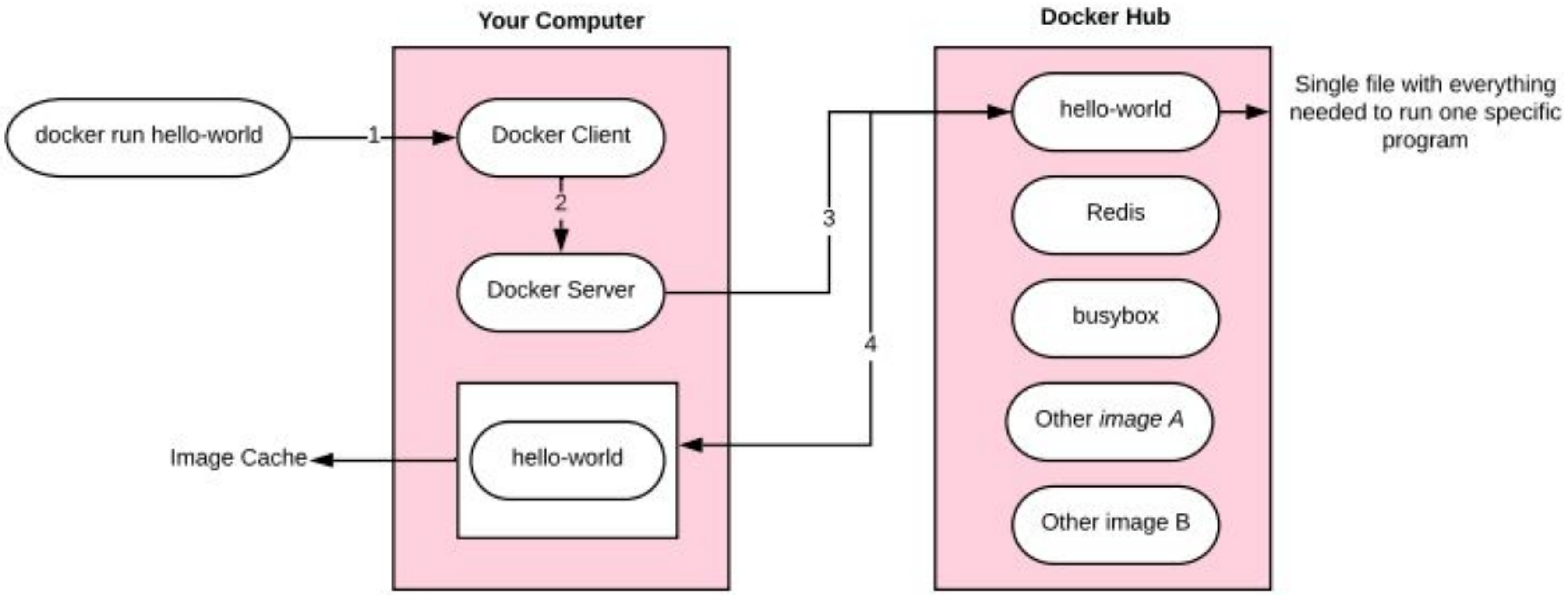
```
$ docker help

Usage:  docker COMMAND

Management Commands:
  container   Manage containers
  image       Manage images

Commands:
  attach      Attach to a running container
  build       Build an image from a Dockerfile
  create      Create a new container
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  run         Run a command in a new container
```

Docker installation

- https://docs.docker.com/get-docker/
- https://podman.io/docs/installation

- Docker Desktop - https://www.docker.com/products/docker-desktop
- Podman Desktop - https://podman-desktop.io/

- https://labs.play-with-docker.com/


Your Computer
Docker Hub
docker run hello-world
1
Docker Client
2
Docker Server
3
4
hello-world
Image Cache
hello-world
Single file with everything needed to run one specific program
Redis
busybox
Other *image A*
Other image B

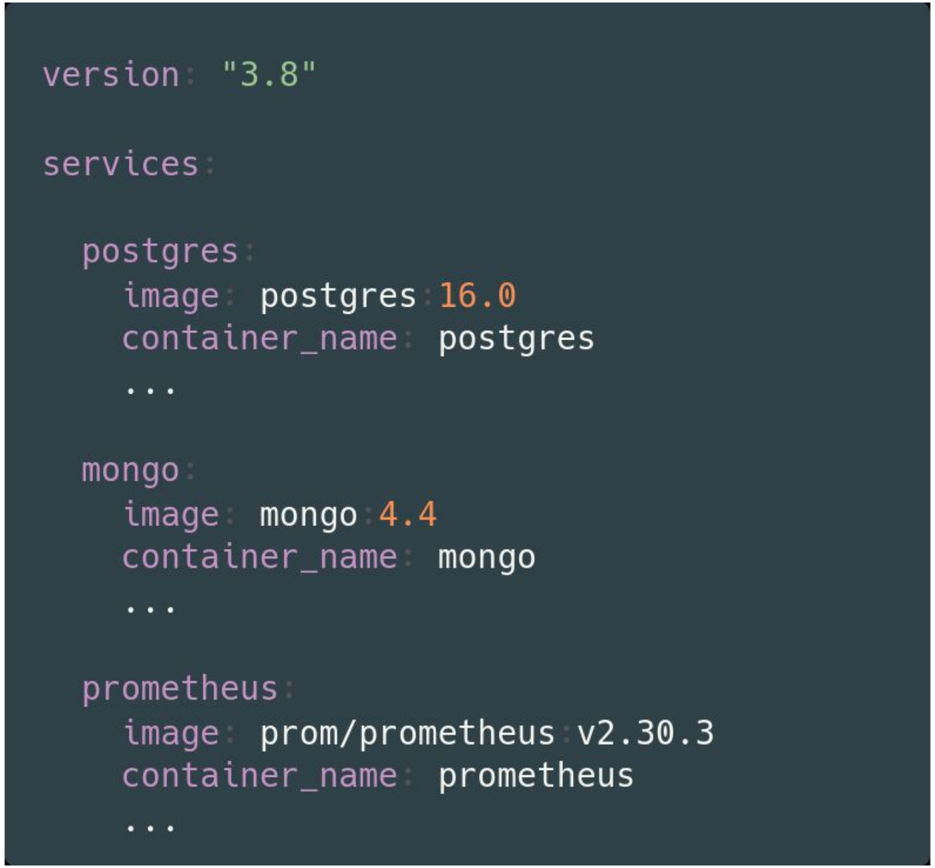
Docker compose

- Often we need several containers to work in unison
- Declarative definition of several containers
- https://docs.docker.com/compose/

Docker compose YAML

- Configuration file docker-compose.yml

  - Start all containers (logs combined)
    - docker-compose up
  - Start all containers in the background
    - docker-compose up -d
  - Stop all containers
    - docker-compose down



```yaml
version: "3.8"

services:

  postgres:
    image: postgres:16.0
    container_name: postgres
    ...

  mongo:
    image: mongo:4.4
    container_name: mongo
    ...

  prometheus:
    image: prom/prometheus:v2.30.3
    container_name: prometheus
    ...
```

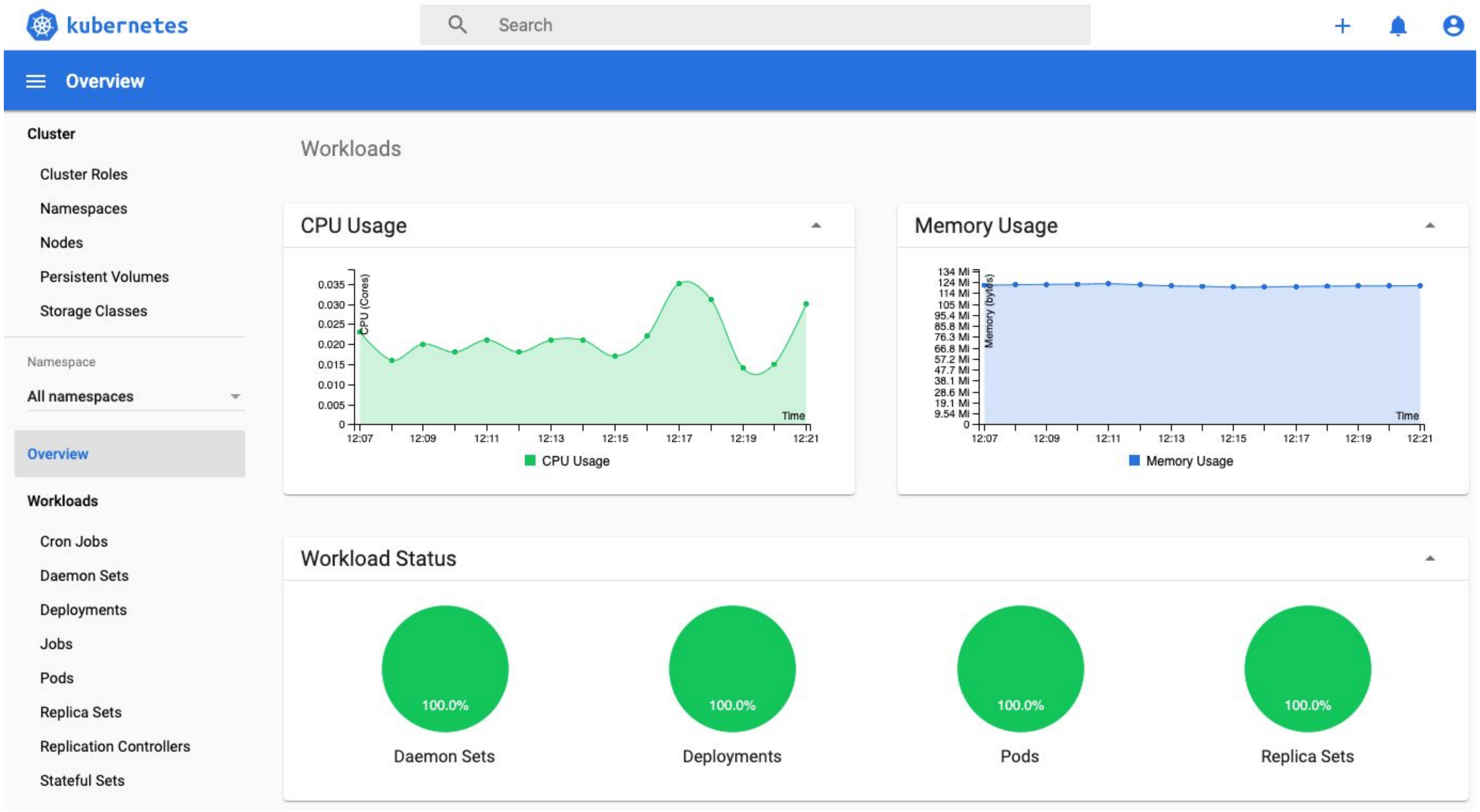
Docker compose installation

- https://docs.docker.com/compose/install/linux/
- https://github.com/containers/podman-compose#installation

```
$ podman-compose help
...
command:
    ...
    pull                pull stack images
    push                push stack images
    build               build stack images
    up                  Create and start the entire stack or some of its services
    down                tear down entire stack
    ps                  show status of containers
    run                 create a container similar to a service to run a one-off command
    exec                execute a command in a running container
    start               start specific services
    stop                stop specific services
    restart             restart specific services
    logs                show logs from services
    config              displays the compose file
    port                Prints the public port for a port binding.
    pause               Pause all running containers
    unpause             Unpause all running containers
    kill                Kill one or more running containers with a specific signal
```

Kubernetes

- Container orchestration
- an open-source system for automating deployment, scaling, and management of containerized applications
- Groups containers to logical units
- Easy administration
- De facto standard for cloud deployments



@xstefank @RedHat

Kubernetes objects

- **Pod** - basic executions unit
  - Process running in the cluster
  - One or multiple containers
  - Replaceable unit, can be restarted anytime (health checks)
- **Service** - exposure of application (pods) as a network service
  - Abstraction of the access to pods

Kubernetes objects

- **Volume** - storage shared between containers in the pod
- **Deployment** - declarative updates for pods
  - User describes the desired state
  - Deployment controller (dc) changes the actual state to the desired state at controlled rate
  - New state of the pods, rollbacks, scaling,...

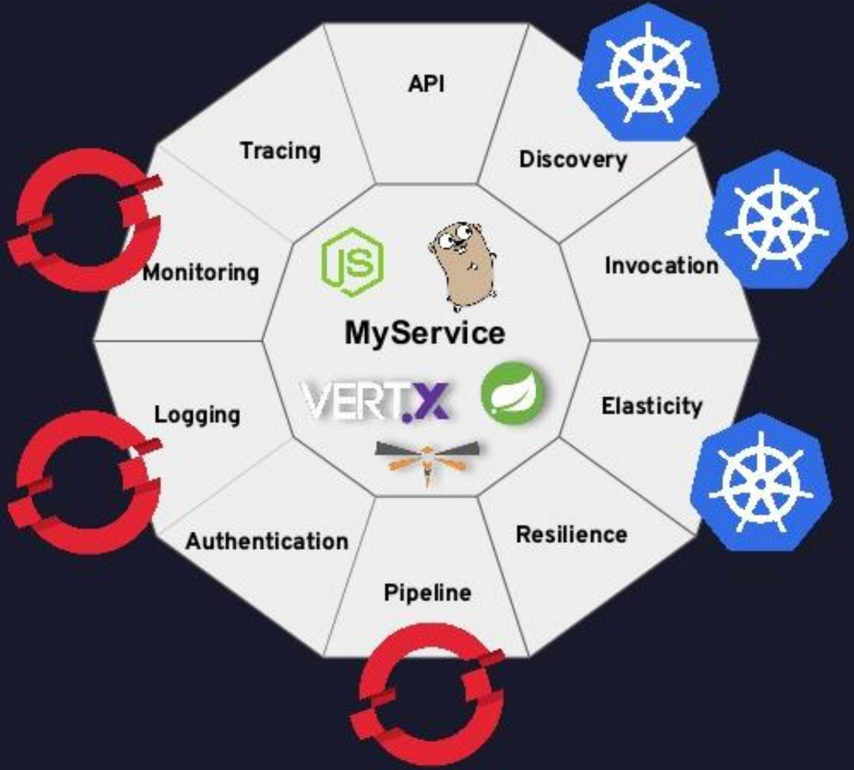
```
$ kubectl help

Basic Commands (Beginner):
  create         Create a resource from a file or from stdin.
  expose         Take a replication controller, service, deployment or pod and expose it as a new

Kubernetes Service
  run            Run a particular image on the cluster
  set            Set specific features on objects

Deploy Commands:
  rollout        Manage the rollout of a resource
  scale          Set a new size for a Deployment, ReplicaSet, Replication Controller, or Job
  autoscale      Auto-scale a Deployment, ReplicaSet, or ReplicationController
```

OpenShift

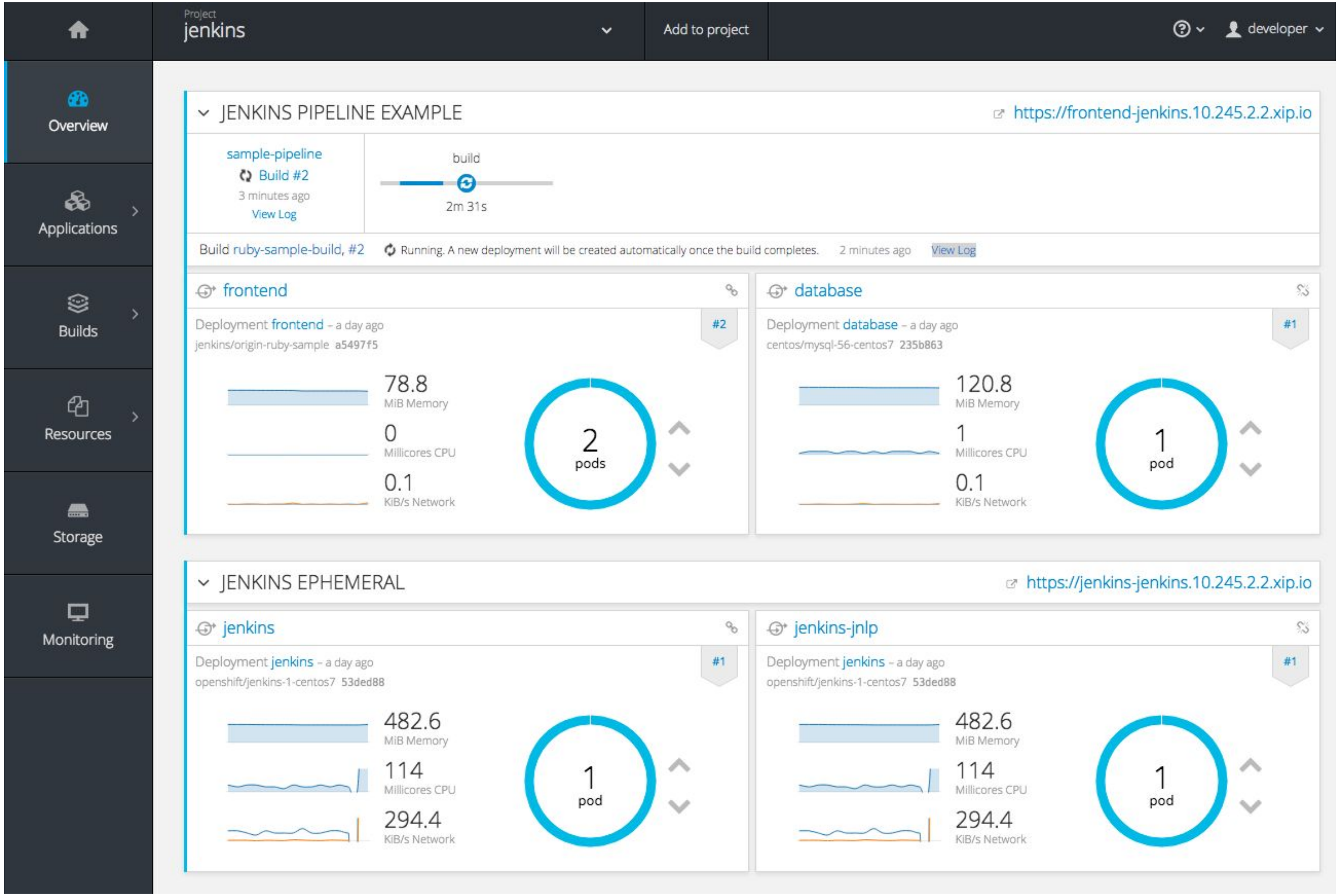
OPENSHIFT

- Fork of Kubernetes developed and maintained at Red Hat
- Commercial product with support
- Automated installation, upgrades, and lifecycle management throughout the container stack

Microservices'ilities + OpenShift



RED HAT DEVELOPERS



JENKINS PIPELINE EXAMPLE https://frontend-jenkins.10.245.2.2.xip.io

sample-pipeline
Build #2
3 minutes ago
View Log

build
2m 31s

Build ruby-sample-build, #2 Running. A new deployment will be created automatically once the build completes. 2 minutes ago View Log

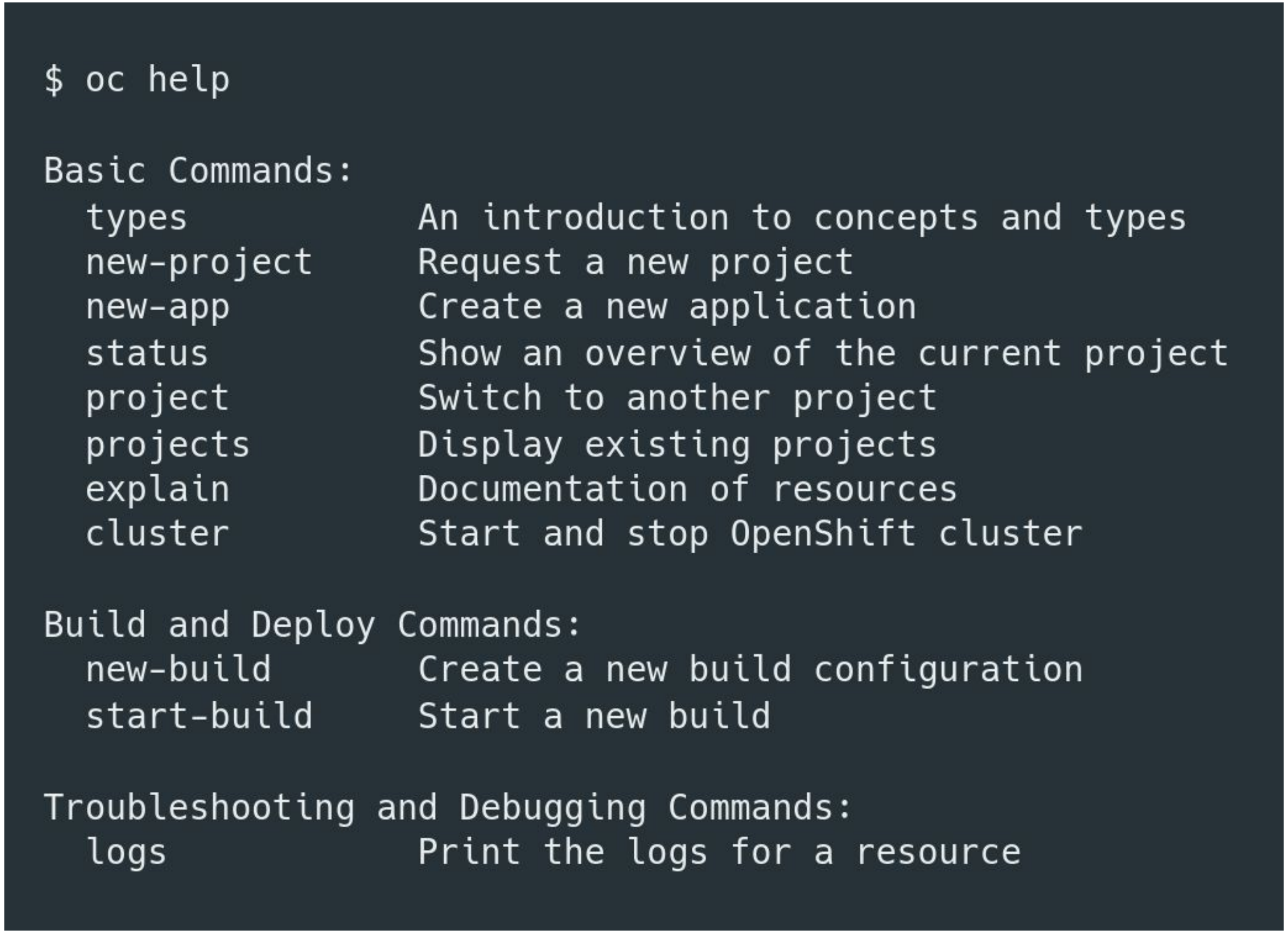
frontend
Deployment frontend – a day ago #2
jenkins/origin-ruby-sample a5497f5
78.8 MiB Memory
0 Millicores CPU
0.1 KiB/s Network
2 pods

database
Deployment database – a day ago #1
centos/mysql-56-centos7 235b863
120.8 MiB Memory
1 Millicores CPU
0.1 KiB/s Network
1 pod

JENKINS EPHEMERAL https://jenkins-jenkins.10.245.2.2.xip.io

jenkins
Deployment jenkins – a day ago #1
openshift/jenkins-1-centos7 53ded88
482.6 MiB Memory
114 Millicores CPU
294.4 KiB/s Network
1 pod

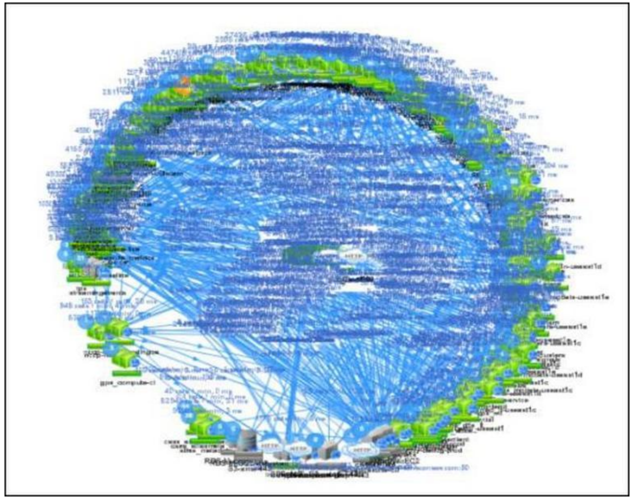
jenkins-jnlp
Deployment jenkins – a day ago #1
openshift/jenkins-1-centos7 53ded88
482.6 MiB Memory
114 Millicores CPU
294.4 KiB/s Network
1 pod



```
$ oc help

Basic Commands:
  types           An introduction to concepts and types
  new-project     Request a new project
  new-app         Create a new application
  status          Show an overview of the current project
  project         Switch to another project
  projects        Display existing projects
  explain         Documentation of resources
  cluster         Start and stop OpenShift cluster

Build and Deploy Commands:
  new-build       Create a new build configuration
  start-build     Start a new build

Troubleshooting and Debugging Commands:
  logs            Print the logs for a resource
```

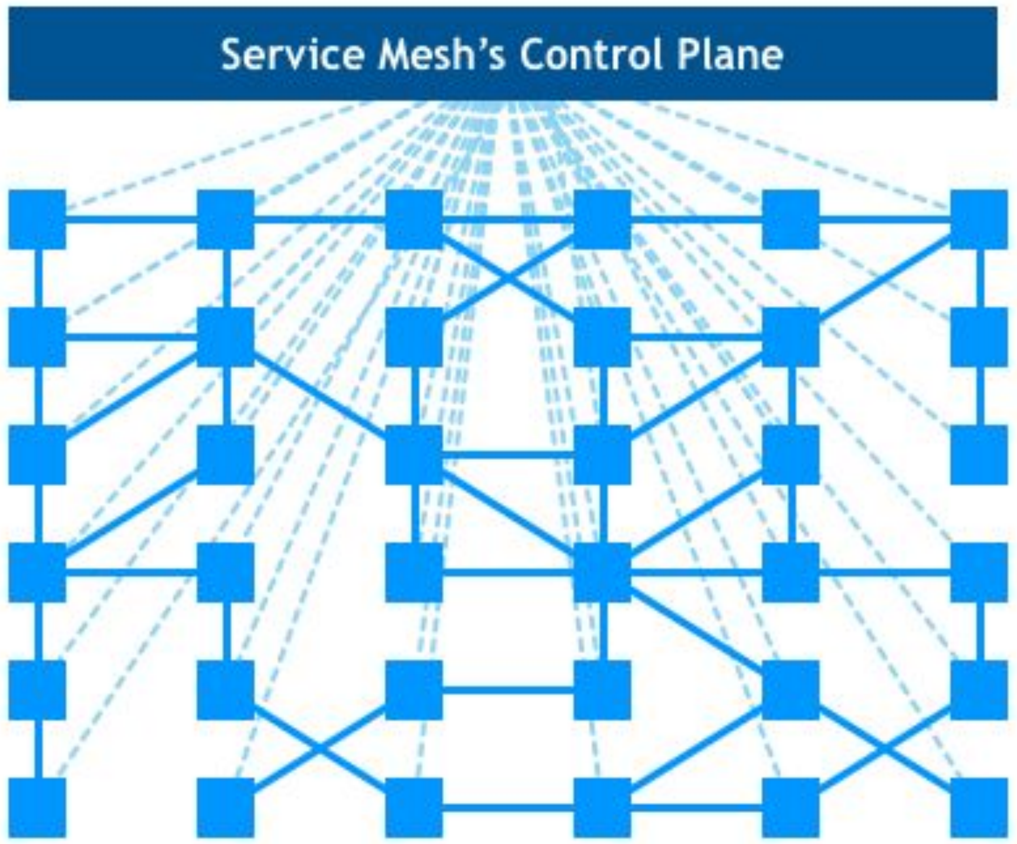
Istio – service mesh

- **Service mesh** – the network of microservices that make up the application and the interactions between them



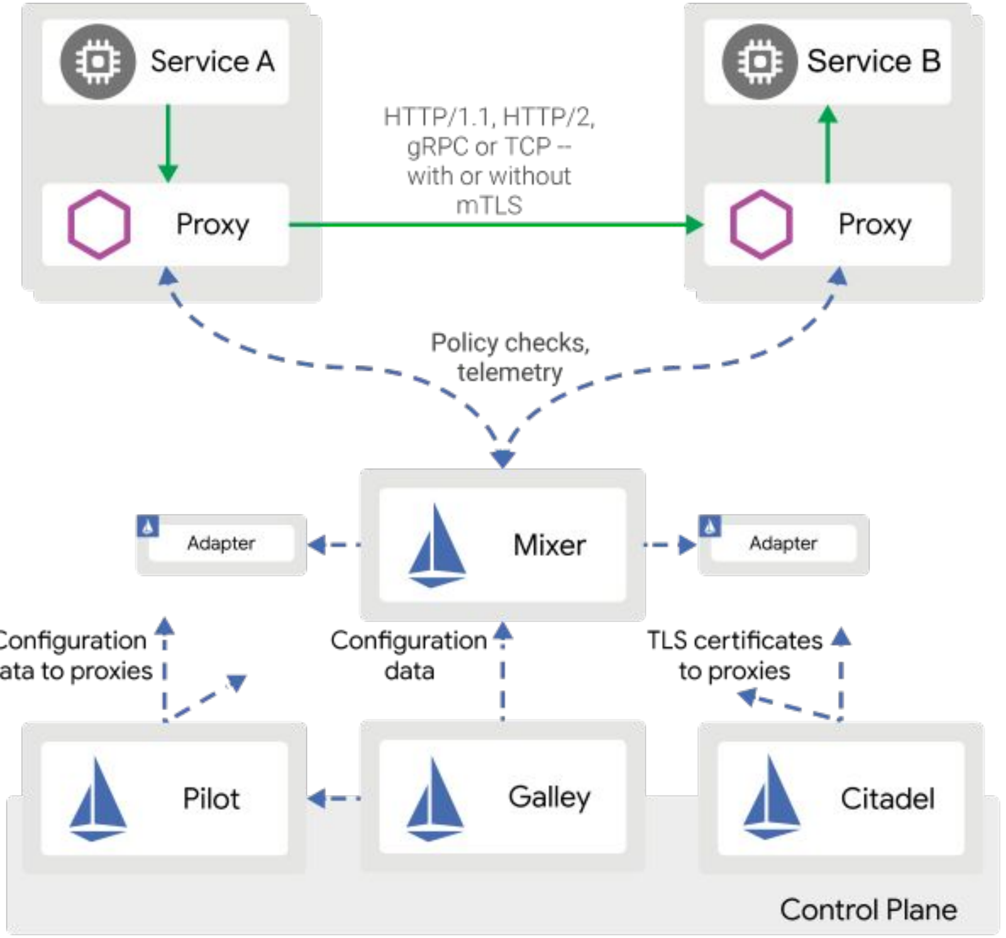
Istio – service mesh

- As a service mesh grows in **size and complexity**, it can become **harder to understand and manage**
- requirements include **discovery**, **load balancing**, **failure recovery**, **metrics**, **and monitoring**
- operational requirements, like **A/B testing**, **canary rollouts**, **rate limiting**, **access control**, **and end-to-end authentication**

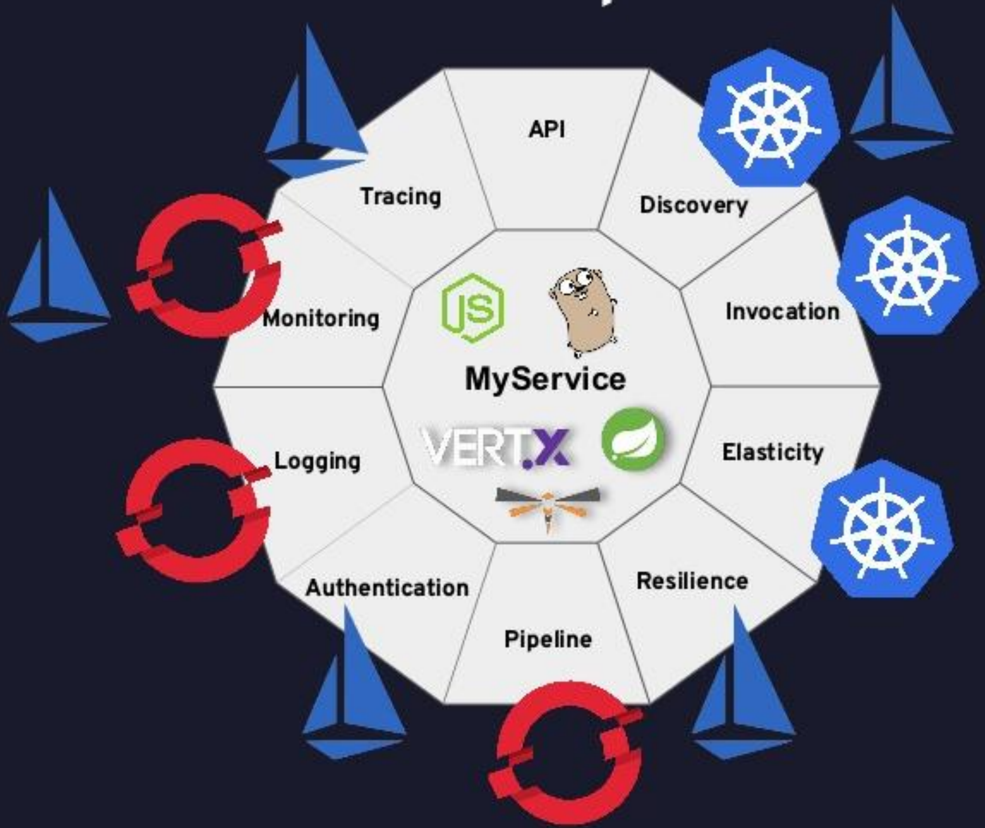

Service Mesh's Control Plane

Istio - Envoy

- Sidecar container
- Deployed in the same pod as the application container
- All network traffic goes through the Envoy proxy


Service A
Service B
Proxy
Proxy
HTTP/1.1, HTTP/2, gRPC or TCP -- with or without mTLS
Policy checks, telemetry
Adapter
Mixer
Adapter
Configuration data to proxies
Configuration data
TLS certificates to proxies
Pilot
Galley
Citadel
Control Plane

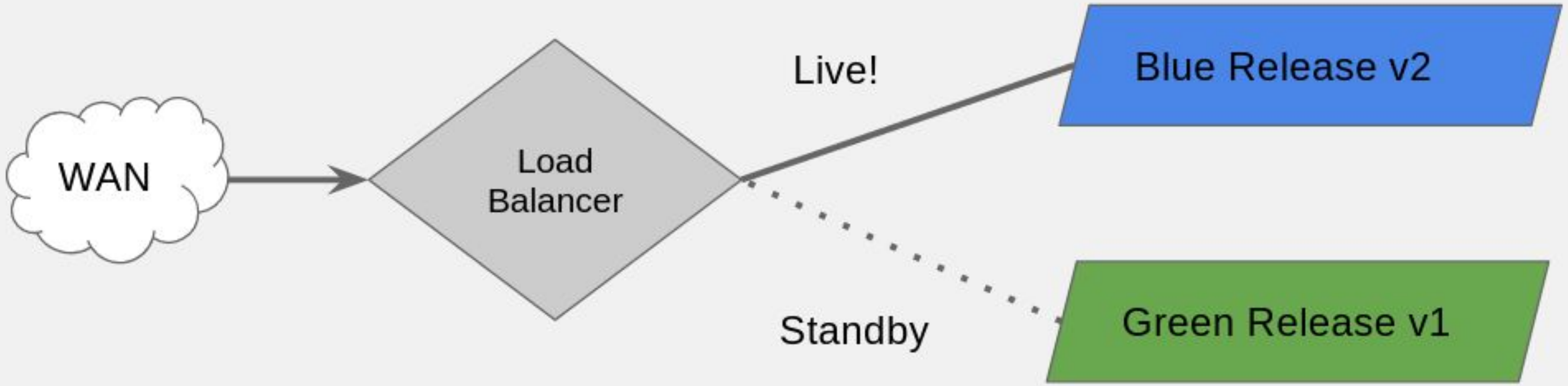
Microservices'ilities + OpenShift + Istio

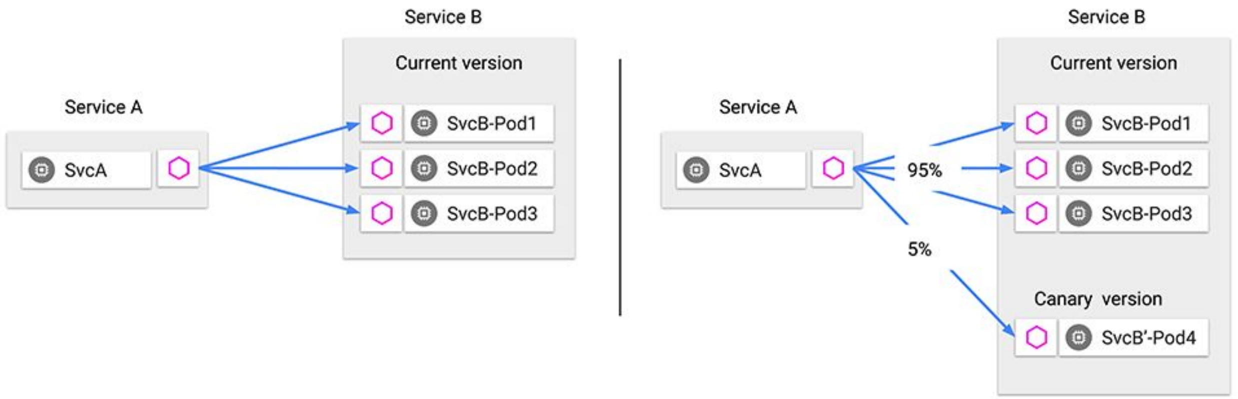


RED HAT DEVELOPERS

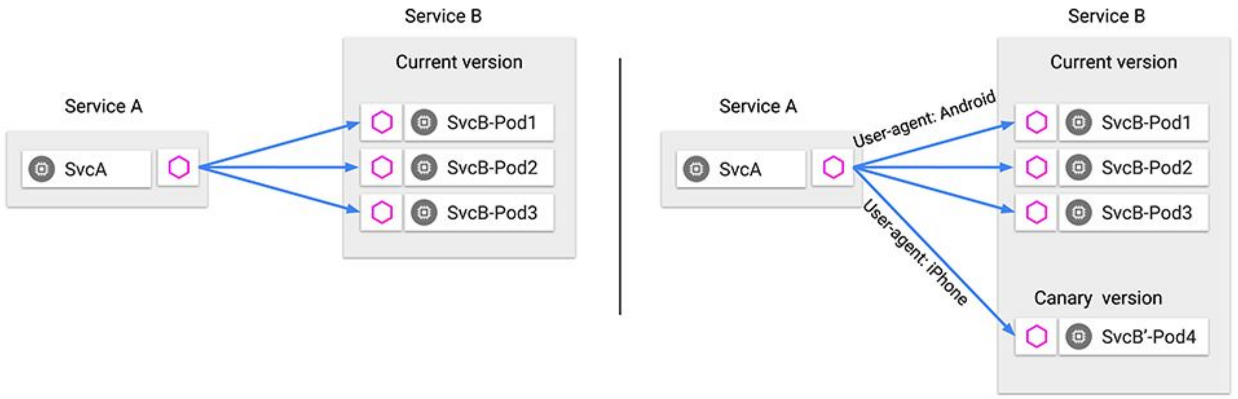
Blue / Green Deployment

Keep a hot standby ready in case a new release is flawed.

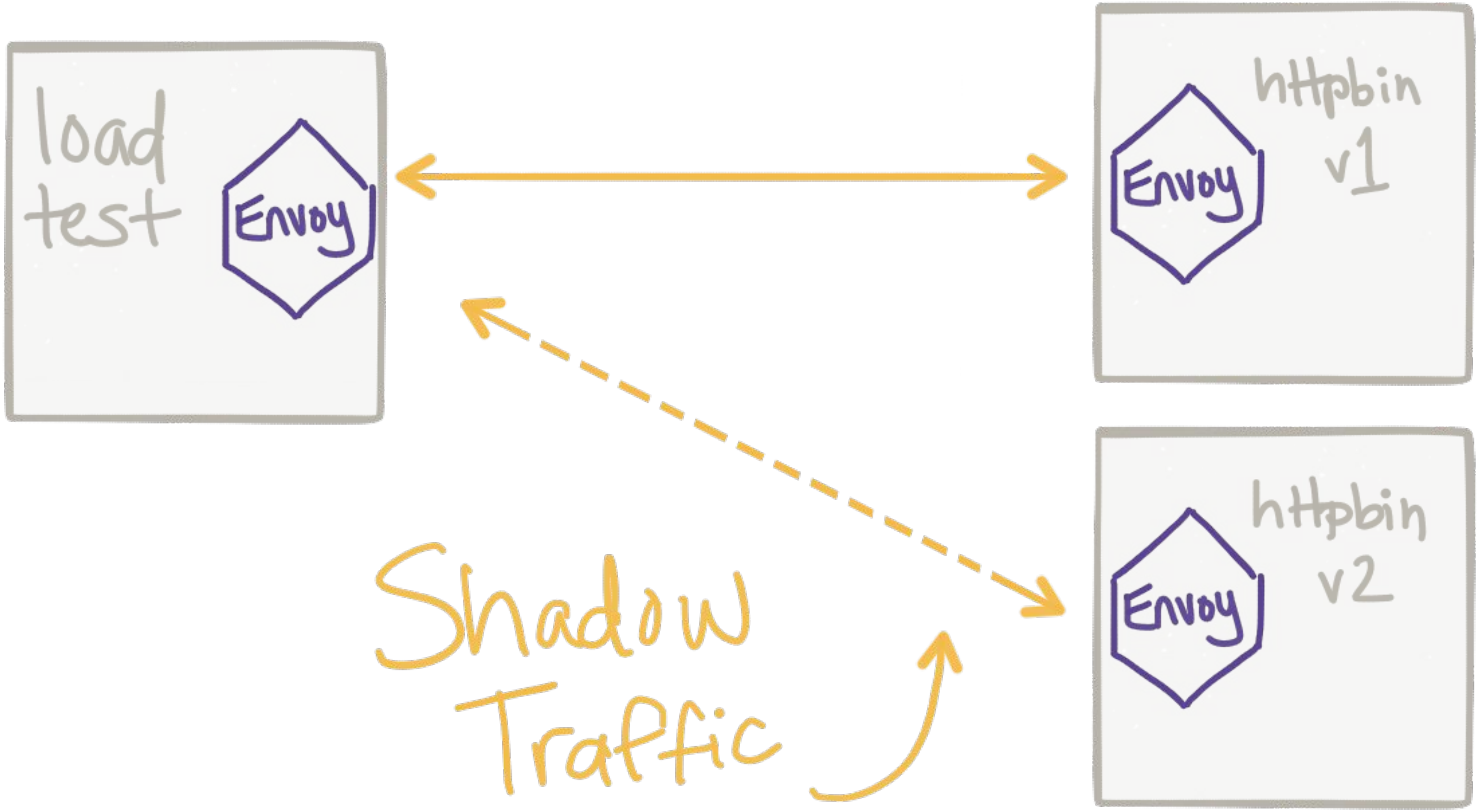




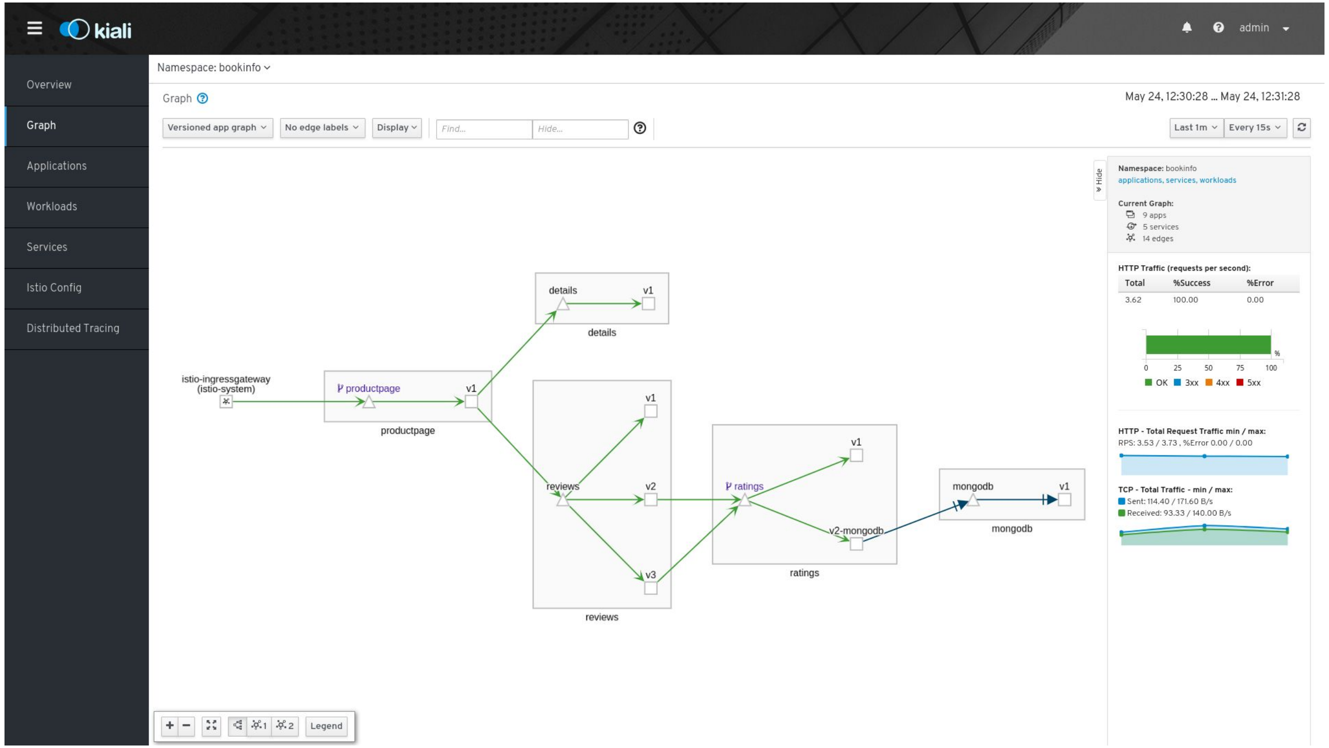
Traffic splitting decoupled from infrastructure scaling - proportion of traffic routed to a version is independent of number of instances supporting the version



Content-based traffic steering - The content of a request can be used to determine the destination of a request


load test
Envoy
httpbin v1
Envoy
httpbin v2
Envoy
Shadow Traffic

Kiali – service mesh observability



Namespace: bookinfo

Graph

May 24, 12:30:28 ... May 24, 12:31:28

Namespace: bookinfo
applications, services, workloads

Current Graph:
9 apps
5 services
14 edges

HTTP Traffic (requests per second):

Total	%Success	%Error
3.62	100.00	0.00

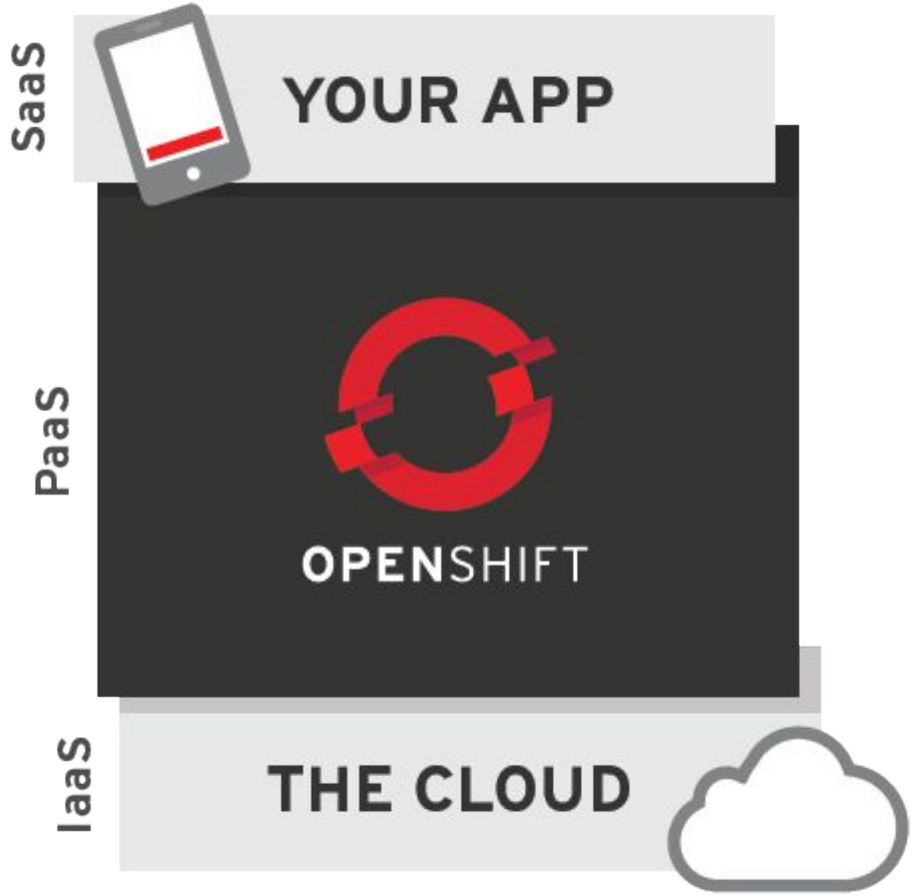
🟩 OK 🟦 3xx 🟧 4xx 🟥 5xx

HTTP - Total Request Traffic min / max:
RPS: 3.53 / 3.73 , %Error 0.00 / 0.00

TCP - Total Traffic - min / max:
🟦 Sent: 114.40 / 171.60 B/s
🟩 Received: 93.33 / 140.00 B/s

Cloud computing

- IaaS - Infrastructure as a service
  - VMs, servers, storage, network
- PaaS - Platform as a service
  - Execution runtime, database, application server managed Kubernetes, Openshift
- SaaS - Software as a Service
  - Provided applications, CRM, Email, communication


SaaS
YOUR APP
PaaS
OPENSHIFT
IaaS
THE CLOUD

Relational DBs

ORACLE DATABASE

Microsoft SQL Server

PostgreSQL

TERADATA

MariaDB

MySQL

IBM DB2

SYBASE

H2

Access

Apache Derby

HIVE

SQLite

HyperSQL

Distributed tracing

JAEGER

ZIPKIN

Fault tolerance

HYSTRIX DEFEND YOUR APP

Failsafe

Security

KEYCLOAK

NoSQL DBs

APACHE HBASE

Cassandra

riak

CouchDB relax

mongoDB

HYPERTABLE INC

Neo4j

redis

Metrics / monitoring

Prometheus

Grafana

Demo

Thank you

- @xstefank
- xstefank
- xstefank122@gmail.com

Resources

- https://www.zdnet.com/article/to-be-a-microservice-how-smaller-parts-of-bigger-applications-could-remake-it/ originally by Bruce Wong
- https://medium.com/refraction-tech-everything/how-netflix-works-the-hugely-simplified-complex-stuff-that-happens-every-time-you-hit-play-3a40c9be254b
- https://dzone.com/articles/microservices-vs-soa-whats-the-difference
- https://martinfowler.com/articles/microservices.html
- https://www.docker.com/resources/what-container
- https://github.com/kubernetes/kubernetes/blob/master/logo/logo.svg
- https://docs.aws.amazon.com/eks/latest/userguide/dashboard-tutorial.html
- https://www.slideshare.net/asotobu/service-mesh-patterns
- https://access.redhat.com/documentation/en-us/openshift_container_platform/3.3/html/release_notes/release-notes-ocp-3-3-release-notes
- https://thenewstack.io/history-service-mesh/
- https://philcalcado.com/2017/08/03/pattern_service_mesh.html
- https://istio.io/docs/concepts/what-is-istio/
- http://dougbtv.com/nfvpe/2017/06/05/istio-deploy/
- https://blog.aquasec.com/istio-service-mesh-traffic-control
- https://blog.christianposta.com/microservices/traffic-shadowing-with-istio-reduce-the-risk-of-code-release/
- https://github.com/kiali/kiali
- https://blog.openshift.com/what-is-platform-as-a-service-paas/
- https://serverless.zone/abstracting-the-back-end-with-faas-e5e80e837362
- https://softwareengineeringdaily.com/2016/09/08/relational-databases-with-craig-kerstiens/
- https://www.getfilecloud.com/blog/2014/08/leading-nosql-databases-to-consider/
- https://www.jaegertracing.io/
- https://blog.twitter.com/engineering/en_us/a/2012/distributed-systems-tracing-with-zipkin.html
- https://www.trzcacak.rs/imgm/iTJioJh_prometheus-logo-logo-prometheus/
- https://en.wikipedia.org/wiki/File:Grafana_logo.png
- https://design.jboss.org/keycloak/index.htm
- https://github.com/Netflix/Hystrix
- https://www.analyticsvidhya.com/blog/2022/06/writing-dockerfile-is-simple/