Microservices, Containers, Kubernetes and Istio

Vishal Charegaonkar

(cvishal@in.ibm.com)

Agenda

Session 1 (21st Feb 2018)

- MicroServices
- Containers
- Orchestration
- Service Mesh

Session 2 (23rd Feb 2018)

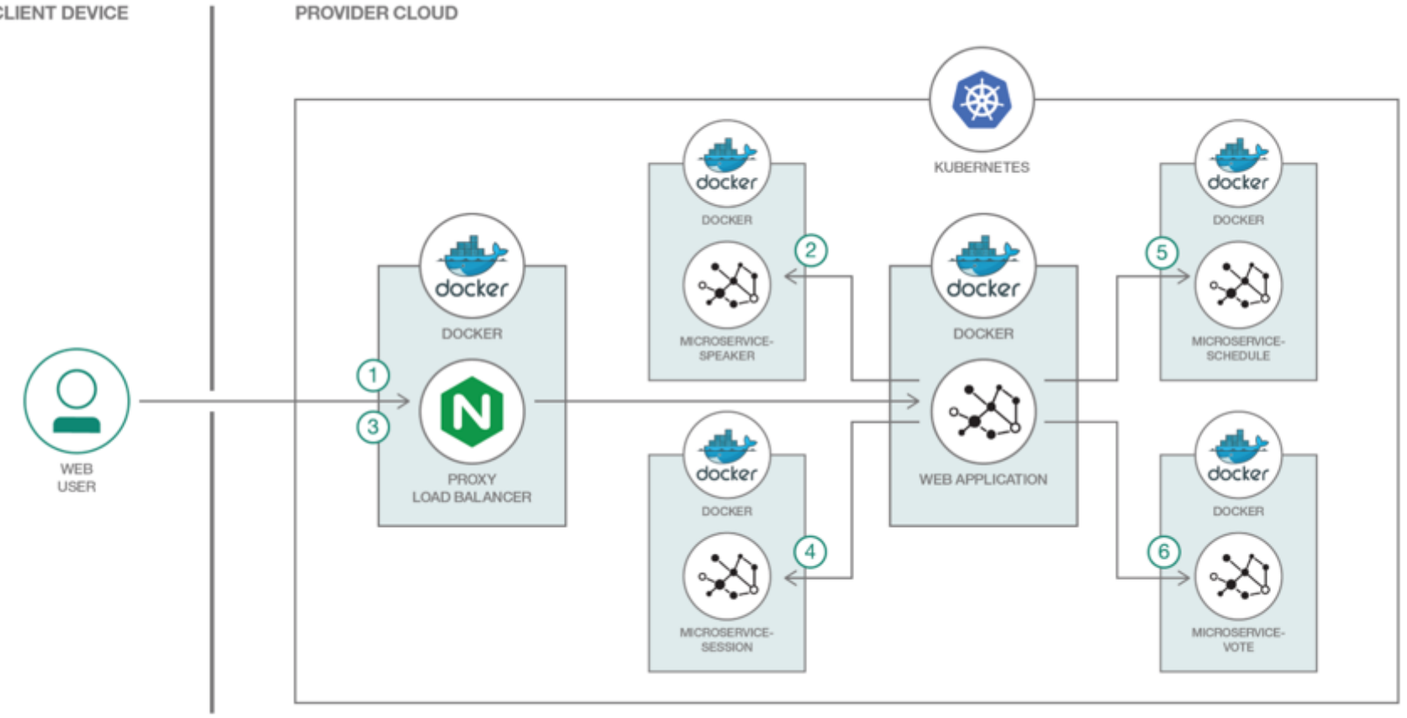
- Enterprise Cloud Platforms for hosting MicroServices Applications
- IBM Cloud Private Solution
- IBM Cloud Public
- Discuss on DevOps, Monitoring, logging concept for cloud.

What are Microservices?

An **engineering approach** focused on **decomposing** an application into **single-function** modules with **well defined interfaces** which are **independently** deployed and operated by **small teams** who own the **entire lifecycle** of the service.

Microservices accelerate delivery by **minimizing communication** and coordination between people while **reducing the scope** and risk of change.

Microservices Example



BUILT IN RESILIENCY IN THE APPLICATION

CENTRALIZED LOGGING AND MONITORING

FASTER UPDATES. APPLY INNOVATION USE OF DEVOPS

AUTO SCALING UP AND DOWN. SAVE COST

PUBLIC NETWORK

CUSTOMER DATA CENTER

OPERATIONS VIEW

Advantages and Disadvantage of MicroServices

Advantages

- Polyglot Programming
- Speed for delivery
- Scale
- Built in resiliency
- Availability

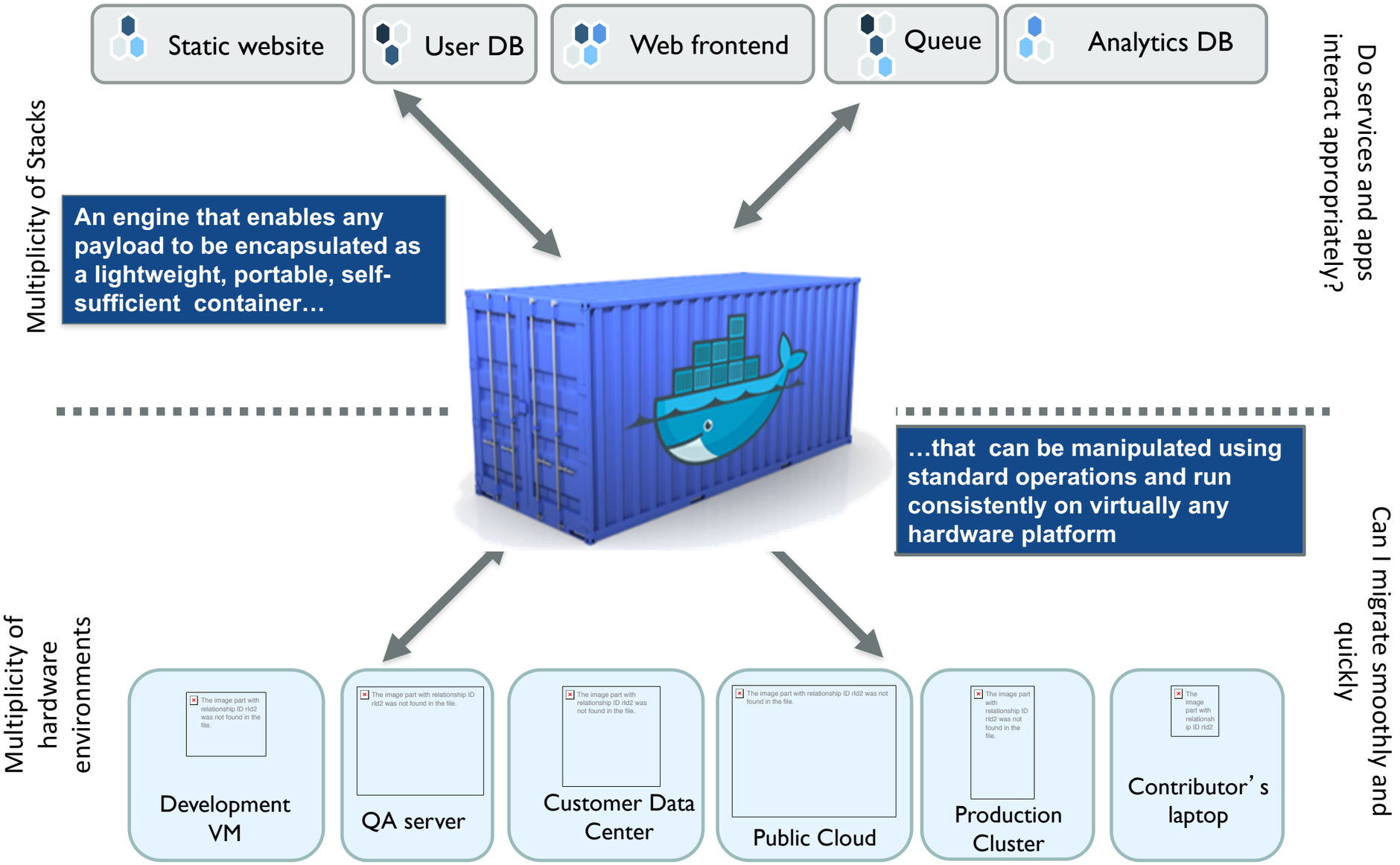
Dis-advantages

- Complex Infra
- Overhead database

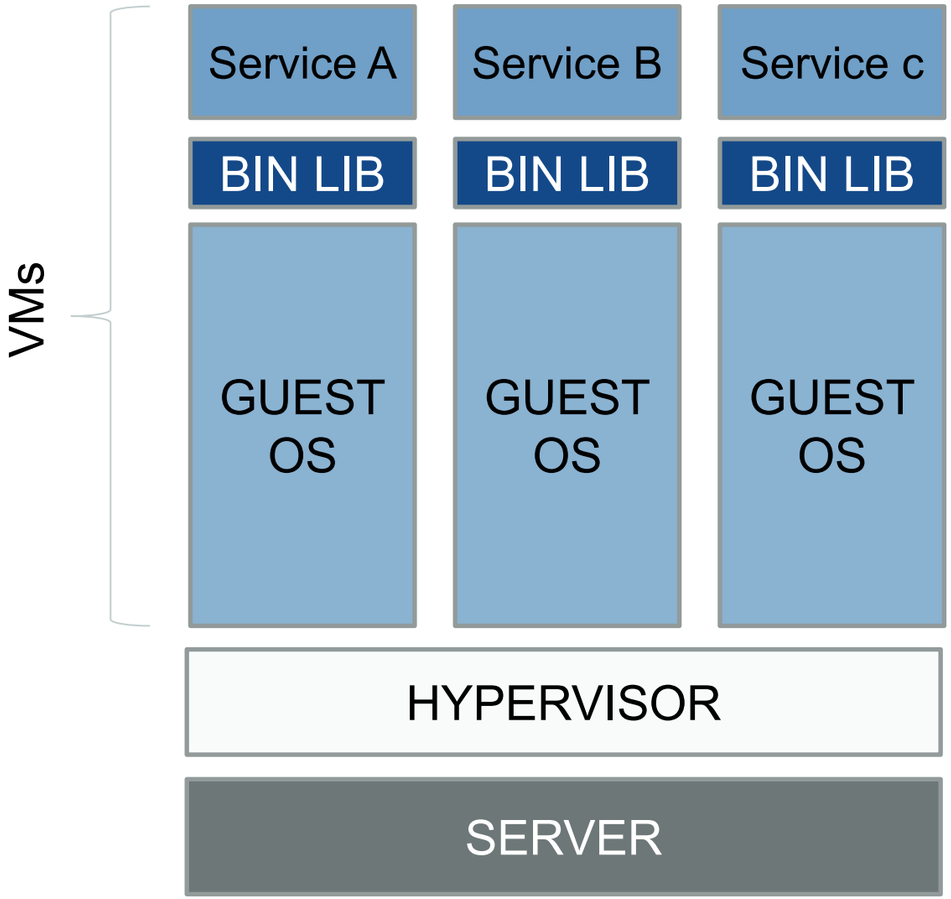
The image part with relationship ID rId2 was not found in the file.

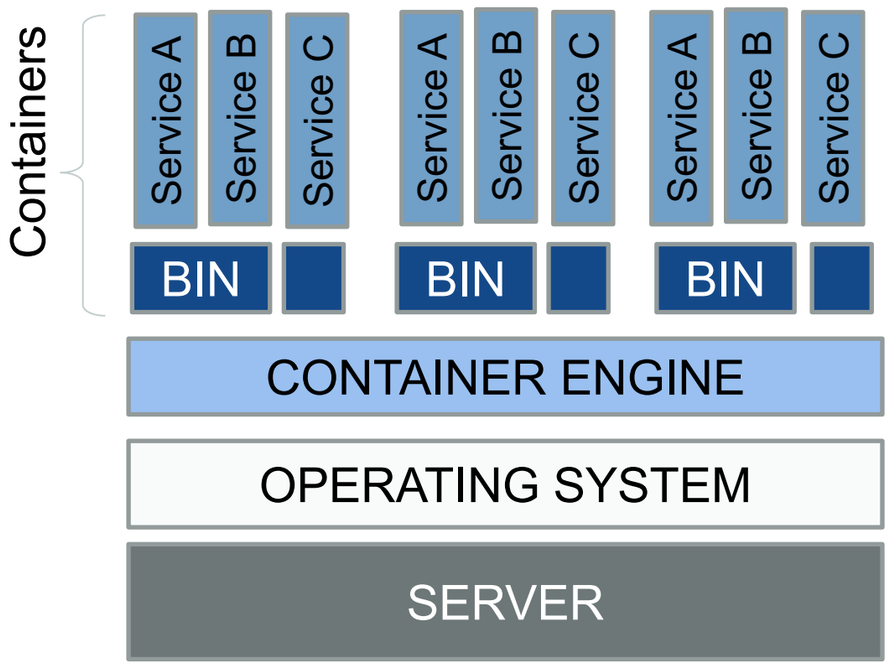
What is container?

A Shipping Container for Code



Containers (Impact on Infrastructure)

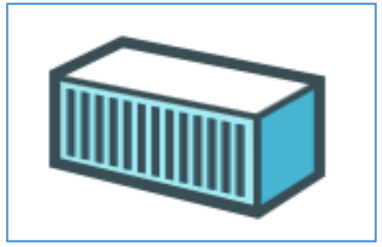




Docker Components

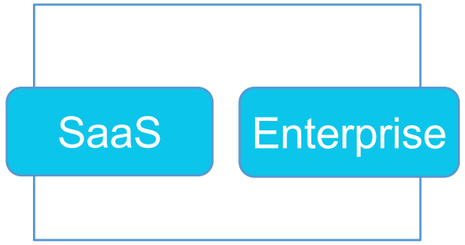
Image (Built using Dockerfile)

A read-only snapshot of a container stored in Docker Hub to be used as a template for building containers



Container

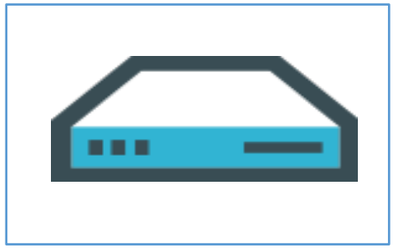
The standard unit in which the application service resides or transported



Docker Hub/Registry

Available in SaaS or Enterprise to deploy anywhere you choose

Stores, distributes, and shares container images

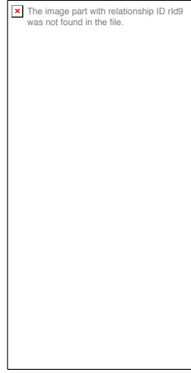


Docker Engine

A program that creates, ships, and runs application containers

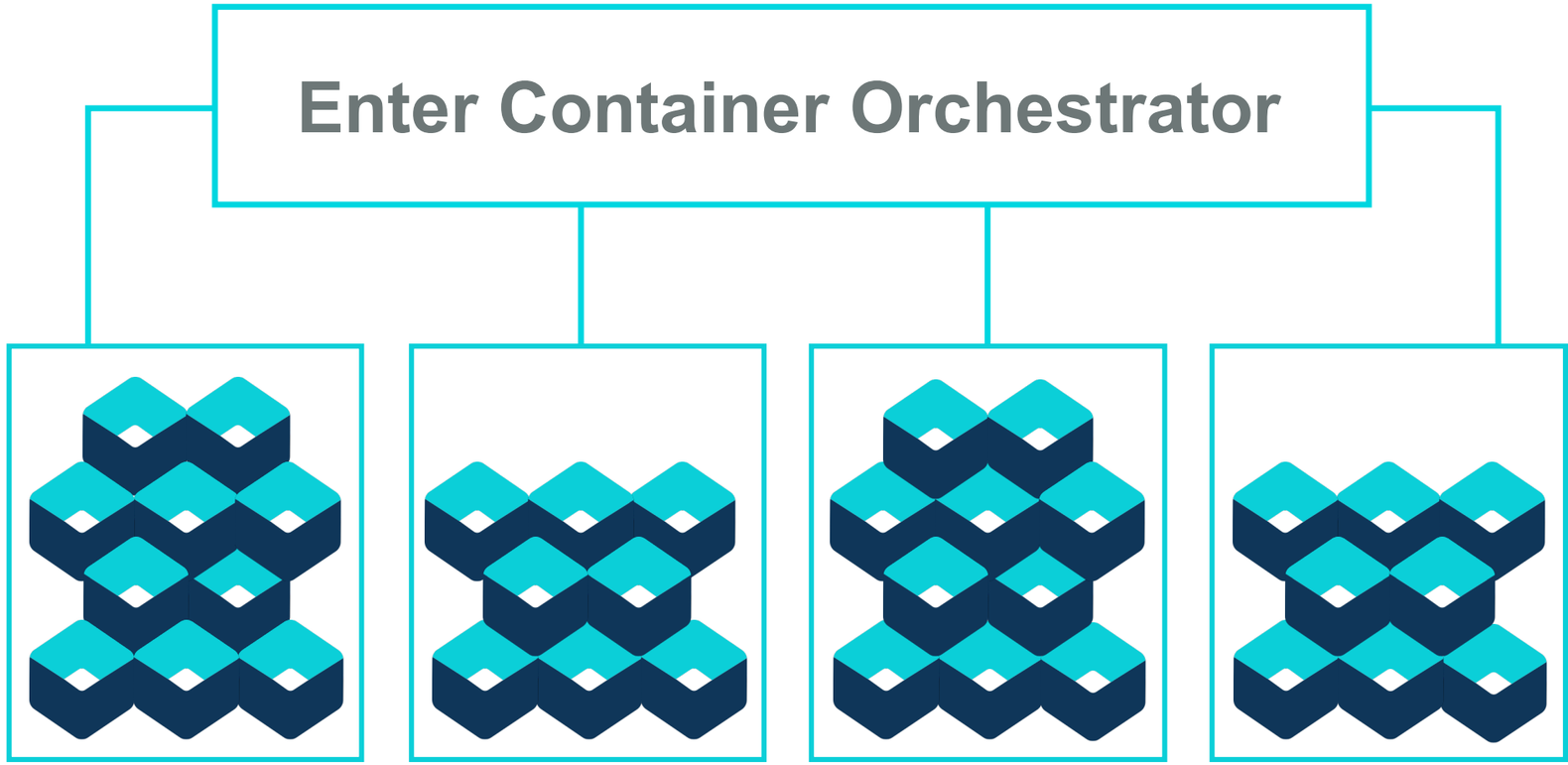
Runs on any physical and virtual machine or server locally, in private or public cloud

Client communicates with Engine to execute commands

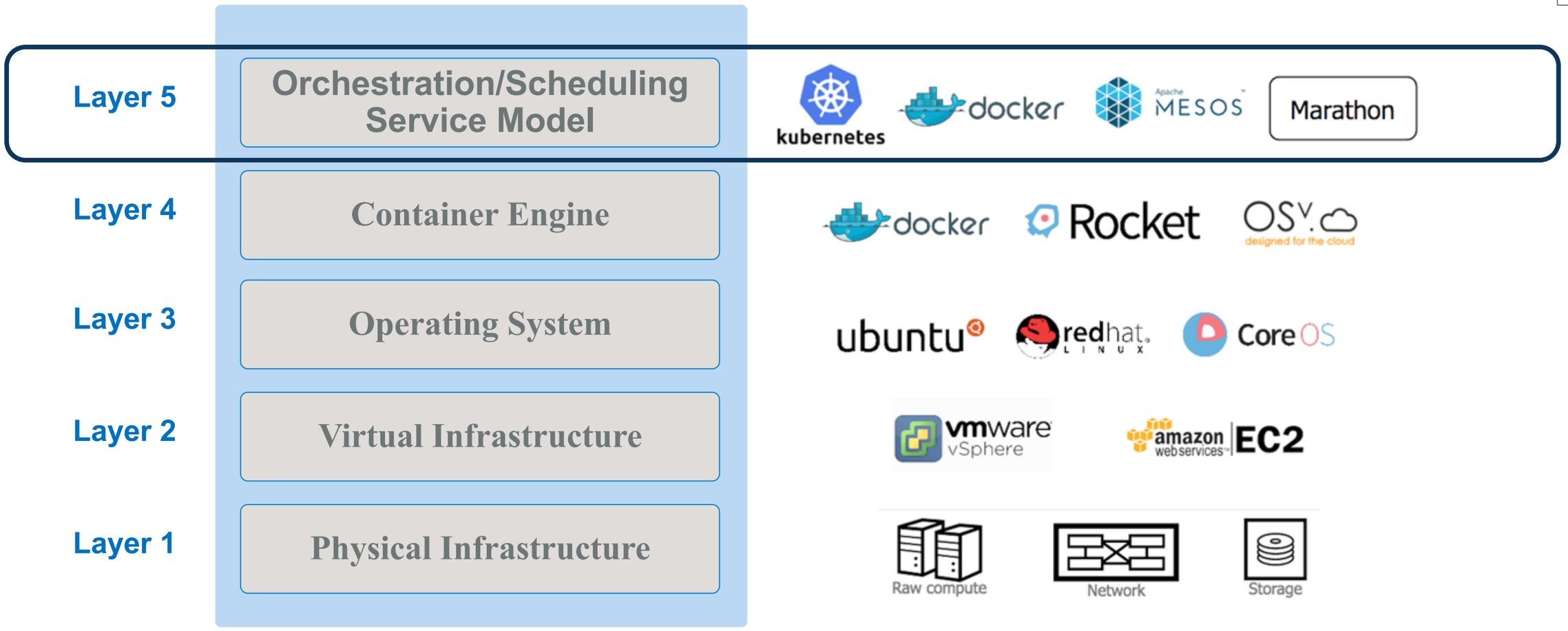


Demo : A quick look at Dockers

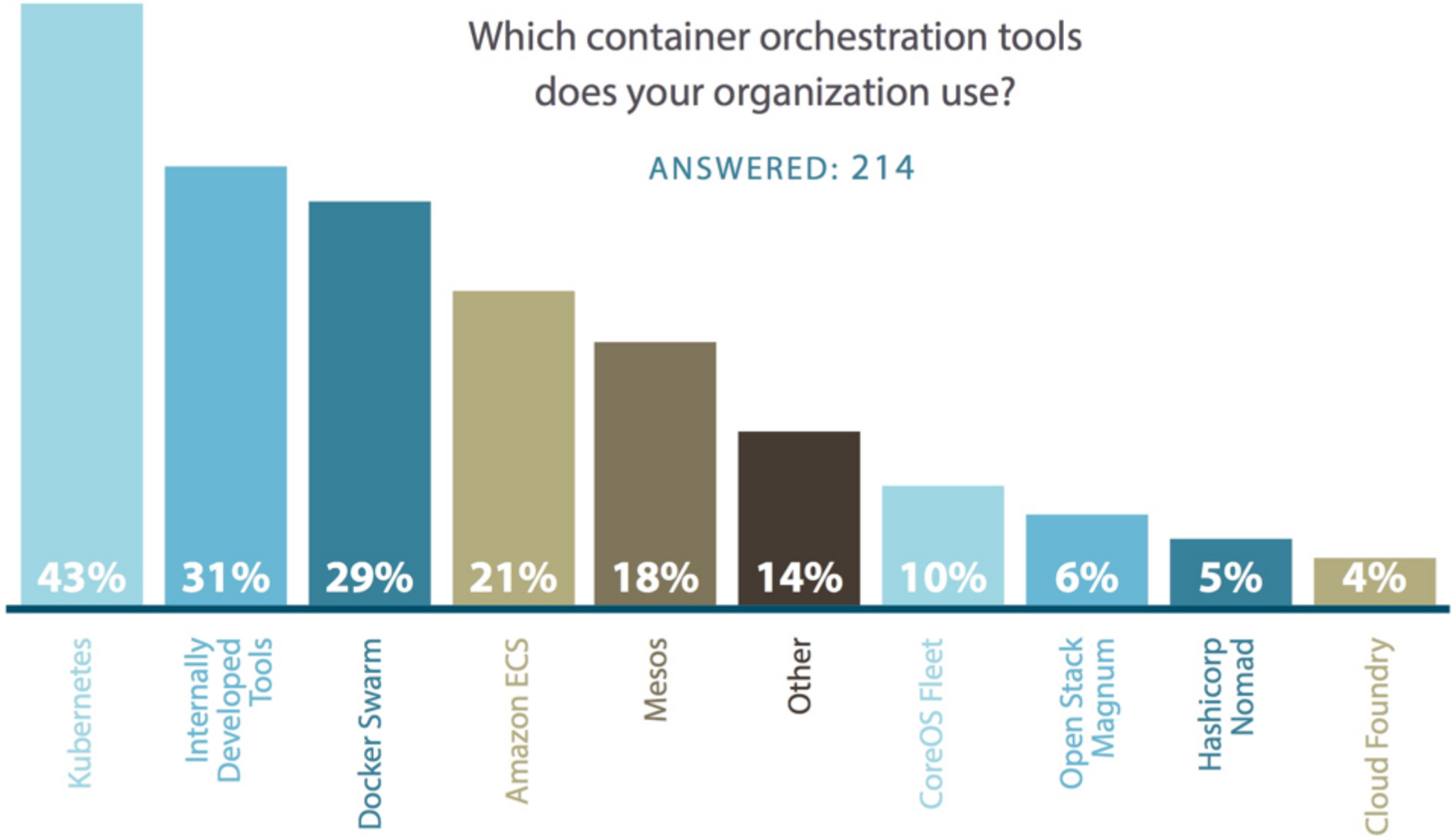
Kubernetes



Container Stack

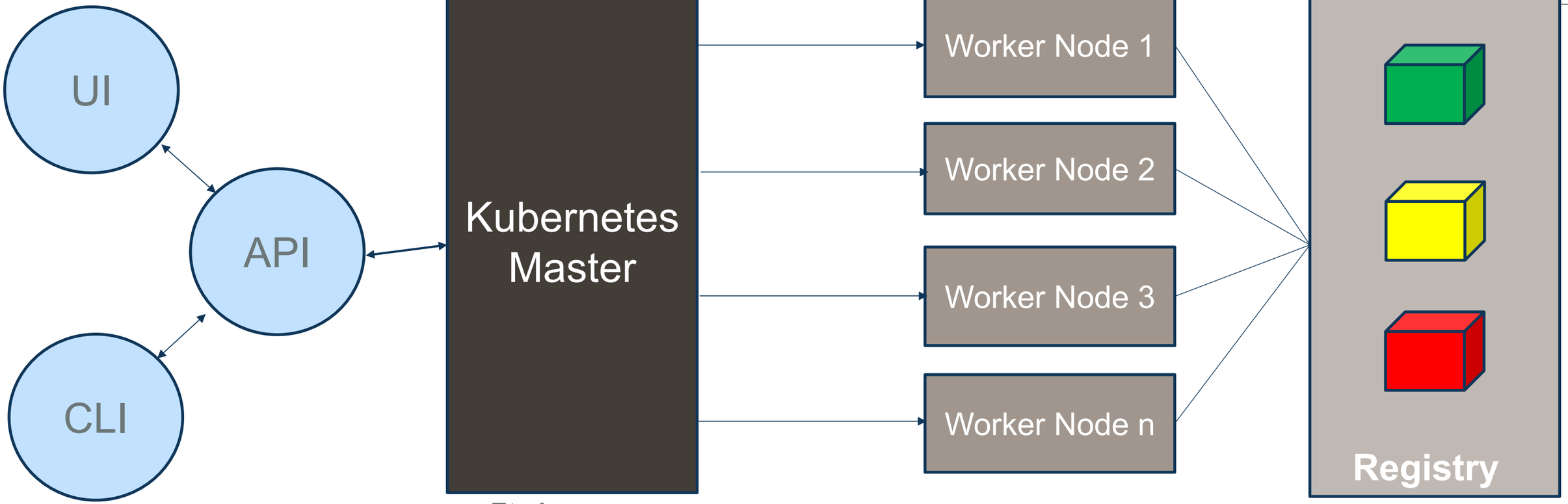


Container Orchestration



Source: devops.com

Kubernetes Architecture



- **Etcd**
- **API Server**
- **Controller Manager Server**
- **Scheduler Server**

Benefits of Kubernetes

Intelligent Scheduling

Self-healing

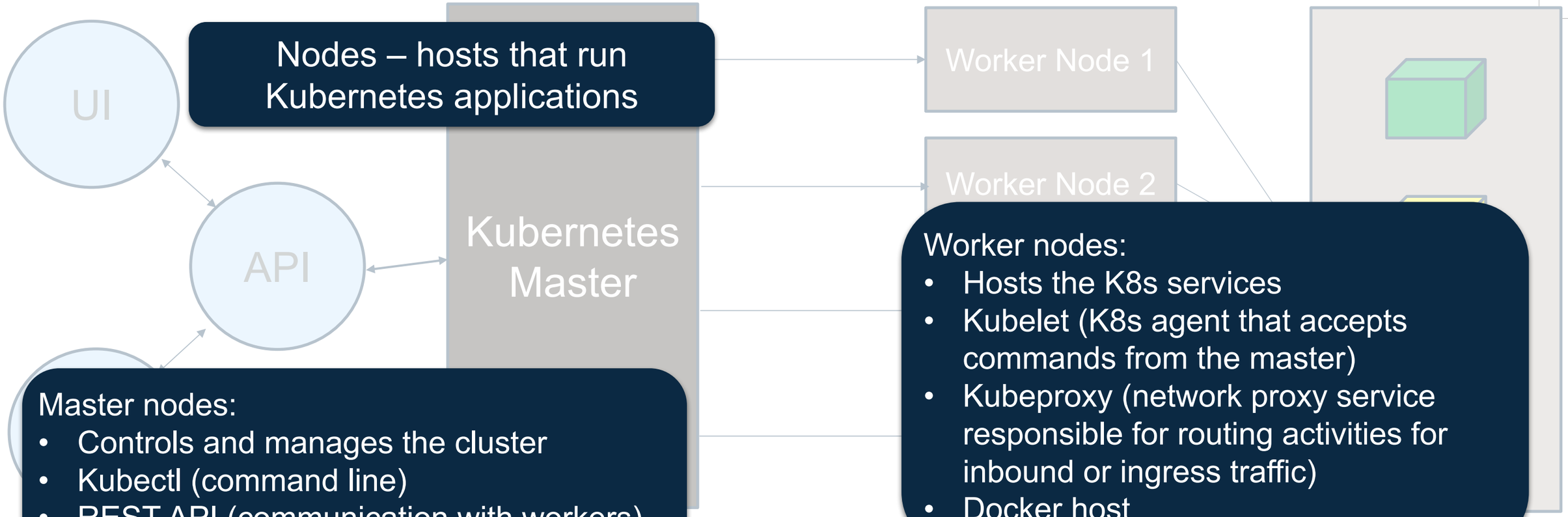
Horizontal scaling

Service discovery & load balancing

Automated rollouts and rollbacks

Secret and configuration management

Kubernetes Architecture



Nodes – hosts that run Kubernetes applications

Master nodes:

- Controls and manages the cluster
- Kubectl (command line)
- REST API (communication with workers)
- Scheduling and replication logic

Worker nodes:

- Hosts the K8s services
- Kubelet (K8s agent that accepts commands from the master)
- Kubeproxy (network proxy service responsible for routing activities for inbound or ingress traffic)
- Docker host

Kubernetes Architecture

UI

Pods:
- Smallest deployment unit in K8s
- Collection of containers that run on a worker node
- Each has its own IP
- Pod shares a PID namespace, network, and hostname

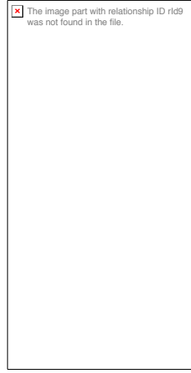
Labels:
- Metadata assigned to K8s resources
- Key-value pairs for identification
- Critical to K8s as it relies on querying the cluster for resources that have certain labels

Worker Node 3

Replication controller:
- Ensures availability and scalability
- Maintains the number of pods as requested by user
- Uses a template that describes specifically what each pod should contain

Service:
- Collections of pods exposed as an endpoint
- Information stored in the K8s cluster state and networking info propagated to all worker nodes

- Scheduler Server

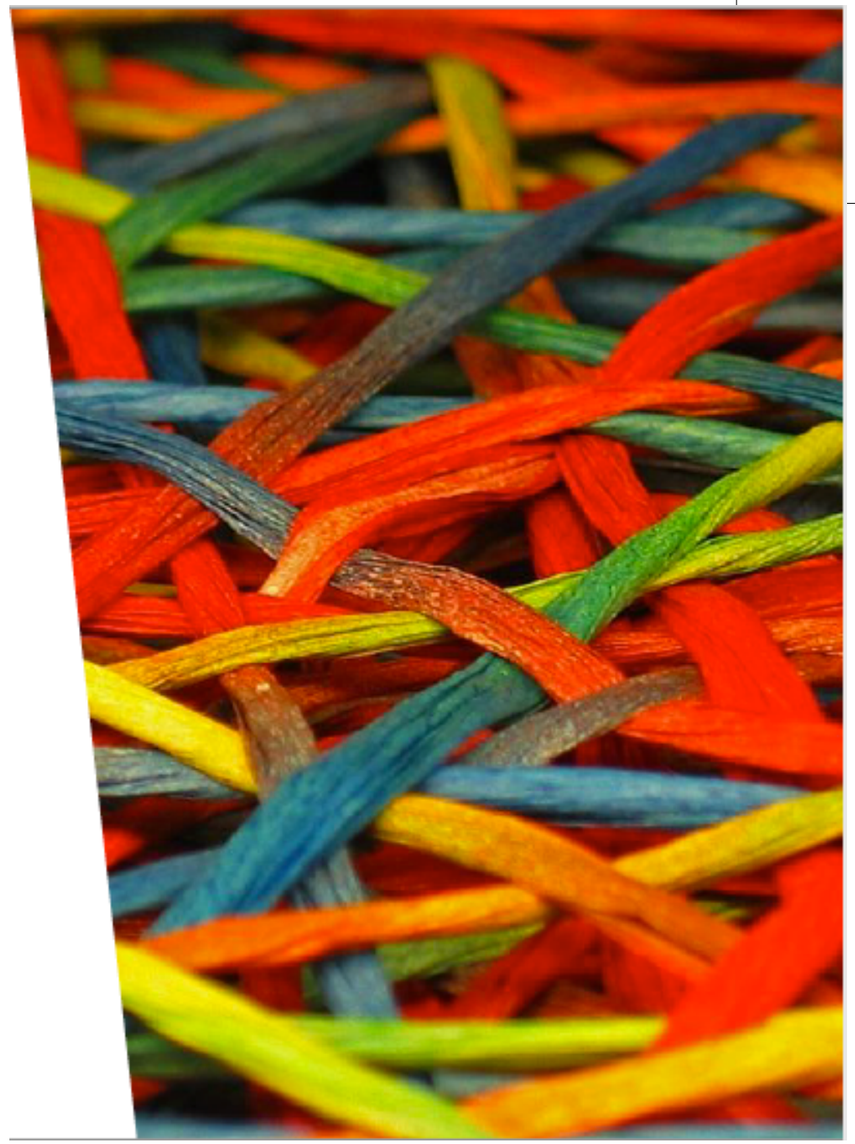


Demo : A quick look at Kubernetes

What is a ‘Service Mesh’ ?

A network for Services

- Visibility
- Resiliency and Efficiency
- Traffic Control
- Security
- Policy Enforcement



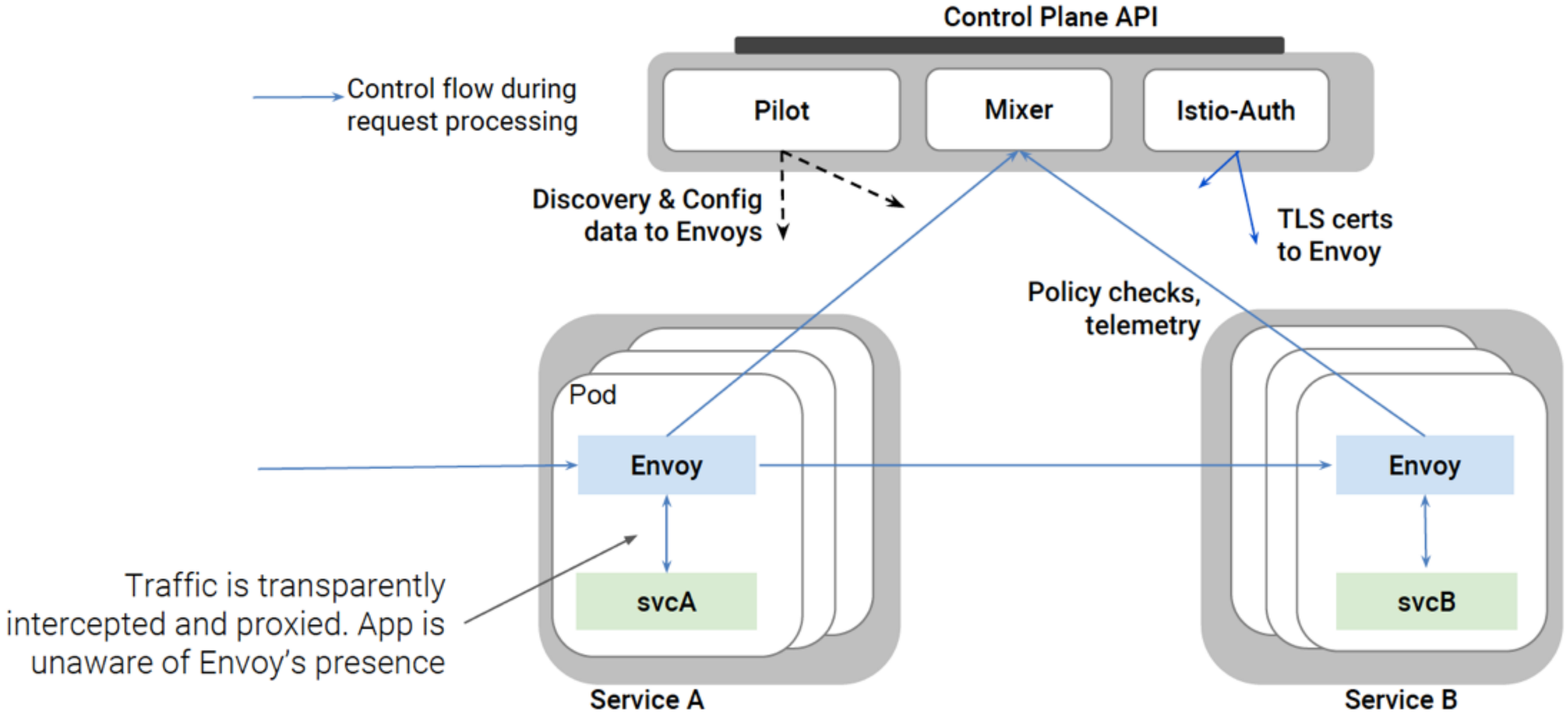
So you want to build a service mesh? What do you need?

You need control over load balancing. But stop (mis)using the kernel for it!

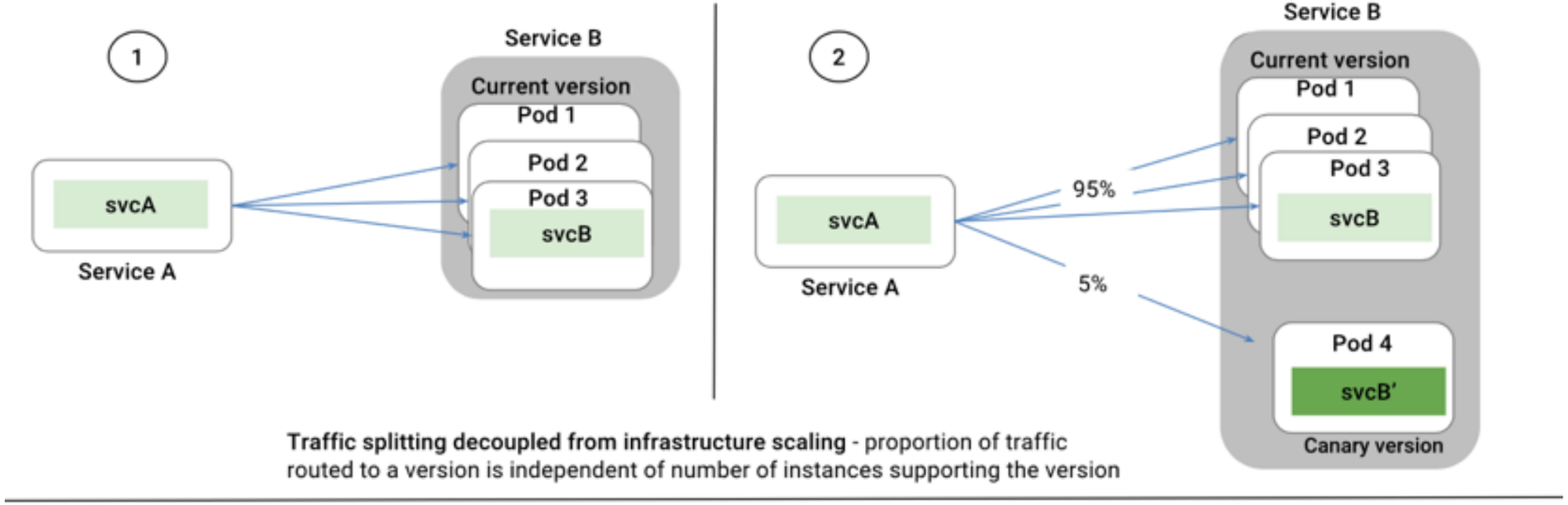
Lightweight sidecars to manage traffic between services

Sidecars can do much more than just load balancing!

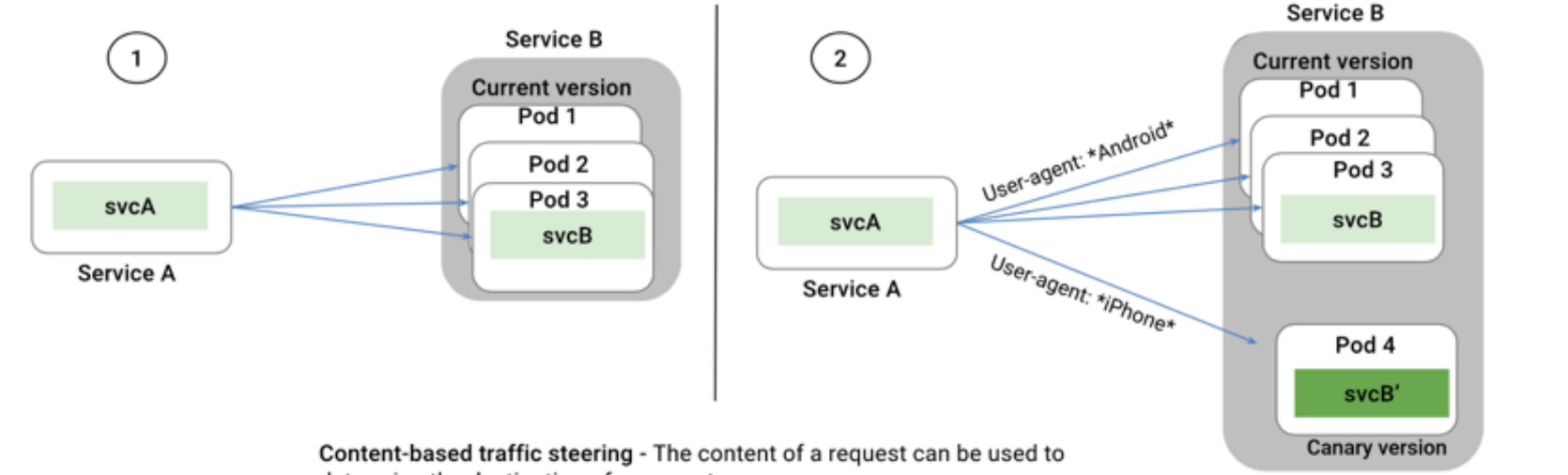
Architecture of Istio



Traffic Management (Service Routing - Options)



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

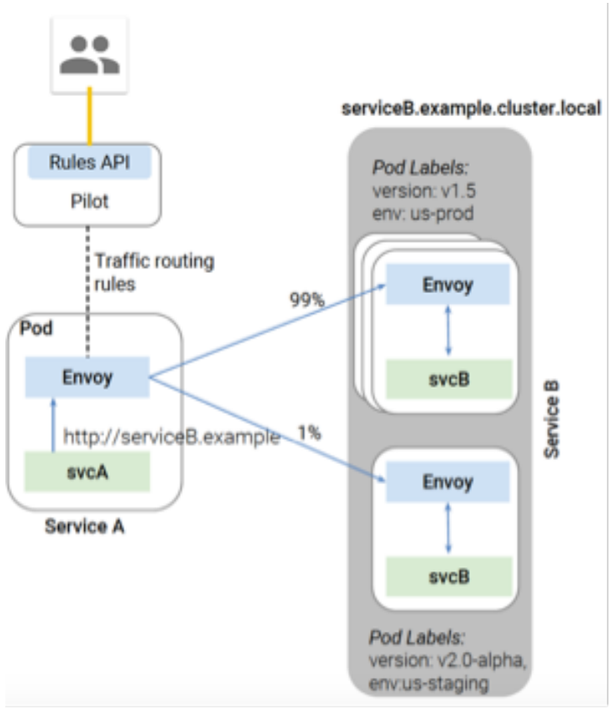
Traffic Management (Service Routing)

Traffic control is decoupled from infrastructure scaling

Sample Rule for routing

```
// A simple traffic splitting rule

destination: serviceB.example.cluster.local
match:
  source: serviceA.example.cluster.local
route:
- tags:
    version: v1.5
    env: us-prod
  weight: 99
- tags:
    version: v2.0-alpha
    env: us-staging
  weight: 1
```



Resiliency – Circuit Breakers

Istio adds fault tolerance to your application without any changes to code

```
// Circuit breakers

destination: serviceB.example.cluster.local
policy:
- tags:
    version: v1
  circuitBreaker:
    simpleCb:
      maxConnections: 100
      httpMaxRequests: 1000
      httpMaxRequestsPerConnection: 10
      httpConsecutiveErrors: 7
      sleepWindow: 15m
      httpDetectionInterval: 5m
```

Resilience features

Timeouts

Retries with timeout budget

Circuit breakers

Health checks

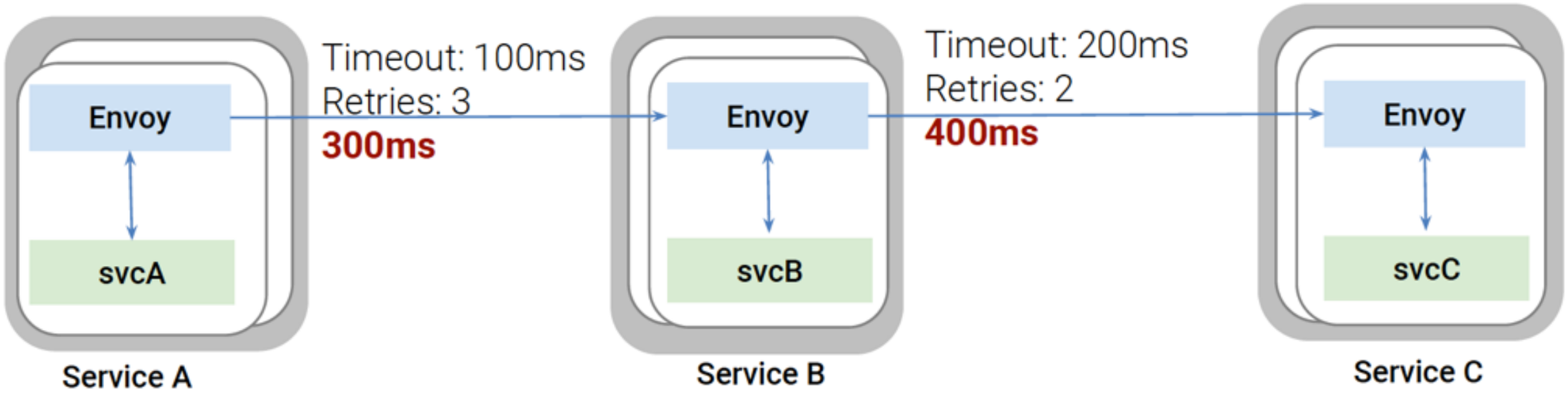
Control connection pool size and request load

Systematic fault injection

Resiliency Testing - Fault Injection

Systematic fault injection to identify weaknesses in failure recovery policies

-
-



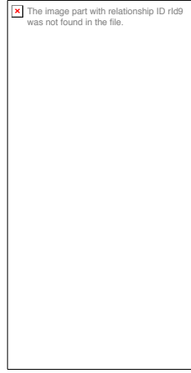
Fault Injections

Injecting Delay for 10% of requests

```
destination: reviews.default.svc.cluster.local
route:
- tags:
    version: v1
httpFault:
  delay:
    percent: 10
    fixedDelay: 5s
```

- Returns error 400 for 10% of requests

```
destination: "ratings.default.svc.cluster.local"
route:
- tags:
    version: v1
httpFault:
  abort:
    percent: 10
    httpStatus: 400
```



Samples (demo) for Istio