Zoom In, Zoom Out.

A Fresh look at Kubernetes Security

Rob Richardson
Technical Evangelist - MemSQL
@Rob_Rich | @MemSQL

Kavya Pearlman
Global Cybersecurity Strategist - Wallarm
@KavyaPearlman | @Wallarm

wallarm

memSQL

Introducing Rob...

wallarm

Rob Richardson

- Tech Evangelist for MemSQL
- Microsoft MVP
- Leads the Southeast Valley .NET User Group
- AZGiveCamp Organizer

Personal interests

Travel, Coding, and Teaching

Introducing Kavya...

Kavya Pearlman

- Well known as the “Cyber Guardian”
- Cybersecurity Strategist at Wallarm
- An Award-winning Cybersecurity Professional
- Founder and CEO of XR Safety Initiative
- Former Information Security Director Linden Lab
- Former Facebook Third Party Security Risk Advisor

Personal interests

Travel, Gaming, Virtual Worlds

Agenda

Let's Talk About Kubernetes!

- Overview of Containers
- Monolithic vs Microservices
- What is Kubernetes and its Benefits
- Securing K8 - Zooming in
  - Essentials to build a secure Kubernetes environment
- Securing K8 - Zooming Out
  - Do’s and Don’ts for Containerized Environments
- How Istio and Service Mesh can affect security
- Conclusion

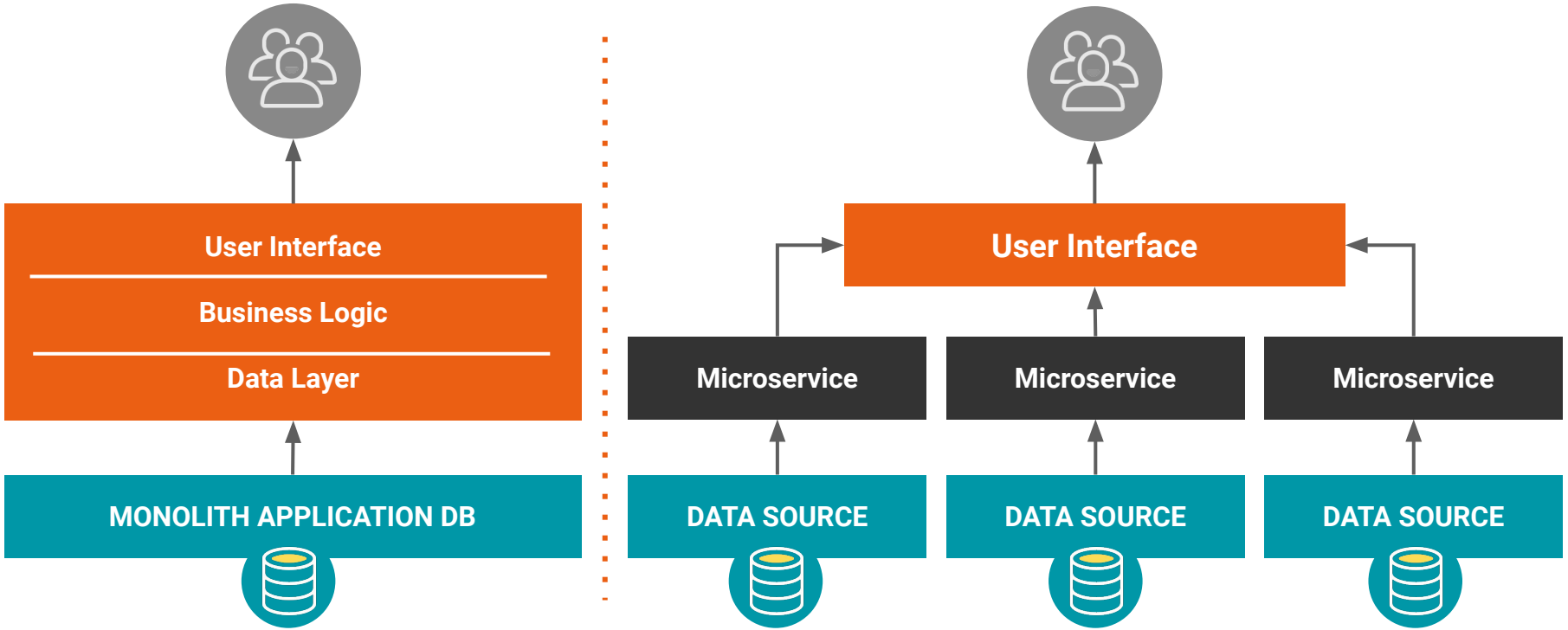
Kubernetes - Getting started

KUBERNETES NEEDS NEW SECURITY MINDSET

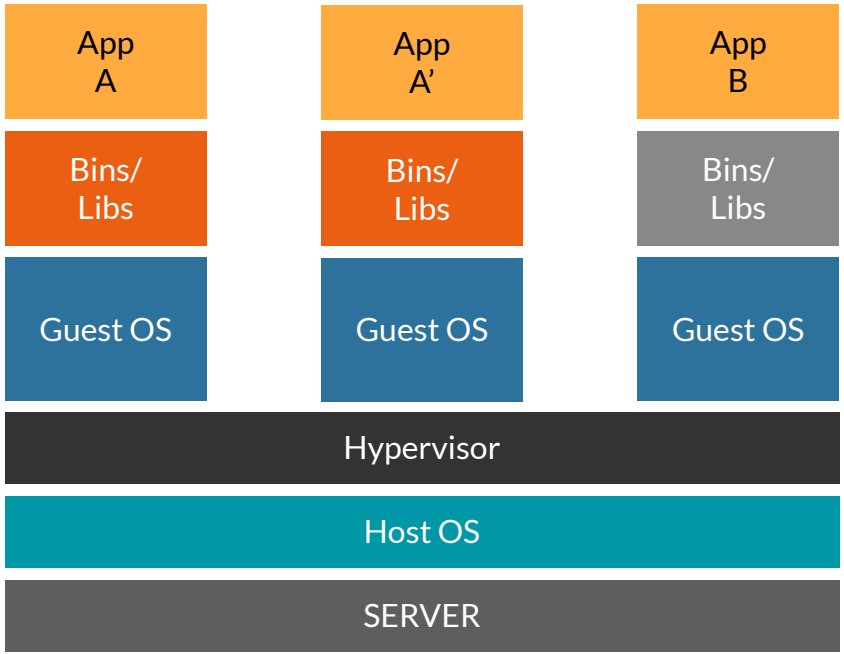
Cloud-native applications and infrastructure create several new challenges for all of us security professionals. We need to establish new security programs, have a new mindset and adopt advanced new tools that are focused primarily on securing cloud-native technologies."

- Kavya Pearlman

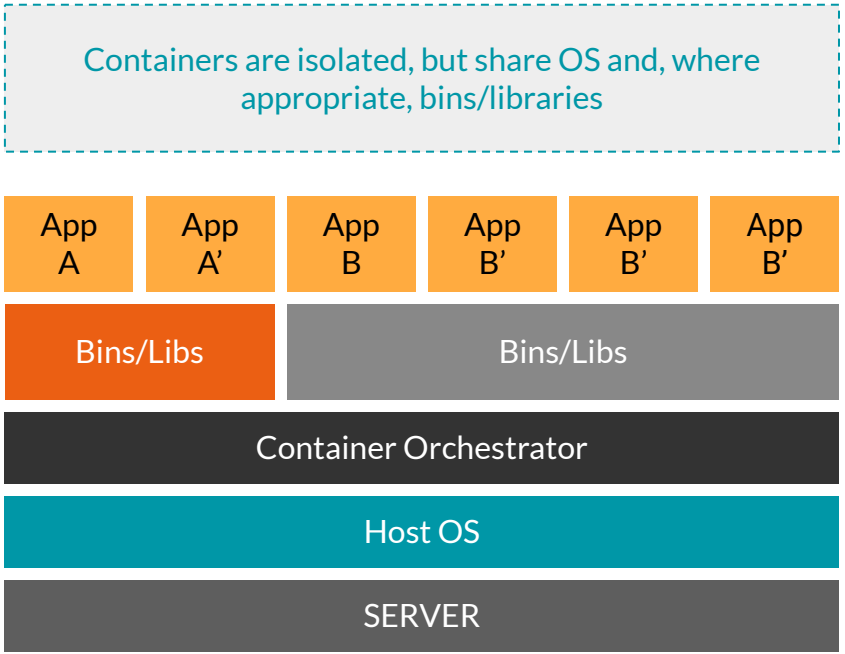
Monolith vs. Microservices



Containers vs. VMs

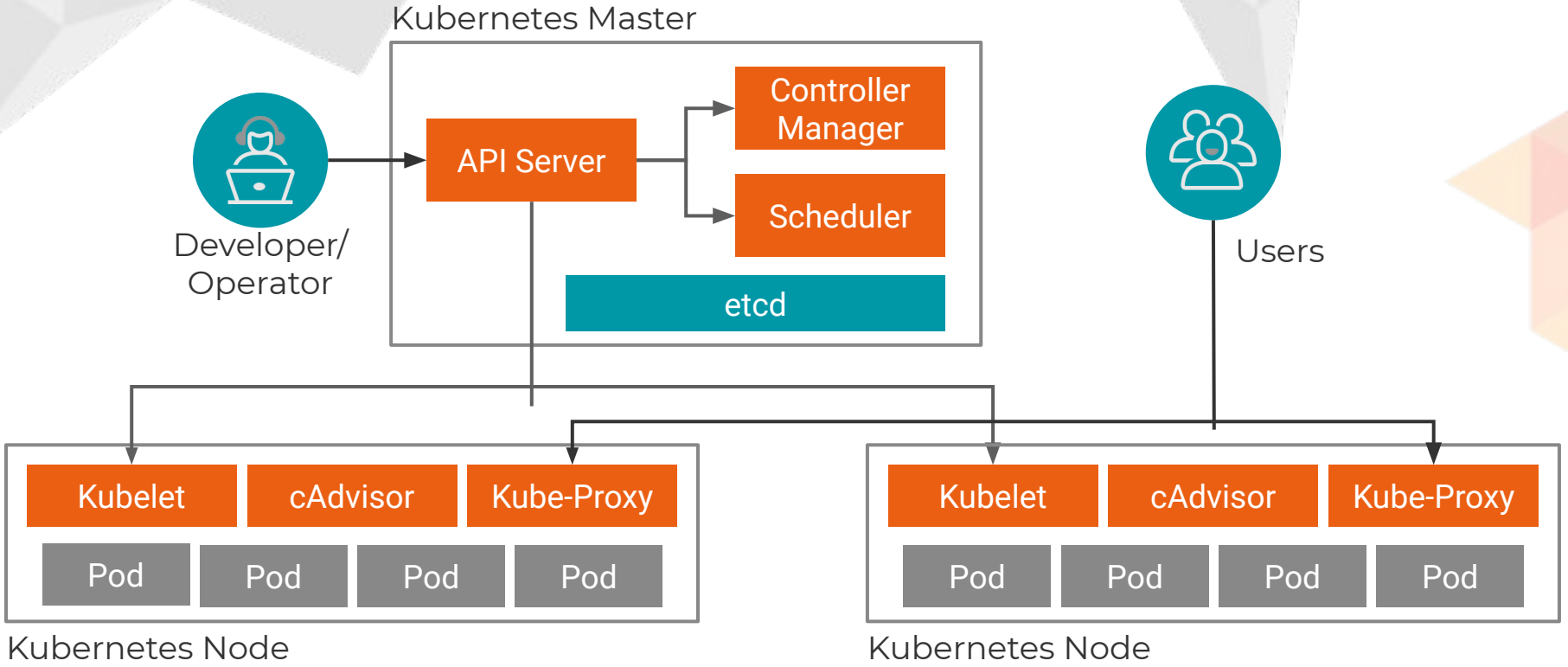


VIRTUAL MACHINE



CONTAINERS

What is Kubernetes?



Benefits of using Kubernetes

Bring new products
to market faster

Avoid vendor lock-in

Enjoy peace of mind that
your applications are always on
Kubernetes self-heals
Kubernetes auto-scales

Benefits of using Kubernetes

It’s the de facto standard for running cloud-native applications at scale

Free community support or paid professional services

Kubernetes - Zooming In

The Essentials for Building a Secure Kubernetes Environment

wallarm

wallarm memSQL

Shopify Breach

Caused by lack of
K8 security Essentials

Exploited Weakness

API configuration flaw

Type of attack

SSRF Attack whereby metadata used to steal API keys and credential packets

Effect

Thousands of stores and store-clients information was exposed

TIMELINE

0xacb submitted a report to **Shopify**. Apr 22nd (about 1 year ago)

The Exploit Chain - How to get root access on all Shopify instances

1 - Access Google Cloud Metadata

- 1: Create a store (partners.shopify.com)
- 2: Edit the template `password.liquid` and add the following content:

```
<script>
window.location="http://metadata.google.internal/computeMetadata/v1beta1/instance/service-accounts/default
// iframes don't work here because Google Cloud sets the `X-Frame-Options: SAMEORIGIN` header.
</script>
```

- 3: Go to https://exchange.shopify.com/create-a-listing and install the Exchange app
- 4: Wait for the store screenshot to appear on the Create Listing page
- 5: Download the PNG and open it using image editing software or convert it to JPEG (Chrome displays a black PNG)

wallarm

memSQL

Tesla Breach

Caused by lack of
K8 security Essentials

Exploited Weakness:

Kubernetes instance and an insecure administrative console

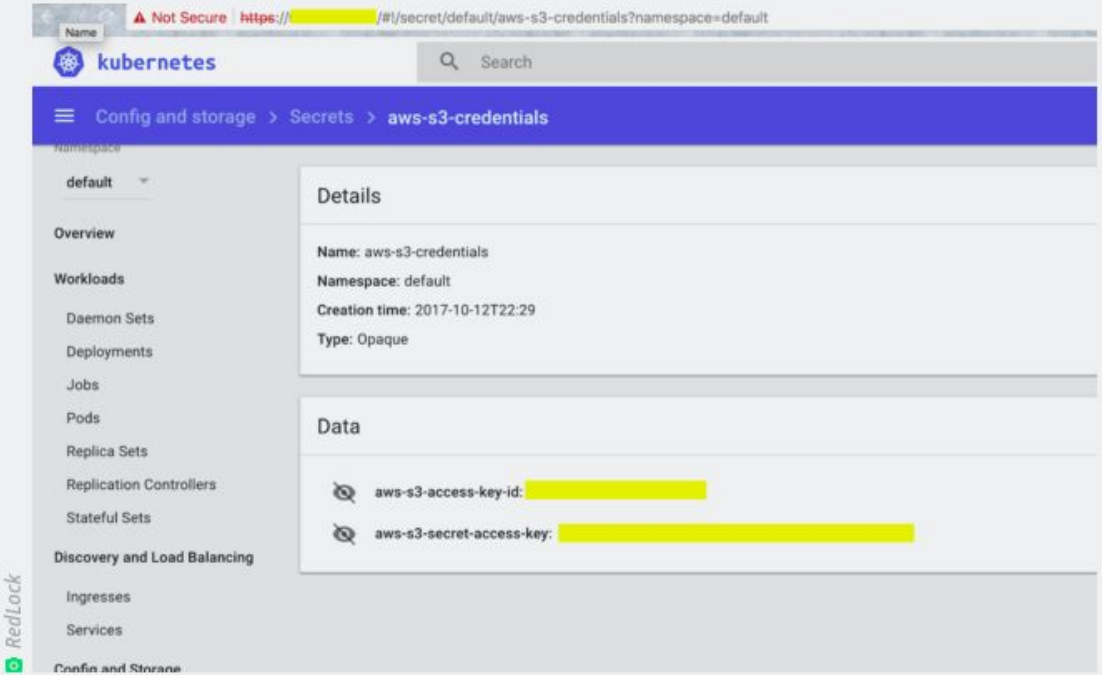
Type of attack

False credentials

Effect

The total scope of the breach is yet unknown

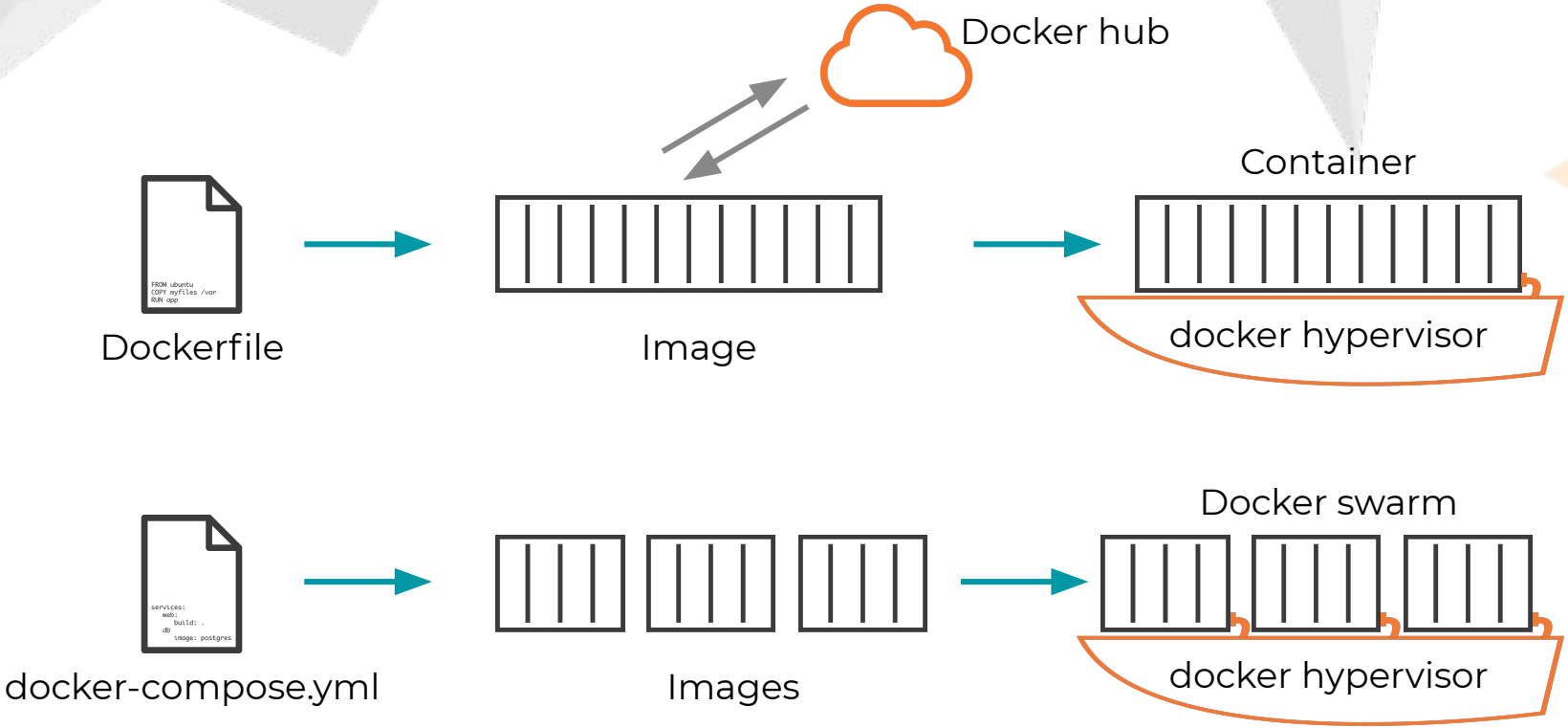
The initial point of entry for the Tesla cloud breach, Tuesday's report said, was an unsecured administrative console for Kubernetes, an open source package used by companies to deploy and manage large numbers of cloud-based applications and resources.



wallarm

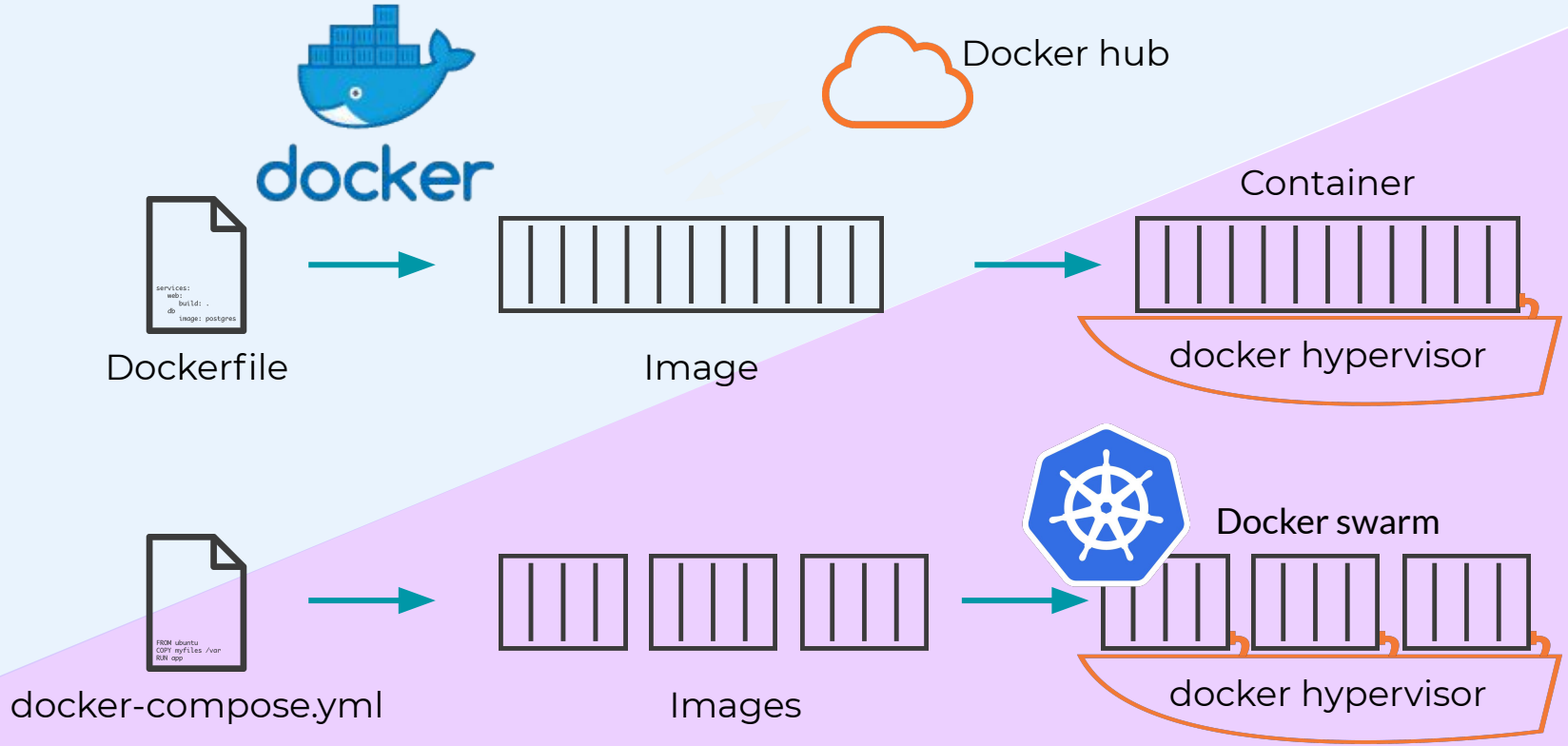
memSQL

What is Docker?



Docker ecosystem, infographic by Rob Richardson robrich.org

What is Kubernetes?



Docker ecosystem, infographic by Rob Richardson robrich.org

Namespaces

“K8s does not provide a mechanism to enforce security across Namespaces. You should only use it within trusted domains and not use when you need to be able to provide guarantees that a user of the cluster or pods be unable to access any of the other Namespaces resources”

--GCP Team

tl;dr: A namespace is not a security boundary for inter-pod communication.

Role based access control (RBAC)

wallarm

Roles and **ClusterRoles** are a **whitelist**; essentially a list of the allowed permissions.

RoleBindings and **ClusterRoleBindings** marry users to roles:

- **Subject** includes the person, place, or thing that has been whitelisted.

  Ex) a developer, DevOps, a team member, user, or process.
- **Resource** is the kind of object

  Ex) pod, service, the cluster itself, or another logic instance related to Kubernetes.
- **Operations** that are whitelisted are action we permit the system to do. It's an action related to REST method.
- **Namespace** is the kubernetes section that is allowed.

Network Policies

wallarm

“By default, pods are not isolated; they accept traffic from any source.”

GCP – https://kubernetes.io/docs/concepts/services-networking/network-policies/

Secure traffic between containers

using service mesh tools like Istio

Disable legacy APIs

etcd access from worker nodes (Shopify)

Restrict API/ Dashboard access

(Tesla)

Kubernetes: Pod security policies

wallarm

Smallest base container

Smallest base container

Don’t install unnecessary software

Note: Don’t run as Root

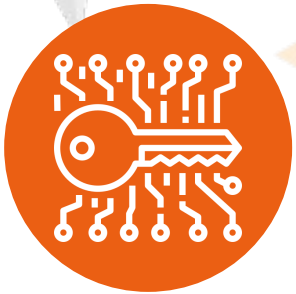
Configuration Management

Config File in Container

must trust developers, registry, git repo

Environment Variables

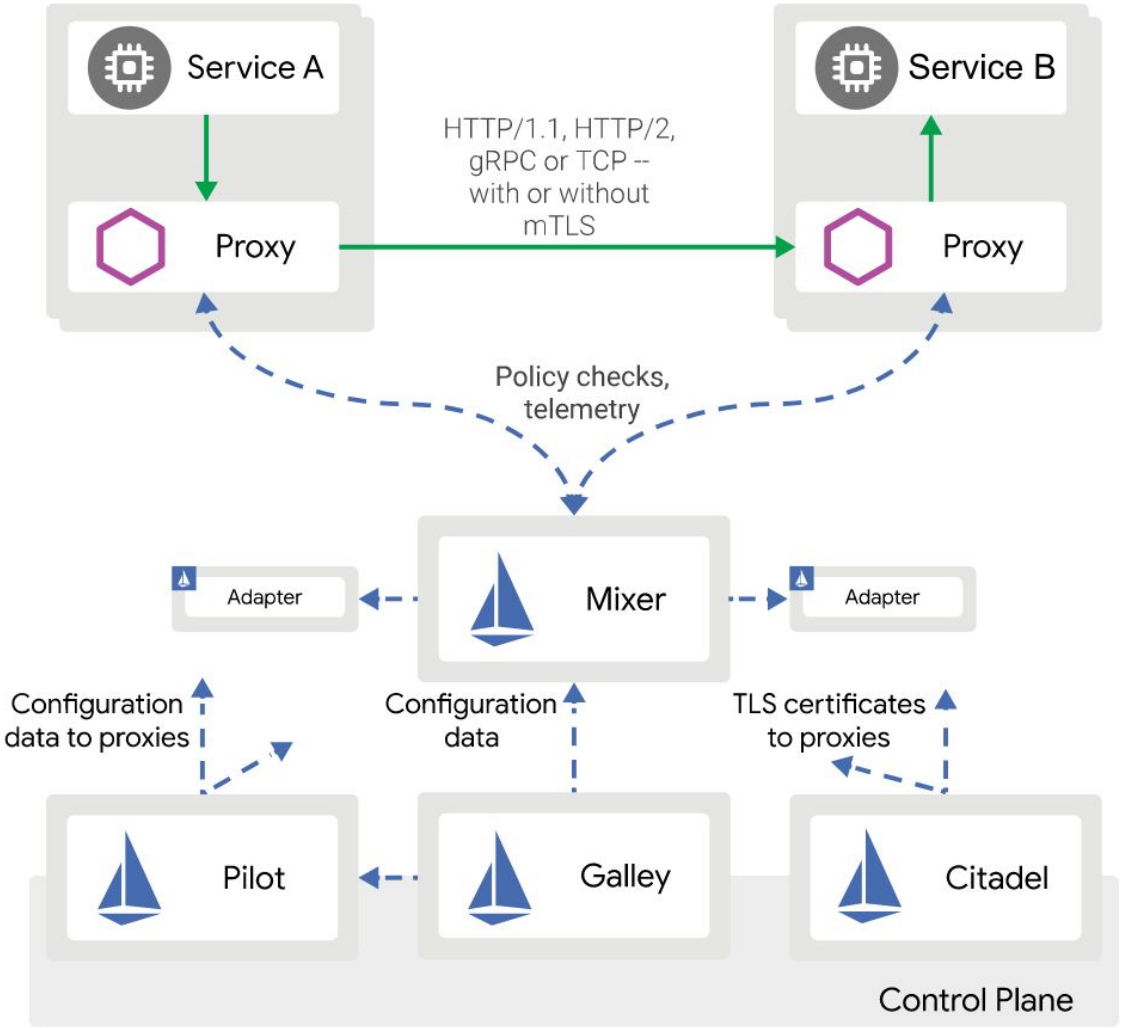
Must trust operations



External Key Vault

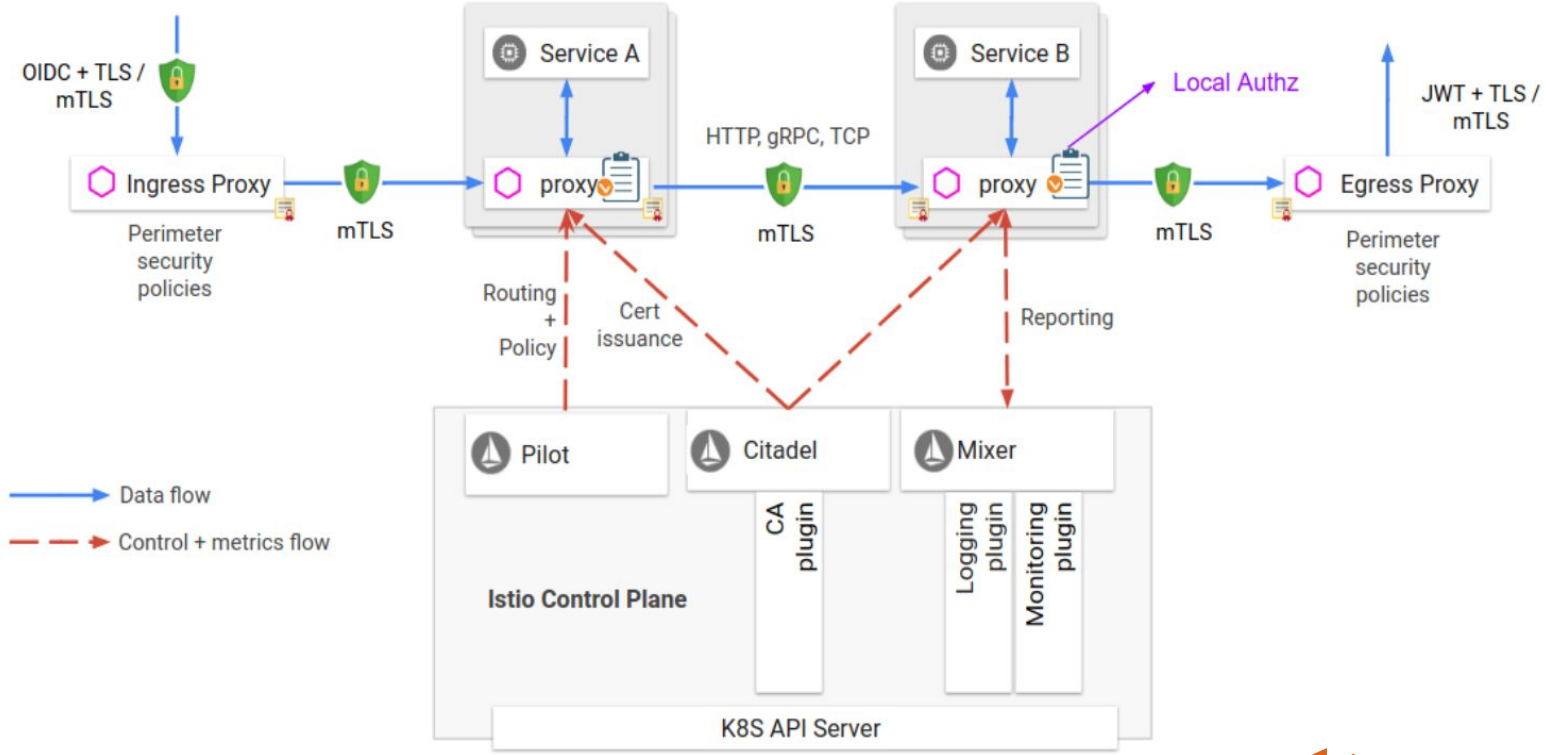
Must change application

Note: RBAC is usually best

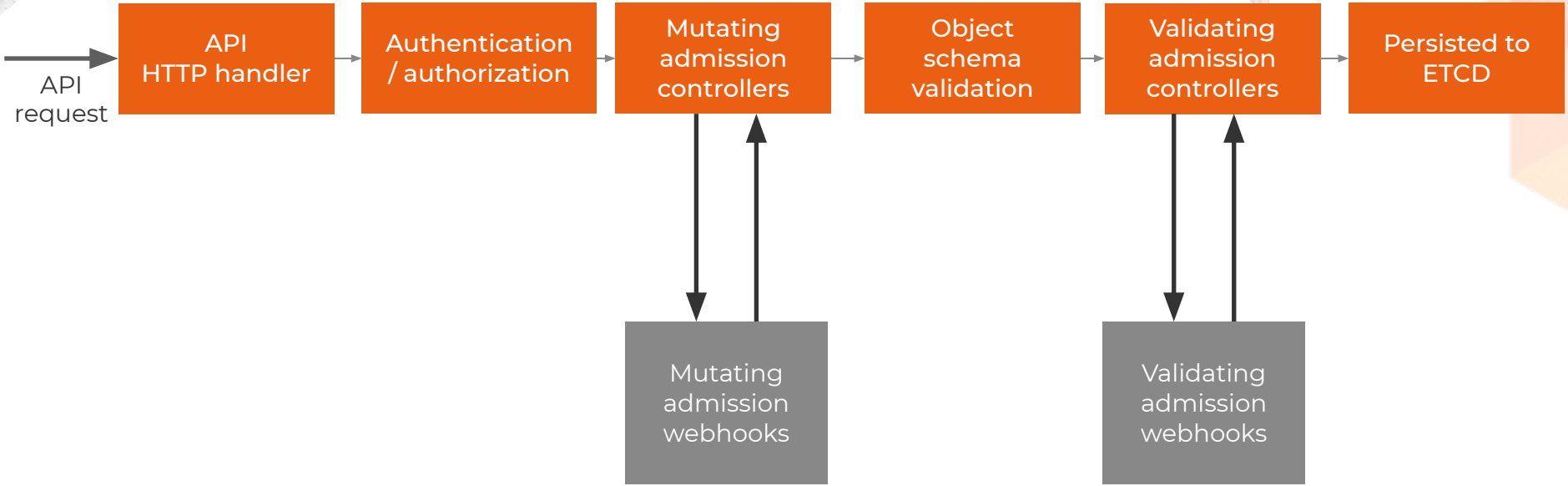


Istio Service Mesh

Istio Service Mesh



Kubernetes API request lifecycle

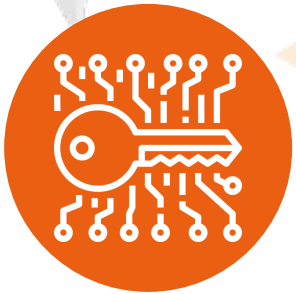


What's next?

wallarm

Orchestrator vulnerabilities

Container Content Vulnerabilities



Client-side Vulnerabilities

Injection attacks and cross-site scripting

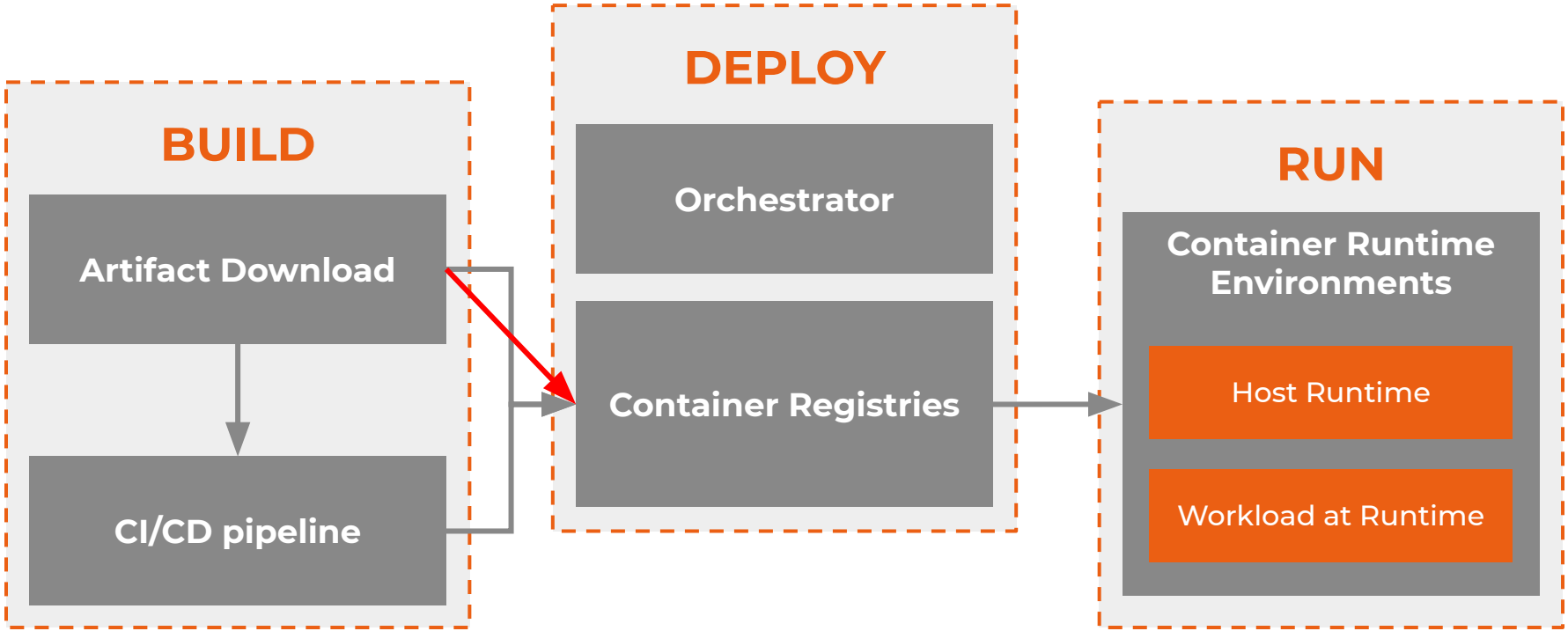
Note: enumerate and secure all the things

wallarm

Kubernetes - Zooming Out

Do’s and Don’ts for Containerized Environments

Build. Deploy. Run.

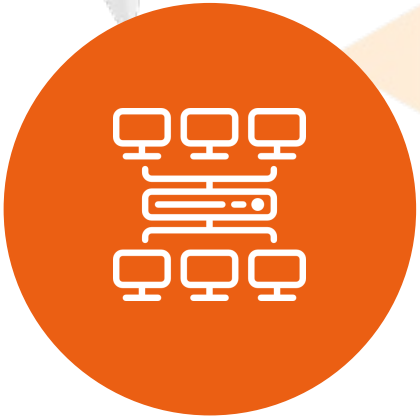


DOs for Containerized Environments

wallarm

CREATE IMMUTABLE CONTAINERS

RUN IMAGES ONLY FROM TRUSTED SOURCES



USE CONTAINER-NATIVE MONITORING TOOLS

Open Source Tools For Container Security

wallarm

anchore

DAGDA

NOT To Dos for Containerized Environments

wallarm

- Don’t install an operating system in a container
- Don’t run unnecessary services
- Don’t store critical data in a container
- Don’t put hard-coded credentials for accessing Registry
- DON’T run a container as root

Securing Kubernetes with a Service Mesh like Istio

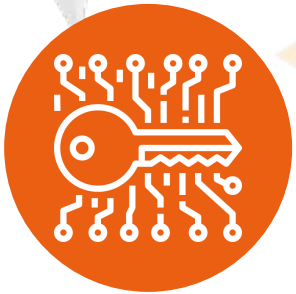
wallarm

Observe

map, log, discover

Control

access policies, rate limits, a/b testing, canary channel, inject faults, circuit breaker



Secure

mutual tls between containers

wallarm

Kavya Pearlman

@KavyaPearlman

www.wallarm.com

Rob Richardson

@rob_rich

robrich.org