Cloud Native Security 101

Michael Hausenblas, Developer Advocate, Red Hat

@mhausenblas

2019-02-02, FOSDEM lightning talk

Cloud Native !?

redhat.

Two definitions

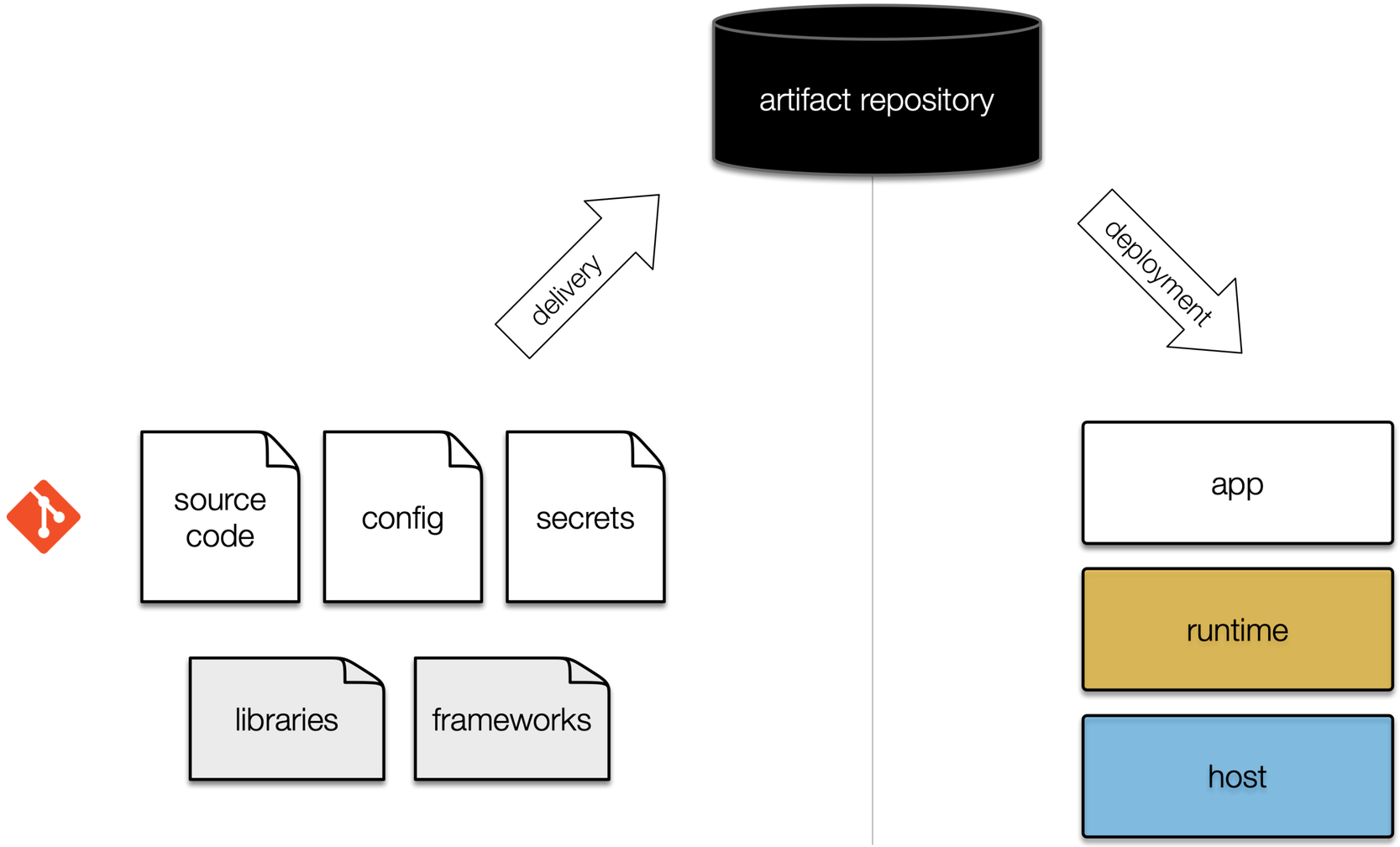
- Using APIs of a public cloud provider
- Cloud Native Computing Foundation (CNCF)

"Now! … *That* should clear up a few things around here!"

> Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.
>
> These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

redhat.

Cloud native flow



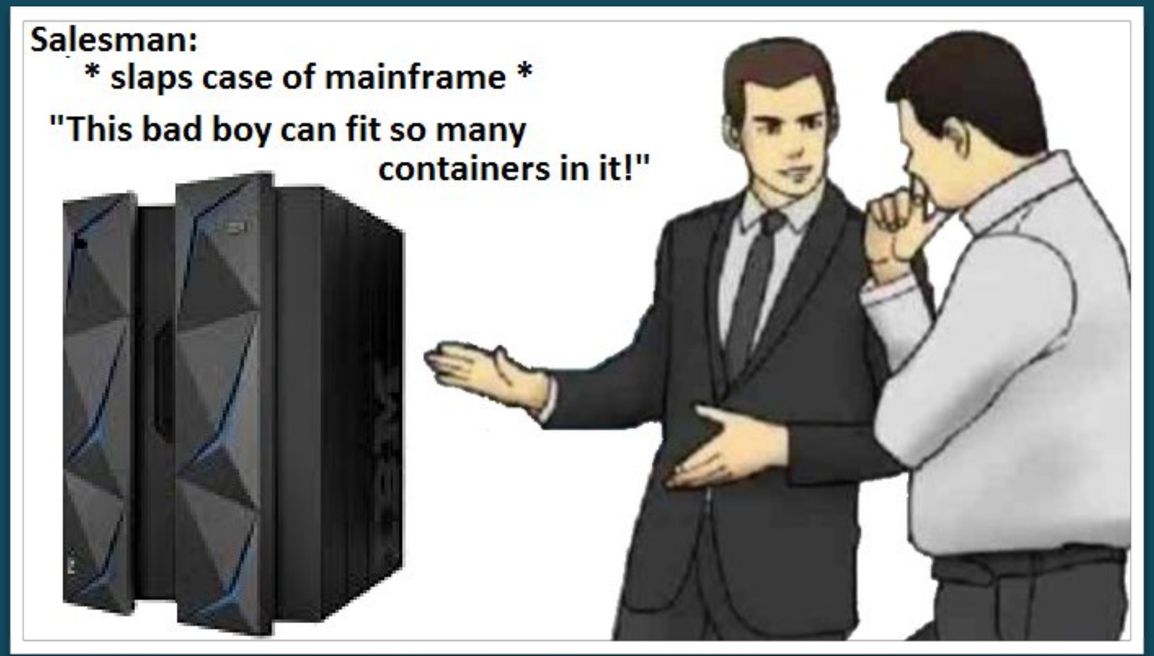
Containers and serverless: similar, yet different

	containerized microservices (Kubernetes)	serverless (AWS Lambda)
unit of deployment	pod	function
build artifact	container image	ZIP, JAR file
artifact distribution	container registry	S3 buckets
event triggers	no built-ins, requires framework	built-in (API Gateway, S3 buckets, etc.)
state	can be stateful, requires some effort	stateless, but lots of integrations

Containers and serverless: similar, yet different

	containerized microservices (Kubernetes)	serverless (AWS Lambda)
latency	generally good	can be challenging
observability	not opinionated, integration points	AWS specific
billing	pay for resources, no matter if used or not	pay what you consume
lift and shift	possible	no
local development	doable	limited

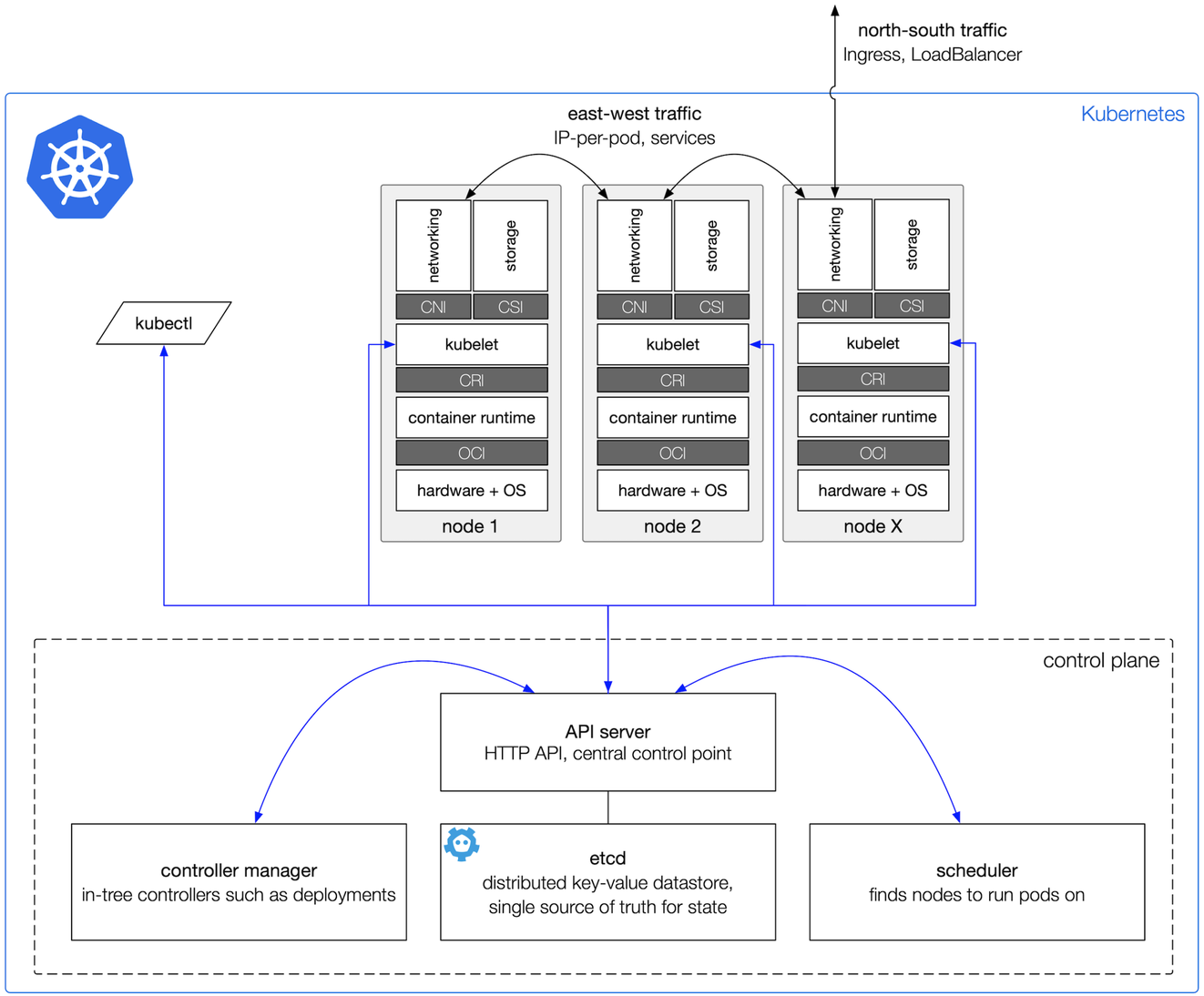
Kubernetes



redhat.

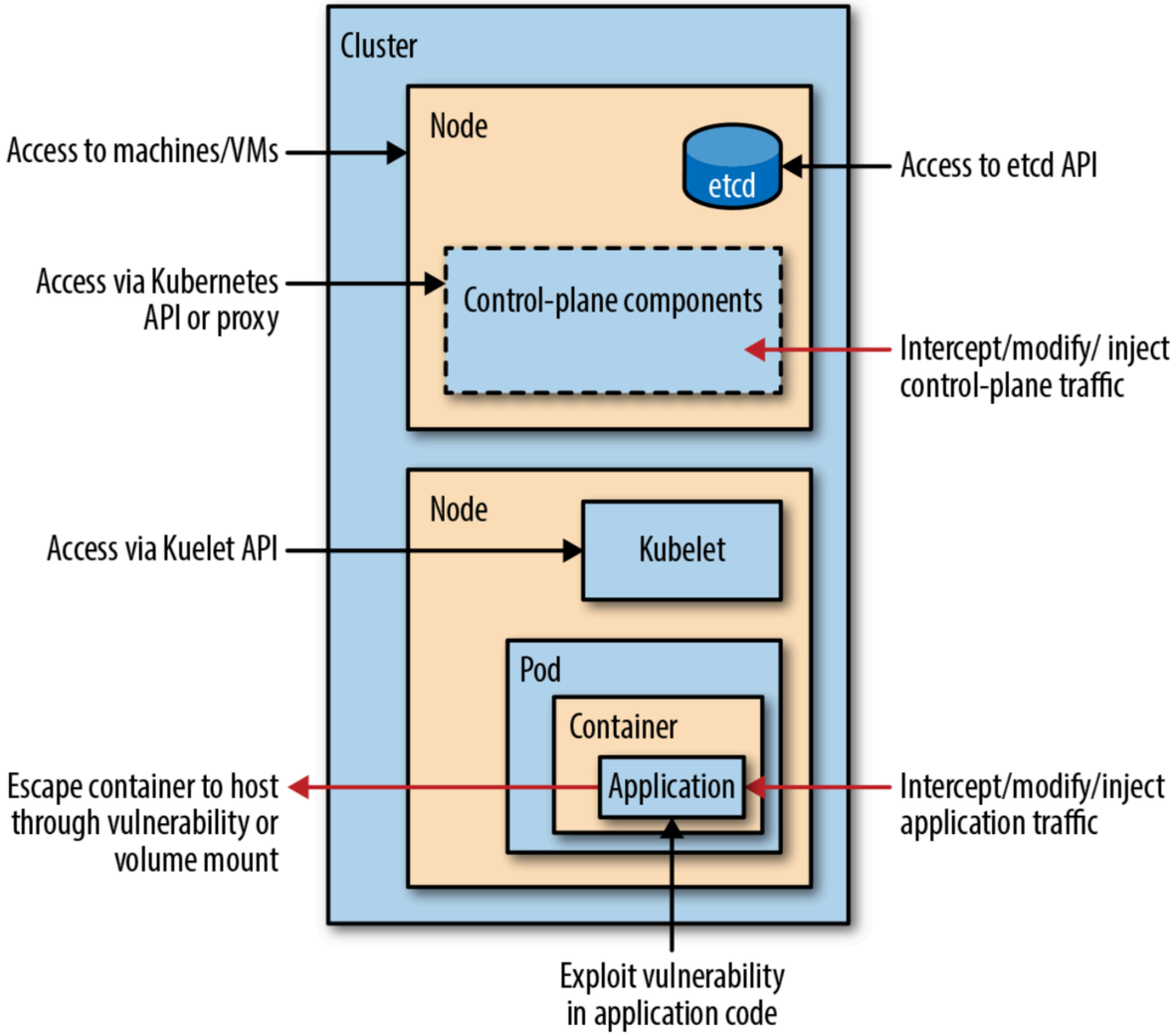
Kubernetes

- Container lifecycle management
- Declarative API + control loops
- Robust, flexible, scalable
- Extensible



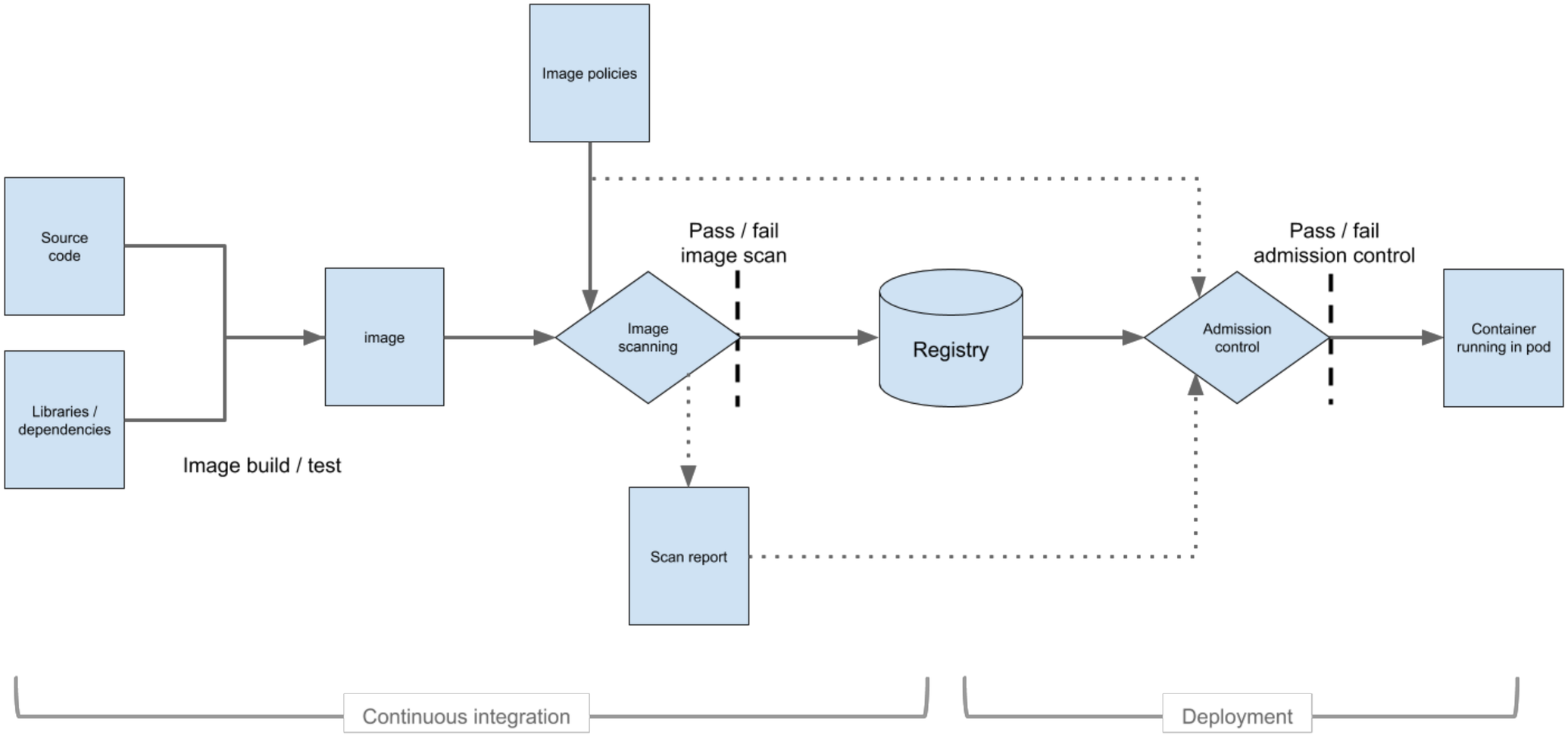
redhat.

Attack vectors



redhat.

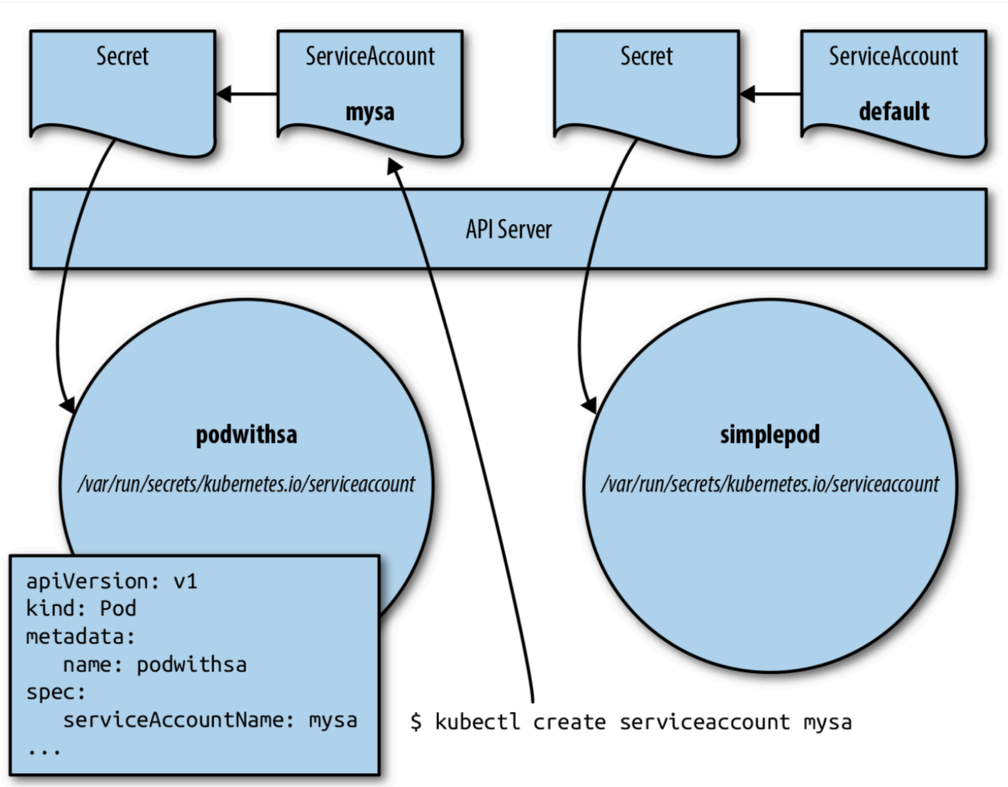
Continuous Integration (CI) & Continuous Deployment (CD)



redhat.

Service accounts

- Provide identity for an app
- Credentials via secret mounted into pod
- Default service account per namespace
- Basis for permissions (access control)

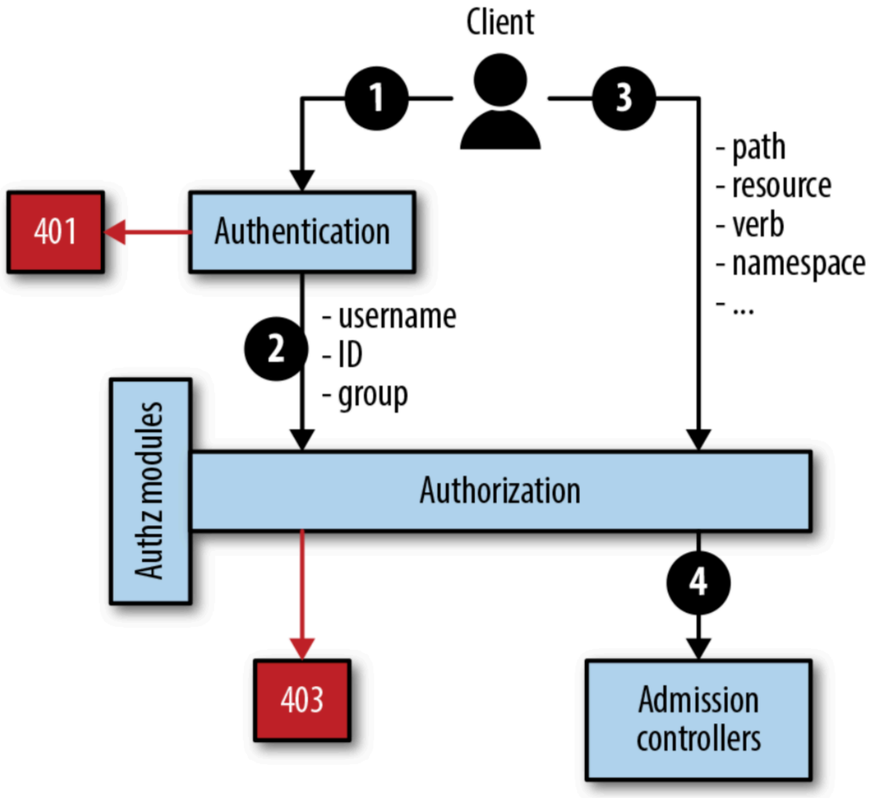


`system:serviceaccount:NAMESPACE:SERVICEACCOUNTNAME`

redhat.

Authentication & authorization

- static password/token file
- X509 client certs
- proxy+header
- OpenID Connect
- custom via Webhook



- Node (kubelet)
- ABAC (outdated)
- RBAC
- Webhook (external)

redhat.

Secrets

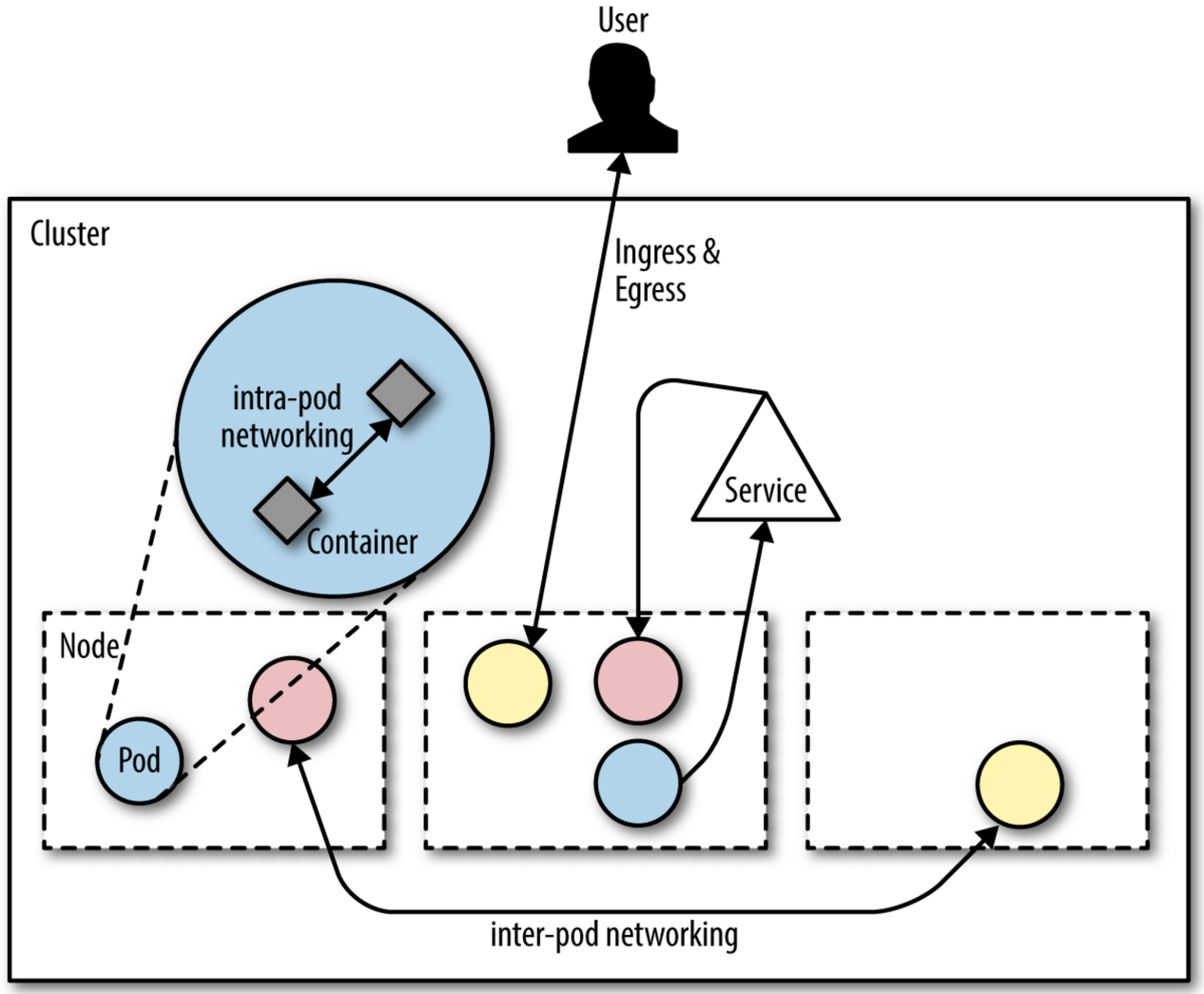
- Namespaced objects for sensitive data
- Access via volume or environment variable
- Data is stored in tmpfs volumes
- Only base64 encoded, need to enable encryption at rest

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```


redhat.

Networking

- East-west traffic (services, network policies)
- North-south traffic (ingress, API gateways, load-balancer)
- mTLS
- service meshes (Envoy, Istio, Linkerd 2, etc.)



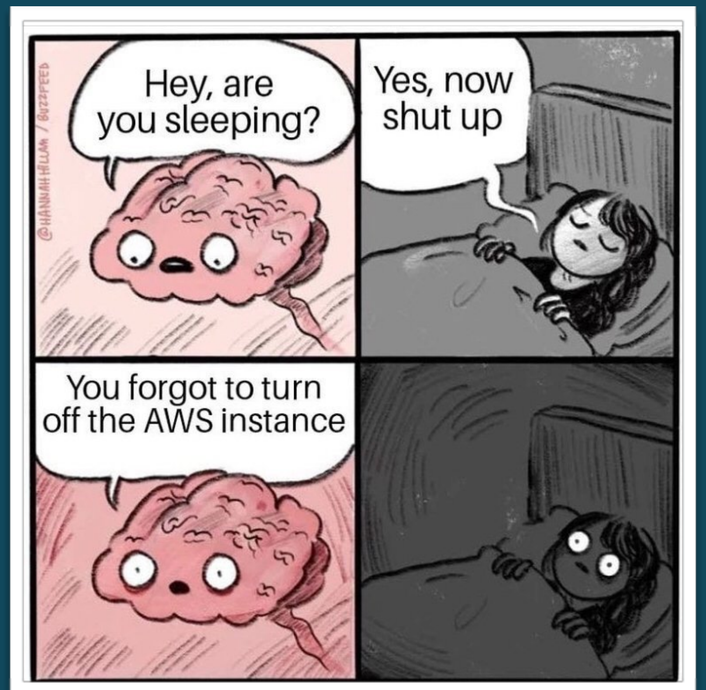
redhat.

Good practices

- use trusted base images & define (non-root) user
- perform automated CVE scans
- use private registries
- use namespaces
- use service account per app
- use RBAC
- define network policies

redhat.

Serverless



redhat.

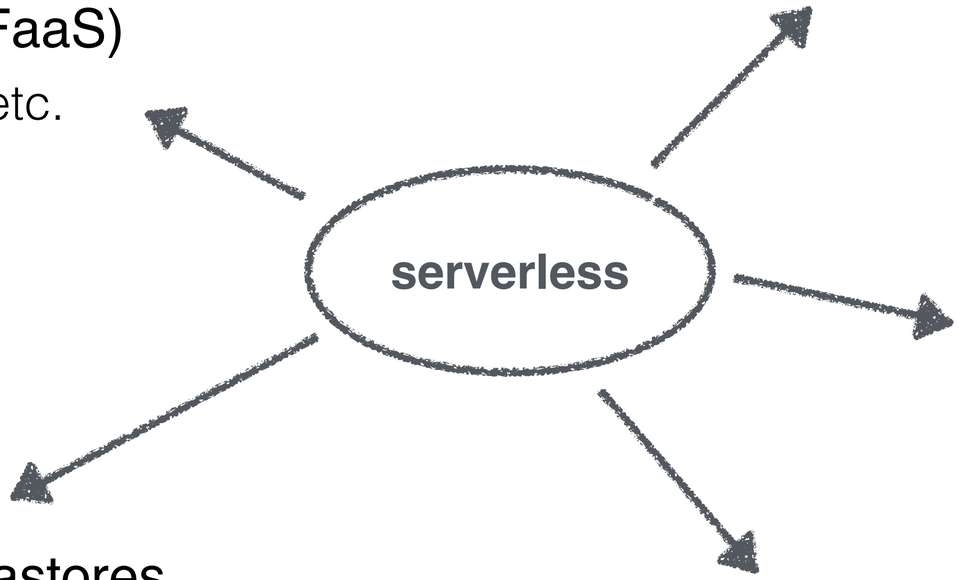
Some terminology …

query

BigQuery, Amazon Athena, etc.

Function-as-a-Service (FaaS)

AWS Lambda, knative, etc.



message queues

SQS, Azure Service Bus, etc

databases & datastores

Redshift, Azure Data Lake, etc.

object storage

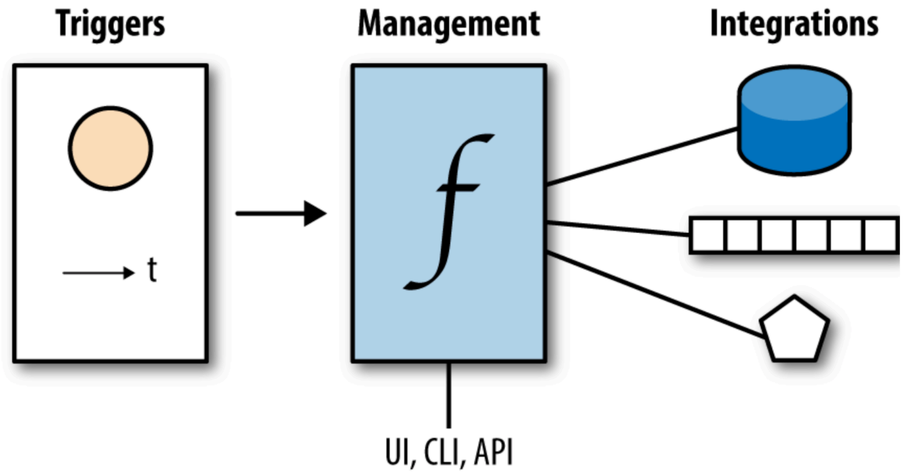
S3, Google Cloud Storage, etc.

boringis.cool/#lets-talk-about-serverless

redhat.

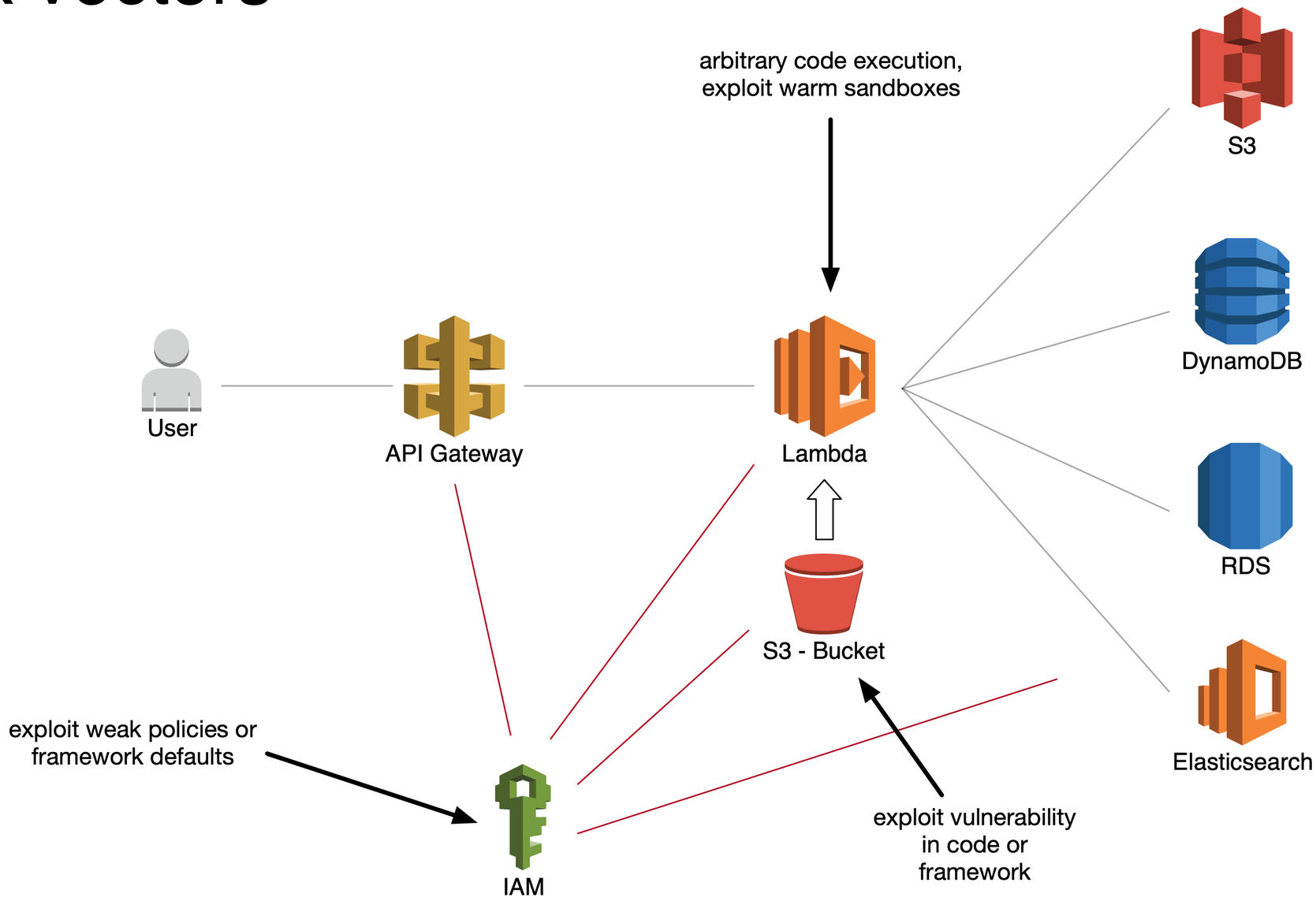
Function-as-a-Service concept

- event-driven (i.e. needs trigger)
- short-running (practically minutes)
- stateless (externalize state/integrations)
- cold start characteristics very important



redhat.

Attack vectors



redhat.

Good practices

- static code analysis
- dependencies vulnerability scans
- input validation
- secrets handling
- use strict IAM roles and policies
- auditing

redhat.

Resources

redhat.

- Many-faced threats to Serverless security
- Hacking Serverless Runtimes: Profiling AWS Lambda Azure Functions & More
- Security Best Practices for Serverless Applications
- Just say no to root (in containers)
- Securing Kubernetes Cluster Networking
- Service Mesh Network Security
- Hacking and Hardening Kubernetes Clusters by Example
- Dynamic secrets on Kubernetes pods using Vault
- kubernetes-security.info

redhat.

Vendors

redhat.