Kubernetes in Docker

Alex Mavrogiannis

Docker EE Engineering

Agenda

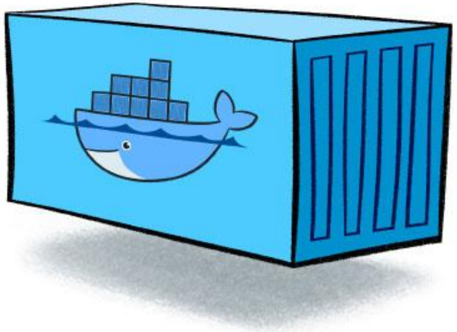
1. Introduction
2. Demo: Kubernetes in Docker EE 2.0
3. General CE/EE Architectures
4. EE: Topics on mixed workloads
5. EE: AuthN/AuthZ
6. Q&A

Introduction

What are Docker containers?

Processes running on the same host OS using the following mechanisms:

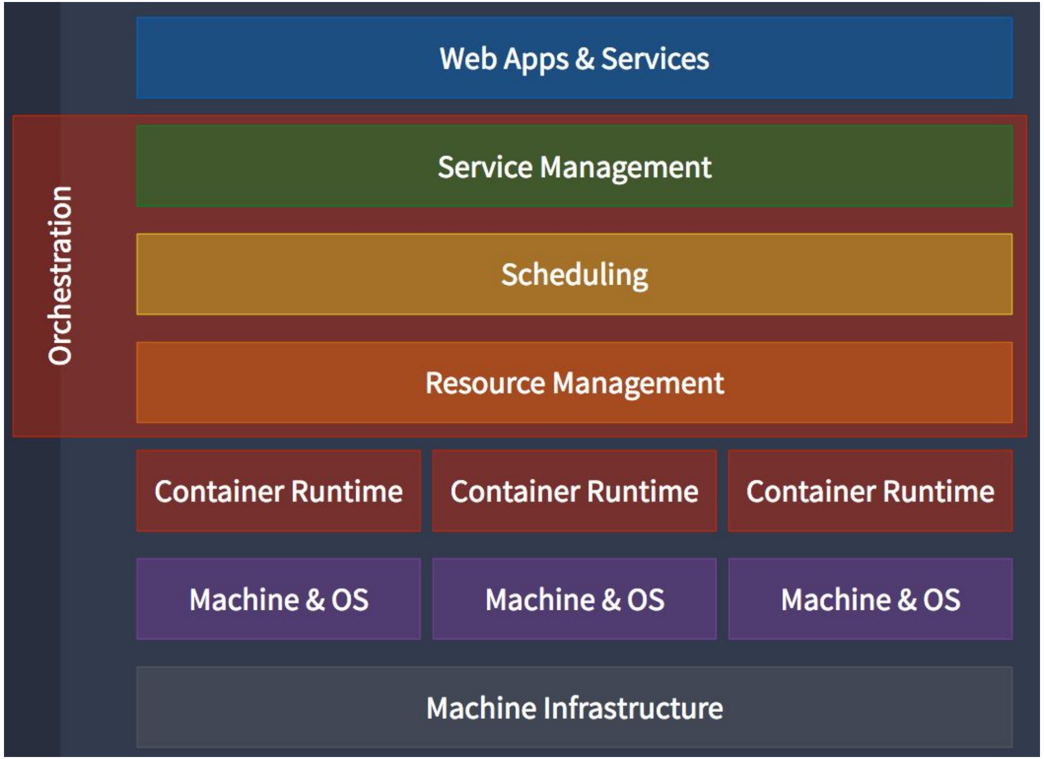
- IPC Namespaces
- PID Namespaces
- Network Namespaces
- Control Groups (Memory/CPU)
- Union Filesystems and image distribution mechanisms



Containers are managed by the Container Runtime process running on the host OS, the Docker Engine.

What is a container orchestrator?

Management of containers running in one or more container runtimes



Orchestrator: Docker Swarm

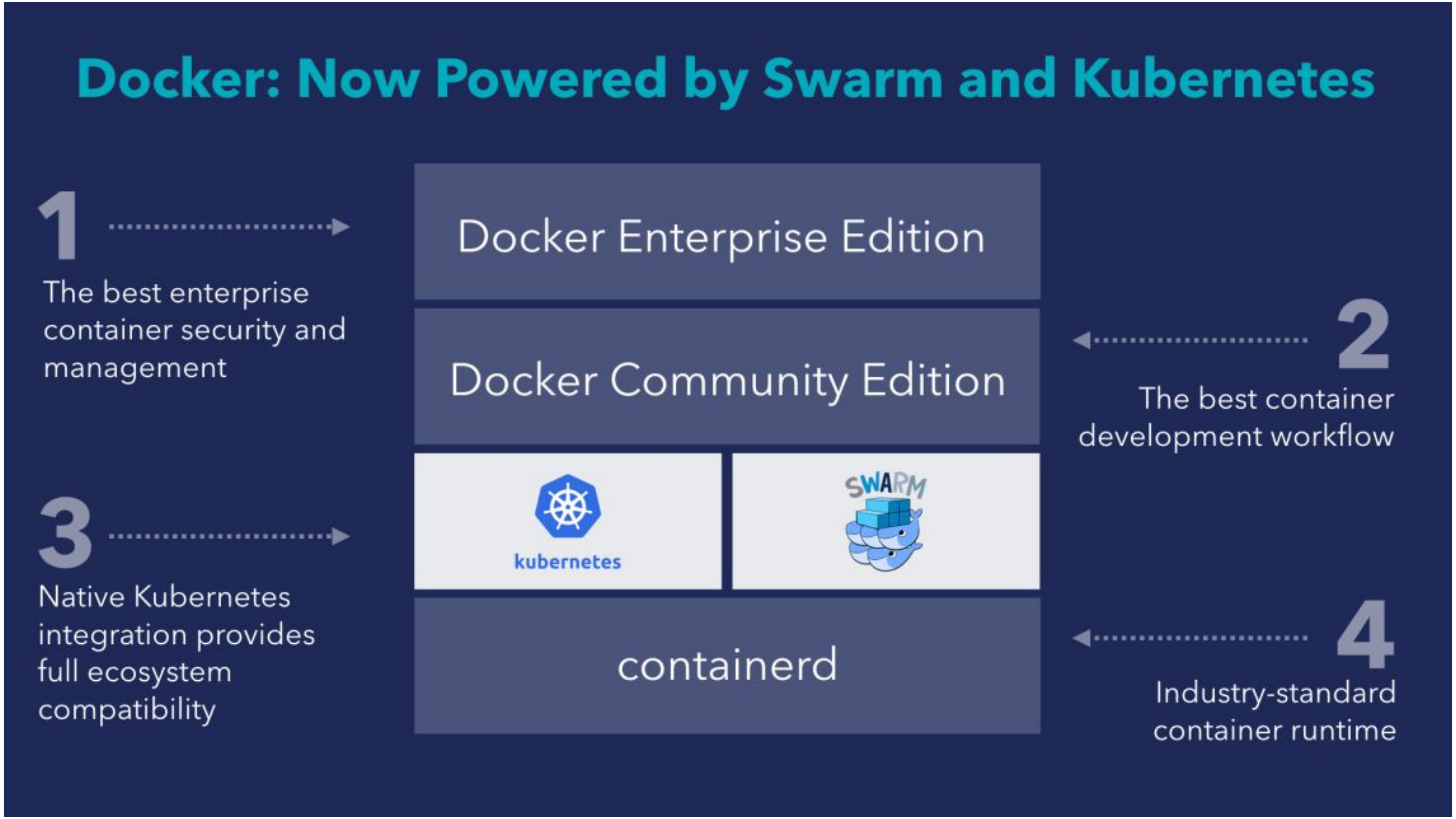
- github.com/docker/swarm
- Cluster-wide imperative API based on the single-node API of the Docker Engine
- High Availability and peer discovery managed through a pluggable discovery backend: etcd, consul
- “Leader” caches entire cluster state: containers, volumes, networks etc.

Orchestrator: Docker Engine with Swarm-Mode Enabled

- github.com/docker/swarmkit
- Declarative State through the “Service” construct
- Built-in Routing Mesh & Overlay networking
- In-memory Raft Store for all state (persisted to disk)
- Built-in CA, per-node cryptographic node identity, mTLS between all endpoints

Orchestrator: Kubernetes

- github.com/kubernetes/kubernetes
- Scheduling Unit: Pods
- Declarative State through “Controllers”: Deployment, ReplicaSet, DaemonSet …
- Flat Networking model delegated to plugins



Docker: Now Powered by Swarm and Kubernetes

1 The best enterprise container security and management

2 The best container development workflow

3 Native Kubernetes integration provides full ecosystem compatibility

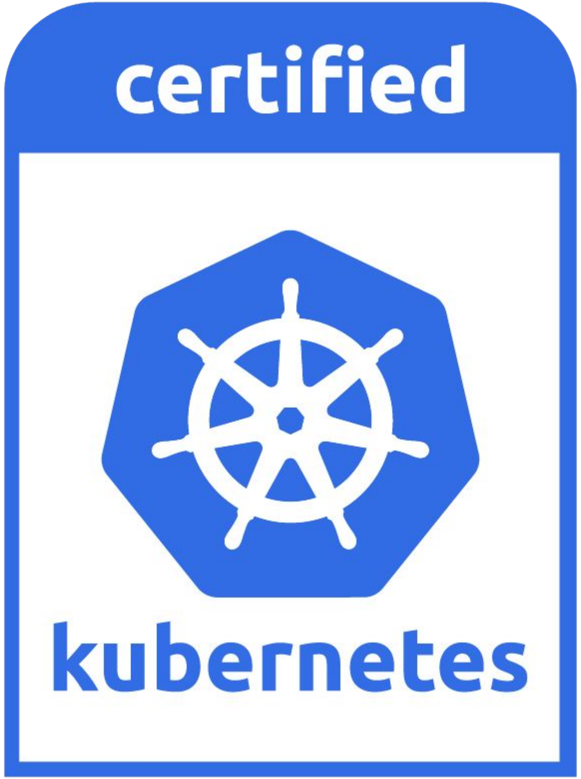
4 Industry-standard container runtime

Docker Enterprise Edition

Docker Community Edition

containerd

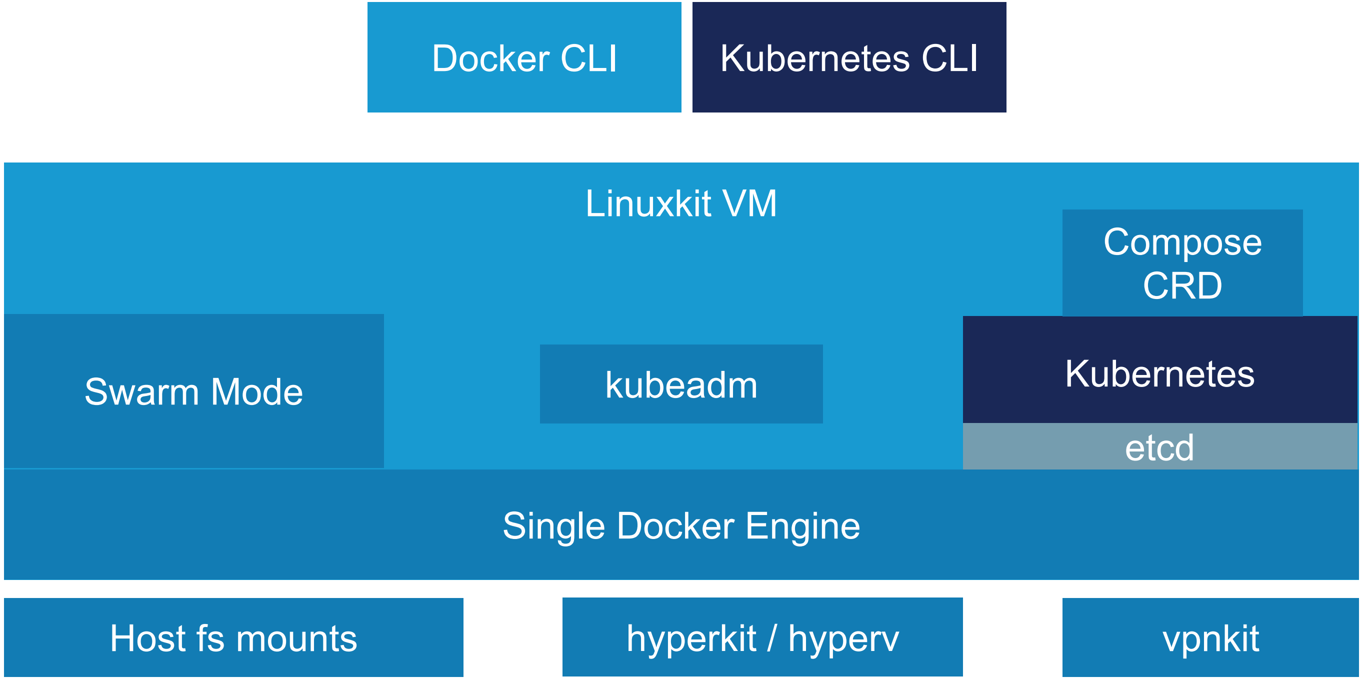
Docker EE 2.0: A conformant kubernetes distribution



Demo: Kubernetes in Docker EE 2.0

General CE/EE Architecture

Kubernetes in Docker CE (Windows and Mac)



Docker CLI

Kubernetes CLI

Linuxkit VM

Compose CRD

Swarm Mode

kubeadm

Kubernetes

etcd

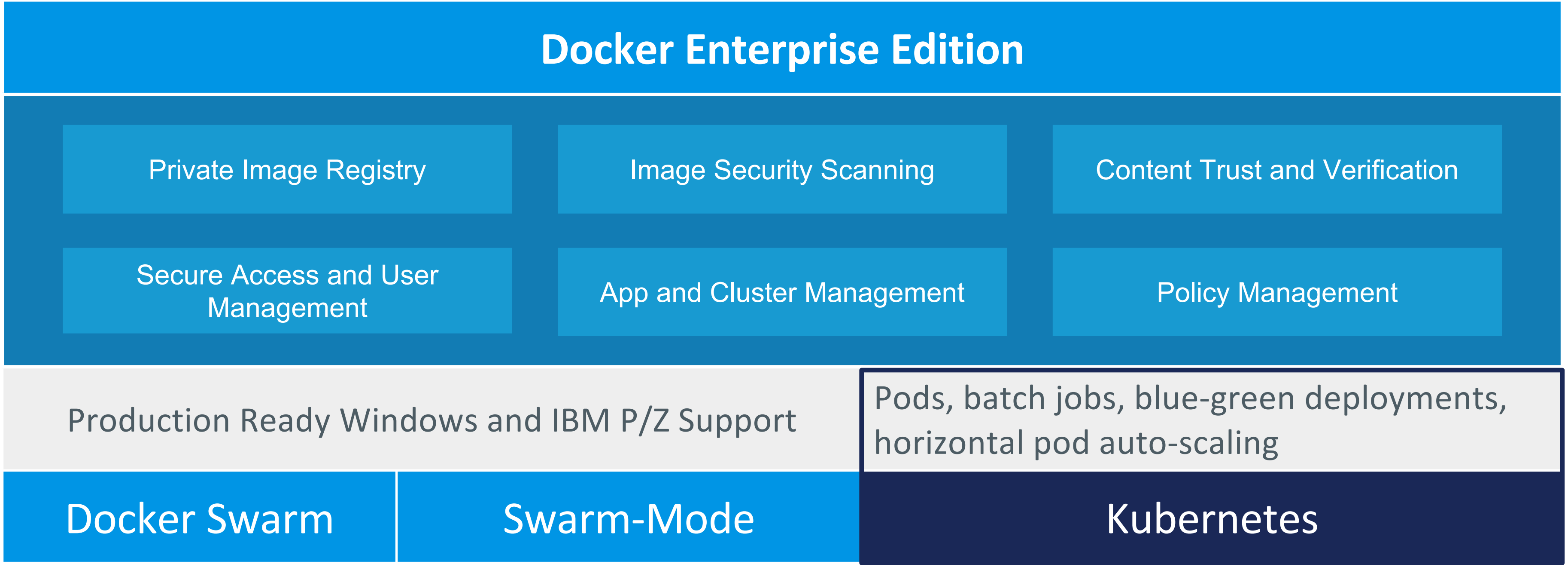
Single Docker Engine

Host fs mounts

hyperkit / hyperv

vpnkit

Docker EE to include Kubernetes



Docker Enterprise Edition

Private Image Registry

Image Security Scanning

Content Trust and Verification

Secure Access and User Management

App and Cluster Management

Policy Management

Production Ready Windows and IBM P/Z Support

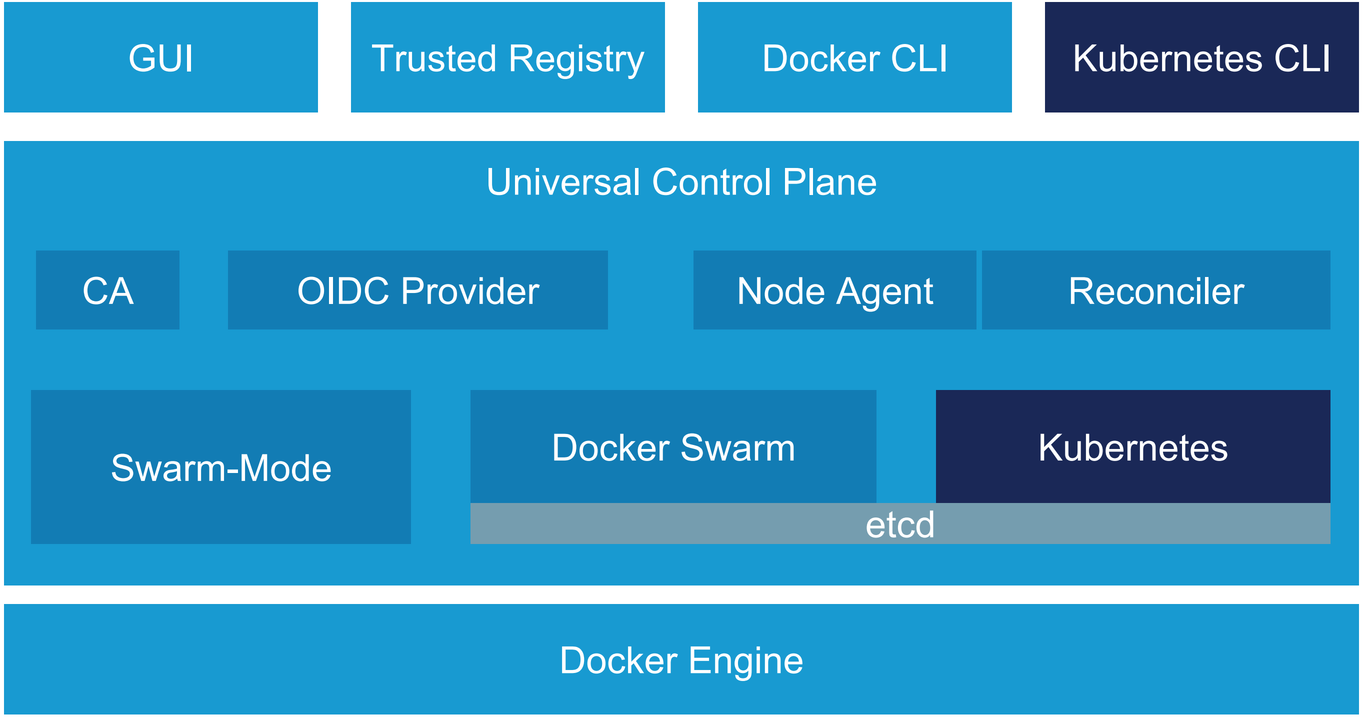
Pods, batch jobs, blue-green deployments, horizontal pod auto-scaling

Docker Swarm

Swarm-Mode

Kubernetes

Kubernetes in Docker EE



GUI

Trusted Registry

Docker CLI

Kubernetes CLI

Universal Control Plane

CA

OIDC Provider

Node Agent

Reconciler

Swarm-Mode

Docker Swarm

Kubernetes

etcd

Docker Engine

Docker EE Architectural Highlights

- Conformant Kubernetes components ran as Docker containers
- Swarm Managers are Kubernetes Masters
- Swarmkit node inventory is source of truth
- Cryptographic Node Identity and mTLS used throughout

Kubernetes Plugin Interfaces in Docker EE

- General:
  - Native API extensibility supported
  - Some apiserver/kubelet flags modifiable by users
- Networking:
  - Support for CNI plugin during install
  - Ingress
- Storage: Docker Volume Plugins supported via built-in flexvolume driver, CSI in future
- Metrics: Heapster Storage Backends or Prometheus

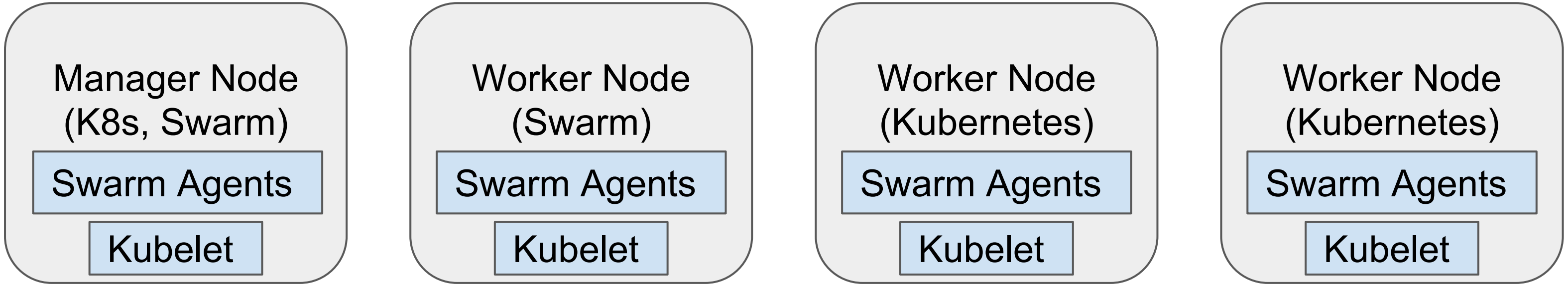
Topics on Mixed Workloads

Resource Contention

- Allocatable Resources: The set of CPU and Memory resources available for scheduling by an orchestrator on a single node
- Multiple orchestrators = Different definitions of allocatable resources
    - Docker Swarm: Respectful of CPU/Memory limits, but container cache may be stale
    - Docker Engine with Swarm-Mode: Only aware of its own reservations
    - Kubernetes: Effective handling of out-of-resource situations, but only for kubernetes workloads
- When a node is at/near capacity:
    - All CPU shares throttled equally
    - The OS’s OOM killer kills processes
    - All orchestrators will reschedule on OOM, but potential workload interruption

Orchestrator Selection

- Each node is running both kubernetes and swarm system components
- Administrators can toggle between (kubernetes, swarm or mixed) scheduling for any given node.
- When toggling orchestrators, workloads of the previous orchestrator will be evicted
- If a node is not enabled for a given orchestrator, users will not be able to schedule workloads on that node using that orchestrator.



Manager Node (K8s, Swarm)
Swarm Agents
Kubelet

Worker Node (Swarm)
Swarm Agents
Kubelet

Worker Node (Kubernetes)
Swarm Agents
Kubelet

Worker Node (Kubernetes)
Swarm Agents
Kubelet

Workload Interoperability

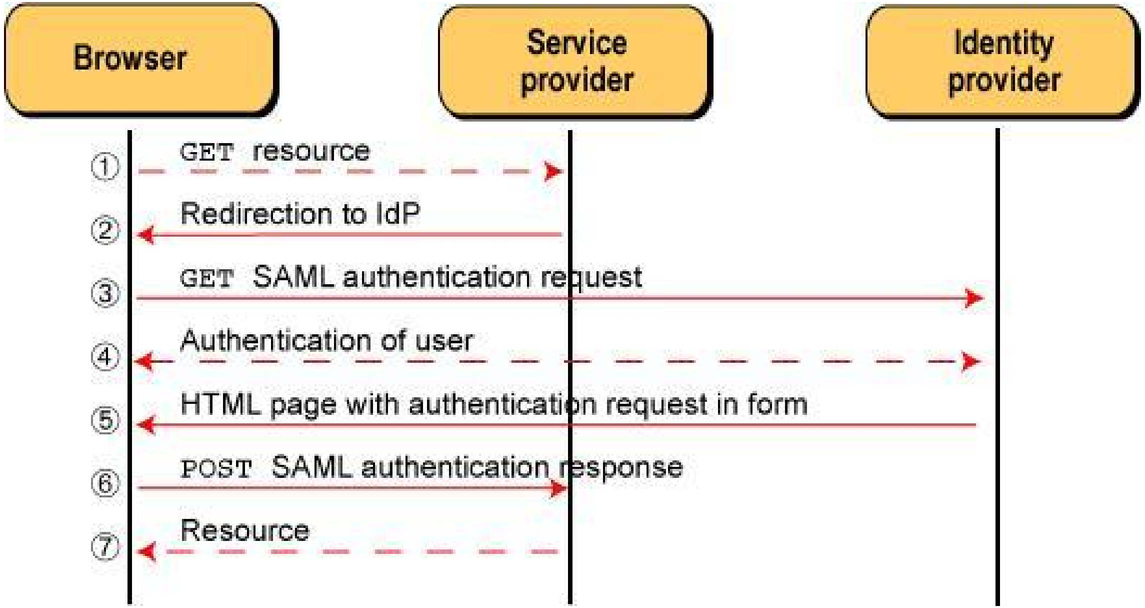
- Networking
  - Layer 3 not connected between kubernetes & swarm
  - Batteries-included kubernetes ingress controller
  - Layer 7 routing for swarm workloads
  - Configure external DNS
- Storage: Kubernetes workloads with docker volumes via flexvolume

AuthN / AuthZ

Identity Providers

Systems that manage identity information for principals and provides user authentication as a service.

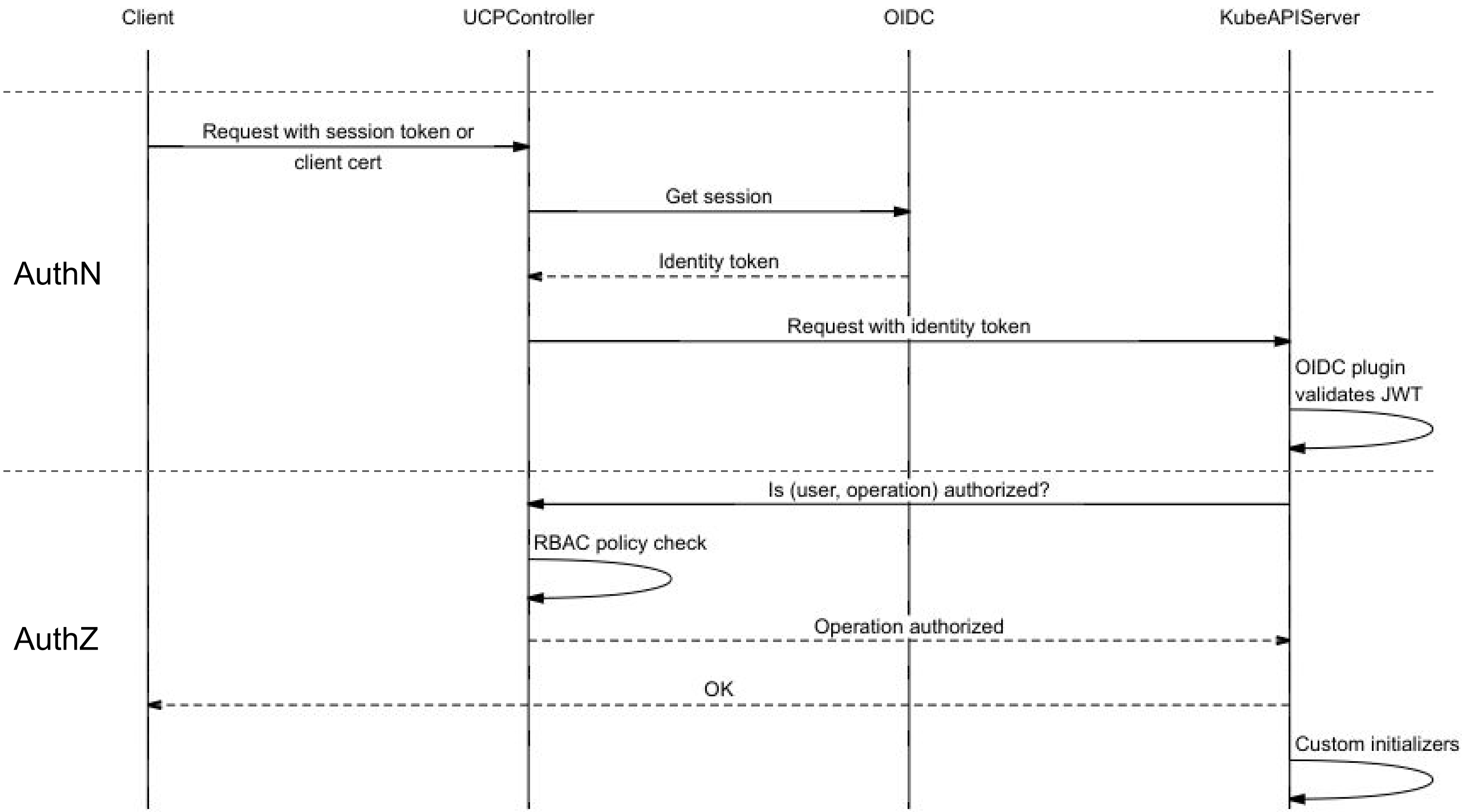
- SAML
- OpenID Connect (OIDC)



Browser
Service provider
Identity provider
① GET resource
② Redirection to IdP
③ GET SAML authentication request
④ Authentication of user
⑤ HTML page with authentication request in form
⑥ POST SAML authentication response
⑦ Resource

Actors in Docker EE Authentication/Authorization

- Client (Browser, Docker CLI or kubectl)
- UCP Controller
- OIDC Provider
- Kubernetes API server



Client

UCPController

OIDC

KubeAPIServer

AuthN

Request with session token or client cert

Get session

Identity token

Request with identity token

OIDC plugin validates JWT

AuthZ

Is (user, operation) authorized?

RBAC policy check

Operation authorized

OK

Custom initializers

In Summary...

- Docker EE and CE will include a conformant Kubernetes distribution.
- Resource Contention mitigated via orchestrator selection
- In EE, Authentication and Authorization integrated via standard plugin interfaces.

Thank You!

Sign up for the beta at docker.com/kubernetes

alexmavr

alex.mavrogiannis@docker.com

dockercon17 EU