# AWS Containers

## ECS, EKS, Fargate



aws

# Why Containers?

- App isolation
- Speed
- Efficiency
- Easier packaging
- Less risky deployments
- Better Development experience
- Microservices


Photo & Licence


© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Typical Use Cases

- Microservices: Java, Node.js, Go, Web Apps, etc.
- Continuous Integration and Continuous Deployment (CICD)
- Batch Processing and ETL jobs
- Common PaaS Stack for Application Deployment
- Legacy Application Migration to the Cloud
- Hybrid Workloads
- AI/ML
- Scale Testing
- Backend for IoT use cases



aws

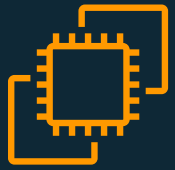
# AWS container services landscape

## Management
Deployment, Scheduling, Scaling & Management of containerized applications

**Amazon Elastic Container Service**

**Amazon Elastic Container Service for Kubernetes**

## Hosting
Where the containers run



**Amazon EC2**

**AWS Fargate**

## Image Registry
Container Image Repository

**Amazon Elastic Container Registry**

## Application Networking
Service Discovery, Service Mesh

**AWS Cloud Map**

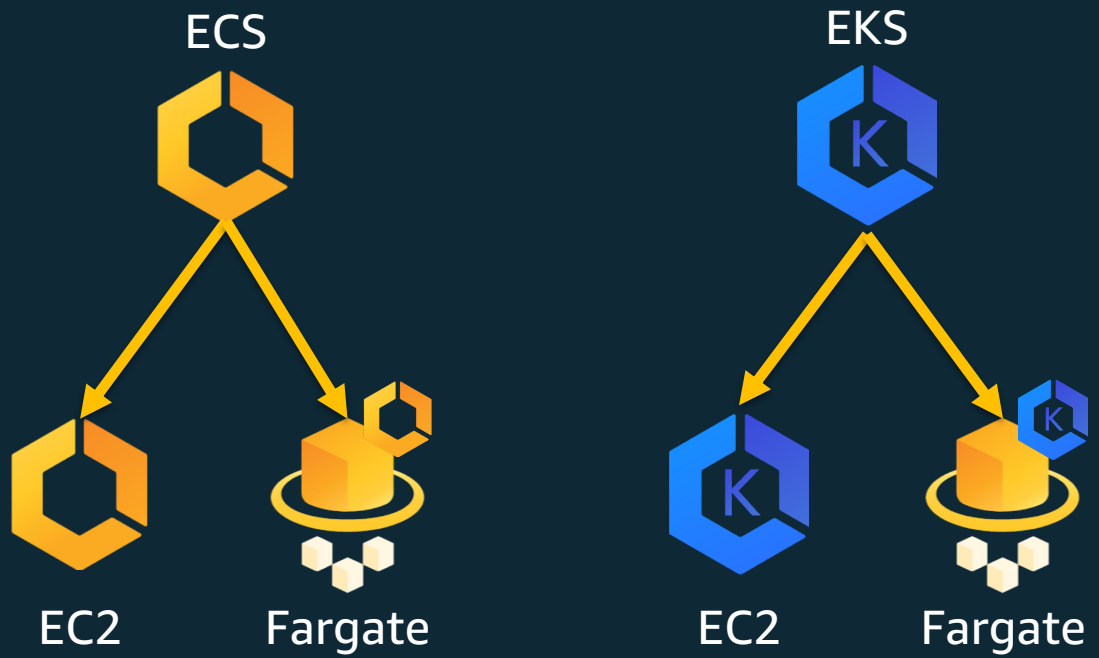
**AWS App Mesh**



aws

# We Give You The Power To Choose:

1. Choose your orchestration tool
2. Choose your launch type





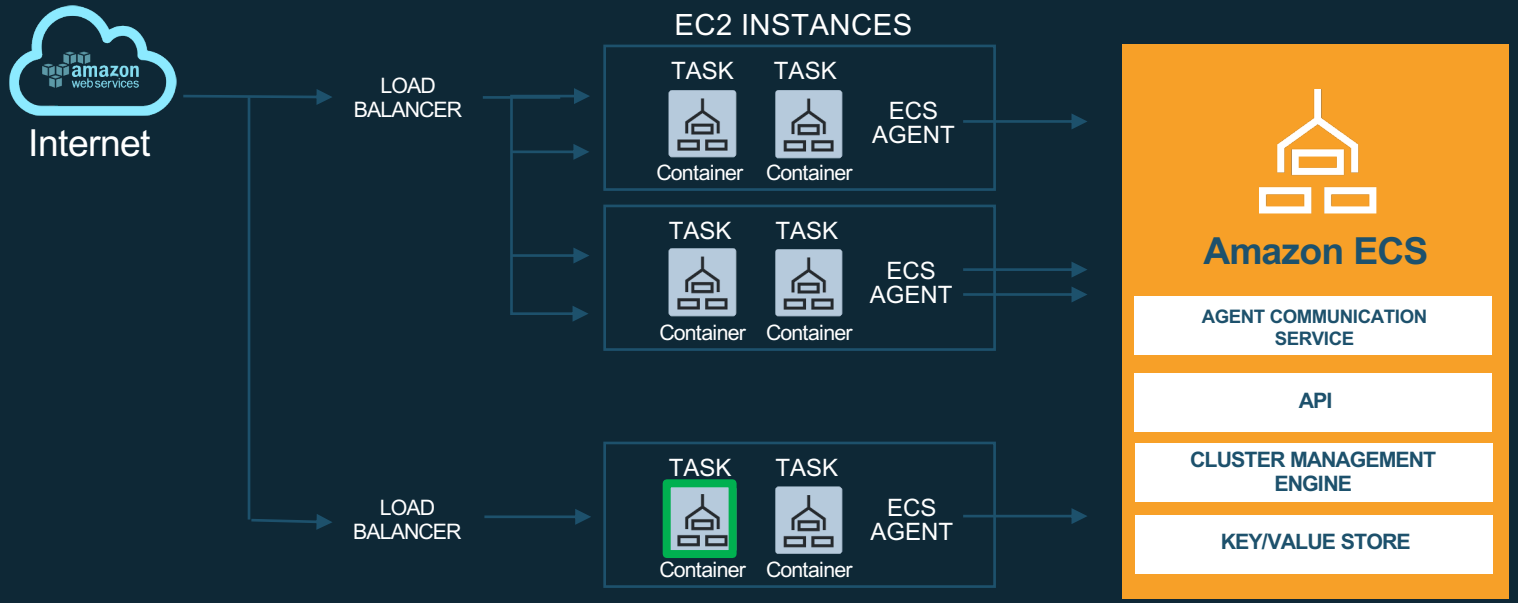

aws

# Amazon ECS

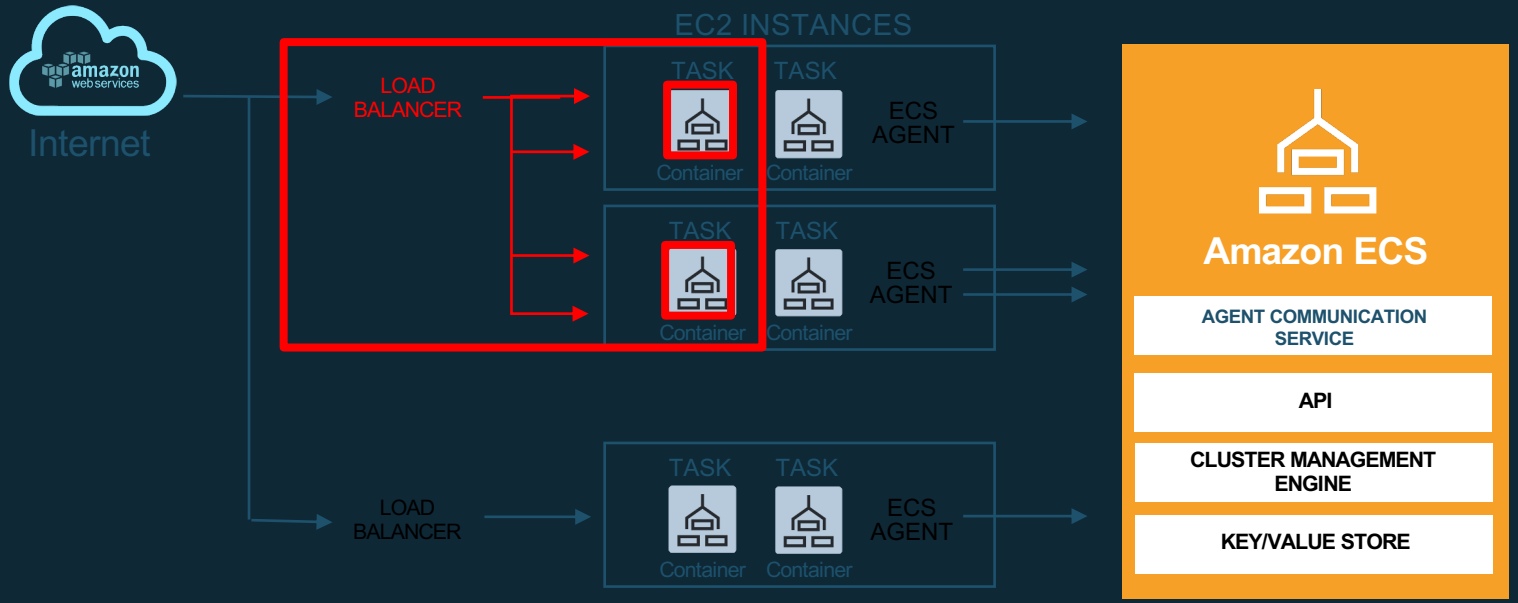


aws

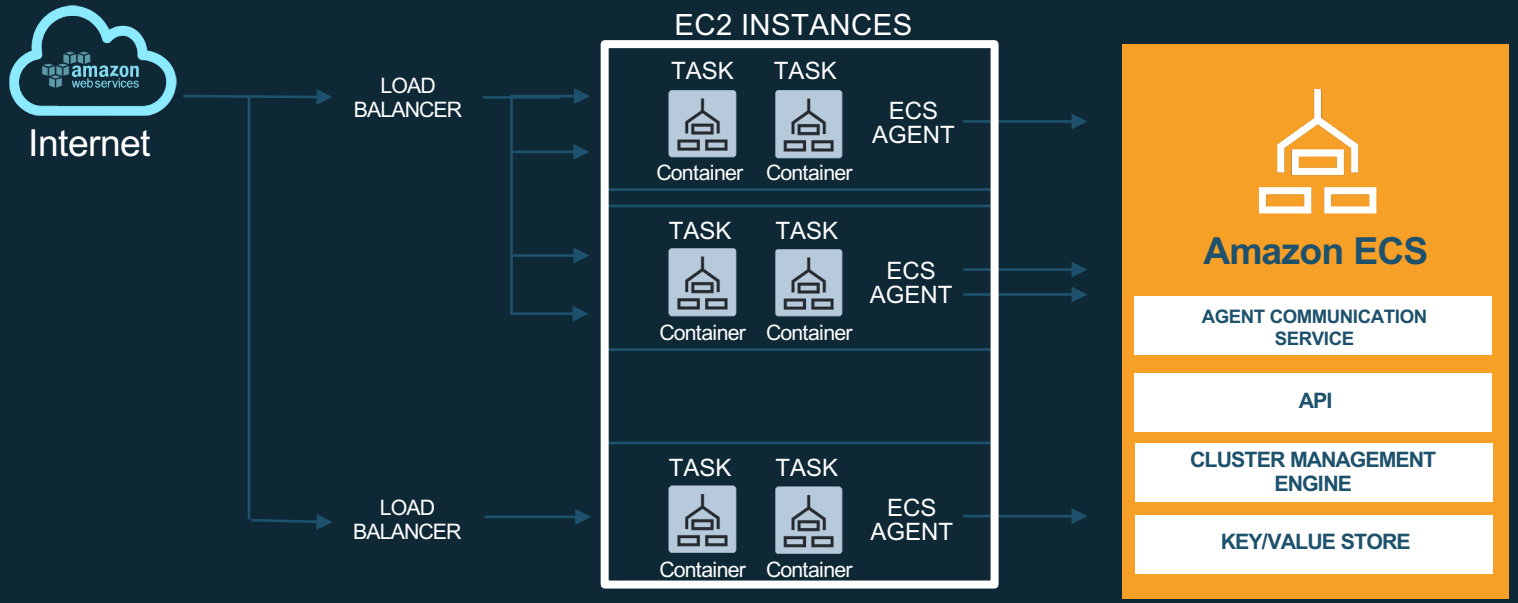
# Amazon ECS - Task

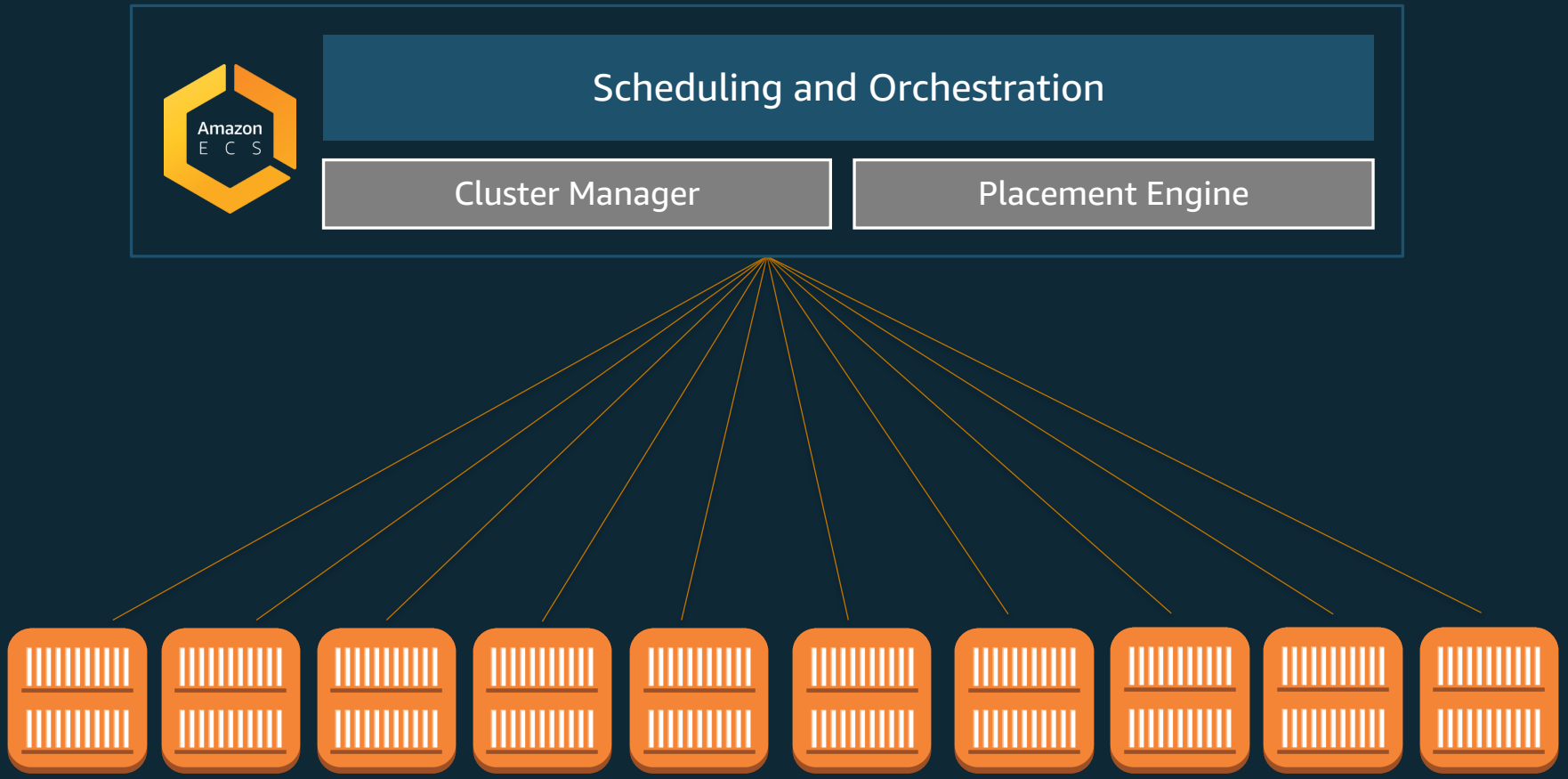


# Amazon ECS - Service



# Amazon ECS - Cluster

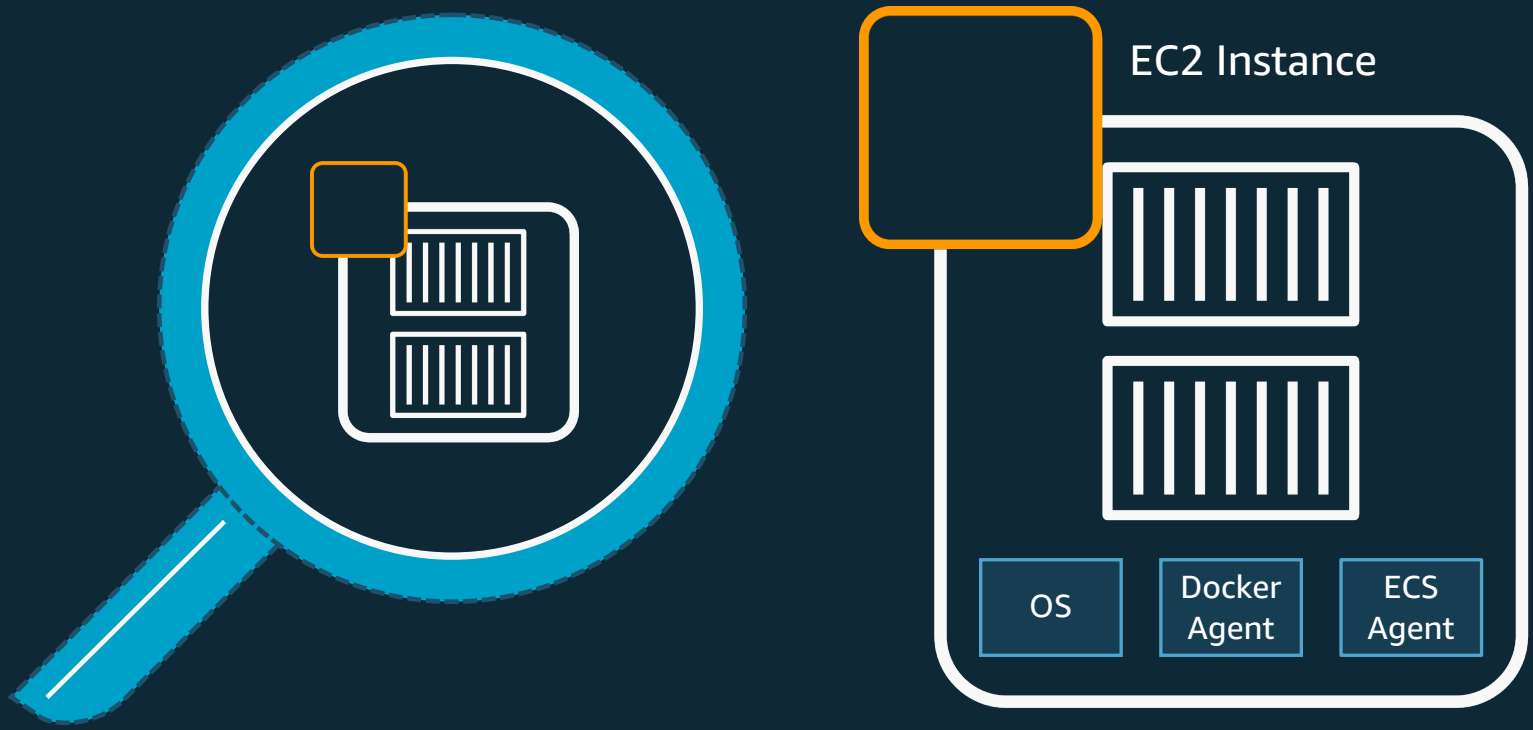







aws

# But you still end up managing more than just containers



- Patching and Upgrading OS, agents, etc.

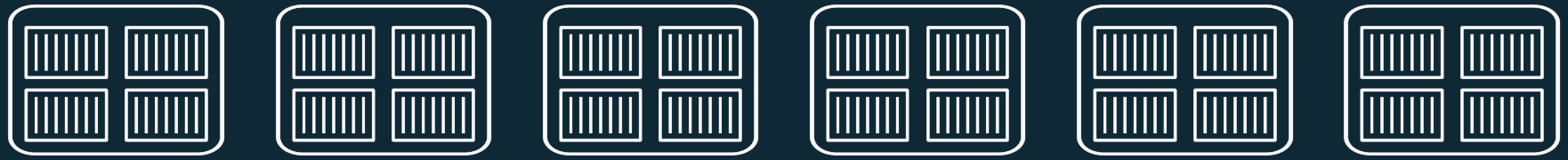
- Scaling the instance fleet for optimal utilization

# Amazon Elastic Container Service



aws

# Amazon Elastic Container Service



**AWS** Fargate

run serverless containers



aws

# AWS FARGATE

**Your Containerized Applications**

## MANAGED BY AWS

No EC2 Instances to provision, scale or manage

## ELASTIC

Scale up & down seamlessly. Pay only for what you use

## INTEGRATED

with the AWS ecosystem: VPC Networking,
Elastic Load Balancing, IAM Permissions, Cloudwatch and more.



aws

# RUNNING FARGATE CONTAINERS WITH ECS

Same *Task Definition* schema

Use ECS APIs to launch Fargate Containers

Easy migration – Run *Fargate* and *EC2* launch type tasks in the same cluster



aws

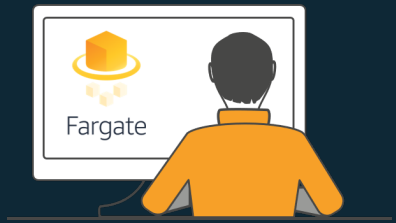
# TASK PROVISIONING MODEL

No instances
to manage

Container
native API

Resource
based pricing

=




Simple,
easy to use, powerful
consumption model



aws

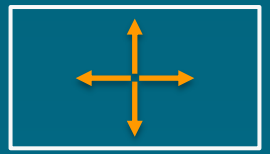
# Amazon EKS



aws

# What is Kubernetes?

Open source container management platform



Helps you run containers at scale

Gives you primitives for building modern applications


© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Kubernetes on AWS

3x Kubernetes masters for HA



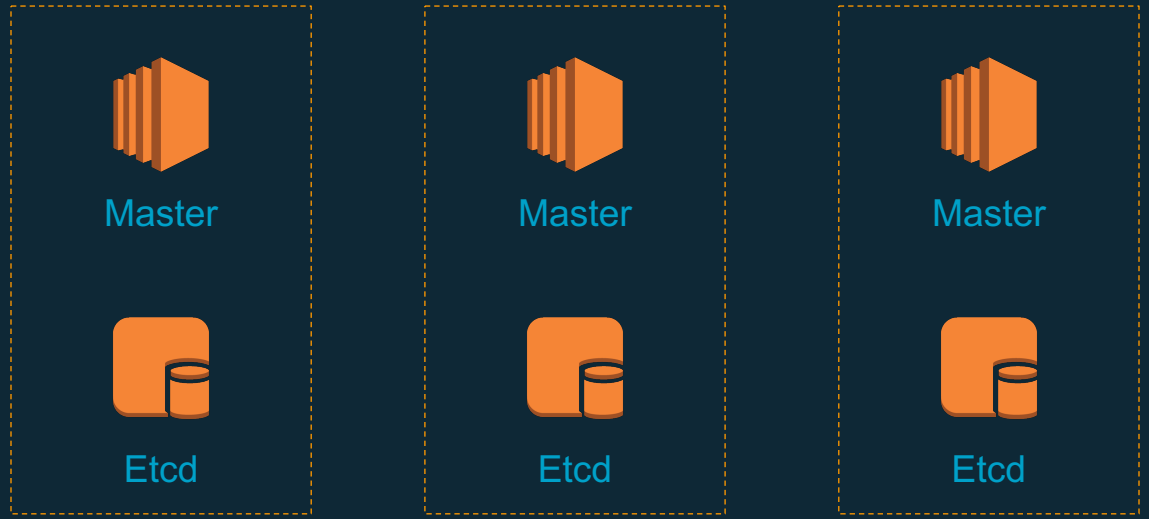
aws

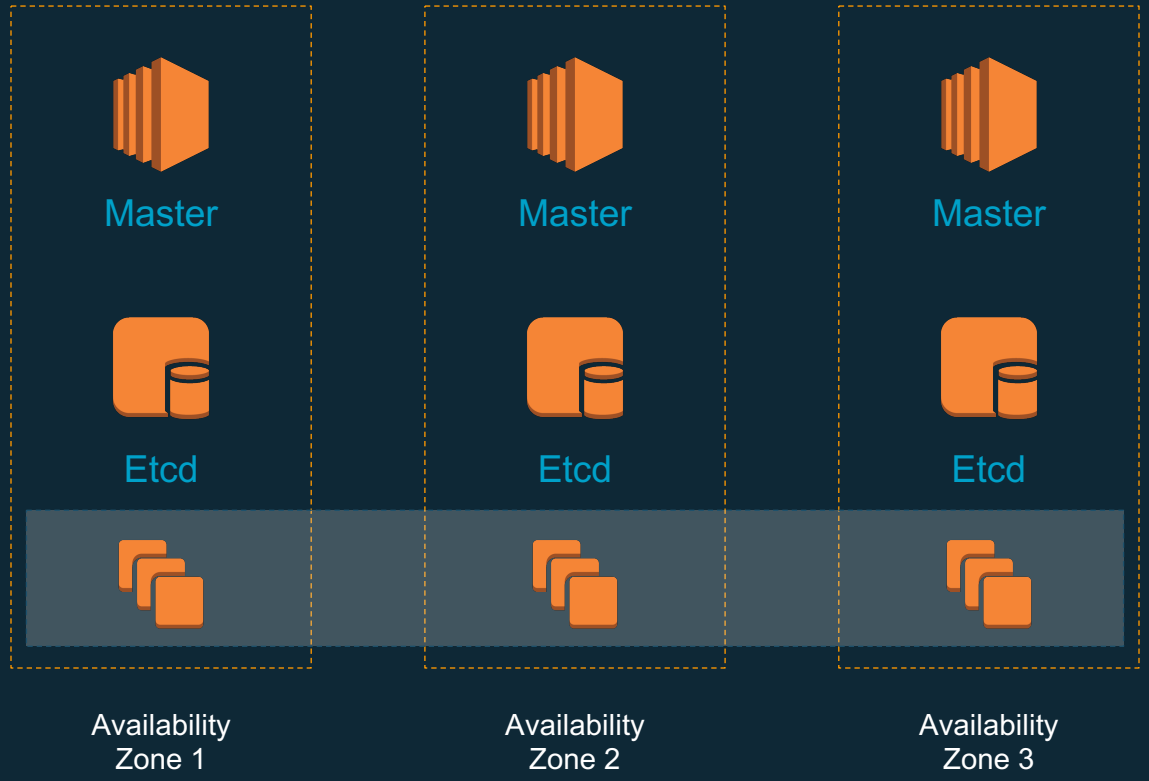
# Kubernetes master

| | | |
|---|---|---|
| API server | Controller manager | Cloud controller |
| Scheduler | KubeDNS | Add-ons |

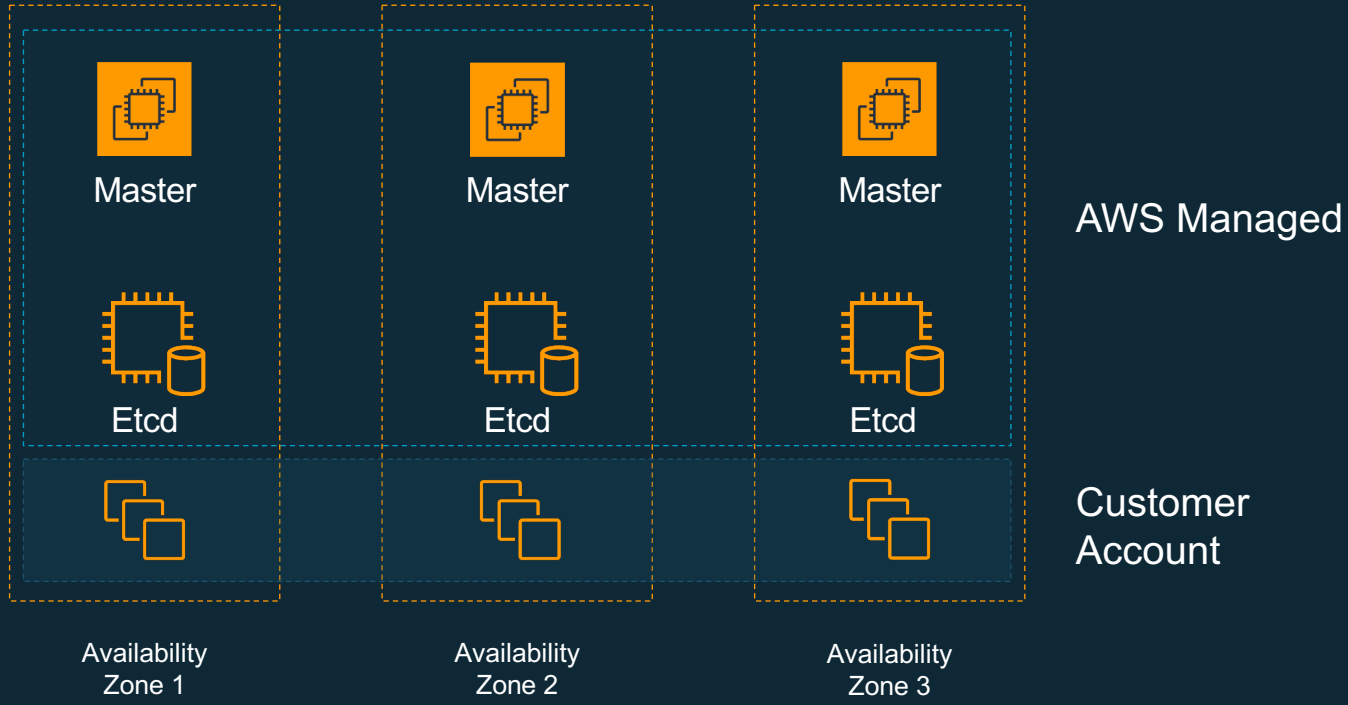


aws

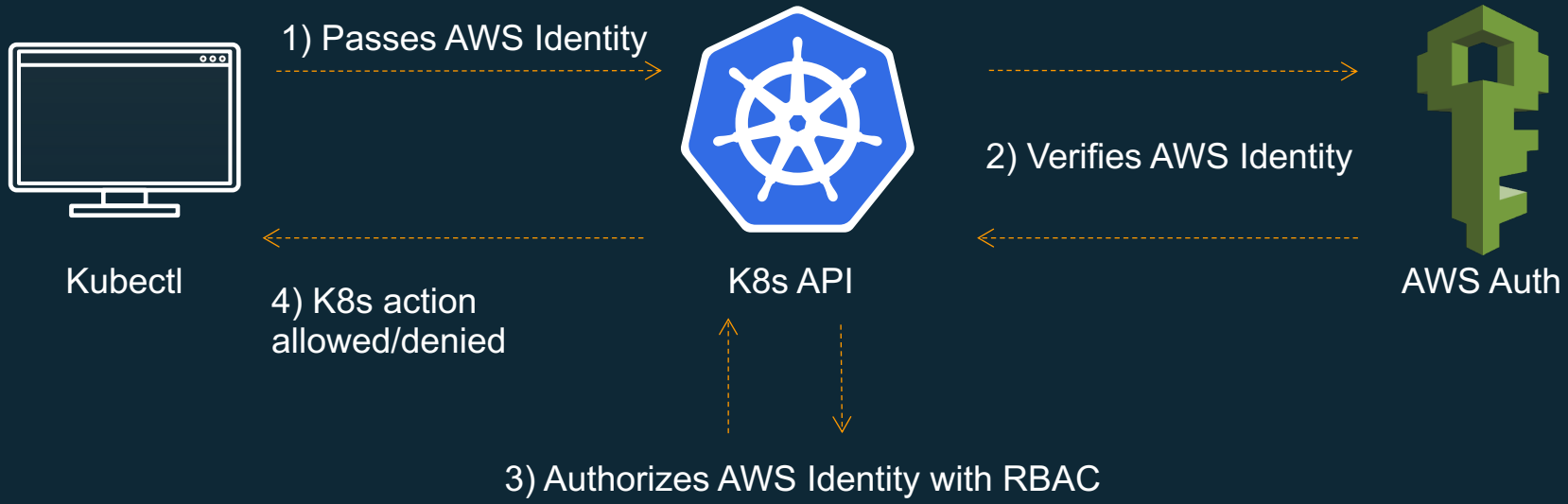




# Amazon EKS High Availability Control Plane



# IAM Authentication + Kubectl





Native VPC networking with CNI plugin

Pods have the same VPC address inside the pod as on the VPC

Simple, secure networking

Open source and on Github

https://github.com/aws/amazon-vpc-cni-k8s



# Amazon Elastic Container Registry (Amazon ECR)

- Deep integration with AWS platform
- Integrated with Amazon ECS and Docker CLI
- Scalable and highly available
- 100 percent cloud-based Docker container registry

# Service Mesh & Discovery

AWS App Mesh

- Service mesh for application level networking
- Observability & traffic control
- Works across clusters and container services
- AWS built and run

AWS **Cloud Map**

- Service discovery for all your cloud resources
- Increase developer productivity
- Integration with Amazon container services



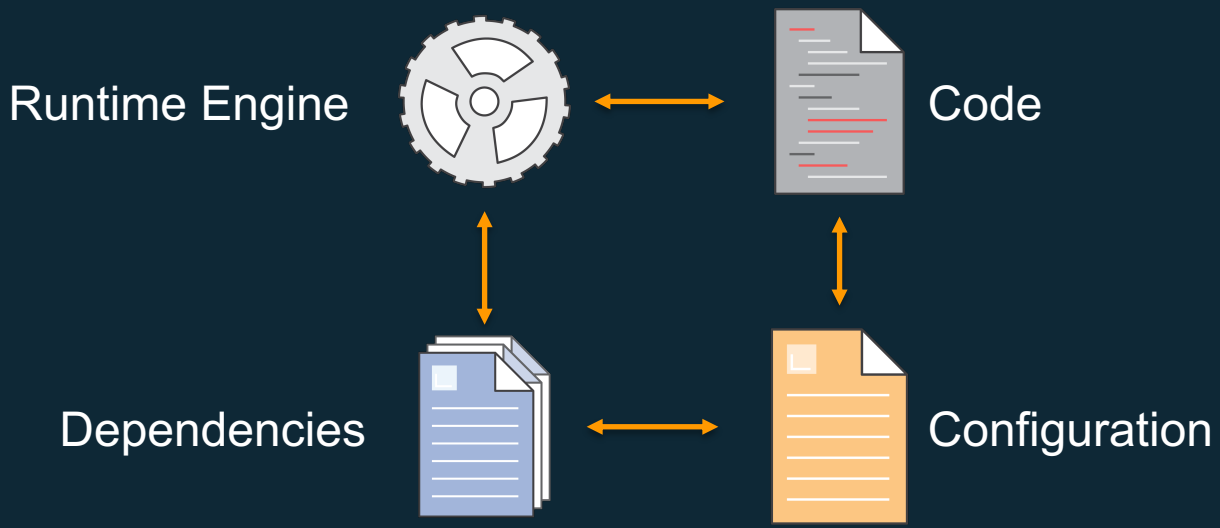
aws

# THANK YOU

https://aws.amazon.com/containers

# Introduction to Containers and Docker

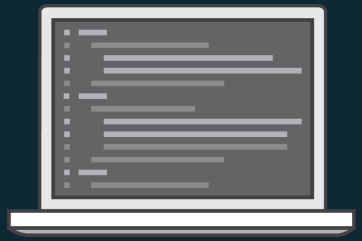
# Application environment components






aws

# Different environments



Local
Laptop

Staging /
QA

Production

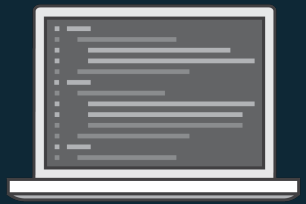
On-Prem



aws

# It worked on my machine, why not in prod?

v6.0.0



Local Laptop

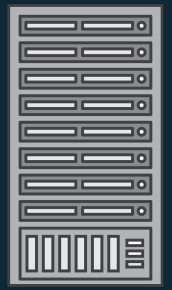
v7.0.0

Staging / QA

v4.0.0

Production

v7.0.0



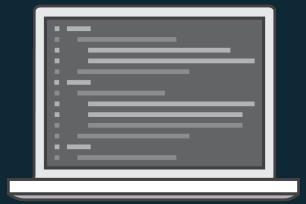
On-Prem



aws

# It worked on my machine, why not in prod?

v6.0.0



Local Laptop

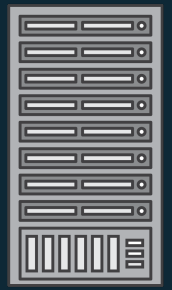
v7.0.0

Staging / QA

v4.0.0

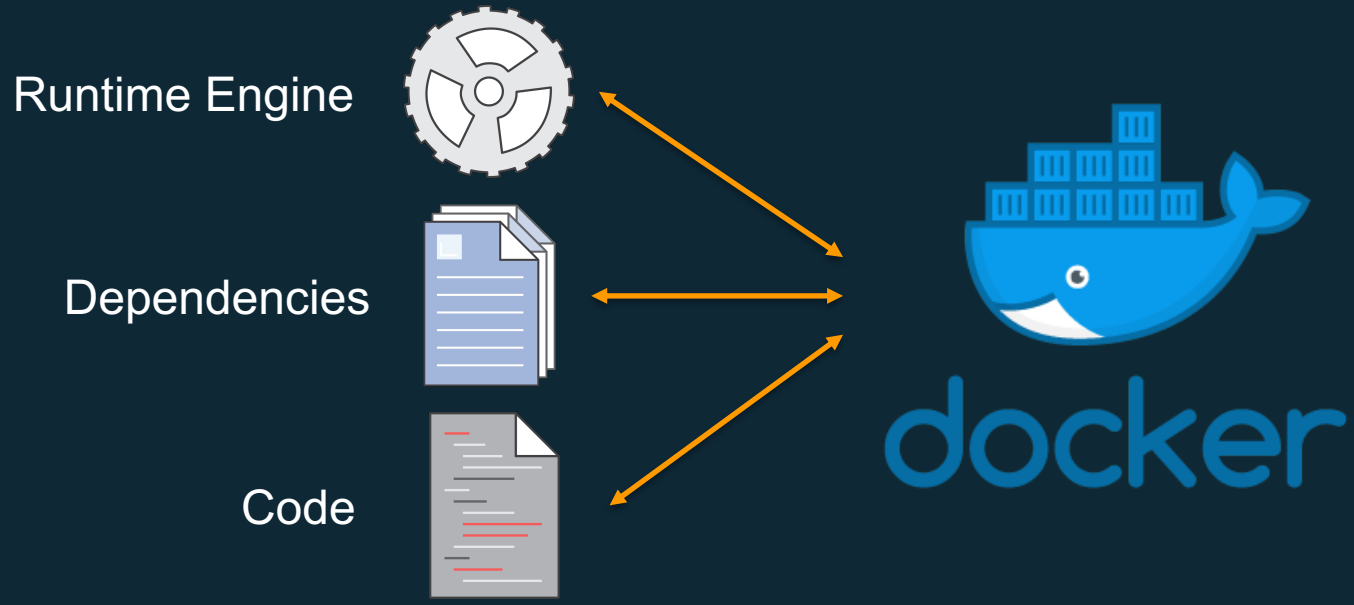
Production

v7.0.0



On-Prem

# Docker to the rescue





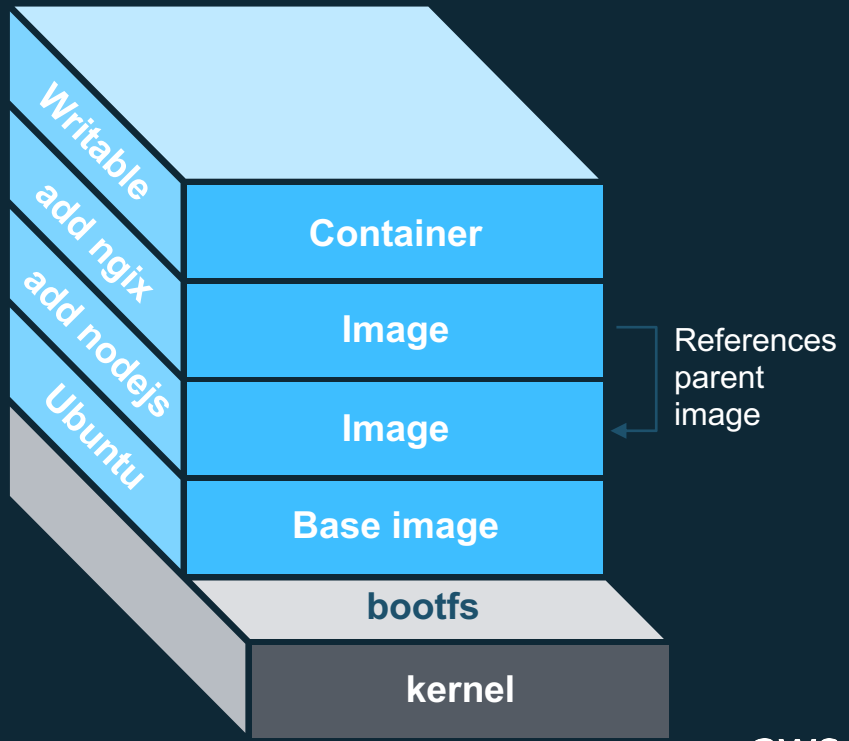

aws

# Docker container image

Read only image that is used as a template to launch a container.

Start from base images that have your dependencies, add your custom code.

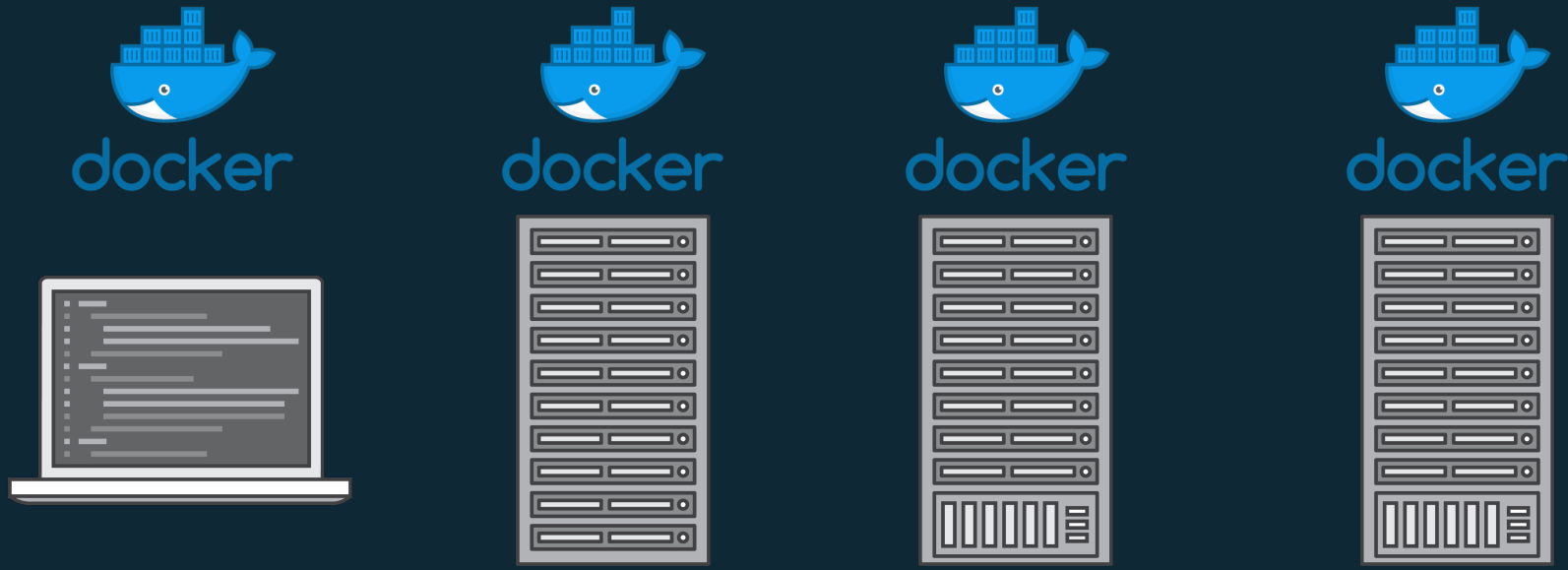
Docker file for easy, reproducable builds.






aws

# Four environments, same container




Local Laptop

Staging / QA

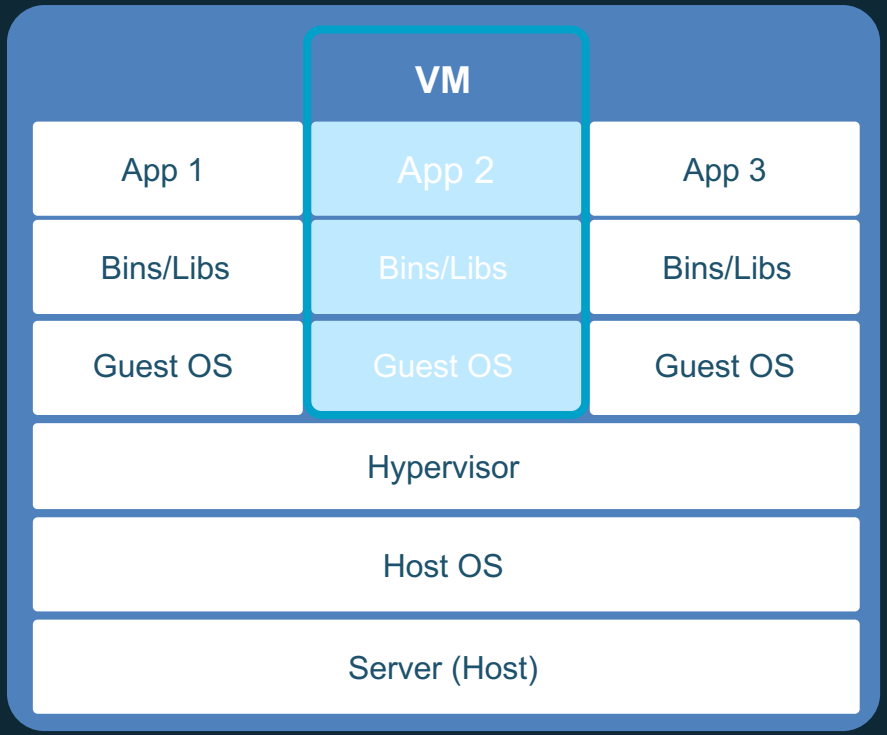
Production

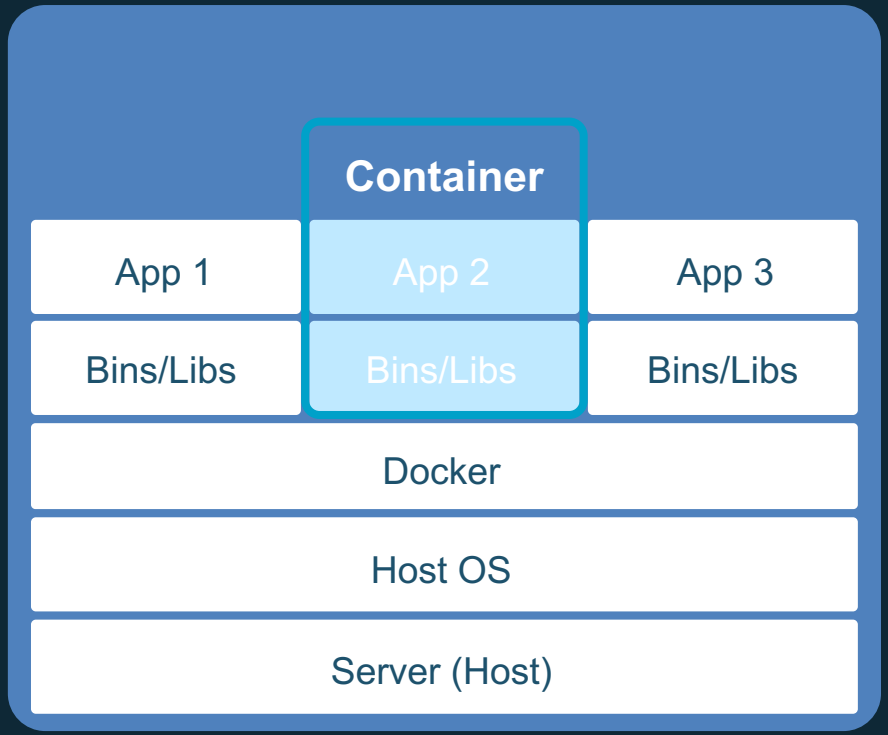
On-Prem



aws

# Virtual machine versus Docker






aws

# Container & Docker Benefits

Portable application artifact that runs reliably everywhere

Run different applications or application versions with different dependencies simultaneously

Better resource utilization by running multiple lightweight containers per host



aws

# Reference Architectures (Informational)

- Rebalancing Amazon ECS Tasks using AWS Lambda
- NGINX Reverse Proxy on Amazon EC2 Container Service
- Java Microservices Deployed on EC2 Container Service
- Amazon ECS Reference Architecture: Batch Processing
- Node.js Microservices Deployed on EC2 Container Service
- Amazon EC2 Container Service - Reference Architecture: Service Discovery  to containers using CloudWatch Events, Lambda and Route 53 private hosted zones
- Service Discovery for AWS EC2 Container Service via DNS
- Canary Blue/Green deployments on ECS
- Blue/Green deployments on ECS
- Blue/Green deployments using Fargate
- ECS Reference Architecture: Continuous Deployment
- Amazon ECS Scheduler Driver to integrate Apache Mesos with ECS



aws