Deploying Container-Based Microservices in AWS

A Note About This Class

This Class Covers

- Deep technical details
  - Docker Swarm on EC2
  - Amazon ECS
  - Amazon EKS
- CI/CD
  - CodePipeline
  - CodeBuild
  - Amazon ECR

This Class Does Not Cover

- Fundamentals
- General architecture
- Every service…

For More In-Depth Training

- AWS Fundamentals
- AWS Linux Operations
- Networking in AWS
- AWS Certified Cloud Practitioner CVC
- AWS Certified Solutions Architect Crash Course

Pearson

Containers Overview

Review of containers

Scheduling and Orchestration

Container networking

Service discovery

Terms

Host

- Virtual or bare metal
- Provides varying levels of CPU, memory, network IO, etc

Cluster

- Collection of hosts

Registry

- Repository for container images

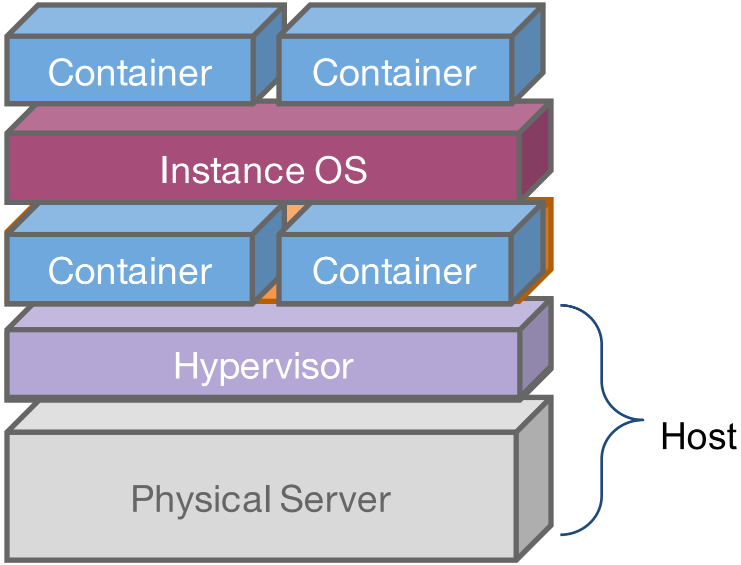
Service

- Specifies set number of container replicas
- Generally long-lived
- May include load balancer

Schedulers

- Determine optimal placement for containers among hosts
- Replace failed containers in a service
- Rebalance containers when host fails

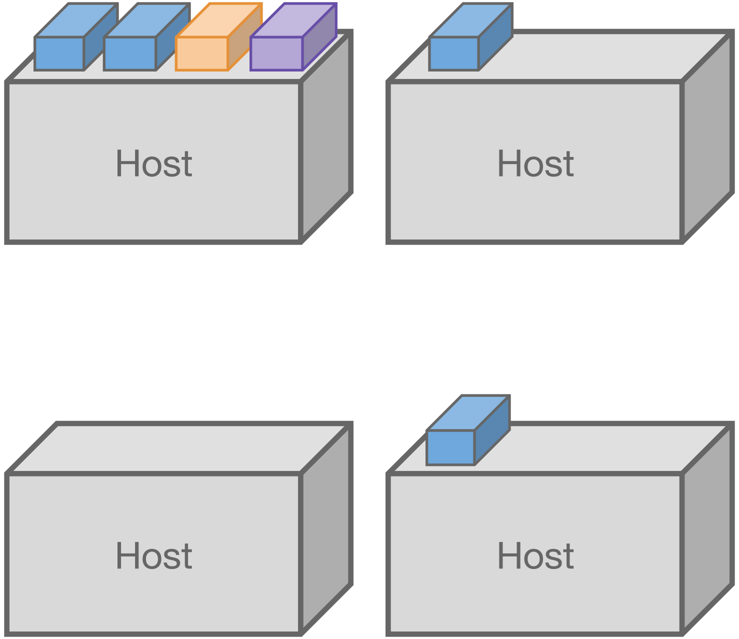
Containers on Amazon EC2



Container

- Runs a process
  (can fork background processes)
- Provides isolation between processes
- Shares the host
  - OS kernel
  - CPU & memory
  - Network and disk IO
- Allows efficient use of available compute resources

Container Scheduling and Orchestration



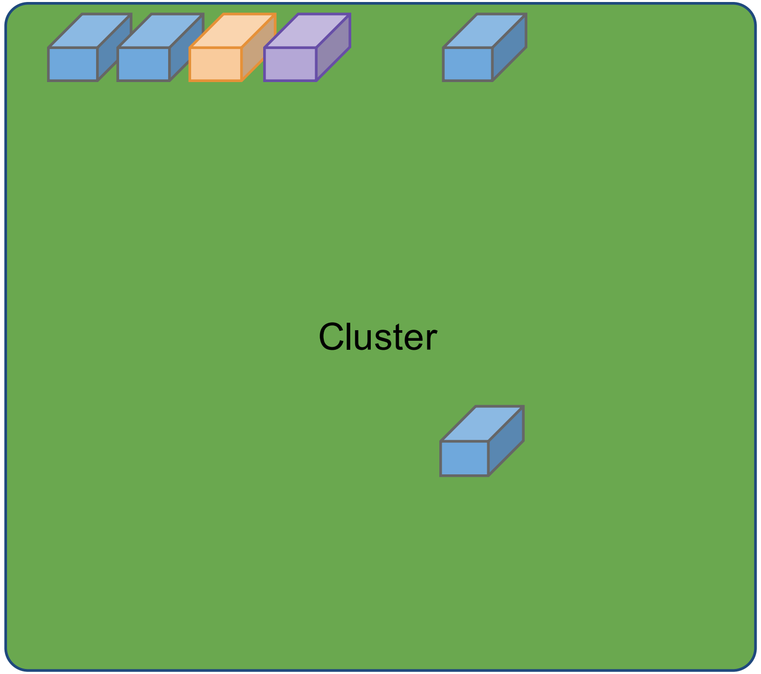
Schedulers track

- State of the cluster
- Current placement of containers
- Available resources per host
- May coordinate with service discovery

Available Schedulers

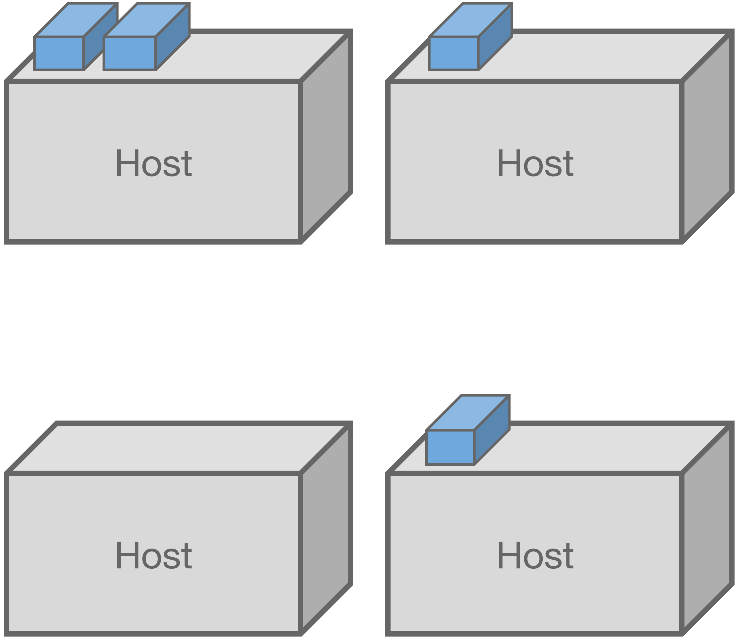
- Docker Swarm
- ECS
- Kubernetes (K8s)
- Mesos

Deployments Simplified



- Cluster provides *single* pool of resources
- Containers are deployed to the *cluster*
- Relieves the burdens of container placement

Service Discovery



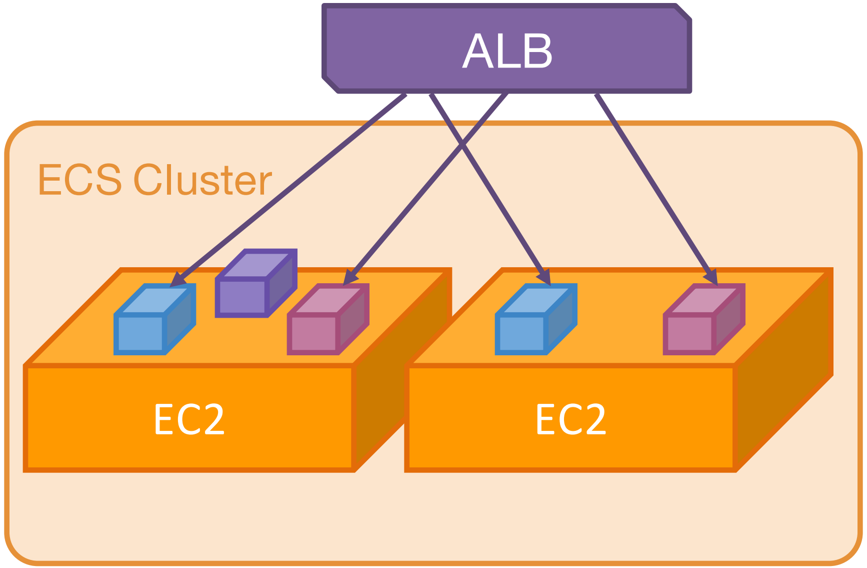
How do we find a service?

- Could have many container replicas
- Each at different IP
- And different port
- Could have multiple versions running concurrently

Service Discovery Options

- Elastic load balancer
- Route 53 auto naming
- Consul
- Etcd
- Zookeeper

Service Discovery with Amazon ECS



- ELB
  - Consistent DNS endpoint
  - ALB supports path-based routing
- ECS Service Discovery
  - Uses Route 53 Auto Naming
  - Registry of service names
  - Names mapped to set of DNS records
  - Supports health checks

CI/CD Pipelines

Image registry

Build tools

Pipelines

Container Images

- Form the basis of containers

  (You start with the image)

- Contains process and all dependencies
- *Are the deployment artifact*
- Replace, by including
  - JAR,EAR,WAR
  - Node, php, python
- Need to be built
  - based upon another image

Container Image Registry

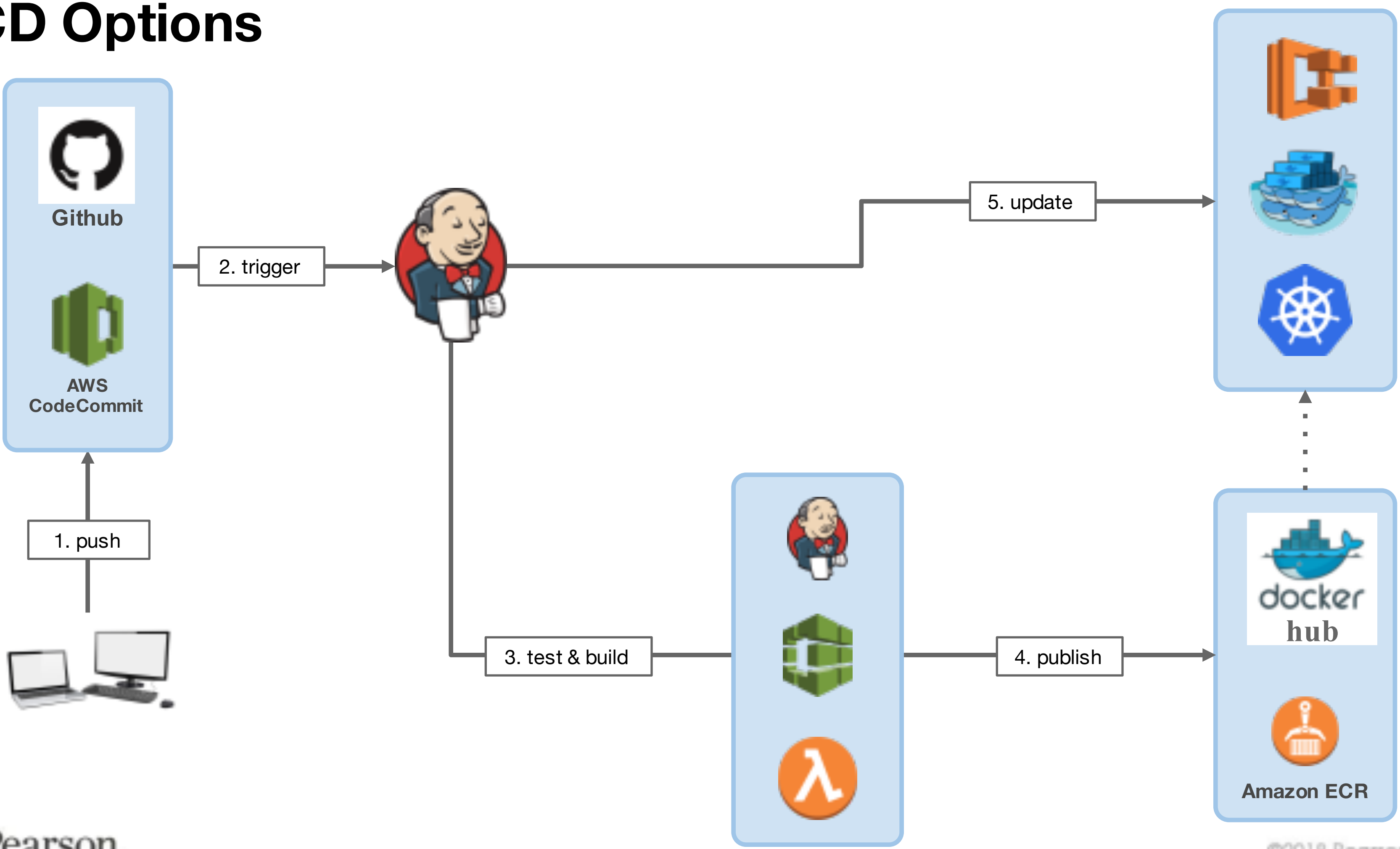
- Is a repository for images
  - therefore your artifact repo
- Public options include
  - Dockerhub
  - Quay.io
  - Artifactory
- Private options include
  - Amazon ECR
  - DIY
    - EC2
    - On-premises

Build Servers/Services

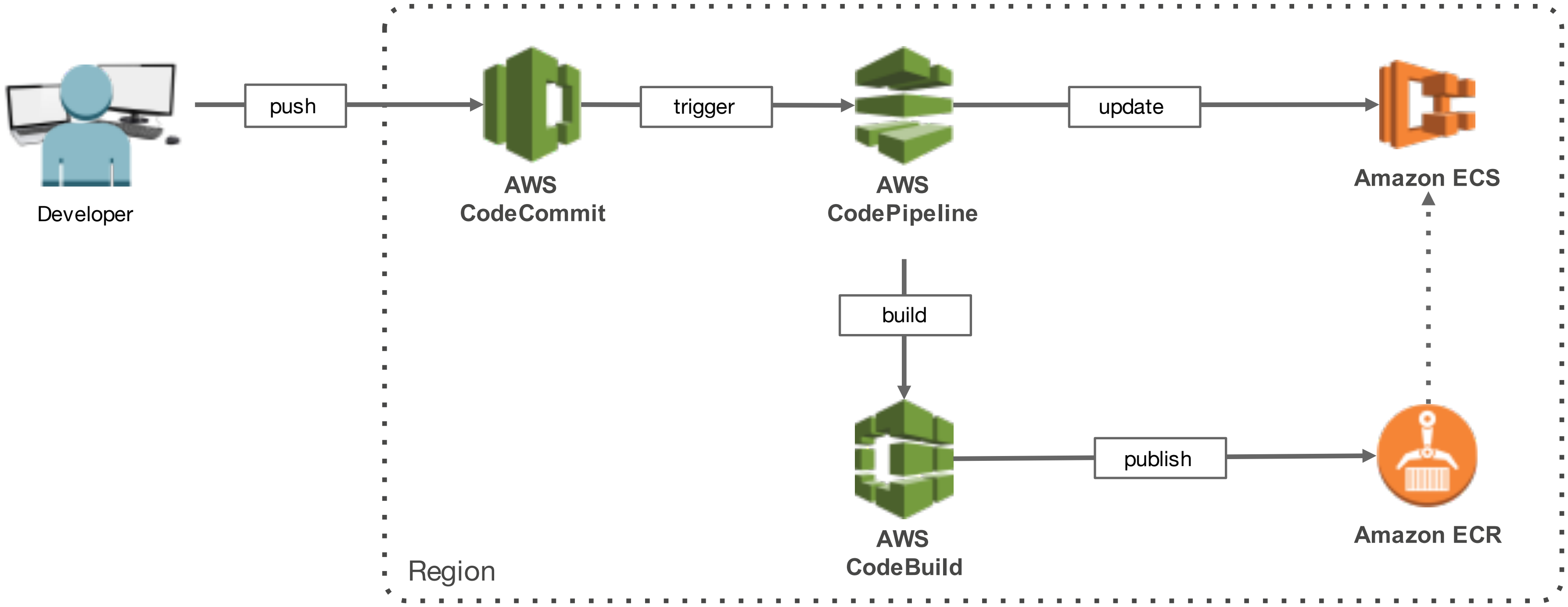
How does the image get created?

- Build servers
  - Jenkins
  - Etc
- Build services
  - Bitbucket pipelines
  - AWS CodeBuild
  - Dockerhub automated builds
  - CircleCI
  - Travis-CI

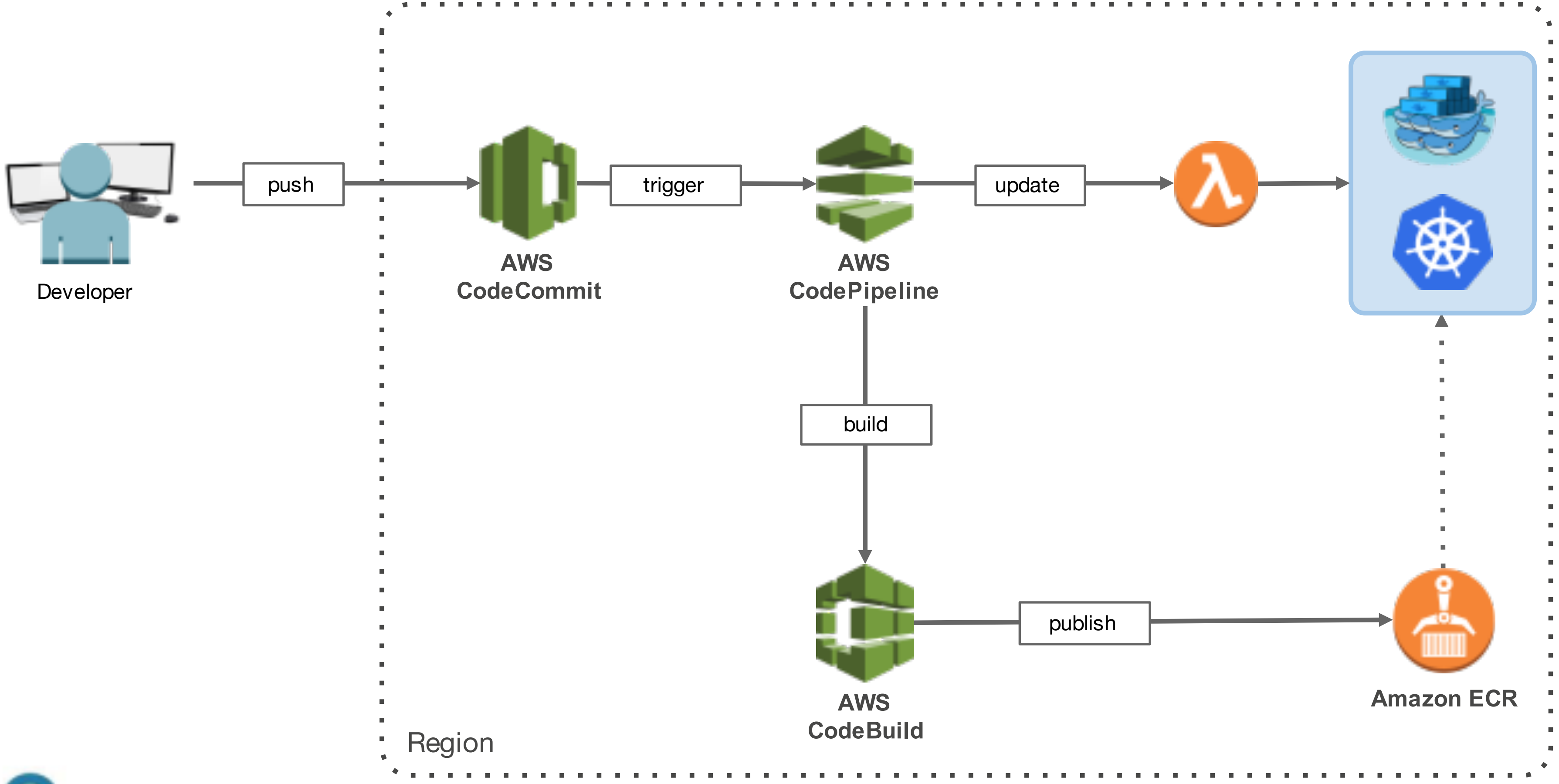
CI/CD Options



CI/CD with AWS Native Services



CI/CD with AWS Services



Docker Swarm on EC2

Terms

Options

Discussion

Docker Swarm on EC2

- Two auto scaling groups
  - 1 for manager nodes
  - 1 for worker nodes
- System containers run on nodes
  - Don’t change them!
- Services with ports will be exposed through Classic Load Balancer

- Service
  - Single container app
  - Can be replicated
- Stack
  - Collection of services
  - Mult-container apps

Docker Swarm for EC2

https://docs.docker.com/docker-for-aws/

Kubernetes on EC2

Terms

Options

Discussion

Kubernetes on EC2

- Numerous options
  - Conjure-up
  - Kubernetes operations (kops)
  - CoreOS Tectonic

- Terms
  - Pod
  - ReplicaSet
  - Service
  - Deployment

Kubernetes on EC2

https://github.com/kubernetes/kops/blob/master/docs/aws.md

Must have kops and kubectl installed

Macs can use homebrew

Elastic Container Service

Terms

Container Networking

Demo: Building and Deploying

ECS Terms

- **Cluster**
  - Logical grouping of *services*
  - May include EC2 instances

- **Service**
  - Maintains long-running *tasks*
  - Can coordinate with ELB

- **Task**
  - Collection of containers
  - Can be deployed with/without Service

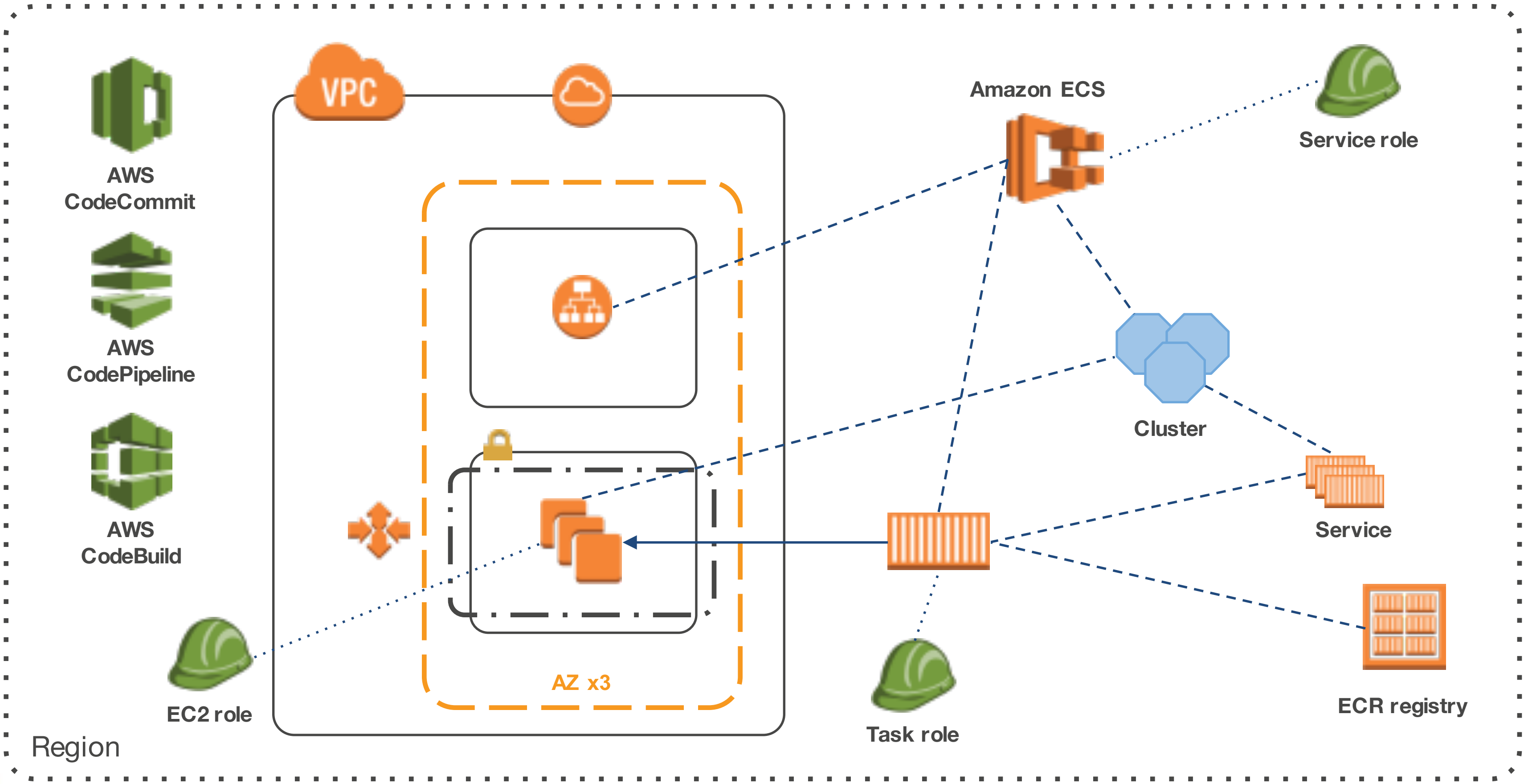
- **Task Definition**
  - JSON template
  - Defines containers
  - Specify CPU, memory, volumes

- **Elastic Container Registry**
  - Private docker image repositories

ECS Networking Modes

- **none**
  - No port mappings
  - No external connectivity
- **host**
  - Container ports mapped directly to host
  - Limits one container per host
  - Better performance than bridge
- **bridge**
  - Virtualized by docker engine
  - Container ports map to *other* ports on host
  - Allows multiple containers on same host
- **awsvpc**
  - Each task gets it own ENI
  - ENI shared by all containers in task
  - Only option supported by Fargate
  - EC2 instance type limits number of ENIs
    - c4.large : 3 ENIs
    - Default ENI (eth0) counts as 1

ECS Architecture



ECS Build Steps

1. Create network stack
2. Create Internet stack
3. Find the latest ECS optimized AMI
4. Create auto scaling group and ECS cluster
5. Create application load balancer
6. Create ECS service with task definition (per service)
7. Create CICD pipeline (per service)

Private Docker Registries

Terms

Architecture

Demo: Building and Deploying

Public or Managed Registries

- Dockerhub
- AWS Elastic Container Registry (ECR)
- Codefresh.io
- Quay.io (“key”)
- Artifactory
- Google Container Registry

- Benefits
  - Managed
  - Some access control
- Cons
  - You don’t control the registry itself
  - Can’t use your own certificates

Private Registries

- Provision a server
- Install Docker
- Configure certs for TLS
- Configure storage volume
- Run registry container

https://docs.docker.com/registry/

- Benefits
  - You control the registry
  - You control all access
  - Use your own certificates (TLS)
- Cons
  - You are responsible for
    - HA
    - Fault tolerance
    - Durability
    - Security

Docker Trusted Registry

- Manage users via LDAP or AD
- Role Based Access Control
  - Fine grained access
- Built-in security scanner
- Image signing (DevSecOps!)

https://docs.docker.com/ee/dtr/

https://docs.docker.com/engine/security/trust/content_trust/

- Benefits
  - Docker content trust
  - Verify publisher
  - Verify data integrity
  - Trust managed through keys
- Cons
  - More complex
  - You are responsible for
    - HA
    - Fault tolerance
    - Durability
    - Security

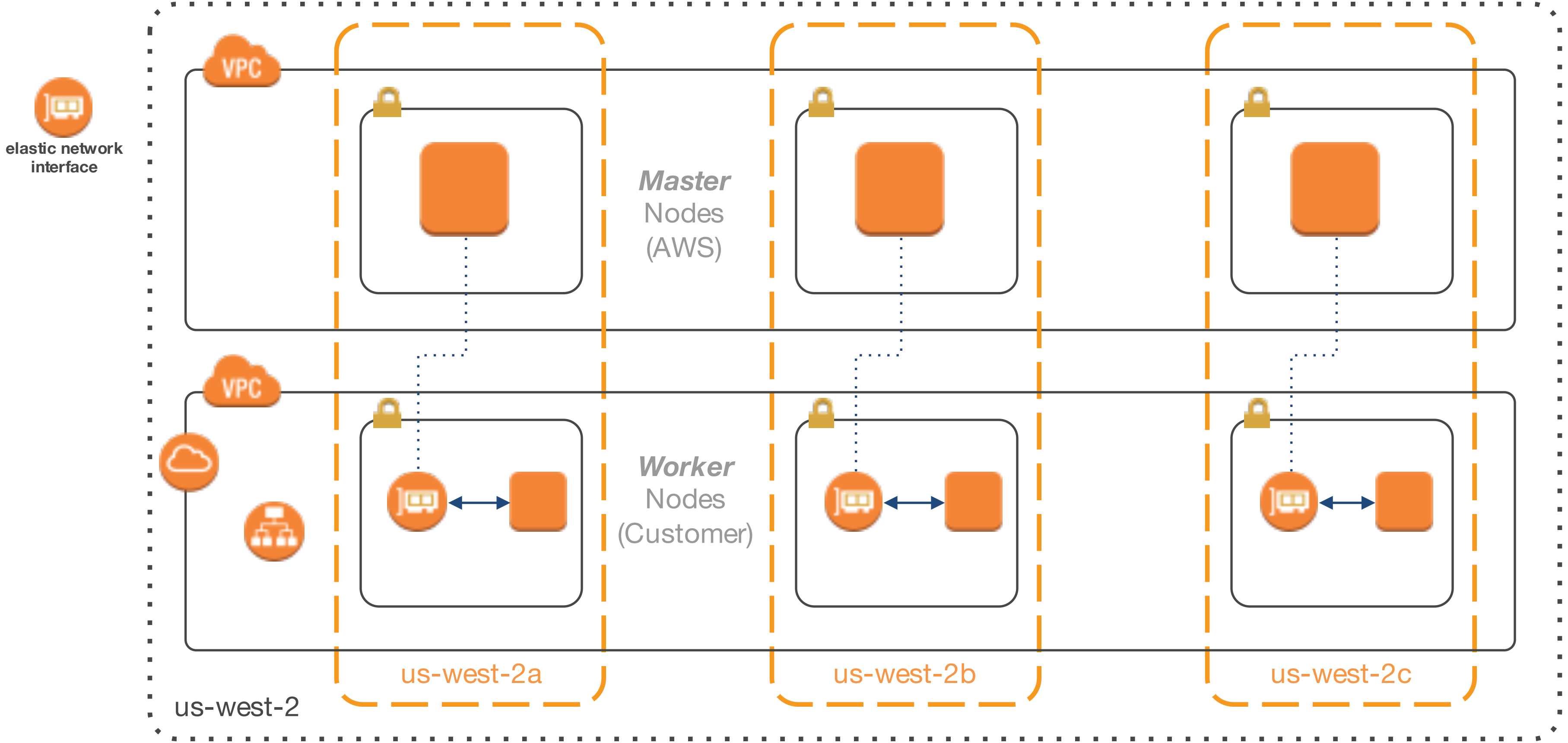
ECS for Kubernetes (EKS)

Terms

Architecture

Demo: Building and Deploying

EKS Architecture



Useful links

https://docs.aws.amazon.com/AmazonECR/latest/userguide/ECR_GetStarted.html
https://jfrog.com/artifactory/
https://quay.io/
https://technologyconversations.com/2015/09/08/service-discovery-zookeeper-vs-etcd-vs-consul/
https://medium.com/@ArmandGrillet/comparison-of-container-schedulers-c427f4f7421
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/service-discovery.html
https://platform9.com/blog/compare-kubernetes-vs-mesos/
https://www.infoworld.com/article/3268073/containers/what-is-kubernetes-container-orchestration-explained.html
https://technologyconversations.com/2017/12/14/what-is-a-container-scheduler/
https://rhelblog.redhat.com/2015/07/29/architecting-containers-part-1-user-space-vs-kernel-space/
https://medium.com/containers-on-aws/choosing-your-container-environment-on-aws-with-ecs-eks-and-fargate-cfbe416ab1a
https://aws.amazon.com/blogs/mt/the-right-way-to-store-secrets-using-parameter-store/
https://docs.aws.amazon.com/codebuild/latest/userguide/build-env-ref-available.html
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-networking.html
https://docs.aws.amazon.com/AmazonECS/latest/developerguide/ecs-optimized_AMI.html
https://blog.giantswarm.io/understanding-basic-kubernetes-concepts-using-deployments-manage-services-declaratively/
https://thenewstack.io/about-etcd-the-distributed-key-value-store-used-for-kubernetes-googles-cluster-container-manager/