upGrad

#LifeKoKaroLift

AWS ECS

Managed Container Orchestration Service by AWS

upGrad

Course: Containerisation at Scale

Lecture On: AWS ECS

Instructor: Dipesh Garg

upGrad

Prerequisites for this Session

- Basic understanding of docker
- Basic understanding of container orchestration

Today's Agenda

- Quick recap of container orchestration
- What is ECS? What are the components of ECS?
- Concepts of ECR, EC2 and Fargate, and task definition and services
- ECS pricing
- Launching the Getting Started cluster
- Launching your own Fargate cluster. Creating your own task definition and services
- Logging and monitoring
- ECS load balancing with ALB

upGrad

Poll 1 (15 seconds)

What is the most cost-effective EC2 pricing option to use for a non-critical overnight workload?

A. On demand

B. Reserved instances that can offer 70% off

C. Dedicated instances for security

D. Spot

Poll 1 (15 seconds)

What is the most cost-effective EC2 pricing option to use for a non-critical overnight workload?

A. On demand

B. Reserved instances that can offer 70% off

C. Dedicated instances for security

D. Spot

upGrad

Poll 2 (15 seconds)

For which of these services does Amazon not charge customers?

A. EC2

B. ECS

C. VPC

D. S3

upGrad

Poll 2 (15 seconds)

For which of these services does Amazon not charge customers?

A. EC2

B. ECS

C. VPC

D. S3

Container Orchestration

Container Orchestration

upGrad

What is container orchestration?

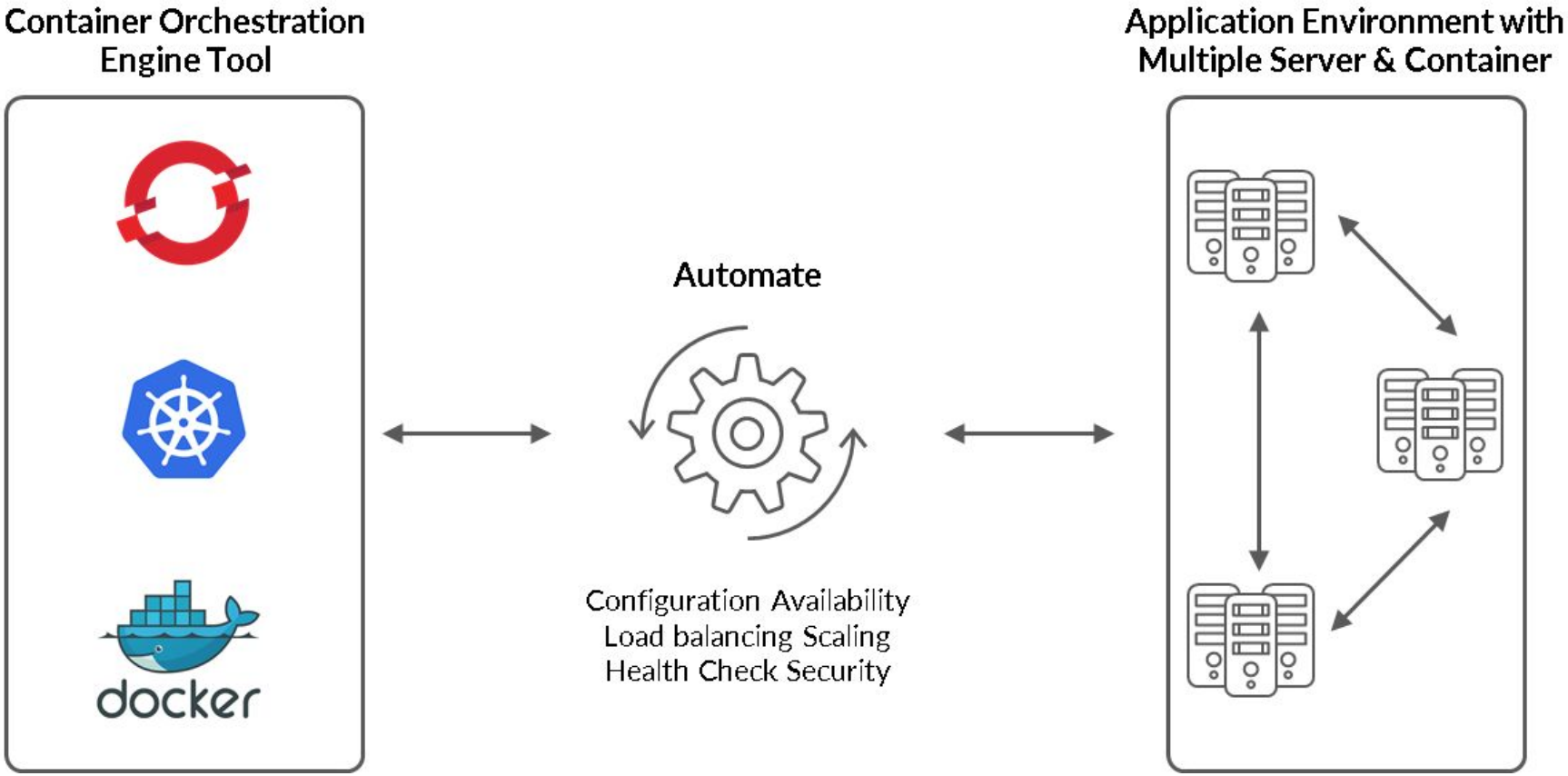
- Container orchestration is all about managing the life cycle of the containers.
- It helps with managing large, dynamic environments.

Life Cycle of Containers

- **Starting phase:** Configuration phase - Container name, image name, port mapping, volume, network configuration
- **Running phase:** Scaling, load balancing, availability, health check
- **Deletion phase:** Clean-up of the resources associated with the container

Container Orchestration

upGrad



Container Orchestration Tools

upGrad

Different Orchestration Tools Available

- Docker Swarm
- AWS managed service - ECS
- Kubernetes - EKS, AKS, GKS
- DigitalOcean Kubernetes Service
- Red Hat OpenShift Online

Introduction to AWS ECS

AWS ECS

upGrad

- AWS Elastic Container Service (AWS ECS) is a container management service that can quickly launch, manage and exit docker containers on a cluster.

- It helps with managing the life cycle of the containers, which includes these three phases:

    - Starting phase
    - Running phase
    - Deletion phase

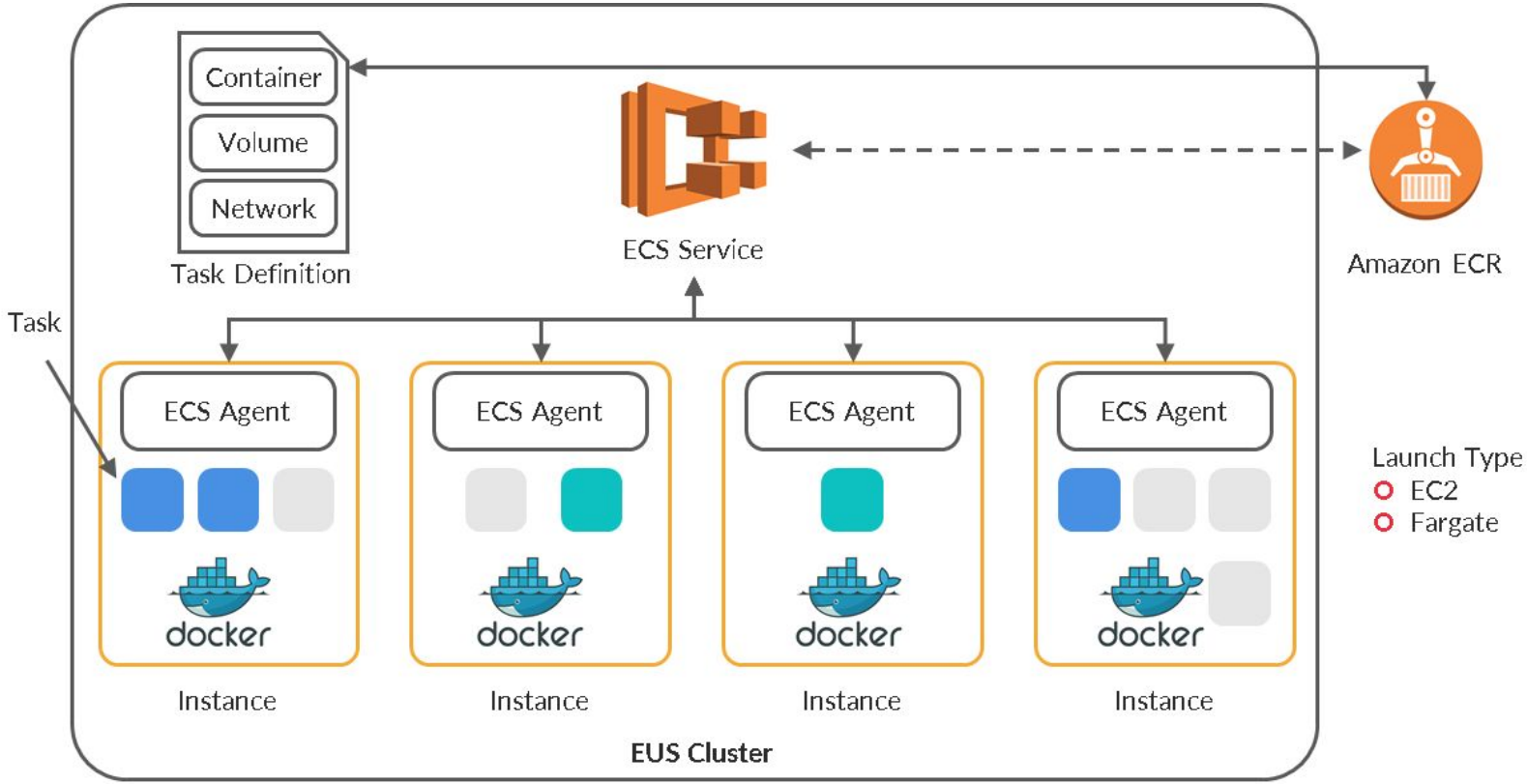
Features of AWS ECS

upGrad

- A service provided by ECS
- A reliable service
- Highly scalable and available
- Rich integration with other AWS services
- Optimised for cost
- Natural fit with all types of workloads
- Provides serverless offerings too
- Integration with third-party CI/CD tools

Architecture of ECS

upGrad



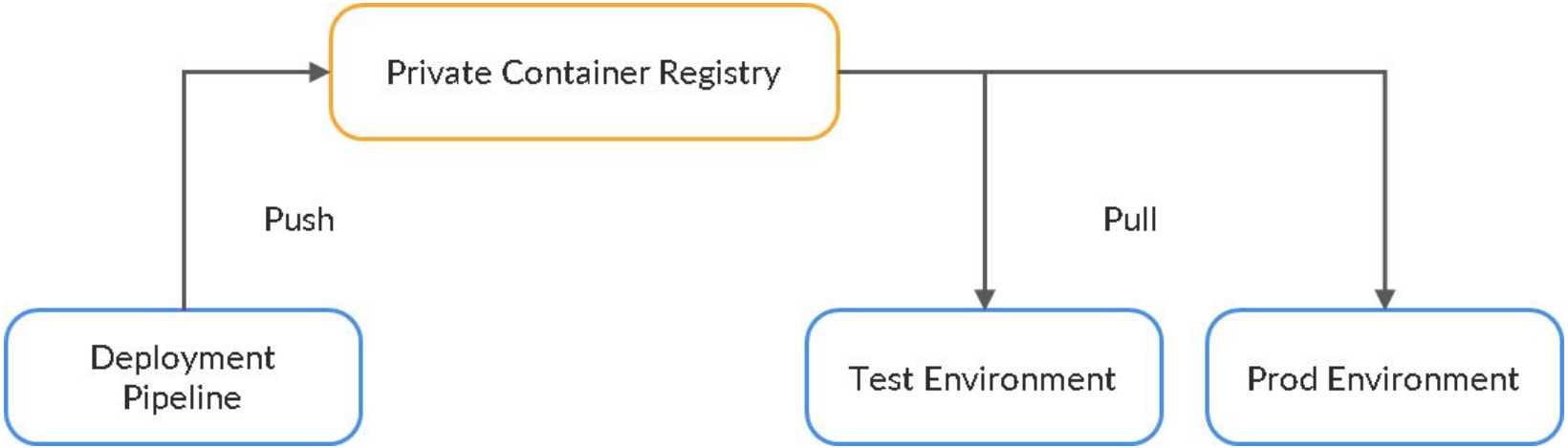
AWS ECR

upGrad

- A fully managed container registry by AWS
- Makes it easy to manage, store, deploy and share container images
- Highly secure and available
- Eases the deployment workflow for pushing and pulling the images
- Enables you to put life cycle policies on the images. Here are some examples:
    - Delete Docker images older than 365 days
    - Delete Docker images if the count is more than ‘n’, say, 25

AWS ECR

upGrad



upGrad

DEMO 1 : ECR

- ECR console and Repository creation
- Docker image building and tagging
- ECR login and ECR image push
- ECR repository lifecycle policies

ECR Versus Docker Hub

upGrad

Feature	ECR	Docker Hub
Repository Type	Private	Public and private
Authentication	AWS IAM	Password/token
Multi-Factor Authentication	Yes	No
Availability SLA	99.9%	NA
Regions	More than 20	Unknown
Pricing	On a storage basis	On the basis of number of users/repositories

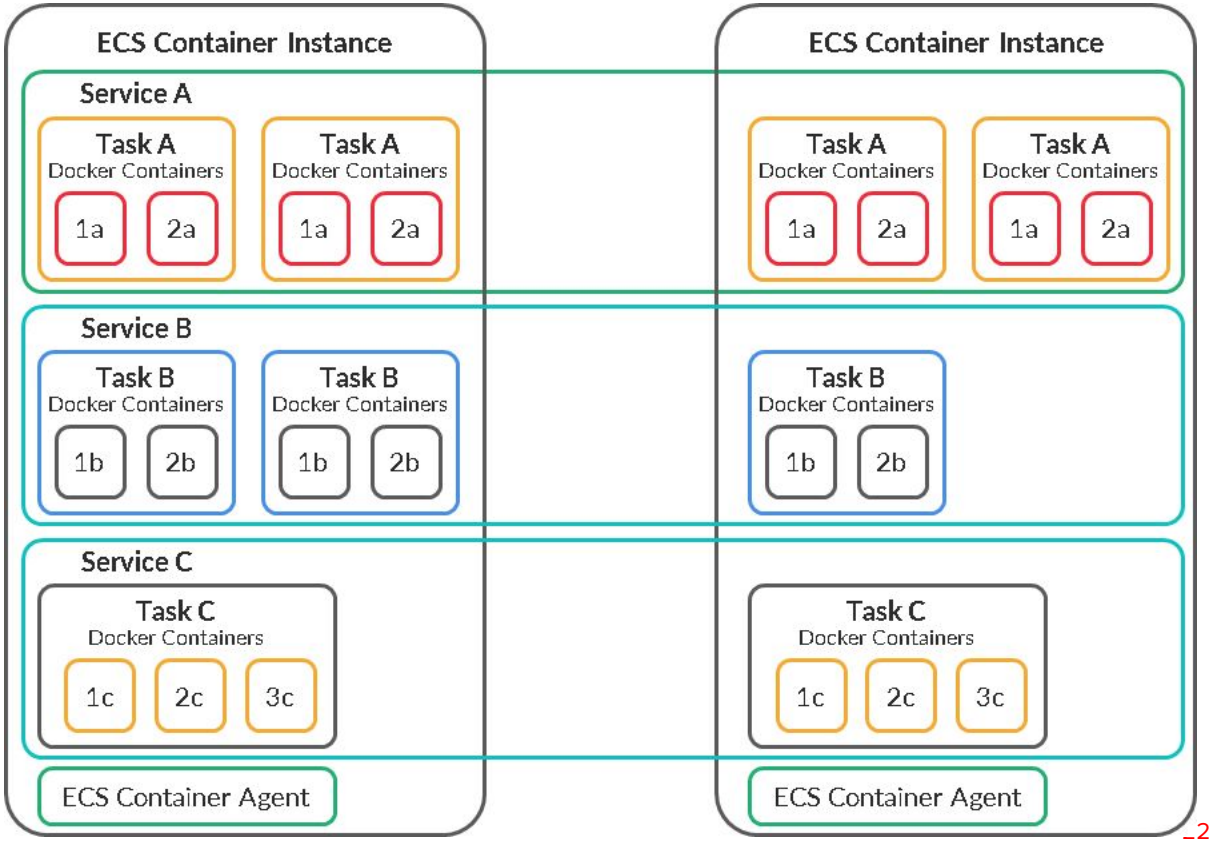
ECS Cluster

upGrad

- An ECS cluster is a grouping of tasks or services, or EC2 instances.
- The clusters are region-specific.
- A cluster can be in the following possible states:
  - ACTIVE
  - PROVISIONING
  - DEPROVISIONING
  - FAILED
  - INACTIVE

ECS Cluster

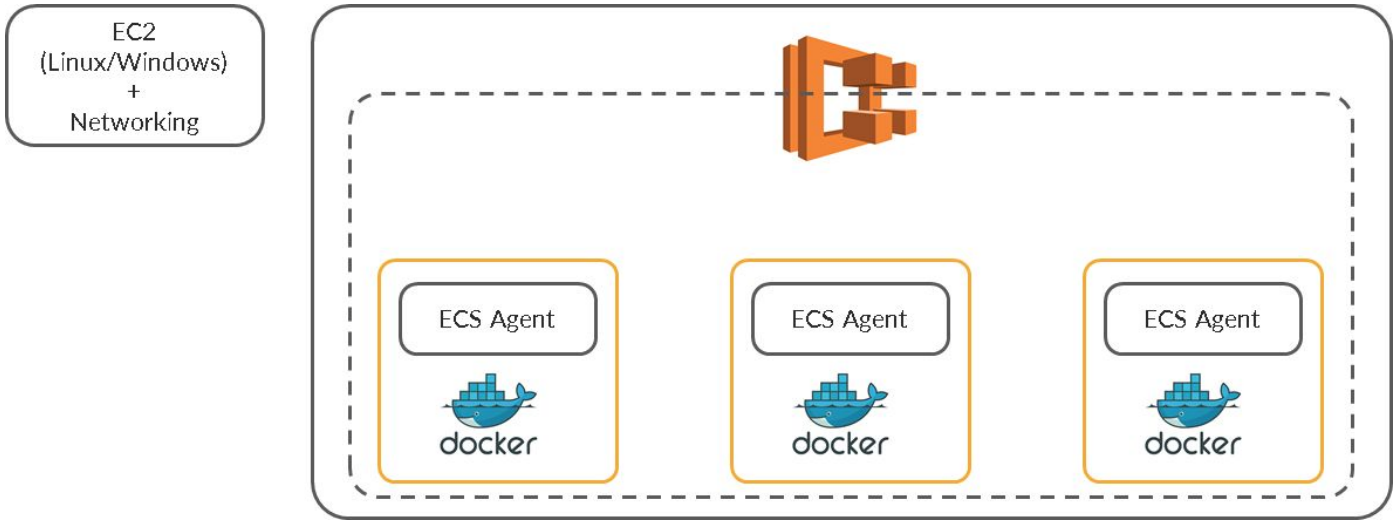
upGrad



ECS Cluster Launch Types

upGrad

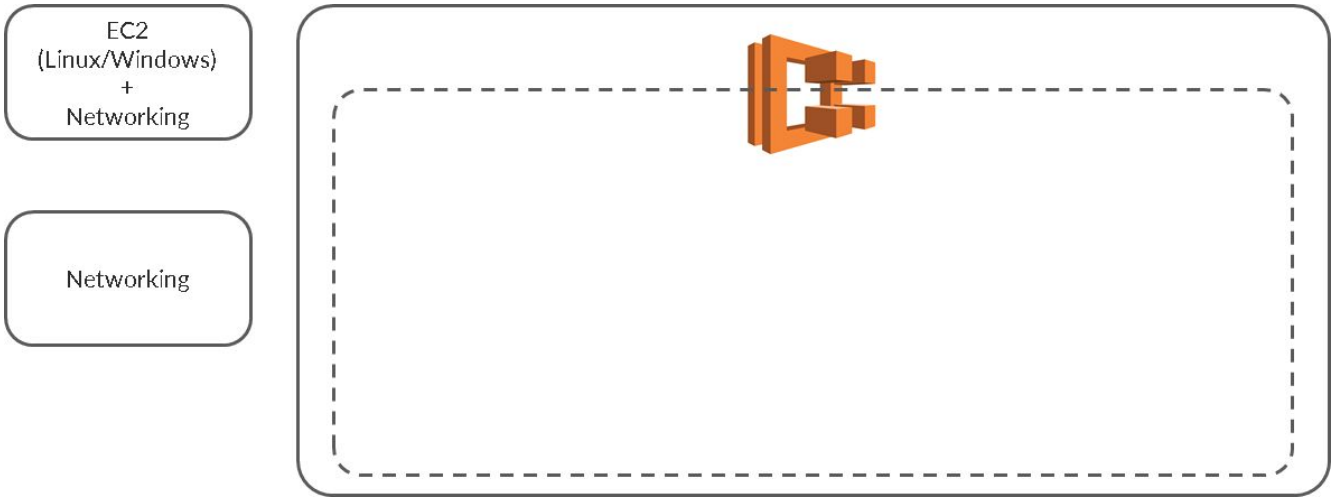
- It determines the type of infrastructure on which your services and tasks are placed.
    - EC2 launch type



ECS Cluster Launch Types

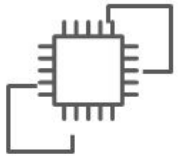
upGrad

- It determines the type of infrastructure on which your services and tasks are placed.
  - EC2 launch type
  - Fargate launch type

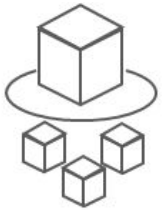


ECS Cluster Launch Types

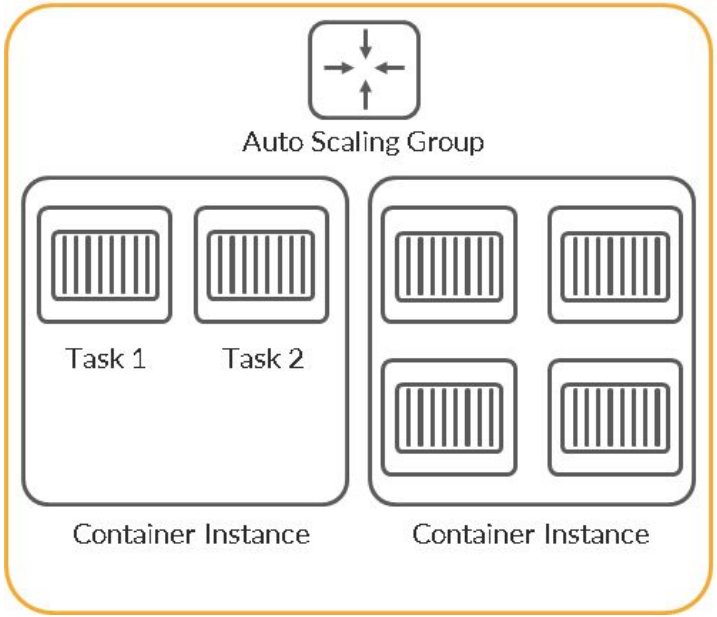
upGrad

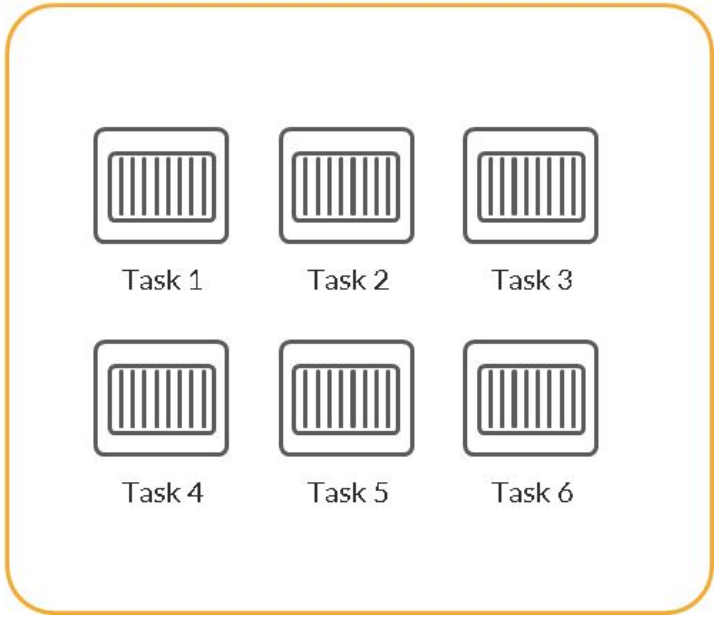


Amazon EC2



AWS Fargate





Serverless Offerings

upGrad

- The AWS Fargate launch type model comes under serverless offerings.
- What is serverless?
  - With serverless offerings, all infrastructure tasks, such as provisioning, managing and patching, are handled by AWS.
  - It means infrastructure exists at the back end but is completely managed by AWS.
  - It comes with automatic scaling, built-in high availability and a pay-for-value billing model.
  - **Examples:** AWS ECS Fargate, Lambda, S3

ECS Pricing

upGrad

- **Fargate launch type model**
  - Pay for the amount of vCPU and memory resources that your containerised application requests
  - The Fargate model is preferred in the following scenarios:
    - **While running batch workloads:** Ex -a cron job that runs once an hour.
    - **Application with an occasional burst:** Run small containers on fargate with auto scaling enabled to handle bursts.
    - For environment where low cpu usage us required, AWS Fargate is a perfect fit. It is generally wasteful to run a tiny application environment on an EC2 as you will pay for the full instance.

ECS Pricing

upGrad

- EC2 launch type model
  - Pay for the AWS resources (e.g., EC2 instances or EBS volumes) you create to store and run your application
  - EC2 model is preferred in following scenarios:
    - You need server-level and granular control over infrastructure
    - Workload is consistent for CPU or memory requirements: You can choose a reserved EC2
    - When you are not sure about the container's resource needs.

Poll 3 (15 seconds)

Rohan would prefer to use AWS-managed infrastructure for running the containers as he wants to pay only for the resources that the application will consume and does not want to pay for all the underlying unused capacity.

Which of these solution options would fulfil these requirements?

A. AWS ECS EC2 mode

B. AWS ECR

C. AWS ECS Fargate mode

D. Spot instances

Poll 3 (15 seconds)

Rohan would prefer to use AWS-managed infrastructure for running the containers as he wants to pay only for the resources that the application will consume and does not want to pay for all the underlying unused capacity.

Which of these solution options would fulfil these requirements?

A. AWS ECS EC2 mode

B. AWS ECR

C. AWS ECS Fargate mode

D. Spot instances

Poll 4 (15 seconds)

AWS Lambda and ECS EC2 mode come under AWS serverless offerings.

A. False

B. True

Poll 4 (15 seconds)

AWS Lambda and ECS EC2 mode come under AWS serverless offerings.

A. False

B. True

ECS Task Definition

upGrad

- It is similar to the docker run command.
- It is the starting phase of the container.
- It defines:
  - The docker image,
  - The CPU and memory required,
  - The networking mode,
  - Logging,
  - The Command (CMD/Entrypoint) and
  - The IAM roles.

ECS Service

upGrad

- It enables maintaining and running a desired number of instances of a task definition simultaneously.
- If any task fails, the ECS service launches a new task to maintain the desired state.

upGrad

DEMO 2 : ECS getting started

- ECS Console : a region based service
- EC2 and Fargate mode overview
- Launch getting started cluster
- ECS task and service overview

ECS Task Components

upGrad

- Standard : Container name, Image
- Memory limits (soft & hard)
- Port mappings
- Network settings
- Storage
- Logging
- Labels
- Environment Variables
- Task IAM roles
- Placement Constraints

ECS Task Definition: Networking Modes

upGrad

Networking Modes

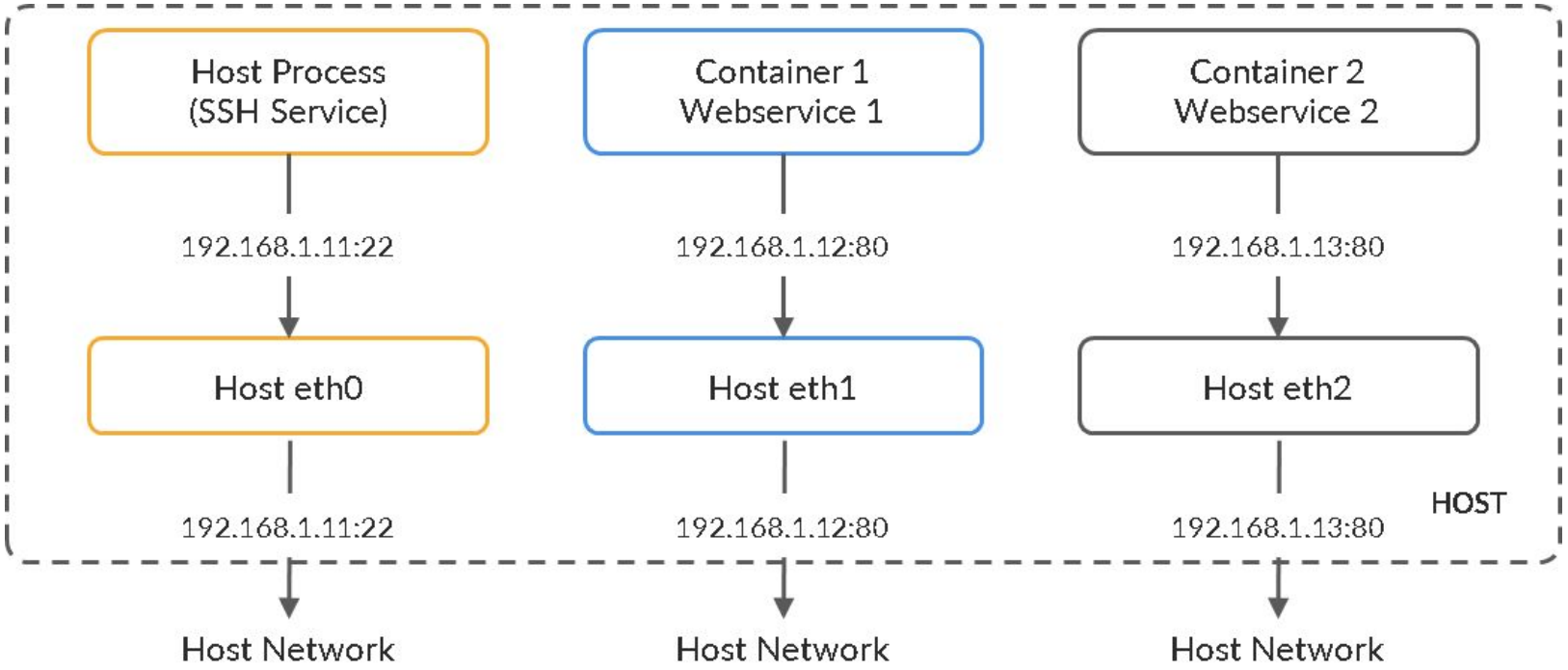
- **awsvpc:** The task is allocated its own elastic network interface (ENI) and a primary private IPv4 address. This provides the task the same networking properties as Amazon EC2 instances.
- **Bridge:** The task utilises docker's built-in virtual network, which runs inside each Amazon EC2 instance hosting tasks.
- **Host:** The task bypasses docker's built-in virtual network and maps the container ports directly to the ENI of the Amazon EC2 instance hosting the task. As a result, you cannot run multiple instantiations of the same task on a single Amazon EC2 instance when port mappings are used.
- **None:** The task has no external network connectivity.

Networking Mode: awsvpc

upGrad

- **awsvpc:** It offers faster performance. Fargate supports only awsvpc networking mode.

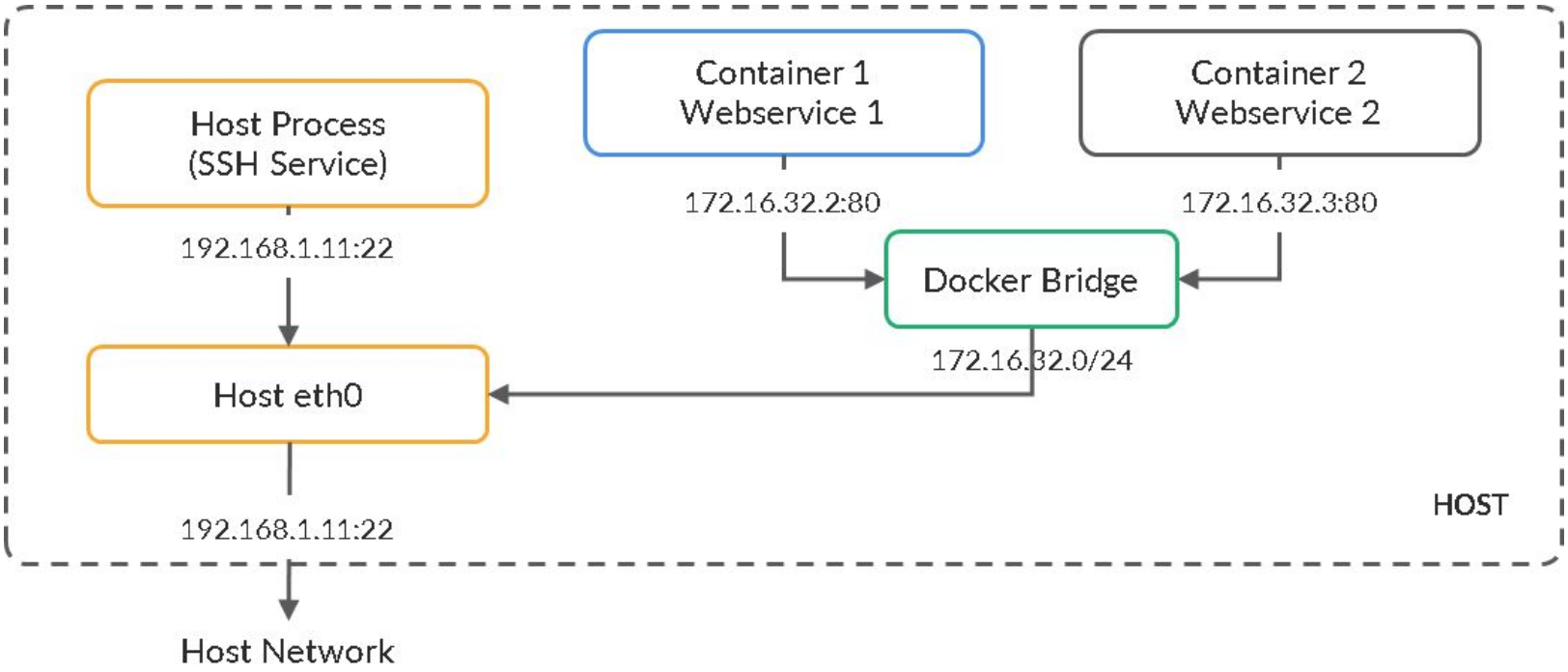
https://tutorialsdojo.com/ecs-network-modes-comparison/



Networking Mode: Bridge

upGrad

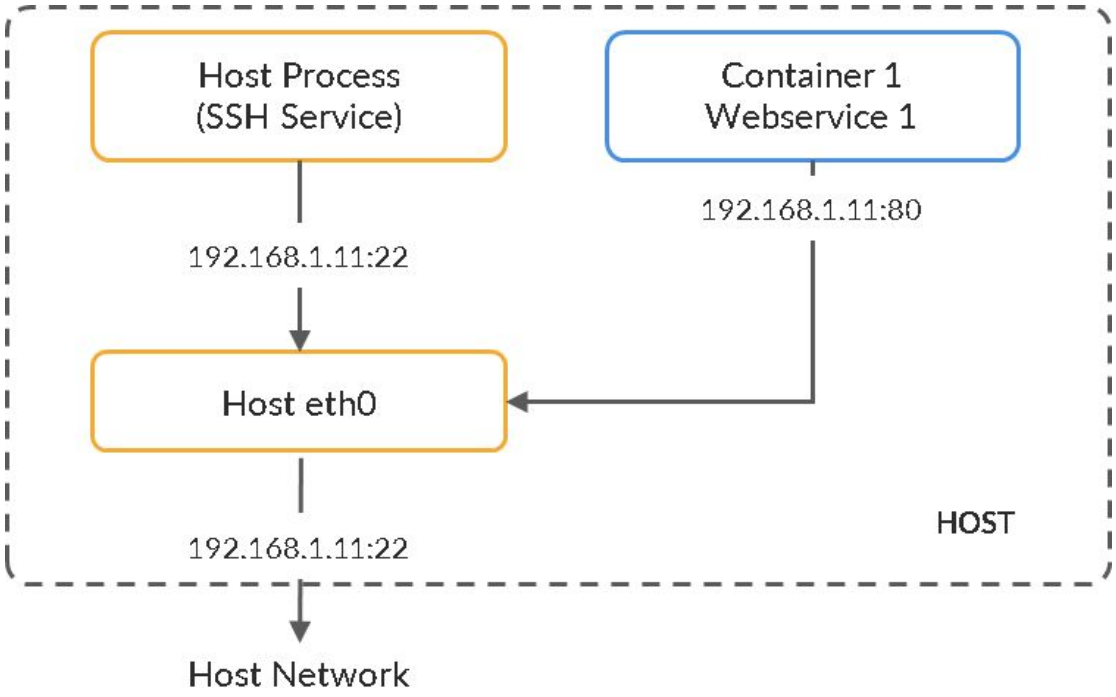
- **Bridge:** It does not provide the best networking mode, since the bridge network is virtualised and Docker software handles the traffic.



Networking Mode: Host

upGrad

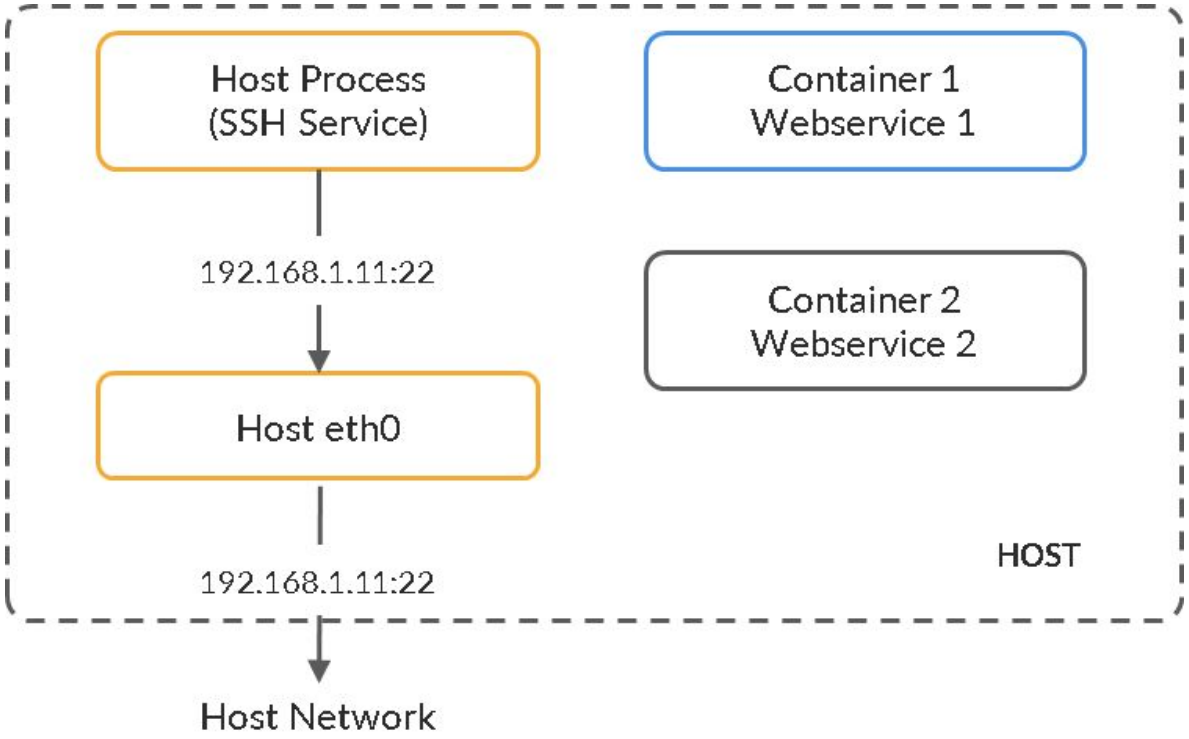
- **Host:** It bypasses docker's built-in network. Drawback: You cannot have multiple containers on the host using the same port.



Networking Mode: None

upGrad

- **None:** You can use this if you do not want the container to access the host network.



ECS Task Definition: Volume

upGrad

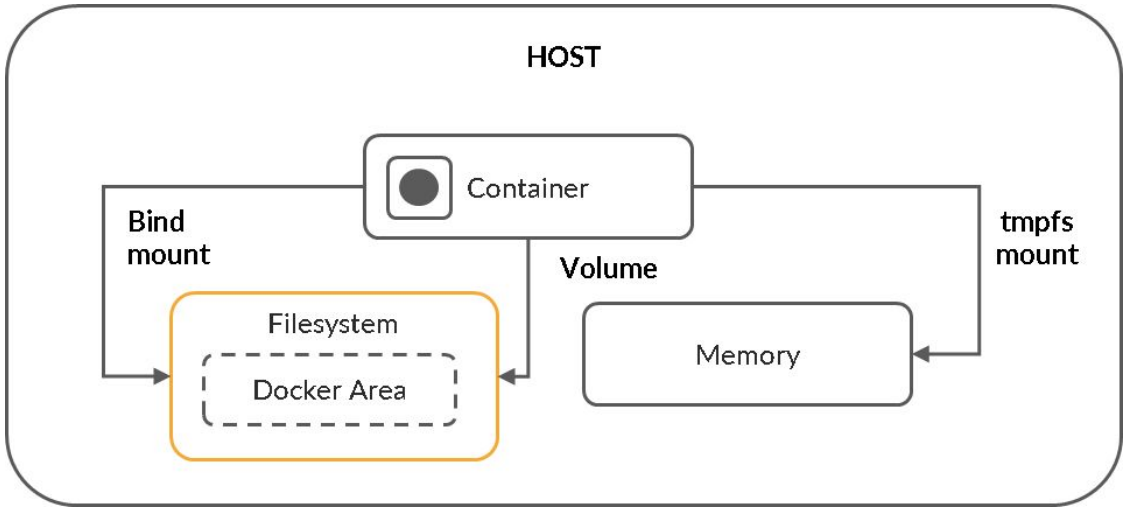
- Data volumes common examples:
  - Persistent data volumes for use with a container
  - Mount non-persistent and empty volumes on multiple containers
  - To share defined data volumes at different locations
  - Mount the volume managed by a third-party volume driver

- Types of data volumes:
  - Docker volumes
  - Bind mounts

ECS Task Definition: Volume Types

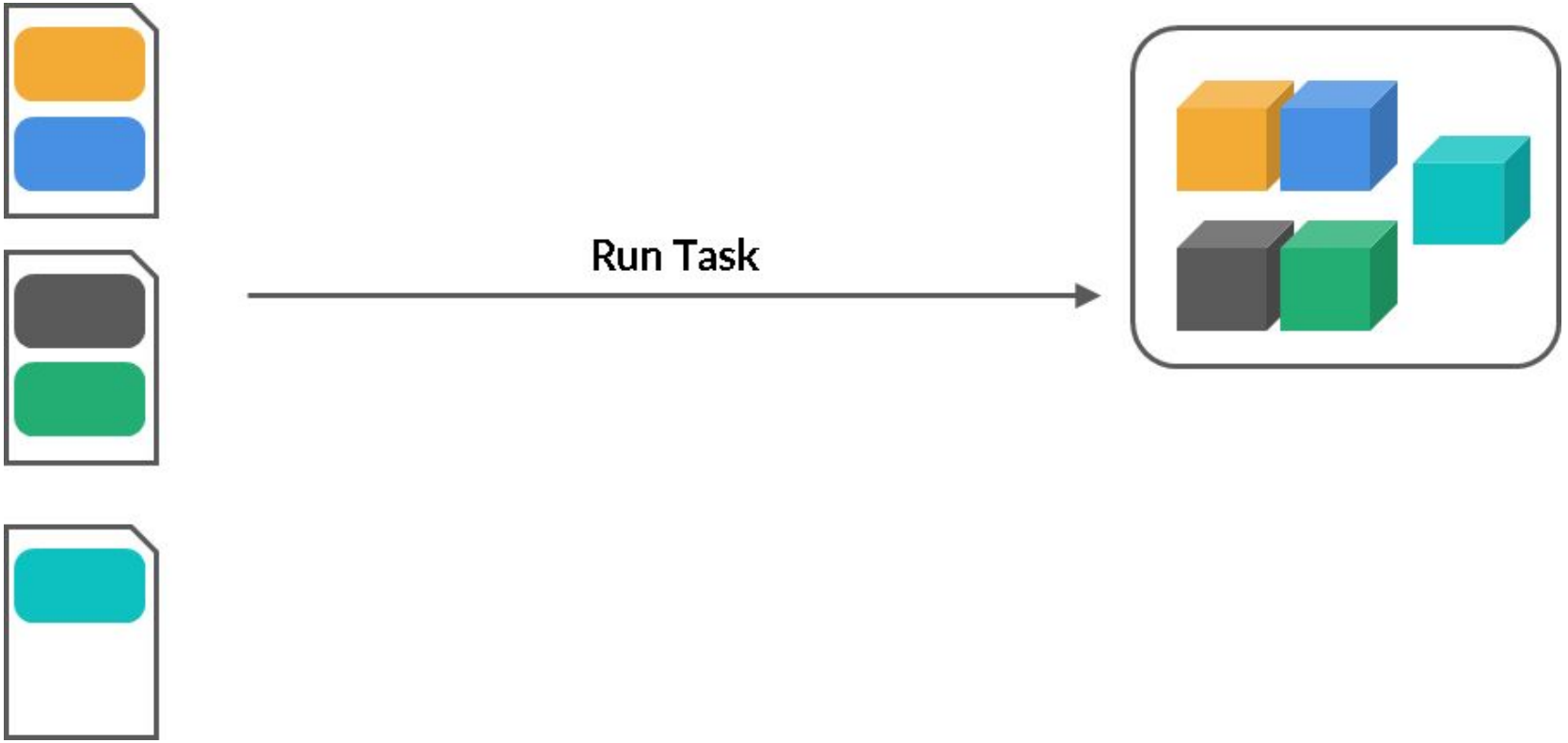
upGrad

- **Docker volume:** A new directory is created within the docker storage directory on the host machine and docker manages that directory's contents.
- **Bind mounts:** A file or directory on the host machine is mounted into a container.



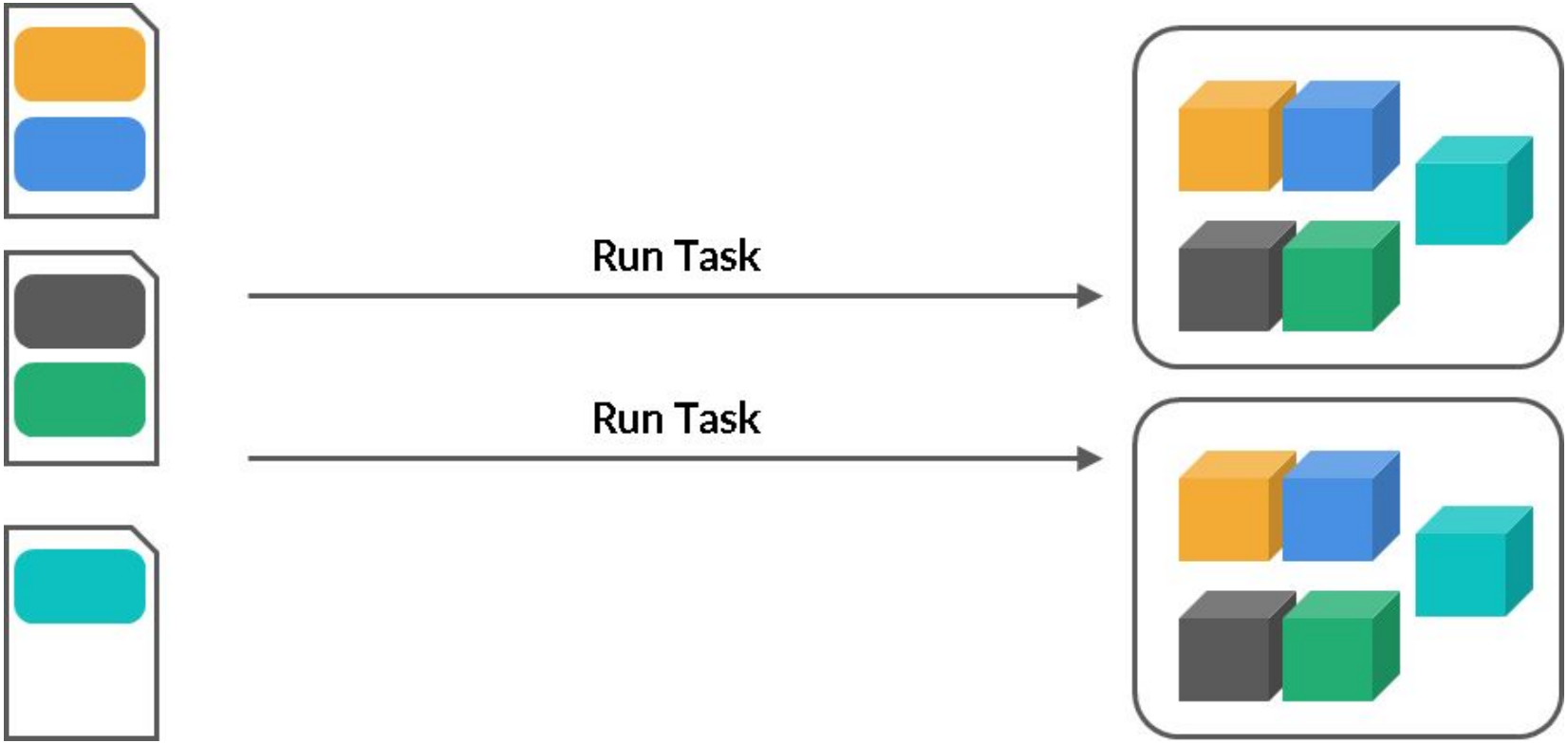
ECS Service

upGrad



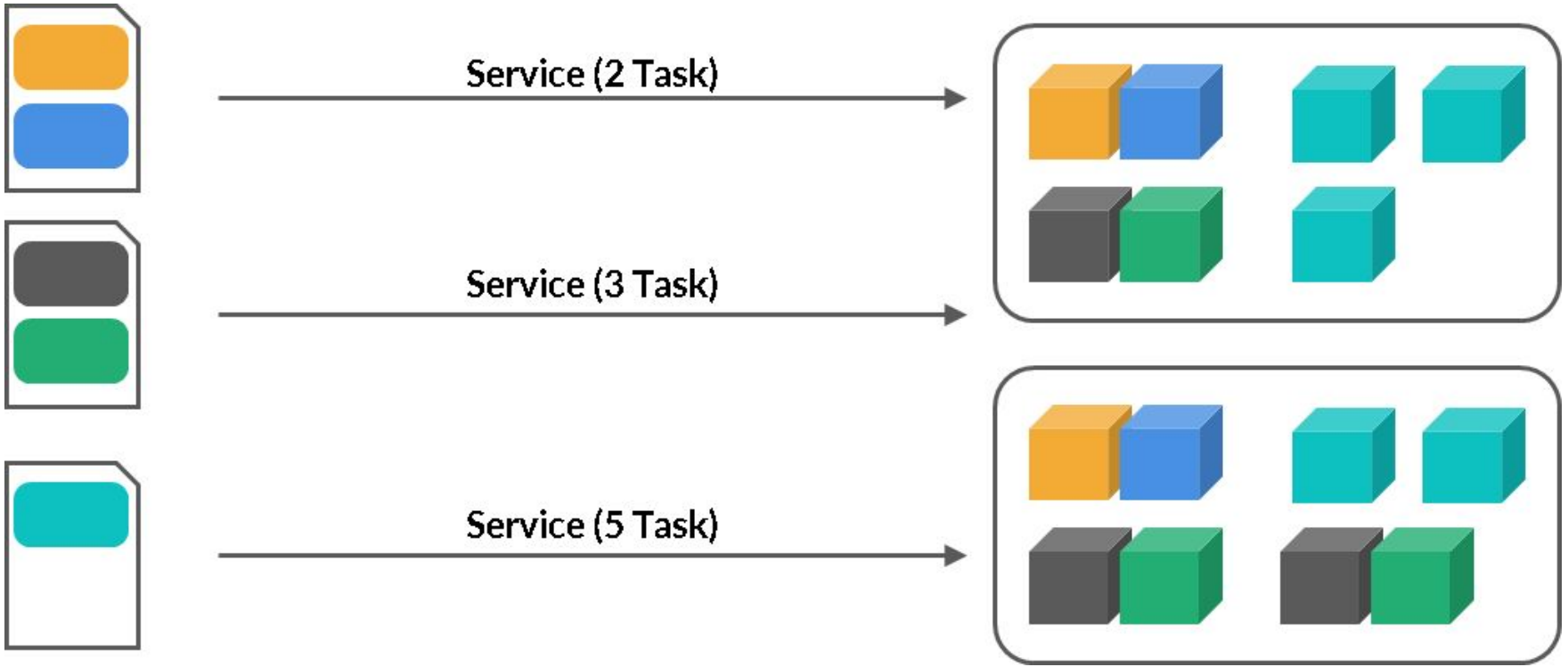
ECS Service

upGrad



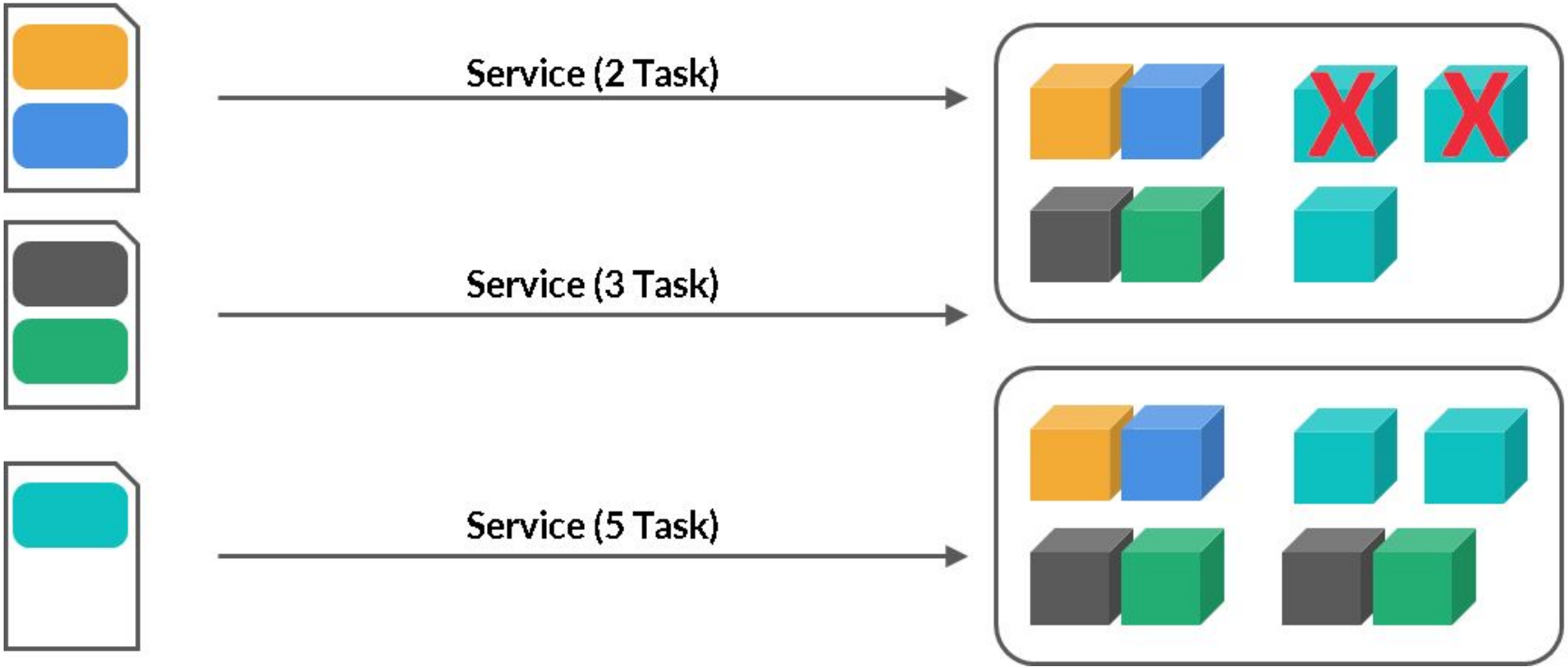
ECS Service

upGrad



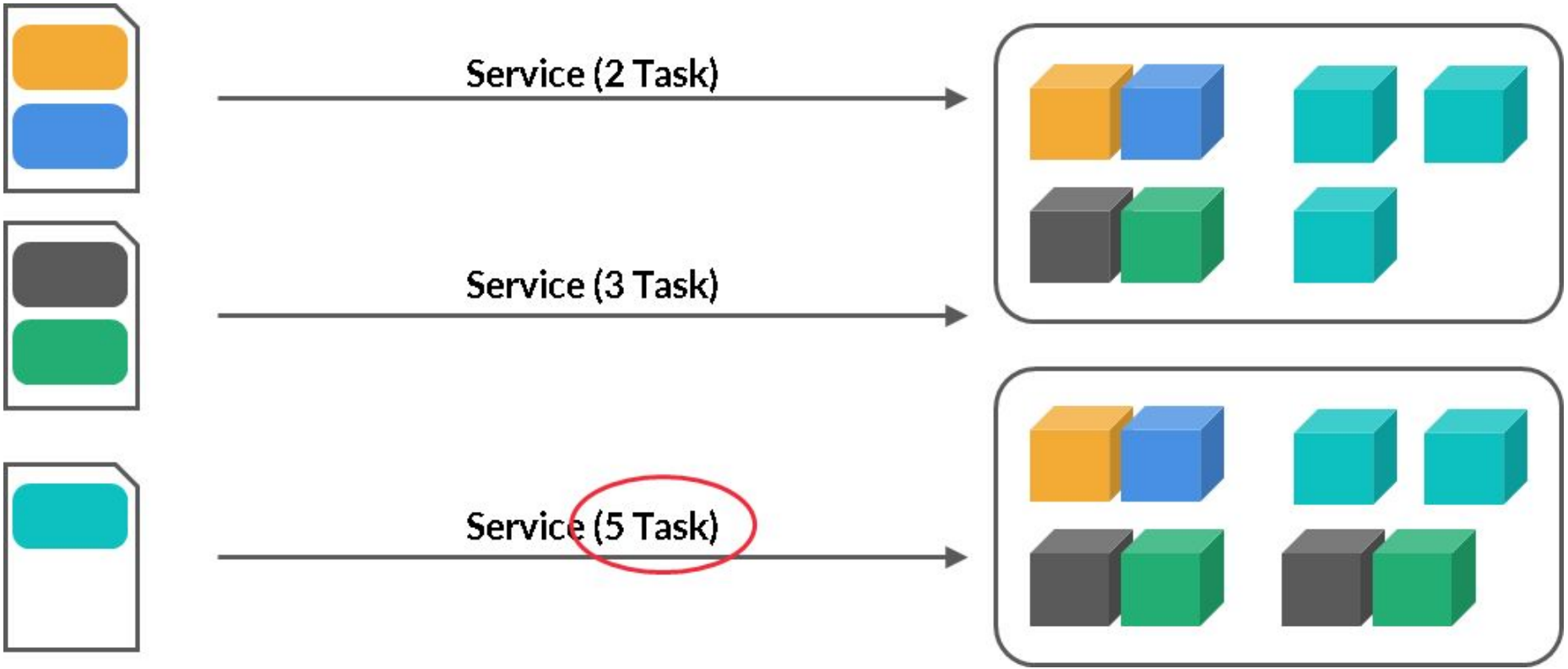
ECS Service

upGrad



ECS Service

upGrad



ECS Service

upGrad

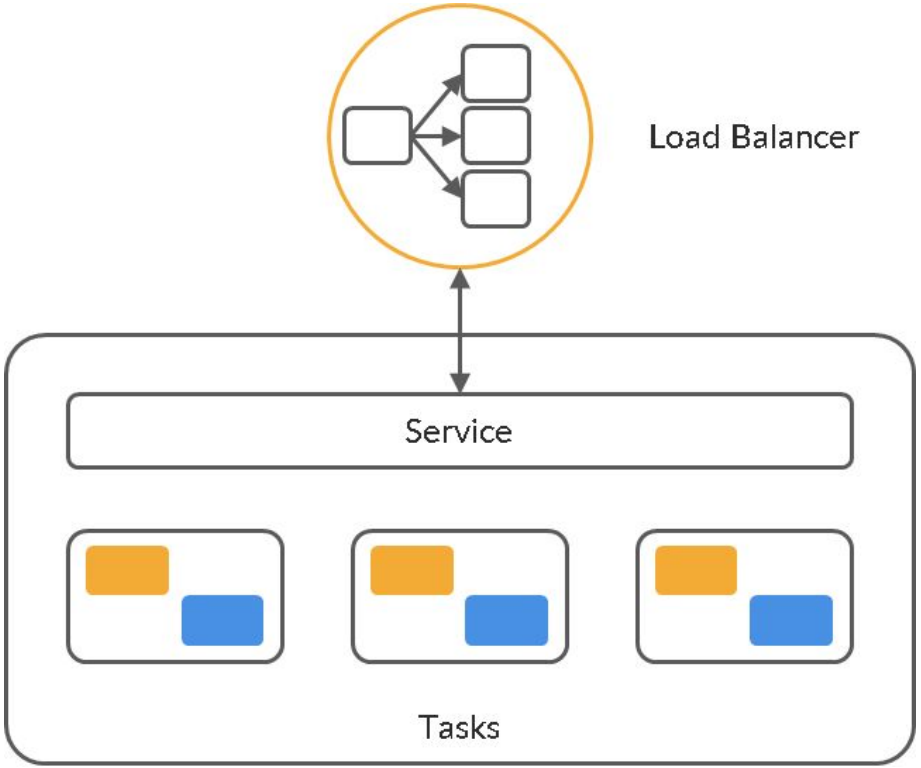
- It enables maintaining and running a desired number of instances of a task definition simultaneously.
- If any task fails, the ECS service launches a new task to maintain the desired state.

- Service scheduler concepts:
  - **Replica** Specify the desired count
  - **Daemon:** Exactly one task on each EC2 Instance

ECS Load Balancing

upGrad

- You can use elastic load balancing to distribute traffic evenly across the tasks in the service.



ECS Monitoring

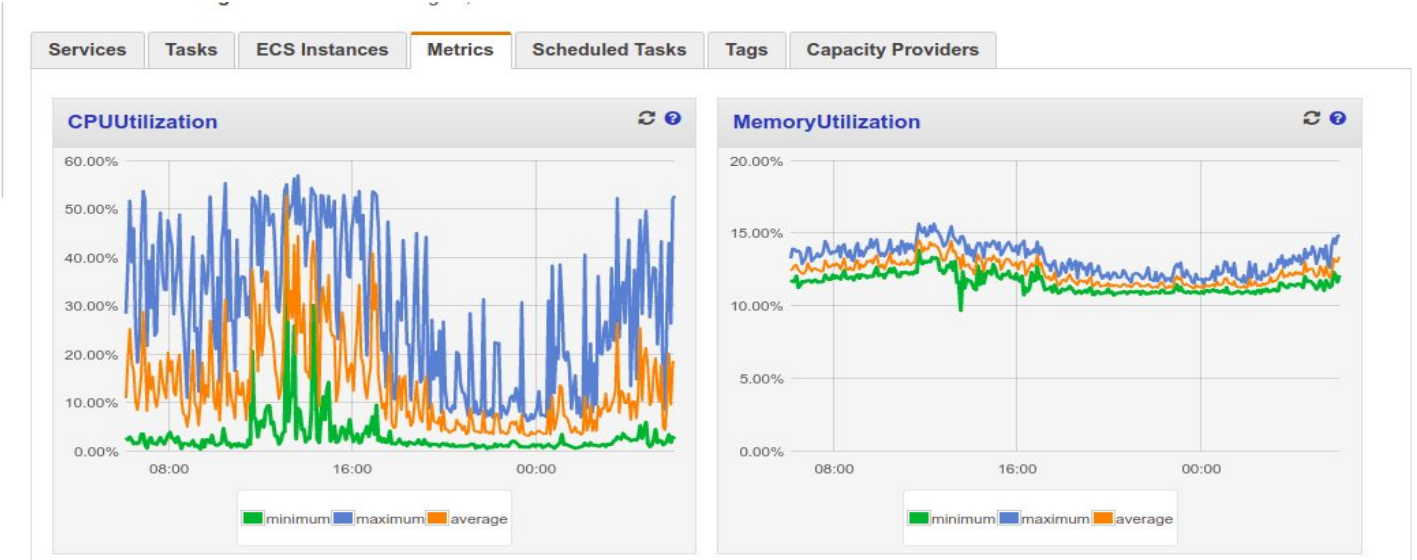
upGrad

- Consider:
  - Monitoring systems should head off small problems before they impact the end users and become big.
  - How often and what are the resources that you would monitor?
  - What are the monitoring tools that you will use?
  - Who should be notified when something goes wrong?

ECS Monitoring

upGrad

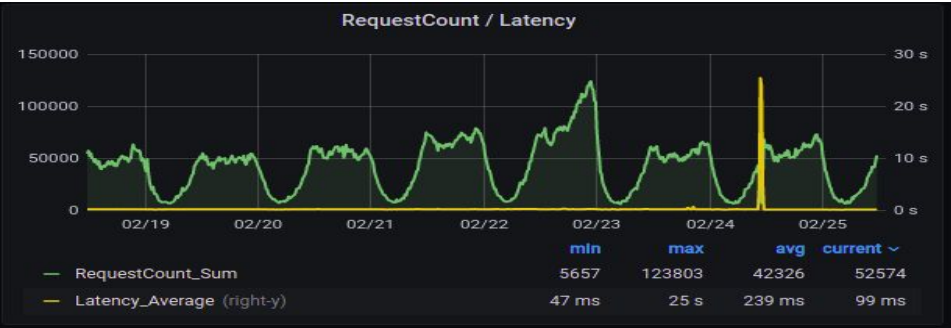
- Easily monitored using CloudWatch:
  - The CPU and memory of the ECS clusters
  - The CPU and memory utilisation metrics for the ECS services

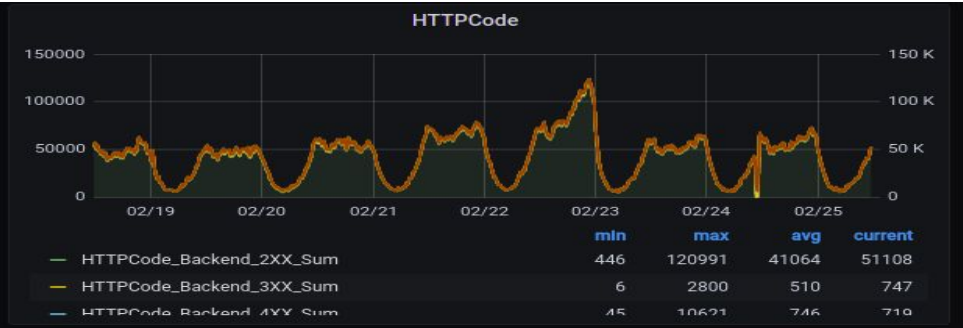


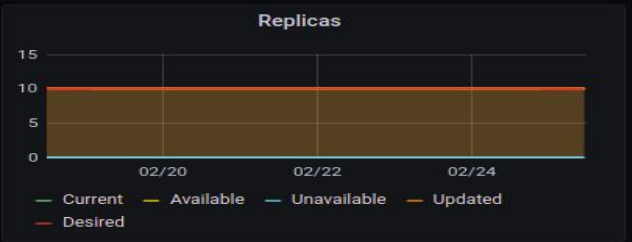
ECS Monitoring

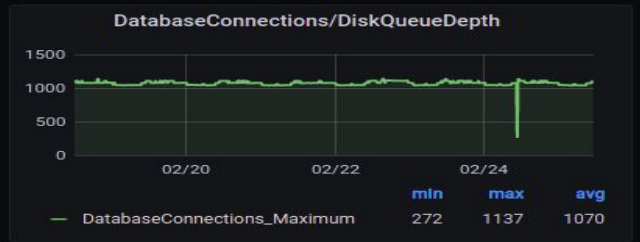
upGrad

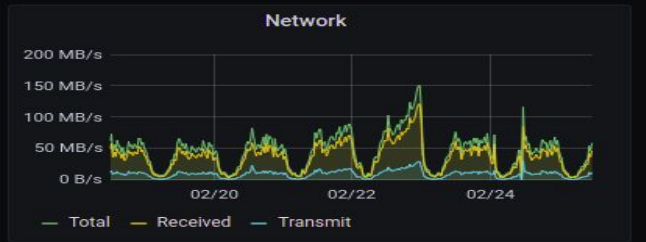
- Important and advanced monitoring metrics:
  - Request count, latency, healthy instances, HTTP codes and many more







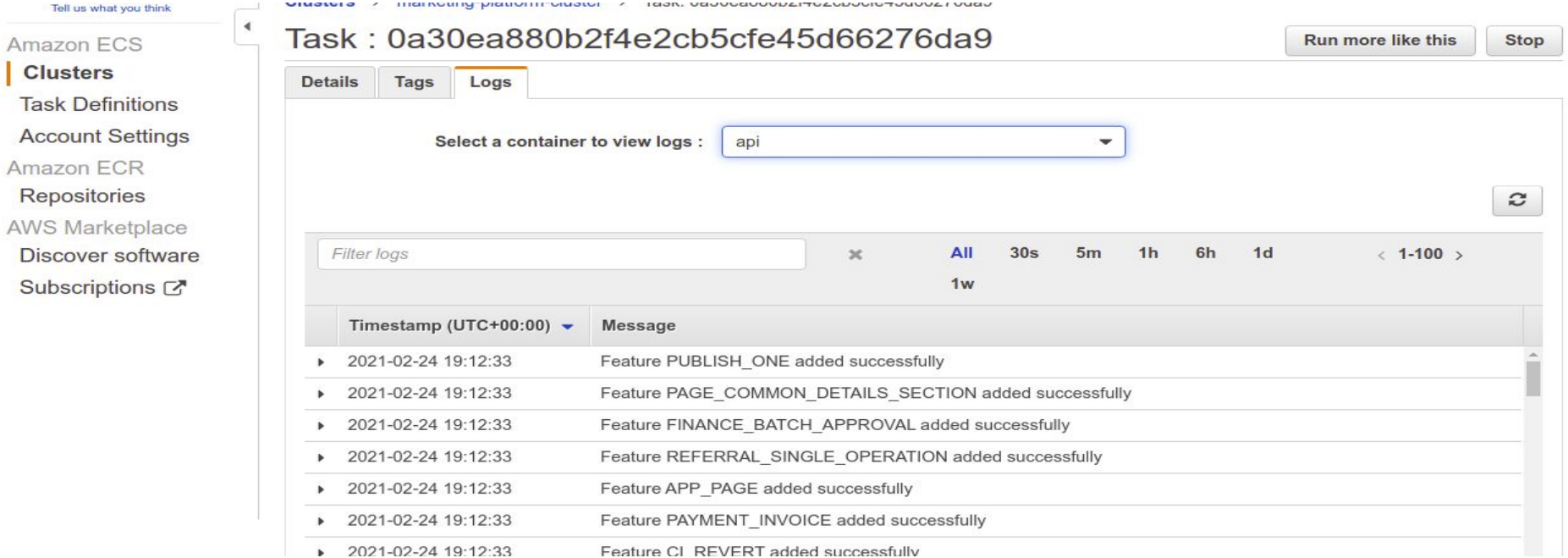




ECS Logging

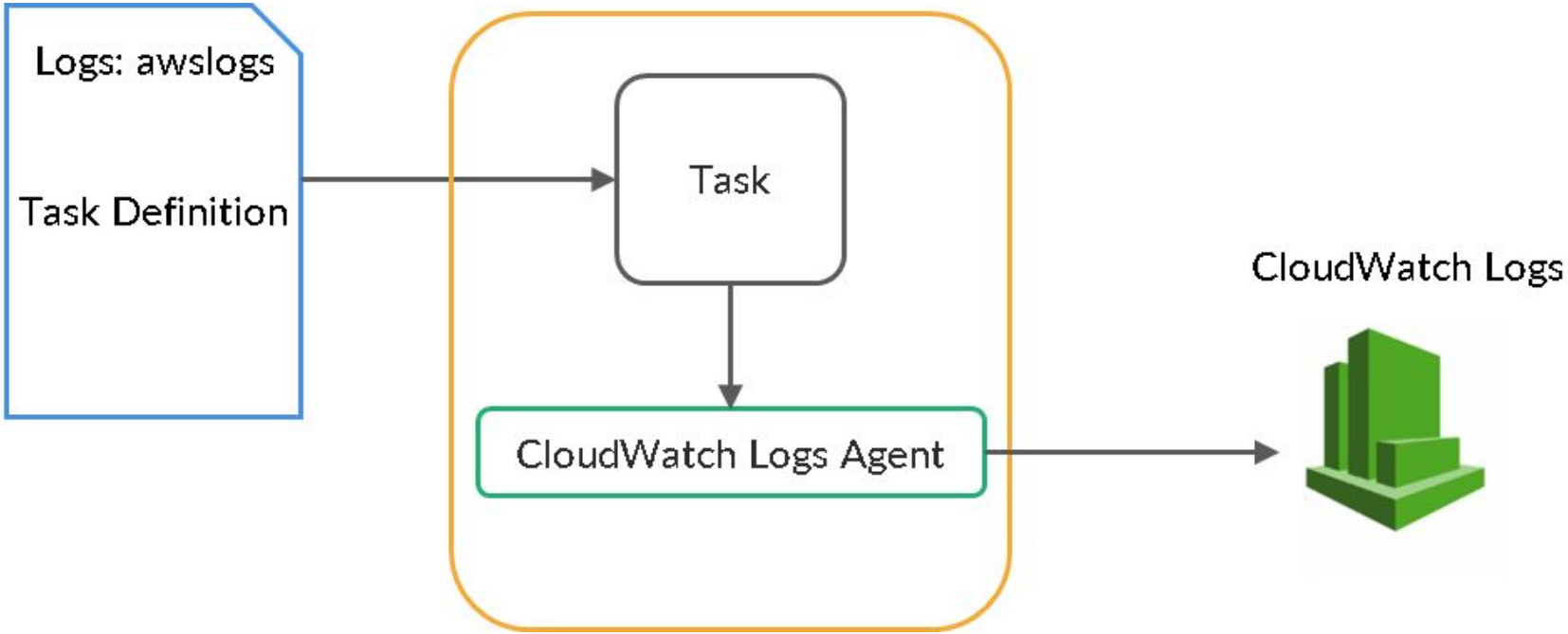
upGrad

- Access, monitor and store the log files from the containers in the AWS ECS tasks by specifying the awslogs log driver in the task definitions



ECS Logging

upGrad



ECS Service Discovery

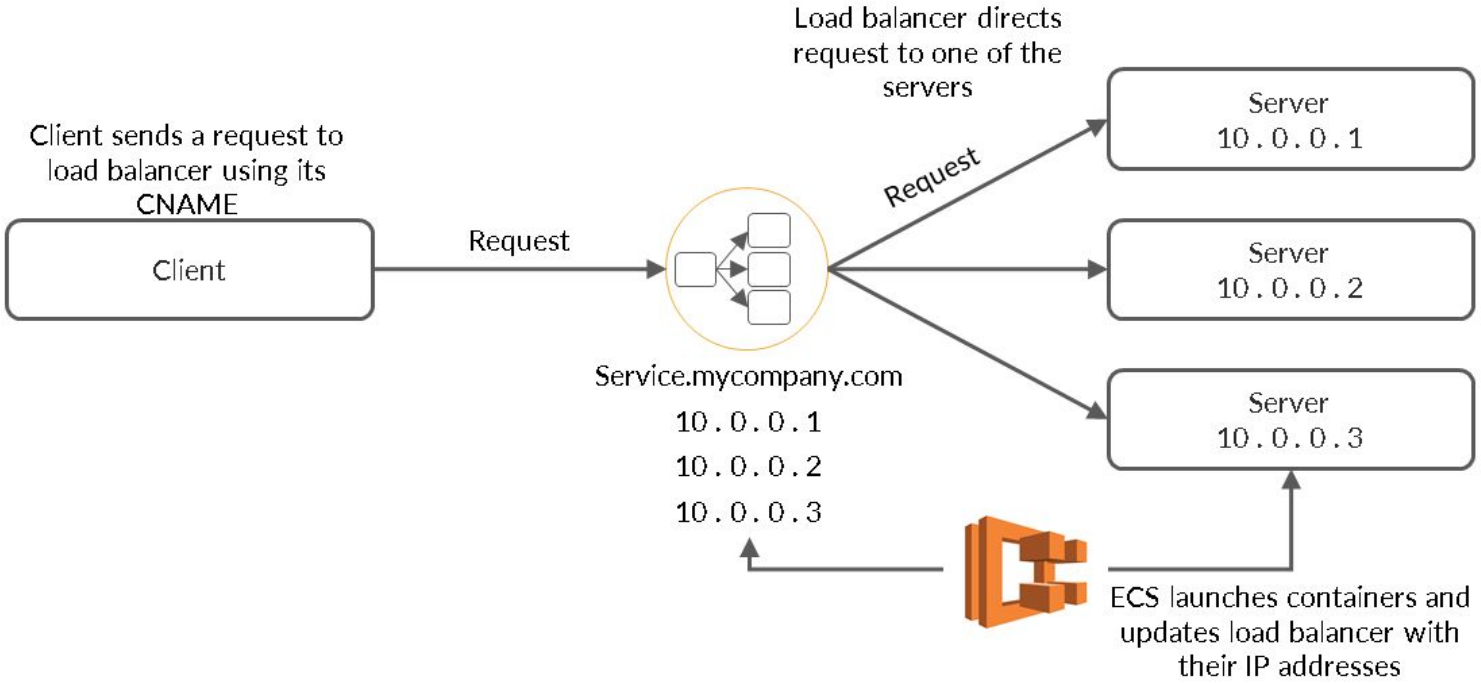
upGrad

- The services provide network connectivity to the running tasks. However, you need a mechanism so that multiple services can find and communicate with one another.
- Service discovery is the actual process of determining how to connect to a service. It keeps track of the list of backend servers so that requests from users can be distributed across these backend servers.
- Service discovery maintains a database with information regarding which service is running on which host and on which port.
- In ECS, you can achieve service discovery using:
  - Elastic Load Balancers and
  - Route53.

ECS Service Discovery: ELB

upGrad

- One of the traditional and most commonly used models for service discovery is via a load balancer.



ECS Service Discovery: ELB

upGrad

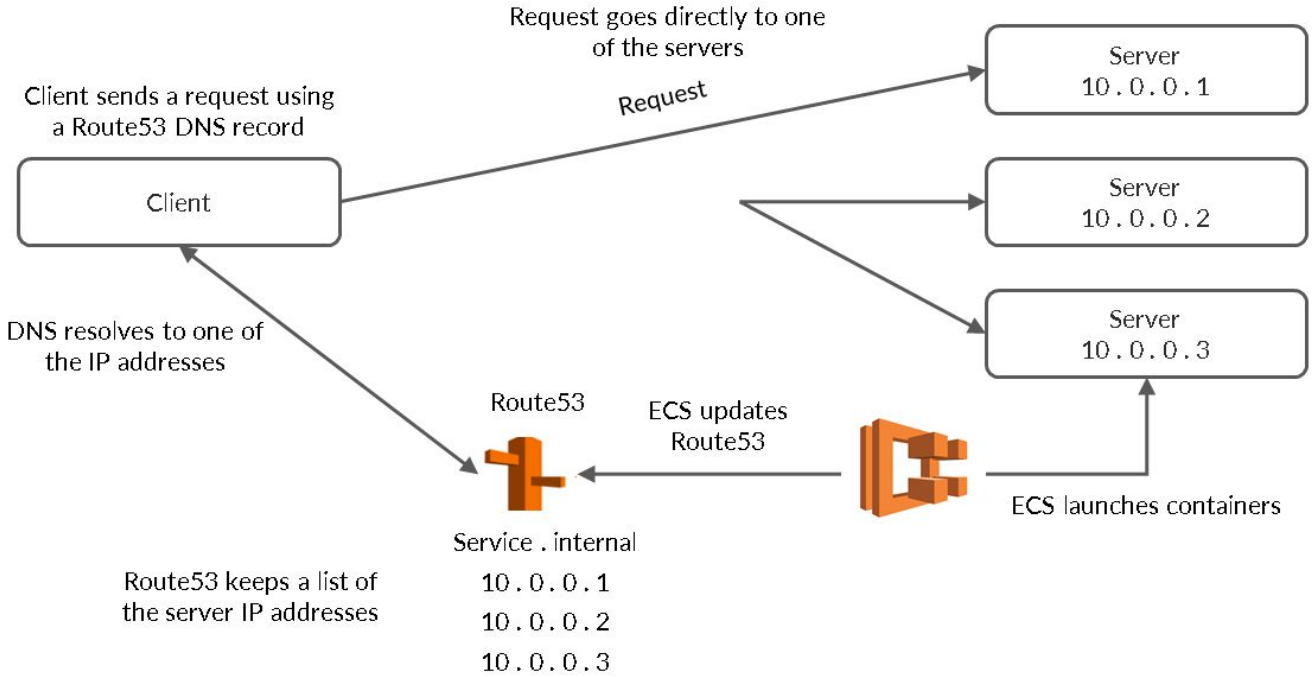
- As each container starts and becomes healthy, ECS updates the load balancer so that it knows the address of the new container.
- Users/other services send requests to the load balancer using its CNAME and the load balancer sends the requests to one of the backend servers.
- This approach is highly reliable and scalable.

- **Downsides**
  - Results in the addition of one extra hop
  - Slightly more latency as compared with direct communication from the user to the server

ECS Service Discovery: Route53

upGrad

- The Amazon ECS new service discovery works by communicating with the AWS Route53 Service Registry and autonaming APIs.



ECS Service Discovery: Route53

upGrad

- Route53 provides APIs to create: namespaces, A records per task IP, and SRV records per task IP and port.
- ECS also updates Route53 on launching new containers. So, it always has an up-to-date list of healthy containers.
- When a user performs a DNS resolution, it is served from Route53 with the same high availability and scalability that Route53 guarantees.

upGrad

DEMO 3 : ECS Fargate

- Voting Application Overview
- Launching a fargate cluster
- Task definition for five components
- Launching tasks using ECS service
- Service discovery using Route53
- Logging and Monitoring of the application
- Verifying the workflow

Important Concepts and Questions

Important Questions

upGrad

1. What is container orchestration? Why do we need it?
2. What is ECS? List some features of ECS.
3. Explain the different components of the ECS architecture.
4. What are different pricing models in ECS? Mention some use cases of each of them.
5. How can you enable logging and monitoring in ECS?
6. Explain two types of service schedulers.

Doubt Clearance Window

This class has covered the following topics:

upGrad

1. Quick recap of Container Orchestration
2. What is ECS? What are the components of ECS?
3. Concepts of ECR, EC2 and Fargate, and task definition and services
4. ECS pricing
5. Launching the Getting Started cluster
6. Launching your own Fargate cluster. Creating your own task definition and services
7. Logging and monitoring

What's Next?

upGrad

1. ECS advanced
2. Environment variables
3. ECS IAM roles
4. Service discovery
5. Update services and ECS rolling updates
6. Placement constraints
7. ECS prod-ready setup:
    a. Custom VPC, public private subnets, IGW, NAT and route tables
    b. EC2 mode ECS cluster
    c. Task definition and services
    d. Monitoring, logging, constraints and service discovery
8. Learning the KPIs for a DevOps engineer

Homework

upGrad

1. Launch a “getting started” cluster and understand the terms of the ECS architecture with the help of the Nginx image
2. Set up voting application on AWS ECS Fargate mode:
    a. Set the desired count of services
    b. Try deleting a task
    c. Scale the services
    d. Run a service in daemon mode
    e. Enable logging and monitoring

Tasks to Complete After Today's Session

MCQs
Conceptual Questions
Project Checkpoint

upGrad

#RahoAmbitious

Thank You!