AWS CONTAINERS

The Tech Stuff

Table of CONTENTS

01

Introduction

INTRODUCTION

AWS container services make it easier to manage your underlying infrastructure, whether on premises or in the cloud, so you can focus on innovation and your business needs. Nearly 80 percent of all containers in the cloud run on AWS today. Customers such as Samsung, Expedia, GoDaddy, and Snap choose to run their containers on AWS for security, reliability, and scalability.

02

Introduction to Docker

Introduction To Docker

Docker is a platform designed to make it easier to create, deploy, and run applications by using containers. Containers allow developers to package an application with all its parts, including libraries and other dependencies, and ship it all out as one package. This ensures that the application will run on any other Docker-hosted environment, regardless of any customized settings that the machine might have that could differ from the machine used for writing and testing the code.

03

Key Concepts

Key CONCEPTS

01

Containers

Lightweight, standalone, executable packages that include everything needed to run a piece of software, including the code, runtime, libraries, and settings.

02

Docker Engine

The core component of Docker, which is a client-server application with three main components: a server, a REST API, and a command-line interface (CLI).

03

Image

Read-only templates used to create containers. Images are built from a set of instructions written in a Dockerfile.

04

Docker File

A text document that contains all the commands a user could call on the command line to assemble an image.

04

Docker vs. VMs

Architecture

Docker

- Runs on Docker Engine
- Shares OS kernel
- Lower resource usage

VM

- Runs on hypervisor
- Includes guest OS and virtual hardware
- Higher resource usage

Performance

Docker

- Faster to start
- Lower resource consumption

VM

- Slower to start
- Higher resource consumption

Portability and Isolation

Docker

- Highly portable across environments with Docker
- Process-level isolation, shared kernel

VM

- Less portable, tied to hypervisor and hardware
- Strong isolation (own OS)

05

Docker in AWS

AWS

CONTAINERS

AWS ECS

AWS EKS

AWS Fargate

01) AWS ECS

A fully managed container orchestration service that simplifies running, stopping, and managing containers on a cluster.

Key FEATURES

- Supports Docker containers.
- Can run on Amazon EC2 instances or AWS Fargate.
- Integration with AWS services like IAM, CloudWatch, and Route 53.
- Load balancing with (ELB).
- Automatic scaling with ECS Service Auto Scaling.

02) AWS EKS

A managed Kubernetes service that makes it easy to run Kubernetes on AWS without needing to manage the control plane.

Key FEATURES

- Fully compatible with standard Kubernetes.
- Automatically manages the availability and scalability of the Kubernetes control plane.
- Integration with AWS services like IAM, CloudWatch, and Route 53.
- Supports both EC2 instances and AWS Fargate.
- Provides a high-availability architecture and automatic patching of the control plane.

03) AWS FARGATE

A serverless compute engine for containers that works with both ECS and EKS.

Key FEATURES

- Eliminates the need to manage servers or clusters.
- Automatically scales the compute capacity to meet your application's needs.
- Charges are based on the compute and memory resources you use.
- Simplifies the process of running containers by handling the underlying infrastructure.

04) AWS ECR

A fully managed Docker container registry that makes it easy to store, manage, and deploy Docker container images.

Key FEATURES

- Integrated with Amazon ECS, EKS, and AWS Lambda.
- Supports image vulnerability scanning.
- Provides fine-grained access control with AWS IAM.
- High availability and redundancy across multiple AWS Regions.
- Encrypted at rest with AWS Key Management Service (KMS).

06

ECS Launch Types

EC2
LAUNCH TYPE

Runs containers on a cluster of Amazon EC2 instances. Users have control over the EC2 instances, including their configuration and management.

Fargate
LAUNCH TYPE

Runs containers without needing to manage the underlying EC2 instances. AWS handles the infrastructure management, allowing users to focus solely on their applications.

07

LB & ASG Configuration

LB Integration

Application Load Balancer

HTTP/HTTPS traffic, advanced routing

Network Load Balancer

TCP, UDP, TLS traffic, high performance

ASG Integration

Service Autoscaling

Allows users to define scaling policies that adjust the desired count of tasks within a service.

Cluster Autoscaling

Automatically adjusts the number of EC2 instances in the cluster based on the resource requirements of the tasks.

08

Data Volumes With EFS

Data Volumes with EFS

EFS provides a scalable, elastic file system for use with AWS Cloud services and on-premises resources. EFS can be used as a shared data volume for Docker containers, providing persistent storage.

- Integration with ECS: EFS can be mounted as a data volume in ECS tasks, enabling multiple containers to share the same file system.

- Benefits: EFS offers seamless scalability, low latency, and high availability, making it suitable for applications requiring shared storage.

09

Use Cases

01) Microservices Architecture:

Containers are ideal for microservices as they provide isolated environments for different services, allowing for independent deployment, scaling, and management.

02) Batch Processing:

Containers can be used to run batch processing jobs efficiently. Services like AWS Batch can manage the execution of containerized batch jobs.

03 Hybrid Applications:

AWS containers can be used to deploy hybrid applications that run across on-premises and cloud environments, ensuring consistency and scalability.

04) ML & Data Processing:

Containers provide a portable and consistent environment for data processing and machine learning workloads, ensuring reproducibility and scalability.

10

Benefits

BENEFITS

SCALABILITY

Easily scale applications to handle increased load by adding more container instances.

PORTABILITY

Containers provide a consistent environment that ensures applications run the same way regardless of where they are deployed

EFFICIENCY

Containers can improve resource utilization by running multiple isolated applications on the same host machine.

AUTOMATION

the infrastructure management is automated, allowing developers to focus on building and deploying applications.

THANK YOU!

The Tech Stuff

mayamnaizel2013@gmail.com