Academic Year: 2022-23
Semester: VIII
Class / Branch: BE- IT
Subject: Cloud Computing Lab
Name of Instructor: Yaminee Patil

Name of Student:
Student ID:
Roll No.:
Date of Submission:

Experiment No.:09

Aim: To study and Implement Containerization using Docker on AWS.

Theory:

What is containerization?

Containerization is a software deployment process that bundles an application's code with all the files and libraries it needs to run on any infrastructure. Traditionally, to run any application on your computer, you had to install the version that matched your machine's operating system. For example, you needed to install the Windows version of a software package on a Windows machine. However, with containerization, you can create a single software package, or container, that runs on all types of devices and operating systems.

What are the benefits of containerization?

Developers use containerization to build and deploy modern applications because of the following advantages.

Portability

Software developers use containerization to deploy applications in multiple environments without rewriting the program code. They build an application once and deploy it on multiple operating systems. For example, they run the same containers on Linux and Windows operating systems. Developers also upgrade legacy application code to modern versions using containers for deployment.

Scalability

Containers are lightweight software components that run efficiently. For example, a virtual machine can launch a containerized application faster because it doesn't need to boot an operating system. Therefore, software developers can easily add multiple containers for different applications on a single machine. The container cluster uses computing resources from

the same shared operating system, but one container doesn't interfere with the operation of other containers.

Fault tolerance

Software development teams use containers to build fault-tolerant applications. They use multiple containers to run microservices on the cloud. Because containerized microservices operate in isolated user spaces, a single faulty container doesn't affect the other containers. This increases the resilience and availability of the application.

Agility

Containerized applications run in isolated computing environments. Software developers can troubleshoot and change the application code without interfering with the operating system, hardware, or other application services. They can shorten software release cycles and work on updates quickly with the container model.

What are containerization use cases?

The following are some use cases of containerization.

Cloud migration

Cloud migration, or the lift-and-shift approach, is a software strategy that involves encapsulating legacy applications in containers and deploying them in a cloud computing environment. Organizations can modernize their applications without rewriting the entire software code.

Adoption of microservice architecture

Organizations seeking to build cloud applications with microservices require containerization technology. The microservice architecture is a software development approach that uses multiple, interdependent software components to deliver a functional application. Each microservice has a unique and specific function. A modern cloud application consists of multiple microservices. For example, a video streaming application might have microservices for data processing, user tracking, billing, and personalization. Containerization provides the software tool to pack microservices as deployable programs on different platforms.

IoT devices

Internet of Things (IoT) devices contain limited computing resources, making manual software updating a complex process. Containerization allows developers to deploy and update applications across IoT devices easily.

Docker is an open-source project that uses several resource-isolation features of the Linux kernel to sandbox an application, its dependencies, configuration files, and interfaces inside of an atomic unit called a container. This allows a container to run on any host with the appropriate kernel components, while shielding the application from behavioural inconsistencies due to variances in software installed on the host. Containers use operating system level virtualization compared to VMs, which use hardware level virtualization using hypervisor, which is a software or a firmware that creates and runs VMs. Multiple containers can run on a single host OS without needing a hypervisor, while being isolated from neighbouring containers. This layer of isolation allows consistency, flexibility, and portability that enable rapid software deployment and testing. There are many ways in which using containers on AWS can benefit your organization. Docker has been widely employed in use cases such as distributed applications, batch jobs, continuous deployment pipelines, and etc. The use cases for Docker continue to grow in areas like distributed data processing, machine learning, streaming media delivery and genomics.

The following examples show how AWS services can integrate with Docker:

Amazon SageMaker provides pre-built Docker Images for Deep Learning through TensorFlow and PyTorch or lets you bring your custom pre-trained models through Docker images.

Amazon EMR on Amazon EKS provides a deployment option to run open- source big data frameworks on Amazon EKS.

Bioinformatics applications for Genomics within Docker containers on Amazon ECS provide a consistent, reproducible run-time environment.

For many SaaS providers, the profile of Amazon EKS represents a good fit with their multi-tenant microservices development and architectural goals.

Conclusion: Thus, we have successfully implemented Containerization using Docker on AWS.