LU07a - Einführung in Docker Swarm

Ziele

1. Ich kann den Zweck von Docker Swarm beschreiben.
2. Ich kann die grundlegenden Komponenten von Docker Swarm aufzählen.

Docker-Swarm

Cluster and scale containerized applications

Introduction

Docker Swarm is similar Kubernetes — they both orchestrate containerized applications. Kubernetes has a lot more momentum and a more active community and ecosystem. However, Swarm is a lot easier to use and is a popular choice for many small-to-medium businesses and application deployments. Learning Swarm is a stepping-stone to learning Kubernetes.

Docker Swarm consists of two parts:

A secure cluster of Docker hosts

On the clustering front, Swarm groups one or more `Docker nodes` and lets you manage them as a `cluster`. Out-of-the-box, you get an encrypted cluster (with encrypted networks, TLS, secure cluster join tokens, and a PKI) that makes managing and rotating certificates a breeze. You can add and remove nodes without interruption.

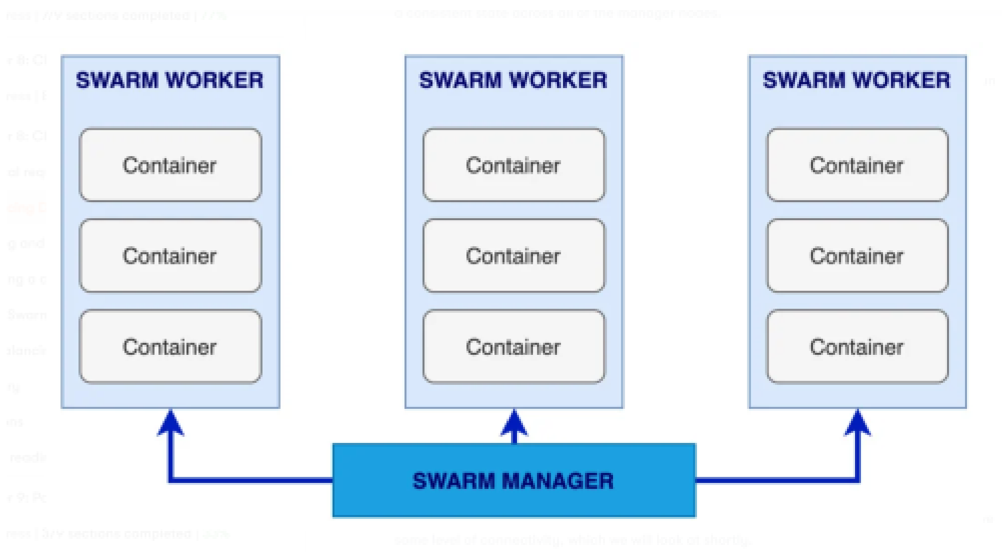
An orchestrator of microservices apps

On the `orchestrator` front, Swarm allows you to deploy and manage complex microservices apps with ease. You can define your apps in declarative files and deploy them to the swarm with native Docker commands. You can even perform rolling updates, rollbacks, and scaling operations. Again, all with simple commands.

Swarm primer

On the clustering front, a swarm consists of one or more `Docker nodes`. These nodes can be physical servers, VMs, Raspberry Pi's, or cloud instances. The only requirement is that they all have `Docker installed` and `can communicate over reliable networks`.

Nodes are configured as `managers or workers`. Managers monitor the state of the cluster and dispatch tasks to workers. Workers accept tasks from managers and execute them.



The configuration and state of the swarm is held in a distributed database replicated on all managers. It’s kept in-memory and is extremely up-to-date. However, the best thing is that it requires zero configuration — it’s installed as part of the swarm and just takes care of itself.

TLS is so tightly integrated that it’s impossible to build a swarm without it. In today’s security conscious world, things like this deserve all the plaudits they get. Swarm uses TLS to encrypt communications, authenticate nodes, and authorize roles. Automatic key rotation is also thrown in as the icing on the cake. It all happens so smoothly that you don’t even know it’s there.

On the orchestration front, the atomic unit of scheduling on a swarm is the service. This is a high-level construct that wraps some advanced features around containers. These features include scaling, rolling updates, and simple rollbacks. It’s useful to think of a service as an enhanced container.