Docker - Swarm

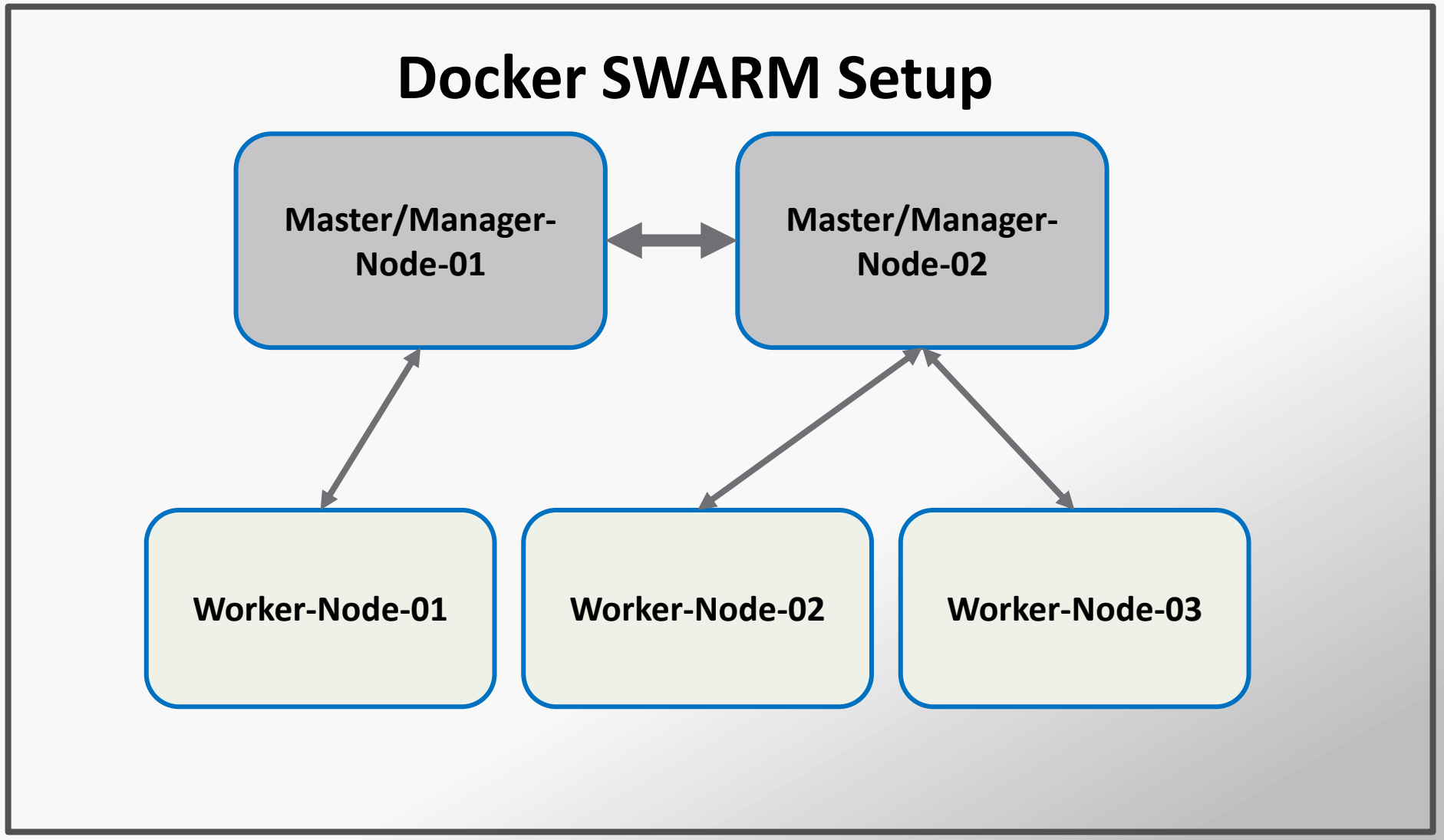
VISHWANATH M S

VISHWACLOUDLAB.COM

Feature highlights

- **Cluster management integrated with Docker Engine** → You don't need additional orchestration software to create or manage a swarm
- **Decentralized design** → You can deploy both kinds of nodes, managers and workers, using the Docker Engine. This means you can build an entire swarm from a single disk image
- **Declarative service model** → Lets you create the complete stack of the application as a bundle.
- **Scaling** → For each service, you can declare the number of tasks you want to run

Architecture of Docker Swarm



Feature highlights

- **Service discovery** → Swarm manager nodes assign each service in the swarm a unique DNS name and load balances running containers
- **Load balancing** → You can expose the ports for services to an external load balancer
- **Rolling updates** → At rollout time you can apply service updates to nodes incrementally
- **Secure by default** → Each node in the swarm enforces TLS mutual authentication and encryption

- NODE == DOCKER ENGINE == VM

Concepts of Docker Swarm

- **SWARM** → The cluster management and orchestration features embedded in the Docker Engine are built using ***swarmkit.***
  - ***A docker host can be a Manager or Worker or BOTH.***
  - Modify a service's configuration, including the networks & volumes it is connect , without the need to manually restart the service.
- **NODE** → A instance of the Docker engine participating in the swarm.
  - To deploy your application to a swarm, you submit a service definition to a **manager node.**
  - **Manager node** then release units of work called ***tasks*** to worker node**.**

Concepts of Docker Swarm

- **SERVICES and TASKS** → A ***service*** is the definition of the tasks to execute on the manager or worker nodes
  - A ***task*** carries a Docker container and the commands to run inside the container.
- **Load balancing** → The swarm manager uses **ingress load balancing** to expose the services.
  - The swarm manager can automatically assign the service a **PublishedPort in the range 30000- 32767.**
  - The swarm manager uses **internal load balancing** to distribute requests among services within the cluster based upon the DNS name of the service.

Basic Docker Swarm Commands

- To Initialize the first **Master/Manager** node as Docker SWARM Leader
  - **docker swarm init –advertise-addr <*ip add of Manager node*>**
- To view the current state of the swarm.
  - **docker info**
- To view information about nodes
  - **docker node ls**
- To get join command on manager node
  - **docker swarm join-token worker** – for adding worker nodes
  - **docker swarm join-token manager** – for adding Manager nodes

Some More commands …

- Deploy a service to the swarm
- **docker service create --replicas 1 --name helloworld alpine ping docker.com**
  - The ***docker service create*** command creates the service.
  - The ***--name*** flag names the service **helloworld**.
  - The ***--replicas*** flag specifies the desired state of 1 running instance.
  - The arguments ***alpine ping docker.com*** define the service as an Alpine Linux container that executes the command ***ping docker.com***.

Commands Continued…

- **docker service inspect** → Inspect a service on the swarm
- The ***docker service create*** command creates the service.
  - The ***--name*** flag names the service **helloworld**.
  - The ***--replicas*** flag specifies the desired state of 1 running instance.
  - The arguments ***alpine ping docker.com*** define the service as an Alpine Linux container that executes the command ***ping docker.com***.