Swarm mode overview

Estimated reading time: 3 minutes

To use Docker in swarm mode, install Docker. See installation instructions (https://docs.docker.com/install/) for all operating systems and platforms.

Current versions of Docker include *swarm mode* for natively managing a cluster of Docker Engines called a *swarm*. Use the Docker CLI to create a swarm, deploy application services to a swarm, and manage swarm behavior.

Feature highlights

- Cluster management integrated with Docker Engine: Use the Docker Engine CLI to create a swarm of Docker Engines where you can deploy application services. You don't need additional orchestration software to create or manage a swarm.
- Decentralized design: Instead of handling differentiation between node roles at deployment time, the Docker Engine handles any specialization at runtime. 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: Docker Engine uses a declarative approach to let you define the desired state of the various services in your application stack. For example, you might describe an application comprised of a web front end service with message queueing services and a database backend.
- Scaling: For each service, you can declare the number of tasks you want to run. When you scale up or down, the swarm manager automatically adapts by adding or removing tasks to maintain the desired state.
- Desired state reconciliation: The swarm manager node constantly monitors the cluster state and reconciles any differences between the actual state and your expressed desired state. For example, if you set up a service to run 10 replicas of a container, and a worker machine hosting two of those replicas crashes, the manager creates two new replicas to replace the replicas that crashed. The swarm manager assigns the new replicas to workers that are running and available.

- Multi-host networking: You can specify an overlay network for your services. The swarm manager automatically assigns addresses to the containers on the overlay network when it initializes or updates the application.
- Service discovery: Swarm manager nodes assign each service in the swarm a unique DNS name and load balances running containers. You can query every container running in the swarm through a DNS server embedded in the swarm.
- Load balancing: You can expose the ports for services to an external load balancer. Internally, the swarm lets you specify how to distribute service containers between nodes.
- Secure by default: Each node in the swarm enforces TLS mutual authentication and encryption to secure communications between itself and all other nodes. You have the option to use self-signed root certificates or certificates from a custom root CA.
- Rolling updates: At rollout time you can apply service updates to nodes incrementally. The swarm manager lets you control the delay between service deployment to different sets of nodes. If anything goes wrong, you can roll-back a task to a previous version of the service.

What's next?

Swarm mode key concepts and tutorial

- Learn swarm mode key concepts (https://docs.docker.com/engine/swarm/key-concepts/).
- Get started with the Swarm mode tutorial (https://docs.docker.com/engine/swarm/swarm-tutorial/).

Swarm mode CLI commands

Explore swarm mode CLI commands

- swarm init (https://docs.docker.com/engine/reference/commandline/swarm_init/)
- swarm join (https://docs.docker.com/engine/reference/commandline/swarm_join/)
- service create (https://docs.docker.com/engine/reference/commandline/service_create/)
- service inspect (https://docs.docker.com/engine/reference/commandline/service_inspect/)
- service ls (https://docs.docker.com/engine/reference/commandline/service_ls/)
- service rm (https://docs.docker.com/engine/reference/commandline/service_rm/)
- service scale (https://docs.docker.com/engine/reference/commandline/service_scale/)