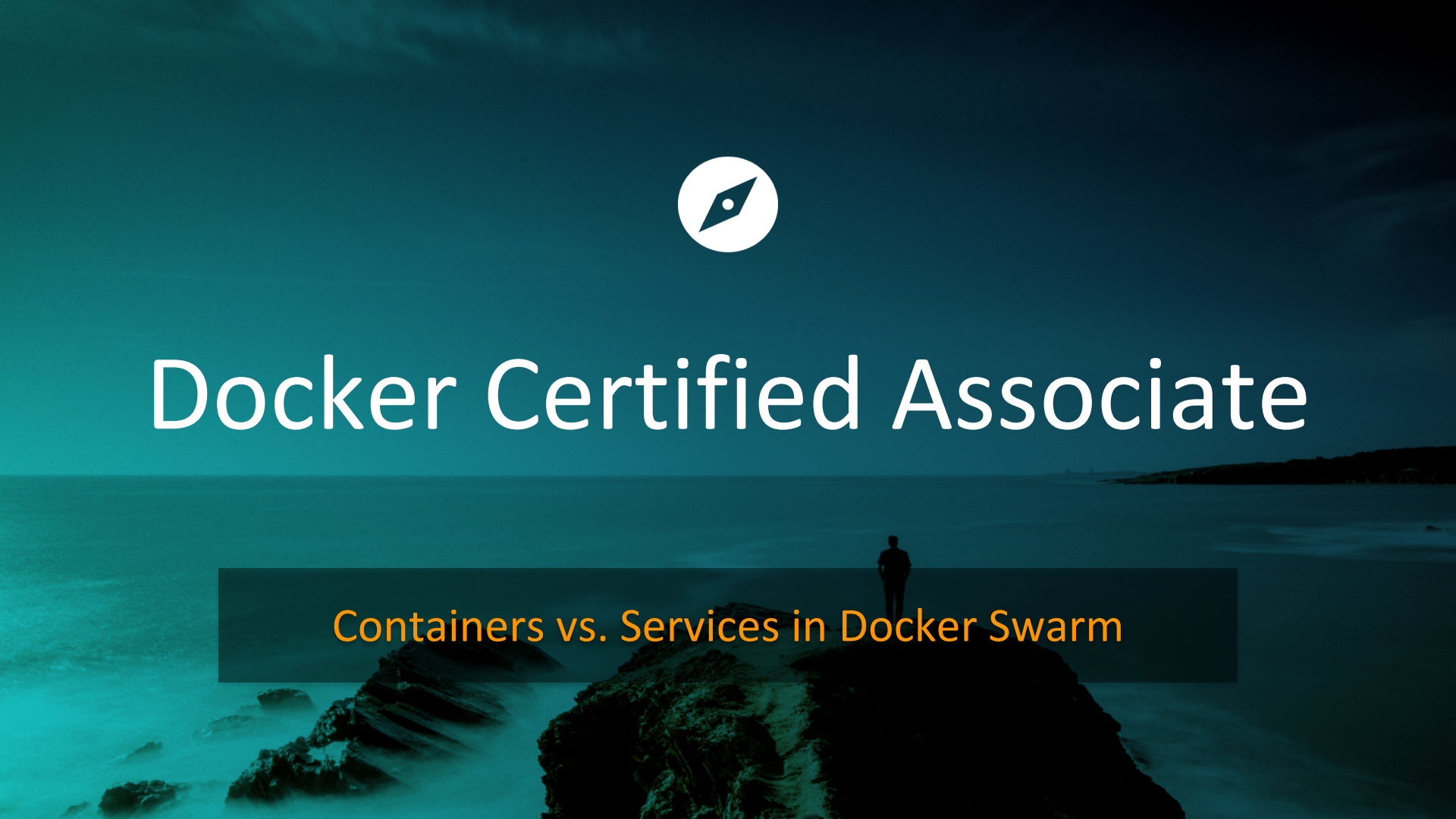


Docker Certified Associate

Containers vs. Services in Docker Swarm

Containers – docker run

The ‘docker run’ command was originally the container equivalent of ‘the face that launched a thousand ships’. It is responsible for the container revolution that we are in now, but times are changing.

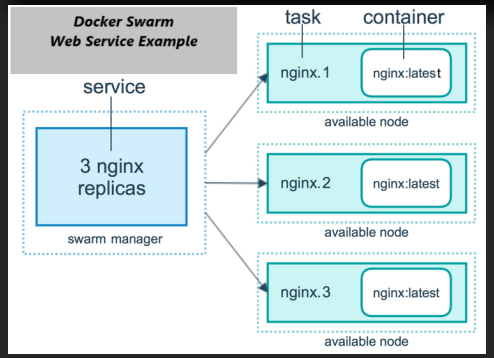
Although containers give us the flexibility, portability, granularity, and abstraction to get the most out of our environments and deployments, it is quickly becoming ‘too limited’.

We need an easier way to deploy complex configurations in highly available and easily scalable implementations. This requires the development of cluster management and control software (like Docker Swarm or Kubernetes) to work directly with Docker containers. As a result, a new paradigm is needed to address the requirements of highly scalable, clustered container environments.

Services – docker service

Using ‘docker service’ is the solution to managing containers deployed in highly available, easily scalable cluster implementations.

We can now think of a service as something that is consumed within the context of a larger application (which can include other Docker Services in its makeup).



Whereas containers are limited to the single host they are started on; services are containers that live on a scalable number of ‘workers’ in a cluster of systems. Docker Swarm handles access to, and the availability of, that service across those worker nodes, eliminating the challenges of routing and accessing individual containers.

Scalability is key in the enterprise, both up and down, in order to maximize your infrastructure spend. The scalability services allow you to take granular control of your CPU, Memory, Disk, Network, and more.

Quick Hits

Containers

- Encapsulate an application or function.
- Run on a single host at a time.
- Require manual steps to expose functionality outside of the host system (ports, network and/or volumes).
- Require more complex configuration to use multiple instances (proxies for example).
- Not highly available.
- Not easily scalable.

Services

- Encapsulate an application or function.
- Can run on ‘1 to n’ nodes at any time.
- Functionality is easily accessed using features like ‘routing mesh’ outside the worker nodes.
- Multiple instances are set to launch in a single command.
- Highly available clusters are available.
- Easily scalable (up or down as needed).