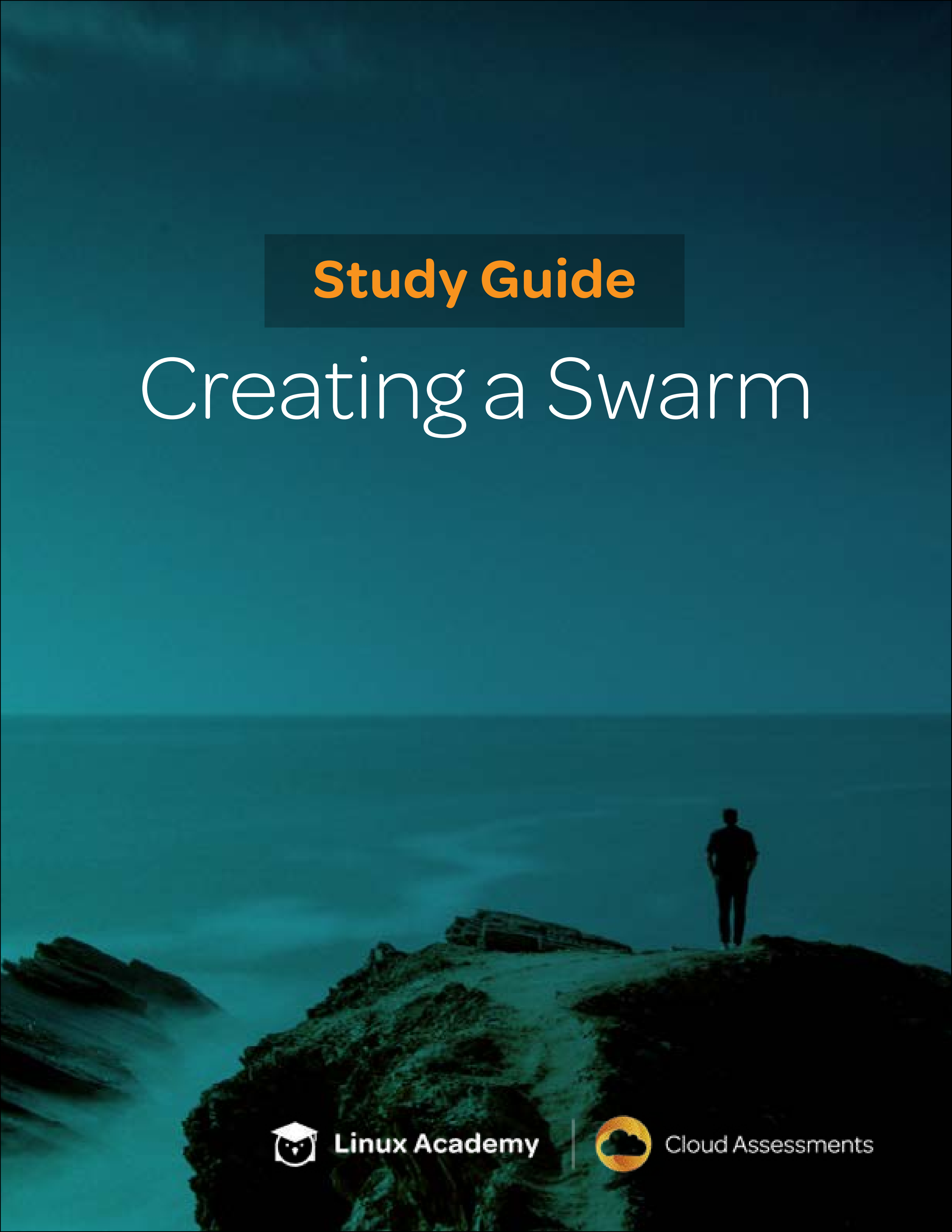

Study Guide

Creating a Swarm

Linux Academy | Cloud Assessments

Contents

The notes below accompany the Linux Academy Docker Certified Associate Prep Course videos:

- Setting up Swarm (Configure Managers)
- Setting up Swarm (Add Nodes)
- Setting up Swarm(Backup and Restore)

Configure Managers

1. Use the IP address of your server while configuring your manager:

```
docker swarm init -advertise-addr 172.31.21.64
```

2. We can copy the output to another file or safe location so we may have easy access to the token we'll need to join additional nodes to the swarm.

3. To avoid any confusion, for this example, we have changed the name of the manager server to \manager1\\:

```
[user@manager1 ~]$ docker swarm init -advertise-addr 172.31.21.64
Swarm initialized: current node (6h59lkua4alffyoneglx6l427) is now
a manager.
```

4. To add a worker to this swarm, run the following command:

```
docker swarm join -token SWMTKN-1-40xcrb7c92mjnapxvoulp15zkky0zopk
7u8cc6yd9vglya12v6-epqvj5m5859m05fry5x1p02dg 172.31.21.64:2377
```

5. However, we can request the token be provided again:

```
[user@manager1 ~]$ docker swarm join-token worker
```

6. To add a worker to this swarm, run the following command:

```
docker swarm join -token SWMTKN-1-40xcrb7c92mjnapxvoulp15zkky0zopk
7u8cc6yd9vglya12v6-epqvj5m58b5m05fry5x1p02dg 172.31.21.64:2377
```

7. To add additional managers:

```
docker swarm join-token manager
```

8. To see what nodes are currently configured:

```
docker node ls
```

Adding Nodes

We will need a second server, which we'll refer to as worker1, with Docker installed.

Complete the following:

1. Using the token provided by the steps above, join worker1 to the swarm. Do not copy and paste the command below as the token will differ:

```
[user@worker1 ~]$ docker swarm join -token SWMTKN-1-40xcrb7c92mj
napxvoulp15zkky0zopk7u8cc6yd9vglya12v6-epqvj5m5859m05fry5x1p02dg
172.31.21.64:2377 This node joined a swarm as a worker.
```

2. Repeat the previous step on all of the workers that need configuration.

Backup and Restore On the Manager

Complete the following:

1. Create a service of webserver instances:

```
docker service create -name backupweb -publish 80:80 httpd -
replicas 2
```

2. Confirm service creation:

```
docker service ls
```

3. You can see what node the containers are running in with docker service ps:

```
[user@manager1 ~]$ docker service ps backupweb ID NAME IMAGE NODE
DESIRED STATE CURRENT STATE ERROR PORTS r5h3eovajscx backupweb.1
httpd:latest worker1 Running Running 27 seconds ago pk05dezy13v2
backupweb.2 httpd:latest manager1 Running Running 27 seconds ago
```

4. You can see here that one container is running on worker1 and the other on manager1.

Test Our Backup Service Using the root User

Inside of the /var/lib/docker/swarm are files that contain information related to our docker swarm. All of these items need to be backed up.

To test the service:

1. Stop the Docker service: systemctl stop docker
2. Make a directory to copy these files in to: mkdir /root/swarm
3. Copy swarm files over to our new directory: cp -rf /var/lib/docker/swarm /root/swarm
4. Start the docker service again: systemctl start docker
5. Confirm we have 2 replicas running: docker service ls
6. Let's create a backup: tar cvf swarm.tar swarm
7. Spin up another server with Docker installed. Do not add it to the swarm!
8. Copy our files over to the new server: scp swarm.tar user@
9. Now stop docker on all of our nodes (this is to mimic a swarm crash).

Recovery Using the root User

1. On the new server, remove the swarm files currently located at /var/lib/docker as we will be using this server as a recovery platform: rm -rf /var/lib/docker/swarm/

2. Make a temporary directory and untar our file: mkdir tmp && cd tmp

3. Untar the swar.tar file: tar xvf ../swarm.tar

4. Move the files over: mv /home/user/tmp/swarm /var/lib/docker

5. Start the Docker service: systemctl start docker

6. As soon as Docker starts, initialize a new cluster: docker swarm init –force-new-cluster