Building Container Images

Relevant Documentation

- Install Docker Engine on Ubuntu
- docker build
- Dockerfile reference

Exam Tips

- Images are files that include all of the software needed to run a container.
- A Dockerfile defines the contents of an image.
- The `docker build` command builds an image using a Dockerfile.

Lesson Reference

On the **control plane node**, install Docker.

Create a `docker` group. Users in this group will have permission to use Docker on the system:

```
sudo groupadd docker
```

Install required packages.

Note: Some of these packages may already be present on the system, but including them here will not cause any problems:

```
sudo apt-get update && sudo apt-get install -y apt-transport-https ca-certificates curl gnupg lsb-release
```

Set up the Docker GPG key and package repository:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg \
| sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Install the Docker Engine:

```
sudo apt-get update && sudo apt-get install -y docker-ce docker-ce-cli
```

Test the Docker setup:

```
sudo docker version
```

Add `cloud_user` to the `docker` group in order to give `cloud_user` access to use Docker:

```
sudo usermod -aG docker cloud_user
```

Log out of the server and log back in.

Test your setup:

```
docker version
```

Create a directory to house your Docker project.

```
mkdir my-website

cd my-website
```

Create a simple homepage for your website.

```
vi index.html
```

Add some basic content to the html file.

```
Hello, World!
```

Create a Dockerfile.

```
vi Dockerfile
```

This will create an image with Nginx and include our html file as the default homepage.

```
FROM nginx:stable

COPY index.html /usr/share/nginx/html/
```

Build the image.

```
docker build -t my-website:0.0.1 .
```

Test the image.

```
docker run --rm --name my-website -d -p 8080:80 my-website:0.0.1

curl localhost:8080
```

Clean up the container.

```
docker stop my-website
```

Save the image to an archive.

```
docker save -o /home/cloud_user/my-website_0.0.1.tar my-website:0.0.1
```

View the archived image file.

```
ls /home/cloud_user
```