Docker Volumes

Virtualisation

Otago Polytechnic
Dunedin, New Zealand

Union File Systems

- Files in a Docker container are presented in a *Union File System*
- An image is stored as a set of read-only layers.
- Only the very top layer in a running container is read-write.
- When we modify a file from an image, we copy it from a lower, read-only layer into the read-write layer.
- The modified copy saved on the top layer masks the unmodified version on the lower layer.
- When we destroy a container, the read-write layer associated with it is destroyed and our changes are lost.

Docker Volumes

- Sometimes we want to persist data created inside a container, or we want to share data between containers.
- A Docker *volume* lets us do this.
- Volumes are files or directories that sit outside the union file system and are saved directly on the host file system.
- Other containers can access volumes, and we can delete a container without deleting its volumes.

Creating Volumes

We can create volumes from the command line:

```
sudo docker run -it --name vol-test -v /data ubuntu /bin/bash
```

Creating Volumes

We can create volumes from the in a Dockerfile:

```
FROM ubuntu:14.04
VOLUME /data
```

Creating Volumes

We can mount volumes that are associated with another container:

```
docker run -it --name vol-test-2
--volumes-from vol-test
ubuntu /bin/bash
```

Note that

- The container `vol-test` *doesn't have to be running*.
- Changes to the volume are immediately visibile in both containers (and the host).

Creating Volumes

We can create volumes from directories that already exist on the host:

```
sudo docker run -it --name vol-test3
-v /home/tclark/data:/data
ubuntu /bin/bash
```

Data-only containers

One application of containers with volumes is creating containers with the express purpose of establishing data volumes that are accessed by other containers. These are called *data-only containers* and they are described at
`http://container42.com/2014/11/18/data-only-container-madness/`