Getting Started with Docker

•••

Workshop

Antonis Kalipetis

@akalipetis

CTO @ SourceLair

Docker Captain and big fan

Python enthusiast

Coffee lover

Agenda

- Docker basics
  - Running containers
  - Building images
  - Linking and exposing containers
- Docker stacks
  - Networks
    - Understanding Docker networks and DNS
  - Volumes
  - Docker compose
    - Creating services
    - Running complex stacks
- QnA

git clone https://github.com/akalipetis/docker-workshop

Access your Docker engine

SSH

- ssh workshop@<engine-id>.akalipetis.com
- Use the provided password
- Run docker ps to check you have access

Remote Docker CLI

- Download www.akalipeits.com/<engine-id>.zip
- Extract (use the provided password)
- Configure the environment:
  - export DOCKER_TLS_VERIFY="1"
  - export DOCKER_HOST="tcp://<engine-id>.akalipetis.com:2376"
  - export DOCKER_CERT_PATH="<path-to-certs>"

Docker basics

Running your first container

```
docker run -it ubuntu bash

# apt-get update &&\

  apt-get install -y net-tools
```

What did just happen

- Docker created a new process
- Isolated the process using namespaced (net, mnt, pid, etc)
- Mounted the given filesystem - in our case Ubuntu
- Attached to this process

Let’s build our first image

```
$ cd containers/net-tools

$ docker build -t net-tools .

$ docker run net-tools ip a
```

While images seem large, they're actually pretty small as they use CoW filesystems.

Listing and inspecting

```
$ docker ps

$ docker ps -a

$ docker inspect a047e3d0ae2b

$ docker ps -a -f exited=0
```

Linking containers

Docker bridges all created veths to docker0, so it can do smart forwarding

```
$ docker run -d redis

$ docker run -it --link=redis:redis redis bash

$ docker run -ti --link=redis:some-redis redis redis-cli -h some-redis
```

Exposing containers

With the bridge, it can forward packets received to ports in the host to the correct veth

```
$ docker run -d -p 80 nginx

$ cd containers/static

$ make build

$ docker run -d -p 80:80 static
```

Docker handles networking from, to and within containers, using docker0 and a veth pair per container

Getting logs

```
$ docker logs a047e3d0ae2b

$ docker attach a047e3d0ae2b
```

Docker stacks

Networks

Create private networks across hosts and connect containers to them

```
$ docker network create demo

$ docker run -d --net=demo --net-alias=redis redis

$ docker network connect --alias=redis demo redis
```

Volumes

Create data volumes, which persist even after a container gets deleted

```
$ docker volume create myvol

$ docker run -v=myvol:/mnt/data ubuntu echo hello > /mnt/data/hey.txt

$ docker run -v=myvol:/mnt/data ubuntu cat /mnt/data/hey.txt

$ docker run --volumes-from=some-container -it ubuntu bash
```

Docker sports a plugin system which allows third parties to provide network or volume drivers for other systems

Compose

Compose stacks, without needing to start all containers all the times

```
$ cd compose/elk

$ docker-compose up
```

Compose

Interfere with containers created by Docker compose

```
$ cd compose/django-postgres

$ docker-compose up

$ docker-compose exec django ./manage.py migrate
```

Docker compose is a tool for defining and running multi-container Docker applications

QnA

Let’s discuss about Docker in General

Thanks!

Antonis Kalipetis

@akalipetis

https://www.sourcelair.com