

<http://neyto.blogspot.co.uk/?view=magazine> – summary for docker commands

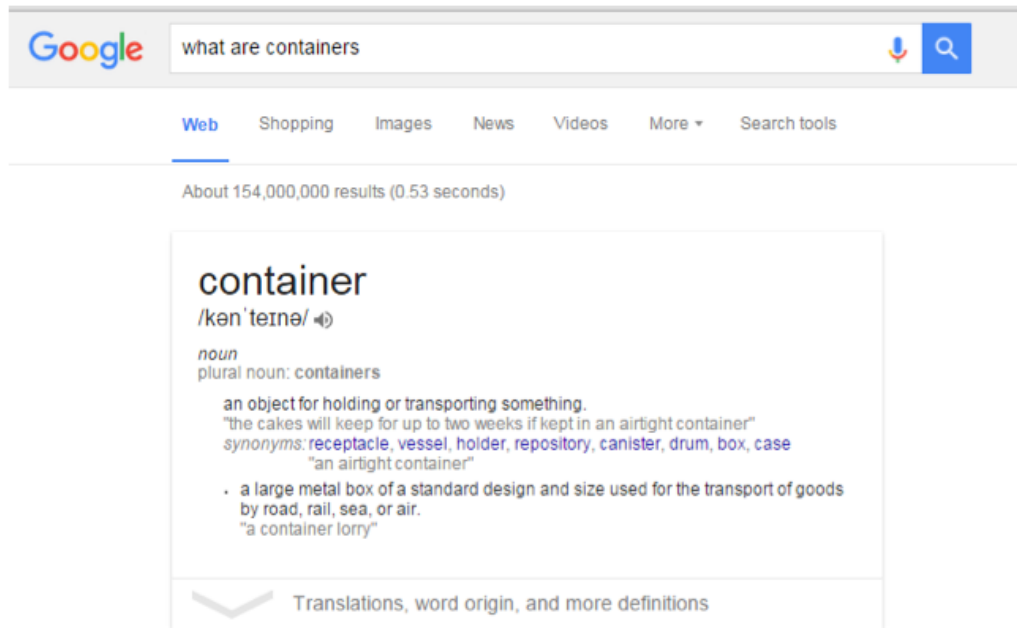
Outline

- **Containerisation**
 - What is containerisation
 - What are its advantages
 - Different options
- **Introducing Docker**
 - How does Docker work
 - Advantages to Docker
- **Using Docker**
 - Install
 - Start a new container
 - Stop a container
 - Connect to the terminal
 - Logs

Objective

- **By the end of this session you should be able to**
 - Discuss what containers are and what Docker is
 - Install Docker
 - Start up and connect to some containers

What are Containers?



4

It does not matter what is in the container, the container is always the same size and shape

It does not matter what is in the container - loading and unloading process is the same

What is Containerisation

Containerization is a lightweight alternative to full machine virtualization that involves encapsulating an application in a container with its own operating environment. This provides many of the benefits of loading an application onto a virtual machine, as the application can be run on any suitable physical machine without any worries about dependencies. – Webopedia

- **What is a virtual machine?**
 - Simulates hardware and software
 - Separate from the host OS
- **Containers are similar, but they are 'light weight'**
- **Containerisation is not a new idea**
 - Dates back to at least 2000
 - LXC – Linux containers have existed for a while

5

<http://www.webopedia.com/TERM/C/containerization.html> - Link accessed 16/6/2015

Different options

- **Docker**
 - Very popular container creator / launcher
 - Able to integrate with many other tools (Jenkins, Puppet to name a few!)
 - 'De facto' standard
- **rkt (Rocket)**
 - CoreOS product
 - Based on the idea of only doing containers (claims Docker has too big and lost its single focus)
 - Can run Docker containers
 - Security focus
- **Linux Containers**
 - LXD containerisation for OpenStack

6

Docker is not only one option for the containerization (there are some different options):

Docker is popular because of the linux containers. It was done many years ago as apart of Linux OS

Docker has taken this idea and gived nice front end. And it giver ability to integrate with many tools out there.

It becomes 'de facto' standard in the world

Main alternative is Rocket:

It the CoreOS project

Based on the idea of only doing containers

It based on the security (base principle)

Its doing only single thing and it dos it very well

Open container initiative has meant that all containerisation engines has sign up to use the same standards.

In the future you will be able to export docker into rocket and vice versa, rocket to windows etc.

Linux Containers

- **Part of the LXC namespace**
 - Low level
 - Allows for containers to be built
 - Layered systems – can have links to host file system
- **Allow for code to be run in self contained places**
 - Can't interfere with the outside operating system
 - Only have access to what is given
- **Light weight**
 - Not hosting the entire operating system
 - Can share common files between host and guest

7

See <https://linuxcontainers.org/> for more information

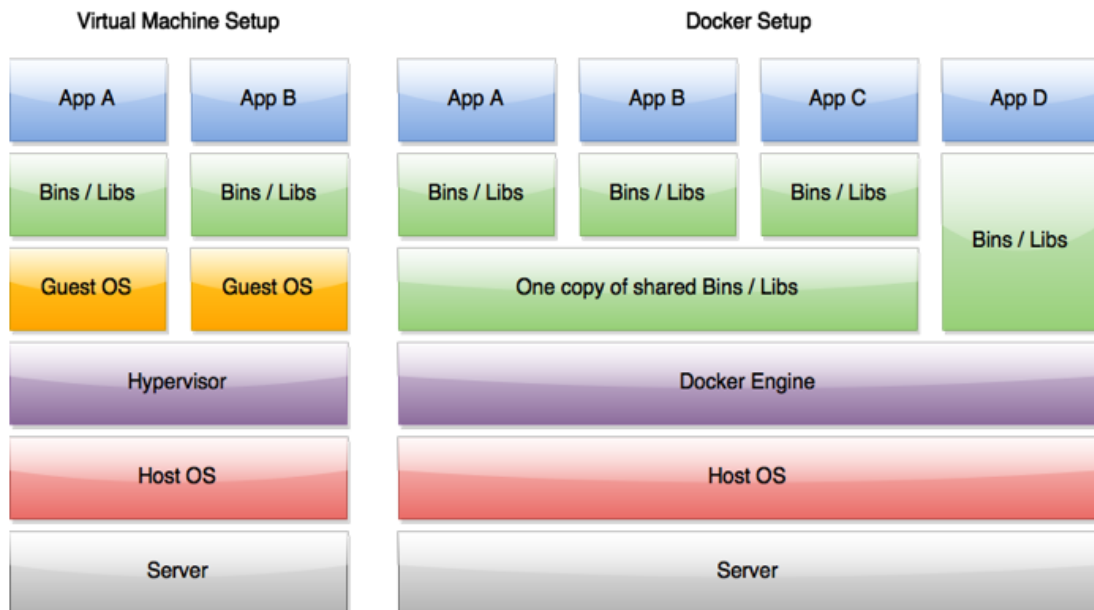
We can run any kind of software, in a self contained environments with its own set of info, own set of resources and it only has access to what ever is given. Your code can cross over something else, it cant to brake it out of this 'jail' what has been given.

Light weight - Its not a virtual machine. On VM you need to simulate your hardware, memory etc.

Containers are only concerned about files. Its all about the file system being changed. Concerns only changes from one file system to the next.

You can share info as long as your file are the same and keep it secure.

Docker Containers



Containers vs VMs

- | | |
|---|--|
| <ul style="list-style-type: none">▪ Virtual machines need to emulate an entire operating system | <ul style="list-style-type: none">▪ Containers are made up of layers, only the changes are stored in each layer |
| <ul style="list-style-type: none">▪ Complete separation between machines | <ul style="list-style-type: none">▪ Shared resources between containers where possible |
| <ul style="list-style-type: none">▪ Better security – enforced from creation | <ul style="list-style-type: none">▪ Security isolation is possible |
| <ul style="list-style-type: none">▪ 1 Gig machine x 1000 instances<ul style="list-style-type: none">▪ 1000 Gig | <ul style="list-style-type: none">▪ 1 Gig machine x 1000 instances<ul style="list-style-type: none">▪ 1 Gig |

9

<http://www.slideshare.net/Flux7Labs/performance-of-docker-vs-vm> has some details comparing the numbers between the two systems

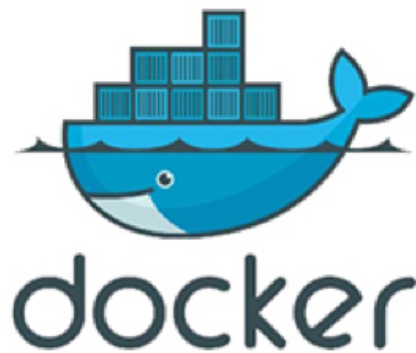
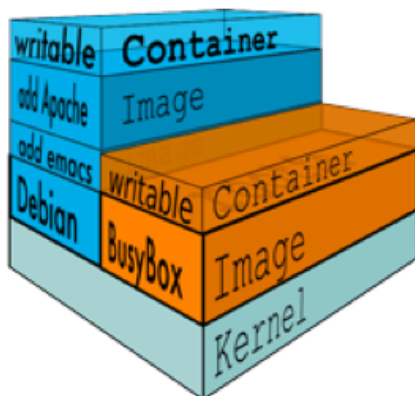
Light weight - Its not a virtual machine. On VM you need to simulate your hardware, memory etc.

Containers are only concerned about files. Its all about the file system being changed. Concerns only changes from one file system to the next.

You can share info as long as your file are the same and keep it secure.

What is Docker?

- “A person employed in a port to load and unload ships.”
- “Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in.”



Introducing Docker

- **Free! Open Source!**
- **Designed to be hardware and software agnostic**
 - Can run on any platform
 - Based on Linux
 - Windows server 2016 can support Docker natively
- **Based on the idea of shipping containers**
 - Self contained items
 - Can store anything
 - We have windows which allow us to see inside and can connect containers together
 - Can be stacked, loaded, unloaded independently

11

<http://azure.microsoft.com/blog/2014/10/15/new-windows-server-containers-and-azure-support-for-docker/>

Why is it popular

- **Fast to set up and start new containers**
 - Uses caching to avoid repeated steps of compilation / setup
- **Avoids hardware emulation**
 - Less overhead compared to traditional virtual machines
- **Sharing common layers**
 - Docker images are created in layers
 - Two or more applications can share the files from the same layer
 - Only need to separate out the changes made
 - More efficient than hypervisors
- **Standardisation in containers**
 - You can build a docker container on any OS and then still run it on any other OS.

Installing Docker

- **Ubuntu**

```
$ wget -qO- https://get.docker.com/ | sh
```

- **Centos 7 & the Amazon Linux AML**

```
$ sudo yum install -y docker
```

- **Docker installs as a service so we need to remember to start it**

```
$ sudo service docker start
```

Installing Docker on Centos 6

- **You need at least kernel version 2.6.32-431**
 - This has specific fixes which allow Docker to run
 - We also need to ensure that the EPEL rpm is enabled
- **There may be an unrelated docker package already installed with Centos**

```
$ sudo yum -y remove docker
```

- **Install Docker-IO**

```
$ sudo yum install docker-io
```

- **Start the service and ensure it starts at boot**

```
$ sudo service docker start  
$ sudo chkconfig docker on
```

14

On Centos 7 we can just install “Docker” this also works on the Linux AMI in EC2

Using Docker – start container

- **All Docker commands use the main Docker binary**
 - This must be run in su mode
- **The run command will see if we have a copy of the container**
 - If not it will download one from the docker hub

```
# docker run [flags] [containername] [command to run]
# docker run -ti ubuntu bash
# docker run centos:centos6 yum update
```

- **Finding images**

```
# docker images          -- images available locally
# docker search [term]    -- search Docker hub for more
# docker pull [image name] -- pull down a copy of that image
```

15

Add docker to maven

<http://www.alexecollins.com/docker-maven-plugin/>

Interacting with running container:

1. Docker attach containerID – drops into the foreground of container but there is no way to go out (ctrl + c only) and it shuts down
2. Docker exec -it containerID command_bash – executes command_bash as a thread inside the container (-l – interactive; -t – sudo terminal); to EXIT is 'exit'
3. Docker logs containerID – shows the outputs of the container

Using Docker – Start a terminal

- **We can see the command line terminal from any container**

```
# docker run -ti ubuntu bash
```

- t flag tells docker we want a psuedo terminal
- i flag tells docker we want it to be interactive
- Sometimes written -t -i

- **We can connect to containers already running**

```
[root@localhost user]# docker run --name scalatra \
                        -d kizzie/hello-scalatra
382eccf0231a3a06ad35f9ce0619a68... etc
```

```
[root@localhost user]# docker exec -ti scalatra bash
root@382eccf0231a:/hello-scalatra#
```

16

`docker run -d -p 80:8080 jenkins`

run the container

-d – runs as detached process in the background (reather then foreground without -d)

-p map port

port of the host machine : port of inside the container (internally)

jenkins program is running

`docker ps` – shpws this image as running

`docker run -d -p 80:8080 --name team1 jenkins` - gives team1 name instead of random setted

Using Docker – Stop container

- **List all the currently running images**

```
# docker ps
```

- -l flag shows the last container run, even if it has finished

- **Stopping an image**

- We can use either the Container ID or the name
 - Only usually need the first four characters of the container id

```
# docker kill [name]
```

- **Naming an image**

```
# docker run --name db training/postgres
```

17

The training repositories are used by Docker's own tutorial set

`docker ps` - shows currently running containers (`docker ps -a` - shows all existing containers)

`Docker search jenkins` - searches for the images in dockerHub

Using Docker - Logs

- **The Docker Logs command can be used to debug containers**
 - Shows what is being generated and displayed by the terminal
 - -f flag tells it to stay alive and update
 - This is often better than having a bash terminal alive as you can accidentally close down the container

```
# docker logs [flags] [name]
# docker logs scalatra
```

Exercise

- **Install docker on your machine**
 - Search for some containers
 - Run the container
 - Run commands through those images

To read more about Docker

Books:

- Mastering Docker - Second Edition (2017. link: <https://www.safaribooksonline.com/library/view/mastering-docker-/9781787280243/>)
- Docker Orchestration (2017. link: <https://www.safaribooksonline.com/library/view/docker-orchestration/9781787122123/>)

Tutorials:

- <https://docker-curriculum.com/>
- <https://www.tutorialspoint.com/docker/>

Video:

- <https://www.youtube.com/watch?v=pGYAg7TMmp0>

Summary

- **Containerisation**
 - What is containerisation
 - What are its advantages
 - Different options
- **Introducing Docker**
 - How does Docker work
 - Why do people like Docker
- **Using Docker**
 - Install
 - Start a new container
 - Stop a container
 - Connect to the terminal
 - Logs