what's in it for me?

Ian Truslove :: Boulder Linux User Group :: 2014-03-13
First: the end...
Docker: What’s in it for me?

Devs:

http://xkcd.com/303/

DevOps:

http://xkcd.com/149/
Docker: What’s in it for me?

*Devs:*
Docker gives you a really easy way to find and use pre-packaged system components to use in conjunction with your app code.

*DevOps:*
Docker gives the Devs a straightforward way to declaratively package deployable, repeatable, self-contained system components.
About me
Context

building and deploying highly scalable app infrastructures
Methodology: The Twelve-Factor App

I. Codebase
One codebase tracked in revision control, many deploys

II. Dependencies
Explicitly declare and isolate dependencies

III. Config
Store config in the environment

IV. Backing Services
Treat backing services as attached resources

V. Build, release, run
Strictly separate build and run stages

VI. Processes
Execute the app as one or more stateless processes

VII. Port binding
Export services via port binding

VIII. Concurrency
Scale out via the process model

IX. Disposability
Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity
Keep development, staging, and production as similar as possible

XI. Logs
Treat logs as event streams

XII. Admin processes
Run admin/management tasks as one-off processes

http://12factor.net/
introducing...
About Docker

From www.docker.io/the_whole_story:

“Docker is an open-source engine that automates the deployment of any application as a lightweight, portable, self-sufficient container that will run virtually anywhere.”
Docker is a shipping container system for code

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container...

...that can be manipulated using standard operations and run consistently on virtually any hardware platform.

https://www.docker.io/static/img/about/docker_container.jpg
Docker is built on LXC

From [linuxcontainers.org](http://linuxcontainers.org):

“LXC is often considered as something in the middle between a chroot on steroids and a full fledged virtual machine. The goal of LXC is to create an environment as close as possible as a standard Linux installation but without the need for a separate kernel.”
Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries.
What does Docker provide?

Again, from [www.docker.io/the_whole_story](http://www.docker.io/the_whole_story):

“Docker runs three ways:

- as a daemon to manage LXC containers on your Linux host
- as a CLI which talks to the daemon's REST API
- as a client of Repositories that let you share what you've built”
Supported platforms

See [www.docker.io/gettingstarted/#h_installation](www.docker.io/gettingstarted/#h_installation)

**Directly:**
- Ubuntu
- Arch Linux
- Gentoo
- Fedora
- OpenSuse
- FrugalWare
- Binaries*

**Virtualized:**
- OSX
- Windows
- EC2
- Rackspace
- Google
Binary requirements

- iproute2 version 3.5 or later (build after 2012-05-21), and specifically the “ip” utility
- iptables version 1.4 or later
- The LXC utility scripts (http://lxc.sourceforge.net) version 0.8 or later
- Git version 1.7 or later
- XZ Utils 4.9 or later
Not (yet) ready for production

https://pbs.twimg.com/profile_images/1263403750/riverscuomo-mustache.jpg

heh, what do they know...
Docker ecosystem
Base image catalog

Official
- Ubuntu
- Centos 6.4
- Busybox

Unofficial
- Gentoo
- Arch
- OpenSUSE
- Debian
- etc...
# Images: 3rd party software

```bash
> docker search database
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
<th>STARS</th>
<th>OFFICIAL</th>
<th>TRUSTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>deis/database</td>
<td>Database server for the Deis open source Platform.</td>
<td>2</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>mminke/mongodb-x86-64-2.4.8</td>
<td>This is a mongo db image which downloads the data from the MongoDB server.</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>bradrydzewski.neo4j</td>
<td>Neo4j graph database running on Ubuntu Precise</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>bbytes/orientdb</td>
<td>Image with orient db. The database and base are provided by OrientDB.</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>kamui/postgresql</td>
<td>PostgreSQL 9.3 with configurable login/password</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>luisarmando/docker-postgresql</td>
<td>Postgresql 9.3 docker repository. This can be used for development</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>tpires.neo4j</td>
<td>Neo4j is a highly scalable, robust (fully functional) graph database</td>
<td>1</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>mhansen/sequelize-contribution</td>
<td>Docker image containing a MySQL and PostgreSQL database</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>skardan/cassandra</td>
<td>Cassandra Database</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>erasche/chado</td>
<td>A Chado database container. Contains a full-featured application server</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>tpires/sonar-mysql</td>
<td>SonarQube is an open platform to manage code quality</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>sminot/database</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hwuethrich/bamboo-agent</td>
<td>Easily scale your Atlassian Bamboo infrastructure</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>stephendl/releasemetrics_postgres93</td>
<td>A postgres database with and empty release</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>turnkeylinux/mysql-13.0</td>
<td>TurnKey MySQL - Relational Database Management</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>dharmamike/dc-pgsql</td>
<td>This container runs a PostgreSQL database</td>
<td>0</td>
<td>[OK]</td>
<td></td>
</tr>
<tr>
<td>&lt;truncated&gt;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Further afield...

A little searching yields:

- Container hosting: Stackdock, Orchard, dotCloud (aka “Docker, Inc”)
- Image hosting: Quay.io
- DIY PaaS: Dokku, Deis, Flynn, Tsuru, ...
Basic Usage
Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
  config.vm.box = "precise64"

  # System provisioning shell script - git, dev tools
  config.vm.provision "shell", path: "bin/provision.sh"

  # Basic docker provisioning
  config.vm.provision "docker", images: ["ubuntu"]

  #...
end
Docker basics: curl a URL

> `docker pull ubuntu`
Pulling repository ubuntu
...
Download complete

> `docker run ubuntu uname -a`
Linux 03faf5dbff1e 3.2.0-23-generic #36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012 x86_64 x86_64 x86_64 GNU/Linux

> `docker run ubuntu curl`
2014/03/07 05:15:55 Unable to locate curl
Docker basics: install software

> `docker run ubuntu apt-get update`

... get:12 http://archive.ubuntu.com precise-security/universe i386 Packages [95.9 kB]
Fetched 3068 kB in 3s (920 kB/s)
Reading package lists...

> `docker ps -l`

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
</tr>
</thead>
<tbody>
<tr>
<td>ce6f4a44dec7</td>
<td>ubuntu:12.04</td>
<td>apt-get update</td>
<td>About a minute ago</td>
</tr>
</tbody>
</table>
Docker basics: commit an image

> `docker commit ce6f iant/basics-1`

4011ace0088fe9389044505e85ed1b03e6c47d3e14856df<snip>

> `docker images`

<table>
<thead>
<tr>
<th>REPOSITORY</th>
<th>TAG</th>
<th>IMAGE ID</th>
<th>CREATED</th>
<th>VIRTUAL SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ian/basics-1</td>
<td>latest</td>
<td>4113cd2b3e07</td>
<td>6 seconds ago</td>
<td>224.5 MB</td>
</tr>
</tbody>
</table>
Docker basics: more installs

> `docker run iant/basics-1 apt-get install -y curl`
Reading package lists...
Building dependency tree...
The following extra packages will be installed:
  ca-certificates krb5-locales libasn1-8-heimdal ...

> `docker commit `docker ps -l -q` iant/basics-2`
34490e7469d5b5b24ab74e4af300ffe0f17a0a31ab8549<snip>
Docker basics: using the image

```bash
docker run iant/basics-2 curl -is www.google.com
```

HTTP/1.1 200 OK
Expires: -1
Cache-Control: private, max-age=0
X-Frame-Options: SAMEORIGIN
Alternate-Protocol: 80:quic
Transfer-Encoding: chunked

<!doctype html><html ...
Docker basics: Dockerfiles

FROM ubuntu:12.04

RUN apt-get update

# Install curl
RUN apt-get install -y curl
Docker basics: Building Dockerfiles

> `docker build -t iant/basics-3 .`
Uploading context 2.56 kB
Uploading context
Step 0 : FROM ubuntu:12.04
  ---> 9cd978db300e
Step 1 : RUN apt-get update
  ---> Running in 3f898a80705a
...

> `docker run iant/basics-3 curl -is www.google.com`
HTTP/1.1 200 OK
Expires: -1
...
...
Docker basics: publishing the image

> docker login
Username (iant):
Password: ******

> docker push iant/basics-2
The push refers to a repository [iant/basics-2] (len: 1)
Sending image list
Pushing repository iant/basics-2 (1 tags)
511136ea3c5a: Image already pushed, skipping
4011ace0088f: Image successfully pushed
34490e7469d5: Image successfully pushed
Pushing tag for rev [d55c21891493] on {https://registry-1.docker.io/v1/repositories/iant/demo2/tags/latest}
Use Cases
real and perceived
Use case: homegrown PaaS

“Build your own Heroku”

Extant tools: Deis, Tsuru, etc.

...but is this really a good idea? Perhaps for internal use.
Baidu

Heard of them? They did ~$4Bn in 2013...

*Announced* on 12/10/13 they’re building Baidu App Engine using Docker
Use case: easy provisioning of dependencies

“How do I install BlingoDB?”

...quickly starting up a database server, or any other dependencies required by the app
Use case: easy provisioning of ElasticSearch

$ vagrant up && vagrant ssh

> docker pull dockerfile/elasticsearch

... 

> docker run -p 9201:9200 -p 9301:9300 dockerfile/elasticsearch
Use case: easy provisioning of dependencies

Also easy to re-package site-specific customizations of dependencies
Use case: build a component for a scalable system

“How do I deploy my app into production the same way it runs on my machine?”
Use case: build a component for a scalable system

- source code + docker file
- build
- Build artifact (image)
- release
- system config (test)
- Integration test system
- release
- system config (release)
- Release system
Use case: build a component for a scalable system

• Built around tcp://ipaddress:port interconnectivity
• Binary images: self contained with all dependencies, efficient with image layering
• Fast tear down and restart / redeploy and start
• Easy to maximise hardware utilization
So What?
Docker: What’s in it for me?

_devs:_

Docker gives you a really easy way to find and use pre-packaged system components to use in conjunction with your app code.
Docker: What’s in it for me?

Devs:

- Quick start-up
- Declarative
- Convenient
- Agnostic
- Easy to create
- Easy to consume
Docker: What’s in it for me?

DevOps:

Docker gives the Devs a straightforward way to declaratively package deployable, repeatable, self-contained system components.
Docker: What’s in it for me?

DevOps:

- Declarative Dockerfile
- Self-contained dependencies - binary distribution of images
- Quick start-up
- Easy testing of topologies
- Easy enough for devs
Docker: What’s in it for me?

System Architect:

- Aligned with the Twelve-Factor App philosophy
- Easy to bootstrap a sophisticated infrastructure
- Easy to deploy Docker into an architecture
- Easy to remove Docker from an architecture
Why not?

- “Please note Docker is currently under heavy development. It should not be used in production (yet).”
- CLI is functional, but not brilliant
- Additional (unnecessary) layer of abstraction
  - Additional security concerns, complexity, failure modes, learning
Demo time
Demo Scenarios

- Pulling in dependencies (ES) and using them for code
- Packaging an app up into a container
- Using packaged containers to test, e.g., ES clustering
Thank you

Docker!  docker.com

Slides:  bit.ly/Ngb8H1 or brownsofa.org/presentations/files/Docker_whats_in_it_for_me.pdf

@iantruslove
ian.truslove@gmail.com