Xen and Docker

Uniting best of both worlds

Who am I?

- Olivier Lambert
- Xen Orchestra's project leader
- Using Xen in production since 2007
- Met a lot of sysadmins from everywhere

Introduction

- Why Xen?
  - mature (2003)
  - used in very large infrastructures (Amazon, Rackspace...)
  - I'm used to it
- Same principles for others (KVM, VMWare...)

Why this talk?

- Heard lot of ops/sysadmin worried by Docker
- We'll see why
- How to react

Virtual machines

- IT usage revolution:
    - hardware abstraction
    - flexibility
    - resource control and isolation
    - resource delegation

Virtual machines

As an ops, VMs are common stuff

- massive usage in the last 15 years
- we are used to it:
    - procedures
    - supervision
    - sized infrastructure
    - we control them

Hypervisors

Built for **ops** needs:

- live migration
- storage migration
- adjust VM resources in live (CPUs, RAM, disks)
- good isolation (security)
- run almost any OS on top of hypervisor
- lot of tools to administrate (CLI, GUI)

Docker: quick tour

- LXC Container + API to manage them
- out in 2013
- environment abstraction
- build for **devs** needs:

Build, ship, and run any app, anywhere

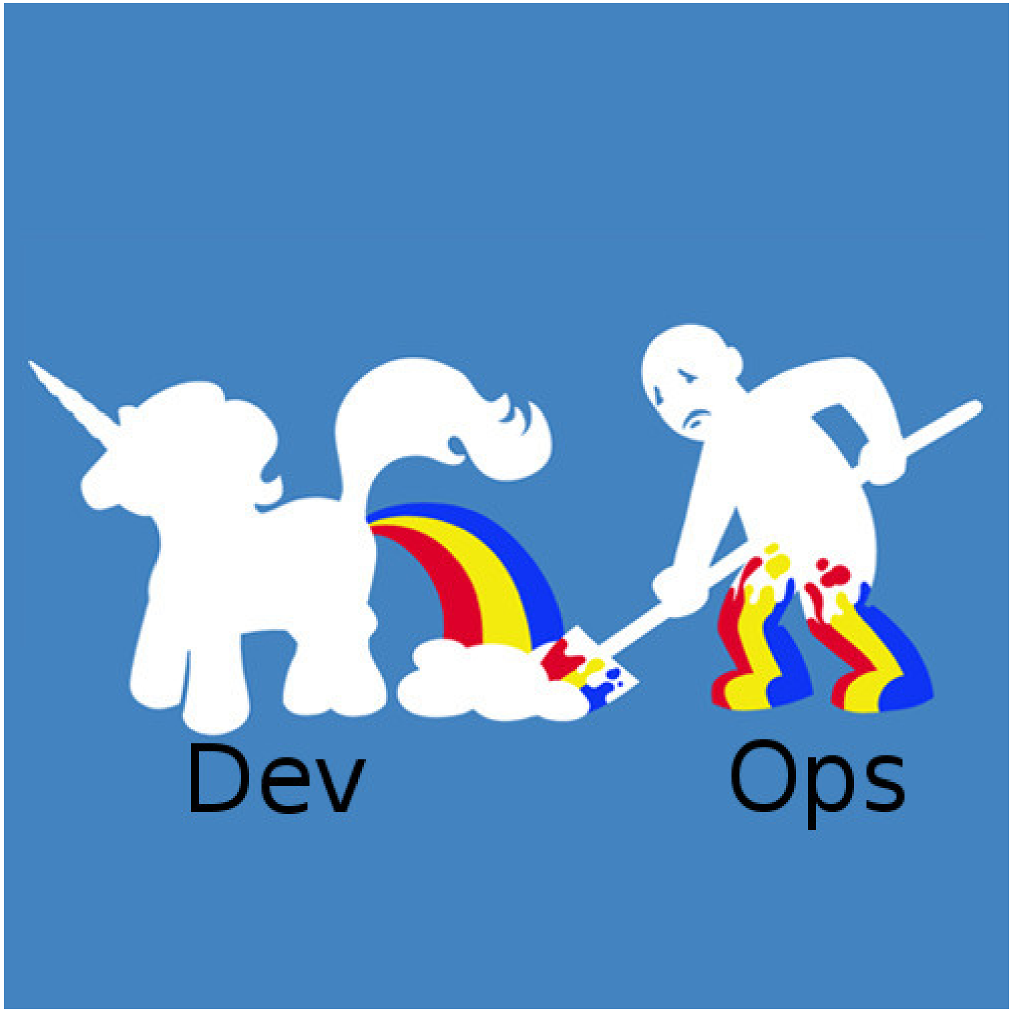
- it means for a dev:
    1. **something working on his laptop** running Docker...
    2. ...will work anywhere else!

Devs first thought:

No more extremist ops to convince for installing

{*insert any controversial technology here*}

Ops first thought



Why?

- Ops fear is:
    - blackbox syndrome (unknown container content)
    - perf impact on the on infrastructure
    - security impact
    - maintenance?

Let's recap the **Ops** feeling:

The power of a developer to push something unknown, anywhere*.

*: where Docker is installed, of course.

"Whoops...!"

KEEP
CALM

Solution

If Docker is a boat shipping containers...

Solution

We'll do this:

Global architecture

schema n1

Uniting powers

schema n2

Results

- Good sides of Xen for **Ops**:
    - all VM flexibility/security package
    - no architecture change
    - Docker resources capped by your VMs...
    - ...but still modifiable in live
    - low overhead (< 10% max)
- **Devs** are happy:
    - they don't care what's underneath
    - they can play with Docker

Is this new?

Nope

1. The Xen+Docker architecture is common usage at Amazon Web Service
    - people create "classical" instances (AWS uses Xen)
    - they install Docker in it
    - tada! Xen+Docker
2. **Docker on top of Xen is here since Docker exists**

Counter-arguments?

Nope

Except:

- **very** specific cases (specific hardware or architecture)
- even the low Xen overhead is not possible

Overcome the fear

1. **Ops**: understand Docker specificities by playing with some dev VMs
2. **Devs**: learn how to use Docker correctly, step by step
3. More teamwork together and/or have **Devops**

"Fear is the path to the dark side. Fear leads to anger. Anger leads to hate. Hate leads to suffering"

As an ops...

1. Start to dedicate VMs for Docker
    - play with it to understand basic principles
    - automatize (template/config) to deploy new Docker VMs quicker
    - start with dedicated VMs for dev environment
2. Gather metrics and trends
    - this way you'll understand what is going on
    - you'll recognize load/pattern behavior later in production
3. Extend the dev environment to test
4. Go in production

These steps are done in parallel with your dev team

As a dev...

1. Start to play with Docker on your own box
    - like **ops**, understand basic principles and workflow
    - learn best practices
    - Docker registry
2. Master your workflow in this dev environment (dev VMs)
    - teamwork with Docker
    - split your app in small bricks
    - Docker compose
3. Start to use it for continuous integration and tests
    - it should be painless, or you have problems
    - good experience before going live
4. **THEN** go in production

No fear!

It's more a matter of workflow and human relationship than technology

Take your time! Remember how much time to master Virtual Machines?

~1 year to master Docker workflow