Retour d'expérience sur 10 ans de virtualisation et présentation de Docker

Gauthier Catteau

Ingénieur Systèmes et Réseaux
Responsable Infrastructure Système de l'Académie de Lille.
gauthier.catteau@ac-lille.fr

13 février 2014
Journée Thématique Min2rien
`http://www.min2rien.fr/`

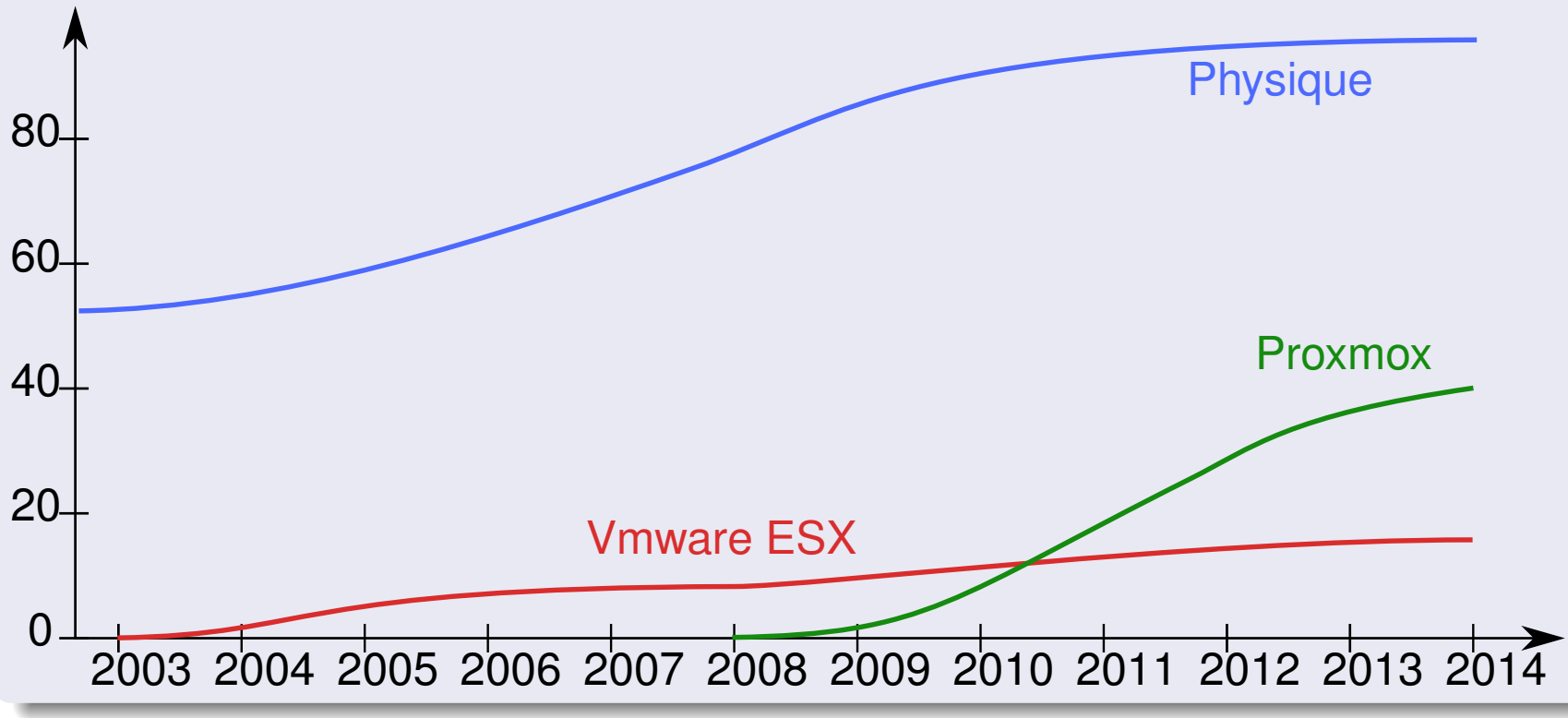
Lignes directrices

Introduction

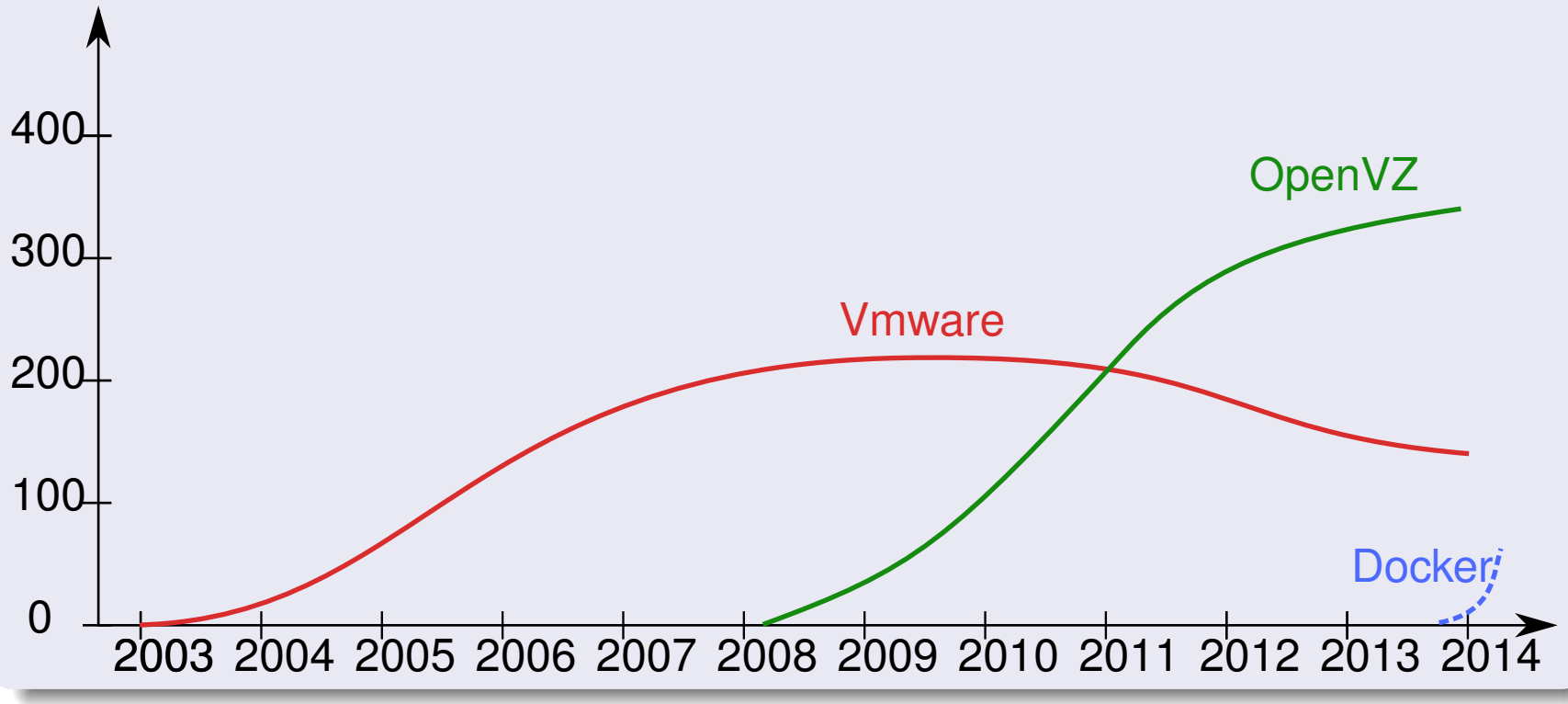
Voila maintenant plus de 10 ans que nous virtualisons notre infrastructure. Vmware, Proxmox et maintenant Docker, explication sur cette évolution et découverte du petit dernier.

Lignes directrices

Répartition des serveurs physiques



Répartition des serveurs virtuels



Lignes directrices

La solution Vmware est *pour l'instant* incontournable pour les raisons suivantes :

- Leader du marché.
- Solution robuste et fiable.
- Migration à chaud efficace.
- Excellente isolation des VM (NSA ?).

Inconvenients de la solution

- Gestion relativement lourde des machines virtuelles complètes (vmwaretools).
- Coût de licence élevé par CPU (> matériel).
- Nécessité d'un SAN pour une utilisation optimum.

La solution Vmware est *pour l'instant* incontournable pour les raisons suivantes :

- Leader du marché.
- Solution robuste et fiable.
- Migration à chaud efficace.
- Excellente isolation des VM (NSA ?).

Inconvenients de la solution

- Gestion relativement lourde des machines virtuelles complètes (vmwaretools).
- Coût de licence élevé par CPU (> matériel).
- Nécessité d'un SAN pour une utilisation optimum.

Dans notre cas, Proxmox est une alternative intéressante pour plusieurs raisons :

- Performance proche de la machine physique.
- Solution facile à mettre en place.
- Modification possible des fichiers des VM éteintes.
- Sauvegarde et PRA plus rapides.
- Virtualisation par conteneur. Démarrage des VM très rapide.

Inconvenients de la solution

- Migration à chaud perfectible.
- Pas de snapshot par VM.
- Solution flexible mais un peu artisanale.

Dans notre cas, Proxmox est une alternative intéressante pour plusieurs raisons :

- Performance proche de la machine physique.
- Solution facile à mettre en place.
- Modification possible des fichiers des VM éteintes.
- Sauvegarde et PRA plus rapides.
- Virtualisation par conteneur. Démarrage des VM très rapide.

Inconvenients de la solution

- Migration à chaud perfectible.
- Pas de snapshot par VM.
- Solution flexible mais un peu artisanale.

Dans quels cas utiliser VMware

- Serveurs avec applicatif nécessitant un support de l'éditeur.
- Serveurs ne pouvant tolérer un arrêt de production.

Dans quels cas utiliser Proxmox

- Services redondés derrière des répartiteurs de charges.
- Ou services moins critiques.

Dans quels cas utiliser VMware

- Serveurs avec applicatif nécessitant un support de l'éditeur.
- Serveurs ne pouvant tolérer un arrêt de production.

Dans quels cas utiliser Proxmox

- Services redondés derrière des répartiteurs de charges.
- Ou services moins critiques.

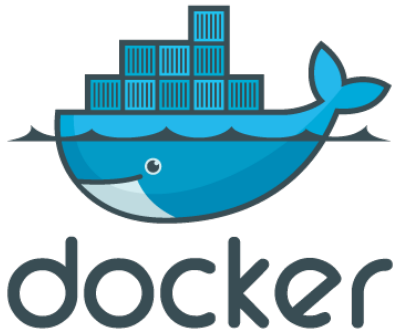
Étude des évolutions possibles

Ces deux solutions fonctionnent mais la charge d'exploitation reste élevée pour quatre personnes. Les serveurs virtuels sont livrés dans la journée, mais il reste souvent beaucoup de réglages à faire après livraison.
D'où la recherche d'une solution :

- Facile à mettre en œuvre.
- Automatisable au maximum.
- Identique pour le Développement, la Qualif, l'Expérimentation et la Production.
- De virtualisation applicative.

Bref une solution Cloud.

Lignes directrices



Docker c'est quoi ?

Docker est un projet open-source créé pour générer rapidement des conteneurs portables pour toutes sortes d'applications.
Le même conteneur peut être utilisé pour les développements, les tests, sur un portable, mis à l'échelle en production dans une VM, Openstack ou un cloud public ...

Origine et première version

- Docker est une réécriture du code de dotCloud PaaS (Plateform As A Service).
- Version d'origine en Python, maintenant réécrite en Go.
- C'est un projet jeune (janvier 2013), première release public 0.1 le 27 mars 2013.
- Version 0.8 en février 2014.
- Utilisé dans plus de 150 projets (OpenStack, CoreOS, Dokku, OpenShift...)
- Dans le TOP10-2013 sur 10 millions de projet GitHub.

Sur quoi est basé Docker ?

- LinuX Conteneurs (LXC)
- Control Groups et Namespaces
- AUFS ou Btrfs (bientôt ZFS)
- API HTTP client-serveur

LinuX Conteneurs (LXC)

- LXC permet :
  - De lancer un linux dans un linux.
  - L'isolation des processus et de l'environnement d'exécution.
- Vu de dedans c'est comme une VM.
- Vu de dehors les process s'exécutent normalement.

Pourquoi des Conteneurs ?

- **Vitesse** : Démarrage dans la seconde.
- **Empreinte** : 100-1000 conteneurs par machine. Peu d'espace disque nécessaire.

LinuX Conteneurs (LXC)

- LXC permet :
  - De lancer un linux dans un linux.
  - L'isolation des processus et de l'environnement d'exécution.
- Vu de dedans c'est comme une VM.
- Vu de dehors les process s'exécutent normalement.

Pourquoi des Conteneurs ?

- **Vitesse** : Démarrage dans la seconde.
- **Empreinte** : 100-1000 conteneurs par machine. Peu d'espace disque nécessaire.

Control Groups et Namespaces

Le noyau Linux implémente des fonctionnalités permettant de limiter, d'isoler et de contrôler les ressources suivantes :

- CPU
- Mémoire
- I/O Disques

Installation : Ubuntu Linux 12.04.3 64bits et suivantes

- AUFS support
  Le kernel 3.8 et suivants intègrent le support LXC et AUFS/Btrfs nécessairent pour Docker.
- Ajout du dépôt Docker

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 36
     A1D7869245C8950F966E92D8576A8BA88D21E9
sudo sh -c "echo␣'deb␣http://get.docker.io/ubuntu␣docker␣main'␣>␣/etc/apt/
     sources.list.d/docker.list"
```

- Installation

```
sudo apt-get update
sudo apt-get install lxc-docker
```

Lignes directrices

Le classique Hello World !

- Téléchargement d'une image de base (ubuntu, centos, busybox ...)

```
$> docker pull ubuntu
```

- Lister les images du système

```
$> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
ubuntu              12.04               8dbd9e392a96        9 months ago        128 MB
ubuntu              latest              8dbd9e392a96        9 months ago        128 MB
ubuntu              precise             8dbd9e392a96        9 months ago        128 MB
ubuntu              12.10               b750fe79269d        9 months ago        175.3 MB
ubuntu              quantal             b750fe79269d        9 months ago        175.3 MB
busybox             latest              e9aa60c60128        10 months ago       4.964 MB
```

- Afficher "Hello World"

```
$> docker run ubuntu:12.04 echo "hello world"
```

Mode détaché

- Lancer Docker en utilisant le flag -d

```
$> docker run -d ubuntu sh -c "while true; do echo hello world; sleep1; done"
```

- Récuperer l'id du conteneur

```
$> docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
9d55d7d2c88d        ubuntu:12.04        sh -c while true; do     16 seconds ago      Up 15 seconds
```

- Attacher la console au conteneur

```
$> docker attach <conteneur_id>
```

- Stop/Start/Restart le conteneur

```
$> docker stop <conteneur_id>
```

Différences entre Conteneurs et Images

- Supprimer un fichier d'une image

```
$> docker run -d busybox rm /etc/passwd
```

- Le fichier est-il toujours là ?

```
$> docker run busybox cat /etc/passwd
root:x:0:0:root:/root:/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
```

- Valider la nouvelle image

```
$> docker ps -n=2
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
754b8e58437f        busybox:latest      cat /etc/passwd     16 seconds ago      Exit 0
6c49f8cce911        busybox:latest      rm /etc/passwd      About a minute ago  Exit 0
$> docker commit 6c49f8cce911 gauthier/broken-busybox
```

- Le fichier est supprimé dans notre nouvelle image !

```
$> docker run gauthier/brocken-busybox cat /etc/passwd
cat: can't open '/etc/passwd': No such file or directory
```

Automatiser la création avec les *Dockerfile*

Docker permet d'automatiser la création d'images qui pourront être ensuite lancées de multiples fois.

```
# Création d'une image à partir de l'image ubuntu du dépôt Docker.
# Lancement de la création avec la commande :
# docker build -t repo/image-name chemin/du/DockerfileDir
FROM        ubuntu:12.04
MAINTAINER Monsieur Linux
ENV         DEBIAN_FRONTEND noninteractive
RUN         apt-get update ; apt-get -y install nginx
ADD         my-vhost /etc/nginx/site-enable/
EXPOSE      80
CMD        /etc/init.d/nginx start
```

À chaque instruction Docker *commit* les changements dans une nouvelle image.
Si une image existe avec le même parent et la même commande, Docker utilisera l'image en cache au lieu d'exécuter de nouveau l'instruction.

Annuaire de conteneurs sur index.docker.io

- Téléchargement d'une image sur l'index public

```
$> docker search apache
$> docker pull creack/apache2
```

- Lancement du conteneur et vérification des ports

```
$> docker run -d creack/apache2
$> docker ps
```

- Mappage des ports du conteneur sur l'hôte

```
$> docker run -d -p 8888:80 -p 4444:443 creack/apache2
$> docker ps
```

Lignes directrices

Passage à la vitesse supérieure

Il existe de nombreuses solutions cloud basées sur Docker.

- Flynn (Solution PaaS libre)
- Deis (chef, Docker, nginx)
- CoreOS (Mini OS avec Docker + etcd)
- Openstack Docker (La solution Cloud OpenSource)
- Redhat Openshift (La solution Cloud de Redhat)

Avantages pour les développeurs :

- Un environnement propre, sûr et portable.
- Pas de soucis de dépendances manquantes.
- Exécution de chaque application dans son propre conteneur isolé, ce qui permet une exécution avec différentes versions de bibliothèques.
- Automatisation des tests, de l'intégration et du packaging.
- Réduction/suppression des problèmes de compatibilité sur des systèmes différents.

Avantages pour les AdminSys :

- Cycle de vie des applications plus efficace, cohérente et reproductible.
- Augmentation de la qualité du code produit par les développeurs.
- Suppression des incohérences entre les environnements de développement, de test et de production.
- Meilleure séparation des tâches.
- Amélioration considérable de la vitesse et de la fiabilité de déploiement.

Avantages pour les développeurs :

- Un environnement propre, sûr et portable.
- Pas de soucis de dépendances manquantes.
- Exécution de chaque application dans son propre conteneur isolé, ce qui permet une exécution avec différentes versions de bibliothèques.
- Automatisation des tests, de l'intégration et du packaging.
- Réduction/suppression des problèmes de compatibilité sur des systèmes différents.

Avantages pour les AdminSys :

- Cycle de vie des applications plus efficace, cohérente et reproductible.
- Augmentation de la qualité du code produit par les développeurs.
- Suppression des incohérences entre les environnements de développement, de test et de production.
- Meilleure séparation des tâches.
- Amélioration considérable de la vitesse et de la fiabilité de déploiement.

Questions ?

Des questions ?