Environnements de dév Drupal automatisés avec LXC et Ansible

Meetup Drupal Lyon, 5 juillet 2016

Qui suis-je ?

Senior Technical Solutions Analyst @Acquia

- Drupaliste depuis 2007
- Membre de la communauté Drupal et communauté lyonnaise depuis 2010

En ligne

- www.drupalfacile.org
- @DrupalFacile
- anavarre.net
- @AurelienNavarre

Première partie

LXC, containers jetables en 3 commandes

Préambule

- Si vous utilisez Windows ou Mac OS, Docker est probablement plus adapté
- Pour tester les containers*, vous pouvez installer Linux dans une machine virtuelle, même sous Windows ou Mac OS
- Si vous utilisez Linux sur vos serveurs, pensez containers et services isolés, plutôt qu'une infrastructure monolithique traditionnelle (tout dans une VM)

* containers et non conteneurs. On utilisera ici le terme anglais communément employé

Pourquoi les containers ? Pourquoi LXC ?

- Un ordinateur n’est pas un environnement isolé (et maîtrisé) de travail
- VMware, Parallels, Virtualbox / Vagrant, QEMU ou encore Boxes ne répondent pas à tous les besoins
- LXC est - idéalement - fait pour des environnements de dév jetables
- LXC est plus bas niveau que Docker
- LXC ne nécessite aucune sur-couche tierce et/ou propriétaire
- LXC prend très peu de place sur votre système
- LXD pour aller plus loin (hyperviseur complet)

Installer LXC

Ubuntu / Debian

```
$ sudo apt-get install lxc libvirt-bin ebtables dnsmasq
```

Pour Red Hat / CentOS / Fedora, remplacez `apt-get` par `yum/dnf`

Quelle version de LXC utilisez-vous ?

```
$ sudo lxc-ls --version
1.1.5
```

Quelques commandes importantes

Les exécutables LXC se trouvent sous `/usr/bin`

- `lxc-create` : créer un container
- `lxc-ls` : lister les containers
- `lxc-info` : obtenir l'état du container (démarré / arrêté)
- `lxc-start` : démarrer un container
- `lxc-attach` : se connecter à un container
- `lxc-stop` : stopper un container
- `lxc-clone` : cloner un container
- `lxc-destroy` : détruire un container

Créer un premier container

Quel template choisir ?

```
$ ls /usr/share/lxc/templates
lxc-archlinux lxc-centos lxc-debian lxc-fedora lxc-ubuntu
(etc.)
```

Créer un container

```
$ sudo lxc-create -t debian -n drupal
```

- `-t` pour passer le nom d'un template à utiliser
- `-n` pour personnaliser le nom du container

Configurer le réseau

Quelles interfaces sont disponibles ?

```
$ ifconfig -s | awk {'print $1'}
Iface lo vethUMI8 virbr0 wlp1s0
```

`$ virsh net-start default` si **virbr0** n'apparaît pas

Configuration minimaliste du réseau dans `/var/lib/lxc/drupal/config`

```
lxc.network.type = veth
lxc.network.link = virbr0
lxc.network.flags = up
```

← type de virtualisation réseau

← lien de connexion vers l'hôte

← activer le réseau

Ou mieux, globalement depuis `/etc/lxc/default.conf`

Démarrer le container

Quel est l'état du container ?

```
$ sudo lxc-info -n drupal | grep State
State:          STOPPED
```

Démarrer le container

```
$ sudo lxc-start -n drupal
```

Confirmer le changement d'état

```
$ sudo lxc-info -n drupal | grep State

State:          RUNNING
```

Première connexion au container

Changer immédiatement le mot de passe root avant de se connecter

```
$ sudo chroot /var/lib/lxc/drupal/rootfs passwd
```

Connexion au container

```
$ sudo lxc-attach -n drupal
```

Avec root, créer un utilisateur `lxc` et l'ajouter dans les sudoers

```
$ usermod -aG sudo lxc
```

Pour Red Hat / CentOS / Fedora, taper `$ adduser lxc -G wheel`

Préparer la connexion SSH au container

Vérifier l’adresse IP attribuée au container

```
$ hostname -I
192.168.124.125
```

Utiliser une clé SSH plutôt qu’un mot de passe lors de la prochaine connexion

```
$ ssh-copy-id lxc@192.168.124.125
lxc@192.168.124.125's password:

Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'lxc@192.
168.124.125'"
```

Tester la connexion SSH au container

```
$ ssh lxc@192.168.124.125
lxc@192.168.124.125's password:
Last login: Sat Jun 23 12:59:02 2016 from 192.168.124.1
```

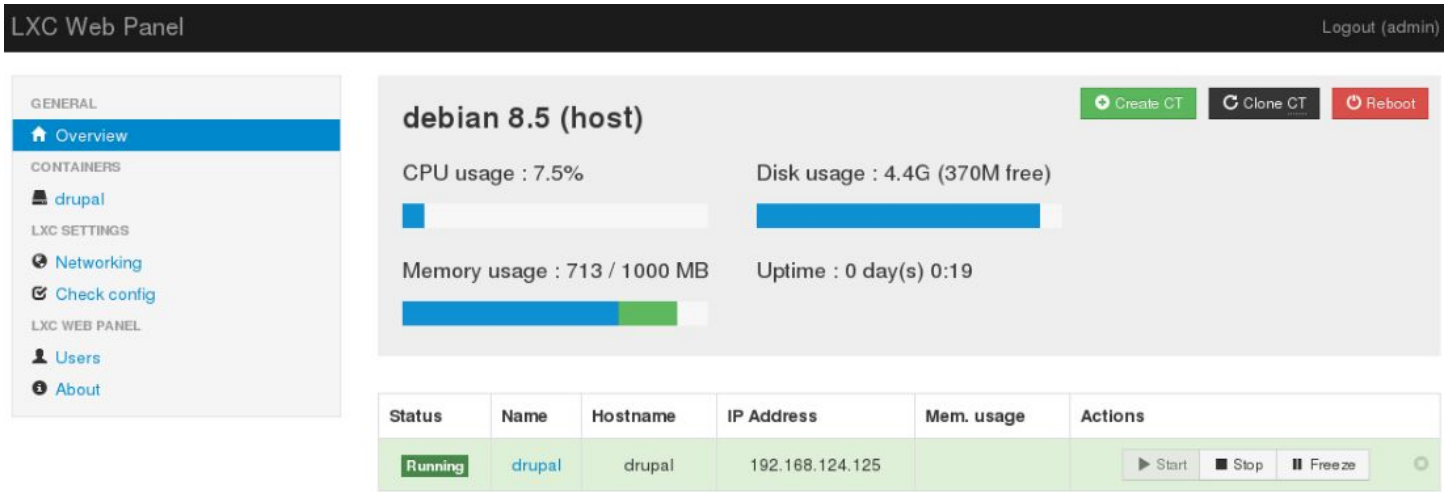
- Le container démarre et se connecte au réseau
- On peut s’y connecter sans mot de passe via SSH
- On a un compte utilisateur autre que root disponible
- Nous sommes désormais prêts pour l’orchestration via Ansible

Le petit plus : LXC Web Panel

La ligne de commande c'est bien, mais une GUI c'est pas mal non plus.

```
$ wget http://lxc-webpanel.github.io/tools/install.sh -O - | bash
```

Accès à l'interface via http://localhost:5000 (admin/admin)



Deuxième partie

Ansible, l’orchestration pour tous

Pourquoi l’orchestration ? Pourquoi Ansible ?

- L’orchestration permet de simplifier et d’automatiser toutes les taches d’administration système pour 1 ou 1000 serveurs à la fois
- Les principaux concurrents d’Ansible sont Puppet et Chef
- Ansible n’installe pas de daemon (une connexion SSH suffit)
- Ansible est simple d’apprentissage (quasi exclusivement du YAML)
- Ansible est Open Source
- Ansible est là pour rester (racheté par Red Hat en 2015)
- Si vous changez de technologie (Vagrant/Virtualbox, LXC, Docker...) pour vos environnements de dév, vous conservez quand même votre orchestration !

Installer Ansible

Ubuntu / Debian

```
$ sudo apt-get install ansible python-simplejson
```

Pour Red Hat / CentOS / Fedora, remplacez `apt-get` par `yum/dnf`

Quelle version d'Ansible utilisez-vous ?

```
$ ansible --version
ansible 1.9.4
```

Définir les hôtes à orchestrer

Le fichier `/etc/ansible/hosts` sert à créer des groupes logiques

```
[lxc]
192.168.124.125

[rackspace]
203.0.113.2

[aws]
203.0.113.10
203.0.113.11
203.0.113.12
```

Tester une connexion au container via Ansible

On a donc crée un groupe logique LXC dans `/etc/ansible/hosts`

```
[lxc]
192.168.124.125
```

Ce qui permet d'envoyer un ping sur le groupe (tous les hôtes) plutôt que les IPs

```
$ ansible lxc -m ping -u lxc
```

```
192.168.124.125 | success >> {
    "changed": false,
    "ping": "pong"
}
```

Le concept de playbook

- Le playbook est à Ansible ce que le livre de cuisine est au cuisinier : un recueil de recettes pour préparer des environnements de développement personnalisés avec un certain nombre “d’ingrédients”

Comment exécuter un playbook ?

```
$ ansible-playbook lxc.yml --user=lxc --ask-become-pass
```

Exemple de playbook minimaliste

```
---
- hosts: lxc
  sudo: yes
  handlers:
    - include: handlers.yml
  vars_files:
    - vars.yml
  tasks:
    - include:system.yml tags=system
    - include:lamp.yml tags=lamp
    - include:drupal.yml tags=drupal
```

Exemple : installer Apache

```
---
- name: Install Apache
  apt:
    name: "{{ item }}"
    state: present
  with_items:
    - apache2
    - apache2-utils
```

Pour Red Hat / CentOS / Fedora, remplacez `apt` par `yum/dnf`

Depuis Ansible 2.0, le module `package` permet d'abstraire le système de packaging.

Comprendre les handlers

Les handlers ne sont exécutés que si une tâche Ansible envoie un évènement `notify`

```
---
- name: Restart Apache
  service:
    name: apache2
    state: restarted
```

Nom du handler

Invocation du handler dans une tâche

```
---
- name: Enable mod_rewrite
  apache2_module:
    name: rewrite
    state: present
    notify: Restart Apache
```

Comprendre les variables

```
user: lxc
drupal_version: 8.1.3
drush_path: /usr/local/bin/drush
```

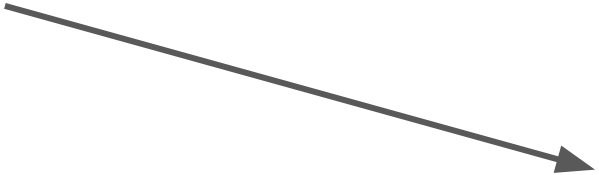
Une variable en action

```
---
- name: Check if Drush is installed
  stat:
    path: "{{ drush_path }}"
  register: drush
```

Comprendre les ‘tasks’

Les tâches dans un playbook pointent vers des fichiers YAML qui définissent - idéalement - des tâches logiques à exécuter sur le/les hôte(s)

```
tasks:
  - include:system.yml
```



```
- name: Install system packages
  package:
    name: "{{ item }}"
    state: present
  with_items:
  - git
  - vim
  - wget
```

Fichier system.yml

Quelques exemples de modules Ansible

package	Gestionnaire de paquets générique (apt, yum...)
service	Gestionnaire de services (lancer, arrêter, redémarrer...)
stat	Récupére le statut d'un fichier/dossier dans le système de fichiers
file	Crée des fichiers/dossiers ou spécifie leurs attributs (chmod, chown...)
command	Exécute une commande Shell sur le/les hôte(s)
get_url	Télécharge des fichiers via HTTP, HTTPs ou FTP
copy	Copie de fichiers vers le/les hôtes
replace	Remplace une chaîne de caractères dans un fichier (utilise regex)

http://docs.ansible.com/ansible/list_of_all_modules.html

Démo

A retenir

- Etudiez LXC mais misez sur Docker pour la portabilité des containers
- En dév, la technologie de virtualisation est intéressante, mais n’est finalement qu’un prétexte pour parler d’orchestration
- Pas besoin d’être développeur pour avoir sa ceinture noire en orchestration
- Depuis Drupal 8, YAML ne vous fait de toute façon plus peur ;-)

Merci. Questions ?