décembre 27th, 2009

OpenVZ (Partie 1)

1. Introduction

J'utilise OpenVZ depuis maintenant 2 mois dans mon travail ainsi que chez moi. Il s'agit d'un logiciel de virtualisation basé sur l'usage de conteneurs. L'idée d'isoler certains services me plais bien, et comme j'ai l'habitude de casser mes distributions rapidement, le fait de repartir sur une base saine rapidement est un réel plus. Ces OS isolés sont nommés des « conteneurs », ou des « environnement virtuel » (« VE »). J'utiliserais ces 3 termes.

Ceci est un tutorial clés en main, qui vas vous permettre de mettre en place une solution rapidement. Néanmoins, je vais aussi aborder l'utilisation de LVM2. afin de cloisonner les environnements sur des espaces disques distincts. Vous pouvez vous passer de cette partie facultative, en lisant directement le tuto sur OpenVZ.

Ce tutorial portera de la mise en place sur Debian Lenny.

2. Mise en place de LVM2 (facultatif)

LVM désigne « Logical Volume Manager ». Il s'agit d'une couche intermédiaire entre le disque dur et un système de fichier. Son avantage est sa souplesse quant à la gestion des espaces disques. On peut créer ou supprimer très rapidement une partition LVM, lui attribuer un système de fichier, puis la redimensionner à chaud (augmentation ainsi que réduction). Sachez que dans le cadre d'OpenVZ, l'**unique intéret** de LVM vous permettra d'effectuer des backup de vos conteneurs à chaud, sans avoir à les désactiver.

2.1. Installation de LVM2

Par défaut, LVM est déjà activé dans le noyau de Debian. Il nous suffit donc d'installer les outils de contrôle:

```
# aptitude install lvm2
```

2.2. Création d'une partition LVM

2.2.1. Libération d'une place sur un disque

Notre disque est occupé par 2 partition: sda1 et sda2. Nous allons réduire la taille de la partition sda de 80 Go, afin de créer une partition LVM. Sda est formaté en ext3 et voici ce que la commande df nous retourne:

```
$ df -h
 Sys. de fich.    Tail. Occ. Disp. %Occ. Monté sur
 /dev/sda1        201G  131G   70G  66% /
 /dev/sda2        100G   10G   90G  10% /mnt/data
```

A ce stade, je vous recommande **chaudement** de sauvegarder toutes les données sensibles. Nous ne sommes jamais a l'abri de pertes éventuelles.

Tuons les processus accédant à cette partition:

```
# fuser -k /dev/sda2
```

Démontons votre partition:

```
# umount /dev/sda2
```

Vérifions notre partition avant de la manipuler:

```
# fsck /dev/sda2
```

Ensuite, nous allons convertir notre partition ext3 ves ext2. En effet, il n'est pas possible de redimensionner du ext3, mais c'est possible avec ext2. La seule différence entre ces systèmes de fichier est la journalisation, présente dans ext3:

```
# tune2fs -O ^has_journal /dev/sda2
```

Redimensionnons la partition, en indiquant la nouvelle taille de la partition:

```
# resize2fs /dev/sda2 20G
```

Réactivons la journalisation de la partition:

```
# tune2fs -j /dev/sda2
```

2.2.2. Remise en service de la partition sda2

Modifions la table des partitions du disque avec fdisk:

```
# fdisk /dev/sda
```

Supprimons la seconde partition de la table:

```
Command (m for help): d
Partition number (1-2): 2
```

Recréons la partition numéro 2:

```
Command (m for help): n
Command action
     l   logical (5 or over)
     p   primary partition (1-4)
p
Partition number (1-4): 2
```

Cette fois-ci, nous n'allons pas utiliser tout le disque dur, mais un espace plus réduit pour sda2. Gardons la valeur de début de partition par défaut. En revanche, nous allons choisir la taille finale en précisant un nombre en méga-octets. Comme cette taille sera transformé en nombre de cylindres, je vous recommande d'ajouter une marge de 4% par sécurité (1040M), ce afin d'éviter les « imprécisions »:

```
First cylinder (2344592-13057890, default 2344592): &lt;ENTRER&gt;
Using default value 2344592
Last cylinder or +size or +sizeM or +sizeK (2344592-13057890, default 13057890): +1
```

Nous ne quittons pas encore fdisk

2.2.3. Création de la partition sda3 (LVM)

Créons la dernière partition:

```
Command (m for help): n
Command action
     l   logical (5 or over)
     p   primary partition (1-4)
p
Partition number (1-4): 3
```

Nous utiliserons tout les 80 Go disponibles, donc laissons les valeurs par défaut:

```
First cylinder (5745696-13057890, default 5745696): &lt;ENTRER&gt;
Using default value 5745696
Last cylinder or +size or +sizeM or +sizeK (5745696-13057890, default 13057890): &l
```

Nous devons maintenant définir la partition comme étant de type LVM:

```
Command (m for help): t
Partition number (1-4): 3
Selected partition 3
Hex code (type L to list codes): 8e
Changed system type of partition 3 to 8e (Linux LVM)
```

Puis nous enregistrons les modifications:

```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Nous pouvons alors redémarrer:

```
# shutdown -r now
```

Enfin, nous créons la partition LVM:

```
# pvcreate /dev/sda3
```

2.3. Création d'un volume group

Le « volume group » peut englober une ou plusieurs partitions LVM. En rassemblant ces partitions physiques en une seule

logique, on dispose d'un grand volume aménageable selon ses besoins. C'est dans ce volume que l'on créera les volumes logiques LVM. Ce volume principal se nommera ici « env »:

```
# vgcreate env /dev/sda4
```

2.4. Création d'un volume logique

On créé une partition qui va accueillir notre environnement virtuel. Elle va se situer dans le volume group « env ». Nous la nommerons « machine1″ et elle aura une taille de 5 Go.

```
# lvcreate -n machine1 -L 5g env
```

2.5. Création d'un système de fichier

Nous trouverons alors cette partition dans */dev/env/machine1*. Nous formatons cette partition en ext3:

```
# mkfs -t ext3 /dev/env/machine1
```

Maintenant que nous avons notre volume prêt à l'emploi, nous pouvons mettre en place OpenVZ.

3. Installation d'OpenVZ sur Debian Lenny

3.1. Mise en place du noyau

Téléchargeons le noyau OpenVZ.

```
# aptitude install linux-image-openvz-686
```

Cette commande installera le noyau, les outils nécessaires et configurera le bootloader pour démarrer avec OpenVZ.

```
# shutdown -r now
```

3.2. Vérifications post-installation

Vérifions que le noyau comporte bien la mention « openvz »:

```
 # uname -r
2.6.26-1-openvz-686
```

Vérifions aussi si le module noyau en enclenché:

```
 # ps ax | grep vz
2478 ?        S      0:00 [vzmond]
```

Enfin veillons à ce qu'il existe bien une interface virtuelle pour les conteneurs:

```
# ifconfig
venet0    Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
UP BROADCAST POINTOPOINT RUNNING NOARP  MTU:1500  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

3.3. Préparation du système

Nous allons maintenant procéder à un petit travail préliminaire avant de créer nos premiers environnements virtuels (enfin!). Pour faciliter les déplacements dans les fichiers d'OpenVZ, nous crérons un lien logique vers la racine du système:

```
# ln -s /var/lib/vz /vz
```

Maintenant modifions le comportement de notre noyau en ajoutant ceci au fichier /etc/sysclt.conf :

```
# On Hardware Node we generally need
# packet forwarding enabled and proxy arp disabled
net.ipv4.conf.default.forwarding=1
net.ipv4.conf.default.proxy_arp = 0
net.ipv4.ip_forward=1

# Enables source route verification
net.ipv4.conf.all.rp_filter = 1
# Enables the magic-sysrq key
kernel.sysrq = 1

# we do not want all our interfaces to send redirects
net.ipv4.conf.default.send_redirects = 1
net.ipv4.conf.all.send_redirects = 0
```

Puis, appliquons les changements:

```
# sysctl -p
```

Ajoutons OpenVZ au démarrage automatique et lancons-le:

```
# update-rc.d vz defaults
# /etc/init.d/vz start
```

3.4. Mise en place d'un VE

3.4.1. Récupération d'un template d'OS

La voie la plus rapide est de télécharger une distribution pré-packagée. Vous pouvez en trouver ici. Dans cet exemple, j'utiliserais Debian.

```
# cd /vz/template/cache
# wget http://download.openvz.org/template/precreated/debian-5.0-x86.tar.gz
```

3.4.2. Si vous avez utilisé LVM

De là, les choses diffèrent si jamais vous avez utilisé LVM ou pas. En effet, chaque VE est identifié par un numéro de votre choix, le conteneur ID (ou CTID). Dans une structure classique, OpenVZ place le répertoire d'un environnement dans /vz/private/CTID.

Si votre point de montage du volume « machine1″ est de ce type: /vz/private/ »CTID », OpenVZ refusera de créer votre conteneur. La solution est donc de créer le conteneur au delà du point de montage. ex: /vz/private/point_de_montage /CTID.

Tout d'abord, créons notre point de montage, puis nous le montons à chaque démarrage de l'hôte:

```
# mkdir /vz/private/point_de_montage
# echo "/dev/env/machine1    /vz/private/point_de_montage      ext3    defaults
# mount -a
```

Ensuite, créons notre conteneur à proprement parler (le nom du template est celui du fichier téléchargé, sans le « .tar.gz ». Ici le CTID sera « 10″)

```
# vzctl create 10 --ostemplate debian-5.0-x86 --private=/vz/private/point_de_montag
```

3.4.3. Si vous n'avez pas utilisé LVM

Créons notre conteneur à proprement parler (le nom du template est celui du fichier téléchargé, sans le « .tar.gz, avec 10 pour CTID

```
# vzctl create 10 --ostemplate debian-5.0-x86
```

3.4.4. Configurer l'environnement virtuel

Il est temps de configurer notre machine afin de la rendre pleinement utilisable.

Toujours sur l'hôte, configurons successivement l'IP , le serveur DNS et son nom:

```
hôte# vzctl set CTID --ipadd a.b.c.d --save
hôte# vzctl set CTID --nameserver a.b.c.d --save
hôte# vzctl set CTID --hostname superbeMachine --save
```

Nous pouvons aussi créer quelques comptes:

```
hôte# vzctl set CTID --userpasswd login:motDePasse
```

3.5. Derniers réglages et gestion des VE

3.5.1. De la création …

Il ne nous reste plus qu'à configurer la taille de notre VE:

```
hôte# vzctl set CTID –-diskspace 80G:80G --save
```

Ainsi que la quantité de RAM allouée:

```
hôte# vzctl set 100 –-privvmpages 250M:250M --save
```

Maintenant, nous pouvons démarrer notre environnement virtuel et entrer dedans:

```
hôte# vzctl start 101
hôte# vzctl enter CTID
entered into container CTID
[container]#
```

Préférez les accès SSH plutôt que de passer par l'hôte par le suite. Moins vous utiliserez la machine hôte, moins vous compromettrez vos environnements.

3.5.2. … a la destruction

nous pouvons aussi arrêter notre environnement (!) et le détruire (!!!):

```
hôte# vzctl stop CTID
hôte# vzctl destroy CTID
```

Si vous avez utilisé un volume LVM, n'oubliez pas de le détruire:

```
sudo umount /vz/private/point_de_montage
sudo lvremove /dev/env/machine1
sudo rmdir /vz/private/point_de_montage
```

Le petit mot de la fin

Voila c'est fini pour aujourd'hui. Ce tutorial est resté succin en concentrant sur la création de volumes LVM et l'installation d'OpenVZ. Mais il reste pas mal de petites subtilités à découvrir.

Entre autres, il apparaitra bientôt des billets sur le management des ressources et des quotas sur OpenVZ, comment retailler vos volumes LVM, et quelques petites *surprises*. ;-)

Sources:

Une approche simple sur LVM:
-> *http://linux.developpez.com/lvm/*

La seule doc complète que j'ai trouvé sur l'installation d'OpenVZ avec LVM, par Jérôme Bousquié:
-> *https://sites.google.com/a/bousquie.fr/jerome/Home/openvz-sur-ubuntu-hardy-avec-lvm*

La bible d'OpenVZ à consulter tous les soirs avant de se coucher:
-> *http://wiki.openvz.org/*

Mes bases sur le redimensionnement de partitions, dont le blog de Cep, très renseigné sur la chose
-> *http://tldp.org/HOWTO/Partition/fdisk_partitioning.html*
-> *http://www.howtoforge.com/linux_resizing_ext3_partitions*
-> *http://www.cepcasa.info/blog/?p=38*

Tags: lvm, openvz, virtualization
Posted in lvm, openvz, virtualisation | No Comments »

Commentaires (0)