



IUT DE NANCY CHARLEMAGNE

PROJET TUTEURÉ

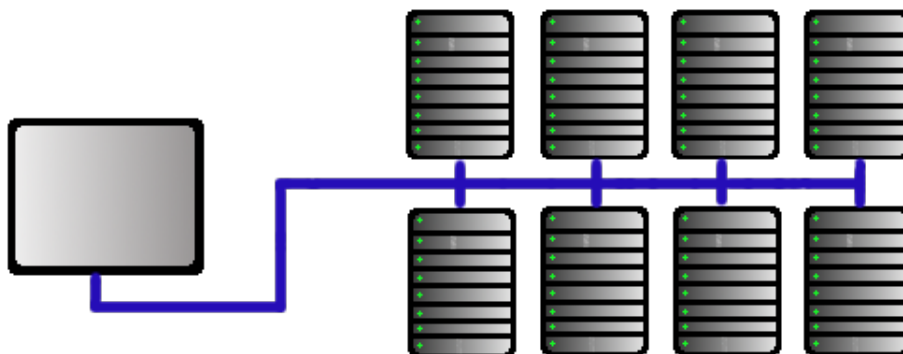
Gestion centralisée de machines virtuelles

Auteurs :

Augustin BOCCA
Mathieu LAMOUROUX
Sébastien MICHAUX
Julien TOURNOIS

Tuteur :

Lucas NUSSBAUM



25 mars 2012

Remerciements

Avant tout développement sur cette expérience, il apparaît opportunt de commencer ce rapport par dess remerciements, à ceux qui nous on beaucoup appris au cours de cette période, et également à ceux qui ont eu la gentillesse de faire de ce projet un moment très profitable

Aussi nous remercions Lucas Nussbaum, notre maître de stage qui nous a accompagné avec beaucoup de patience et de pédagogie.

Table des matières

1	Introduction	4
1.1	Présentation du projet	4
1.2	Introduction à la virtualisation	4
1.2.1	Machine virtuelle	4
1.2.2	Hyperviseur	4
1.2.3	Au commencement, la virtualisation des mainframes	5
1.2.4	Histoire de la virtualisation	5
1.2.5	Enjeux de la virtualisation	5
1.3	Présentation de Grid5000	6
1.3.1	Infrastructure des sites	7
1.3.2	Réseau	8
1.3.3	Environnement logiciel	9
1.4	Répartition des tâches	9
2	Ganeti	10
2.1	Introduction	10
2.2	Installation	11
2.2.1	Modification des sources	11
2.2.2	Mise à jour et installation	12
2.3	Configuration	12
2.3.1	Configuration du fichier hosts	12
2.3.2	Copie des fichier du noyau	12
2.3.3	Création du bridge xen-br0	12
2.3.4	Création du LVM	12
2.3.5	Edition de /usr/share/ganeti/os/debootstrap/common.sh	13
2.3.6	Configuration et initialisation du cluster	13
2.4	Utilisation des nodes	13
2.4.1	Ajouter un node	13
2.4.2	Reconfigurer un node	13
2.4.3	Roles des nodes et opérations	14
2.4.4	Supprimer un node :	15
2.4.5	Manipulation du stockage :	15
2.5	Utilisation des instances	16
2.5.1	Ajouter une instance	16
2.5.2	Supprimer une instance	16
3	OpenXENManager	20
3.1	Présentation	20
3.2	Installation	20
3.3	Problèmes rencontrés avec OpenXenManager	20
3.3.1	Préparation du système	21
3.3.2	Installation de XCP	21

3.4	Xen Cloud Platform	22
4	Archipel Project	26
4.1	Introduction	26
4.2	Architecture	26
4.3	Haute disponibilité et montée en charge	27
4.4	Sécurité	27
4.5	Fonctionnalités	28
4.6	Conclusion	28
5	Virt-manager	30
5.1	Pré-requis et considérations pour les hôtes	30
5.2	Installation côté serveur	30
5.3	Installation côté client	31
5.4	Création d'hôtes virtualisés avec virt-manager	32
5.4.1	Controle d'hotes distants	36
5.4.2	Migration de machines	37
6	Comparatif	39
7	Conclusion	42
A	Sources	43
B	Scripts	44
B.1	Scripts spécifiques à l'environnement de Grid5000	44
B.2	Scripts pour le déploiement de ganeti	47
B.2.1	Configuration générale des noeuds	47
B.2.2	Création de machines virtuelles pour Ganetti	48
B.2.3	Installation et configuration de ganeti	52
B.2.4	Installation et configuration du master node	53
B.2.5	Installation et configuration des noeuds esclaves	55
B.2.6	Ajout des noeuds au cluster et création des instances	57
B.2.7	Modification du fichier de configuration de ganeti	57
B.3	Scripts pour le déploiement de virt-manager	59
B.4	Scripts pour le déploiement d'Archipel	62
B.4.1	Installation d'Archipel	62
B.4.2	Configuration générale de Jabber pour Archipel	63

Chapitre 1

Introduction

1.1 Présentation du projet

Mettre en place, évaluer et comparer différents outils permettant de gérer de manière centralisée et automatisée une infrastructure basée sur des machines virtuelles : Ganeti, OpenXenManager, virt-manager, Archipel...

1.2 Introduction à la virtualisation

1.2.1 Machine virtuelle

Une machine virtuelle est un conteneur isolé capable d'exécuter ses propres système d'exploitation et applications. Une machine virtuelle se comporte exactement comme un ordinateur physique et contient ses propres processeurs, mémoire RAM, disque dur et carte d'interface réseau virtuels. Une machine virtuelle a pour but de générer sur une même machine un ou plusieurs environnements d'exécution applicative. On en distingue deux types d'application : d'une part la virtualisation par le biais d'un hyperviseur jouant le rôle d'émulateur de système (PC ou serveur), d'autre part la virtualisation applicative qui permet de faire tourner une application sur un poste client quelque soit le système sous-jacent.

1.2.2 Hyperviseur

La machine virtuelle avec hyperviseur est utilisée pour générer au dessus d'un système d'exploitation serveur, une couche logicielle sous la forme d'un émulateur permettant de créer plusieurs environnements d'exécution serveur. Cet émulateur se place comme un niveau supplémentaire qui se greffe sur le système d'origine.

1.2.3 Au commencement, la virtualisation des mainframes

La virtualisation a été mise en œuvre pour la première fois il y a plus de 30 ans par IBM pour partitionner logiquement des mainframes en machines virtuelles distinctes. Ces partitions permettaient un traitement « multitâche », à savoir l'exécution simultanée de plusieurs applications et processus. Étant donné que les mainframes consommaient beaucoup de ressources en même temps, le partitionnement constituait un moyen naturel de tirer pleinement parti de l'investissement matériel.

1.2.4 Histoire de la virtualisation

La virtualisation est un concept qui a été mis au point pour la première fois dans les années 1960 pour permettre la partition d'une vaste gamme de matériel mainframe et optimiser l'utilisation du matériel. De nos jours, les ordinateurs basés sur l'architecture x86 sont confrontés aux mêmes problèmes de rigidité et de sous-utilisation que les mainframes dans les années 1960. VMware a inventé la virtualisation pour la plate-forme x86 dans les années 1990 afin de répondre notamment aux problèmes de sous-utilisation, et a surmonté les nombreux défis émergeant au cours de ce processus. Aujourd'hui, VMware est devenu le leader mondial de la virtualisation x86 avec plus de 190 000 clients, dont la totalité des membres du classement Fortune 100.

1.2.5 Enjeux de la virtualisation

Actuellement, les entreprises rencontrent des besoins qui ne sont pas couverts.

Au niveau de la sécurité, les entreprises souhaiteraient isoler les services sur des serveurs différents. Pour la maintenance, il serait utile d'améliorer des services tels que la disponibilité, la migration, la redondance, la flexibilité ou le temps de réponse. Il serait également bienvenu de tester, déléguer l'administration d'un système ...

Une des solutions pour répondre à ces besoins serait d'acquérir davantage de plateformes de travail.

La multiplication des serveurs pose cependant un certain nombre de problèmes, augmenter sans cesse son parc informatique est impossible pour plusieurs raisons :

- Tout d'abord au niveau écologique cela entraînerait un surplus de déchets électronique, une consommation d'énergie directe et de l'énergie utilisée pour refroidir les salles serveur.
- Au niveau de la surface utilisée, les salles machine seraient vite encombrées, puis apparaîtraient des problèmes tels que la nuisance sonore, le manque de puissance pour alimenter les salles serveur.
- Au niveau économique les coûts d'achat, de recyclage, de fonctionnement, de maintenance seraient trop chers. La mise en place de serveur de virtualisation est une solution pour résoudre ces problèmes.

Le but de la virtualisation est de donner un environnement système au programme pour qu'il croie être dans un environnement matériel. Pour cela, une machine virtuelle est utilisée. Ainsi, plusieurs environnements d'exécution sont créés sur une seule machine, dont chacun émule la machine hôte. L'utilisateur pense posséder un ordinateur complet pour chaque système d'exploitation alors que toutes les machines virtuelles sont isolées entre elles.



FIGURE 1.1 – Logo de Grid5000

1.3 Présentation de Grid5000

Aujourd'hui, grâce à Internet, il est possible d'interconnecter des machines du monde entier pour traiter et stocker des masses de données. Cette collection hétérogène et distribuée de ressources de stockage et de calcul a donné naissance à un nouveau concept : les grilles informatiques.

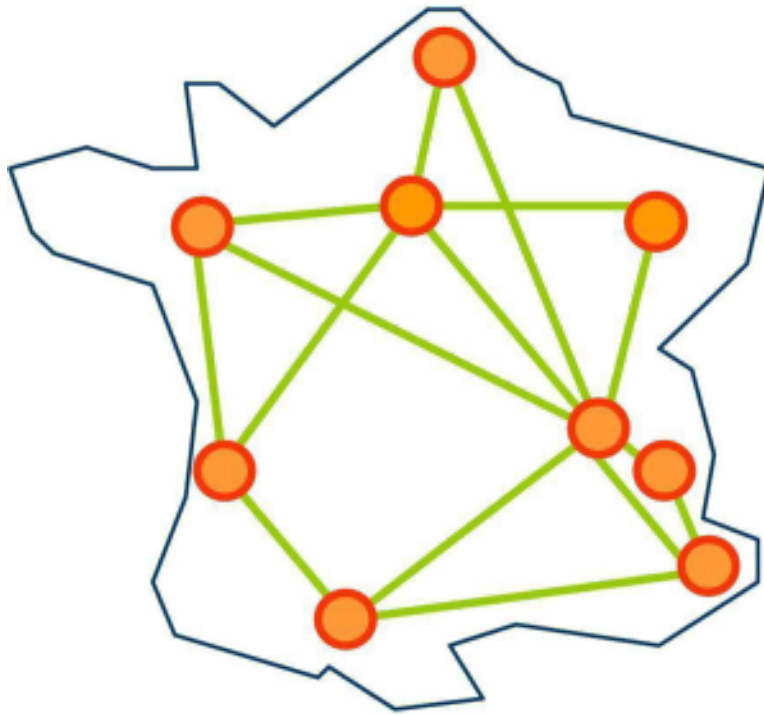
L'idée de mutualiser les ressources informatiques vient de plusieurs facteurs, évolution de la recherche en parallélisme qui, après avoir étudié les machines homogènes, s'est attaquée aux environnements hétérogènes puis distribués ; besoins croissants des applications qui nécessitent l'utilisation toujours plus importante de moyens informatiques forcément répartis.

La notion de grille peut avoir plusieurs sens suivant le contexte : grappes de grappes, environnements de type GridRPC (appel de procédure à distance sur une grille), réseaux pair-à-pair, systèmes de calcul sur Internet, etc... Il s'agit d'une manière générale de systèmes dynamiques, hétérogènes et distribués à large échelle. Un grand nombre de problématiques de recherche sont soulevées par les grilles informatiques. Elles touchent plusieurs domaines de l'informatique : algorithmique, programmation, intergiciels, applications, réseaux.

L'objectif de GRID'5000 est de construire un instrument pour réaliser des expériences en informatique dans le domaine des systèmes distribués à grande échelle (GRID).

Cette plate-forme, ouverte depuis 2006 aux chercheurs de la communauté grille, regroupe un certain nombre de sites répartis sur le territoire national. Chaque site héberge une ou plusieurs grappes de processeurs. Ces grappes sont alors interconnectées via une infrastructure réseau dédiée à 10 Gb/s fournie par RENATER. À ce jour, GRID'5000 est composé de 9 sites : Lille, Rennes, Orsay, Nancy, Bordeaux, Lyon, Grenoble, Toulouse et Nice.

Début 2007, GRID'5000 regroupait plus de 2500 processeurs et près de 3500 cœurs.

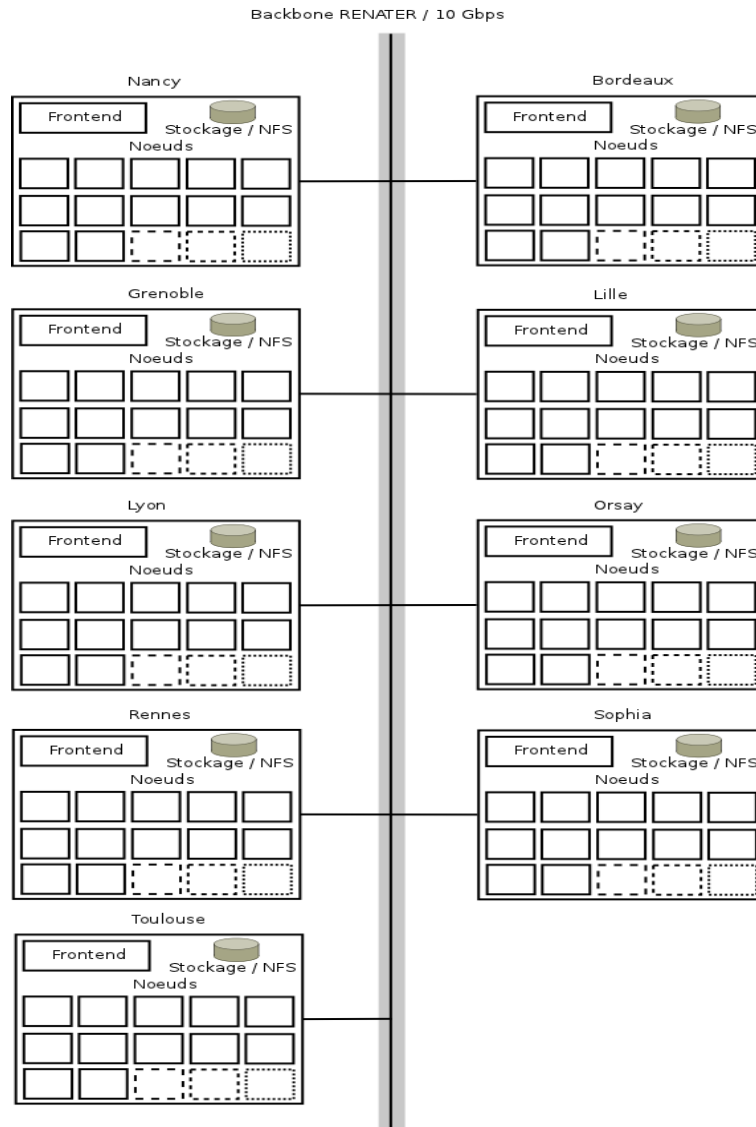


Répartition des sites

1.3.1 Infrastructure des sites

Chaque site héberge :

- un frontend, serveur permettant d'accéder aux clusters disponibles ,
- un serveur de données, pour centraliser les données utilisateurs ,
- plusieurs clusters, c'est-à-dire des grappes de machines homogènes, appelées noeuds (nodes).



Architecture Grid5000

L'utilisateur de Grid 5000 accède à chaque site par son frontend en utilisant le protocole SSH.

Commande :

```
1 ssh utilisateur@access.grid5000.fr
```

Sur tous les serveurs du site, un répertoire home, local à chaque site, est monté avec NFS 2 . A partir du frontend, il est possible d'accéder aux machines des clusters en exectuant des réservations à l'aide de la commande :

```
1 oarsub
```

Grâce à notre tuteur, M. Lucas Nussbaum nous avons pu visiter la salle serveurs du site de Nancy située au Loria, ainsi qu'une présentation de la plate-forme (matériel utilisé, connexions réseau, administration).

Une description détaillée du site de Nancy est disponible sur le site de Grid 5000.

1.3.2 Réseau

Les sites et toutes les machines qu'ils comprennent sont interconnectés par RENATER 3 en 10Gbits/s. De plus, chaque site peut disposer de plusieurs réseaux locaux 4 :

- réseau en ethernet, 1 Gb/s
- réseaux hautes performances (Infiniband 20 Gb/s ou 10 Gb/s, et Myrinet 20 Gb/s)

1.3.3 Environnement logiciel

Tous les serveurs de Grid 5000 fonctionnent sous Debian GNU/Linux. A partir du frontend, l'utilisateur peut réserver des machines en utilisant la suite de logiciels OAR dédiée à la gestion de ressources de clusters, et déployer ses propres images de systèmes à l'aide des outils kadeploy. Il y a deux types de réservation :

- par défaut, pour des besoins de calcul avec OpenMPI;
- pour le déploiement d'environnements (deploy).

Dans le cadre de notre projet, nous effectuons uniquement des réservations de type deploy. La commande oarsub nous permet de réserver des nœuds sur un site (en créant un job). Voici un exemple d'utilisation d'oarsub, pour réserver 3 nœuds pendant 2 heures en déploiement.

```
1 oarsub -I -t deploy -n'virtu' -l slash_22=1+nodes=3,walltime=2
```

Cette ligne nous permet de réserver 3 nœuds avec sous-réseau en /22.

Après réservation, oarsub ouvre un shell dans lequel des variables d'environnements sont définies comme \$OAR_FILE_NODE, qui est le nom d'un fichier avec la liste des nœuds réservés, ou \$OAR_JOB_ID.

```
1 cat $OAR_FILE_NODES | uniq
2 griffon-25. nancy . gr id5000 . f r
3 griffon-5. nancy . gr id5000 . f r
4 graphene-12. nancy . gr id5000 . f r
5
6 echo $OAR_JOB_ID
7 387054
```

Pour supprimer le job et libérer les ressources, on utilise la commande oardel.

```
1 $ oardel 387054
```

Kadeploy permet de déployer des environnements personnalisés sur les nœuds réservés à l'aide d'une commande simple. Lorsque la réservation est terminée, le nœud est automatiquement restauré avec un environnement Debian ou autre. Pour déployer un environnement sur tous les nœuds réservés, il faut utiliser la commande kadeploy3

```
1 kadeploy3 -e squeeze-x64-xen -f $OAR_FILE_NODES -k $HOME/.ssh/id_rsa.pub
```

L'option -e permet de spécifier la distribution, -f permet le déploiement sur tous les nœuds et -k permet de copier la clé ssh du frontend sur les nœuds.

1.4 Répartition des tâches

Nous avons commencé par prendre en main Grid5000 durant les 2 premières semaines du projet. Pour ce faire nous avons suivi avec soin les tutoriels mis à notre disposition sur le site www.grid5000.fr.

Une fois les manipulations de bases bien assimilées. Nous nous sommes divisés en 2 sous-groupes pour tester les différents outils du projet :

- Julien et Augustin se sont chargés de Ganeti et archipel.
- Sébastien et Mathieu pour OpenXenManager et virt-manager.

Chapitre 2

Ganeti

2.1 Introduction

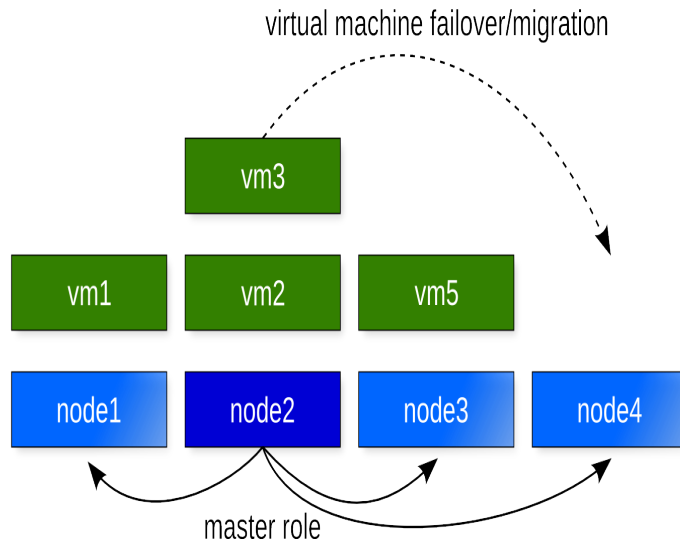


Ganeti est un outil de gestion de machines virtuelles se basant sur les technologies de virtualisation existantes comme XEN et KVM et LXC.

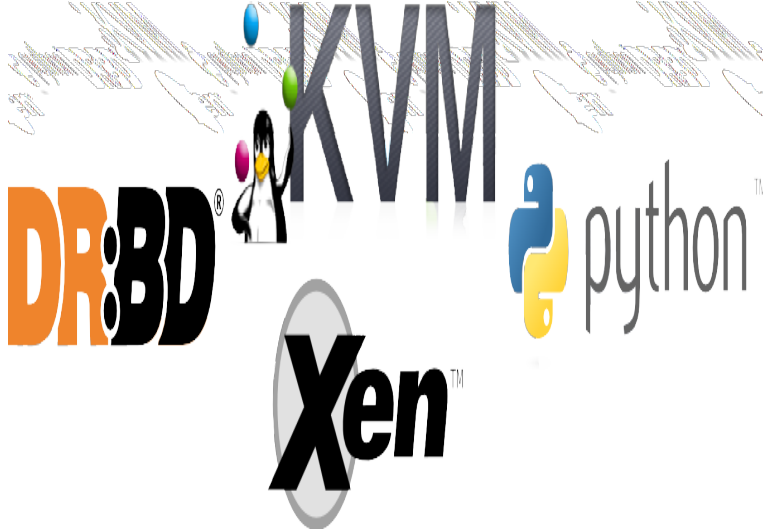
Ganeti nécessite un logiciel de virtualisation pré-installé sur les serveurs afin de pouvoir fonctionner. Une fois installé, l'outil prendra en charge la partie gestion des instances virtuelles (Xen DomU), par exemple, la gestion de création de disque, l'installation du système d'exploitation (en coopération avec les scripts d'installation du système d'exploitation spécifique), et le démarrage, l'arrêt, le basculement entre les systèmes physiques. Il a été conçu pour faciliter la gestion de cluster de serveurs virtuels et de fournir une récupération rapide et simple.

Ganeti est un manager de cluster de machine virtuelles. Il combine la virtualization et la réplication en temps réel de disque. Ganeti offre une plateforme de haute disponibilité. Ce que Ganeti peut faire d'autre :

- Migration en "live" des instances
- Souplesse face aux pannes (Redondance des données avec DRBD)
- "Cluster balancing"
- Facilité pour les réparations et les changements matériels
- Possibilité de superviser simultanément entre 1 et environ 200 hôtes physiques.



Ganeti utilise Python, Xen, KVM, DRBD, LVM, SAN, socat et Haskell. Développé par Google depuis Aout 2007



Ce Projet est sous licence GNU GPLv2.

Site du projet :

1 <http://code.google.com/p/ganeti/>

Terminologie :

- Node : Un hôte physiques
- Instance : Un machine virtuelles
- Cluster : Un groupe de node supervisés
- Job : Une opération de ganeti

2.2 Installation

2.2.1 Modification des sources

Nous avons intallé ganeti à partir de la branche testing de debian. Pour des raisons techniques le système est squeeze. Pour cela il faut ajouter les sources de testing dans le fichier /etc/apt/sources.list :

```

1 ##Wheezy
2 deb http://ftp.fr.debian.org/debian/ wheezy main contrib non-free
3 deb-src http://ftp.fr.debian.org/debian/ wheezy main contrib non-free
4
5 ## wheezy security
6 deb http://security.debian.org/ wheezy/updates main contrib non-free
7 deb-src http://security.debian.org/ wheezy/updates main contrib non-free

```

IL faut ensuite créer le fichier de préférence de apt dans le répertoire /etc/apt/apt.conf.d. Nous avons appelé le fichier 80default-distrib (le nom du fichier est libre). Il faut ajouter cette ligne au fichier qui définit la branche stable comme la branche par défaut :

```

1 APT::Default-Release "stable";

```

2.2.2 Mise à jour et installation

On peut enfin alors mettre le système à jour et installer ganeti :

```

1 apt-get update && apt-get dist-upgrade -q -y --force-yes
2 apt-get -t testing install -q -y --force-yes ganeti2 ganeti-htools ganeti-instance-
  debootstrap

```

2.3 Configuration

La configuration est l'étape la plus complexe.

2.3.1 Configuration du fichier hosts

Dans le fichier hosts il faut renseigner l'adresse et le nom complet du node primaire de cette manière :

```

1 172.16.68.10 griffon-10.nancy.grid5000.fr griffon-10

```

2.3.2 Copie des fichiers du noyau

Dans /etc/boot copier les fichiers vmlinuz-2.6.32-5-xen-amd64 et initrd.img-2.6.32-5-xen-amd64 :

```

1 cp vmlinuz-2.6.32-5-xen-amd64 vmlinuz-2.6-xenU
2 cp initrd.img-2.6.32-5-xen-amd64 initrd.img-2.6-xenU

```

2.3.3 Création du bridge xen-br0

Bien que nous utilisions eth0 comme bridge, la configuration de xen-br0 dans le fichier /etc/network/interfaces est obligatoire pour l'initialisation du cluster.

2.3.4 Création du LVM

Ganeti requiert un LVM d'au moins 20Go pour fonctionner.

Sur les nœuds de grid5000 il est possible de créer un tel LVM sur la partition /dev/sd5.

```

1 umount /dev/sda5
2 pvcreate /dev/sda5
3 vgcreate xenvg /dev/sda5

```

On a créé un VG qui se nomme xenvg sur /dev/sda5.

2.3.5 Edition de /usr/share/ganeti/os/debootstrap/common.sh

Il est nécessaire d'éditer ce fichier pour que ganeti puisse créer des instances :

Par défaut le miroir utilisé par ganeti est `http://ftp.us.debian.org/debian/`. Sur Grid5000 les depots US sont bloqués. Il faut donc indiquer les depots français. On indique aussi l'adresse du proxy de grid5000. On peut choisir la version de Debian que l'on souhaite installer. Ici nous avons opté pour squeeze. L'architecture que nous choisissons est amd64, car l'hôte la supporte. La variable : `EXTRA_PKGS` permet d'installer des paquets supplémentaires.

2.3.6 Configuration et initialisation du cluster

L'initialisation du cluster se fait avec la commande `gnt-cluster init clusterX`

```
1 #initialisation du cluster
2 gnt-cluster init --no-drbd-storage --nic-parameters link=eth0 cluster1
```

Ici nous avons précisé les options `--no-drbd-storage --nic-parameters link=eth0`. La première permet d'utiliser ganeti sans utiliser la haute disponibilité. La seconde permet de préciser un autre bridge, et d'utiliser `eth0` plutôt que `xen-br0`

Enfin il faut renseigner le `inird` et le `root_path`, cela est nécessaire pour la création des instances :

```
1 gnt-cluster modify --hypervisor-parameter xen-pvm:initrd_path='/boot/initrd.img-2.6-xenU
,
2 gnt-cluster modify --hypervisor-parameter xen-pvm:root_path='/dev/xvda1'
```

2.4 Utilisation des nodes

2.4.1 Ajouter un node

Il est possible d'ajouter un node à tout moment :

```
1 root@griffon-8: gnt-node add griffon-78.nancy.grid5000.fr
2
3 -- WARNING --
4 Performing this operation is going to replace the ssh daemon keypair
5 on the target machine (griffon-78.nancy.grid5000.fr) with the ones of the current one
6 and grant full intra-cluster ssh root access to/from it
7
8 Unable to verify hostkey of host griffon-78.nancy.grid5000.fr:
9 30:cb:8f:ec:16:6a:3b:f5:0c:2a:de:a6:4c:1d:00:19. Do you want to accept
10 it?
11 y/[n]/?: y
12 2012-03-12 07:38:09,239: MainThread Authentication to griffon-78.nancy.grid5000.fr via
    public key failed, trying password
13 root password:
14 Mon Mar 12 07:38:16 2012 - INFO: Node will be a master candidate
```

2.4.2 Reconfigurer un node

Il est aussi possible de reconfigurer un node déjà présent :

```
1 root@griffon-8: gnt-node add --readd griffon-78.nancy.grid5000.fr
2
3 Unable to verify hostkey of host griffon-78.nancy.grid5000.fr:
4 6c:10:44:28:e2:2c:fc:7f:d4:5e:a3:bd:83:2c:b2:97. Do you want to accept
5 it?
```

```

6 y/[n]/?: y
7 Mon Mar 12 07:39:36 2012 - INFO: Readding a node, the offline/draind flags were reset
8 Mon Mar 12 07:39:36 2012 - INFO: Node will be a master candidate

```

2.4.3 Roles des nodes et opérations

Les différents nodes ainsi que leurs rôles :

- Master node : Utilise ganeti-masterd, rapi, noded and confd. Peut accueillir des instances. Toutes les opérations de supervision s'effectuent sur ce node.
- Master candidates : Possède une copie complète de la configuration, peut prendre le rôle de Master. Utilise ganeti-confd and noded. Peut accueillir des instances.
- Regular node : Ne peuvent pas devenir Master et ne possèdent qu'une partie de la configuration. Peut accueillir des instances.
- Offline node : Ces nodes sont hors-ligne. Ne peut pas accueillir des instances.

Promouvoir un nœud au rang de master :

Il faut d'abord révoquer le rang de master du node principal, sur un node master-candidate :

```

1 gnt-cluster master-failover
2
3 root@griffon-81: gnt-instance list
4 Failure: prerequisites not met for this operation:
5 This is not the master node, please connect to node 'griffon-8.nancy.grid5000.fr' and
   rerun the command
6
7 root@griffon-81: gnt-cluster master-failover
8
9 root@griffon-81: gnt-node list
10 Node                DTotal DFree MTotat MNode MFree Pinst Sinst
11 griffon-8.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0 0
12 griffon-78.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0 0
13 griffon-81.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0 0

```

On voit que le node griffon-81 n'était pas master avant l'utilisation de la commande. Ensuite il est possible d'exécuter les commandes master.

Passer un nœud en master-candidate :

```

1 root@griffon-81: gnt-node modify -C yes griffon-8.nancy.grid5000.fr
2 Modified node griffon-8.nancy.grid5000.fr
3 - master_candidate -> True
4 - drained -> False

```

Passer un node en status drained :

```

1 root@griffon-81: gnt-node modify -D yes griffon-8.nancy.grid5000.fr
2 Modified node griffon-8.nancy.grid5000.fr
3 - master_candidate -> False
4 - drained -> True

```

Passer un node en offline :

```
1 root@griffon-81: gnt-node modify -O yes griffon-8.nancy.grid5000.fr
2 Modified node griffon-8.nancy.grid5000.fr
3   - master_candidate -> False
4   - offline -> True
```

Passer un node en mode regular (remise à zero de tous les flags) :

```
1 root@griffon-81: gnt-node modify -O no -D no -C no griffon-8.nancy.grid5000.fr
2 Mon Mar 12 08:26:01 2012 - INFO: Ignoring request to unset flag master_candidate,
   already unset
3 Mon Mar 12 08:26:01 2012 - INFO: Ignoring request to unset flag drained, already unset
4 Mon Mar 12 08:26:01 2012 - INFO: Auto-promoting node to master candidate
5 Mon Mar 12 08:26:01 2012 - WARNING: Transitioning node from offline to online state
   without using re-add. Please make sure the node is healthy!
6 Modified node griffon-8.nancy.grid5000.fr
7   - master_candidate -> True
8   - offline -> False
```

Le node est de nouveau en master-candidate comme à l'origine.

2.4.4 Supprimer un node :

```
1 root@griffon-81: gnt-node list
2 Node                DTotal DFree MTotal MNode MFree Pinst Sinst
3 griffon-8.nancy.grid5000.fr 283.2G 282.2G 16.0G 965M 14.7G 1    0
4 griffon-78.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0    0
5 griffon-81.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0    0
6
7 root@griffon-81: gnt-node remove griffon-78.nancy.grid5000.fr
8
9 root@griffon-81: gnt-node list
10 Node                DTotal DFree MTotal MNode MFree Pinst Sinst
11 griffon-8.nancy.grid5000.fr 283.2G 282.2G 16.0G 965M 14.7G 1    0
12 griffon-81.nancy.grid5000.fr 283.2G 283.2G 16.0G 965M 14.8G 0    0
```

Le node griffon-78 à bien été effacer du cluster.

2.4.5 Manipulation du stockage :

Faire la liste des volumes sur lesquels sont les instances :

```
1 root@griffon-81: gnt-node volumes
2 Node                PhysDev VG      Name                               Size
3 Instance
4 griffon-81.nancy.grid5000.fr /dev/sda5 xenvg 4b328fb6-1cc9-4599-9523-2c6a8cf7b861.disk0
   1000M instance4
5 griffon-8.nancy.grid5000.fr /dev/sda5 xenvg 8a85c87b-39fa-4e88-886b-a60c8706cc65.disk0
   1000M instance2
6 griffon-8.nancy.grid5000.fr /dev/sda5 xenvg d84f3842-04d5-4951-aded-b5ea76f19681.disk0
   1000M instance1
7 griffon-8.nancy.grid5000.fr /dev/sda5 xenvg ddfea3fc-d7b1-4a06-ae37-3094b1f0de11.disk0
   1000M instance3
```

Il est possible de lancer une réparation sur les volume de stockage :


```

1 root@griffon-81:~# gnt-node repair-storage griffon-8.nancy.grid5000.fr lvm-vg xenvg
2 Mon Mar 12 09:56:23 2012 Repairing storage unit 'xenvg' on griffon-8.nancy.grid5000.fr
...

```

Cela équivaut à `vgreduce --removemissing`.

2.5 Utilisation des instances

2.5.1 Ajouter une instance

```

1 root@graphene-11: gnt-instance add -n graphene-11.nancy.grid5000.fr -o debootstrap+
  default -t plain -s 1000 instance1
2
3 Sat Mar 10 16:52:35 2012 * disk 0, vg xenvg, name 99370e22-9421-40d6-8d9d-59bb6ecfa959.
  disk0
4 Sat Mar 10 16:52:35 2012 * creating instance disks...
5 Sat Mar 10 16:52:36 2012 adding instance instance1 to cluster config
6 Sat Mar 10 16:52:36 2012 - INFO: Waiting for instance instance1 to sync disks.
7 Sat Mar 10 16:52:37 2012 - INFO: Instance instance1's disks are in sync.
8 Sat Mar 10 16:52:37 2012 * running the instance OS create scripts...
9 Sat Mar 10 16:54:05 2012 * starting instance...
10
11 root@graphene-11: gnt-instance list
12 Instance Hypervisor OS Primary_node Status Memory
13 instance1 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M

```

L'instance est bien créée sur le nœud. Il est possible à partir du maître de créer des instances sur n'importe quel nœud d'un cluster. Il est aussi possible de créer une instance primaire sur un nœud et une instance secondaire sur un autre.

2.5.2 Supprimer une instance

```

1 root@graphene-11: gnt-instance remove instance1
2 This will remove the volumes of the instance instance1 (including
3 mirrors), thus removing all the data of the instance. Continue?
4 y/[n]/?: y
5
6 root@graphene-11: gnt-instance list
7 Instance Hypervisor OS Primary_node Status Memory

```

L'instance a bien été supprimée. Cette commande supprime l'instance quelque soit le ou les nœuds où elle a été créée.

Arrêt et démarrage d'une instance

```

1 root@graphene-11: gnt-instance list
2 Instance Hypervisor OS Primary_node Status Memory
3 instance1 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M
4 instance2 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M
5
6 root@graphene-11: gnt-instance shutdown instance2
7 Waiting for job 21 for instance2...
8
9 root@graphene-11: gnt-instance list
10 Instance Hypervisor OS Primary_node Status Memory
11 instance1 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M
12 instance2 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr ADMIN_down -

```

```

1 root@graphene-11: gnt-instance startup instance2
2 Waiting for job 26 for instance2...
3
4 root@graphene-11: gnt-instance list
5 Instance Hypervisor OS Primary_node Status Memory
6 instance1 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M
7 instance2 xen-pvm debootstrap+default graphene-11.nancy.grid5000.fr running 128M

```

Le status de instance2 est de nouveau "running" ce qui signifie qu'elle est en fonctionnement.
Interroger les instances :

```

1 root@graphene-11: gnt-instance info instance1
2
3 Instance name: instance1
4 UUID: 3c3bd5ac-a261-4cba-a7f3-6cc74e49ce4e
5 Serial number: 2
6 Creation time: 2012-03-10 17:06:56
7 Modification time: 2012-03-10 17:07:04
8 State: configured to be up, actual state is up
9 Nodes:
10   - primary: graphene-11.nancy.grid5000.fr
11   - secondaries:
12 Operating system: debootstrap+default
13 Allocated network port: None
14 Hypervisor: xen-pvm
15   - blockdev_prefix: default (sd)
16   - bootloader_args: default ()
17   - bootloader_path: default ()
18   - initrd_path: default (/boot/initrd.img-2.6-xenU)
19   - kernel_args: default (ro)
20   - kernel_path: default (/boot/vmlinuz-2.6-xenU)
21   - root_path: default (/dev/sda1)
22   - use_bootloader: default (False)
23 Hardware:
24   - VCPUs: 1
25   - memory: 128MiB
26   - NICs:
27     - nic/0: MAC: aa:00:00:d8:c6:8a, IP: None, mode: bridged, link: xen-br0
28 Disk template: plain
29 Disks:
30   - disk/0: lvm, size 1000M
31     access mode: rw
32     logical_id: xenvg/3fe11555-edcd-40dc-bf63-f3fb749825bb.disk0
33     on primary: /dev/xenvg/3fe11555-edcd-40dc-bf63-f3fb749825bb.disk0 (254:0)

```

Cette commande édite les informations relatives à l'instance.

Import et export d'instances :

Export :

```

1 root@graphene-100: gnt-backup export -n graphene-143.nancy.grid5000.fr instance1
2 Sun Mar 11 14:06:04 2012 Shutting down instance instance1
3 Sun Mar 11 14:08:07 2012 Creating a snapshot of disk/0 on node graphene-100.nancy.
  grid5000.fr
4 Sun Mar 11 14:08:08 2012 Starting instance instance1
5 Sun Mar 11 14:08:09 2012 Exporting snapshot/0 from graphene-100.nancy.grid5000.fr to
  graphene-143.nancy.grid5000.fr
6 Sun Mar 11 14:08:13 2012 snapshot/0 is now listening, starting export
7 Sun Mar 11 14:08:17 2012 snapshot/0 is receiving data on graphene-143.nancy.grid5000.fr
8 Sun Mar 11 14:08:17 2012 snapshot/0 is sending data on graphene-100.nancy.grid5000.fr

```

```

9 Sun Mar 11 14:08:22 2012 snapshot/0 sent 14M, 2.8 MiB/s
10 Sun Mar 11 14:08:38 2012 snapshot/0 finished receiving data
11 Sun Mar 11 14:08:38 2012 snapshot/0 finished sending data
12 Sun Mar 11 14:08:38 2012 Removing snapshot of disk/0 on node graphene-100.nancy.grid5000
    .fr
13 Sun Mar 11 14:08:39 2012 Finalizing export on graphene-143.nancy.grid5000.fr
14 Sun Mar 11 14:08:40 2012 Removing old exports for instance instance1

```

L'instance a bien été exporté dans graphene-143.nancy.grid5000.fr

Il est tout à fait possible d'exporter une instance sans la redémarrer en utilisant l'option :
 --noshutdown.

Import :

```

1 root@graphene-100: gnt-instance remove instance1
2 This will remove the volumes of the instance instance1 (including
3 mirrors), thus removing all the data of the instance. Continue?
4 y/[n]/?: y
5
6 root@graphene-100: gnt-instance list
7 Instance Hypervisor OS Primary_node Status Memory
8 instance2 xen-pvm debootstrap+default graphene-100.nancy.grid5000.fr running 128M
9 instance3 xen-pvm debootstrap+default graphene-143.nancy.grid5000.fr running 128M

```

Instance1 a été supprimée du cluster.

```

1 root@graphene-100: gnt-backup import -n graphene-100.nancy.grid5000.fr --src-node=
    graphene-143.nancy.grid5000.fr -t plain instance1
2 Sun Mar 11 14:22:29 2012 * disk 0, vg xenvg, name a4cc7447-5ed7-4417-b222-d33a0c2842a0.
    disk0
3 Sun Mar 11 14:22:29 2012 * creating instance disks...
4 Sun Mar 11 14:22:30 2012 adding instance instance1 to cluster config
5 Sun Mar 11 14:22:30 2012 - INFO: Waiting for instance instance1 to sync disks.
6 Sun Mar 11 14:22:30 2012 - INFO: Instance instance1's disks are in sync.
7 Sun Mar 11 14:22:30 2012 * running the instance OS import scripts...
8 Sun Mar 11 14:22:30 2012 Exporting disk/0 from graphene-143.nancy.grid5000.fr to
    graphene-100.nancy.grid5000.fr
9 Sun Mar 11 14:22:34 2012 disk/0 is now listening, starting export
10 Sun Mar 11 14:22:37 2012 disk/0 is receiving data on graphene-100.nancy.grid5000.fr
11 Sun Mar 11 14:22:37 2012 disk/0 is sending data on graphene-143.nancy.grid5000.fr
12 Sun Mar 11 14:22:42 2012 disk/0 sent 34M, 6.0 MiB/s, 19%, ETA 23s
13 Sun Mar 11 14:23:08 2012 disk/0 finished sending data
14 Sun Mar 11 14:23:14 2012 disk/0 finished receiving data

```

On importe instance1 depuis graphene-143.nancy.grid5000.fr.

```

1 root@graphene-100: gnt-instance list
2 Instance Hypervisor OS Primary_node Status Memory
3 instance1 xen-pvm debootstrap+default graphene-100.nancy.grid5000.fr ADMIN_down -
4 instance2 xen-pvm debootstrap+default graphene-100.nancy.grid5000.fr running 128M
5 instance3 xen-pvm debootstrap+default graphene-143.nancy.grid5000.fr running 128M

```

L'instance a bien été importée.

```

1 root@graphene-100: gnt-instance startup instance1
2 Waiting for job 32 for instance1...
3
4 root@graphene-100: gnt-instance list
5 Instance Hypervisor OS Primary_node Status Memory
6 instance1 xen-pvm debootstrap+default graphene-100.nancy.grid5000.fr running 128M
7 instance2 xen-pvm debootstrap+default graphene-100.nancy.grid5000.fr running 128M
8 instance3 xen-pvm debootstrap+default graphene-143.nancy.grid5000.fr running 128M

```

L'instance est fonctionnelle sur graphene-100.nancy.grid5000.fr

Il est aussi possible d'importer une instance étrangère à ganeti dont le disque est déjà dans un LVM, sans avoir à le recopier.

```
1 gnt-instance add -t plain -n HOME_NODE ... \  
2 --disk 0:adopt=lv_name[,vg=vg_name] INSTANCE_NAME
```

Connexion à la console d'une instance :

```
1 gnt-instance console instance5
```

Une fois connecté à la console, l'utilisateur par défaut est root et il n'y a pas de mots de passe.

Pour ce projet toutes nos instances sont des Debian 6.0. Évidemment il est possible de créer des instances autres que Debian. Nous avons créé les instances à partir de debootstrap, d'autres méthodes existent. On peut par exemple créer des instances à partir d'une image ISO. Ce qui permet d'avoir une grande variété d'instances.

Chapitre 3

OpenXENManager

3.1 Présentation

XenseMaking Project développe un client lourd, ainsi qu'un client web, pour manager XenServer. C'est un clone du XenCenter, qui fonctionne avec Linux, BSD, Windows et MacOSX, alors que le XenCenter ne fonctionne qu'avec Windows. OpenXenManager/OpenXenCenter un le client lourd qui permet de manager XenServer. Il a été développé en Python avec pygtk et gtk-vnc. Les fonctionnalités actuellement implémentées sont les suivantes :

- monitoring des machines virtuelles - accès à la console des machines virtuelles
- opérations d'administration (démarrage, arrêt, reboot, ...)
- création de machines virtuelles

3.2 Installation

Pour l'installation nous avons besoin des paquets suivant :

```
1 apt-get install subversion bzip2 python-glade2 python-gtk-vnc shared-mime-info graphviz
```

On télécharge la dernière version d'openxenmanager dans le dépôt subversion

```
1 svn co https://openxenmanager.svn.sourceforge.net/svnroot/openxenmanager openxenmanager
```

On se déplace dans le répertoire trunk :

```
1 cd openxenmanager/trunk
```

Finalement on lance openxenmanager avec la commande suivante

```
1 python window.py
```

Une interface graphique d'openxenmanager apparaît.

3.3 Problèmes rencontrés avec OpenXenManager

A l'origine OpenXenManager est destiné à manager XenServer qui fonctionne uniquement avec le système d'exploitation windows.

Malheureusement nous travaillons dans un environnement différent de windows, notre premier problème fut de trouver un équivalent sous linux. Après plusieurs jours de recherche nous avons trouvé un programme susceptible de fonctionner avec OpenXenManager qui s'appelle Xen cloud Platform. Très peu de documentation est disponible sur internet, nous avons suivi un tutoriel qui nous semblait être l'un des mieux explicatifs pour essayer de l'installer mais celui-ci n'a pas fonctionné. Nous allons vous présenter un extrait des étapes de la configuration ainsi que les problèmes rencontrés.

3.3.1 Préparation du système

La première étape consistait à préparer le système pour l'installation :

- Rajouter des dépôts dans le source.list
- Récupérer la clé pour y accéder.

Problème rencontrés durant cette étape

Après avoir rajouter les dépôts dans le /etc/apt/source.listes :

```
1 deb http://ppa.launchpad.net/ubuntu-xen-org/xcp-unstable/ubuntu oneiric main
2 deb-src http://ppa.launchpad.net/ubuntu-xen-org/xcp-unstable/ubuntu oneiric main
```

Et utilisé la commande suivante :

```
1 apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 9273A937
```

On obtient le message suivant :

```
1 root@griffon-61:~# apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 9273A937
2 Executing: gpg --ignore-time-conflict --no-options --no-default-keyring --secret-
   keyring /tmp/tmp.BoYHj5Rkxz --trustdb-name /etc/apt/trustdb.gpg --keyring /etc/apt/
   trusted.gpg --primary-keyring /etc/apt/trusted.gpg --keyserver keyserver.ubuntu.com
   --recv-keys 9273A937
3 gpg: requesting key 9273A937 from hkp server keyserver.ubuntu.com
4 gpg: keyserver timed out
5 gpg: keyserver receive failed: keyserver error
```

Il semblerait que nous ne pouvions pas accéder au serveur pour récupérer la clé.

Ensuite nous devons installer python-software-properties une dépendance requise pour que XCP fonctionne. En faisant un

```
1 apt-get update
```

avant installer python-software-properties on obtient le message suivant :

```
1 W: GPG error: http://ppa.launchpad.net oneiric Release: The following signatures couldn't
   be verified because the public key is not available: NO_PUBKEY 79B578FB9273A937
```

Comme nous n'avons pas récupéré la clé il nous est impossible d'installer la dépendance.

3.3.2 Installation de XCP

Il faut tout d'abord ajouter les dépôts d'XCP avec la commande suivante :

```
1 add-apt-repository ppa:ubuntu-xen-org/xcp-unstable
```

Problèmes rencontrés durant cette étape

Il s'en suit un echec de la commande.

```
1 root@griffon-61:~# add-apt-repository ppa:ubuntu-xen-org/xcp-unstable
2 Traceback (most recent call last):
3   File "/usr/bin/add-apt-repository", line 88, in <module>
4     ppa_info = get_ppa_info_from_lp(user, ppa_name)
5   File "/usr/lib/python2.7/dist-packages/softwareproperties/ppa.py", line 80, in
     get_ppa_info_from_lp
6     curl.perform()
7   pycurl.error: (28, 'connect() timed out!')
```

Il nous est impossible d'ajouter les dépôts pour récupérer XCP. Pour les étapes suivantes nous n'avons rencontré aucuns problèmes :

- Installation de dépendences pour XCP
- Installation de XCP

```
1 apt-get update
2 apt-get install openvswitch-datapath-dkms
```

Tout s'installe correctement, ce n'est que lors de l'installation de Xen Cloud Platform que plusieurs problèmes sont survenuent.

```
1 apt-get update
2 apt-get install xcp-xapi
3 apt-get install xcp-xe
```

```
1 Starting the XCP networking daemon: ..... * failed to start xcp
  -networkd.
2
3 invoke-rc.d: initscript xcp-networkd, action "start" failed.
4 dpkg: error processing xcp-networkd (--configure):
5 subprocess installed post-installation script returned error exit status 1
6 dpkg: dependency problems prevent configuration of xcp-xapi:
7 xcp-xapi depends on xcp-networkd; however:
8 Package xcp-networkd is not configured yet.
9 dpkg: error processing xcp-xapi (--configure):
10 dependency problems - leaving unconfigured
11 dpkg: dependency problems prevent configuration of xcp-guest-templates:
12 xcp-guest-templates depends on xcp-xapi; however:
13 Package xcp-xapi is not configured yet.
14 dpkg: error processing xcp-guest-templates (--configure):
15 dependency problems - leaving unconfigured
16 Setting up xcp-vncterm (0.1.1-1ubuntu1) ...
17 Processing triggers for libc-bin ...
18 ldconfig deferred processing now taking place
19 No apport report written because the error message indicates its a followup error from a
  previous failure.
20 No apport report written because the error message indicates its a followup error from a
  previous failure.
21 Processing triggers for initramfs-tools ...
22 update-initramfs: Generating /boot/initrd.img-3.0.0-12-server
23 Errors were encountered while processing:
24 xcp-networkd
25 xcp-xapi
26 xcp-guest-templates
27 E: Sub-process /usr/bin/dpkg returned an error code (1)
```

Suite aux nombreux problèmes rencontrés sur ce sujet nous avons décidé de passer sur notre second projet virt-manager.

3.4 Xen Cloud Platform

Le fait de passer au second sujet nous a permis de débloquer la situation avec OpenXenManager. En effet nous avons pu installer XCP grâce à virt-manager dans une machine virtuelle avec l'aide d'une image iso d'XCP téléchargé depuis le site officiel. L'installation est simple, depuis la machine virtuelle on boot sur sur l'image iso et l'on suis les étapes suivantes.

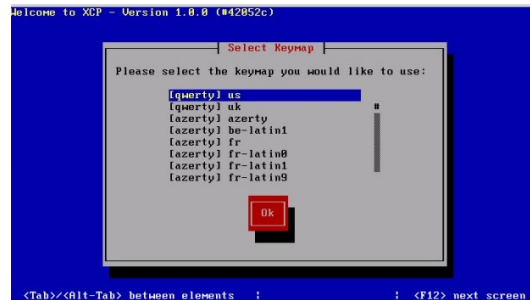


FIGURE 3.1 – On choisit le type de clavier



FIGURE 3.2 – On accepte la licence



FIGURE 3.3 – Choix du disque d'installation



FIGURE 3.4 – Choix de la source d'installation

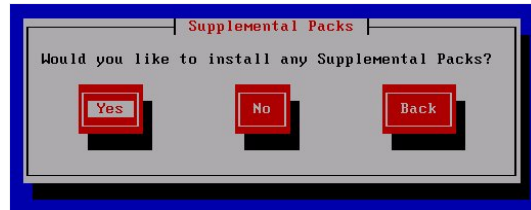


FIGURE 3.5 – Paquets additionnels

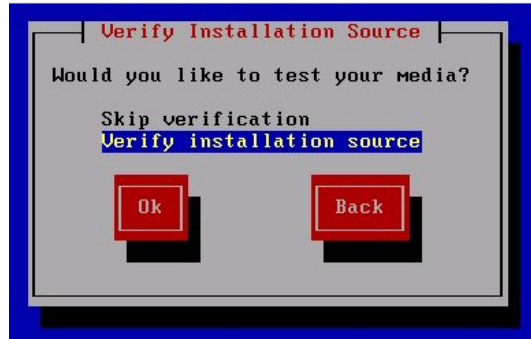


FIGURE 3.6 – Vérification de la source d'installation



FIGURE 3.7 – Choix du password

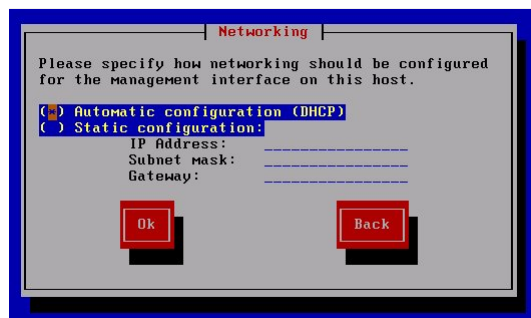


FIGURE 3.8 – Configuration du réseau

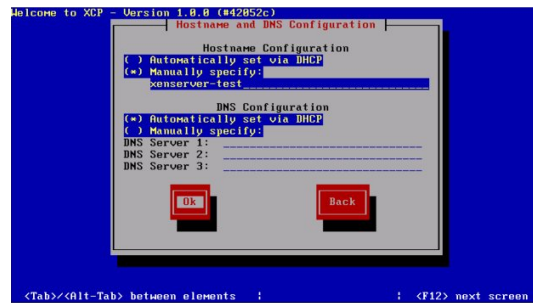


FIGURE 3.9 – Configuration du DNS

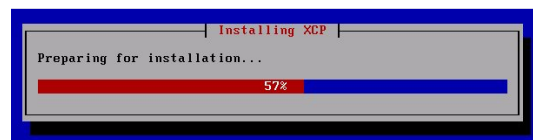


FIGURE 3.10 – Installation



FIGURE 3.11 – Installation complète

Chapitre 4

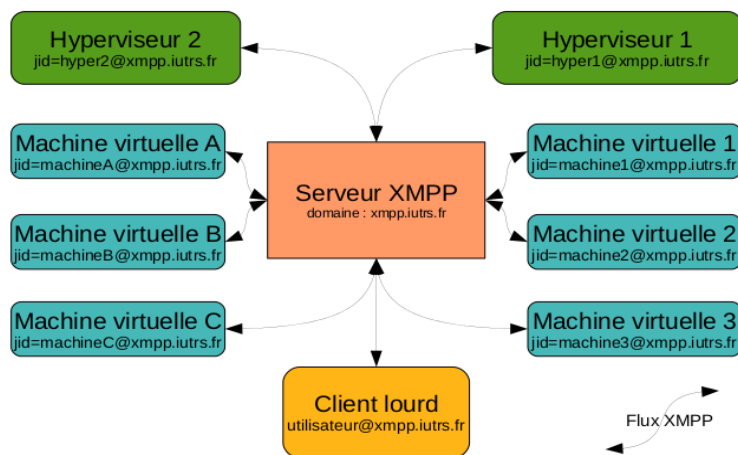
Archipel Project

4.1 Introduction

Archipel repose totalement sur libvirt comme API de gestion des machines virtuelles. De fait, l'outil s'impose d'être de plus haut niveau. Pour la gestion des utilisateurs, on peut directement utiliser un serveur XMPP existant pour authentifier les personnes.

4.2 Architecture

L'utilisation d'un serveur XMPP peut sembler étrange. L'application devra envoyer et recevoir des informations avec les hyperviseurs et les machines virtuelles. Plutôt que de réécrire un n-ième protocole, de l'implémenter, de devoir le déboguer et le sécuriser, l'équipe d'Archipel avec XMPP a fait le choix de la réutilisation. Ce protocole est défini par des RFC, et possède de nombreuses implémentations.



Il suffit donc d'avoir un client adapté pour les hyperviseurs et pour les machines virtuelles. Les machines virtuelles, à un instant donné étant localisées sur un serveur, c'est au niveau de l'hyperviseur que reposera la création des n instances de clients pour chacune des n machines virtuelles. Au niveau de l'application de l'utilisateur, il y a également un client XMPP pour que les messages soient transmis aux entités.

En terme logiciel, un agent est installé sur chaque hyperviseur. Cet agent est scindé en deux parties :

- une incluant un client XMPP chargé de se connecter au serveur XMPP pour lui signaler sa présence.
- l'autre, plus complexe, est chargée de créer autant d'agents qu'il y a de machines virtuelles définies, chacun de ces agents intègre également un client XMPP.

Du côté du client utilisateur, l'interface intègre un client XMPP. Toutes les communications bénéficient donc des fonctionnalités du serveur XMPP, par exemple le roster (liste des contacts autorisés) qui donne une certaine sécurité dans les dialogues entre entités. On « ajoute » donc un contact dans son roster, qui peut être soit une machine virtuelle ou un hyperviseur. Lorsque dans l'interface client, on demande un démarrage de la VM (« Virtual Machine », en français machine virtuelle), un stanza (un message XMPP) est envoyé vers le serveur XMPP. Celui-ci vérifie que l'émetteur est dans le roster du destinataire, et lui transmettra alors le message s'il est autorisé. L'agent, côté machine virtuelle, va recevoir ce message et l'interpréter, donc lancer via l'API libvirt le signal start.

Les agents, qu'il s'agisse de ceux liés à un hyperviseur ou à une machine virtuelle, exploitent la librairie libvirt via le binding python. En particulier, c'est la partie events de libvirt qui est utilisée. De cette manière, même si une interaction parallèle est effectuée sur les hyperviseurs (par exemple, l'utilisation de virsh ou de virt-manager pour démarrer une entité, ajouter un nouveau réseau, ou créer une nouvelle VM par un script), les agents seront informés et pourront remonter le nouvel état via un message XMPP. Ce choix laisse la porte ouverte à une intégration fine dans un système existant : si vous aviez vos propres scripts virsh pour créer des machines virtuelles, l'adoption d'Archipel ne vous force nullement à renoncer à votre travail déjà en place !

4.3 Haute disponibilité et montée en charge

Dans les solutions de gestions d'hyperviseurs, un critère souvent retenu est la Haute Disponibilité (HA). Mais tout le monde ne met pas les mêmes éléments derrière ces mots. En terme de HA au niveau des machines virtuelles, Archipel se repose complètement et uniquement sur libvirt. Au niveau d'Archipel lui même, la haute disponibilité concerne l'échange des messages entre les entités, donc la haute disponibilité du serveur XMPP lui même. Une utilisation d'une solution cluster de serveurs XMPP répond donc à la demande. Ce mode cluster permet aussi de se rassurer au niveau de la montée en charge du serveur car la multiplication des VM et des hyperviseurs va augmenter le nombre de stanza échangés.

Au niveau des messages échangés, la notification de toutes les entités n'est pas nécessaire, ni voulue, pour des raisons de charge de messages. Une solution consisterait à stocker du côté de l'agent, les entités et les messages qu'elles doivent recevoir, mais il faudrait alors maintenir des états. L'architecture a recours à un mécanisme très intéressant du côté du serveur XMPP. Il s'agit de la partie PubSub (PUBlish- SUBscribe) : cela fonctionne comme les groupes de multicast au niveau réseau IP. Les entités qui souhaitent diffuser des informations à plusieurs entités publient via une entrée dans le service PubSub du serveur, et les entités qui souhaitent recevoir les messages pour cette entrée s'y abonnent. Ainsi, c'est encore une fois le serveur XMPP qui va faire le travail de diffuser aux entités abonnées les messages. Cela donne un flux optimal de messages, et simplifie la programmation des agents.

4.4 Sécurité

Dans une telle plate-forme, la gestion de la sécurité est importante. L'utilisation de XMPP en tant que protocole permet déjà d'être rassuré sur l'intégrité des messages échangés : XMPP est issue d'un processus de validation et de standardisation via les RFC et est implémenté depuis de nombreuses années.

La sécurité qui permet de savoir quel utilisateur peut dialoguer avec une entité repose d'abord sur le roster de celle-ci, car comme le roster est stocké côté serveur XMPP, c'est sans doute la

meilleure solution qui puisse être retenue. Il est donc possible de définir clairement quel utilisateur peut dialoguer avec une machine virtuelle ou un hyperviseur donnés, et également de savoir qui a accès à une entité donnée. Archipel va encore plus loin dans la gestion de la sécurité. Comme ce sont des stanza qui définissent les demandes entre les entités, et que cela se traduit par des appels à l'API de libvirt, un filtrage coté agent est en place, il permet d'autoriser ou non certaines actions. Ainsi, actuellement, 110 rôles possibles sont définis et il est possible d'attribuer à un utilisateur, l'accès ou non à chacun de ces rôles. Par exemple, on peut donner à un webmestre, le droit d'accéder à la « console VNC » de sa machine virtuelle, ainsi que l'action start et stop. Ou à un développeur, le droit de prendre des instantanés d'une machine virtuelle, et de les restaurer. Dans un soucis de séparer les utilisateurs des entités machines virtuelles et hyperviseurs, on peut aller encore plus loin (et s'intégrer encore mieux dans le système d'information). Les serveurs XMPP sachant dialoguer ensemble, via une autorisation explicite au niveau du serveur pour autoriser le S2S (Server to Server), il est possible d'utiliser un autre serveur XMPP existant pour les utilisateurs (comme cela, ils peuvent utiliser par exemple une authentification liée à leur ENT). Les messages transitent alors par les deux serveurs successivement avant d'arriver aux entités coté Archipel.

En terme de sécurité, là encore, Archipel se base sur l'existant, et ajoute de la finesse dans la délégations de droits.

4.5 Fonctionnalités

Archipel étant encore jeune, les fonctionnalités les plus évidentes ont été mises en place. Le système de module permet d'ajouter aisément de nouvelles fonctions. Tout ce qui est nécessaire à un travail quotidien existe.

Actuellement, la plupart des fonctions disponibles via l'interface virt-manager sont déjà en place : définition d'une nouvelle VM, manipulations du réseau et du stockage, accès à la console VNC, gestions des snapshots, etc... Les opérations de migration sont également prises en charge. Les nouvelles possibilités sont celles qu'on attend d'un logiciel de management d'hyperviseurs : reporting sur l'état de hyperviseur, création de nouvelles machines à partir de flux RSS (VMCast), planifications de tâches, gestions des droits des différents utilisateurs, création d'une machine avec détection automatique du serveur le moins chargé, etc... Les développeurs sont pleins d'idées, il même est prévu, par exemple, un module de facturation !

La mise en place d'Archipel est très simple : il faut installer un serveur XMPP avec une configuration correcte, puis installer sur chaque agent, un agent écrit en python. Cet agent, une fois correctement configuré, va se cloner pour s'exécuter pour toutes les machines virtuelles, et va joindre le serveur XMPP. A partir de ce moment là, Archipel est exploitable. Les machines virtuelles déjà en place bénéficient des fonctionnalités d'Archipel (sauf au niveau de la gestion disque et migration). On ne peut pas faire beaucoup plus simple.

Et pour couronner le tout, l'application client n'est pas un client lourd standard. Tout le client est en fait une application HTML5, en Javascript. C'est ce qui déconcerte généralement les premiers utilisateurs : il n'y a pas de serveur web où l'application s'exécute. Il faut juste un serveur web pour stocker les fichiers (dans certains cas et avec le navigateur chrome uniquement, il est même possible d'exécuter l'application depuis les fichiers stockés en local). Une fois chargé, c'est uniquement le Javascript qui fait tourner tout le client. Le framework utilisé est Cappuccino, qui donne un aspect très MacOSX. Le client XMPP y est également intégré. Donc quand vous utilisez votre client, vous dialoguez directement vers le serveur jabber !

4.6 Conclusion

Archipel s'intègre bien dans le courant UNIX : Keep It Simple, Stupid. D'autres solutions préfèrent construire des monstres d'infrastructure. Se baser sur libvirt permet de s'assurer d'avoir la main via virsh ou virt-manager, en cas de soucis de client, d'agent, ou même si le serveur XMPP

pose problème, cela permet de dormir tranquille.

Cependant Archipel a encore quelques faiblesses. A force d'exploiter la robustesse de XMPP, peu de serveurs implémentent tout ce qui est nécessaire. Ejabberd est pour l'instant le seul serveur XMPP qui est recommandé. Archipel a également mis en évidence un certain nombre de bugs dans libvirt. En particulier la gestion de Xen et de Vmware dans libvirt reste problématique, et rend inopérant Archipel (la balle est dans le camp des développeurs de libvirt). Enfin Archipel propose déjà un certain nombre de fonctionnalités qui le rend largement utilisable, et il reste de la place pour l'innovation.

La mise en place n'affecte pas l'existant, elle est relativement aisée à effectuer, ce qui permet de faire une migration en douceur.

Enfin, Archipel répond a un besoin récurrent mais simple : pouvoir donner un accès restreint à certaines machines pour certains utilisateurs. La délégation de droits est simple à effectuer. De plus, avec une interface conviviale et sans client lourd, en HTML5, il n'y a pas de problème pour donner un accès à des non informaticiens.

Chapitre 5

Virt-manager

5.1 Pré-requis et considérations pour les hôtes

Divers facteurs doivent être considérés avant de créer des hôtes virtualisés.

Performance

Les hôtes virtualisés doivent être déployé et configuré en fonction de leurs tâches prévues. Certains systèmes (par exemple, les hôtes ou sont hébergés des serveur de base de données) ont besoin de performances plus élevées que d’habitude ; Les hôtes peuvent exiger plus de CPU ou de mémoire attribué en fonction de leur rôle,et de l’utilisation futur qu’il pourrait avoir. projeté la charge du système.

Stockage

Certains hôtes peuvent avoir besoin d’une plus grande priorité d’accès au stockage, de disques plus rapides, ou peuvent exiger un accès exclusif à des zones de stockage. La quantité de stockage utilisée par les hôtes doit être régulièrement surveillée et prise en compte lors du déploiement et le maintien de stockage.

Mise en réseau et l’infrastructure du réseau

En fonction de notre environnement, certains hôtes pourraient exiger des liens réseau plus rapides que d’autres hôtes. La bande passante ou de latence sont souvent des facteurs à prendre en compte lors du déploiement et le maintenance des hôtes.

5.2 Installation côté serveur

Cette partie est facile, un simple apt-get install suffit. Nous installons le paquet qui communique avec Xen et remonte les informations au client virt-manager.

```
1 apt-get install libvirt-bin
```

Du coté de Xen, nous devons vérifier qu’il peut communiquer avec libvirt.

Libvirt accède aux données de Xen via un socket unix. La configuration consiste à activer cette option dans Xen et à relancer les services. Nous éviterons ainsi l’erreur *libvirtError : internal error failed to connect to xend* dont on trouve peu d’explication sur le net.

Un autre bug rencontré nécessite la correction d’un lien symbolique et l’installation de qemu.

```
1 # Installation de qemu-dm pour crer des machines virtuelles en mode full virtualis
2 apt-get install xen-qemu-dm-4.0 -y
3
```

```

4 # Correction d'un bug de qemu qui invalidait un lien symbolique
5 mkdir /usr/lib64/xen
6 mkdir /usr/lib64/xen/bin
7 cd /usr/lib64/xen/bin
8 ln -s /usr/lib/xen-4.0/bin/qemu-dm

```

On édite le fichier de configuration xen

```

1 nano /etc/xen/xend-config.sxp

```

on active, ou on rajoute la ligne suivante

```

1 (xend-unix-server yes)

```

Enfin on relance le service xen avec `/etc/init.d/xend restart`

5.3 Installation côté client

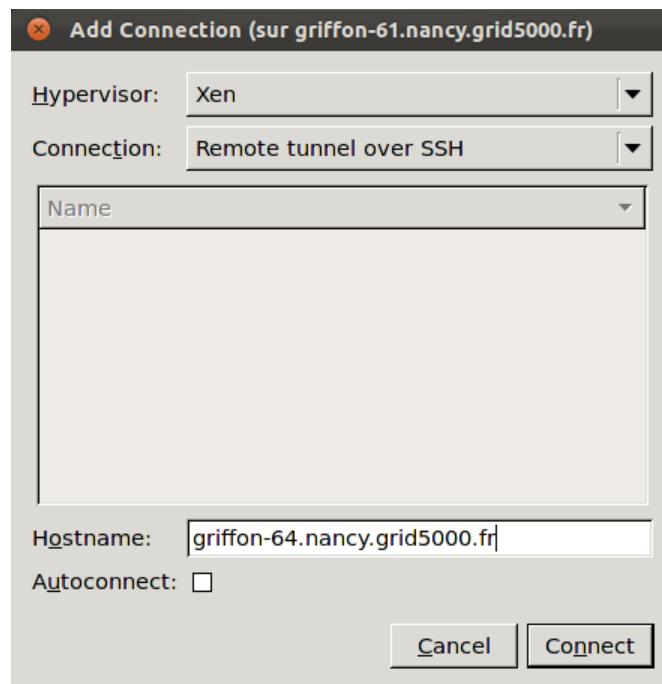
Pour gérer nos serveurs, nous installons virt-manager avec la commande suivante :

```

1 apt-get install virt-manager

```

Au premier lancement, seule la connexion locale est active. Pour ajouter un nouveau noeud, on peut passer par le menu *File>Add Connection* auquel cas il faut modifier le type pour le positionner sur *xen+ssh* et renseigner le nom du noeud.



Virt-manager étant un outil graphique, ses fichiers de configurations sont gérés par gconf. Ainsi, pour automatiser l'ajout des différents noeuds nous avons créé un script qui réécrit le fichier `/.gconf/apps/virt-manager/connections/%gconf.xml`

```

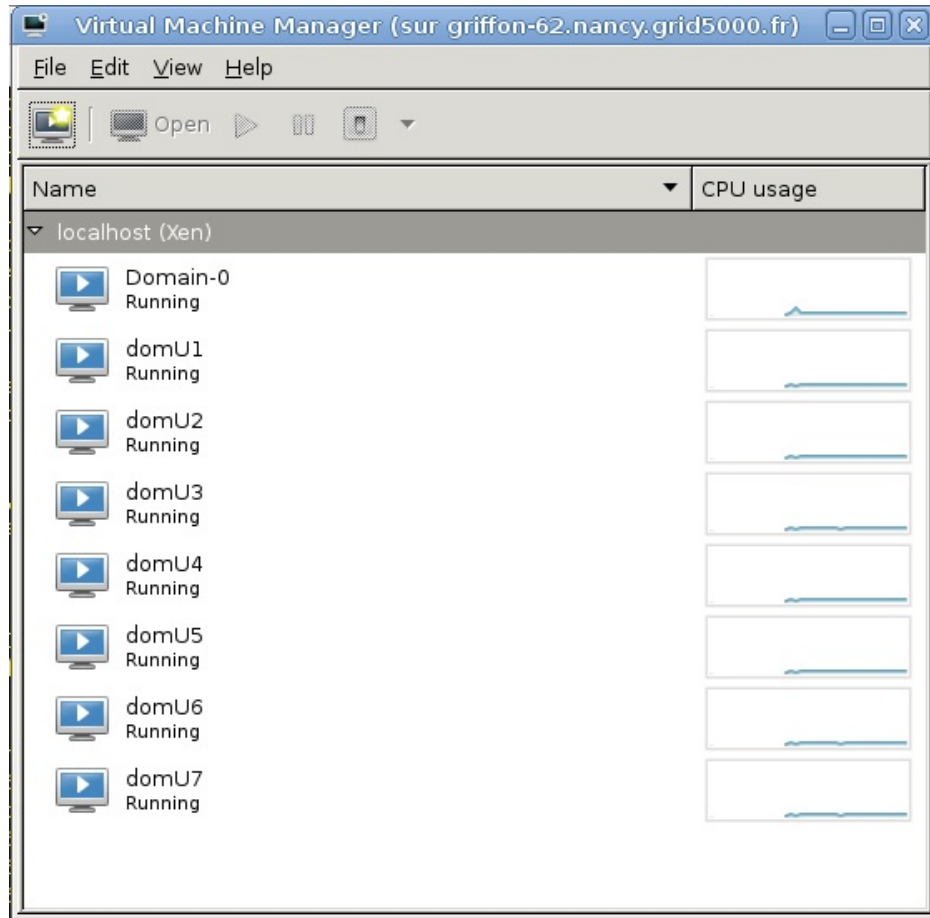
1 for node in $(cat $list_nodes)
2 do
3     echo '<li type="string">' >> $fichier
4     echo "<stringvalue>xen+ssh://root@$node/</stringvalue>" >> $fichier
5     echo '</li>' >> $fichier
6 done

```


Ce fichier est au format xml et contient pour chaque noeud une entrée. Ici, les connexions sont gérées au-dessus d'un tunnel ssh. Comme nous ne pouvions pas accéder, depuis l'extérieur, aux noeuds de Grid5000, nous avons dû installer virt sur les noeuds eux-mêmes (qui sont donc à la fois clients et serveurs) et y accéder via le X Forwarding de la connexion ssh.

5.4 Création d'hôtes virtualisés avec virt-manager

Pour commencer on démarre virt-manager, puis on lance le gestionnaire de machines virtuelles à partir du menu en cliquant sur l'icône en forme de pc.

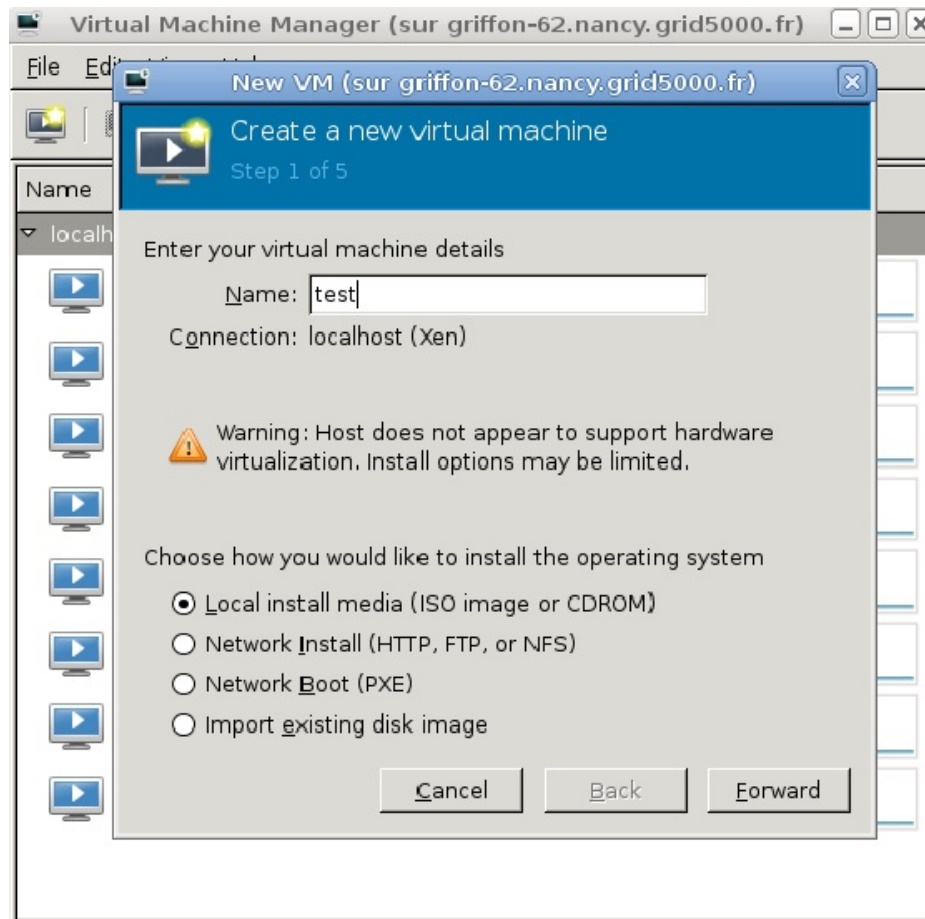


La fenêtre du gestionnaire de machine virtuelle nous autorise à en créer de nouvelles. On clique sur création de nouvelle machine virtuelle pour faire apparaître l'assistant qui va nous aider pour élaborer notre hôte. L'assistant décompose la création en cinq étapes :

- La localisation et la configuration des supports d'installation.
- La configuration de la mémoire et les options de CPU.
- La configuration du stockage de l'invité.
- La configuration réseau, l'architecture, et d'autres paramètres matériels.

Le processus de création d'hôte commence avec la sélection d'un nom et le type d'installation.

Localisation et Configuration des supports d'installation.



Local install media(ISO image or CDROM) : Cette méthode utilise un CD-ROM,DVD ou une image iso.

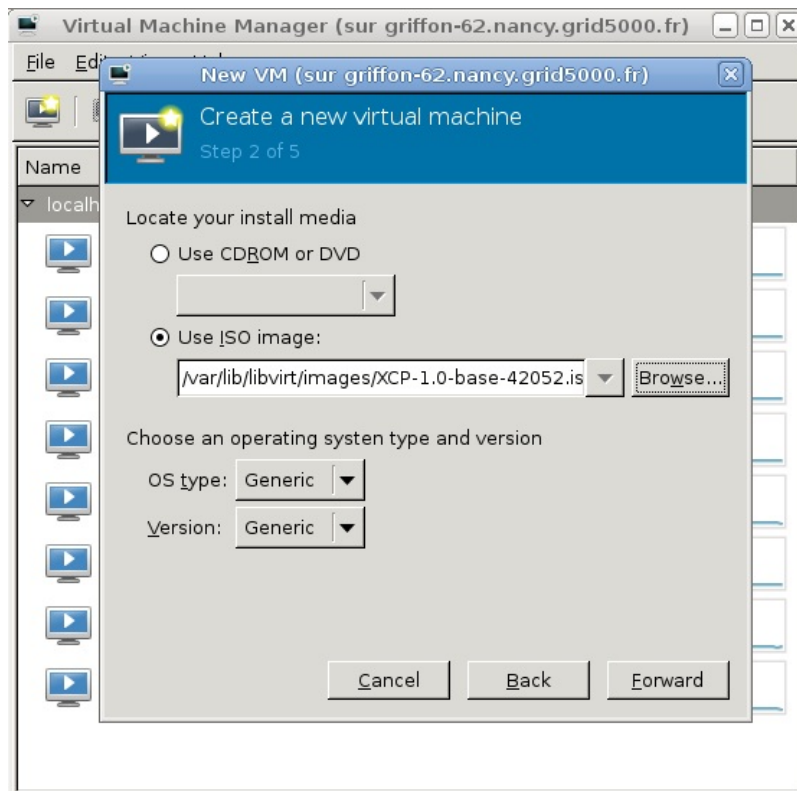
Network install : Cette méthode utilise le réseau pour installer le système d'exploitation.

Network Boot : Ceci permet de récupérer un environnement exécutable directement depuis le réseau et donc sans installation nécessaire.

Import existing disk image : Cette méthode nous permet de créer un nouvelle hôte et d'y importer une image disque.

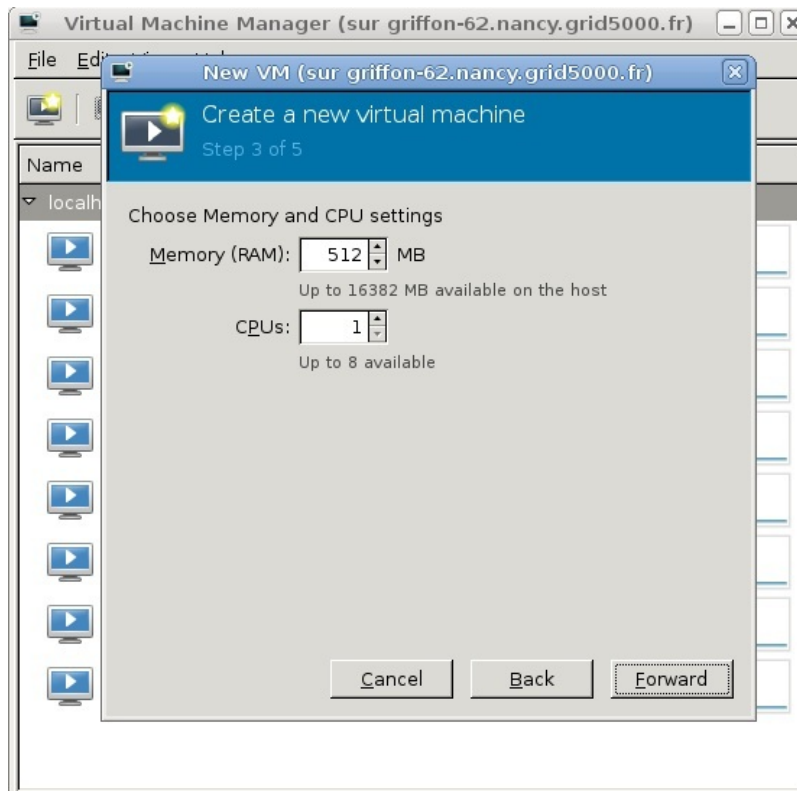
La prochaine étape consiste à configurer l'installation.

On configure le type de système d'exploitation et sa version qui sera installé, cela dépend de la méthode d'installation que l'on a choisie.

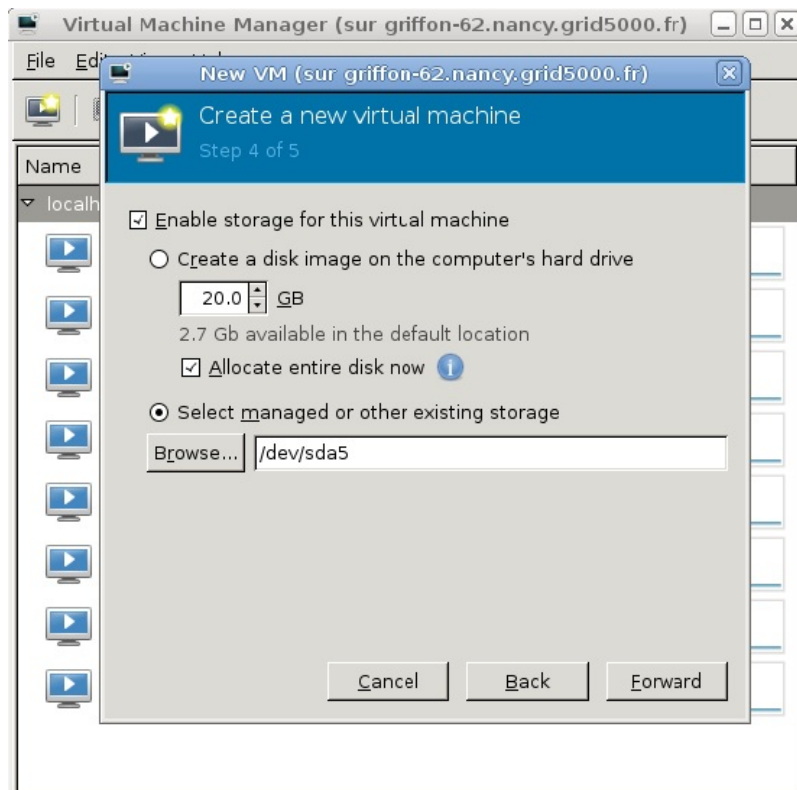


Configuration du CPU et de la mémoire

La prochaine étape consiste à configurer le nombre de CPU et la quantité de mémoire à allouer à la machine virtuelle. L'assistant indique le nombre de processeurs et la quantité de mémoire que l'on peut lui allouer.



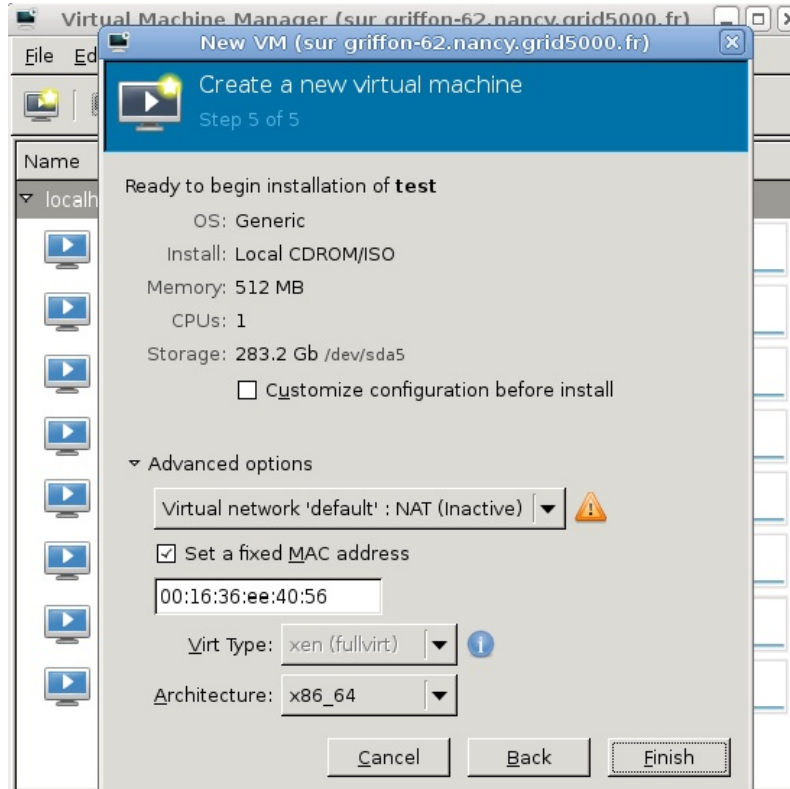
Configuration de l'espace de stockage



Si l'on a choisi d'importer une image de disque existante au cours de la première étape, virt-manager va sauter cette étape. On doit attribuer un espace suffisant pour notre machine virtuelle et toutes les applications que l'hôte a besoin.

Configuration réseau et architecture

On vérifie les paramètres de la machine virtuelle et on clique sur Terminer lorsqu'on est satisfait, cela permettra de créer l'hôte avec les paramètres réseau par défaut, le type de virtualisation, et l'architecture.



5.4.1 Controle d'hotes distants

On peut facilement manager des hotes distants une fois ceux-ci enregistrés dans virt-manager. Cette opération peut être fastidieuse car virt étant un programme graphique, créer un script pour automatiser l'ajout de plusieurs dizaines de noeuds s'est avéré plus compliqué que prévu. Cherchant désespérément un fichier de configuration, nous avons fini par le trouver en lançant une recherche dans tout le système sur les noms des noeuds ajoutés à virt. Le fichier est donc géré par *gconf* en voici un exemple

```

1 <?xml version="1.0"?>
2 <gconf>
3   <entry name="autoconnect" mtime="1332579669" type="list" ltype="string">
4     <li type="string">
5       <stringvalue>xen:///</stringvalue>
6     </li>
7   </entry>
8   <entry name="uris" mtime="1332579669" type="list" ltype="string">
9     <li type="string">
10      <stringvalue>xen:///</stringvalue>
11    </li>
12  </entry>

```

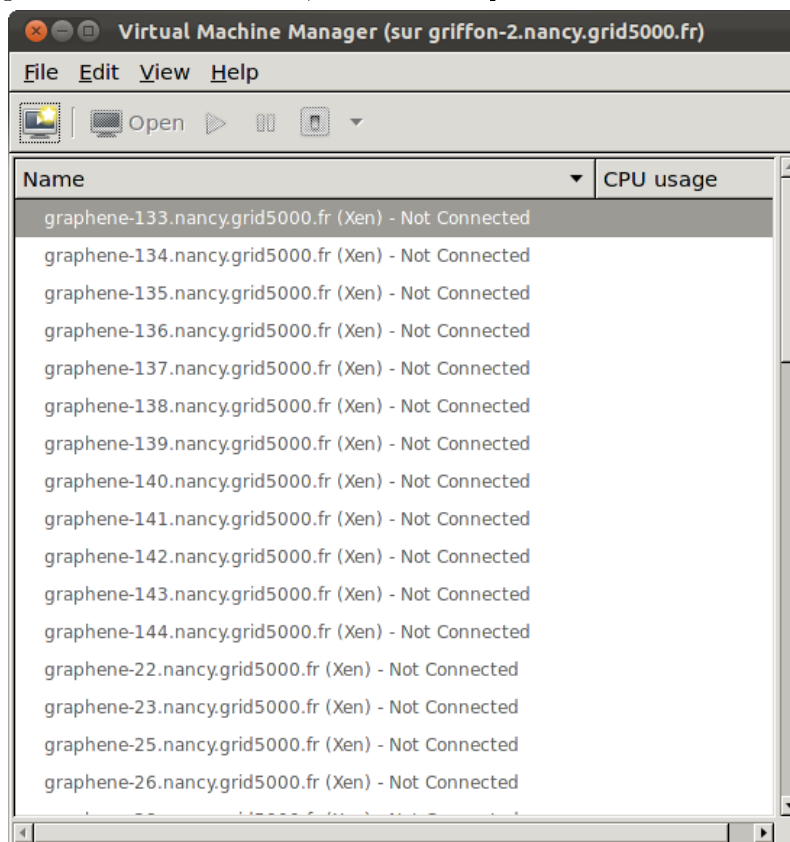
13 </gconf>

Listing 5.1 – Fichier .gconf/apps/virt-manager/connections/%gconf.xml

Après l'installation de virt, le seul lien enregistré est le lien local. Pour ajouter une connexion vers un nouvel hôte xen, on peut utiliser l'interface graphique. Cependant, dans notre cas, il était assez fastidieux de renseigner manuellement chacun des noeuds surtout lors des tests à grande échelle. C'est pourquoi nous avons fait un script (voir annexe B.11 en page 60) donc voici la boucle d'ajout des noeuds :

```
1 #Pour chaque noeud rserv , on ajoute une entre dans le fichier
2 for node in $(cat $list_nodes)
3 do
4     echo '<li type="string">' >> $fichier
5     echo "<stringvalue>xen+ssh://root@$node/</stringvalue>" >> $fichier
6     echo '</li>' >> $fichier
7 done
```

Le script de base ayant configuré chacun des noeuds de la même manière, ils peuvent tous être utilisés pour gérer l'ensemble du réseau, une fois le script exécuté.



Un simple clic sur un noeud établit la connexion puisque les mêmes clefs ssh ont été copiées partout et le fichier *known_hosts* a également été répliqué pour éviter de confirmer l'ajout d'un hôte inconnue (opération qui devait se faire en ligne de commande car non gérée par virt).

On est ensuite en mesure de créer une machine sur n'importe quel noeud, la procédure étant la même que pour une installation locale, nous n'allons pas la détailler à nouveau.

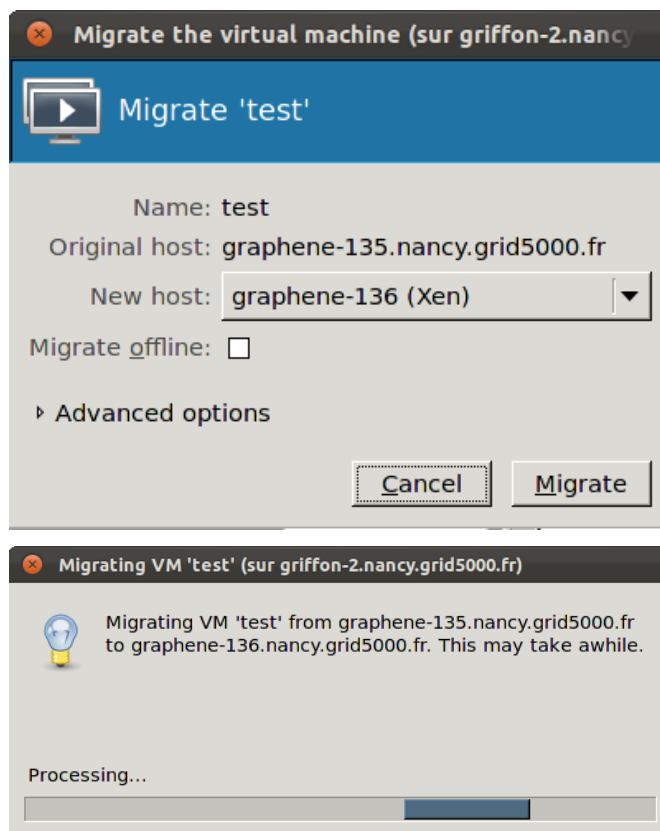
5.4.2 Migration de machines

Une fonctionnalité intéressante d'un gestionnaire de machines virtuelles est la migration des systèmes invités vers un autre hôte pour prévoir des opérations de maintenance, remplacement,...

Encore une fois virt-manger possède un outil graphique afin d'assister cette étape. Pour commencer, en effectuant un click droit sur une machine virtuelle on obtient un menu contextuel que nous allons détailler au fur et à mesure.



En sélectionnant l'option *Migrate* l'assistant demande alors sur quel autre noeud (préalablement ajouté à virt) nous souhaitons transférer la machine virtuelle. L'opération peut prendre plusieurs minutes en fonctions des capacités des machines et du réseau.



Chapitre 6

Comparatif

Voici un petit tableau de comparaisons des différentes solutions de gestion centralisée, en fonction de ce que nous avons pu tester.

	OpenXenManager	Ganeti	Virt-Manager	Archipel
Documentation	Quasi inexistante	Bonne : Toutes les informations se trouvent assez facilement sur le site de ganeti.	Quelques sites proposent des tutoriaux. Nous pouvons qualifier la documentation de bonne bien que les configuration poussées nécessitent des recherches personnelles	Detaillée : Les tutoriels sont assez complets et la documentation sur le logiciel reste relativement jeune et donc complete sur le site du projet archipel
Communauté	Très peu de sites traitent du sujet. Nous avons trouvé très peu de réponse à nos problèmes	Presente : La communaute reste encore active meme si le projet est abouti.	On peut trouver facilement des sites traitant du sujet mais encore une fois, seules les questions de bases sont évoquées. Nos besoins nous poussant souvent à rechercher nous-mêmes une solution.	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.
Maturité	Projet toujours en cours de développement	Aboutie : produit complet beaucoup de fonctionnalisees disponibles.	Projet mature mais disposant de trop peu de fonctionnalités pour permettre une pleine exploitation des possibilités de Grid5000	Jeune : Le projet est actuellement en beta 5.
Installation	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.	Problematic : installation facile mais la configuration peut etre complexe celons l'architecture du reseau et les besoins des utilisateurs	Installation aisée via le logiciel de gestion de paquets apt-get	Longue : L'installation reste complexe car beaucoup de choses restent a configurer pour que le logiciel fonctionne sur la plateforme.

	OpenXenManager	Ganeti	Virt-Manager	Archipel
Réseau	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.	Passable : Déploiement plutôt facile tant sur un réseau complexe que sur un réseau simple, la gestion des ip des machines virtuelles reste compliquée	Configuration automatique, virt s'occupe de configurer le bridge	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.
Sécurité	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.	Bonne : Le dialogue entre tous les nœuds est effectuée en ssh avec transmission de la cle du maitre sur les esclaves.	Communications entres les noeuds au-dessus d'un tunnel SSH	Tres bonne : Le logiciel ce base sur le protocole XMPP (SASL et TLS) pour discuter avec les nœuds.
Simplicité	Bonne : possède une interface graphique. Mais nous n'avons pas pu mettre en oeuvre ce logiciel donc nous n'avons pas d'avis très critique.	Mitige : Pour une architecture classique cela reste assez facile a mettre en œuvre.	Bonne : Interface graphique assez intuitive	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.
Flexibilité	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.	Tres bonne : Tant que l'on reste dans une architecture classique, sa grande modularite est un atout qui permet de facilement le deployer sur une architecture multi-sites.	Assez faible, on ne peut pas rajouter de fonctionnalités	Non-teste : Nous n'avons pas pu mettre en œuvre ce logiciel.

Chapitre 7

Conclusion

L'objectif de ce projet était de déployer et d'évaluer les différentes solutions libres permettant de gérer la gestion de machines virtuelles sur la plateforme Grid 5000.

Cet objectif est partiellement atteint. En effet, nous avons pu tester que deux solutions de gestion : Ganeti et virt-manager.

Ce projet étant ambitieux, nous nous sommes vite heurtés à de nombreux problèmes, que ce soit dû aux solutions de gestion ou à leur intégration sur la plateforme Grid 5000, notamment en ce qui concerne le réseau pour ganeti ou les interfaces graphiques pour les autres solutions.

Tous ces problèmes nous ont montré la complexité d'utiliser une telle plate-forme, et leurs résolutions nous a souvent retardé mais nous ont appris à chercher des solutions afin de contourner les différents problèmes.

Ce projet nous a également apporté une expérience de travail en équipe sur un projet de longue durée, en le découpant en tâches et en répartissant le travail.

Ce projet a été pour nous une chance de découvrir un environnement aussi complexe que Grid 5000 qui est très intéressant, ce qui nous a permis d'acquérir de l'expérience en administration systèmes et réseaux et d'approfondir nos connaissances dans le domaine de la virtualisation et de la gestion de machines virtuelles.

Annexe A

Sources

<http://www.grid5000.fr> : le wiki disponible sur le site internet de grid5000 fut principale source de renseignements pour le démarrage du projet.

<http://www.loria.fr/~lnussbau/> : nous avons pu y consulter des anciens projets sur Grid5000 ce qui nous a permis d'avoir un premier aperçu de ses possibilités. de machines virtuelles.

<http://fr.wikipedia.org/> : Source d'informations générales

<http://code.google.com/p/ganeti/> : site du projet ganeti

<http://docs.ganeti.org/ganeti/> : site regroupant les tutos pour les différentes versions de ganeti

<http://docs.ganeti.org/ganeti/master/html/install.html> : tuto d'installation de la version de ganeti utilisée dans le cadre du projet

<http://archipelproject.org/> : dépôt Git du projet.

Annexe B

Scripts

B.1 Scripts spécifiques à l'environnement de Grid5000

```
1  #!/usr/bin/ruby -w
2  # -*- coding: utf-8 -*-
3  puts "-----"
4  puts "Souhaitez vous reserver des noeuds?(y/n)"
5  loop do
6    test = gets.chomp
7    if test.eql?("n")
8      puts "#####"
9      puts "#sortie du programme#"
10     puts "#####"
11     break;
12   end
13   if test.eql?("y")
14     #Reservation de machines
15     puts "-----Script de reservation-----"
16     puts "Choisir un nombre de noeud:"
17     noeuds = gets.chomp.to_i
18     puts "Choisir un temps de reservation(HH:MM:SS):"
19     temps = gets
20     puts "\nVous avez reserve #{noeuds} noeuds pour une duree de #{temps}"
21     puts "-----"
22     puts "Voulez-vous mettre les noeuds dans un vlan separe(y/n)"
23   loop do
24     test = gets.chomp
25     if test.eql?("n")
26       exec "oarsub -I -t deploy -n'virtu' -l slash_22=1+nodes=#{noeuds},walltime=#{temps}"
27       break;
28     end
29     if test.eql?("y")
30       puts "-----"
31       exec "oarsub -I -t deploy -n'virtu' -l {\\"type='kavlan-local'\\\"}/vlan=1+/slash_22=1+nodes=#{noeuds},walltime=#{temps}"
32       break;
33     end
34   end
35   break;
36 end
37 end
```

Listing B.1 – Réservation de noeuds

```

1  # -*- coding: utf-8 -*-
2  #!/usr/bin/ruby -w
3
4  #Deployer une image cree
5  puts "Voulez vous deployer une image?(y/n)"
6  loop do
7      test = gets.chomp
8      if test.eql?("n")
9          break;
10     end
11     if test.eql?("y")
12         #choix de l'image
13         puts "-----"
14         puts "image disponibles:"
15         puts `ls | grep .env`
16         puts "-----"
17         puts "Choix de la distribution(tout saisir):"
18         debian = gets.chomp
19         puts "-----"
20         puts "Voulez-vous deploye dans un vlan separe(y/n)"
21         loop do
22             test = gets.chomp
23             if test.eql?("n")
24                 puts "-----"
25                 exec"kadeploy3 -f $OAR_FILE_NODES -a #{debian} -k $HOME/.ssh/id_rsa.pub"
26                 break;
27             end
28             if test.eql?("y")
29                 puts "-----"
30                 vlan = `kavlan -V`
31                 exec"kadeploy3 -f $OAR_FILE_NODES -a #{debian} --vlan #{vlan} -d -k $HOME/.ssh/
32                     id_rsa.pub"
33                 break;
34             end
35             break;
36         end
37     end
38
39     #####
40
41     #Deploiment de l'environnement sur les noeuds reserves
42     puts "Voulez vous deployer un environnement?(y/n)"
43     loop do
44         test = gets.chomp
45         if test.eql?("n")
46             puts "#####"
47             puts "#sortie du programme#"
48             puts "#####"
49             break;
50         end
51         if test.eql?("y")
52             #choix de la version a deployer
53             puts "-----"
54             puts "distributions disponibles:"
55             puts `kaenv3 -l | cut -d - -f1 | uniq | tail -n +3`
56             puts "-----"
57             puts "Choix de la distribution:"

```

```

58  debian = gets.chomp
59  puts "-----"
60  puts "version de la distribution:"
61  puts `kaenv3 -l | grep #{debian}| cut -d ' ' -f1`
62  debian = debian+"-x64-"
63  puts "-----"
64  puts "Choix de la version de la distribution a deployee (sans #{debian}):"
65  version = gets.chomp
66  version = debian+version
67  puts "version utilisee:"
68  puts version
69  puts "-----"
70  puts "Voulez-vous deploye dans un vlan separe(y/n)"
71  loop do
72    test = gets.chomp
73    if test.eql?("n")
74      #choix de la version  deployer
75      puts "-----"
76      exec"kadeploy3 -e #{version} -f $OAR_FILE_NODES -k $HOME/.ssh/id_rsa.pub"
77      break;
78    end
79    if test.eql?("y")
80      puts "-----"
81      vlan = `kavlan -V`
82      exec"kadeploy3 -e #{version} -f $OAR_FILE_NODES --vlan #{vlan} -k $HOME/.ssh/
      id_rsa.pub"
83      break;
84    end
85  end
86  break;
87 end
88 end

```

Listing B.2 – Déploiement des environnements sur les noeuds réservés

B.2 Scripts pour le déploiement de ganeti

B.2.1 Configuration générale des noeuds

```
1 #!/bin/bash
2
3 #####
4 #-----Configuration minimale-----#
5 #-----#
6
7 config_ssh () {
8     #ajoute les lignes permettant la connexion a un kvlan
9     Echo "Host *--kavlan-1 *--kavlan-1.grid5000.fr"
10    echo "ProxyCommand ssh -a -x kavlan-1 nc -q 0 %h %p"
11    echo "Host *--kavlan-2 *--kavlan-2.grid5000.fr"
12    echo "ProxyCommand ssh -a -x kavlan-2 nc -q 0 %h %p"
13    echo "Host *--kavlan-3 *--kavlan-3.grid5000.fr"
14    echo "ProxyCommand ssh -a -x kavlan-3 nc -q 0 %h %p"
15 }
16
17 #Verification et configuration SSH de l'utilisateur
18 #-----#
19 ##On regarde si le fichier .ssh existe
20 echo "---"
21 echo "Verification de la configuration ssh de l'utilisateur "$USER
22
23 trouver=0
24
25 if [ -f $HOME/.ssh/config ]
26 then
27     trouver=1
28     for line in $(cat $HOME/.ssh/config)
29     do
30         if [ $line = "kavlan-$vlan" ]
31         then
32             trouver=2
33         fi
34     done
35 else
36     echo "Erreur. Aucun fichier, .ssh/config, n'a pas t trouv..."
37     echo "Arret de l'installation."
38     exit
39 fi
40 if [ $trouver -eq 1 ]
41 then
42     echo "Ajout des lignes de 'transparence'"
43     config_ssh >> $HOME/.ssh/config
44 fi
45
46 echo "-----"
47 echo "Vos noeuds sont dans un kavlan?(y/n)"
48 read choix
49 if [ $choix == "n" ]
50 then
51     #suppression des know_host
52     rm $HOME/.ssh/known_hosts
53     #Recuperation des noeuds reserves
54     cat $OAR_FILE_NODES | uniq > $HOME/script_base/list_nodes
```



```

55 list_nodes="$HOME/script_base/list_nodes"
56
57 echo "-----"
58 echo "Liste des machines reservee:"
59 cat $list_nodes
60 echo "-----"
61 echo "Copie des clees SSH vers toutes les machines."
62 cat $HOME/.ssh/id_dsa.pub >> $HOME/.ssh/authorized_keys
63 for node in $(cat $list_nodes)
64 do
65     scp $HOME/.ssh/id_* root@$node:~/.ssh/
66     taktuk -l root -s -m $node broadcast exec [ 'cat ~/.ssh/id_dsa.pub >> ~/.ssh/
        authorized_keys' ]
67 done
68 echo "-----"
69
70 else
71     #recuperation des noeuds reserves
72     kavlan -l > list_nodes_kavlan
73     list_nodes_kavlan="$HOME/script_base/list_nodes_kavlan"
74
75     echo "-----"
76     echo "Liste des machines reservee:"
77     cat $list_nodes_kavlan
78     echo "-----"
79     echo "Copie des clees SSH vers toutes les machines."
80     for node in $(cat $list_nodes_kavlan)
81     do
82         scp $HOME/.ssh/id_rsa* root@$node:~/.ssh/
83     done
84     echo "-----"
85
86     #changement des mdp root
87     taktuk -l root -f $list_nodes_kavlan broadcast exec [ 'echo -e "pttvirtu\npttvirtu" |
        passwd root' ]
88 fi

```

Listing B.3 – Configuration générale des noeuds

B.2.2 Création de machines virtuelles pour Ganetti

```

1 #
2
3 # Copyright (C) 2007, 2008, 2009 Google Inc.
4 #
5 # This program is free software; you can redistribute it and/or modify
6 # it under the terms of the GNU General Public License as published by
7 # the Free Software Foundation; either version 2 of the License, or
8 # (at your option) any later version.
9 #
10 # This program is distributed in the hope that it will be useful, but
11 # WITHOUT ANY WARRANTY; without even the implied warranty of
12 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 # General Public License for more details.
14 #
15 # You should have received a copy of the GNU General Public License
16 # along with this program; if not, write to the Free Software
17 # Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA

```

```

18 # 02110-1301, USA.
19
20 CLEANUP=( )
21
22 log_error() {
23     echo "$@" >&2
24 }
25
26
27 get_api5_arguments() {
28     GETOPT_RESULT=$*
29     # Note the quotes around '$TEMP': they are essential!
30     eval set -- "$GETOPT_RESULT"
31     while true; do
32         case "$1" in
33             -i|-n) instance=$2; shift 2;;
34
35             -o) old_name=$2; shift 2;;
36
37             -b) blockdev=$2; shift 2;;
38
39             -s) swapdev=$2; shift 2;;
40
41             --) shift; break;;
42
43             *) log_error "Internal error!" >&2; exit 1;;
44         esac
45     done
46     if [ -z "$instance" -o -z "$blockdev" ]; then
47         log_error "Missing OS API Argument (-i, -n, or -b)"
48         exit 1
49     fi
50     if [ "$SCRIPT_NAME" != "export" -a -z "$swapdev" ]; then
51         log_error "Missing OS API Argument -s (swapdev)"
52         exit 1
53     fi
54     if [ "$SCRIPT_NAME" = "rename" -a -z "$old_name" ]; then
55         log_error "Missing OS API Argument -o (old_name)"
56         exit 1
57     fi
58 }
59
60 get_api10_arguments() {
61     if [ -z "$INSTANCE_NAME" -o -z "$HYPERVISOR" -o -z "$DISK_COUNT" ]; then
62         log_error "Missing OS API Variable:"
63         log_error "(INSTANCE_NAME HYPERVISOR or DISK_COUNT)"
64         exit 1
65     fi
66     instance=$INSTANCE_NAME
67     if [ $DISK_COUNT -lt 1 -o -z "$DISK_O_PATH" ]; then
68         log_error "At least one disk is needed"
69         exit 1
70     fi
71     if [ "$SCRIPT_NAME" = "export" ]; then
72         if [ -z "$EXPORT_DEVICE" ]; then
73             log_error "Missing OS API Variable EXPORT_DEVICE"
74         fi
75         blockdev=$EXPORT_DEVICE
76     elif [ "$SCRIPT_NAME" = "import" ]; then

```

```

77     if [ -z "$IMPORT_DEVICE" ]; then
78         log_error "Missing OS API Variable IMPORT_DEVICE"
79     fi
80     blockdev=$IMPORT_DEVICE
81 else
82     blockdev=$DISK_O_PATH
83 fi
84 if [ "$SCRIPT_NAME" = "rename" -a -z "$OLD_INSTANCE_NAME" ]; then
85     log_error "Missing OS API Variable OLD_INSTANCE_NAME"
86 fi
87 old_name=$OLD_INSTANCE_NAME
88 }
89
90
91 format_disk0() {
92     # Create one big partition, and make it bootable
93     # some versions of sfdisk need manual specification of
94     # head/sectors for devices such as drbd which don't
95     # report geometry
96     sfdisk -H 255 -S 63 --quiet --Linux "$1" <<EOF
97 O,,L,*
98 EOF
99 }
100
101 map_disk0() {
102     blockdev="$1"
103     filesystem_dev_base='kpartx -l -p- $blockdev | \
104                         grep -m 1 -- "-1.*$blockdev" | \
105                         awk '{print $1}';
106     if [ -z "$filesystem_dev_base" ]; then
107         log_error "Cannot interpret kpartx output and get partition mapping"
108         exit 1
109     fi
110     kpartx -a -p- $blockdev > /dev/null
111     filesystem_dev="/dev/mapper/$filesystem_dev_base"
112     if [ ! -b "$filesystem_dev" ]; then
113         log_error "Can't find kpartx mapped partition: $filesystem_dev"
114         exit 1
115     fi
116     echo "$filesystem_dev"
117 }
118
119 unmap_disk0() {
120     kpartx -d -p- $1
121 }
122
123 cleanup() {
124     if [ ${#CLEANUP[*]} -gt 0 ]; then
125         LAST_ELEMENT=$(( ${#CLEANUP[*]} - 1 ) )
126         REVERSE_INDEXES=(seq $LAST_ELEMENT -1 0)
127         for i in $REVERSE_INDEXES; do
128             ${CLEANUP[$i]}
129         done
130     fi
131 }
132
133 trap cleanup EXIT
134
135 DEFAULT_FILE="/etc/default/ganeti-instance-debootstrap"

```

```

136 if [ -f "$DEFAULT_FILE" ]; then
137     . "$DEFAULT_FILE"
138 fi
139
140 # note: we don't set a default mirror since debian and ubuntu have
141 # different defaults, and it's better to use the default
142
143 # only if the user want to specify a mirror in the defaults file we
144 # will use it, this declaration is to make sure the variable is set
145 : ${MIRROR:="http://ftp.fr.debian.org/debian/"}
146 : ${PROXY:="http://proxy:3128/"}
147 : ${SUITE:="squeeze"}
148 : ${ARCH:="amd64"}
149 : ${EXTRA_PKGS:=""}
150 : ${CUSTOMIZE_DIR:="/etc/ganeti/instance-debootstrap/hooks"}
151 : ${VARIANTS_DIR:="/etc/ganeti/instance-debootstrap/variants"}
152 : ${GENERATE_CACHE:="yes"}
153 : ${CLEAN_CACHE:="14"} # number of days to keep a cache file
154 if [ -z "$OS_API_VERSION" -o "$OS_API_VERSION" = "5" ]; then
155     DEFAULT_PARTITION_STYLE="none"
156 else
157     DEFAULT_PARTITION_STYLE="msdos"
158 fi
159 : ${PARTITION_STYLE:=$DEFAULT_PARTITION_STYLE} # disk partition style
160
161 CACHE_DIR="/var/cache/ganeti-instance-debootstrap"
162
163 SCRIPT_NAME=$(basename $0)
164
165 if [ -f /sbin/blkid -a -x /sbin/blkid ]; then
166     VOL_ID="/sbin/blkid -o value -s UUID"
167     VOL_TYPE="/sbin/blkid -o value -s TYPE"
168 else
169     for dir in /lib/udev /sbin; do
170         if [ -f $dir/vol_id -a -x $dir/vol_id ]; then
171             VOL_ID="$dir/vol_id -u"
172             VOL_TYPE="$dir/vol_id -t"
173         fi
174     done
175 fi
176
177 if [ -z "$VOL_ID" ]; then
178     log_error "vol_id or blkid not found, please install udev or util-linux"
179     exit 1
180 fi
181
182 if [ -z "$OS_API_VERSION" -o "$OS_API_VERSION" = "5" ]; then
183     OS_API_VERSION=5
184     GETOPT_RESULT=`getopt -o o:n:i:b:s: -n '$0' -- "$@"`
185     if [ $? != 0 ] ; then log_error "Terminating..."; exit 1 ; fi
186     get_api5_arguments $GETOPT_RESULT
187 elif [ "$OS_API_VERSION" = "10" -o "$OS_API_VERSION" = "15" ]; then
188     get_api10_arguments
189 else
190     log_error "Unknown OS API VERSION $OS_API_VERSION"
191     exit 1
192 fi
193
194 if [ -n "$OS_VARIANT" ]; then

```

```

195 if [ ! -d "$VARIANTS_DIR" ]; then
196     log_error "OS Variants directory $VARIANTS_DIR doesn't exist"
197     exit 1
198 fi
199 VARIANT_CONFIG="$VARIANTS_DIR/$OS_VARIANT.conf"
200 if [ -f "$VARIANT_CONFIG" ]; then
201     . "$VARIANT_CONFIG"
202 else
203     if grep -qxF "$OS_VARIANT" variants.list; then
204         log_error "ERROR: instance-debootstrap configuration error"
205         log_error " Published variant $OS_VARIANT is missing its config file"
206         log_error " Please create $VARIANT_CONFIG or unpublish the variant"
207         log_error " (by removing $OS_VARIANT from variants.list)"
208     else
209         log_error "Unofficial variant $OS_VARIANT is unsupported"
210         log_error "Most probably this is a user error, forcing a wrong name"
211         log_error "To support this variant please create file $VARIANT_CONFIG"
212     fi
213     exit 1
214 fi
215 fi

```

B.2.3 Installation et configuration de ganeti

```

1  #!/bin/bash
2
3  #####
4  #-----Installation de Ganeti-----#
5  #-----#
6
7  #chargement des noeuds reserve
8  list_nodes="$HOME/script_base/list_nodes"
9  list_node='cat $HOME/script_base/list_nodes'
10
11 #sauvegarde le 1er noeud qui sera le maitre
12 sed -n "1 p" $list_nodes > $HOME/ganeti/ganeti_manager
13 ganeti_manager="$HOME/ganeti/ganeti_manager"
14 ganeti_man='cat $HOME/ganeti/ganeti_manager'
15
16 #sauvegarde des noeud esclaves
17 cat $list_nodes | grep -v $ganeti_man > $HOME/ganeti/ganeti_slaves
18 ganeti_slaves="$HOME/ganeti/ganeti_slaves"
19 ganeti_slav='cat $HOME/ganeti/ganeti_slaves'
20
21
22 echo "#####Ganeti Manager#####"
23 cat $ganeti_manager
24 echo "#####"
25 echo "#####Ganeti Slaves#####"
26 cat $ganeti_slaves
27 echo "#####"
28
29 # rcupre l'adresse du cluster1 pour l'envoyer au node x.x.x.1
30 g5k-subnets -i -o ip_list.txt
31 ipnode='head -1 ip_list.txt'
32 echo $ipnode > ipcluster
33

```

```

34 #recupere l'ip de la gateway x.x.x.254
35 ipgateway='head -254 ip_list.txt | tail -1'
36 echo $ipgateway > ipgateway
37
38 # rcupre l'ip du reseau
39 g5k-subnets -a > tempipreseau
40 ipnetwork='head -1 tempipreseau | cut -d'/' -f1'
41 echo $ipnetwork > ipnetwork
42
43 #envoi via ssh au node
44 while read node
45 do
46     scp common.sh node-add.sh ganeti_slaves ganeti_master.sh ganeti_slave.sh ipgateway
47     ipnetwork ipcluster tempipreseau ip_list.txt root@$node:/root/
48 done < $list_nodes
49
50 #suppression des fichiers
51 rm ipgateway ipnetwork ipcluster tempipreseau ip_list.txt
52
53 #lancement du script d'installation de ganeti sur le manager
54 taktuk -l root -f $ganeti_manager broadcast exec [ sh ganeti_master.sh ]
55
56 #idem sur les esclaves
57 taktuk -l root -f $ganeti_slaves broadcast exec [ sh ganeti_slave.sh ]
58
59 mkdir keys
60 cd keys
61 scp root@$ganeti_man:/root/.ssh/authorized_keys .
62 mv authorized_keys authorized
63 tail -n 1 authorized > authorized_keys && rm authorized
64 scp root@$ganeti_man:/root/.ssh/id_dsa* .
65 taktuk -l root -f $ganeti_manager broadcast exec [ rm /usr/lib/ganeti/tools/setup-ssh ]
66 scp setup-ssh root@$ganeti_man:/usr/lib/ganeti/tools/
67
68 while read slavenode
69 do
70     scp id_dsa* authorized_keys root@$slavenode:/root/
71 done < $ganeti_slaves
72 rm id_dsa*
73 taktuk -l root -f $ganeti_slaves broadcast exec [ cp /root/authorized_keys /root/.ssh/ ]

```

Listing B.4 – Installation et configuration de ganeti

B.2.4 Installation et configuration du master node

```

1 #!/bin/bash
2
3 ##VERIFIER QUE postinstall.sh et ganeti.sh soient dans le meme dossier puis executer
4 postinstall.sh (Sur le frontend)!
5
6 #Ajout des sources testing pour installer la derniere version de ganeti
7 echo "## wheezy security" >> /etc/apt/sources.list
8 echo "deb http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/apt/
9 sources.list
10 echo "deb-src http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/
11 apt/sources.list
12 echo " " >> /etc/apt/sources.list
13 echo "#wheezy" >> /etc/apt/sources.list

```

```

11 echo "deb http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt/
    sources.list
12 echo "deb-src http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt
    /sources.list
13
14 #Ajout du fichier de preference
15 echo "APT::Default-Release \"stable\";" > /etc/apt/apt.conf.d/80default-distrib
16
17 #Installation de ganeti
18 apt-get update && apt-get dist-upgrade -q -y --force-yes
19 apt-get -t testing install -q -y --force-yes ganeti2 ganeti-htools ganeti-instance-
    debootstrap
20
21 echo "Ajout du node dans /etc/hosts"
22 hostname='cat /etc/hostname'
23
24 #recuperation des variables
25
26 #ip du node
27 ifconfig eth0 > troll
28 ipnode='head -2 troll | tail -1 | cut -d':' -f2 | cut -d' ' -f1'
29 echo $ipnode " " $hostname >> /etc/hosts
30
31 #ip du broadcast
32 ipbroadcast='head -2 troll | tail -1 | cut -d'B' -f2 | cut -d':' -f2 | cut -d' ' -f1'
33
34 #ip du masque de sous rseau
35 ipmask='head -2 troll | tail -1 | cut -d'M' -f2 | cut -d':' -f2 | cut -d' ' -f1'
36
37 #ip du reseau
38 ipnetwork='head -1 ipnetwork'
39
40 #ip de la passerelle
41 ipgateway='head -1 ipgateway'
42
43 #ajout de cluster1 dans dans /etc/hosts
44 echo "ajout de cluster1 dans /etc/hosts"
45 ipcluster='cat ipcluster'
46 echo $ipcluster "cluster1" >> /etc/hosts
47
48 #Dans /boot/ creer des liens symboliques :
49 cd /boot
50 cp vmlinuz-2.6.32-5-xen-amd64 vmlinuz-2.6-xenU
51 cp initrd.img-2.6.32-5-xen-amd64 initrd.img-2.6-xenU
52
53 #Pour le moment changera surement.
54 echo "creation du LVM"
55 umount /dev/sda5
56 pvcreate /dev/sda5
57 vgcreate xenvg /dev/sda5
58
59 #Creation du bridge xen-br0
60 echo " " >> /etc/network/interfaces
61 echo "auto xen-br0" >> /etc/network/interfaces
62 echo "iface xen-br0 inet static" >> /etc/network/interfaces
63 echo "address" $ipnode >> /etc/network/interfaces
64 echo "netmask " $ipmask >> /etc/network/interfaces
65 echo "network" $ipnetwork >> /etc/network/interfaces
66 echo "gateway" $ipgateway >> /etc/network/interfaces

```

```

67 echo "broadcast" $ipbroadcast>> /etc/network/interfaces
68 echo "bridge_ports eth0" >> /etc/network/interfaces
69 echo "bridge_stp off" >> /etc/network/interfaces
70 echo "bridge_fd 0" >> /etc/network/interfaces
71
72 #Suppression des ligne de eth0
73 sed -i '9d' /etc/network/interfaces
74 sed -i '9d' /etc/network/interfaces
75
76 #Redemarage du network
77 /etc/init.d/networking stop
78 /etc/init.d/networking start
79
80 #supression des fichier temporaires et copie de common.sh
81 cd /root/
82 rm troll ipcluster ipgateway ipnetwork
83 rm /usr/share/ganeti/os/debootstrap/common.sh
84 mv common.sh /usr/share/ganeti/os/debootstrap/
85
86 #initialisation du cluster
87 gnt-cluster init --master-netdev xen-br0 --no-drbd-storage --nic-parameters link=eth0
    cluster1
88
89 #et verifier
90 gnt-node list
91
92 #reiseigner le inird pour les instances
93 gnt-cluster modify --hypervisor-parameter xen-pvm:initrd_path='/boot/initrd.img-2.6-xenU
    ,
94 gnt-cluster modify --hypervisor-parameter xen-pvm:root_path='/dev/xvda1'

```

Listing B.5 – Installation et configuration du master node

B.2.5 Installation et configuration des noeuds esclaves

```

1  #!/bin/bash
2
3  ##VERIFIER QUE postinstall.sh et ganeti.sh soient dans le meme dossier puis executer
    postinstall.sh (Sur le frontend)!
4  #Ajout des sources testing pour installer la derniere version de ganeti
5  echo "## wheezy security" >> /etc/apt/sources.list
6  echo "deb http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/apt/
    sources.list
7  echo "deb-src http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/
    apt/sources.list
8  echo " " >> /etc/apt/sources.list
9  echo "#wheezy" >> /etc/apt/sources.list
10 echo "deb http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt/
    sources.list
11 echo "deb-src http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt
    /sources.list
12
13 #Ajout du fichier de preference
14 echo "APT::Default-Release \"stable\";" > /etc/apt/apt.conf.d/80default-distrib
15
16 #Installation de ganeti
17 apt-get update && apt-get dist-upgrade -y --force-yes

```



```

18 apt-get -t testing install -y --force-yes ganeti2 ganeti-htools ganeti-instance-
    debootstrap
19
20 echo "Ajout du node dans /etc/hosts"
21 hostname='cat /etc/hostname'
22
23 #recuperation des variables
24
25 #ip du node
26 ifconfig eth0 > troll
27 ipnode='head -2 troll | tail -1 | cut -d':' -f2 | cut -d' ' -f1'
28 echo $ipnode " " $hostname >> /etc/hosts
29
30 #ip du broadcast
31 ipbroadcast='head -2 troll | tail -1 | cut -d'B' -f2 | cut -d':' -f2 | cut -d' ' -f1'
32
33 #ip du masque de sous reseau
34 ipmask='head -2 troll | tail -1 | cut -d'M' -f2 | cut -d':' -f2 | cut -d' ' -f1'
35
36 #ip du reseau
37 ipnetwork=172.16.64.0
38
39 #ip de la passerelle
40 ipgateway='head -1 ipgateway'
41 ipgateway=10.0.1.254
42
43 #ajout de cluster1 dans /etc/hosts
44 echo "ajout de cluster1 dans /etc/hosts"
45 ipcluster='cat ipcluster'
46 echo $ipcluster "cluster1" >> /etc/hosts
47
48 #Dans /boot/ creer des liens symboliques :
49 cd /boot
50 cp vmlinuz-2.6.32-5-xen-amd64 vmlinuz-2.6-xenU
51 cp initrd.img-2.6.32-5-xen-amd64 initrd.img-2.6-xenU
52
53 #Pour le moment changera surement.
54 echo "creation du LVM"
55 umount /dev/sda5
56 pvcreate /dev/sda5
57 vgcreate xenvg /dev/sda5
58
59 #Creation du bridge xen-br0
60 echo " " >> /etc/network/interfaces
61 echo "auto xen-br0" >> /etc/network/interfaces
62 echo "iface xen-br0 inet static" >> /etc/network/interfaces
63 echo "address" $ipnode >> /etc/network/interfaces
64 echo "netmask " $ipmask >> /etc/network/interfaces
65 echo "network" $ipnetwork >> /etc/network/interfaces
66 echo "gateway" $ipgateway >> /etc/network/interfaces
67 echo "broadcast" $ipbroadcast >> /etc/network/interfaces
68 echo "bridge_ports eth0" >> /etc/network/interfaces
69 echo "bridge_stp off" >> /etc/network/interfaces
70 echo "bridge_fd 0" >> /etc/network/interfaces
71
72 #Suppression des ligne de eth0
73 sed -i '9d' /etc/network/interfaces
74 sed -i '9d' /etc/network/interfaces
75

```

```

76 #Redemarage du network
77 /etc/init.d/networking stop
78 /etc/init.d/networking start
79
80 #Supression des fichier temporaires et copie de common.sh
81 cd /root/
82 rm troll ipcluster ipgateway ipnetwork
83 rm /usr/share/ganeti/os/debootstrap/common.sh
84 mv common.sh /usr/share/ganeti/os/debootstrap/

```

Listing B.6 – Installation et configuration des noeuds esclaves

B.2.6 Ajout des noeuds au cluster et création des instances

```

1 #!/bin/bash
2 ganeti_slaves="/root/ganeti_slaves"
3
4
5 while read node
6 do
7     gnt-node add $node
8 done < $ganeti_slaves
9
10 echo "Combien d'instances par node ?? "
11 read nb
12 b=$nb
13 while read line
14 do
15     for i in `seq $a $nb`;
16     do
17         echo "10.0.1.$i instance$i" >> /etc/hosts
18         gnt-instance add -n $line -o debootstrap+default -t plain -s 2000 instance$i
19     done
20     nb=`echo $(( $nb + $b ))`
21     a=`echo $(( $a + $b ))`
22 done < ganeti_slaves

```

Listing B.7 – Ajout des noeuds au cluster et création des instances

B.2.7 Modification du fichier de configuration de ganeti

```

1 #!/usr/bin/python
2 #
3
4 # Copyright (C) 2010 Google Inc.
5 #
6 # This program is free software; you can redistribute it and/or modify
7 # it under the terms of the GNU General Public License as published by
8 # the Free Software Foundation; either version 2 of the License, or
9 # (at your option) any later version.
10 #
11 # This program is distributed in the hope that it will be useful, but
12 # WITHOUT ANY WARRANTY; without even the implied warranty of
13 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
14 # General Public License for more details.
15 #
16 # You should have received a copy of the GNU General Public License

```

```

17 # along with this program; if not, write to the Free Software
18 # Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA
19 # 02110-1301, USA.
20
21 """Tool to setup the SSH configuration on a remote node.
22
23 This is needed before we can join the node into the cluster.
24
25 """
26
27 def _VerifyServerKey(transport, host, host_keys):
28     """Verify the server keys.
29
30     @param transport: A paramiko.transport instance
31     @param host: Name of the host we verify
32     @param host_keys: Loaded host keys
33     @raises HostKeyVerificationError: When the host identify couldn't be verified
34
35     """
36
37     server_key = transport.get_remote_server_key()
38     keytype = server_key.get_name()
39
40     our_server_key = host_keys.get(host, {}).get(keytype, None)
41     if not our_server_key:
42         #hexified_key = _FormatFingerprint(server_key.get_fingerprint())
43         #msg = ("Unable to verify hostkey of host %s: %s. Do you want to accept"
44             #" it?" % (host, hexified_key))
45
46         #if cli.AskUser(msg):
47             our_server_key = server_key
48
49     if our_server_key != server_key:
50         raise HostKeyVerificationError("Unable to verify host identity")

```

Listing B.8 – Modification du fichier de configuration de ganeti

Ce fichier a été modifié pour permettre l'automatisation de l'ajout d'un nouveau noeud. Pour faire cela nous avons commenté les lignes

```

1 #hexified_key = _FormatFingerprint(server_key.get_fingerprint())
2     #msg = ("Unable to verify hostkey of host %s: %s. Do you want to accept"
3         #" it?" % (host, hexified_key))
4
5     #if cli.AskUser(msg):

```

B.3 Scripts pour le déploiement de virt-manager

```
1  #!/bin/bash
2
3  # On rcupre la liste des noeuds rserve et on les stocke dans list_nodes
4  cat $OAR_FILE_NODES | sort -u > $HOME/list_nodes
5  list_nodes="$HOME/list_nodes"
6
7  # Rappel des machines rserve
8  echo "-----"
9  echo "Liste des machines reservee:"
10 cat $list_nodes
11 echo "-----"
12 echo "Copie des fichiers vers toutes les machines."
13 for node in $(cat $list_nodes)
14 do
15     # Copie des clefs ssh sur chaque noeud pour faciliter les communications
16     scp $HOME/.ssh/id_rsa* root@$node:~/.ssh/
17     # Copie du script d'ajout de 7 vm au noeud
18     scp $HOME/vm.sh root@$node:~/
19     # Copie du script d'installation de virt-manager
20     scp $HOME/virt-install.sh root@$node:~/
21     # Copie de la liste des noeuds sur chacun des noeuds
22     scp $HOME/list_nodes root@$node:~/
23     # Copie d'un fichier de configuration de xend
24     scp $HOME/xend-config.sxp root@$node:~/
25     # Copie du script ajoutant automatiquement les noeuds virt-manager
26     scp $HOME/ajout_noeuds.sh root@$node:~/
27 done
28 echo "-----"
29
30 # Mise jour des paquets sur les noeuds
31 echo "Mise a jour des noyaux"
32 taktuk -l root -f $list_nodes broadcast exec [ apt-get update ]
33 taktuk -l root -f $list_nodes broadcast exec [ apt-get upgrade -y ]
34 taktuk -l root -f $list_nodes broadcast exec [ apt-get dist-upgrade -y ]
35
36 # Changement des mdp root
37 taktuk -l root -f $list_nodes broadcast exec [ 'echo -e "pttvirtu\npttvirtu" | passwd
    root' ]
38
39 # On s'assure que les droits sont bien positionns sur les diffrents scripts
40 taktuk -l root -f $list_nodes broadcast exec [ chmod 777 vm.sh ]
41 taktuk -l root -f $list_nodes broadcast exec [ chmod 777 virt-install.sh ]
42
43 echo "Lancement des scripts d'initialisation"
44 taktuk -l root -f $list_nodes broadcast exec [ bash vm.sh ]
45 taktuk -l root -f $list_nodes broadcast exec [ bash virt-install.sh ]
46
47 # On ajoute la clef aux clefs autorisees sur chaque noeud
48 taktuk -l root -f $list_nodes broadcast exec [ cat .ssh/id_rsa.pub >> .ssh/
    authorized_keys ]
49
50 # Une fois que taktuk s'est connect sur tous les noeuds, le fichier known_host contient
    les signatures de tous les noeuds.
51 # On le copie donc partout pour viter de valider lors de l'ajout d'une nouvelle
    connexion (ce qui provoquait un message d'erreur sous virt-manager)
52 for node in $(cat $list_nodes)
```

```

53 do
54     scp $HOME/.ssh/known_hosts root@$node:~/.ssh/known_hosts
55 done

```

Listing B.9 – Configuration générale et copie des scripts pour l’installation de virt-manager

```

1  #!/bin/bash
2  #####
3  ## Script d'installation de virt-manager ##
4  #####
5
6  # On s'assure qu'on est dans le home de root
7  cd /root
8
9  # Mise a jour du systme
10 #apt-get update
11 #apt-get upgrade -y
12
13 # Installation de libvirt pour communiquer avec le dmon xend
14 apt-get install libvirt-bin -y
15
16 # Installation de qemu-dm pour crer des machines virtuelles en mode full virtualis
17 apt-get install xen-qemu-dm-4.0 -y
18
19 # Correction d'un bug de qemu qui invalidait un lien symbolique
20 mkdir /usr/lib64/xen
21 mkdir /usr/lib64/xen/bin
22 cd /usr/lib64/xen/bin
23 ln -s /usr/lib/xen-4.0/bin/qemu-dm
24
25 # Activation des sockets Unix
26 # (necessaire pour la communication entre libvirt et xend)
27 # echo "(xend-unix-server yes)" >> /etc/xen/xend-config.sxp
28
29 # Maintenant on dploie un fichier complet plutot que d'ajouter la ligne ci-dessus
30 cp $HOME/xend-config.sxp /etc/xen/xend-config.sxp
31
32 # Redmarrage de xend pour application des changements
33 /etc/init.d/xend restart
34
35 # Installation de virt-manager
36 apt-get install virt-manager -y
37
38 #####
39 # Optionnel #
40 #####
41 # Lancement du script pour le dploiement de 7 machines virtuelles
42 #./vm.sh

```

Listing B.10 – Installation et configuration de virt-manager

```

1  #!/bin/bash
2  #####
3  # Virt-manager se configure via le gestionnaire gconf.
4  # On a donc opt pour la rcriture complte du fichier
5  # %gconf.xml
6  # Il est ncessaire d'attendre 1 min aprs la fermeture
7  # de virt-manager pour que le chemin de gconf soit cre
8  #####

```

```

9
10 # On recupre la liste des noeuds rserve (copi par scp)
11 list_nodes="$HOME/list_nodes"
12 echo "Cratation du fichier de config"
13
14 fichier=".gconf/apps/virt-manager/connections/%gconf.xml"
15
16 # Sauvegarde de l'ancien fichier de configuration
17 mv $fichier .gconf/apps/virt-manager/connections/%gconf.xml.BACKUP
18
19 # Cratation du nouveau fichier
20 touch $fichier
21 echo ".....OK"
22 echo " Gnration de l'entete"
23 echo '<?xml version="1.0"?>' >> $fichier
24 echo '<gconf>' >> $fichier
25 echo ' <entry name="autoconnect" mtime="1332080750" type="list" ltype="string">'
26 ' >> $fichier
27 echo ' <li type="string">' >> $fichier
28 echo '<stringvalue>xen:///</stringvalue>' >> $fichier
29 echo '</li>' >> $fichier
30 echo '</entry>' >> $fichier
31 echo '<entry name="uris" mtime="1332080863" type="list" ltype="string">' >> $fichier
32 echo ".....OK"
33 echo "ajout des noeuds"
34
35 #Pour chaque noeud rserve , on ajoute une entre dans le fichier
36 for node in $(cat $list_nodes)
37 do
38     echo '<li type="string">' >> $fichier
39     echo "<stringvalue>xen+ssh://root@$node/</stringvalue>" >> $fichier
40     echo '</li>' >> $fichier
41 done
42
43 echo ".....OK"
44 echo " Gnration de la fin du fichier"
45 echo '<li type="string">' >> $fichier
46 echo '<stringvalue>xen:///</stringvalue>' >> $fichier
47 echo '</li>' >> $fichier
48 echo '</entry>' >> $fichier
49 echo '</gconf>' >> $fichier
50 echo ".....OK"
51 echo "Vous pouvez relancer virt-manager"

```

Listing B.11 – Ajout des noeuds réservés à virt-manager

B.4 Scripts pour le déploiement d'Archipel

B.4.1 Installation d'Archipel

```
1  #!/bin/bash
2
3  #####
4  #-----Installation de Archipel-----#
5  #-----#
6
7  #chargement des noeuds reserve
8  list_nodes="$HOME/script_base/list_nodes"
9  list_node='cat $HOME/script_base/list_nodes'
10
11 #sauvegarde le 1er noeud qui sera le maitre
12 sed -n "1 p" $list_nodes > $HOME/archipel/archipel_manager
13 archipel_manager="$HOME/archipel/archipel_manager"
14 archipel_man='cat $HOME/archipel/archipel_manager'
15
16 #sauvegarde des noeud esclaves
17 cat $list_nodes | grep -v $archipel_man > $HOME/archipel/archipel_slaves
18 archipel_slaves="$HOME/archipel/archipel_slaves"
19 archipel_slav='cat $HOME/archipel/archipel_slaves'
20
21 echo $USER > username
22
23 echo "####Archipel Manager#####"
24 cat $archipel_manager
25 echo "#####"
26 echo "####Archipel Slaves#####"
27 cat $archipel_slaves
28 echo "#####"
29
30 #copie des fichier nssaisaire la configuration de archipel
31 while read line
32 do
33     scp -r sources/* root@$line:/usr/local/src
34     scp -r username root@$line:/usr/local/src
35     scp -r config_archipel.sh root@$line:/usr/local/src
36 done < $list_nodes
37
38 #lancement de la configuration de archipel sur le noeud maitre
39 taktuk -l root -f $archipel_manager broadcast exec [ sh /usr/local/src/config_archipel.
    sh ]
40
41 #scp archipel_master.sh root@$archipel_man:/root/
42
43 #lancement du script d'installation de ganeti sur le manager
44 #taktuk -l root -f $archipel_manager broadcast exec [ sh archipel_master.sh ]
45
46 #while read line
47 #do
48 #    scp ganeti_slave.sh root@$line:/root/
49 #    taktuk -l root -f $line broadcast exec [ sh archipel_slave.sh ]
50 #done < archipel_slaves
```

Listing B.12 – Installation d'Archipel

B.4.2 Configuration générale de Jabber pour Archipel

```
1 #Ajout des sources testing pour installer la derniere version de archipel
2 echo "## wheezy security" >> /etc/apt/sources.list
3 echo "deb http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/apt/
  sources.list
4 echo "deb-src http://security.debian.org/ wheezy/updates main contrib non-free" >> /etc/
  apt/sources.list
5 echo " " >> /etc/apt/sources.list
6 echo "#wheezy" >> /etc/apt/sources.list
7 echo "deb http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt/
  sources.list
8 echo "deb-src http://ftp.fr.debian.org/debian/ wheezy main contrib non-free" >> /etc/apt
  /sources.list
9
10 #Ajout du fichier de preference
11 echo "APT::Default-Release \"stable\";" > /etc/apt/apt.conf.d/80default-distrib
12
13 #installation des composants utils archipel
14 apt-get install -t testing ejabberd erlang-dev erlang-xmerl build-essential erlang-tools
  python-setuptools -y --force-yes
15
16 #configuration de ejabber
17 cd /usr/local/src/xmlrpc-1.13/src
18 make
19
20 cd /usr/local/src
21 cp -a xmlrpc-1.13 /usr/lib/erlang/lib/
22
23 cd /usr/local/src/ejabberd-modules/ejabberd_xmlrpc/trunk/
24 sh build.sh
25 cp ebin/ejabberd_xmlrpc.beam /usr/lib/ejabberd/ebin/
26
27 cd /usr/local/src/ejabberd-modules/mod_admin_extra/trunk/
28 sh build.sh
29 cp ebin/mod_admin_extra.beam /usr/lib/ejabberd/ebin/
30
31 #suppression du fichier de conf de base
32 rm /etc/ejabberd/ejabberd.cfg
33 cp /usr/local/src/ejabberd.cfg /etc/ejabberd/
```

Listing B.13 – Configuration générale de Jabber pour Archipel