Mise en place d'une plateforme IaaS basée sur OpenStack à l'Université Lille 1 : retour d'expérience

Xavier Buche
UFR IEEA
Universite Lille 1
Cité Scientifique - Bât. M5
59655 Villeneuve d'Ascq

Mohammed Khabzaoui
UFR de mathématiques
Universite Lille 1
Cité Scientifique - Bât. M2
59655 Villeneuve d'Ascq

Sébastien Fillaudeau
CRI
Universite Lille 1
Cité Scientifique - Bât. M4
59655 Villeneuve d'Ascq

Résumé

Le nouveau paradigme du cloud computing est en ce moment sur toutes les lèvres. Tout le monde en parle, mais bien peu connaissent les rouages de son fonctionnement. Et c'est la conséquence de l'une de ses caractéristiques attendues : celle d'être opaque. L'utilisateur n'a pas à connaître les détails de sa mise en œuvre.

Cet article s'adresse aux administrateurs systèmes et aux informaticiens curieux, avec l'ambition de leur faire entrevoir la machinerie intime d'un nuage et de le leur montrer sous un angle de vue inhabituel : celui de l'intérieur.

Mais des nuages, il en existe moult variétés. Celui dont nous parlons dans ces lignes est de type infrastructure. En d'autres termes, les choses manipulées en son sein sont les pendants virtuels d'objets du monde réel (serveurs, réseaux, routeurs, etc.).

En outre, bien que les services de cloud public soient légion, l'Université Lille 1 a considéré que sa taille et ses besoins étaient suffisants pour projeter la création d'un cloud privé.

De nombreuses solutions logicielles permettent de mettre en place ce type de nuage. Le choix s'est porté sur OpenStack, et l'essentiel de ce texte aborde les principaux aspects de sa configuration.

L'objet de cet article est ainsi de partager notre expérience acquise lors des différentes phases qui conduisent à la mise en production d'une plate-forme de cloud privé basée sur OpenStack. En particulier, plutôt que de borner nos explications à des généralités à l'intérêt limité, nous évoquons les raisons qui nous ont poussés à faire tel ou tel choix, ainsi que les pierres d'achoppement auxquelles nous avons été confrontés.

Nous abordons les points suivants :

- *les besoins*
- *le choix d'OpenStack*
- *le matériel*
- *les composants mis en place*
- *les principaux choix de configuration*
- *les principaux obstacles rencontrés*
- *le bilan*

Mots-clefs

openstack, cloud privé, IaaS, virtualisation, mutualisation, système, réseau

1 Introduction

L'augmentation continue des performances et des capacités des serveurs physiques ainsi que le développement des techniques de virtualisation ont permis ces dernières années la consolidation des infrastructures informatiques. Cette tendance de fond permet de tirer les bénéfices d'une mutualisation qui peut se faire soit dans le cadre d'une externalisation en recourant aux services d'hébergement commerciaux (Amazon, Google, Microsoft, OVH, etc.), soit en interne pour un établissement d'une taille suffisante, comme c'est le cas de l'Université Lille 1, ou bien encore dans un cadre multi-établissements (Université de Lille en perspective).

Pour amplifier la dynamique de mutualisation des serveurs informatiques au niveau de Lille 1 et répondre aux nouveaux besoins, un groupe de travail a été constitué en 2013 afin d'étudier les nouvelles solutions logicielles qui permettent de proposer un service d'hébergement de serveurs virtuels à grande échelle de type IaaS (Infrastructure as a Service) et pouvoir ainsi préparer l'évolution de l'offre de service d'hébergement mutualisée. La solution OpenStack s'est rapidement imposée.

Sébastien Fillaudeau, Mohammed Khabzahoui et moi-même, tous trois ingénieurs systèmes à l'Université de Lille 1, nous sommes alors proposés de mettre en place cette nouvelle plate-forme et de l'administrer. Pour ma part, ce projet a fait l'objet d'un mémoire d'ingénieur du CNAM.

2 Objectifs

L'Université Lille 1 doit fournir à ses utilisateurs un nouveau service simplifié de mise à disposition de machines virtuelles à la demande, c'est-à-dire une plate-forme de cloud IaaS.

La mise en place d'un cloud privé offre plusieurs avantages :

- mutualiser des ressources matérielles éparpillées et sous ou sur-utilisées
- permettre aux utilisateurs (enseignants, chercheurs, personnels des composantes, étudiants) de gérer eux-mêmes leurs machines virtuelles (création, suppression, allocation de ressources, etc.)

Un certain nombre de critères doivent orienter les décisions prises et les choix réalisés :

- allocation de ressources immédiate et à la demande
- supervision centralisée de l'infrastructure et facturation des utilisateurs en fonction de métriques précises
- gestion poussée des autorisations (droits d'accès, délégations, etc.)
- ergonomie de l'interface utilisateur

- facilités dans l'ajout ou la suppression de ressources matérielles
- sécurité et performances
- normalisation de l'API
- coûts

Quelques exemples d'utilisation plausibles :

- un étudiant doit réaliser un projet d'application web avec un framework de son choix
- un enseignant souhaite organiser des Travaux Pratiques de nosql avec hbase et hadoop
- un chercheur désire lancer ponctuellement des calculs
- un ingénieur voudrait mettre à disposition un nouveau service, comme une forge
- un développeur veut faire quelques tests ou mettre en ligne son travail avant de le valider

3 Solution logicielle

Une fois choisie, une plate-forme de cloud privé ne peut pas être remplacée facilement par une autre. L'expertise de sa mise en place, son administration, les développements engendrés nécessitent un investissement conséquent.

La pérennité de cet engagement est un critère de décision essentiel. Et OpenStack constitue, de ce point de vue, la solution libre offrant les meilleures garanties : la plupart des grands noms du secteur de l'informatique participent à ce projet.

OpenNebula , Nimbus, Cloudstack, Stratuslab, Eucalyptus sont d'autres solutions libres de cloud privé. Elles sont toutes éclipsées par OpenStack. La raison ? En quelques années, OpenStack s'est imposé comme un standard de fait. Et même si certaines des ces alternatives, comme Cloudstack, sont très abouties, il est peu probable qu'elles lui fassent un jour de l'ombre.

Dans l'avenir, les interactions entre des plates-formes de cloud différentes vont avoir tendance à se multiplier. Le cloud computing est donc un domaine de l'informatique qui, à l'instar des réseaux, favorise l'émergence de technologies convergentes. En d'autres termes, OpenStack risque fort, dans quelques années, d'être au cloud computing ce qu'IP est au réseau Internet : incontournable.

Dans ces conditions, les produits commerciaux comme VMware Vcloud ou Microsoft Azure ont un réel handicap, car ils tendent à emprisonner leurs clients dans l'utilisation de technologies (logiciels, protocoles, API) qui fonctionnent uniquement dans un cercle limité au bon vouloir du fournisseur. Et une fois à l'intérieur de ce cercle, difficile d'en sortir...

4 Mise en œuvre

Après une étude comparative des solutions IaaS et des premiers tests sur maquette d'OpenStack réalisés début 2014, six machines et un switch ont été commandés pour la mise en place d'un embryon de plate-forme permettant d'accueillir quelques centaines de machines virtuelles (environ 500 en fonction de leur usage). Une baie de disques Netapp est utilisée pour un stockage centralisé, via NFS, des images et des instances de machines virtuelles.

Ce chapitre décrit de manière partielle et synthétique la configuration de la plate-forme mise en place.

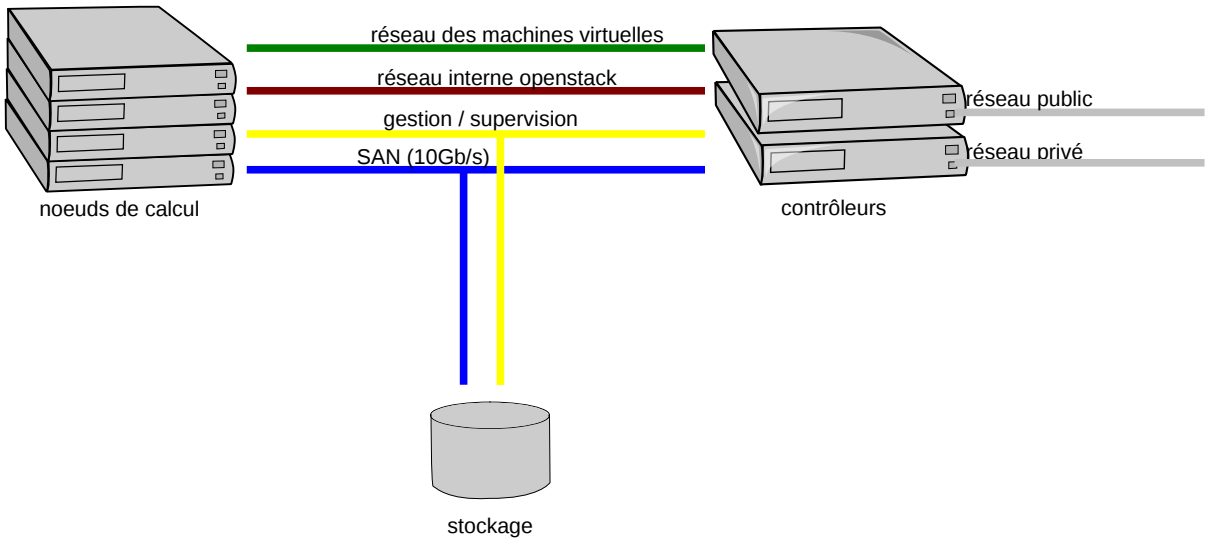

Figure 1 : schéma général

Les machines virtuelles sont exécutées par les hyperviseurs sur les nœuds de calcul. Les démons qui assurent le fonctionnement d'OpenStack (interfaces, schedulers, bus de messagerie, etc.) tournent sur les contrôleurs. Habituellement un seul des deux contrôleurs est actif. Nous avons prévu une procédure manuelle de basculement en cas de défaillance grâce du primaire.

L'application OpenStack est un projet fédérateur regroupant un certain nombre de « briques » (Nova, keystone, etc.). Chaque brique, qu'on appellera composant, remplit une fonction bien identifiée, et peut être commandée ou interrogée via une API REST basée sur HTTP.

Pour gérer leurs infrastructures virtuelles, les utilisateurs peuvent utiliser ces API. Mais, plus simplement, ils peuvent aussi utiliser des clients en ligne de commande ou l'interface web Horizon (voir plus loin).

En interne, les différents démons nécessaires au fonctionnement d'une brique communiquent ensemble par l'intermédiaire d'un bus de données logiciel AMQP implémenté par RabbitMQ. Chaque composant utilise sa propre base de données SQL, mise en œuvre par une grappe de trois serveurs MariaDB/Galera.

Nous utilisons la version Icehouse d'OpenStack fournie par le système Linux Ubuntu 14.04 LTS.

4.1 Keystone (gestion des identités)

Keystone est la pierre angulaire de la sécurité. Il réalise ou sous-traite l'authentification et assure les autorisations d'accès.

Il se base principalement sur trois « entités » :

- l'utilisateur
- le rôle
- le projet

On peut simplifier la relation entre ces trois entités en une phrase : un utilisateur a un certain rôle dans un certain projet.

Une ressource virtuelle (instance de machine, réseau, etc.) ne peut pas être associée directement à un utilisateur. Elle est associée soit à un projet, soit à un couple utilisateur-projet.

Des jetons à usage limité dans le temps, générés par Keystone lors de l'authentification, permettent d'identifier les utilisateurs et de vérifier leurs droits d'accès.

Ces droits d'accès sont configurés dans des fichiers qui associent chaque action à des rôles. Par exemple, on peut définir que seul le rôle *admin* dispose du droit de créer un nouveau projet.

Plutôt que de demander aux utilisateurs de créer un compte (login et mot de passe) pour accéder à OpenStack, avec ce que cela implique en terme d'administration, nous avons choisi d'utiliser la base des comptes de l'université, accessible via LDAP.

Le composant keystone inclut un backend LDAP, ce qui rend la configuration relativement simple. Cependant ce backend est très limité et n'est pas utilisable en production (pas de cache, pas de pool de connexions, une seule adresse de serveur LDAP configurable). Nous avons donc choisi d'installer un serveur LDAP local, sur les contrôleurs, qui maintient un cache des connexions LDAP de keystone vers les serveurs de l'université.

L'association des utilisateurs avec les rôles et les projets doit se faire dans une base de données. Nous avions le choix d'utiliser soit le SGBD MySQL, déjà utilisé par les différents composants d'OpenStack, soit le serveur LDAP local, utilisé pour le cache des comptes utilisateurs.

Nous avons opté pour l'utilisation de l'annuaire LDAP pour le stockage des rôles et des projets. Il nous a en effet semblé préférable de regrouper dans une même base des objets de même nature. Et dans une moindre mesure, pour des raisons de performances, car une base de données hiérarchique est plus adaptée pour ce type d'utilisation (très peu d'écritures, beaucoup de lectures).

Les versions supérieures d'OpenStack devraient nous permettre d'utiliser soit le SSO CAS de l'université, soit la fédération d'identité Shibboleth, pour authentifier les utilisateurs et récupérer quelques-uns de leurs attributs.

C'est une prochaine étape importante du projet.

Keystone se charge également de la gestion des droits d'accès. Pour cela, il se fonde sur les rôles associés aux utilisateurs et aux projets.

Chaque composant d'OpenStack dispose d'un fichier au format JSON, qui associe les différentes actions de ce composant avec des rôles.

Par exemple, la ligne suivante, présente dans le fichier /etc/glance/policy.json interdit aux utilisateurs dont le rôle est « etudiant » de rendre publique leurs images de machine virtuelle.

```
"publicize_image": "not role:etudiant"
```

4.2 Glance (gestion des images)

Glance est le composant qui gère les images de machines virtuelles. Autrement dit les modèles qui servent à créer des instances de machines virtuelles.

La configuration de Glance est triviale et n'a posé aucun problème particulier. Chaque utilisateur peut, selon un certain quota, transférer sur le cloud des images de machines virtuelles, qui sont alors stockées sur la baie de disques et éventuellement partagées avec d'autres utilisateurs.

Des images adaptées à une utilisation avec OpenStack de la plupart des systèmes d'exploitations et distributions Linux sont fournies par les éditeurs. Mais il est également possible pour l'utilisateur de préparer une image à une utilisation dans le cloud avec des outils comme cloud-init.

Nous avons mis à disposition des utilisateurs quelques images « standard » basées sur les principales distributions Linux, ainsi qu'une image MS Windows Server[1].

1. en version d'évaluation (http://www.microsoft.com/en-us/evalcenter/evaluate-windows-server-2012)

4.3 Nova (gestion des instances)

Nova constitue le cœur du cloud OpenStack. Glance lui fournit des images qu'il « transforme » en instances de machines virtuelles exécutées sur les différents nœuds de calculs.

Nous avons configuré Nova de manière à tirer parti d'un cache disque situé sur la baie de stockage et partagé par tous les nœuds de calcul. Ce cache contient une copie des images, utilisables en lecture seule et partagées par les hyperviseurs. De plus, les instances ne sont donc pas répliquées : seules les modifications effectuées sur chaque instance sont écrites dans les fichiers des disques virtuels, au format QCOW2. Ces derniers occupent uniquement l'espace disque utilisé par les données qu'ils contiennent, et non la taille du disque de la machine virtuelle.

Ces mécanismes offrent le grand avantage de restreindre drastiquement l'espace occupé sur la baie de disque. Ceci au détriment (relatif) des performances.

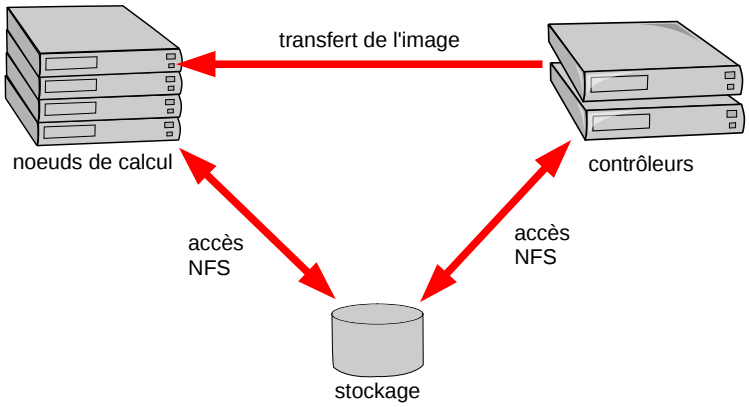

Figure 2 : mise en cache des images

Par ailleurs, le stockage centralisé nous donne la possibilité de réaliser la migration à chaud des machines virtuelles, par exemple pour procéder aux opérations de maintenance en « évacuant » successivement les nœuds.

La plupart des hyperviseurs actuels peuvent être utilisés avec Nova. Nous avons choisi KVM, via libvirt, car il est fiable, relativement performant, offre une bonne isolation des instances, et surtout, il fournit le plus grand support de la part de la communauté OpenStack.

Les agrégats OpenStack permettent d'utiliser, côte à côte, des hyperviseurs différents répartis sur des groupes de nœuds de calculs. Nous prévoyons à l'avenir d'utiliser éventuellement ces agrégats pour ajouter, en plus du cluster de nœuds KVM actuels, des clusters VMware ou Xen utilisés actuellement en dehors de l'infrastructure OpenStack.

Les utilisateurs ont la possibilité de créer un instantané de leurs machines virtuelles. Dans le contexte d'OpenStack, cette notion n'est pas tout à fait identique à celle habituellement employée sur les volumes disques, par exemple. Ici, en plus de conserver l'état d'une machine à un instant donné, l'idée est de transformer cette machine en image. Les instantanés peuvent donc être employés, par exemple, par un enseignant désirant créer une nouvelle image à partir d'une image existante après l'avoir modifiée pour y inclure les applications nécessaires à son TP.

L'accès aux machines virtuelles peut se faire avec une console VNC intégrée à l'interface web Horizon. Cette console étant très limitée (performances réduites, bogues en mode graphique, mappage du clavier défaillant), nous incitons les utilisateurs à utiliser plutôt ssh sous Linux et RDP sous Windows. Nous avons également testé le client HTML SPICE comme alternative à VNC, sans plus de succès.

Pour se connecter à la machine virtuelle qu'il vient de créer, que ce soit par ssh, la console ou un autre moyen, l'utilisateur doit avoir défini un identifiant. Pour ceci, nous avons configuré trois méthodes différentes :

- un mot de passe « root » peut être indiqué par l'utilisateur et injecté dans la machine virtuelle au moment de sa création
- l'utilisateur peut également écrire un script cloud-init ou shell qui sera exécuté lors du premier démarrage de l'instance pour créer un compte, par exemple
- enfin, l'utilisateur peut générer une paire de clefs ssh et injecter la clef publique dans la machine virtuelle lors de sa création

4.4 Neutron (gestion des réseaux)

Deux composants, au choix, peuvent être utilisés pour fournir un accès réseau aux machines virtuelles.

- Nova-network est le composant historique. Ses possibilités sont limitées.
- Neutron est la solution d'avenir. Il fournit aux utilisateurs la possibilité de gérer ses propres ressources virtuelles (réseaux, routeurs, pare-feu, etc.)

Nous avons choisi Neutron et l'avons configuré de manière à ce que les réseaux ethernet virtuels créés par les utilisateurs soient discriminés, sur le réseau physique, par le protocole 802.1Q (VLAN). Neutron offre deux alternatives à ce protocole : GRE et VXLAN. Mais quelques tests ont révélé qu'ils sont beaucoup moins performants. Cependant cet inconvénient pourrait probablement être pallié, au moins partiellement, par l'utilisation de cartes réseau supportant le déchargement matériel de ces protocoles. A l'avenir, s'il existe plusieurs sites OpenStack distants, nous pourrons éventuellement les utiliser afin d'établir des tunnels et de partager les réseaux virtuels des utilisateurs entre les sites. Soit dit en passant, conserver les réseaux virtuels existants lors d'un changement de protocole n'est pas anodin.

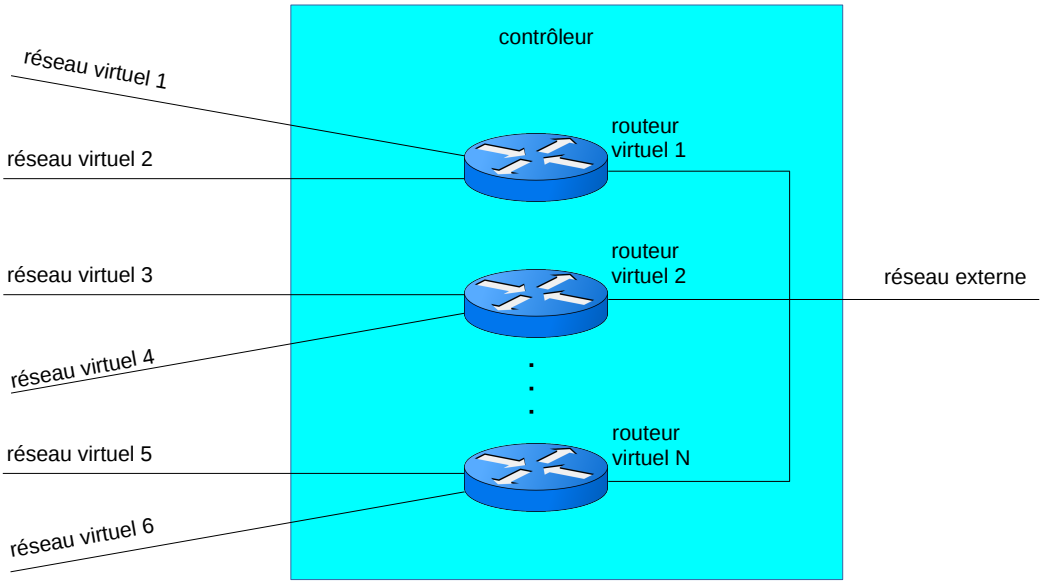

Figure 3 : réseaux virtuels

Au niveau de la topologie physique, seuls les contrôleurs, qui assurent la fonction de routage et de translation d'adresses IP, sont visibles de l'extérieur. Les nœuds de calcul, sur lesquels les machines

virtuelles sont exécutées, sont connectés à un réseau physique interne accessible uniquement des contrôleurs.

Plutôt que de gérer les règles de filtrage réseau au niveau de chacune de leurs machines virtuelles, les utilisateurs peuvent configurer des groupes de sécurité, qui sont des listes de contrôle d'accès qui s'appliquent aux instances de machines virtuelles. Par exemple, un utilisateur peut définir un groupe de sécurité nommé « serveurs web », qui ouvre uniquement quelques ports dont les ports TCP 80 et 443 en entrée, et appliquer ce groupe à toutes les instances qui font office de serveur web.

4.5 Ceilometer (mesures)

Un service de cloud computing doit pouvoir mesurer les ressources matérielles consommées par chaque utilisateur. Cette mesure permet de réaliser une facturation selon un processus en trois étapes :

- mesure
- classification
- facturation

Dans ce processus, Ceilometer assure uniquement la première étape. Les autres étapes sont à la charge d'outils tiers. A l'heure actuelle, nous n'avons pas développé cet aspect d'OpenStack. Néanmoins nous avons configuré Ceilometer pour récupérer régulièrement diverses données suffisantes pour une exploitation ultérieure (temps processeur, mémoire, espace disque, etc.).

Pour libérer automatiquement des ressources consommées inutilement, nous réfléchissons à la mise en place d'un système qui désactiverait les machines virtuelles inutilisées, après envoi de mails aux utilisateurs.

4.6 Horizon (interface web de management)

Horizon est l'application web faisant office de panneau de contrôle d'OpenStack. Les utilisateurs se connectent à cette interface pour gérer leurs machines virtuelles, leurs images, leurs réseaux, etc.

Mais Horizon est plus qu'une simple interface, c'est aussi un framework, basé sur Django, que nous pouvons utiliser pour modifier facilement l'apparence du site et ses possibilités.

Jusqu'à présent, la grande majorité des utilisateurs se basent sur Horizon pour gérer leurs infrastructures virtuelles. Nous l'avons légèrement modifiée pour faciliter certaines tâches, comme l'attribution d'adresses IP ou la création de machines virtuelles.

Cette interface, même si elle est moderne et ergonomique, ne dispose pas de toutes les possibilités qu'offrent les API d'OpenStack. Par exemple, elle ne permet pas de partager une image avec des utilisateurs particuliers, ni de connecter une machine virtuelle à plusieurs réseaux. Elle mérite donc quelques améliorations que nous envisageons de réaliser par la suite.

A l'inverse, cette interface fournit des possibilités qui vont bien au-delà des besoins d'une grande partie des utilisateurs, comme celles relatives aux réseaux virtuels. Par conséquent, nous prévoyons de développer une interface simplifiée à l'extrême et intégrée au portail de l'université, pour satisfaire les besoins les plus basiques : créer une machine virtuelle, l'arrêter, la redémarrer, la supprimer, etc.

4.7 Quotas

Il est évident que des milliers d'utilisateurs ne peuvent pas se partager des ressources sans que des limites ne leurs soient imposées. Un des éléments primordiaux d'un logiciel de cloud computing est donc la gestion de quotas d'utilisation des ressources.

Avec OpenStack, les quotas des images, des instances et des réseaux ne sont pas administrés de manière identique. Un inconvénient issu des origines d'OpenStack, qui regroupe des composants développés, au départ, sans liens entre eux.

Ainsi, Glance a une gestion extrêmement limitée des quotas qui consiste essentiellement à définir un espace disque maximum par utilisateur pour le stockage des images. Alors que Nova et Neutron permettent de régler beaucoup plus finement les quotas (par projet ou par couple utilisateur/projet) sur des ressources variées (nombre d'instances, quantité de mémoire vive, nombre d'adresses IP externes, etc.).

4.8 Outils annexes

Pour un fonctionnement en production, OpenStack a besoin de nombreux outils tiers. Nous avons notamment utilisé :

- cluster MySQL + Galera avec réplication des bases de données en temps réel
- OpenLDAP pour le cache des comptes utilisateurs et le stockage des projets et des rôles
- Openvswitch pour la gestion des commutateurs virtuels
- Nagios et Cacti pour la supervision
- l'infrastructure existante de l'université pour la sauvegarde des images, instances et bases de données

5 Bilan et perspectives

Par rapport aux autres solutions de cloud computing, la mise en place et la maîtrise d'OpenStack est singulièrement complexe. Cette complexité n'est cependant pas gratuite : elle est la contre-partie d'une richesse fonctionnelle hors du commun, d'une modularité poussée à l'extrême.

Et pourtant, à plusieurs reprises, nous avons été surpris de constater certaines lacunes, comme par exemple l'impossibilité d'assigner un quota général par utilisateur (il faut passer par les projets), ou encore l'absence de fonction d'assignation automatique d'adresses IP externes aux instances.

Par ailleurs, et malgré le soin et l'importance portés sur le développement d'OpenStack par des spécialistes du monde entier, nous avons été confrontés à un nombre non négligeable de bugs ou d'incohérences. Certaines fois, une simple mise à jour suffisait à en venir à bout. Mais d'autres fois, c'est une mise à jour qui était à l'origine de l'ennui... Il fallait alors chercher des moyens de contournement, et parfois signaler certains problèmes sur la plate-forme communautaire de gestion des bugs d'OpenStack.

Cela dit, et pour sa défense, OpenStack est très jeune, il évolue très vite et de nombreuses nouvelles fonctionnalités sont régulièrement proposées et ajoutées. Précisons enfin que les problèmes se sont toujours posés après que des modifications ont été faites. Jamais nous n'avons constaté d'aléas ou d'imprévus inexpliqués dans le fonctionnement d'OpenStack. Une fois correctement configuré, il s'est toujours révélé parfaitement stable.

Le bilan est également mitigé en ce qui concerne la documentation officielle. Elle est conséquente et de bonne facture, et elle suffit à réaliser une installation basique et à comprendre les grandes lignes du fonctionnement d'OpenStack. Mais elle est tout sauf exhaustive : certaines parties essentielles, comme la gestion des droits d'accès, sont abordées trop succinctement, et la plupart des milliers de directives de configuration sont expliquées de façon lapidaire, en une phrase.

Heureusement, de nombreux blogs et autres wikis, souvent rédigés par des développeurs d'OpenStack reconnus, parsèment le réseau des réseaux et lèvent le voile sur les facettes obscures du paramétrage de certains composants.

A ce stade, de nombreuses choses ont été réalisées, mais il reste encore beaucoup d'améliorations à apporter :

- traiter les données engrangées par Ceilometer pour obtenir des statistiques d'usage des ressources
- détecter l'inactivité des ressources virtuelles pour les désactiver automatiquement et économiser des ressources
- attribuer automatiquement un nom DNS aux instances, en plus d'une adresse IP externe
- faire en sorte que seuls les utilisateurs autorisés puissent obtenir des adresses flottantes sur le réseau externe public, et en quantité limitée
- migrer vers la dernière version stable d'OpenStack
- utiliser le protocole Shibboleth pour authentifier les utilisateurs
- éventuellement, utiliser Cinder en conjonction avec les pilotes NetApp pour optimiser le processus de création des volumes et des instances
- intégrer une interface web simplifiée au portail de l'université
- mettre en place une reprise sur panne automatique du contrôleur
- synchroniser les projets avec les groupes LDAP de type « OpenStack » créés par les utilisateurs avec l'outil du CRI

Le cloud OpenStack est actuellement en cours de beta-test et est essentiellement utilisé pour réaliser des plate-forme de tests ou pour des besoins liés à l'enseignement (projets d'étudiants et TP). Elle s'ouvre progressivement à un nombre croissant d'utilisateurs.

6 Conclusion

La mise en place d'OpenStack, le fer de lance des plates-formes d'infrastructure en nuage, est à la croisée des chemins entre de multiples domaines de l'informatique : architecture système, réseau, virtualisation, gestion de bases de données, développement web... Force est de constater que la réalisation d'un tel projet exige une grande rigueur et mobilise des compétences éclectiques.

Ce projet a mis au jour un grand nombre d'écueils, des bugs, des anomalies ou incohérences diverses, qu'il a fallu corriger ou contourner. Mais surtout, il nous a donné la satisfaction d'aboutir à un résultat en rapport avec la popularité d'OpenStack.

Nous espérons avoir su partager et dévoiler, dans ces pages, quelques-uns des aspects techniques du cloud computing en général, et de la plate-forme OpenStack en particulier.