FORMATION OPENSTACK UTILISATEUR

CONCERNANT CES SUPPORTS DE COURS

CONCERNANT CES SUPPORTS DE COURS 1/2

Supports de cours réalisés par **Osones** https://osones.com

Logo Osones

AUTEURS

- Adrien Cunin adrien.cunin@osones.com
- Pierre Freund pierre.freund@osones.com
- Jean-François Taltavull jft@osones.com
- Romain Guichard romain.guichard@osones.com
- Kevin Lefevre kevin.lefevre@osones.com

CONCERNANT CES SUPPORTS DE COURS 2/2

INTRODUCTION

OBJECTIFS DE LA FORMATION

- Assimiler les concepts et le vocabulaire liés au cloud
- Définir, déployer et maintenir une infrastructure dans le cloud
- Concevoir une application cloud-ready
- Manipuler et orchestrer des ressources dans un cloud OpenStack

Public visé : intégrateurs et développeurs d'application

PRÉ-REQUIS DE LA FORMATION

- Pratique du langage de commande Linux (Shell)
- Notions de virtualisation
- Optionnel :
  - Notions de Python (langage et environnement)

LE CLOUD : VUE D'ENSEMBLE

LE CLOUD : LES CONCEPTS

- Puissance de calcul, capacité de stockage et fonctionnalités réseau sous forme de services en-ligne
- Flexibilité et élasticité des infrastructures
- Libre-service, notion de catalogue
- Accès en mode programmatique par des APIs, automatisation
- Facturation à l'usage

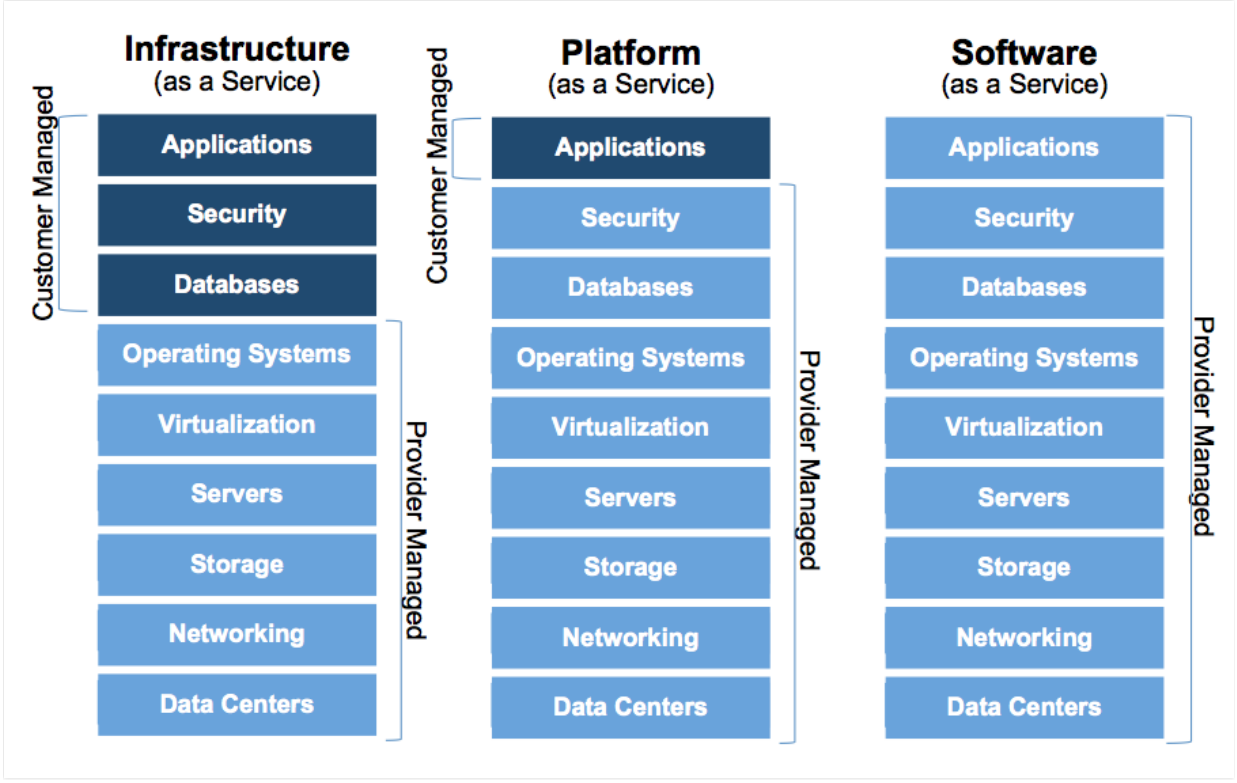
WAAS : WHATEVER AS A SERVICE

- Principalement :
  - IaaS → Infrastructure as a Service
  - PaaS → Platform as a Service
  - SaaS → Software as a Service
  - Mais aussi :
    - Object Storage as a Service
    - Database as a Service
    - Load Balancing as a Service
    - DNS as a Service
    - $Application as a Service

CLOUD PUBLIC, CLOUD PRIVÉ, CLOUD HYBRIDE ?

- Public → services cloud proposés par un fournisseur externe (AWS, Rackspace, OVH, etc.)
- Privé → services cloud proposés par une entreprise/organisation à ses propres départements
- Hybride → utilisation des services d'un ou plusieurs clouds publics au sein d'un cloud privé

LE CLOUD EN UN SCHÉMA



IaaS - PaaS - SaaS

POURQUOI MIGRER VERS LE CLOUD ?

Motivations d'ordre économique :

- appréhender les ressources IT comme des services "fournisseur"
- réduire les coûts en mutualisant les ressources
- réduire les délais d'approvisionnement de ressources
- faire glisser le budget "investissement" (Capex) vers le budget "fonctionnement" (Opex)
- aligner les coûts sur la consommation réelle des ressources
- automatiser les opérations sur le SI et le rendre ainsi plus flexible

POURQUOI MIGRER VERS LE CLOUD ?

Motivations d'ordre technique :

- abstraire les couches basses (serveur, réseau, OS, stockage)
- s'affranchir de l'administration technique des ressources et services (BdD, pare-feux, load balancing,...)
- concevoir des infrastructures scalables à la volée
- agir sur les ressources via des lignes de code et gérer les infrastructures "comme du code"

IAAS : INFRASTRUCTURE AS A SERVICE

LE LEADER DU IAAS PUBLIC : AMAZON WEB SERVICES (AWS)

- Pionnier du marché (dès 2006)
- Elastic Compute Cloud (EC2) → puissance de calcul
- Elastic Block Storage (EBS) → stockage bloc
- Simple Storage Service (S3) → stockage objet

Logo Amazon Web Services (AWS)

LES CLOUDS PUBLICS CONCURRENTS D'AWS

- dans le monde :
  - Google Cloud Platform
  - Microsoft Azure
  - Rackspace
  - DreamHost
  - DigitalOcean
- en France :
  - Cloudwatt (Orange Business Services)
  - Numergy (SFR)
  - Outscale
  - OVH
  - Ikoula
  - Scaleway (Iliad)

LOGICIELS LIBRES PERMETTANT LE DÉPLOIEMENT D'UN CLOUD PRIVÉ

- **OpenStack** (https://www.openstack.org)
- Eucalyptus (société rachetée par HP en septembre 2014)
- CloudStack (https://cloudstack.apache.org/)
- OpenNebula (http://opennebula.org/)

CORRESPONDANCE OPENSTACK - AWS

- Compute → EC2 → Nova
- Block storage → EBS → Cinder
- Object storage → S3 → Swift
- Orchestration → CFN → Heat

VIRTUALISATION DANS LE CLOUD

- Un cloud IaaS repose souvent sur la virtualisation
- Ressources “compute” ← virtualisation
- Hyperviseurs : KVM, Xen (libvirt), ESX
- Conteneurs : OpenVZ, LXC, LXD
- Conteneurs : Docker

NOTIONS ET VOCABULAIRE IAAS 1/4

- Identité et accès
  - Tenant/Projet (Project) : locataire du cloud, propriétaire de ressources.
  - Utilisateur (User) : compte autorisé à utiliser les API OpenStack.
  - Quota : contrôle l'utilisation des ressources (vcpu, ram, fip, security groups,...) dans un tenant.
  - Catalogue (de services) : services disponibles et accessibles via les API.
  - Endpoint : URL permettant l'accès à une API. Un endpoint par service.

NOTIONS ET VOCABULAIRE IAAS 2/4

- Calcul/Serveurs (Compute)
  - Image : généralement, un OS bootable et “cloud ready”.
  - Instance : forme dynamique d’une image.
  - Type d’instance (flavor) : mensurations d’une instance (cpu, ram, capacité disque,...).
  - Metadata et user data : informations gérées par le IaaS et mises à disposition de l’instance.
  - Cloud-init, cloud-config : mécanismes permettant la configuration finale automatique d’une instance.

NOTIONS ET VOCABULAIRE IAAS 3/4

- Stockage (Storage)
  - Volume : disque virtuel accessible par les instances (stockage “block”).
  - Conteneur (Container) : entités logiques pour le stockage de fichiers et accessibles via une URL (stockage “objet”).
- Réseau et sécurité (Network, Security)
  - Groupe de sécurité (Security groups) : ensemble de règles de filtrage de flux appliqué à l’entrée des instances.
  - Paire de clés (Keypairs) : clé privée + clé publique permettant les connexions aux instances via SSH.
  - IP flottantes (Floating IP) : adresse IP allouée à la demande et utilisée par les instances pour communiquer avec le réseau “externe”.

NOTIONS ET VOCABULAIRE IAAS 4/4

- Orchestration
  - Stack : ensemble des ressources IaaS utilisées par une application.
  - Template : fichier texte contenant la description d'une stack.

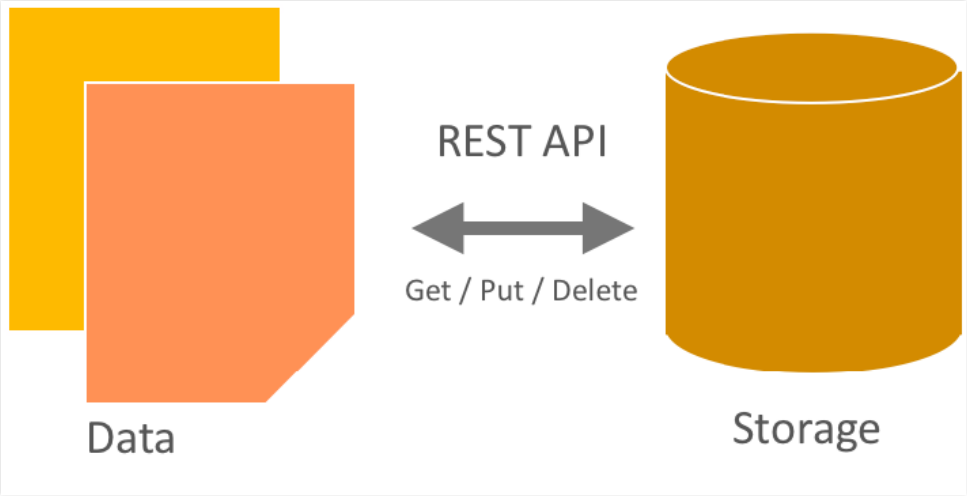
STOCKAGE BLOCK

- Accès à des raw devices type */dev/vdb*
- Possibilité d'utiliser n'importe quel système de fichiers
- Compatible avec toutes les applications legacy

STOCKAGE OBJET

- Pousser et retirer des objets dans un container/bucket
- Pas de hiérarchie des données, pas de système de fichiers
- Accès par les APIs
- L'application doit être conçue pour tirer parti du stockage objet

STOCKAGE OBJET : UNE API



API de stockage objet

ORCHESTRER LES RESSOURCES DE SON IAAS

- Définir tout une infrastructure dans un seul fichier texte
- Être en capacité d'instancier une infrastructure entière en un appel API
- Autoscaling
  - Adapter ses ressources en fonction de ses besoins en temps réel
  - Fonctionnalité incluse dans le composant d'orchestration d'OpenStack

OPENSTACK EST UNE API

- *Application Programming Interface*
- Au sens logiciel : Interface permettant à un logiciel d'utiliser une bibliothèque
- Au sens cloud : Interface permettant à un logiciel d'utiliser un service (XaaS)
- Il s'agit le plus souvent d'API HTTP REST

REST

- Une ressource == une URI (*Uniform Resource Identifier*)
- Utilisation des verbes HTTP pour caractériser les opérations (CRUD)
  - GET
  - POST
  - PUT
  - DELETE
- Utilisation des codes de retour HTTP
- Représentation des ressources dans le corps des réponses HTTP

REST - EXEMPLES

```
GET http://endpoint/volumes/
GET http://endpoint/volumes/?size=10
POST http://endpoint/volumes/
DELETE http://endpoint/volumes/xyz
```

EXEMPLE CONCRET

```
GET /v2.0/networks/d32019d3-bc6e-4319-9c1d-6722fc136a22
{
  "network":{
    "status":"ACTIVE",
    "subnets":[ "54d6f61d-db07-451c-9ab3-b9609b6b6f0b" ],
    "name":"private-network",
    "provider:physical_network":null,
    "admin_state_up":true,
    "tenant_id":"4fd44f30292945e481c7b8a0c8908869",
    "provider:network_type":"local",
    "router:external":true,
    "shared":true,
    "id":"d32019d3-bc6e-4319-9c1d-6722fc136a22",
    "provider:segmentation_id":null
  }
}
```

PAAS : PLATFORM AS A SERVICE

PAAS : CONCEPT D'APPLICATION MANAGÉE

- Fourniture d'une plate-forme de "build, deploy and scale"
- Pour un langage / un framework (Python, Java, PHP, etc.)
- Principalement utilisé par des développeurs d'applications

- Peut également désigner les *as a Service* type : DB, Queue, etc.

EXEMPLES DE PAAS PUBLIC

- Amazon Elastic Beanstalk (https://aws.amazon.com/fr/elasticbeanstalk)
- Google App Engine (https://cloud.google.com/appengine)
- Heroku (https://www.heroku.com)

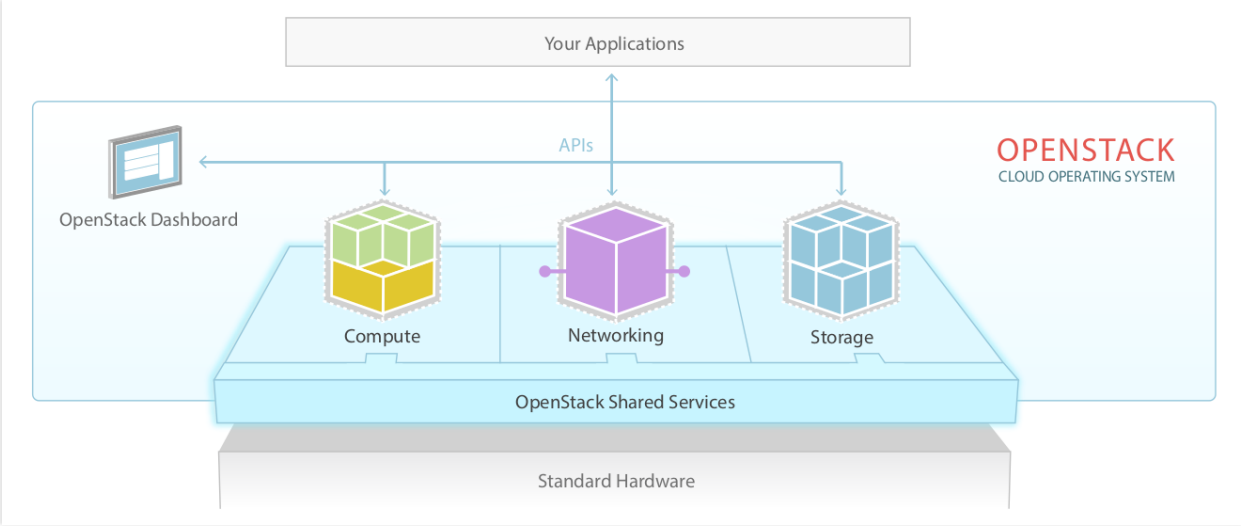
SOLUTIONS DE PAAS PRIVÉ

- Cloud Foundry (https://www.cloudfoundry.org)
- OpenShift (http://www.openshift.org)
- Solum (https://wiki.openstack.org/wiki/Solum)

OPENSTACK : LE PROJET

TOUR D'HORIZON

VUE HAUT NIVEAU



Version simple

HISTORIQUE

- Démarrage en 2010
- Objectif : le Cloud Operating System libre
- Fusion de deux projets de Rackspace (Storage) et de la NASA (Compute)
- Logiciel libre distribué sous licence Apache 2.0
- Naissance de la Fondation en 2012

LES RELEASES

- Austin (2010.1)
- Bexar (2011.1), Cactus (2011.2), Diablo (2011.3)
- Essex (2012.1), Folsom (2012.2)
- Grizzly (2013.1), Havana (2013.2)
- Icehouse (2014.1), Juno (2014.2)
- Kilo (2015.1), Liberty (2015.2)
- Mitaka (2016.1), **Newton** (2016.2)
- Début 2017 : Ocata

STATISTIQUES : KILO

- 1492 contributeurs (Liberty : 1933)
- 169 organisations
- 394 nouvelles fonctionnalités et 7257 bugs corrigés
- 113 drivers/plugins
- 792200 chaines traduites

Source : http://lists.openstack.org/pipermail/foundation-board/2015-April/000050.html

QUELQUES SOUTIENS/CONTRIBUTEURS ...

- Editeurs : Red Hat, HP, IBM, Suse, Canonical, Vmware, ...
- Constructeurs : Dell, Hitachi, Juniper, Cisco, NetApp, ...
- En vrac : NASA, Yahoo, OVH, Citrix, SAP, Rackspace, ...
- **Google** ! (depuis juillet 2015)

http://www.openstack.org/foundation/companies/

... ET UTILISATEURS

- Tous les contributeurs précédemment cités
- En France : **Cloudwatt** et **Numergy**
- Wikimedia
- CERN
- Paypal
- Comcast
- BMW
- Etc. Sans compter les implémentations confidentielles

http://www.openstack.org/user-stories/

LES DIFFÉRENTS SOUS-PROJETS

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron
- OpenStack Image Service - Glance
- OpenStack Identity Service - Keystone
- OpenStack Dashboard - Horizon
- OpenStack Telemetry - Ceilometer
- OpenStack Orchestration - Heat
- OpenStack Database Service - Trove

LES DIFFÉRENTS SOUS-PROJETS (2)

- Mais aussi :
  - Bare metal (Ironic)
  - Queue service (Zaqar)
  - Data processing (Sahara)
  - DNS service (Designate)
  - Shared File Systems (Manila)
  - Key management (Barbican)
  - PaaS (Solum)
  - Container (Magnum)
- Autres
  - Les clients (python-*client)

LES 4 OPENS

- Open Source
- Open Design
- Open Development
- Open Community

LA FONDATION OPENSTACK

- Entité de gouvernance principale du projet
- Représentation juridique du projet
- Les membres du board sont issus des entreprises sponsors et élus par les membres individuels
- Tout le monde peut devenir membre individuel (gratuitement)

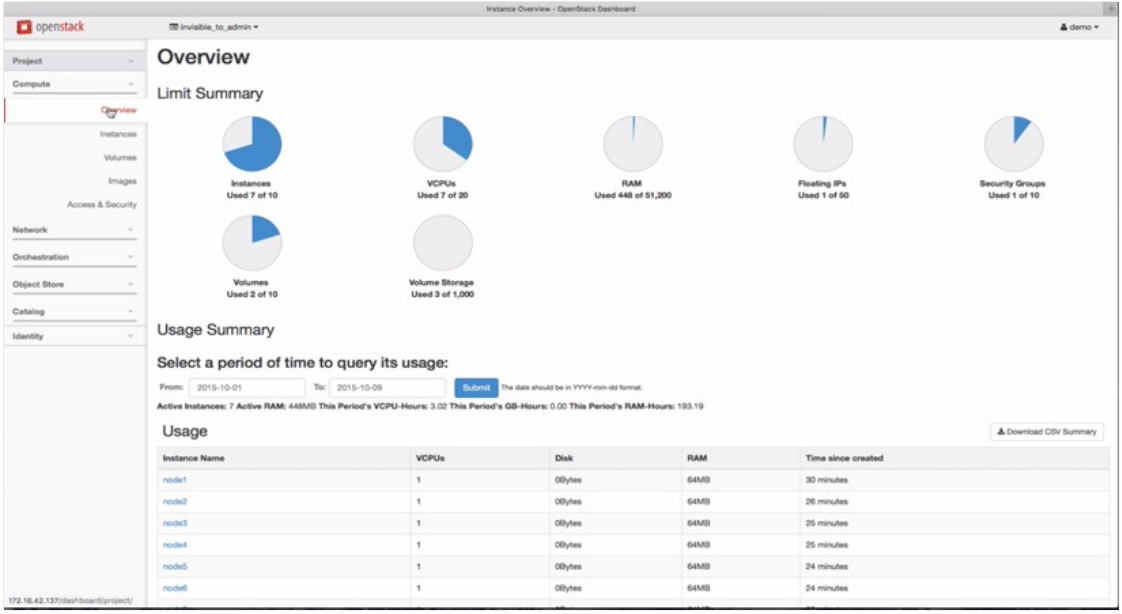
LA FONDATION OPENSTACK

- La fondation supporte le projet par différents moyens :
  - Événements : organisation (Summits) ou participation (OSCON, etc.)
  - Infrastructure de développement (serveurs)
  - Ressources humaines : marketing, release manager, quelques développeurs (principalement sur l'infrastructure)
- 500 organisations à travers le monde
- 23000 membres individuels dans 160 pays

LA FONDATION OPENSTACK

Les principales entités de la fondation

INTERFACE WEB / DASHBOARD : HORIZON



Screenshot Horizon (Liberty)

RESSOURCES

- Annonces/sécurité : openstack-announce@lists.openstack.org
- Documentation : http://docs.openstack.org/
- Gouvernance du projet : http://governance.openstack.org/
- Support :
  - https://ask.openstack.org
  - openstack@lists.openstack.org
  - #openstack@Freenode
- SDK/APIs : http://developer.openstack.org/

RESSOURCES

- Applications : http://apps.openstack.org/
- Actualités :
  - Blog officiel : http://www.openstack.org/blog/
  - Planet : http://planet.openstack.org
  - Superuser : http://superuser.openstack.org/
  - OpenStack Community Weekly Newsletter
- Ressources commerciales : http://www.openstack.org/marketplace/ entre autres

RESSOURCES - COMMUNAUTÉ FRANCOPHONE

Logo OpenStack-fr

- Site web : http://openstack.fr/
- Association des utilisateurs francophones d'OpenStack : https://asso.openstack.fr/
- Meetups : Paris, Rhônes-Alpes, Toulouse, Montréal, etc.
- Présence à des événements tels que *Paris Open Source Summit*
- Canaux de communication :
  - openstack-fr@lists.openstack.org
  - #openstack-fr@Freenode

TIRER PARTI DU IAAS

DEUX VISIONS

Une fois un cloud IaaS en place, deux optiques possibles :

- Garder les mêmes pratiques tout en profitant du self service et de l'agilité de la solution pour des besoins test/dev
- Faire évoluer ses pratiques, tant côté applicatif que système "Pets vs Cattle"

SINON ?

Faire tourner des applications *legacy* dans le cloud est une mauvaise idée :

- Interruptions de service
- Pertes de données
- Incompréhensions “le cloud ça marche pas”

CÔTÉ APPLICATIONS

ADAPTER OU PENSER SES APPLICATIONS "CLOUD READY" 1/3

Cf. les design tenets du projet OpenStack et Twelve-Factor
http://12factor.net/

- Architecture distribuée plutôt que monolithique
  - Facilite le passage à l'échelle
  - Limite les domaines de *failure*
- Couplage faible entre les composants

ADAPTER OU PENSER SES APPLICATIONS "CLOUD READY" 2/3

- Bus de messages pour les communications inter-composants
- Stateless : permet de multiplier les routes d'accès à l'application
- Dynamicité : l'application doit s'adapter à son environnement et se reconfigurer lorsque nécessaire
- Permettre le déploiement et l'exploitation par des outils d'automatisation

ADAPTER OU PENSER SES APPLICATIONS "CLOUD READY" 3/3

- Limiter autant que possible les dépendances à du matériel ou du logiciel spécifique qui pourrait ne pas fonctionner dans un cloud
- Tolérance aux pannes (*fault tolerance*) intégrée
- Ne pas stocker les données en local, mais plutôt :
  - Base de données
  - Stockage objet
- Utiliser des outils standards de journalisation

CÔTÉ SYSTÈME

ADOPTER UNE PHILOSOPHIE DEVOPS

- Infrastructure as Code
- Scale out plutôt que scale up (horizontalement plutôt que verticalement)
- HA niveau application plutôt qu'infrastructure
- Automatisation, automatisation, automatisation
- Tests

MONITORING

- Prendre en compte le cycle de vie des instances
- Monitorer le service plus que le serveur

BACKUP

- Être capable de recréer ses instances (et le reste de son infrastructure)
- Données (applicatives, logs) : block, objet

UTILISER DES IMAGES CLOUD

Une image cloud c'est :

- Une image disque contenant un OS déjà installé
- Une image qui peut être instanciée en n machines sans erreur
- Un OS sachant parler à l'API de metadata du cloud (cloud-init)
- Détails : http://docs.openstack.org/image-guide/openstack-images.html
- La plupart des distributions fournissent aujourd'hui des images cloud.

CIRROS

- Cirros est l'image cloud par excellence
- OS minimaliste
- Contient cloud-init

https://launchpad.net/cirros

CLOUD-INIT

- Cloud-init est un moyen de tirer parti de l'API de metadata, et notamment des user data
- L'outil est intégré par défaut dans la plupart des images cloud
- À partir des user data, cloud-init effectue les opérations de personnalisation de l'instance
- cloud-config est un format possible de user data

EXEMPLE CLOUD-CONFIG

```
#cloud-config
mounts:
 - [ xvdc, /var/www ]
packages:
 - apache2
 - htop
```

COMMENT GÉRER SES IMAGES ?

- Utilisation d'images génériques et personnalisation à l'instanciation
- Création d'images intermédiaires et/ou totalement personnalisées : *Golden images*
  - libguestfs, virt-builder, virt-sysprep
  - diskimage-builder (TripleO)
  - Packer
  - solution "maison"

CONFIGURER ET ORCHESTRER SES INSTANCES

- Outils de gestion de configuration (les mêmes qui permettent de déployer OpenStack)
- Juju

ARCHITECTURES CLOUD-READY

CONCEVOIR UNE APPLICATION POUR LE CLOUD

12-FACTOR

“The Twelve-Factor App” http://12factor.net/

- Écrit par Heroku
- Suivre (tout) le code dans un VCS
- Configuration

MODULAIRE

- Multiples composants de taille raisonnable
- Philosophie Unix
- Couplage faible et interface documentée

PASSAGE À L'ÉCHELLE

- Vertical vs Horizontal
- Scale up vs Scale out
- Plusieurs petites instances plutôt qu'une grosse instance

STATEFUL VS STATELESS

- Beaucoup de stateful dans les applications legacy
- Nécessite de partager l'information d'état lorsque plusieurs workers
- Le stateless élimine cette contrainte

TOLÉRANCE AUX PANNES

- L’infrastructure n’est pas hautement disponible
- L’API d’infrastructure est hautement disponible
- L’application doit anticiper et réagir aux pannes

STOCKAGE DES DONNÉES

- Base de données relationnelles
- Base de données NoSQL
- Stockage bloc
- Stockage objet
- Stockage éphémère
- Cache, temporaire

DESIGN TENETS D'OPENSTACK (EXEMPLE) 1/2

1. Scalability and elasticity are our main goals
2. Any feature that limits our main goals must be optional
3. Everything should be asynchronous. If you can't do something asynchronously, see #2
4. All required components must be horizontally scalable

DESIGN TENETS D'OPENSTACK (EXEMPLE) 2/2

5. Always use shared nothing architecture (SN) or sharding. If you can't Share nothing/shard, see #2
6. Distribute everything. Especially logic. Move logic to where state naturally exists.
7. Accept eventual consistency and use it where it is appropriate.
8. Test everything. We require tests with submitted code. (We will help you if you need it)

https://wiki.openstack.org/wiki/BasicDesignTenets

CONCEVOIR UNE INFRASTRUCTURE POUR LE CLOUD

AUTOMATISATION

- Automatiser la gestion de l’infrastructure : indispensable
- Création des ressources
- Configuration des ressources

INFRASTRUCTURE AS CODE

- Travailler comme un développeur
- Décrire son infrastructure sous forme de code (Heat, Ansible)
- Suivre les changements dans un VCS (git)
- Utiliser des outils de tests

BESOIN D'ORCHESTRATION

- Manager tous les types de ressources par un point d'entrée
- Description de l'infrastructure dans un fichier (*template*)
- Heat (intégré à OpenStack), Terraform

TESTS ET INTÉGRATION CONTINUE

- Style de code
- Validation de la syntaxe
- Voire plus si possible

AUTOSCALING GROUP

- Groupe d’instances similaires
- Nombre variable d’instances
- Scaling automatique en fonction de métriques

L'ISOLATION

- Niveau control plane : Tenant (projet)
- Niveau réseau : L2, L3, security groups

MULTI-TENANT

- Notion générale : un déploiement du logiciel permet de multiples utilisations
- Un cloud OpenStack permet aux utilisateurs de travailler dans des environnements isolés
- Les instances, réseaux, images, etc. sont associés à un tenant
- Certaines ressources peuvent être partagées entre tenants (exemple :image publique)
- On peut aussi parler de “projet”

LES INSTANCES

- Éphémère
- Pets vs Cattle
- Basé sur une *image*
- API de metadata

L'API DE METADATA

- API à destination des instances
- Standard de fait initié par AWS
- Accessible depuis l'instance sur http://169.254.169.254/
- Expose des informations relatives à l'instance
- Expose un champ libre dit "userdata"

RÉSEAU

- Fixed IP
- Multiples interfaces réseaux
- Floating IPs : pool, allocate, associate

FLOATING IPS

- *IP flottantes*
- Surcharge des “Fixed IPs”
- Non portée par l’instance
- Souvent une IP “publique”
- Une fois allouée à un tenant, l’IP est réservée
- Elle est ensuite associable et désassociable à loisir

SECURITY GROUPS

- Équivalent à un firewall devant chaque instance
- Une instance peut être associée à un ou plusieurs groupes de sécurité
- Gestion des accès en entrée et sortie
- Règles par protocole (TCP/UDP/ICMP) et par port
- Cible une adresse IP, un réseau ou un autre groupe de sécurité

FLAVORS

- *Gabarit*
- Équivalent des “instance types” d’AWS
- Définit un modèle d’instance en termes de CPU, RAM, disque (racine), disque éphémère
- Un disque de taille nul équivaut à prendre la taille de l’image de base
- Le disque éphémère a, comme le disque racine, l’avantage d’être souvent local donc rapide

KEYPAIRS

- *Paires de clé*
- Clés SSH publiques/privés
- Le cloud a connaissance de la clé publique
- La clé publique est injectée dans les instances

MONITORING

Monitoring

- Prendre en compte le cycle de vie des instances : DOWN != ALERT
- Monitorer le service plus que le serveur

BACKUP

Backuper, quoi ?

- Être capable de recréer ses instances (et le reste de son infrastructure)
- Données (applicatives, logs) : block, objet

UN EXEMPLE : L'ÉQUIPE OPENSTACK-INFRA

- Équipe projet en charge de l'infrastructure de développement d'OpenStack
- Travaille comme les équipes de dev d'OpenStack et utilise les mêmes outils
- Infrastructure as code
- Infrastructure ouverte : code "open source"
- Utilise du cloud (hybride)

CONCLUSION

POUR CONCLURE

- Le cloud révolutionne l'IT
- OpenStack est le projet libre phare sur la partie IaaS
- Déployer OpenStack n'est pas une mince affaire
- L'utilisation d'un cloud IaaS implique des changements de pratique
- Les métiers d'architecture logicielle et infra évoluent