

Formation OpenStack



Arnaud Morin

Orange Labs




Concernant ces supports de cours


Auteurs initiaux :

- Adrien Cunin <adrien.cunin@osones.com> 
- Pierre Freund <pierre.freund@osones.com> 

Modifiés et adaptés par :


- Arnaud Morin <arnaud1.morin@orange.com> 

Licence

- Copyright ©2014 Osones
- Creative Commons BY-SA 4.0 

<https://creativecommons.org/licenses/by-sa/4.0/deed.fr>

Sources

- <https://github.com/Osones/OpenStack-Formations/>
- <https://github.com/arnaudmorinol/OpenStack-Formations/> 

Objectifs de la formation

- Virtualisation
- Cloud
- Focus sur OpenStack

Objectifs de la formation : Virtualisation

- Comprendre les principes de la virtualisation et son intérêt
- Connaitre le vocabulaire inhérent à la virtualisation
- Avoir une vue d'ensemble sur les solutions existantes de virtualisation

Objectifs de la formation : Cloud

- Comprendre les principes du cloud et son intérêt
- Connaitre le vocabulaire inhérent au cloud
- Avoir une vue d'ensemble sur les solutions existantes en cloud public et privé
- Posséder les clés pour tirer partie au mieux de l'IaaS
- Pouvoir déterminer ce qui est compatible avec la philosophie cloud ou pas
- Adapter ses méthodes d'administration système à un environnement cloud



Objectifs de la formation : OpenStack

- Connaitre le fonctionnement du projet OpenStack et ses possibilités
- Comprendre le fonctionnement de chacun des composants d'OpenStack
- Pouvoir faire les bons choix de configuration
- Savoir déployer manuellement un cloud OpenStack pour fournir de l'IaaS
- Connaitre les bonnes pratiques de déploiement d'OpenStack
- Être capable de déterminer l'origine d'une erreur dans OpenStack
- Savoir réagir face à un bug

Plan

1 Le Cloud : vue d'ensemble

Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
- PaaS : Platform as a Service
- IaaS : Infrastructure as a Service
- Stockage : block, objet, SDS
- Orchestrer les ressources de son IaaS
- APIs : quel rôle ?

Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
 - PaaS : Platform as a Service
 - IaaS : Infrastructure as a Service
 - Stockage : block, objet, SDS
 - Orchestrer les ressources de son IaaS
 - APIs : quel rôle ?

Le cloud, c'est large !

- Stockage
- Calcul
- Virtualisation++
- Abstraction du matériel
- Accès normalisé par des APIs
- Service et facturation à la demande
- Flexibilité, élasticité

WaaS : Whatever as a Service

- Principalement

 - IaaS Infrastructure as a Service

 - PaaS Platform as a Service

 - SaaS Software as a Service

- Mais aussi :

 - ▶ Database as a Service
 - ▶ Network as a Service
 - ▶ Firewall as a Service
 - ▶ Load balancing as a Service
 - ▶ Whatever as a Service

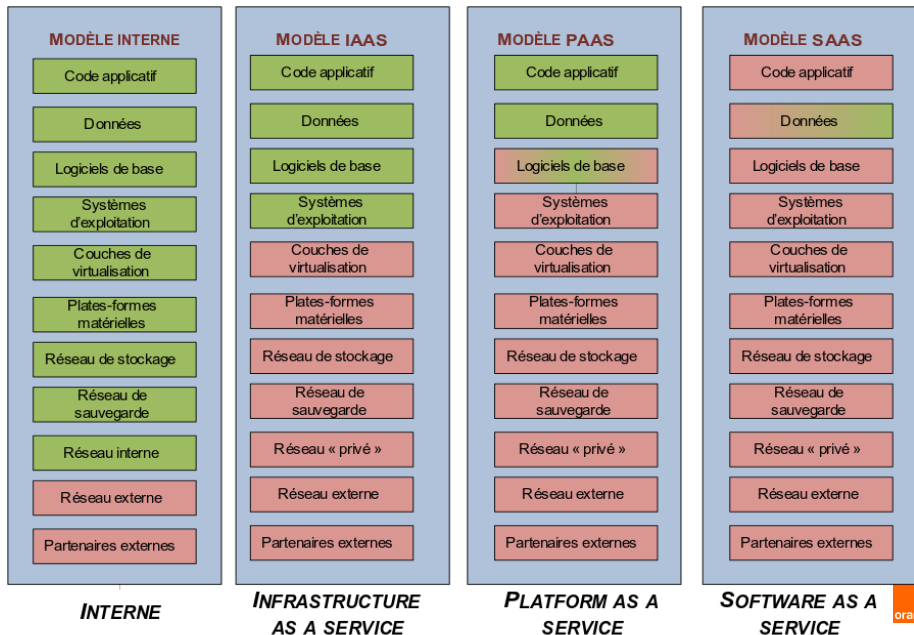
Cloud public ou cloud privé ?

Public fourni par un hébergeur à des clients (AWS, Rackspace Cloud, Azure, etc.)

Privé plateforme et ressources internes

Hybride utilisation de ressources publiques au sein d'un cloud privé

Les modèles cloud en un schéma



Pourquoi du cloud ? Côté business

- Baisse des coûts par la mutualisation des ressources
- Utilisation uniquement des ressources qui sont nécessaires
- À grande échelle, garantie de service

Pourquoi du cloud ? Côté technique

- Abstraction des couches plus basses
- On peut tout programmer à son gré (tout est API !)
- Permet la mise en place d'architectures scalables (en théorie)

Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
- **PaaS : Platform as a Service**
- IaaS : Infrastructure as a Service
- Stockage : block, objet, SDS
- Orchestrer les ressources de son IaaS
- APIs : quel rôle ?

PaaS : les principes

- Fourniture d'une plateforme de développement
- Fourniture d'une plateforme de déploiement
- Pour un langage / un framework
- Principalement utilisé par des développeurs

Exemples de PaaS publics

- Amazon OpsWork / Elastic Beanstalk
- Google App Engine
- Heroku
- OVH (Lamp Stack et Ruby Stack pour développeurs)
<http://www.ovh.com/fr/vps/systeme-exploitation.xml>

Solutions de PaaS privé

- Cloud Foundry
- OpenShift (Red Hat)
- Solum

Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
- PaaS : Platform as a Service
- **IaaS : Infrastructure as a Service**
- Stockage : block, objet, SDS
- Orchestrer les ressources de son IaaS
- APIs : quel rôle ?

Amazon Web Services (AWS) et les autres

- Service (cloud public) : [AWS](#)
 - ▶ Pionnier du genre (dès 2002)
 - ▶ Elastic Compute Cloud ([EC2](#))
 - ▶ Elastic Block Storage ([EBS](#))
 - ▶ Simple Storage Service ([S3](#))
- Logiciels libres permettant le déploiement d'un cloud privé :
 - ▶ Eucalyptus
 - ▶ CloudStack
 - ▶ OpenNebula
 - ▶ **OpenStack**

Les clouds publics concurrents d'AWS

- Google Compute Engine
- Rackspace
- HP Cloud
- Microsoft Azure
- En France
 - ▶ **Cloudwatt**
 - ▶ **Numergy**
 - ▶ **Outscale**

Virtualisation dans le cloud

- Le cloud IaaS repose souvent sur la virtualisation
- Ressources compute ← virtualisation
- Virtualisation complète : KVM, Xen
- Virtualisation containers : OpenVZ, LXC, Docker

Notions et vocabulaire IaaS

- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité
- Paires de clés
- API REST
- API de metadata et user data
- Cloud-init



Notions et vocabulaire IaaS

- L'instance est par définition éphémère
- Elle doit être utilisée comme ressource de calcul
- Une image se personnalise lors de son instantiation grâce à l'API de metadata
- Séparer les données des instances
- Choix du type de stockage : éphémère, volume, objet

Virtualisation \neq IaaS

Regardons l'interface web d'Amazon

Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
- PaaS : Platform as a Service
- IaaS : Infrastructure as a Service
- **Stockage : block, objet, SDS**
- Orchestrer les ressources de son IaaS
- APIs : quel rôle ?

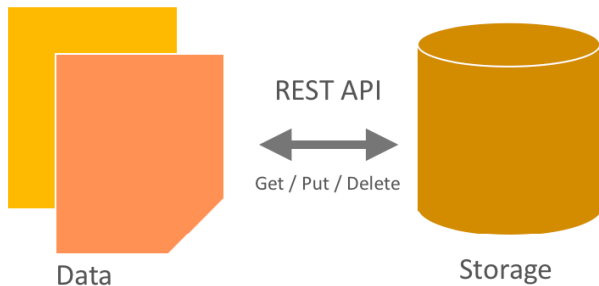
Stockage block

- Accès à des raw devices type `/dev/vdb`
- Possibilité d'utiliser n'importe quel système de fichiers
- Compatible avec toutes les applications legacy
- Technos : iSCSI, Fiber Channel, FCoE

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 partie du stockage objet

Stockage objet : schéma



SDS : Software Defined Storage

- Utilisation de commodity hardware
- Pas de RAID matériel
- Le logiciel est responsable de garantir les données



Deux solutions : OpenStack Swift et Ceph

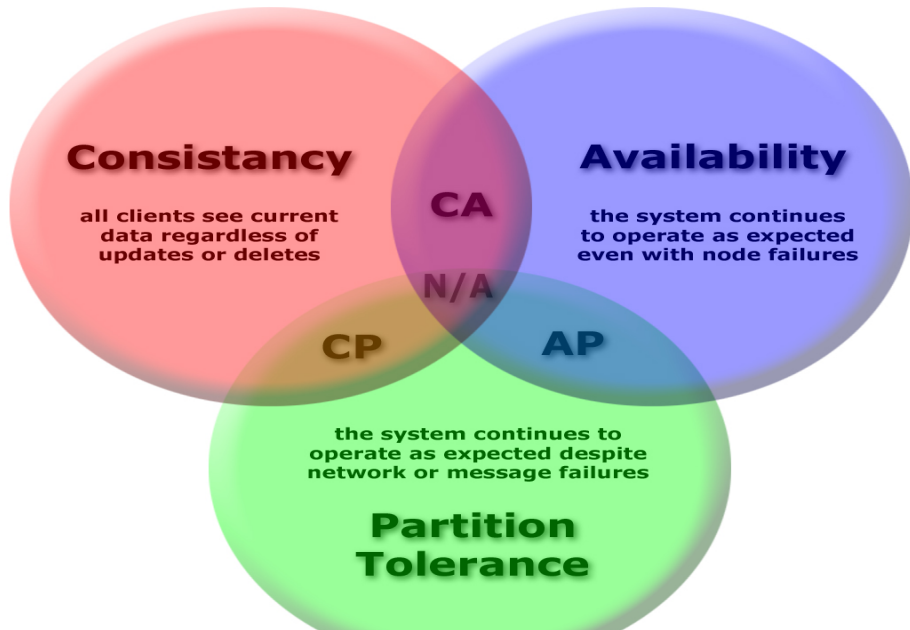
- Swift fait partie du projet OpenStack et fournit du stockage objet
- Ceph fournit du stockage objet, block et FS
- Les deux sont du SDS
- Théorème CAP : on en choisit deux

Consistency Tous les noeuds du système voient exactement les mêmes données au même moment

Availability Toutes les requêtes doivent recevoir une réponse

Partition Tolerance En cas de morcellement en sous-réseaux, chacun doit pouvoir fonctionner de manière autonome

Théorème CAP



Swift

- Swift est un projet OpenStack
- Le projet est né chez Rackspace avant la création d'OpenStack
- Swift est en production chez Rackspace depuis lors
- C'est le composant le plus mature d'OpenStack
- Théorème CAP : AP

- Projet totalement parallèle à OpenStack
- Supporté par une entreprise (Inktank) récemment rachetée par Red Hat
- Fournit d'abord du stockage objet
- L'accès aux données se fait via RADOS :
 - ▶ Accès direct depuis une application avec librados
 - ▶ Accès via une API REST grâce à radosgw
- La couche RBD permet d'accéder aux données en mode block (volumes)
- CephFS permet un accès par un système de fichiers POSIX
- Théorème CAP : CP

Plan

- 1 Le Cloud : vue d'ensemble
 - Le Cloud : les concepts
 - PaaS : Platform as a Service
 - IaaS : Infrastructure as a Service
 - Stockage : block, objet, SDS
 - Orchestrer les ressources de son IaaS
 - APIs : quel rôle ?

Pourquoi orchestrer

- Définir toute une infrastructure dans un seul fichier texte
- Être en capacité d'instancier une infrastructure entière en un appel API
- Adapter ses ressources en fonction de ses besoins en temps réel (autoscaling)

Exemples d'orchestration

- Amazon CloudFormation
- OpenStack Heat
- Microsoft Azure Automation

Exemples d'orchestration

```
resources:
  my_instance:
    type: AWS::EC2::Instance
    properties:
      KeyName: { get_param: KeyName }
      ImageId: { get_param: ImageId }
      InstanceType: { get_param: InstanceType }
```


Plan

1 Le Cloud : vue d'ensemble

- Le Cloud : les concepts
- PaaS : Platform as a Service
- IaaS : Infrastructure as a Service
- Stockage : block, objet, SDS
- Orchestrer les ressources de son IaaS
- APIs : quel rôle ?

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

Exemple concret

```
GET /v2.0/networks/network_id
{
  "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
  }
}
```

