



# Formation OpenStack

Formation OpenStack

2014-08-11

Arnaud Morin

Orange Labs



## Formation OpenStack

2014-08-11

## Concernant ces supports de cours

## Concernant ces supports de cours

## Auteurs initiaux :

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

## 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

## Formation OpenStack

### └ Introduction

#### └ Objectifs de la formation

2014-08-11

- Virtualisation
- Cloud
- Focus sur OpenStack

# Objectifs de la formation : Virtualisation

## Formation OpenStack

### └ Introduction

#### └ Objectifs de la formation : Virtualisation

2014-08-11

- 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

## Formation OpenStack

### └ Introduction

#### └ Objectifs de la formation : Cloud

2014-08-11



- 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

## Formation OpenStack

### └ Introduction

#### └ Objectifs de la formation : OpenStack

2014-08-11

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

- Virtualisation
- Le Cloud : vue d'ensemble
- OpenStack : projet, logiciel et utilisation
- Déployer OpenStack
- Tirer partie de l'IaaS

## Formation OpenStack

### └ Introduction

### └ Plan

2014-08-11

# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation
- 4 Déployer OpenStack
- 5 Tirer partie de l'IaaS



## 1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

2014-08-11



Selon Wikipedia :  
« La virtualisation consiste à faire fonctionner un ou plusieurs systèmes d'exploitation comme un simple logiciel, sur un ou plusieurs ordinateurs (serveurs), au lieu de ne pouvoir en installer qu'un seul par machine. »

# Définition

Formation OpenStack  
└ Virtualisation  
    └ Définition

2014-08-11

Selon Wikipedia :

« La virtualisation consiste à faire fonctionner un ou plusieurs systèmes d'exploitation comme un simple logiciel, sur un ou plusieurs ordinateurs (serveurs), au lieu de ne pouvoir en installer qu'un seul par machine. »



# Plan

## 1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

Formation OpenStack

Virtualisation

Histoire de la virtualisation

Plan

2014-08-11

Plan

Virtualisation

● Histoire de la virtualisation

⋮

Le Cloud : vue d'ensemble

● OpenStack : projet, logiciel et utilisation

● Déployer OpenStack

● Tirer partie de l'IaaS



# Historique - 1

## Formation OpenStack

### Virtualisation

- Histoire de la virtualisation
  - Historique - 1

2014-08-11

### 1946 Premiers ordinateurs < Turing-complet > (ex : ENIAC)

Une machine de Turing est un modèle abstrait du fonctionnement des ordinateurs, inventé en 1936 par Alan Turing  
un système Turing-complet est un système formel ayant une puissance de calcul au moins équivalente à celle des machine de Turing. Dans un tel système, il est possible de programmer n'importe quelle machine de Turing, mais également tout ce que l'on peut programmer dans une machine de Turing

Hôte = simple arbitre entre les systèmes invités

First layer virtualization is provided by the Processor Resource and System Manager (PR/SM, hypervisor type-1) to deploy Logical Partitions (LPARs)  
A System z hypervisor called z/VM can also be booted as the second layer virtualization in LPARs to create as many Virtual Machines (VM) as there are resources assigned to the LPARs to support them



1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)  
1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.

## Historique - 1

### Formation OpenStack

- └ Virtualisation
  - └ Histoire de la virtualisation
    - └ Historique - 1

2014-08-11

1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)

1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.

Une machine de Turing est un modèle abstrait du fonctionnement des ordinateurs, inventé en 1936 par Alan Turing

un système Turing-complet est un système formel ayant une puissance de calcul au moins équivalente à celle des machines de Turing. Dans un tel système, il est possible de programmer n'importe quelle machine de Turing, mais également tout ce que l'on peut programmer dans une machine de Turing

Hôte = simple arbitre entre les systèmes invités

First layer virtualization is provided by the Processor Resource and System Manager (PR/SM, hypervisor type-1) to deploy Logical Partitions (LPARs)

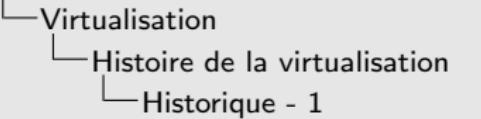
A System z hypervisor called z/VM can also be booted as the second layer virtualization in LPARs to create as many Virtual Machines (VM) as there are resources assigned to the LPARs to support them



1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)  
 1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.  
 196x Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X

# Historique - 1

## Formation OpenStack



2014-08-11

- 1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)
- 1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.
- 196x Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X

Une machine de Turing est un modèle abstrait du fonctionnement des ordinateurs, inventé en 1936 par Alan Turing  
 un système Turing-complet est un système formel ayant une puissance de calcul au moins équivalente à celle des machines de Turing. Dans un tel système, il est possible de programmer n'importe quelle machine de Turing, mais également tout ce que l'on peut programmer dans une machine de Turing

Hôte = simple arbitre entre les systèmes invités  
 First layer virtualization is provided by the Processor Resource and System Manager (PR/SM, hypervisor type-1) to deploy Logical Partitions (LPARs)  
 A System z hypervisor called z/VM can also be booted as the second layer virtualization in LPARs to create as many Virtual Machines (VM) as there are resources assigned to the LPARs to support them

1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)  
 1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.  
 196x Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X  
 1972 IBM Mainframe Virtual Machine Facility/370 : premier système de « full virtualisation » !

## Formation OpenStack

- └ Virtualisation
  - └ Histoire de la virtualisation
    - └ Historique - 1

2014-08-11

Une machine de Turing est un modèle abstrait du fonctionnement des ordinateurs, inventé en 1936 par Alan Turing  
 un système Turing-complet est un système formel ayant une puissance de calcul au moins équivalente à celle des machines de Turing. Dans un tel système, il est possible de programmer n'importe quelle machine de Turing, mais également tout ce que l'on peut programmer dans une machine de Turing

Hôte = simple arbitre entre les systèmes invités

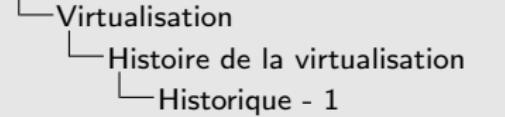
First layer virtualization is provided by the Processor Resource and System Manager (PR/SM, hypervisor type-1) to deploy Logical Partitions (LPARs)  
 A System z hypervisor called z/VM can also be booted as the second layer virtualization in LPARs to create as many Virtual Machines (VM) as there are resources assigned to the LPARs to support them



# Historique - 1

- 1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)
- 1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.
- 196x Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X
- 1972 IBM Mainframe Virtual Machine Facility/370 : **premier système de « full virtualisation » !**
- 1988 IBM sort son Hyperviseur PR/SM de type 1

## Formation OpenStack



2014-08-11

Historique - 1

- 1946 Premiers ordinateurs « Turing-complet » (ex : ENIAC)
- 1958 Ordinateurs multitâches (Gamma 60 de Bull) : faire tourner plusieurs programmes en même temps, concept proche de la virtualisation.
- 1960 Time-sharing systems, premières notions de Virtual Machine avec le projet IBM M44/44X
- 1972 IBM Mainframe Virtual Machine Facility/370 : **premier système de « full virtualisation » !**
- 1988 IBM sort son Hyperviseur PR/SM de type 1

Une machine de Turing est un modèle abstrait du fonctionnement des ordinateurs, inventé en 1936 par Alan Turing  
un système Turing-complet est un système formel ayant une puissance de calcul au moins équivalente à celle des machines de Turing. Dans un tel système, il est possible de programmer n'importe quelle machine de Turing, mais également tout ce que l'on peut programmer dans une machine de Turing

Hôte = simple arbitre entre les systèmes invités  
First layer virtualization is provided by the Processor Resource and System Manager (PR/SM, hypervisor type-1) to deploy Logical Partitions (LPARs)  
A System z hypervisor called z/VM can also be booted as the second layer virtualization in LPARs to create as many Virtual Machines (VM) as there are resources assigned to the LPARs to support them



# Historique - 2

## Formation OpenStack

### Virtualisation

- Histoire de la virtualisation
  - Historique - 2

2014-08-11

1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)

NUMA : Non Uniform Memory Access

Intel VT-x permet aux hyperviseurs de se positionner dans le Ring -1 et de controller les VM entry/exit

QEMU fait par un Français : Fabrice Bellard



1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)  
1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)

## Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)

### Formation OpenStack

- Virtualisation
  - Histoire de la virtualisation
    - Historique - 2

2014-08-11

NUMA : Non Uniform Memory Access

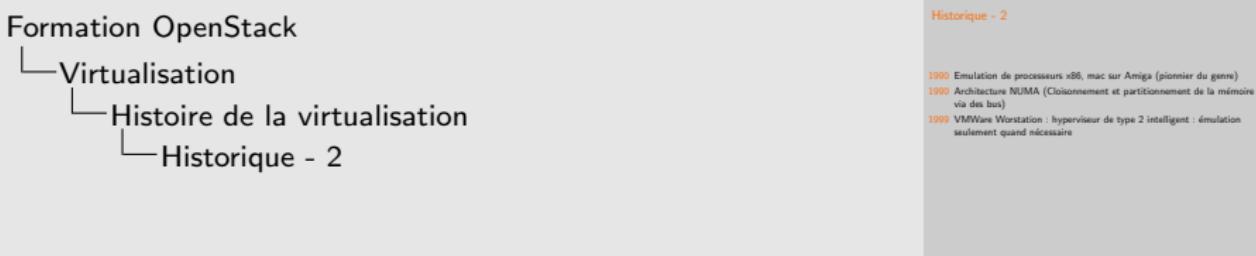
Intel VT-x permet aux hyperviseurs de se positionner dans le Ring -1 et de controller les VM entry/exit

QEMU fait par un Français : Fabrice Bellard



# Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)
- 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire



NUMA : Non Uniform Memory Access

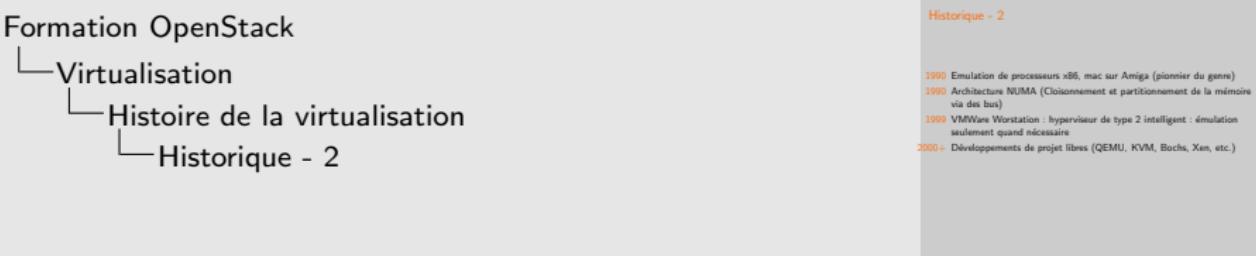
Intel VT-x permet aux hyperviseurs de se positionner dans le Ring -1 et de controller les VM entry/exit

QEMU fait par un Français : Fabrice Bellard



# Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)
- 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire
- 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)



NUMA : Non Uniform Memory Access  
Intel VT-x permet aux hyperviseurs de se positionner dans le Ring -1 et de controller les VM entry/exit  
QEMU fait par un Français : Fabrice Bellard



1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)  
 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)  
 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire  
 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)  
 2004 Intel VT-x : Les VM ont directement accès au CPU. Les hyperviseurs ne font plus d'émulation mais contrôlent qui a accès au CPU

## Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)
- 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire
- 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)
- 2004 Intel VT-x : Les VM ont directement accès au CPU. Les hyperviseurs ne font plus d'émulation mais contrôlent qui a accès au CPU

### Formation OpenStack

- Virtualisation
  - Histoire de la virtualisation
    - Historique - 2

2014-08-11



1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)  
 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)  
 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire  
 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)  
 2004 Intel VT-x : Les VM ont directement accès au CPU. Les hyperviseurs ne font plus d'émulation mais contrôlent qui a accès au CPU

## Historique - 2

- 1990 Emulation de processeurs x86, mac sur Amiga (pionnier du genre)
- 1990 Architecture NUMA (Cloisonnement et partitionnement de la mémoire via des bus)
- 1999 VMWare Worstation : hyperviseur de type 2 intelligent : émulation seulement quand nécessaire
- 2000+ Développements de projet libres (QEMU, KVM, Bochs, Xen, etc.)
- 2004 Intel VT-x : Les VM ont directement accès au CPU. Les hyperviseurs ne font plus d'émulation mais contrôlent qui a accès au CPU

### Formation OpenStack

- Virtualisation
  - Histoire de la virtualisation
    - Historique - 2

2014-08-11



2014-08-11

# Plan

## 1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS



# Principes généraux de Popek et Goldberg

Popek et Goldberg sont deux chercheurs qui ont introduits des conditions pour qu'un système supporte la virtualisation :

**équivalence** fonctionnement identique dans une VM comme sur une machine physique

**efficacité** une part majoritaire d'instructions doit être exécutée directement sans intervention de l'hyperviseur

**contrôle** l'hyperviseur garde le contrôle des ressources et les partage entre les VM

Formation OpenStack

Virtualisation

Principes généraux

Principes généraux de Popek et Goldberg

2014-08-11

Principes généraux de Popek et Goldberg

Popek et Goldberg sont deux chercheurs qui ont introduits des conditions pour qu'un système supporte la virtualisation :

- équivalence fonctionnement identique dans une VM comme sur une machine physique
- efficacité une part majoritaire d'instructions doit être exécutée directement sans intervention de l'hyperviseur
- contrôle l'hyperviseur garde le contrôle des ressources et les partage entre les VM



# Principes généraux de Popek et Goldberg

Popek et Goldberg sont deux chercheurs qui ont introduits des conditions pour qu'un système supporte la virtualisation :

**équivalence** fonctionnement identique dans une VM comme sur une machine physique

**efficacité** une part majoritaire d'instructions doit être exécutée directement sans intervention de l'hyperviseur

**contrôle** l'hyperviseur garde le contrôle des ressources et les partage entre les VM

**Ces principes datent de 1974 !**

Seulement à partir de 2004 (avec Intel VT) qu'ils sont respectés dans l'architecture x86.

Formation OpenStack

Virtualisation

Principes généraux

Principes généraux de Popek et Goldberg

2014-08-11

Principes généraux de Popek et Goldberg

Popek et Goldberg sont deux chercheurs qui ont introduits des conditions pour qu'un système supporte la virtualisation :  
**équivalence** fonctionnement identique dans une VM comme sur une machine physique  
**efficacité** une part majoritaire d'instructions doit être exécutée directement sans intervention de l'hyperviseur  
**contrôle** l'hyperviseur garde le contrôle des ressources et les partage entre les VM  
**Ces principes datent de 1974 !**  
Seulement à partir de 2004 (avec Intel VT) qu'ils sont respectés dans l'architecture x86.



# Intérêt de la virtualisation

Formation OpenStack

└ Virtualisation

└ Principes généraux

└ Intérêt de la virtualisation

2014-08-11

- Trois avantages principaux
- Sécurité
  - Coût
  - Criticité et performances

## Trois avantages principaux

- Sécurité
- Coût
- Criticité et performances



# Intérêt de la virtualisation - Sécurité

## Formation OpenStack

### Virtualisation

#### Principes généraux

##### Intérêt de la virtualisation - Sécurité

2014-08-11

- Sécurité

- ▶ Isolation et cloisonnement

- ★ Ignorance de la présence d'autres environnements
    - ★ Utilisation des protocoles conventionnels

- ▶ Etudes de sécurité

- ★ Contrôle et étude d'environnements infectés
    - ★ Répétition de scénarios

Chercheurs qui analysent un virus par exemple

- Sécurité
  - ▶ Isolation et cloisonnement
    - Ignorance de la présence d'autres environnements
    - Utilisation des protocoles conventionnels
  - ▶ Etudes de sécurité
    - Contrôle et étude d'environnements infectés
    - Répétition de scénarios



# Intérêt de la virtualisation - Coût

- Coût

- ▶ Mutualisation de ressources physiques

- ★ Sous utilisation actuelle des serveurs
    - ★ Coût de l'énergie électrique
    - ★ Coût de l'espace en centre de données
    - ★ Coût opérationnel

## Formation OpenStack

### Virtualisation

#### Principes généraux

##### Intérêt de la virtualisation - Coût

2014-08-11

- Coût
  - Mutualisation de ressources physiques
    - Sous utilisation actuelle des serveurs
    - Coût de l'énergie électrique
    - Coût de l'espace en centre de données
    - Coût opérationnel

Taux moyen d'utilisation serveur physique : 10%

Taux moyen d'utilisation serveur virtuel : 35% : besoin de 3 fois moins de serveurs

Telle machine utilisera 1 CPU, telle autre 1 autre CPU sur un même serveur, je peux faire tourner une VM le jour, une autre la nuit



# Intérêt de la virtualisation - Coût

- Coût

- Mutualisation de ressources physiques

- ★ Sous utilisation actuelle des serveurs
    - ★ Coût de l'énergie électrique
    - ★ Coût de l'espace en centre de données
    - ★ Coût opérationnel

- Meilleure gestion des ressources physiques

- ★ Allocation exclusive
    - ★ Allocation temporelle

## Formation OpenStack

### Virtualisation

#### Principes généraux

##### Intérêt de la virtualisation - Coût

2014-08-11

- Coût
  - Mutualisation de ressources physiques
    - Sous utilisation actuelle des serveurs
    - Coût de l'énergie électrique
    - Coût de l'espace en centre de données
    - Coût opérationnel
  - Meilleure gestion des ressources physiques
    - Allocation exclusive
    - Allocation temporelle

Taux moyen d'utilisation serveur physique : 10%

Taux moyen d'utilisation serveur virtuel : 35% : besoin de 3 fois moins de serveurs

Telle machine utilisera 1 CPU, telle autre 1 autre CPU sur un même serveur, je peux faire tourner une VM le jour, une autre la nuit



- Criticité et performances

- ▶ Possibilité de mettre en pause et de copier un environnement logiciel complet
  - ★ Sauvegarde
  - ★ Clonage
- ▶ Migration d'environnements logiciels
  - ★ Transfert d'un environnement logiciel vers une autre machine physique
- ▶ Allocation dynamique de ressources
  - ★ Flexibilité de l'offre
  - ★ Adaptabilité en cas de montée en charge

- Criticité et performances

- ▶ Possibilité de mettre en pause et de copier un environnement logiciel complet
  - ★ Sauvegarde
  - ★ Clonage
- ▶ Migration d'environnements logiciels
  - ★ Transfert d'un environnement logiciel vers une autre machine physique
- ▶ Allocation dynamique de ressources
  - ★ Flexibilité de l'offre
  - ★ Adaptabilité en cas de montée en charge

# Plan

## 1 Virtualisation

- Histoire de la virtualisation
- Principes généraux
- Comprendre la virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

### Formation OpenStack

#### Virtualisation

##### Comprendre la virtualisation

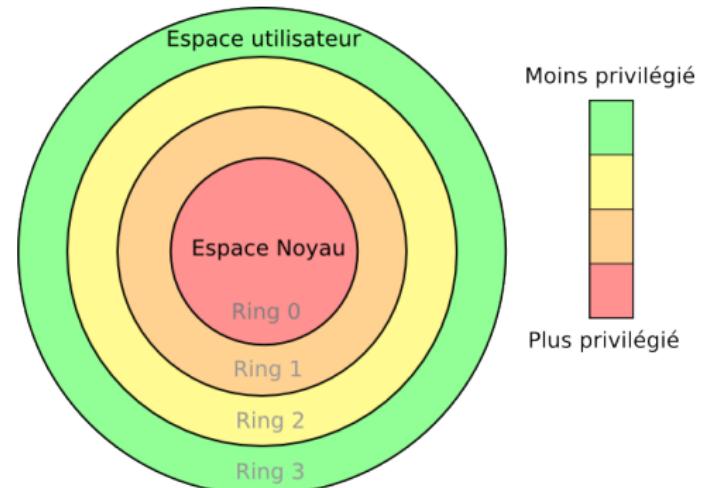
###### Plan

2014-08-11



# Les anneaux de protection

- 4 niveaux de priviléges dans l'architecture x86
- Ring 0 : Espace Noyau
- Ring 1 et Ring 2 : généralement pas utilisés
- Ring 3 : Espace Utilisateur



Crédit image : « Hertzprung » sur wikipedia, modifié pour le cours



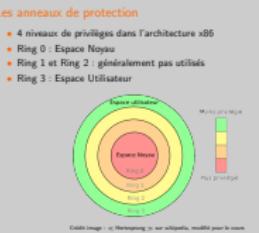
## Formation OpenStack

### Virtualisation

#### Comprendre la virtualisation

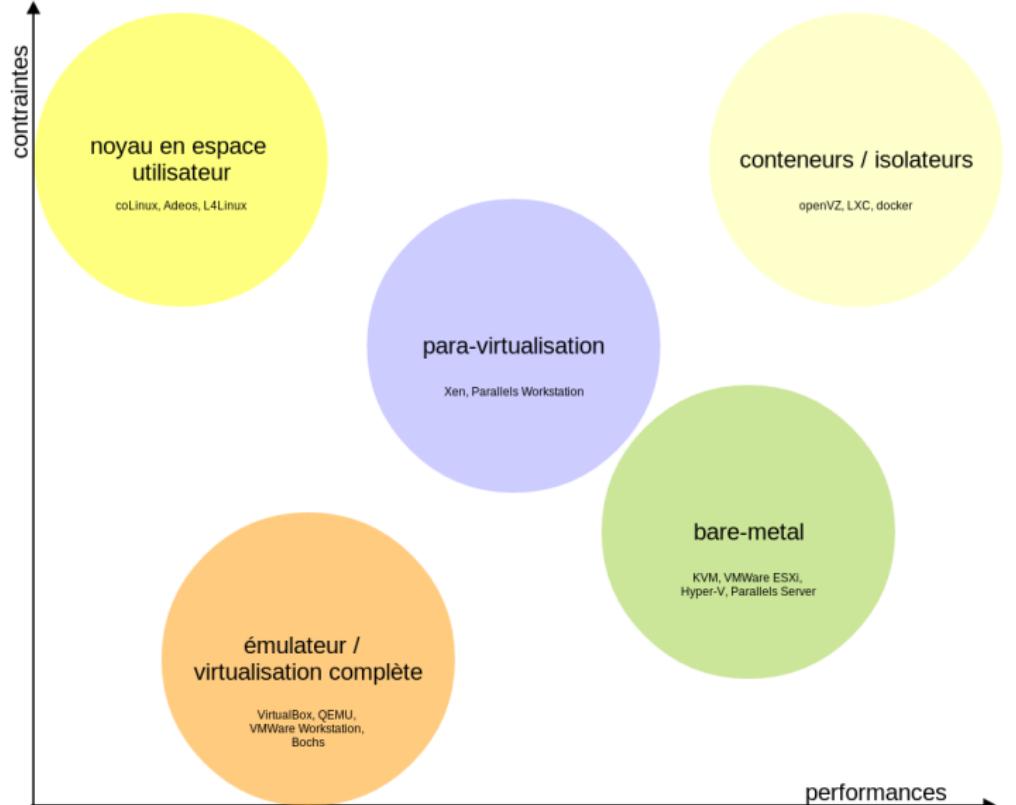
##### Les anneaux de protection

2014-08-11



La transition d'un mode à un autre est assurée par l'instruction en langage assembleur SYSENTER.

# Les modèles de virtualisation en un schéma



Formation OpenStack

Virtualisation

Comprendre la virtualisation

Les modèles de virtualisation en un schéma

2014-08-11

Les modèles de virtualisation en un schéma



# Les conteneurs / isolateurs

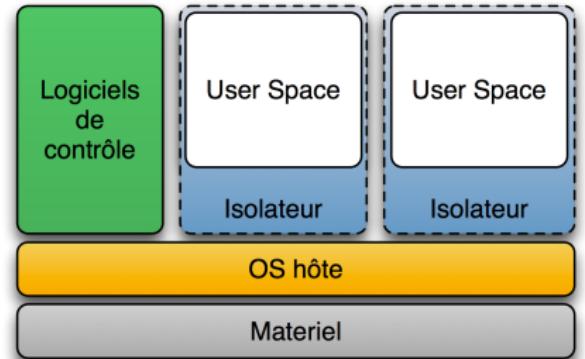
Principe : isoler l'exécution d'applications dans un contexte (aussi appelé zone d'exécution)

- Pros :

- ▶ Très performant car peu d'overhead
- ▶ Permet de faire tourner la même application en mode multi-instance  
(ex : serveur web)

- Cons :

- ▶ Même noyau pour toutes les applications
- ▶ Seulement linux
- ▶ Pas vraiment de la virtualisation



## Formation OpenStack

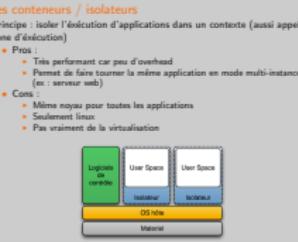
### Virtualisation

#### Comprendre la virtualisation

- Les conteneurs / isolateurs

2014-08-11

Un peu comme un chroot



# Noyau en espace utilisateur

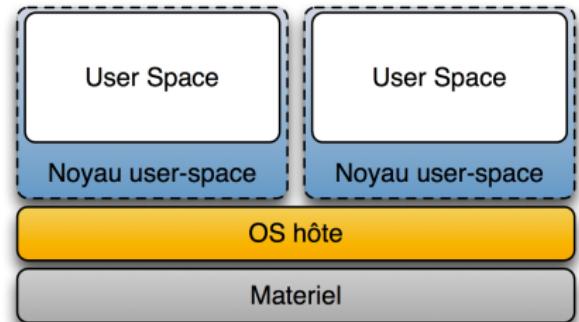
Principe : faire tourner un noyau Linux dans l'espace utilisateur

- Pros :

- ▶ Utile pour tester un noyau linux ou faire du développement de noyau

- Cons :

- ▶ Pas très performant (empilement de deux noyaux)
- ▶ Pas très sécurisé (pas d'isolation entre les noyaux)
- ▶ Nécessite une modification du noyau invité (possible avec linux seulement)



## Formation OpenStack

### Virtualisation

#### Comprendre la virtualisation

##### Noyau en espace utilisateur

2014-08-11

- Noyau en espace utilisateur
- Principe : faire tourner un noyau Linux dans l'espace utilisateur
- Pros :
    - ▶ Utile pour tester un noyau linux ou faire du développement de noyau
  - Cons :
    - ▶ Pas très performant (empilement de deux noyaux)
    - ▶ Pas très sécurisé (pas d'isolation entre les noyaux)
    - ▶ Nécessite une modification du noyau invité (possible avec linux seulement)



# Émulation / Full virtualisation (Hyperviseur de type 2)

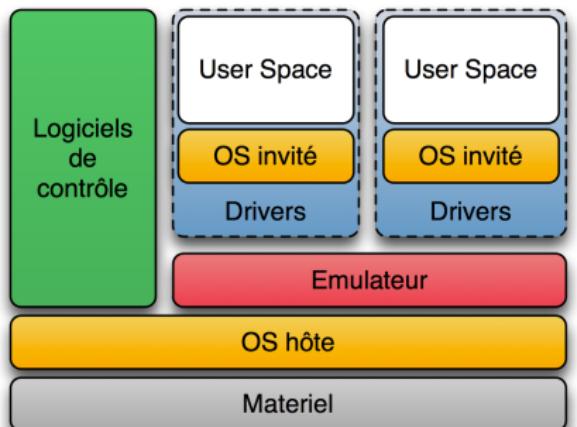
Principe : la machine hôte émule le matériel pour la machine invitée

- Pros :

- ▶ Bonne isolation entre les OS invités
- ▶ Cohabitation d'architecture CPU et OS hétérogènes

- Cons :

- ▶ Pas très performant (émulation provoque beaucoup d'overhead)



## Formation OpenStack

### Virtualisation

#### Comprendre la virtualisation

##### Émulation / Full virtualisation (Hyperviseur de type 2)

2014-08-11

émulateur NES, PS utilisent ce principe là

## Emulation / Full virtualisation (Hyperviseur de type 2)

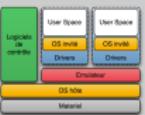
Principe : la machine hôte émule le matériel pour la machine invitée

- Pros :

- ▶ Bonne isolation entre les OS invités
- ▶ Cohabitation d'architecture CPU et OS hétérogènes

- Cons :

- ▶ Pas très performant (émulation provoque beaucoup d'overhead)



# Para-virtualisation

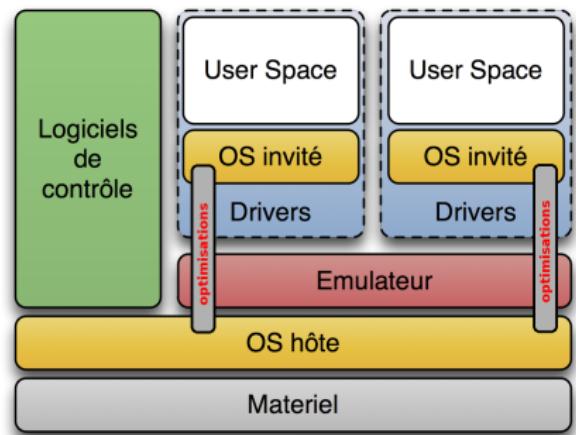
Principe : évolution de l'émulation par modification des OS invités

- Pros :

- ▶ Bonne isolation entre les OS invités
- ▶ Cohabitation d'architecture CPU et OS hétérogènes
- ▶ Performance correctes

- Cons :

- ▶ Nécessite la modification du noyau de l'OS invité (possible sur Linux seulement)



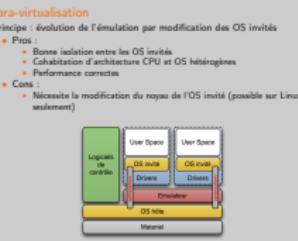
## Formation OpenStack

### Virtualisation

#### Comprendre la virtualisation

##### Para-virtualisation

2014-08-11



# Bare metal (Hyperviseur de type 1)

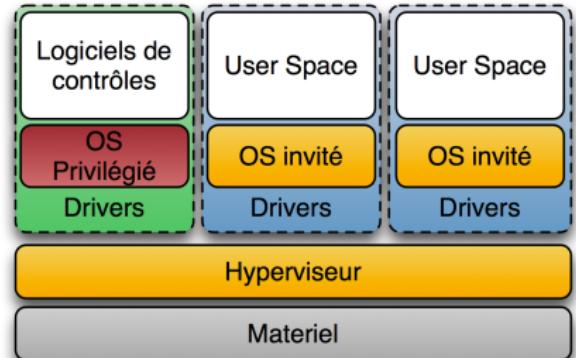
Principe : noyau système léger qui partage l'accès aux ressources matérielles avec les OS invités

- Pros :

- ▶ Bonne isolation entre les OS invités
- ▶ Cohabitation d'architecture CPU et OS hétérogènes
- ▶ Très performant (couche d'abstraction minimale)

- Cons :

- ▶ Nécessite la virtualisation matérielle (Intel VT-x ou AMD-V)



## Formation OpenStack

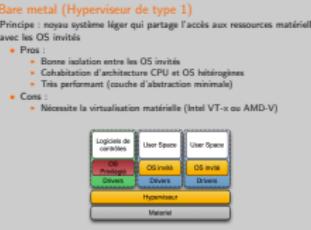
### Virtualisation

#### Comprendre la virtualisation

##### Bare metal (Hyperviseur de type 1)

2014-08-11

l'hyperviseur de type 1 fait le lien entre les VM et le matériel, il se positionne souvent dans le ring -1, voir slide suivant



À partir de 2004, Intel et AMD ont ajouté à leurs processeurs des instructions CPU supplémentaires pour aider à la virtualisation :

- Intel VT-x, VT-c, VT-d
- AMD-V

« To assist virtualization, VT and Pacifica insert a new privilege level beneath Ring 0. Both add nine new machine code instructions that only work at "Ring -1," intended to be used by the hypervisor. »

## Formation OpenStack

## Virtualisation

## Comprendre la virtualisation

## Assistance matérielle

2014-08-11

## Assistance matérielle

À partir de 2004, Intel et AMD ont ajouté à leurs processeurs des instructions CPU supplémentaires pour aider à la virtualisation :

- Intel VT-x, VT-c, VT-d
- AMD-V

« To assist virtualization, VT and Pacifica insert a new privilege level beneath Ring 0. Both add nine new machine code instructions that only work at "Ring -1," intended to be used by the hypervisor. »



## Démo

Formation OpenStack

└ Virtualisation

└ Comprendre la virtualisation

└ Démo

2014-08-11

# Exemples de création de machines virtuelles



# Plan

1 Virtualisation

2 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 ?

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

5 Tirer partie de l'IaaS

## Formation OpenStack

Le Cloud : vue d'ensemble

Plan

2014-08-11



# Plan

## 1 Virtualisation

## 2 Le Cloud : vue d'ensemble

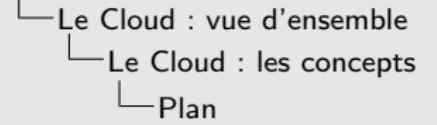
- Le Cloud : les concepts
  - PaaS : Platform as a Service
  - IaaS : Infrastructure as a Service
  - Stockage : block, objet, SDS
  - Orchestrer les ressources de son IaaS
  - APIs : quel rôle ?

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

### Formation OpenStack



2014-08-11



# Le cloud, c'est large !

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

• Stockage

- Stockage



# Le cloud, c'est large !

- Stockage
- Calcul

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

- Stockage
- Calcul



# Le cloud, c'est large !

- Stockage
- Calcul
- Virtualisation++

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
  - └ Le Cloud : les concepts
    - └ Le cloud, c'est large !

2014-08-11

- Stockage
- Calcul
- Virtualisation++



# Le cloud, c'est large !

- Stockage
- Calcul
- Virtualisation++
- Abstraction du matériel

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

- Stockage
- Calcul
- Virtualisation++
- Abstraction du matériel



# Le cloud, c'est large !

- Stockage
- Calcul
- Virtualisation++
- Abstraction du matériel
- Accès normalisé par des APIs

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

- Stockage
- Calcul
- Virtualisation++
- Abstraction du matériel
- Accès normalisé par des APIs



# Le cloud, c'est large !

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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

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



# Le cloud, c'est large !

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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

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



# Le cloud, c'est large !

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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

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



# Le cloud, c'est large !

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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Le cloud, c'est large !

2014-08-11

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



# WaaS : Whatever as a Service

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ WaaS : Whatever as a Service

• Principalement

2014-08-11

- Principalement



# WaaS : Whatever as a Service

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ WaaS : Whatever as a Service

2014-08-11

- Principalement

- IaaS Infrastructure as a Service



# WaaS : Whatever as a Service

- Principalement

IaaS Infrastructure as a Service

PaaS Platform as a Service

## Formation OpenStack

└ Le Cloud : vue d'ensemble

└ Le Cloud : les concepts

└ WaaS : Whatever as a Service

2014-08-11

- Principalement
  - IaaS Infrastructure as a Service
  - PaaS Platform as a Service



# WaaS : Whatever as a Service

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ WaaS : Whatever as a Service

2014-08-11

- Principalement
  - IaaS Infrastructure as a Service
  - PaaS Platform as a Service
  - SaaS Software as a Service

### • Principalement

IaaS Infrastructure as a Service

PaaS Platform as a Service

SaaS Software as a Service



# WaaS : Whatever as a Service

- Principalement

IaaS Infrastructure as a Service

PaaS Platform as a Service

SaaS Software as a Service

## Formation OpenStack

Le Cloud : vue d'ensemble

Le Cloud : les concepts

WaaS : Whatever as a Service

2014-08-11

- Principalement
  - IaaS Infrastructure as a Service
  - PaaS Platform as a Service
  - SaaS Software as a Service



2014-08-11

- Principalement
  - IaaS Infrastructure as a Service
  - PaaS Platform as a Service
  - SaaS Software as a Service
- Mais aussi :

# WaaS : Whatever as a Service

- Principalement

IaaS Infrastructure as a Service

PaaS Platform as a Service

SaaS Software as a Service

- Mais aussi :



# 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

## Formation OpenStack

Le Cloud : vue d'ensemble

Le Cloud : les concepts

WaaS : Whatever as a Service

2014-08-11

- 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



# 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

## Formation OpenStack

Le Cloud : vue d'ensemble

Le Cloud : les concepts

WaaS : Whatever as a Service

2014-08-11

- 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 ?

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
- └ Cloud public ou cloud privé ?

2014-08-11

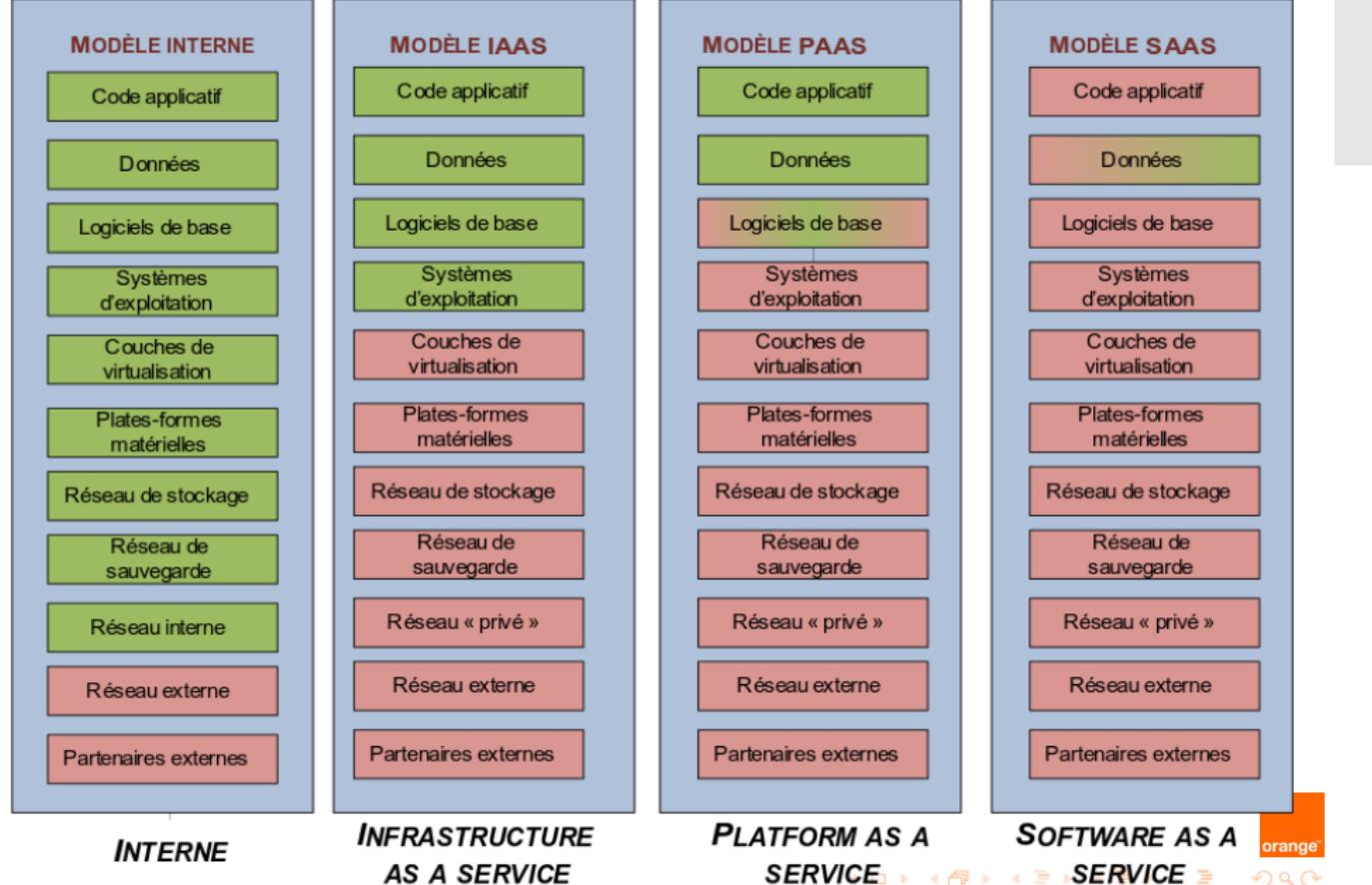
**Public** fourni par un hébergeur à des clients (AWS, Rackspace Cloud, Azure, etc.)

**Privé** plateforme et ressources internes

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



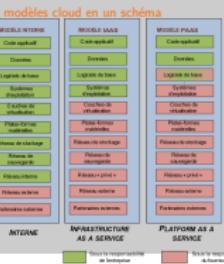
# Les modèles cloud en un schéma



## Formation OpenStack

- Le Cloud : vue d'ensemble
- Le Cloud : les concepts
- Les modèles cloud en un schéma

2014-08-11



# Pourquoi du cloud ? Côté business

## Formation OpenStack

Le Cloud : vue d'ensemble

Le Cloud : les concepts

Pourquoi du cloud ? Côté business

2014-08-11

- 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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Le Cloud : les concepts
  - └ Pourquoi du cloud ? Côté technique

2014-08-11

- 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 Virtualisation

## 2 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 ?

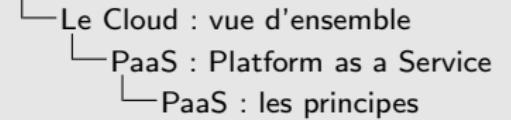
## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

# PaaS : les principes

## Formation OpenStack



2014-08-11

- 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

## Formation OpenStack

Le Cloud : vue d'ensemble

PaaS : Platform as a Service

Exemples de PaaS publics

2014-08-11

- Amazon OpsWork / Elastic Beanstalk
- Google App Engine
- Heroku



# Exemples de PaaS publics

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

## Formation OpenStack

- Le Cloud : vue d'ensemble
- PaaS : Platform as a Service
  - Exemples de PaaS publics

2014-08-11

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



# Solutions de PaaS privé

## Formation OpenStack

Le Cloud : vue d'ensemble

PaaS : Platform as a Service  
Solutions de PaaS privé

2014-08-11

- Cloud Foundry
- OpenShift (Red Hat)
- Solum

- Cloud Foundry
- OpenShift (Red Hat)
- Solum



# Plan

## 1 Virtualisation

## 2 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 ?

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

# Amazon Web Services (AWS) et les autres

## Formation OpenStack

2014-08-11

### Le Cloud : vue d'ensemble

#### IaaS : Infrastructure as a Service

##### 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**

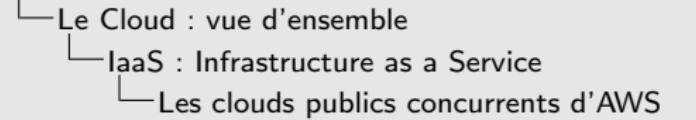
- Service (cloud public) : [AWS](#)
  - ▶ Pionnier du genre (dès 2002)
  - ▶ Elastic Compute Cloud ([EC2](#))
  - ▶ Elastic Block Storage ([EBS](#))
  - ▶ Simple Storage Service ([S3](#))
- Logiciels libres permettant le déploiement d'un cloud privé :
  - ▶ Eucalyptus
  - ▶ CloudStack
  - ▶ OpenNebula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



# Les clouds publics concurrents d'AWS

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

## Formation OpenStack



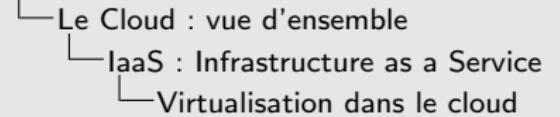
2014-08-11

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



# Virtualisation dans le cloud

## Formation OpenStack



2014-08-11

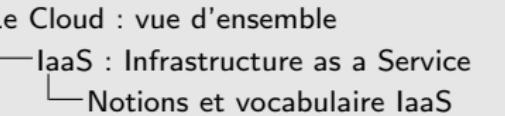
- Le cloud IaaS repose souvent sur la virtualisation
- Ressources compute ← virtualisation
- Virtualisation complète : KVM, Xen
- Virtualisation containers : OpenVZ, LXC, Docker

# Notions et vocabulaire IaaS

- Images
- Instances
- Types d'instance (flavors)
- Volumes

## Formation OpenStack

2014-08-11



- Images
- Instances
- Types d'instance (flavors)
- Volumes

**Image** : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

**Instance** : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Flavors** : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

**Volumes** : Disque dur associé à une instance.

**Stockage block** : Type de stockage utilisé pour les volumes.

**Stockage objet** : Type de stockage utilisé pour les images ou tout autre fichier.

**Floating IP** : IP publique NATée attribuée à une machine

**Groupe de sécurité** : Règles d'accès aux machines

**Paire de clés** : clef SSH

**API REST** : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

**API metadata et userdata** : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine)

**Cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud

## Notions et vocabulaire IaaS

- Images
  - Instances
  - Types d'instance (flavors)
  - Volumes
  - Stockage block
  - Stockage objet

## Information OpenStack

- Le Cloud : vue d'ensemble
  - IaaS : Infrastructure as a Service
    - Notions et vocabulaire IaaS

Image : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

Instance : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Emplacements** : Type d'instance prédéfini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

lumes : Disque dur associé à une instance.

ockage block : Type de stockage utilisé pour les volumes.

o<sup>ù</sup>ckage objet : Type de stockage utilis<sup>é</sup> pour les images ou tout autre fichier.

floating IP : IP publique NATée attribuée à une machine

#### **Groupe de sécurité : Règles**

API REST : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniformes, gestion de ressources.

**cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud.



# Notions et vocabulaire IaaS

- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques

## Formation OpenStack

2014-08-11

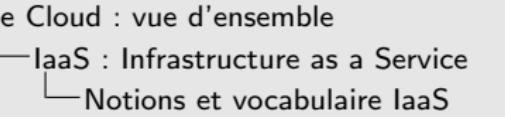


Image : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

Instance : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

Flavors : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

Volumes : Disque dur associé à une instance.

Stockage block : Type de stockage utilisé pour les volumes.

Stockage objet : Type de stockage utilisé pour les images ou tout autre fichier.

Floating IP : IP publique NATée attribuée à une machine

Groupe de sécurité : Règles d'accès aux machines

Paire de clés : clef SSH

API REST : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

API metadata et userdata : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine

Cloud-init : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud



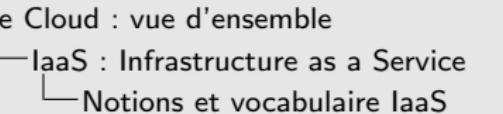
- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité

# Notions et vocabulaire IaaS

- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité

## Formation OpenStack

2014-08-11



**Image** : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

**Instance** : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Flavors** : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

**Volumes** : Disque dur associé à une instance.

**Stockage block** : Type de stockage utilisé pour les volumes.

**Stockage objet** : Type de stockage utilisé pour les images ou tout autre fichier.

**Floating IP** : IP publique NATée attribuée à une machine

**Groupe de sécurité** : Règles d'accès aux machines

**Paire de clés** : clef SSH

**API REST** : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

**API metadata et userdata** : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine)

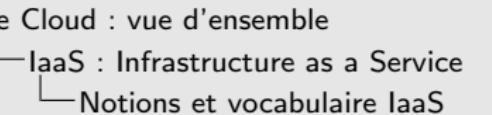
**Cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud

# Notions et vocabulaire IaaS

- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité
- Paires de clés

## Formation OpenStack

2014-08-11



**Image** : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

**Instance** : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Flavors** : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

**Volumes** : Disque dur associé à une instance.

**Stockage block** : Type de stockage utilisé pour les volumes.

**Stockage objet** : Type de stockage utilisé pour les images ou tout autre fichier.

**Floating IP** : IP publique NATée attribuée à une machine

**Groupe de sécurité** : Règles d'accès aux machines

**Paire de clés** : clef SSH

**API REST** : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

**API metadata et userdata** : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine)

**Cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud

- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité
- Paires de clés

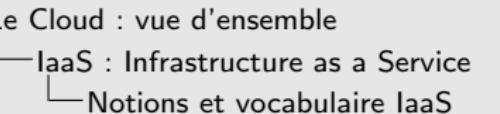
- Images
- Instances
- Types d'instance (flavors)
- Volumes
- Stockage block
- Stockage objet
- IP flottantes/élastiques
- Groupes de sécurité
- Paires de clés
- API REST

# 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

## Formation OpenStack

2014-08-11



**Image** : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

**Instance** : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Flavors** : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

**Volumes** : Disque dur associé à une instance.

**Stockage block** : Type de stockage utilisé pour les volumes.

**Stockage objet** : Type de stockage utilisé pour les images ou tout autre fichier.

**Floating IP** : IP publique NATée attribuée à une machine

**Groupe de sécurité** : Règles d'accès aux machines

**Paire de clés** : clef SSH

**API REST** : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

**API metadata et userdata** : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine)

**Cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud

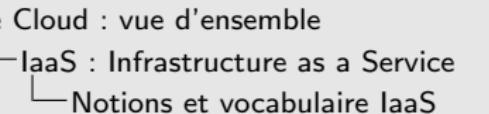


# Notions et vocabulaire IaaS

- 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

## Formation OpenStack

2014-08-11



- 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

**Image** : Photo d'une machine virtuelle contenant les informations nécessaires pour lancer une instance. Contient généralement un OS et/ou des applications. C'est un genre de template.

**Instance** : Une machine virtuelle lancée depuis une Image. On peut lancer plusieurs instances depuis la même image.

**Flavors** : Type d'instance prédefini, détermine par exemple CPU et RAM. Utile pour positionner l'instance au bon endroit dans le datacenter.

**Volumes** : Disque dur associé à une instance.

**Stockage block** : Type de stockage utilisé pour les volumes.

**Stockage objet** : Type de stockage utilisé pour les images ou tout autre fichier.

**Floating IP** : IP publique NATée attribuée à une machine

**Groupe de sécurité** : Règles d'accès aux machines

**Paire de clés** : clef SSH

**API REST** : Interface de programmation de type Representational State Transfer. En gros, architecture de communication client-serveur, avec interfaces uniforme, gestion de ressources,

**API metadata et userdata** : API permettant d'accéder à des données pour configurer et gérer une VM (ex : récupérer l'adresse d'ip de la machine)

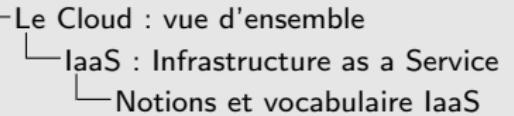
**Cloud-init** : Ensemble d'outils permettant de faciliter l'initialisation de machines virtuelles dans un cloud



# Notions et vocabulaire IaaS

## Formation OpenStack

2014-08-11



- 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

- 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



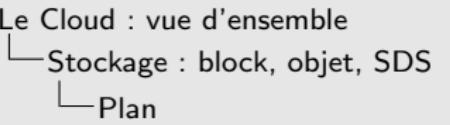
2014-08-11

# Virtualisation ≠ IaaS



2014-08-11

# Regardons l'interface web d'Amazon



2014-08-11

# Plan

## 1 Virtualisation

## 2 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 ?

## 3 OpenStack : projet, logiciel et utilisation

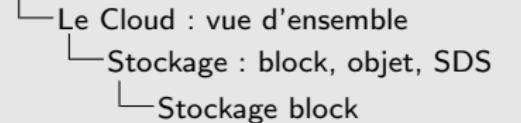
## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

- 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 block

## Formation OpenStack



2014-08-11

Samba, NFS, CIFS sont pas des block storage, mais des file storage

- 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

## Formation OpenStack

Le Cloud : vue d'ensemble

Stockage : block, objet, SDS

Stockage objet

2014-08-11

- 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

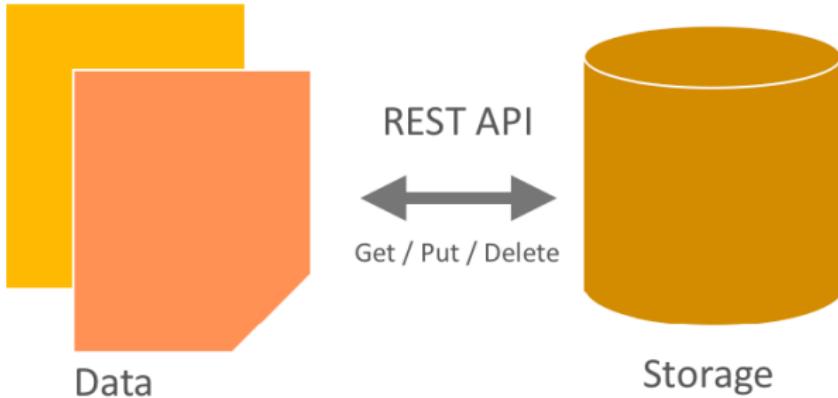
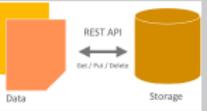


# Stockage objet : schéma

Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Stockage : block, objet, SDS
- └ Stockage objet : schéma

Stockage objet : schéma



2014-08-11

# SDS : Software Defined Storage

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Stockage : block, objet, SDS
- └ SDS : Software Defined Storage

2014-08-11

- 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

## Formation OpenStack

Le Cloud : vue d'ensemble

Stockage : block, objet, SDS

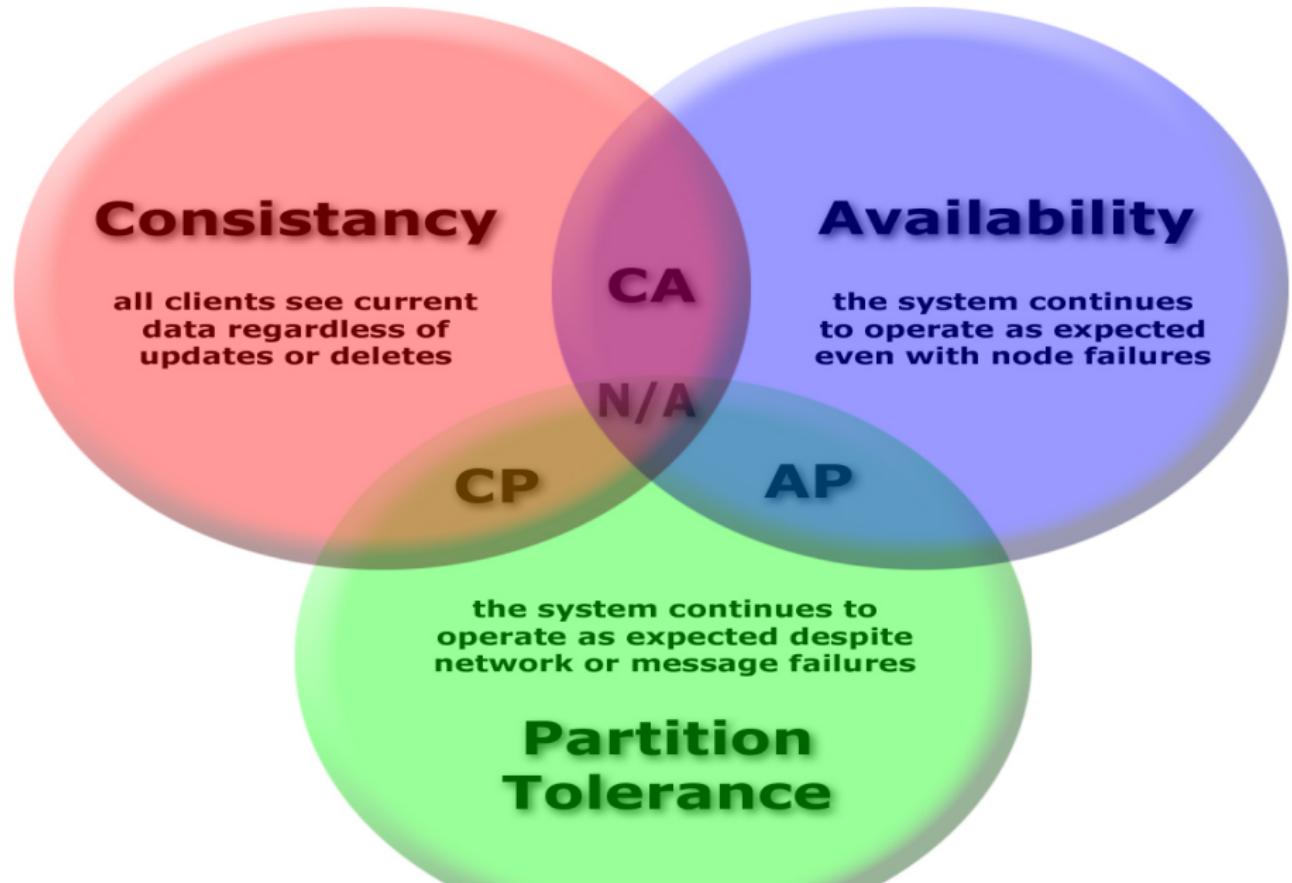
Deux solutions : OpenStack Swift et Ceph

2014-08-11

- 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



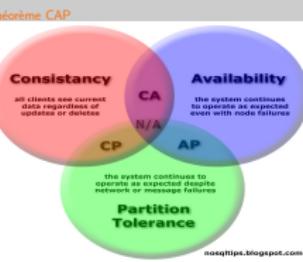
# Théorème CAP



## Formation OpenStack

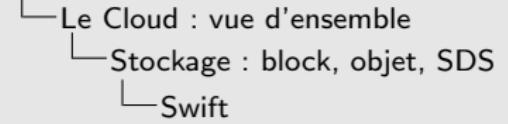
- Le Cloud : vue d'ensemble
- Stockage : block, objet, SDS
- Théorème CAP

2014-08-11



# Swift

## Formation OpenStack



2014-08-11

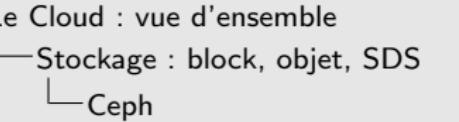
- Swift est un projet OpenStack
- Le projet est né chez Rackspace avant la création d'OpenStack
- Swift est en production chez Rackspace depuis lors
- C'est le composant le plus mature d'OpenStack
- Théorème CAP : AP

# Ceph

- 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

## Formation OpenStack

2014-08-11



- 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

RADOS : reliable autonomic distributed object store

RBD : RADOS Block Device

CephFS peut être monté par linux avec mount.ceph



2014-08-11

# Plan

## 1 Virtualisation

## 2 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 ?

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

# Pourquoi orchestrer

- Définir tout 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)

## Formation OpenStack

### Le Cloud : vue d'ensemble

#### Orchestrer les ressources de son IaaS

##### Pourquoi orchestrer

2014-08-11

- Définir tout 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

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Orchestrer les ressources de son IaaS
- └ Exemples d'orchestration

2014-08-11

- Amazon CloudFormation
- OpenStack Heat
- Microsoft Azure Automation

- Amazon CloudFormation
- OpenStack Heat
- Microsoft Azure Automation



# Exemples d'orchestration

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ Orchestrer les ressources de son IaaS
- └ Exemples d'orchestration

2014-08-11

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

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



2014-08-11

# Plan

## 1 Virtualisation

## 2 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 ?

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS

- 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  
  }
```

## Formation OpenStack

- └ Le Cloud : vue d'ensemble
- └ APIs : quel rôle ?
- └ Exemple concret

2014-08-11

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  
  }  
}
```



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

- Présentation du projet et du logiciel
- DevStack : faire tourner rapidement un OpenStack
- Utiliser OpenStack

4 Déployer OpenStack

5 Tirer partie de l'IaaS

## Formation OpenStack

└ OpenStack : projet, logiciel et utilisation

└ Plan

2014-08-11



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

- Présentation du projet et du logiciel
- DevStack : faire tourner rapidement un OpenStack
- Utiliser OpenStack

4 Déployer OpenStack

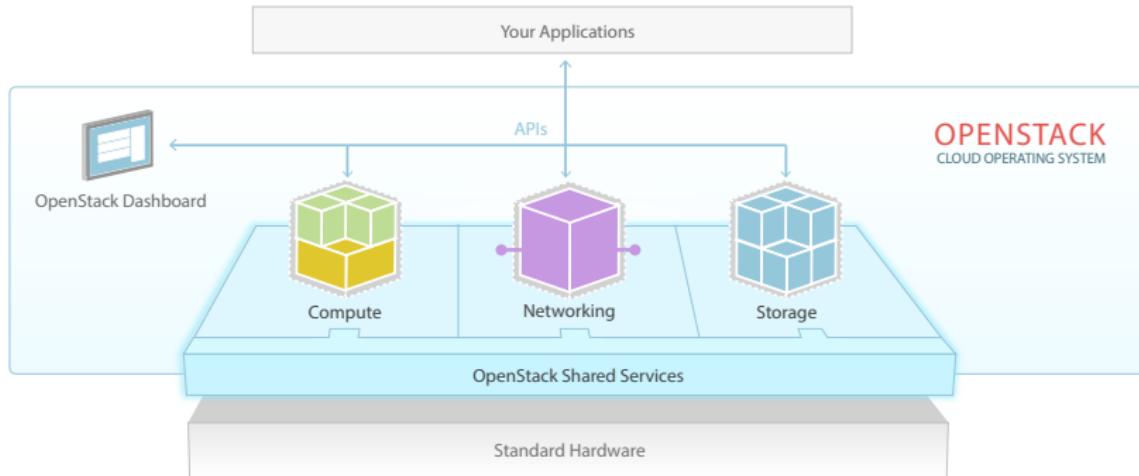
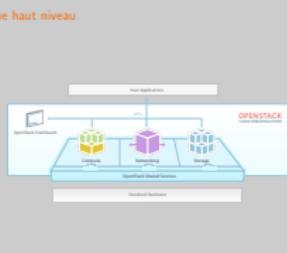
5 Tirer partie de l'IaaS



# Vue haut niveau

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Vue haut niveau



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

# Historique

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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Historique

2014-08-11



- Démarrage en 2010
- Objectif : le Cloud Operating System libre
- Fusion de deux projets de Rackspace (Storage) et de la NASA (Compute)
- Développé en Python et distribué sous licence Apache 2.0
- Les releases jusqu'à aujourd'hui :
  - ▶ 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)**

# Historique

- Démarrage en 2010
- Objectif : le Cloud Operating System libre
- Fusion de deux projets de Rackspace (Storage) et de la NASA (Compute)
- Développé en Python et distribué sous licence Apache 2.0
- Les releases jusqu'à aujourd'hui :
  - ▶ 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)**

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Historique

2014-08-11



- Démarrage en 2010
- Objectif : le Cloud Operating System libre
- Fusion de deux projets de Rackspace (Storage) et de la NASA (Compute)
- Développé en Python et distribué sous licence Apache 2.0
- Les releases jusqu'à aujourd'hui :
  - ▶ 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)**
  - ▶ Novembre 2014 : Juno

# Historique

- Démarrage en 2010
- Objectif : le Cloud Operating System libre
- Fusion de deux projets de Rackspace (Storage) et de la NASA (Compute)
- Développé en Python et distribué sous licence Apache 2.0
- Les releases jusqu'à aujourd'hui :
  - ▶ 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)**
  - ▶ Novembre 2014 : Juno

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Historique

2014-08-11



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull
- Mais aussi : Mirantis, StackOps, eNovance

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull
- Mais aussi : Mirantis, StackOps, eNovance



# Quelques soutiens/contributeurs ...

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull
- Mais aussi : Mirantis, StackOps, eNovance

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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Quelques soutiens/contributeurs ...

2014-08-11

- Rackspace et la NASA
- Canonical
- Red Hat
- Suse
- HP
- IBM
- Dell, Intel
- Cisco, Juniper
- NetApp, VMWare
- Yahoo, Bull
- Mais aussi : Mirantis, StackOps, eNovance

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



# ... et utilisateurs

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ ... et utilisateurs

2014-08-11

- Tous les contributeurs précédemment cités



# ... et utilisateurs

- Tous les contributeurs précédemment cités
- En France : **CloudWatt** et **Numergy**

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ ... et utilisateurs

2014-08-11

... et utilisateurs

- Tous les contributeurs précédemment cités
- En France : **CloudWatt** et **Numergy**



# ... et utilisateurs

- Tous les contributeurs précédemment cités
- En France : **CloudWatt** et **Numergy**
- Wikimedia
- CERN
- Paypal
- Comcast

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ ... et utilisateurs

2014-08-11

... et utilisateurs

- Tous les contributeurs précédemment cités
- En France : **CloudWatt** et **Numergy**
- Wikimedia
- CERN
- Paypal
- Comcast



# ... et utilisateurs

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

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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ ... et utilisateurs

2014-08-11

... et utilisateurs

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

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



- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift

# Les différents sous-projets

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11



# Les différents sous-projets

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder

# Les différents sous-projets

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron



# Les différents sous-projets

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron
- OpenStack Image Service - Glance

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron
- OpenStack Image Service - Glance



# 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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- OpenStack Compute - Nova
- OpenStack (Object) Storage - Swift
- OpenStack Block Storage - Cinder
- OpenStack Networking - Neutron
- OpenStack Image Service - Glance
- OpenStack Identity Service - Keystone



# 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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- 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



# 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 Metering - Ceilometer

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- 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 Metering - Ceilometer



# 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 Metering - Ceilometer
- OpenStack Orchestration - Heat

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11

- 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 Metering - Ceilometer
- OpenStack Orchestration - Heat



- 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 Metering - Ceilometer
- OpenStack Orchestration - Heat
- OpenStack Database Service - Trove

# 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 Metering - Ceilometer
- OpenStack Orchestration - Heat
- OpenStack Database Service - Trove

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets

2014-08-11



# Les différents sous-projets (2)

- Incubating (à jour Icehouse)
  - ▶ Bare metal (Ironic)
  - ▶ Queue service (Marconi)
  - ▶ Data processing (Sahara)

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets (2)

2014-08-11

- Incubating (à jour Icehouse)
  - Bare metal (Ironic)
  - Queue service (Marconi)
  - Data processing (Sahara)

# Les différents sous-projets (2)

- Incubating (à jour Icehouse)
  - ▶ Bare metal (Ironic)
  - ▶ Queue service (Marconi)
  - ▶ Data processing (Sahara)
- Intéressants
  - ▶ DNS service (Designate)
- Autres

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets (2)

2014-08-11

- Incubating (à jour Icehouse)
  - ▶ Bare metal (Ironic)
  - ▶ Queue service (Marconi)
  - ▶ Data processing (Sahara)
- Intéressants
  - ▶ DNS service (Designate)
- Autres



# Les différents sous-projets (2)

- Incubating (à jour Icehouse)
  - ▶ Bare metal (Ironic)
  - ▶ Queue service (Marconi)
  - ▶ Data processing (Sahara)
- Intéressants
  - ▶ DNS service (Designate)
- Autres
  - ▶ Oslo
  - ▶ rootwrap : wrapper pour les commandes root utilisé par les projets
  - ▶ TripleO
  - ▶ Tempest, Grenade
  - ▶ Les clients (python-\*client)

## Formation OpenStack

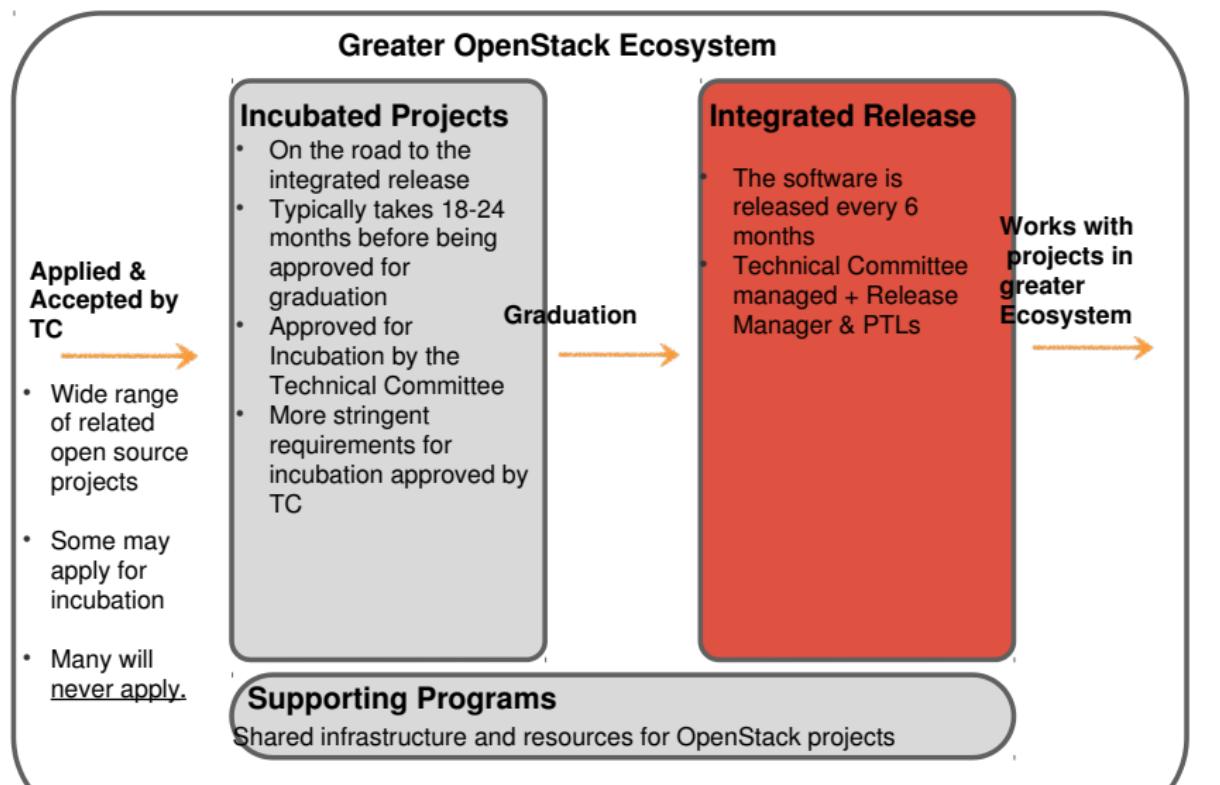
- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Les différents sous-projets (2)

2014-08-11

- Incubating (à jour Icehouse)
  - ▶ Bare metal (Ironic)
  - ▶ Queue service (Marconi)
  - ▶ Data processing (Sahara)
- Intéressants
  - ▶ DNS service (Designate)
- Autres
  - ▶ Oslo
  - ▶ rootwrap : wrapper pour les commandes root utilisé par les projets
  - ▶ TripleO
  - ▶ Tempest, Grenade
  - ▶ Les clients (python-\*client)



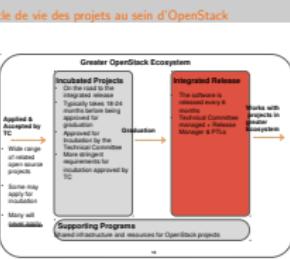
# Cycle de vie des projets au sein d'OpenStack



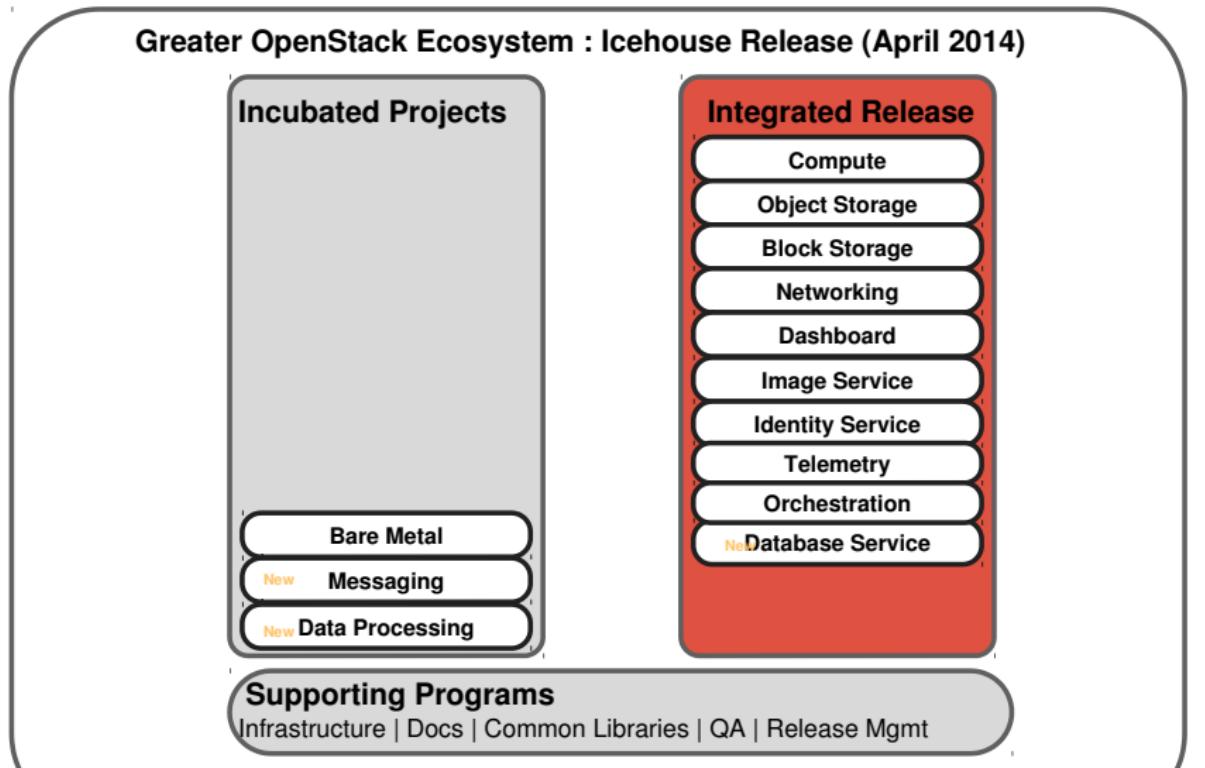
## Formation OpenStack

- OpenStack : projet, logiciel et utilisation
- Présentation du projet et du logiciel
  - Cycle de vie des projets au sein d'OpenStack

2014-08-11



# Les projets dans Icehouse

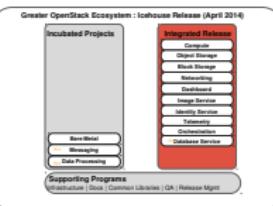


## Formation OpenStack

- OpenStack : projet, logiciel et utilisation
  - Présentation du projet et du logiciel
    - Les projets dans Icehouse

Supporting programs : Gerrit, Jenkins

Les projets dans Icehouse



# ICEHOUSE

THE NINTH OPENSTACK RELEASE

- Nova : rolling upgrades
- Nouvelles fonctionnalités disponibles dans Horizon
- Nouveaux drivers dans Neutron
- Nouveaux drivers dans Cinder
- Le nouveau format HOT est recommandé pour les templates Heat
- etc.
- En détails :

<https://wiki.openstack.org/wiki/ReleaseNotes/Icehouse>



## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Nouveautés de Icehouse

- Nova : rolling upgrades
- Nouvelles fonctionnalités disponibles dans Horizon
- Nouveaux drivers dans Neutron
- Nouveaux drivers dans Cinder
- Le nouveau format HOT est recommandé pour les templates Heat
- etc.
- En détails :  
<https://wiki.openstack.org/wiki/ReleaseNotes/Icehouse>

# Traduction

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Traduction

2014-08-11

- La question de la traduction est dorénavant prise en compte
- Seules certaines parties sont traduites, comme Horizon
- La traduction française est aujourd'hui une des plus avancées
- Utilisation de la plateforme Transifex



# Développement du projet

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

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet

Licence Apache 2Tout le monde peut proposer des blueprintsTout le monde peut proposer des patchs, reviewTout le monde peut participer

2014-08-11

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



# Développement du projet : en détails

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)



- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit

# Développement du projet : en détails

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit



# Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)



# Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)



# Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu



# Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu
- Développement hyper actif : 17000 commits dans Icehouse (+25%)

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu
- Développement hyper actif : 17000 commits dans Icehouse (+25%)



# Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu
- Développement hyper actif : 17000 commits dans Icehouse (+25%)
- Fin 2012, création d'une entité indépendante de gouvernance : la fondation OpenStack

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Développement du projet : en détails

2014-08-11

Développement du projet : en détails

- Ouvert à tous (individuels et entreprises)
- Cycle de développement de 6 mois débuté par un (design) summit
- Outils : Launchpad (blueprints, bugs) + Git + GitHub (code)
- Sur chaque commit : peer review (Gerrit) + intégration continue (exécution de différents tests par Jenkins)
- Plateforme de référence et modèle de développement : Ubuntu
- Développement hyper actif : 17000 commits dans Icehouse (+25%)
- Fin 2012, création d'une entité indépendante de gouvernance : la fondation OpenStack

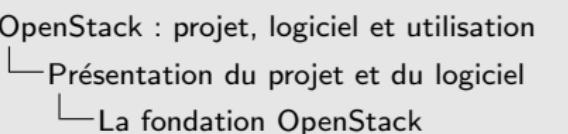


# La fondation OpenStack

- Entité de gouvernance principale 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 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)
- 850 organisations à travers le monde
- 9500 membres individuels dans 100 pays

## Formation OpenStack

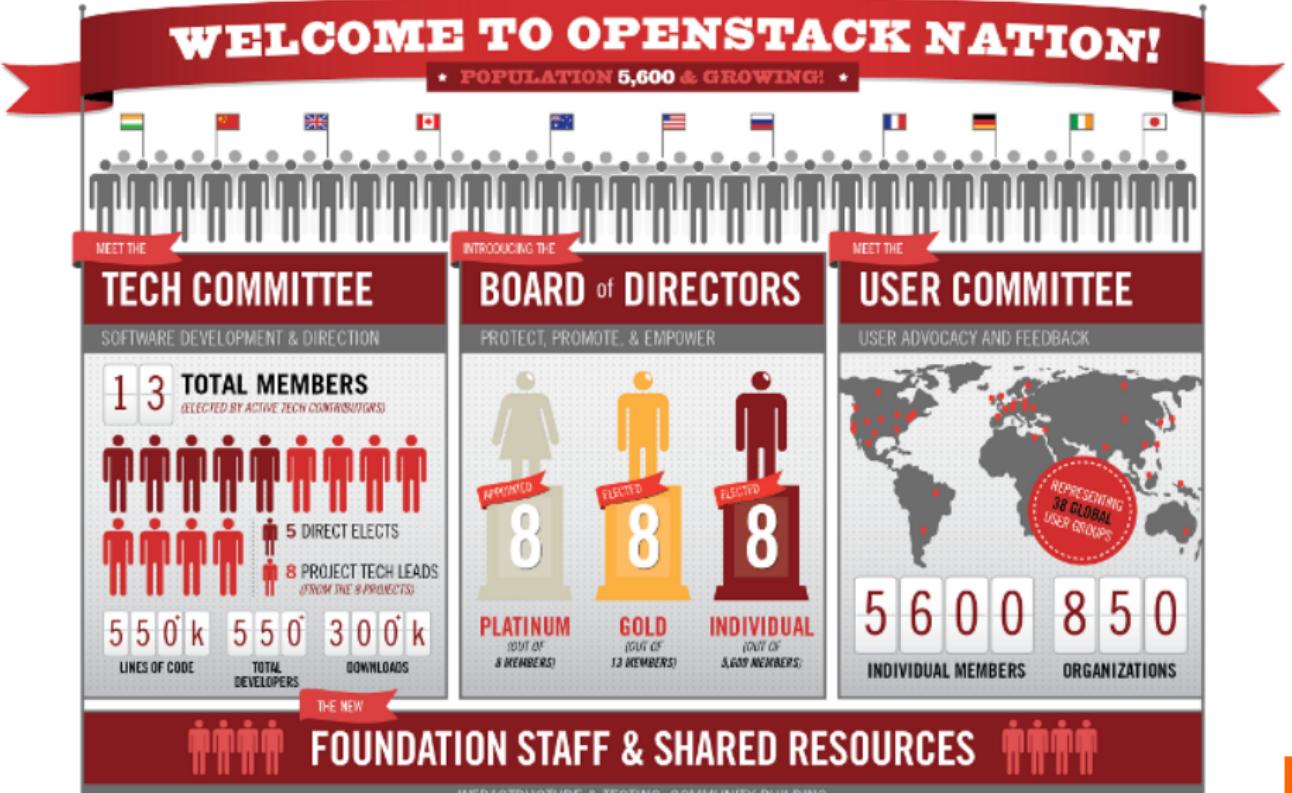
2014-08-11



- Entité de gouvernance principale 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 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)
- 850 organisations à travers le monde
- 9500 membres individuels dans 100 pays



# La fondation OpenStack



## Formation OpenStack

- OpenStack : projet, logiciel et utilisation
- Présentation du projet et du logiciel
  - La fondation OpenStack

2014-08-11



# Cycle de développement : 6 mois

- Le planning est publié, exemple : [https://wiki.openstack.org/wiki/Icehouse\\_Release\\_Schedule](https://wiki.openstack.org/wiki/Icehouse_Release_Schedule)
- Milestone releases
- Freezes : FeatureProposal, Feature, String
- RC releases
- Stable releases
- Ce modèle de cycle de développement a évolué depuis le début du projet
- Cas particulier de Swift

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Cycle de développement : 6 mois

2014-08-11

- Le planning est publié, exemple : [https://wiki.openstack.org/wiki/Icehouse\\_Release\\_Schedule](https://wiki.openstack.org/wiki/Icehouse_Release_Schedule)
- Milestone releases
- Freezes : FeatureProposal, Feature, String
- RC releases
- Stable releases
- Ce modèle de cycle de développement a évolué depuis le début du projet
- Cas particulier de Swift



# Où trouver des informations sur le développement d'OpenStack

## Formation OpenStack

OpenStack : projet, logiciel et utilisation

Présentation du projet et du logiciel

Où trouver des informations sur le développement d'OpenStack

2014-08-11

- Principalement sur le wiki : <https://wiki.openstack.org>
- Les blueprints et bugs sur Launchpad :  
<https://launchpad.net/openstack>
- Les patchs proposés et leurs reviews sont sur Gerrit :  
<https://review.openstack.org>
- Le code est disponible dans Git : <https://git.openstack.org>
- Les sources (tarballs) sont disponibles :  
<http://tarballs.openstack.org/>

- Principalement sur le wiki : <https://wiki.openstack.org>
- Les blueprints et bugs sur Launchpad :  
<https://launchpad.net/openstack>
- Les patchs proposés et leurs reviews sont sur Gerrit :  
<https://review.openstack.org>
- Le code est disponible dans Git : <https://git.openstack.org>
- Les sources (tarballs) sont disponibles :  
<http://tarballs.openstack.org/>

- Forge pour les nouveaux projets en lien avec OpenStack
- Ils bénéficient de l'infrastructure du projet OpenStack, mais la séparation reste claire
- Les projets démarrent dans Stackforge et peuvent ensuite rejoindre le projet OpenStack

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Stackforge

2014-08-11

- Forge pour les nouveaux projets en lien avec OpenStack
- Ils bénéficient de l'infrastructure du projet OpenStack, mais la séparation reste claire
- Les projets démarrent dans Stackforge et peuvent ensuite rejoindre le projet OpenStack



# OpenStack Summit

- Aux USA jusqu'en 2013
- Aujourd'hui : alternance USA et Asie/Europe
- Quelques centaines au début à 4500 de participants aujourd'hui
- En parallèle : conférence (utilisateurs, décideurs) et Design Summit (développeurs)
- Détermine le nom de la release : lieu/ville à proximité du Summit

## Formation OpenStack

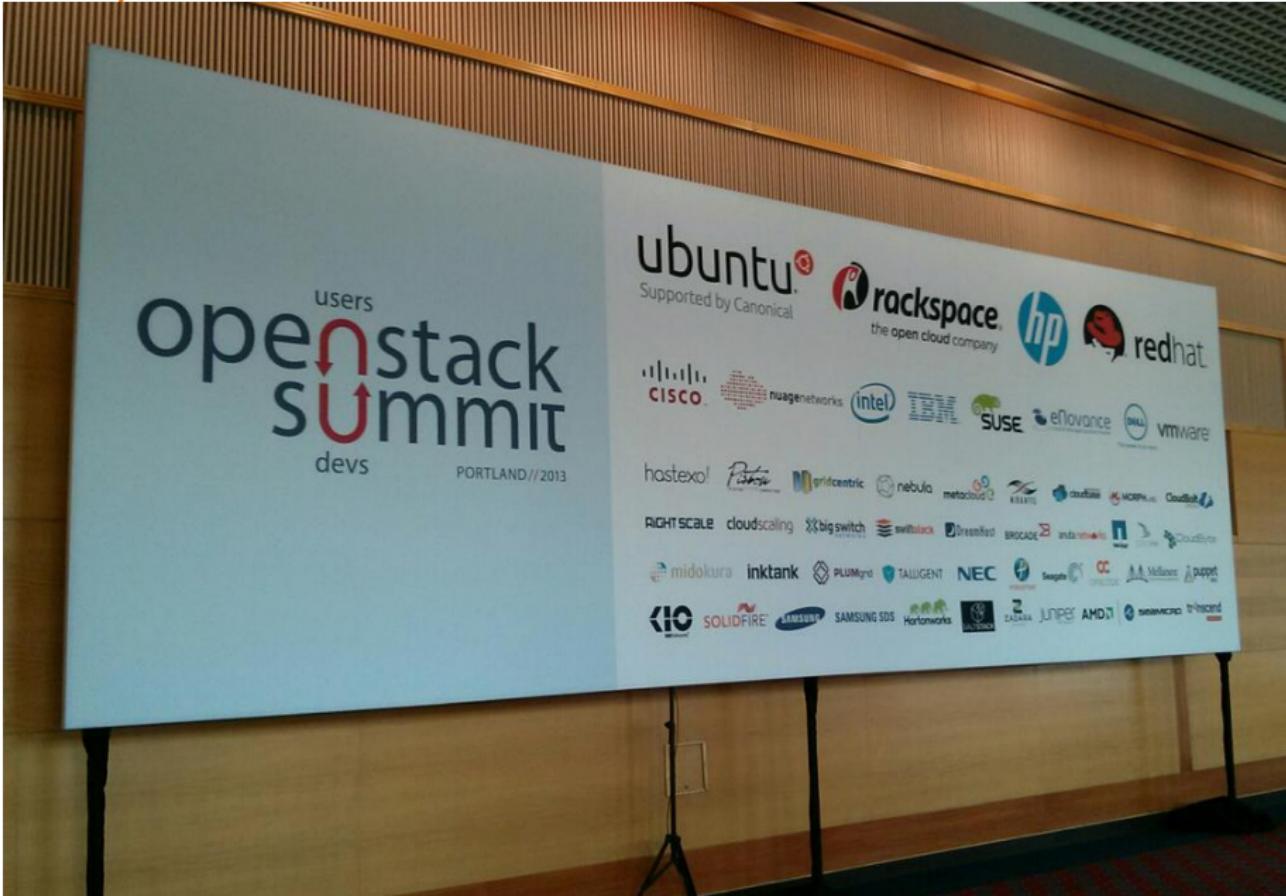
- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ OpenStack Summit

2014-08-11

- Aux USA jusqu'en 2013
- Aujourd'hui : alternance USA et Asie/Europe
- Quelques centaines au début à 4500 de participants aujourd'hui
- En parallèle : conférence (utilisateurs, décideurs) et Design Summit (développeurs)
- Détermine le nom de la release : lieu/ville à proximité du Summit



# Exemple du Summit d'avril 2013 à Portland



## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Exemple du Summit d'avril 2013 à Portland

2014-08-11

Exemple du Summit d'avril 2013 à Portland



# Exemple du Summit de mai 2014 à Atlanta



## Formation OpenStack

└ OpenStack : projet, logiciel et utilisation

  └ Présentation du projet et du logiciel

    └ Exemple du Summit de mai 2014 à Atlanta

Exemple du Summit de mai 2014 à Atlanta



## Exemple du Summit de mai 2014 à Atlanta



### Formation OpenStack

└ OpenStack : projet, logiciel et utilisation

  └ Présentation du projet et du logiciel

    └ Exemple du Summit de mai 2014 à Atlanta

2014-08-11



# Principes de conceptions

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

- Scalability and elasticity are our main goals
- Any feature that limits our main goals must be optional
- Everything should be asynchronous. If you can't do something asynchronously, see #2
- All required components must be horizontally scalable
- Always use shared nothing architecture (SN) or sharding. If you can't Share nothing/shard, see #2
- Distribute everything. Especially logic. Move logic to where state naturally exists.
- Accept eventual consistency and use it where it is appropriate.
- Test everything. We require tests with submitted code. (We will help you if you need it)

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Principes de conceptions

2014-08-11

Shared Nothing :

[http://en.wikipedia.org/wiki/Shared\\_nothing\\_architecture](http://en.wikipedia.org/wiki/Shared_nothing_architecture) :  
en gros, pas de SPOF

Principes de conceptions

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

- Scalability and elasticity are our main goals
- Any feature that limits our main goals must be optional
- Everything should be asynchronous. If you can't do something asynchronously, see #2
- All required components must be horizontally scalable
- Always use shared nothing architecture (SN) or sharding. If you can't Share nothing/shard, see #2
- Distribute everything. Especially logic. Move logic to where state naturally exists.
- Accept eventual consistency and use it where it is appropriate.
- Test everything. We require tests with submitted code. (We will help you if you need it)



# Implémentation

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

- Chaque sous-projet est découpé en plusieurs services



- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)

# Implémentation

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

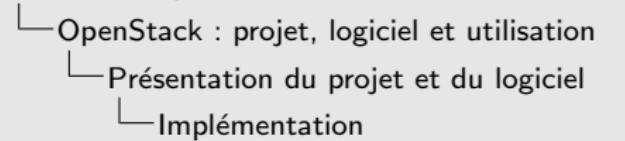
- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)



- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL

# Implémentation

## Formation OpenStack



2014-08-11

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL



- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants

# Implémentation

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants



# Implémentation

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch



- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)

# Implémentation

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)



- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)
- APIs supportées : OpenStack et équivalent Amazon

# Implémentation

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Implémentation

2014-08-11

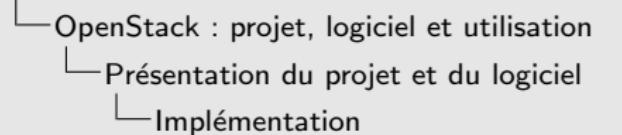
- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)
- APIs supportées : OpenStack et équivalent Amazon



# Implémentation

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)
- APIs supportées : OpenStack et équivalent Amazon
- Multi tenancy

## Formation OpenStack



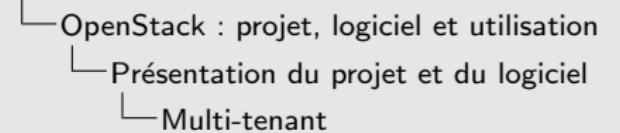
2014-08-11

- Chaque sous-projet est découpé en plusieurs services
- Communication entre les services : AMQP (RabbitMQ)
- Base de données : MySQL
- Réutilisation de nombreux composants existants
- OpenVSwitch
- Tout est développé en Python (Django pour Horizon)
- APIs supportées : OpenStack et équivalent Amazon
- Multi tenancy

# 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"

## Formation OpenStack

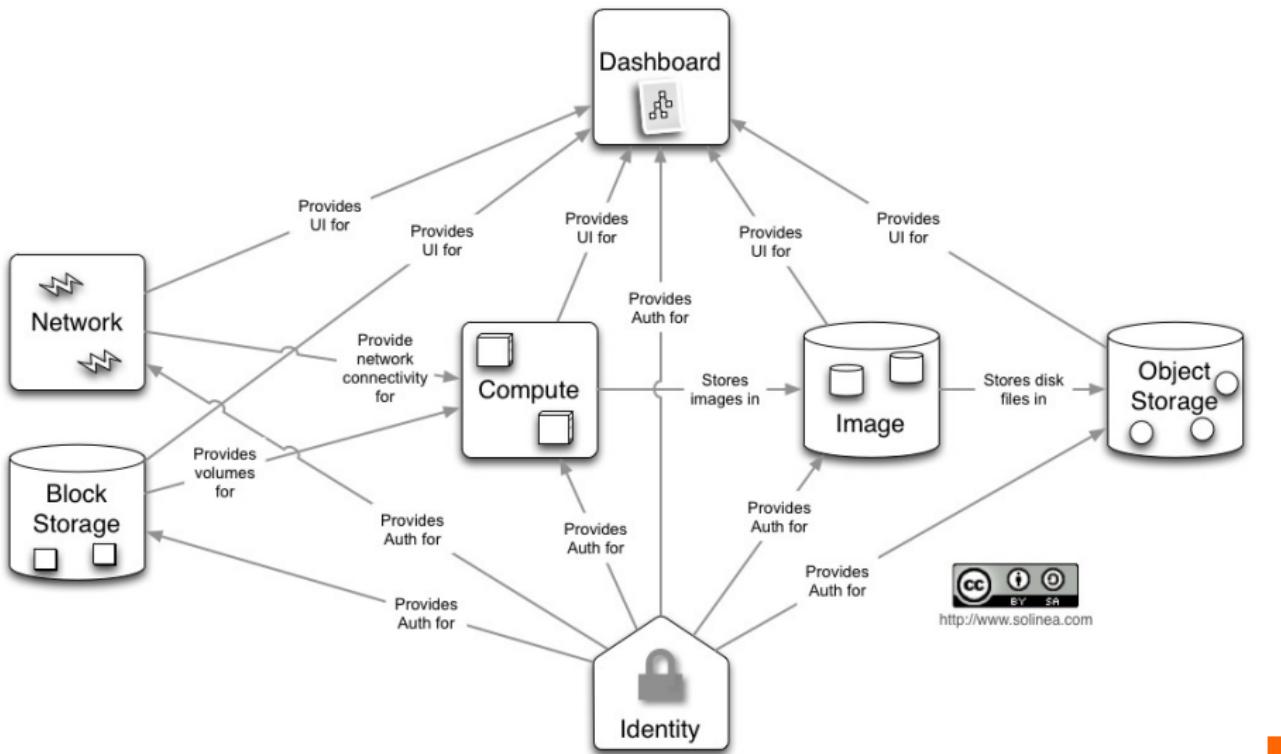


2014-08-11

- 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"



# Architecture

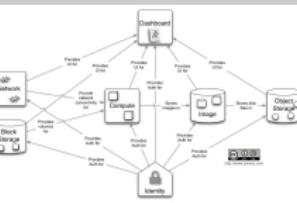


## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Architecture

2014-08-11

Architecture



# Interface web / Dashboard : Horizon

The screenshot shows the OpenStack Horizon dashboard for the 'ubuntu' project. The top navigation bar includes a 'Caution: You are acting as an admin user in the project dashboard.' message and a 'Learn More' button. A success message 'Success: Launched instance named "ubuntu"' is displayed. The main content area is titled 'Instances' and lists three instances: 'ubuntu' (IP 10.92.193.4), 'server' (IP 10.92.193.6), and 'server1' (IP 10.92.193.3). Each instance row includes columns for Instance Name, IP Address, Size, Keypair, Status, Task, Power State, and Actions. The 'ubuntu' instance is currently selected. On the left, a sidebar menu lists 'Project' (selected), 'Admin', 'Manage Compute' (selected), 'Overview', 'Instances' (selected), 'Volumes', 'Images & Snapshots', 'Access & Security', and 'Networks'. A status bar at the bottom shows various system icons.

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Interface web / Dashboard : Horizon

2014-08-11

Interface web / Dashboard : Horizon



- <http://docs.openstack.org/>
- <https://ask.openstack.org>
- [openstack@lists.openstack.org](mailto:openstack@lists.openstack.org)
- [#openstack@Freenode](#)
- <http://api.openstack.org/>
- Communauté française :
  - ▶ <http://openstack.fr/>
  - ▶ [openstack-fr@lists.openstack.org](mailto:openstack-fr@lists.openstack.org)
  - ▶ [#openstack-fr@Freenode](#)
  - ▶ Association

# Ressources

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Présentation du projet et du logiciel
    - └ Ressources

2014-08-11

- <http://docs.openstack.org/>
- <https://ask.openstack.org>
- [openstack@lists.openstack.org](mailto:openstack@lists.openstack.org)
- [#openstack@Freenode](#)
- <http://api.openstack.org/>
- Communauté française :
  - ▶ <http://openstack.fr/>
  - ▶ [openstack-fr@lists.openstack.org](mailto:openstack-fr@lists.openstack.org)
  - ▶ [#openstack-fr@Freenode](#)
  - ▶ Association



# Plan

## 1 Virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

- Présentation du projet et du logiciel
- DevStack : faire tourner rapidement un OpenStack
- Utiliser OpenStack

## 4 Déployer OpenStack

## 5 Tirer partie de l'IaaS



# Utilité de DevStack

- Déployer rapidement un OpenStack
- Utilisé par les développeurs pour tester leurs changements
- Utilisé pour faire des démonstrations
- Utilisé pour tester les APIs sans se préoccuper du déploiement
- Ne doit PAS être utilisé pour la production

## Formation OpenStack

└ OpenStack : projet, logiciel et utilisation

  └ DevStack : faire tourner rapidement un OpenStack

    └ Utilité de DevStack

2014-08-11

- Déployer rapidement un OpenStack
- Utilisé par les développeurs pour tester leurs changements
- Utilisé pour faire des démonstrations
- Utilisé pour tester les APIs sans se préoccuper du déploiement
- Ne doit PAS être utilisé pour la production



2014-08-11

- Un script shell qui fait tout le travail : stack.sh
- Un fichier de configuration : local.conf
- Installe toute les dépendances nécessaires (paquets)
- Clone les dépôts git (branche master par défaut)
- Lance tous les composants dans un screen

- Un script shell qui fait tout le travail : stack.sh
- Un fichier de configuration : local.conf
- Installe toute les dépendances nécessaires (paquets)
- Clone les dépôts git (branche master par défaut)
- Lance tous les composants dans un screen



# Configuration : local.conf

Exemple

```
[[local|localrc]]
ADMIN_PASSWORD=secrete
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
SERVICE_TOKEN=a682f596-76f3-11e3-b3b2-e716f9080d50
#FIXED_RANGE=172.31.1.0/24
#FLOATING_RANGE=192.168.20.0/25
#HOST_IP=10.3.4.5
```

## Formation OpenStack

- OpenStack : projet, logiciel et utilisation

- DevStack : faire tourner rapidement un OpenStack
  - Configuration : local.conf

2014-08-11

Exemple

```
[[local|localrc]]
ADMIN_PASSWORD=secrete
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
SERVICE_TOKEN=a682f596-76f3-11e3-b3b2-e716f9080d50
#FIXED_RANGE=172.31.1.0/24
#FLOATING_RANGE=192.168.20.0/25
#HOST_IP=10.3.4.5
```



# Conseils d'utilisation

## Formation OpenStack

OpenStack : projet, logiciel et utilisation

DevStack : faire tourner rapidement un OpenStack

Conseils d'utilisation

2014-08-11

- DevStack installe beaucoup de choses sur la machine
- Il est recommandé de travailler dans une VM
- Pour tester tous les composants OpenStack dans de bonnes conditions, plusieurs Go de RAM sont nécessaires
- L'utilisation de Vagrant est conseillée



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

- Présentation du projet et du logiciel
- DevStack : faire tourner rapidement un OpenStack
- Utiliser OpenStack

4 Déployer OpenStack

5 Tirer partie de l'IaaS

## Formation OpenStack

- OpenStack : projet, logiciel et utilisation
  - Utiliser OpenStack
    - Plan

2014-08-11



# Le principe

- Toutes les fonctionnalités sont accessibles par l'API
- Les clients (y compris Horizon) utilisent l'API
- Des crédentails sont nécessaires
  - ▶ API OpenStack : utilisateur + mot de passe + tenant
  - ▶ API AWS : access key ID + secret access key

## Formation OpenStack

- └ OpenStack : projet, logiciel et utilisation
  - └ Utiliser OpenStack
    - └ Le principe

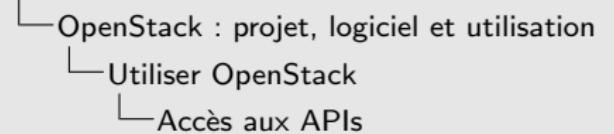
2014-08-11

- Toutes les fonctionnalités sont accessibles par l'API
- Les clients (y compris Horizon) utilisent l'API
- Des crédentails sont nécessaires
  - API OpenStack : utilisateur + mot de passe + tenant
  - API AWS : access key ID + secret access key



# Accès aux APIs

## Formation OpenStack



2014-08-11

- Direct, en HTTP, via des outils comme curl
- Avec une bibliothèque
  - Les implémentations officielles en Python
  - D'autres implémentations pour d'autres langages (exemple : jclouds)
- Avec les outils officiels en ligne de commande
- Avec Horizon

- Direct, en HTTP, via des outils comme curl
- Avec une bibliothèque
  - Les implémentations officielles en Python
  - D'autres implémentations pour d'autres langages (exemple : jclouds)
- Avec les outils officiels en ligne de commande
- Avec Horizon



# Clients officiels

## Formation OpenStack



2014-08-11

- Le projet fournit des clients officiels : python-PROJETclient
- Bibliothèques Python
- Outils CLI
  - ▶ L'authentification se fait en passant les credentials par paramètres ou variables d'environnement
  - ▶ L'option –debug affiche la communication HTTP

- Le projet fournit des clients officiels : python-PROJETclient
- Bibliothèques Python
- Outils CLI
  - ▶ L'authentification se fait en passant les credentials par paramètres ou variables d'environnement
  - ▶ L'option –debug affiche la communication HTTP





# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS

# Ce qu'on va voir

- Installer OpenStack à la main <http://docs.openstack.org/icehouse/install-guide/install/apt/content/>
- Donc comprendre son fonctionnement
- Tour d'horizon des solutions de déploiement

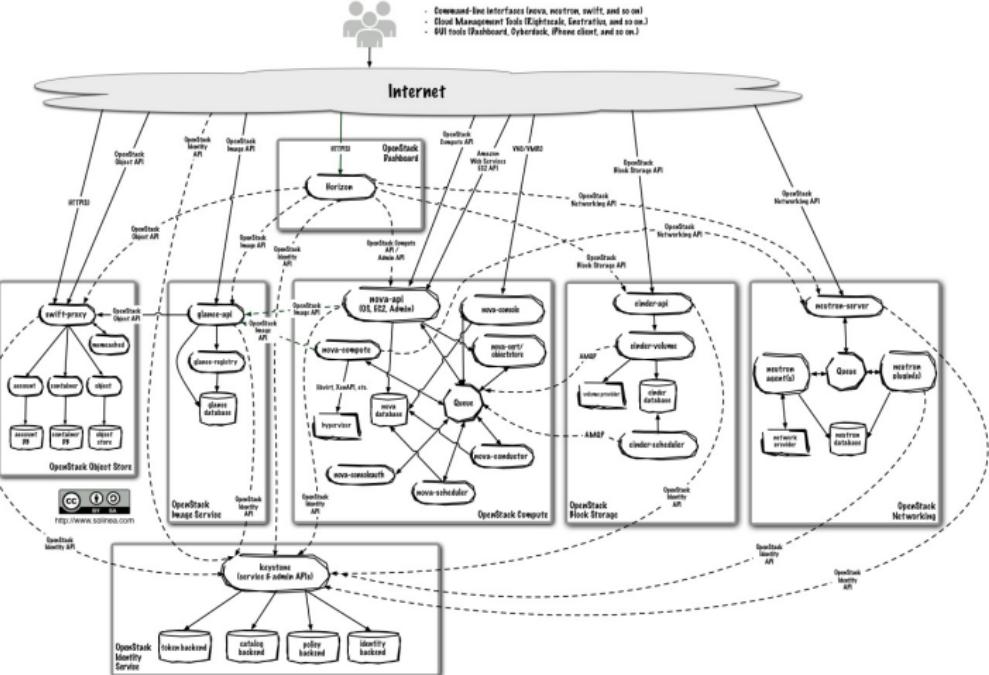
Formation OpenStack  
└ Déployer OpenStack  
    └ Ce qu'on va voir

2014-08-11

- Installer OpenStack à la main <http://docs.openstack.org/icehouse/install-guide/install/apt/content/>
- Donc comprendre son fonctionnement
- Tour d'horizon des solutions de déploiement



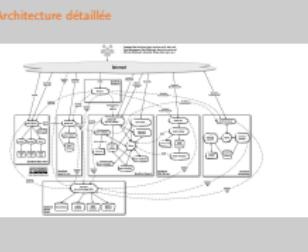
## Architecture détaillée



## Formation OpenStack

- └ Déployer OpenStack

#### └ Architecture détaillée



# Quelques éléments de configuration généraux

- Tous les composants doivent être configurés pour communiquer avec Keystone
- La plupart doivent être configurés pour communiquer avec MySQL et RabbitMQ
- Les composants découpés en plusieurs services ont souvent un fichier de configuration par service
- api-paste.ini contient des paramètres concernant le service API

Formation OpenStack

└ Déployer OpenStack

└ Quelques éléments de configuration généraux

2014-08-11

Quelques éléments de configuration généraux

- Tous les composants doivent être configurés pour communiquer avec Keystone
- La plupart doivent être configurés pour communiquer avec MySQL et RabbitMQ
- Les composants découplés en plusieurs services ont souvent un fichier de configuration par service
- api-paste.ini contient des paramètres concernant le service API



# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

- Les briques nécessaires
  - Keystone : Authentification, autorisation et catalogue de services
  - Nova : Compute
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
  - Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

## 5 Tirer partie de l'IaaS

2014-08-11

Formation OpenStack

- └ Déployer OpenStack
  - └ Les briques nécessaires
    - └ Plan

Tirer partie de l'IaaS

# Système d'exploitation

- OS Linux avec Python
- Historiquement : Ubuntu
- Red Hat s'est largement rattrapé
- SUSE, etc.

## Formation OpenStack

### Déployer OpenStack

#### Les briques nécessaires

##### Système d'exploitation

2014-08-11

- OS Linux avec Python
- Historiquement : Ubuntu
- Red Hat s'est largement rattrapé
- SUSE, etc.





- OpenStack est aujourd'hui compatible Python 2.7
- Afin de ne pas réinventer la roue, beaucoup de dépendances sont nécessaires
- Un travail de portage vers Python 3 est en cours

# Python



- OpenStack est aujourd'hui compatible Python 2.7
- Afin de ne pas réinventer la roue, beaucoup de dépendances sont nécessaires
- Un travail de portage vers Python 3 est en cours

## Formation OpenStack

- Déployer OpenStack
  - Les briques nécessaires
    - Python

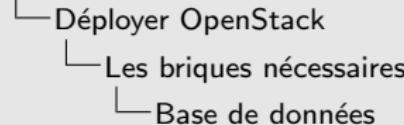
2014-08-11



# Base de données

- Permet de stocker la majorité des données gérées par OpenStack
- Chaque composant a sa propre base
- OpenStack utilise l'ORM Python SQLAlchemy
- Support théorique équivalent à celui de SQLAlchemy
- MySQL est l'implémentation la plus largement testée et utilisée
- SQLite est principalement utilisé dans le cadre de tests et démo
- Certains déploiements fonctionnent avec PostgreSQL

## Formation OpenStack



2014-08-11

- Permet de stocker la majorité des données gérées par OpenStack
- Chaque composant a sa propre base
- OpenStack utilise l'ORM Python SQLAlchemy
- Support théorique équivalent à celui de SQLAlchemy
- MySQL est l'implémentation la plus largement testée et utilisée
- SQLite est principalement utilisé dans le cadre de tests et démo
- Certains déploiements fonctionnent avec PostgreSQL





# Pourquoi l'utilisation de SQLAlchemy

# SQLAlchemy

- Support de multiples BDD
- Gestion des migrations (changement de structure des données)



## Formation OpenStack

### Déployer OpenStack

#### Les briques nécessaires

##### Pourquoi l'utilisation de SQLAlchemy

2014-08-11

# Passage de messages

## Formation OpenStack

### Déployer OpenStack

#### Les briques nécessaires

##### Passage de messages

2014-08-11

- AMQP : Advanced Message Queuing Protocol
- Messages, file d'attente, routage
- Les processus OpenStack communiquent via AMQP
- Plusieurs implémentations possibles : Qpid, 0MQ, etc.
- RabbitMQ par défaut

- AMQP : Advanced Message Queuing Protocol
- Messages, file d'attente, routage
- Les processus OpenStack communiquent via AMQP
- Plusieurs implémentations possibles : Qpid, 0MQ, etc.
- RabbitMQ par défaut

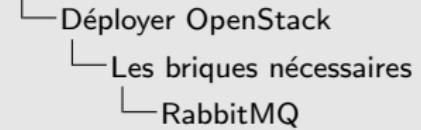


- RabbitMQ est implémenté en Erlang
- Une machine virtuelle Erlang est donc nécessaire



- RabbitMQ est implémenté en Erlang
- Une machine virtuelle Erlang est donc nécessaire

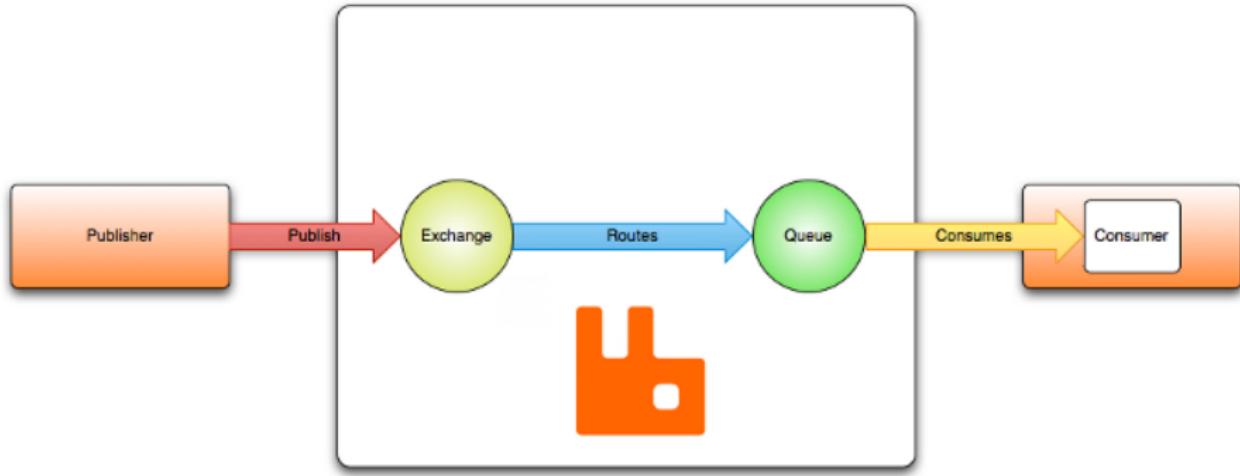
## Formation OpenStack



2014-08-11

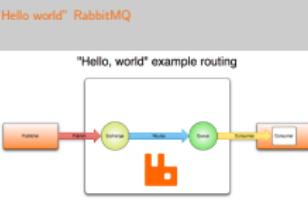
# "Hello world" RabbitMQ

## "Hello, world" example routing



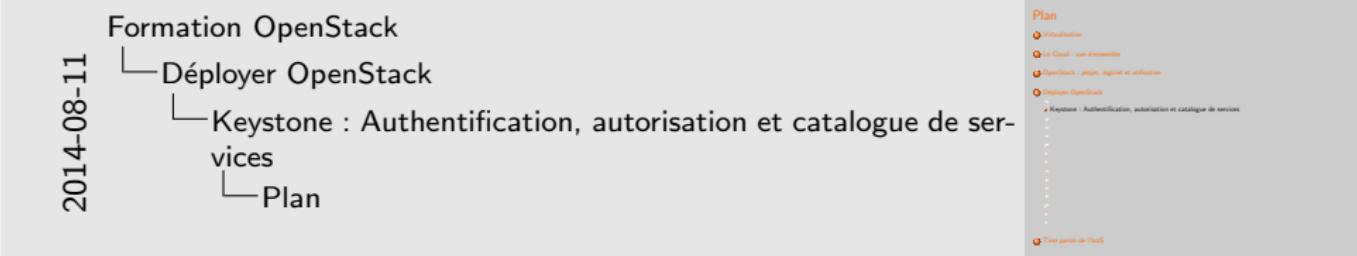
Formation OpenStack  
└ Deployer OpenStack  
  └ Les briques nécessaires  
    └ "Hello world" RabbitMQ

2014-08-11



# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation
- 4 Déployer OpenStack
  - Les briques nécessaires
  - Keystone : Authentification, autorisation et catalogue de services
  - Nova : Compute
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
  - Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes
- 5 Tirer partie de l'IaaS



- Annuaire des utilisateurs et des groupes
- Catalogue de services
- Gère l'authentification et l'autorisation
- Support des domaines dans l'API v3
- Fournit un token à l'utilisateur

# Principes

## Formation OpenStack

### Déployer OpenStack

#### Keystone : Authentification, autorisation et catalogue de services

- Principes

2014-08-11

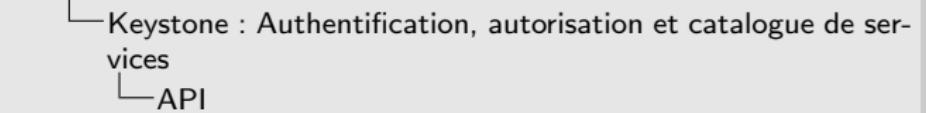
- Annuaire des utilisateurs et des groupes
- Catalogue de services
- Gère l'authentification et l'autorisation
- Support des domaines dans l'API v3
- Fournit un token à l'utilisateur



- API admin : port 35357
- API utilisateur : port 5000
- Deux versions : v2 (actuelle) et v3 (future)
- Gère *utilisateurs, groupes, domaines* (APIv3)
- Les utilisateurs ont des *rôles* sur des *tenants* (projets)

## Formation OpenStack

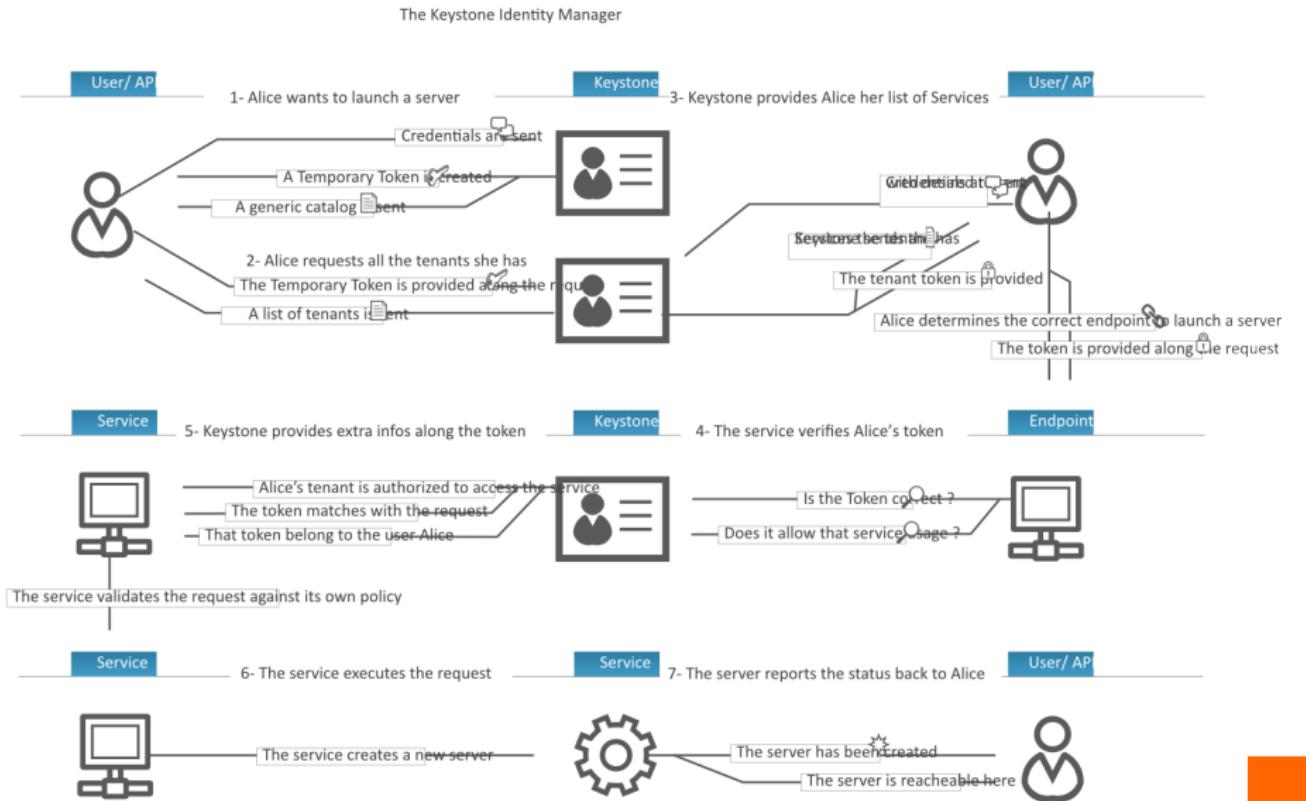
### Déployer OpenStack



2014-08-11

- API admin : port 35357
- API utilisateur : port 5000
- Deux versions : v2 (actuelle) et v3 (future)
- Gère utilisateurs, groupes, domaines (APIv3)
- Les utilisateurs ont des rôles sur des tenants (projets)

## Scénario d'utilisation typique

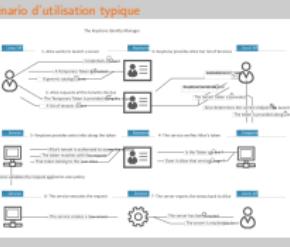


## Formation OpenStack

## └ Déployer OpenStack

Keystone : Authentification, autorisation et catalogue de services

## — Scénario d'utilisation typique



# Installation et configuration

- Paquet : keystone
- Configuration d'un token ADMIN pour la configuration initiale
- Backends : choix de SQL ou LDAP (ou AD)
- Configurer les différents services
- Policy.json
- Services et endpoints
- Utilisateurs, groupes, domaines

## Formation OpenStack

### Déployer OpenStack

#### Keystone : Authentification, autorisation et catalogue de services

##### Installation et configuration

2014-08-11

- Paquet : keystone
- Configuration d'un token ADMIN pour la configuration initiale
- Backends : choix de SQL ou LDAP (ou AD)
- Configurer les différents services
- Policy.json
- Services et endpoints
- Utilisateurs, groupes, domaines



```
Il faut renseigner l'existence des différents services (catalogue) dans Keystone :
$ keystone service-create --name=cinderv2 --type=volumev2 \
--description="Cinder Volume Service V2"
$ keystone endpoint-create \
--region=myRegion
--service-id=...
--publicurl=http://controller:8776/v2/\(tenant_id\)\s \
--internalurl=http://controller:8776/v2/\(tenant_id\)\s \
--adminurl=http://controller:8776/v2/\(tenant_id\)\s
```

## Formation OpenStack

## Déployer OpenStack

## Keystone : Authentification, autorisation et catalogue de services

## Enregistrer un service et son endpoint

2014-08-11

## Enregistrer un service et son endpoint

Il faut renseigner l'existence des différents services (catalogue) dans Keystone :

```
$ keystone service-create --name=cinderv2 --type=volumev2 \
--description="Cinder Volume Service V2"
$ keystone endpoint-create \
--region=myRegion
--service-id=...
--publicurl=http://controller:8776/v2/\(tenant_id\)\s \
--internalurl=http://controller:8776/v2/\(tenant_id\)\s \
--adminurl=http://controller:8776/v2/\(tenant_id\)\s
```



```
$ keystone service-list  
...  
$ keystone user-list  
...  
$ keystone token-get  
...
```

2014-08-11

```
$ keystone service-list  
...  
$ keystone user-list  
...  
$ keystone token-get  
...
```





# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- **Nova : Compute**
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
  - Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

5 Tirer partie de l'IaaS

# Principes

Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Principes

2014-08-11

- Gère les instances
- IP flottantes, groupes de sécurité
- Les instances sont créées à partir des images fournies par Glance
- Les interfaces réseaux des instances sont associées à des ports Neutron



# Interactions avec les autres composants

Instance

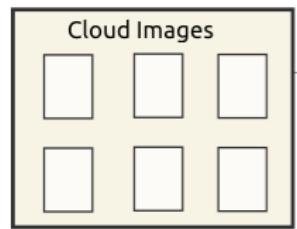
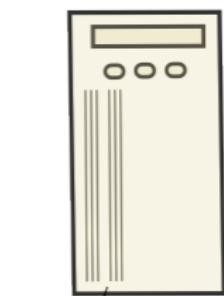
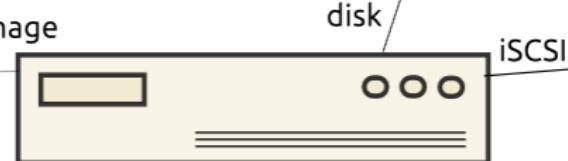
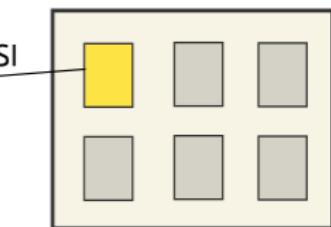


Image Store

Get Image



Compute Node



Volume Storage

Formation OpenStack

- └ Déployer OpenStack
- └ Nova : Compute

└ Interactions avec les autres composants

2014-08-11



Interactions avec les autres composants



Instance  
Compute Node  
Volume Storage

2014-08-11

# API

Gère :

- Instances
- Flavors (types d'instance)
- Security Groups (groupes de sécurité)
- Floating IPs (IPs flottantes)

Les instances sont redimensionnables et migrables d'un hôte physique à un autre.



# Les groupes de sécurité

## Formation OpenStack

### Déployer OpenStack

#### Nova : Compute

##### Les groupes de sécurité

2014-08-11

- 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



- Équivalent des "instance types" d'AWS
- Définit un modèle d'instance en termes de CPU, RAM, disque

# Rôle des flavors

Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Rôle des flavors

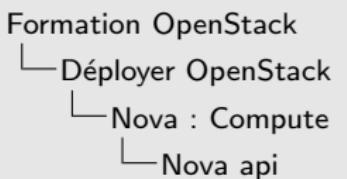
2014-08-11

- Équivalent des "instance types" d'AWS
- Définit un modèle d'instance en termes de CPU, RAM, disque



# Nova api

- Double rôle
- API de manipulation des instances par l'utilisateur
- API à destination des instances : API de metadata
- L'API de metadata doit être accessible à l'adresse  
`http://169.254.169.254/`
- L'API de metadata fournit des informations de configuration personnalisées à chacune des instances



2014-08-11

- Double rôle
- API de manipulation des instances par l'utilisateur
- API à destination des instances : API de metadata
- L'API de metadata doit être accessible à l'adresse  
`http://169.254.169.254/`
- L'API de metadata fournit des informations de configuration personnalisées à chacune des instances



2014-08-11

Nova compute

- Exécute les machines virtuelles
- Tire partie de libvirt ou d'autres APIs comme XenAPI
- Drivers : libvirt, XenAPI, VMWare ESX, Docker
- Permet de récupérer les logs de la console et une console VNC



# Nova scheduler

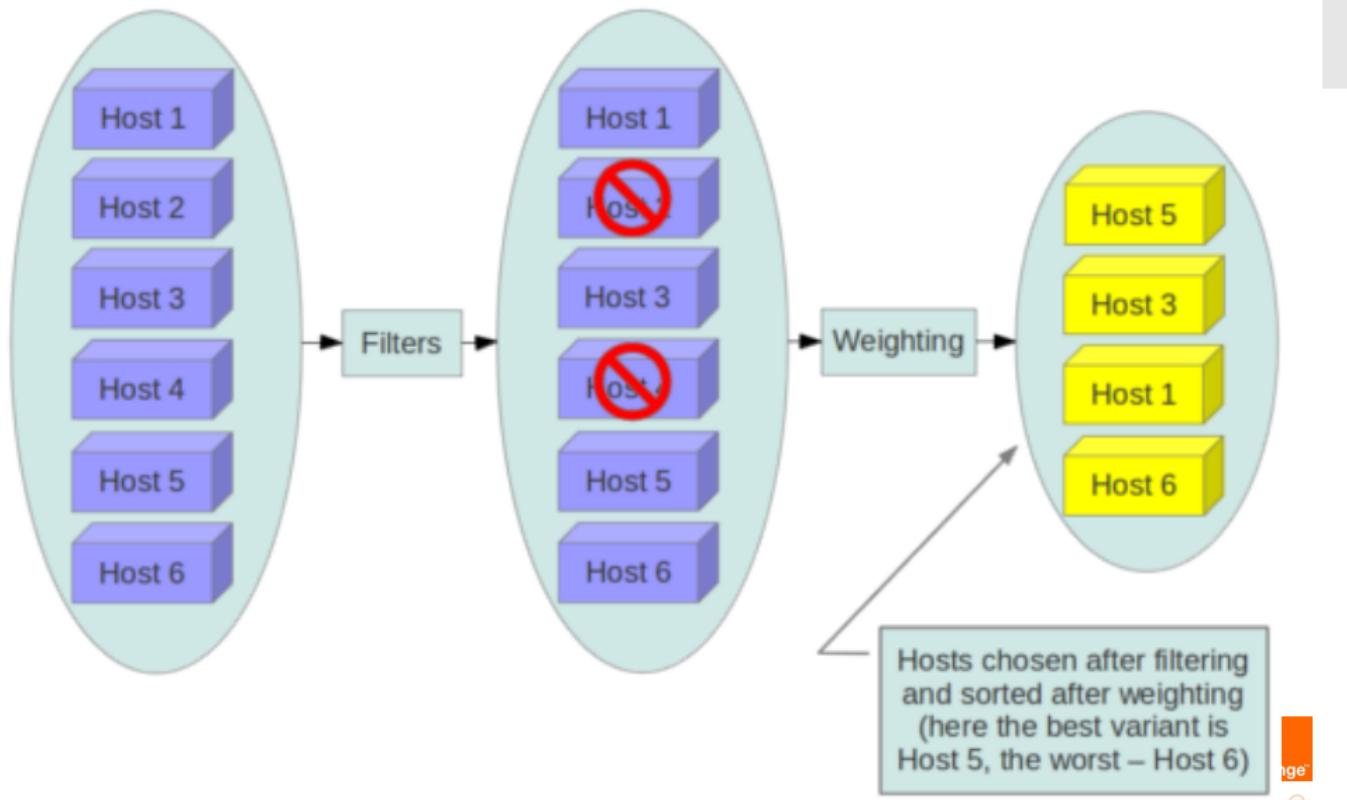
Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Nova scheduler

2014-08-11

- Service qui distribue les demandes d'instances sur les noeuds compute
- Filter, Chance, Multi Scheduler
- Filtres, par défaut : AvailabilityZoneFilter,RamFilter,ComputeFilter
- Tri par poids, par défaut : RawWeigher

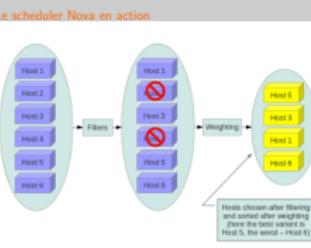


# Le scheduler Nova en action



Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Le scheduler Nova en action

2014-08-11



# Nova conductor

Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Nova conductor

2014-08-11

- Service facultatif qui améliore la sécurité
- Fait office de proxy entre les noeuds compute et la BDD
- Les noeuds compute, vulnérables, n'ont donc plus d'accès à la BDD



# Tester

Formation OpenStack  
└ Déployer OpenStack  
  └ Nova : Compute  
    └ Tester

2014-08-11

```
$ nova list  
...  
$ nova create  
...
```

```
$ nova list  
...  
$ nova create  
...
```



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



# Principes

- Registre d'images (et des snapshots)
- Propriétés sur les images
- Est utilisé par Nova pour démarrer des instances
- Peut utiliser Swift comme back-end de stockage

## Formation OpenStack

### Déployer OpenStack

#### Glance : Registre d'images

##### Principes

2014-08-11

- Registre d'images (et des snapshots)
- Propriétés sur les images
- Est utilisé par Nova pour démarrer des instances
- Peut utiliser Swift comme back-end de stockage



- L'utilisateur peut définir un certain nombre de propriétés dont certaines seront utilisées lors de l'instanciation
  - Type d'image
  - Architecture
  - Distribution
  - Version de la distribution
  - Espace disque minimum
  - RAM minimum
  - Publique ou non

# Propriétés des images dans Glance

## Formation OpenStack

### Déployer OpenStack

#### Glance : Registre d'images

##### Propriétés des images dans Glance

2014-08-11

L'utilisateur peut définir un certain nombre de propriétés dont certaines seront utilisées lors de l'instanciation

- Type d'image
- Architecture
- Distribution
- Version de la distribution
- Espace disque minimum
- RAM minimum
- Publique ou non



Glance supporte un large éventail de types d'images, limité par le support de l'hyperviseur sous-jacent à Nova

- raw
- qcow2
- ami
- vmdk
- iso

# Types d'images

## Formation OpenStack

### Déployer OpenStack

#### Glance : Registre d'images

- Types d'images

2014-08-11

Glance supporte un large éventail de types d'images, limité par le support de l'hyperviseur sous-jacent à Nova

- raw
- qcow2
- ami
- vmdk
- iso



# Backends

- Swift ou S3
- Ceph
- HTTP
- Répertoire local

## Formation OpenStack

### Déployer OpenStack

#### Glance : Registre d'images

- Backends

2014-08-11

- Swift ou S3
- Ceph
- HTTP
- Répertoire local



2014-08-11

- Paquet glance-api : fournit l'API
- Paquet glance-registry : démon du registre d'images en tant que tel



# Tester

## Formation OpenStack

### Déployer OpenStack

#### Glance : Registre d'images

##### Tester

2014-08-11

```
$ glance image-list  
...  
$ glance image-create  
...
```

```
$ glance image-list
```

```
...
```

```
$ glance image-create
```

```
...
```



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- **Neutron : Réseau en tant que service**
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



# Principes

- SDN
- Auparavant Quantum et nova-network
- neutron-server : fournit l'API
- Agent DHCP : fournit le service de DHCP pour les instances
- Agent L3 : gère la couche 3 du réseau, le routage
- Plugin : OpenVSwitch par défaut, d'autres implémentations libres/propriétaires, logicielles/matérielles existent

## Formation OpenStack

### Déployer OpenStack

#### Neutron : Réseau en tant que service

- Principes

2014-08-11

- SDN
- Auparavant Quantum et nova-network
- neutron-server : fournit l'API
- Agent DHCP : fournit le service de DHCP pour les instances
- Agent L3 : gère la couche 3 du réseau, le routage
- Plugin : OpenVSwitch par défaut, d'autres implémentations libres/propriétaires, logicielles/matérielles existent



Outre les fonctions réseau de base niveaux 2 et 3, Neutron peut fournir d'autres services :

- Load Balancing (HAProxy, ...)
- Firewall (vArmour, ...) : diffère des groupes de sécurité
- VPN (OpenSwan, ...) : permet d'accéder à un réseau privé sans IP flottantes

Ces fonctionnalités se basent également sur des plugins

## Formation OpenStack

### Déployer OpenStack

#### Neutron : Réseau en tant que service

- Fonctionnalités supplémentaires

2014-08-11

# Fonctionnalités supplémentaires

Outre les fonctions réseau de base niveaux 2 et 3, Neutron peut fournir d'autres services :

- Load Balancing (HAProxy, ...)
- Firewall (vArmour, ...) : diffère des groupes de sécurité
- VPN (OpenSwan, ...) : permet d'accéder à un réseau privé sans IP flottantes

Ces fonctionnalités se basent également sur des plugins



2014-08-11

API

L'API permet notamment de manipuler ces ressources

- Réseau : niveau 2
- Subnet : niveau 3
- Port

Les ports peuvent correspondre à des interfaces d'instance

L'API permet notamment de manipuler ces ressources

- Réseau : niveau 2
- Subnet : niveau 3
- Port

Les ports peuvent correspondre à des interfaces d'instance



# Plugins ML2

- Modular Layer 2
- OpenVSwitch
- OpenDaylight
- Contrail, OpenContrail
- Nuage Networks
- cf. OpenFlow

## Formation OpenStack

### └ Déployer OpenStack

#### └ Neutron : Réseau en tant que service

##### └ Plugins ML2

2014-08-11

- Modular Layer 2
- OpenVSwitch
- OpenDaylight
- Contrail, OpenContrail
- Nuage Networks
- cf. OpenFlow



# Implémentation

## Formation OpenStack

### Déployer OpenStack

#### Neutron : Réseau en tant que service

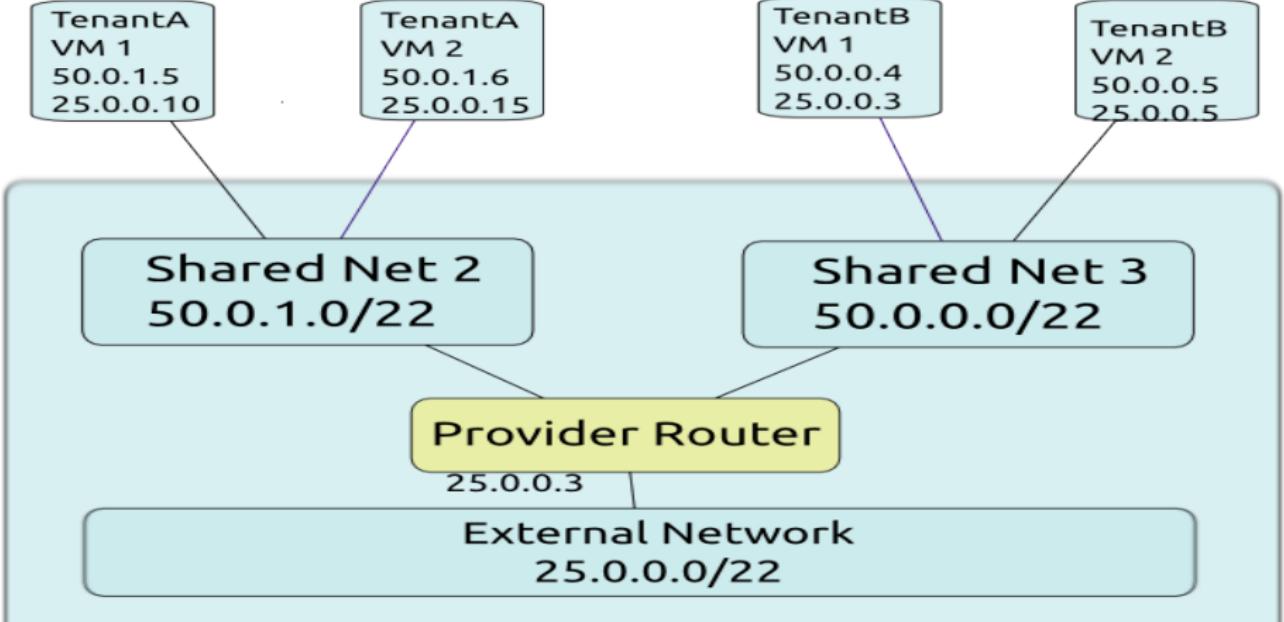
##### Implémentation

2014-08-11

- Neutron tire partie des namespaces réseaux du noyau Linux pour permettre l'IP overlapping
- Le proxy de metadata est un composant qui permet aux instances isolées dans leur réseau de joindre l'API de metadata fournie par Nova



# Schéma



## OpenStack Neutron

25.0.0.1

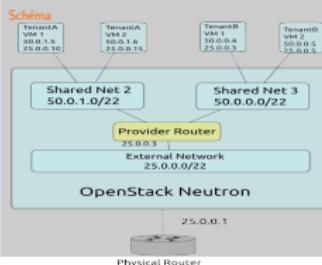


## Formation OpenStack

### Déployer OpenStack

- Neutron : Réseau en tant que service
  - Schéma

2014-08-11

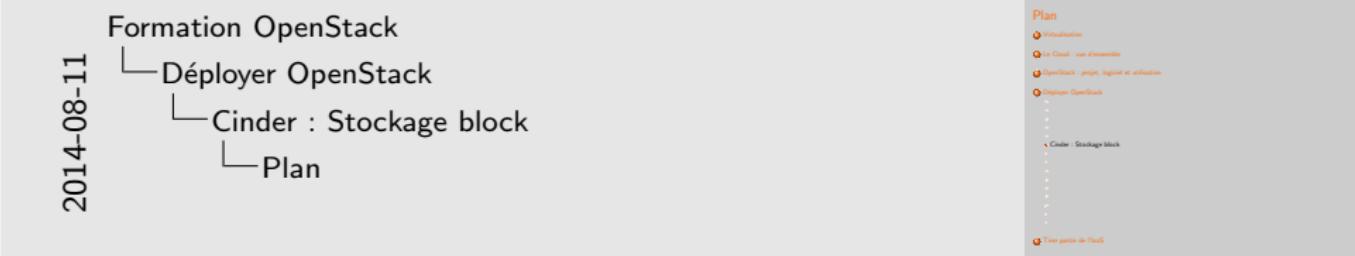


# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation

- 4 Déployer OpenStack
  - Les briques nécessaires
  - Keystone : Authentification, autorisation et catalogue de services
  - Nova : Compute
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
  - Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

- 5 Tirer partie de l'IaaS



# Principes

- Auparavant nova-volume
- Fournit des volumes (stockage block) attachables aux instances
- Gère différents types de volume
- Gère snapshots et backups de volumes
- Attachement via iSCSI par défaut

## Formation OpenStack

### Déployer OpenStack

#### Cinder : Stockage block

##### Principes

2014-08-11

- Auparavant nova-volume
- Fournit des volumes (stockage block) attachables aux instances
- Gère différents types de volume
- Gère snapshots et backups de volumes
- Attachement via iSCSI par défaut



- Cinder n'est **pas** une solution de stockage partagé comme NFS
- OpenStack (tout comme AWS) ne fournit pas de solution *NFS as a Service*
- Cf. le projet Manila

# Du stockage partagé ?

## Formation OpenStack

### Deployer OpenStack

#### Cinder : Stockage block

##### Du stockage partagé ?

2014-08-11

- Cinder n'est **pas** une solution de stockage partagé comme NFS
- OpenStack (tout comme AWS) ne fournit pas de solution *NFS as a Service*
- Cf. le projet Manila



# Utilisation

## Formation OpenStack

### Déployer OpenStack

#### Cinder : Stockage block

##### Utilisation

2014-08-11

- Volume supplémentaire (et stockage persistant) sur une instance
- Boot from volume : l'OS est sur le volume
- Fonctionnalité de backup vers un object store (Swift ou Ceph)



# Installation

- Paquet cinder-api : fournit l'API
- Paquet cinder-volume : création et gestion des volumes
- Paquet cinder-scheduler : distribue les demandes de création de volume
- Paquet cinder-backup : backup vers un object store

## Formation OpenStack

### Déployer OpenStack

#### Cinder : Stockage block

##### Installation

2014-08-11

- Paquet cinder-api : fournit l'API
- Paquet cinder-volume : création et gestion des volumes
- Paquet cinder-scheduler : distribue les demandes de création de volume
- Paquet cinder-backup : backup vers un object store



# Backends

## Formation OpenStack

### Déployer OpenStack

#### Cinder : Stockage block

##### Backends

2014-08-11

- Utilisation de plusieurs backends en parallèle possible
- LVM (par défaut)
- GlusterFS
- Ceph
- Systèmes de stockage propriétaires type NetApp



# Plan

1 Virtualisation

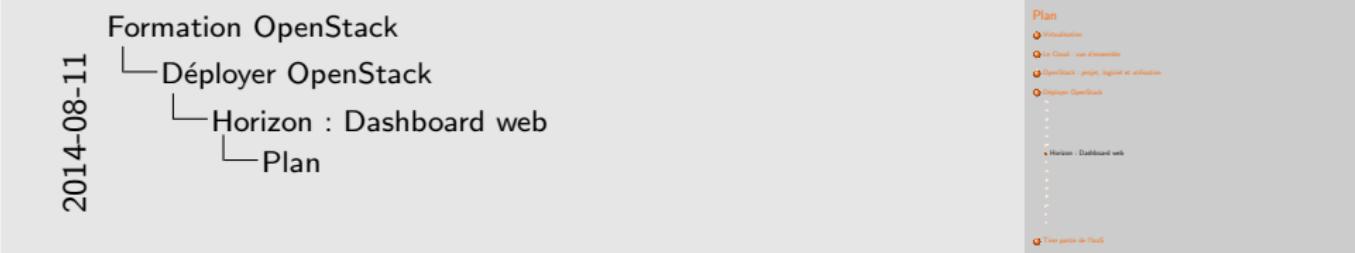
2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



- Utilise les APIs existantes pour fournir une interface
- Horizon est un module Django
- OpenStack Dashboard est l'implémentation officielle de ce module

# django

## Principes

- Utilise les APIs existantes pour fournir une interface
- Horizon est un module Django
- OpenStack Dashboard est l'implémentation officielle de ce module

# django

### Formation OpenStack

#### Déployer OpenStack

##### Horizon : Dashboard web

###### Principes

2014-08-11

# Configuration

## Formation OpenStack

### Déployer OpenStack

#### Horizon : Dashboard web

##### Configuration

local\_settings.py

Les services apparaissent dans Horizon s'ils sont répertoriés dans le catalogue de services de Keystone

2014-08-11

- local\_settings.py
- Les services apparaissent dans Horizon s'ils sont répertoriés dans le catalogue de services de Keystone



- Une zone "admin" restreinte
- Une interface par tenant

## Formation OpenStack

### Déployer OpenStack



2014-08-11

# Utilisation

- Une zone "admin" restreinte
- Une interface par tenant



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

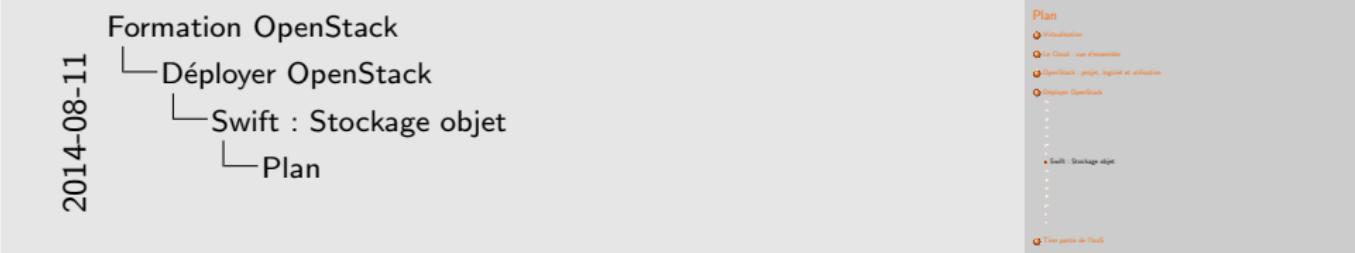
4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web

● Swift : Stockage objet

- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



# Principes

- SDS : Software Defined Storage
- Utilisation de commodity hardware
- Théorème CAP : on en choisit deux
- Accès par les APIs
- Architecture totalement acentrée
- Pas de base de données centrale

## Formation OpenStack

- └ Déployer OpenStack
- └ Swift : Stockage objet
- └ Principes

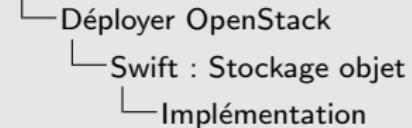
2014-08-11

- SDS : Software Defined Storage
- Utilisation de commodity hardware
- Théorème CAP : on en choisit deux
- Accès par les APIs
- Architecture totalement acentrée
- Pas de base de données centrale



# Implémentation

## Formation OpenStack



2014-08-11

- Proxy : serveur API par lequel passent toutes les requêtes
- Object server : serveur de stockage
- Container server : maintient des listes d'objects dans des containers
- Account server : maintient des listes de containers dans des accounts
- Chaque objet est répliqué n fois (3 par défaut)

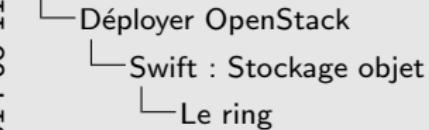
- Proxy : serveur API par lequel passent toutes les requêtes
- Object server : serveur de stockage
- Container server : maintient des listes d'objects dans des containers
- Account server : maintient des listes de containers dans des accounts
- Chaque objet est répliqué n fois (3 par défaut)



# Le ring

- Problème : comment décider quelle donnée va sur quel object server
- Le ring est découpé en partitions
- On situe chaque donnée dans le ring afin de déterminer sa partition
- Une partition est associée à plusieurs serveurs

## Formation OpenStack

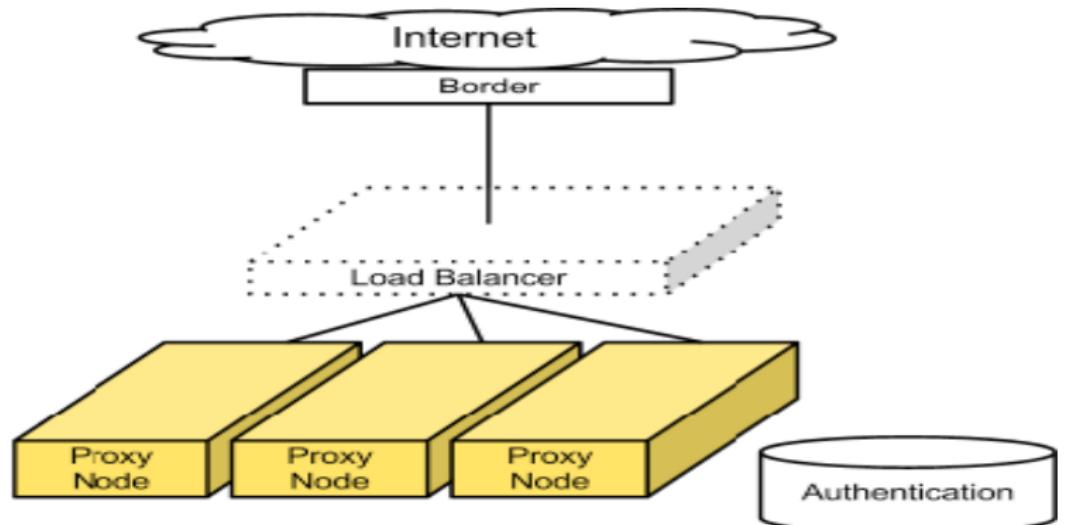


2014-08-11

- Problème : comment décider quelle donnée va sur quel object server
- Le ring est découpé en partitions
- On situe chaque donnée dans le ring afin de déterminer sa partition
- Une partition est associée à plusieurs serveurs



# Schéma



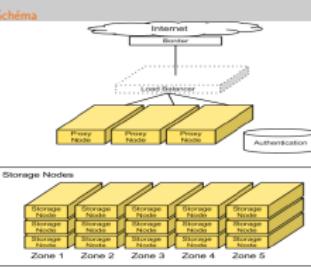
## Formation OpenStack

### Déployer OpenStack

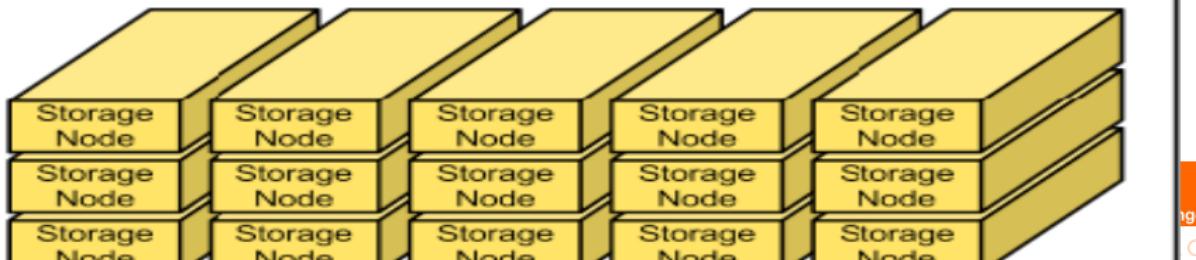
#### Swift : Stockage objet

##### Schéma

2014-08-11



## Storage Nodes



# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation

- 4 Déployer OpenStack
  - Les briques nécessaires
  - Keystone : Authentification, autorisation et catalogue de services
  - Nova : Compute
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
  - Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

- 5 Tirer partie de l'IaaS



2014-08-11

- Indexe différentes métriques concernant l'utilisation des différents services du cloud
- Fournit des APIs permettant de récupérer ces données
- Base pour construire des outils de facturation
- Utilise MongoDB (par défaut) pour le stockage

- Indexe différentes métriques concernant l'utilisation des différents services du cloud
- Fournit des APIs permettant de récupérer ces données
- Base pour construire des outils de facturation
- Utilise MongoDB (par défaut) pour le stockage

Surveiller l'utilisation de son infrastructure avec Ceilometer



# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation

- 4 Déployer OpenStack
  - Les briques nécessaires
  - Keystone : Authentification, autorisation et catalogue de services
  - Nova : Compute
  - Glance : Registre d'images
  - Neutron : Réseau en tant que service
  - Cinder : Stockage block
  - Horizon : Dashboard web
  - Swift : Stockage objet
  - Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
  - Trove : Database as a Service
  - Designate : DNS as a Service
  - Quelques autres composants intéressants
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

- 5 Tirer partie de l'IaaS



# Orchestrer son infrastructure avec Heat

## Formation OpenStack

### Déployer OpenStack

#### Heat : Orchestration des ressources

##### Orchestrer son infrastructure avec Heat

2014-08-11

- Équivalent d'Amazon Cloud Formation
- Orchestrer les ressources compute, storage, network, etc.
- Doit se coupler avec cloud-init

- Équivalent d'Amazon Cloud Formation
- Orchestrer les ressources compute, storage, network, etc.
- Doit se coupler avec cloud-init



# Orchestrer son infrastructure avec Heat

## Formation OpenStack

### Déployer OpenStack

#### Heat : Orchestration des ressources

##### Orchestrer son infrastructure avec Heat

2014-08-11

- Équivalent d'Amazon Cloud Formation
- Orchestrer les ressources compute, storage, network, etc.
- Doit se coupler avec cloud-init
- Description de son infrastructure dans un fichier template, format JSON (CFN) ou YAML (HOT)

- Équivalent d'Amazon Cloud Formation
- Orchestrer les ressources compute, storage, network, etc.
- Doit se coupler avec cloud-init
- Description de son infrastructure dans un fichier template, format JSON (CFN) ou YAML (HOT)



# Autoscaling avec Heat

## Formation OpenStack

### Déployer OpenStack

#### Heat : Orchestration des ressources

- Autoscaling avec Heat

2014-08-11

Heat implémente la fonctionnalité d'autoscaling

- Se déclenche en fonction des alarmes produites par Ceilometer
- Entraîne la création de nouvelles instances

Heat implémente la fonctionnalité d'autoscaling

- Se déclenche en fonction des alarmes produites par Ceilometer
- Entraîne la création de nouvelles instances



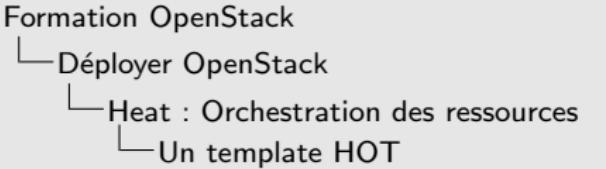
# Un template HOT

```
heat_template_version: 2013-05-23
```

```
description: Simple template to deploy a single compute instance
```

```
resources:
```

```
my_instance:  
  type: OS::Nova::Server  
  properties:  
    key_name: my_key  
    image: F18-x86_64-cfntools  
    flavor: m1.small
```



2014-08-11

```
heat_template_version: 2013-05-23  
description: Simple template to deploy a single compute instance  
resources:  
  my_instance:  
    type: OS::Nova::Server  
    properties:  
      key_name: my_key  
      image: F18-x86_64-cfntools  
      flavor: m1.small
```



# Fonctionnalités avancées de Heat

Formation OpenStack

└ Déployer OpenStack

└ Heat : Orchestration des ressources  
└ Fonctionnalités avancées de Heat

- Nested stacks
- Environments
- Providers

2014-08-11

- Nested stacks
- Environments
- Providers



# Plan

1 Virtualisation

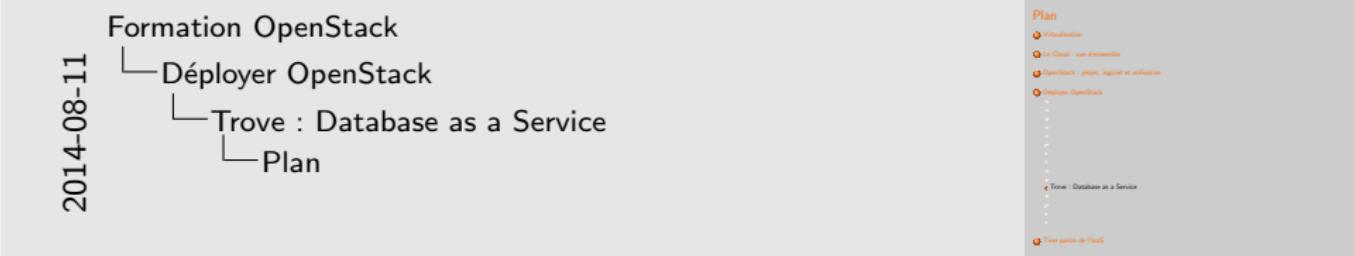
2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



- Fournit des bases de données relationnelles, à la AWS RDS
- A vocation à supporter des bases NoSQL aussi
- Gère notamment MySQL comme back-end
- Se repose sur Nova pour les instances dans lesquelles se fait l'installation d'une BDD

# Principe

## Formation OpenStack

### Déployer OpenStack

#### Trove : Database as a Service

##### Principe

2014-08-11

- Fournit des bases de données relationnelles, à la AWS RDS
- A vocation à supporter des bases NoSQL aussi
- Gère notamment MySQL comme back-end
- Se repose sur Nova pour les instances dans lesquelles se fait l'installation d'une BDD



# Services

## Formation OpenStack

### Déployer OpenStack

#### Trove : Database as a Service

##### Services

- trove-api : API
- trove-taskmanager : gère les instances BDD
- trove-guestagent : agent interne à l'instance

2014-08-11

- trove-api : API
- trove-taskmanager : gère les instances BDD
- trove-guestagent : agent interne à l'instance



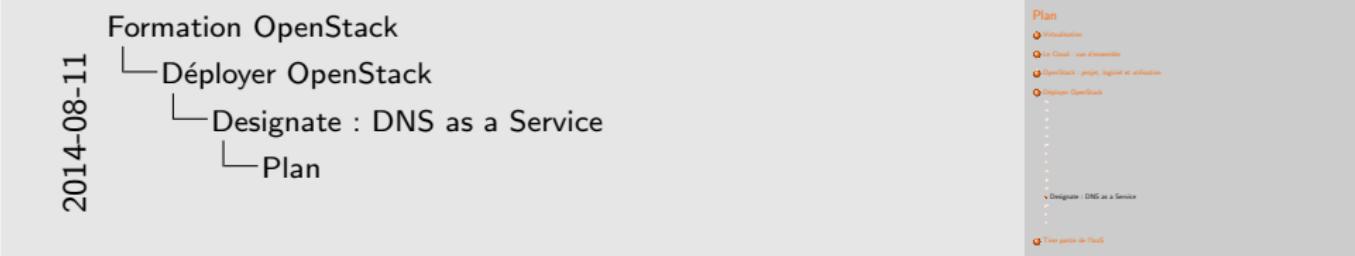
# Plan

- 1 Virtualisation
- 2 Le Cloud : vue d'ensemble
- 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

## 5 Tirer partie de l'IaaS



# Principe

Formation OpenStack

  └ Déployer OpenStack

    └ Designate : DNS as a Service

      └ Principe

2014-08-11

- Équivalent d'AWS Route 53
- Gère différents backends : BIND, etc.





# Formation OpenStack

## Déployer OpenStack

- Quelques autres composants intéressants
  - Plan

2014-08-11

# Plan

## 1 Virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants**
  - Bonnes pratiques pour un déploiement en production
  - Faire face aux problèmes

## 5 Tirer partie de l'IaaS

- Anciennement Nova bare-metal
- Permet le déploiement d'instance sur des machines physiques (plutôt que VMs)
- Repose sur des technologies telles que PXE, TFTP

# Ironic

## Formation OpenStack

### Déployer OpenStack

- Quelques autres composants intéressants
  - Ironic

2014-08-11



# Oslo, ou OpenStack common

Formation OpenStack

└ Déployer OpenStack

└ Quelques autres composants intéressants

└ Oslo, ou OpenStack common

2014-08-11

- Oslo contient le code commun à plusieurs composants d'OpenStack
- Son utilisation est transparente pour le déployeur



# rootwrap

## Formation OpenStack

### Déployer OpenStack

#### Quelques autres composants intéressants

- rootwrap

- Wrapper pour l'utilisation de commandes en root
- Configuration au niveau de chaque composant qui l'utilise
- Permet d'écrire des filtres sur les commandes

2014-08-11

- Wrapper pour l'utilisation de commandes en root
- Configuration au niveau de chaque composant qui l'utilise
- Permet d'écrire des filtres sur les commandes



- OpenStack On OpenStack
- Objectif : pouvoir déployer un cloud OpenStack (overcloud) à partir d'un cloud OpenStack (undercloud)
- Autoscaling du cloud lui-même : déploiement de nouveaux noeuds compute lorsque cela est nécessaire
- Fonctionne conjointement avec Ironic pour le déploiement bare-metal

2014-08-11

- OpenStack On OpenStack
- Objectif : pouvoir déployer un cloud OpenStack (overcloud) à partir d'un cloud OpenStack (undercloud)
- Autoscaling du cloud lui-même : déploiement de nouveaux noeuds compute lorsque cela est nécessaire
- Fonctionne conjointement avec Ironic pour le déploiement bare-metal



# Tempest

## Formation OpenStack

### Déployer OpenStack

#### Quelques autres composants intéressants

- Tempest

2014-08-11

- Suite de tests d'un cloud OpenStack
- Effectue des appels à l'API et vérifie le résultat
- Est très utilisé par les développeurs via l'intégration continue
- Le déployeur peut utiliser Tempest pour vérifier la bonne conformité de son cloud

- Suite de tests d'un cloud OpenStack
- Effectue des appels à l'API et vérifie le résultat
- Est très utilisé par les développeurs via l'intégration continue
- Le déployeur peut utiliser Tempest pour vérifier la bonne conformité de son cloud



# Plan

1 Virtualisation

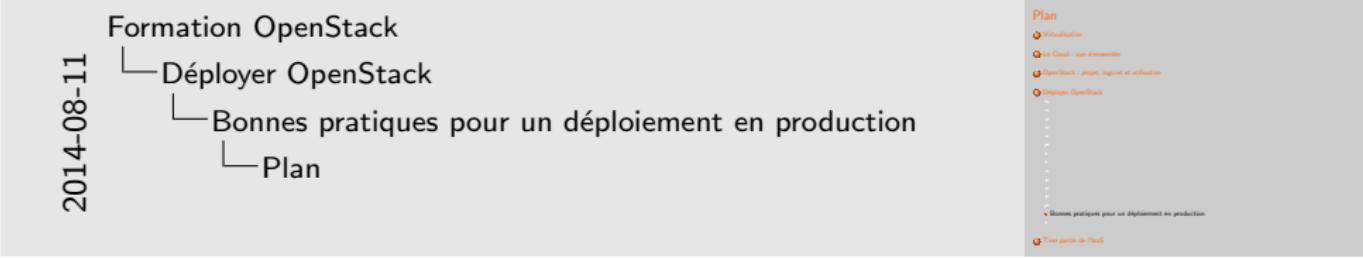
2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

5 Tirer partie de l'IaaS



# Quels composants dois-je installer ?

## Formation OpenStack

### Déployer OpenStack

- └ Bonnes pratiques pour un déploiement en production
  - └ Quels composants dois-je installer ?

2014-08-11

- Keystone est indispensable
- L'utilisation de Nova va de paire avec Glance et Neutron
- Cinder s'avérera souvent utile
- Ceilometer et Heat vont souvent ensemble
- Swift est indépendant des autres composants

- Keystone est indispensable
- L'utilisation de Nova va de paire avec Glance et Neutron
- Cinder s'avérera souvent utile
- Ceilometer et Heat vont souvent ensemble
- Swift est indépendant des autres composants



# Penser dès le début aux choix structurants

## Formation OpenStack

### Déployer OpenStack

- └ Bonnes pratiques pour un déploiement en production
  - └ Penser dès le début aux choix structurants

- Distribution et méthode de déploiement
- Hyperviseur
- Réseau : quelle architecture et quels drivers
- Politique de mise à jour

2014-08-11

- Distribution et méthode de déploiement
- Hyperviseur
- Réseau : quelle architecture et quels drivers
- Politique de mise à jour



# Les différentes méthodes d'installation

## Formation OpenStack

### Déployer OpenStack

- └ Bonnes pratiques pour un déploiement en production
  - └ Les différentes méthodes d'installation

2014-08-11

- DevStack est à oublier pour la production
- TripleO est encore trop jeune
- Le déploiement à la main comme vu précédemment n'est pas recommandé car peu maintenable
- Distributions OpenStack packagées et prêtes à l'emploi
- Distributions classiques et gestion de configuration

- DevStack est à oublier pour la production
- TripleO est encore trop jeune
- Le déploiement à la main comme vu précédemment n'est pas recommandé car peu maintenable
- Distributions OpenStack packagées et prêtes à l'emploi
- Distributions classiques et gestion de configuration



Beaucoup de documentations font référence à ces rôles :

- Controller node : APIs, BDD, AMQP
- Network node : Routeur
- Compute node : Hyperviseur

Ce modèle simplifié n'est pas HA.

# Assigner des rôles aux machines

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Assigner des rôles aux machines

2014-08-11

Beaucoup de documentations font référence à ces rôles :

- Controller node : APIs, BDD, AMQP
- Network node : Routeur
- Compute node : Hyperviseur

Ce modèle simplifié n'est pas HA.



# Haute disponibilité

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Haute disponibilité

2014-08-11

##### Haut disponibilité de l'IaaS

- MySQL, RabbitMQ : HA classique (Galera, Clustering)
- Les services APIs sont stateless et HTTP : scale out et load balancers
- La plupart des autres services OpenStack sont capables de scale out également

Guide HA :

<http://docs.openstack.org/high-availability-guide/content/>

## Haut disponibilité de l'IaaS

- MySQL, RabbitMQ : HA classique (Galera, Clustering)
- Les services APIs sont stateless et HTTP : scale out et load balancers
- La plupart des autres services OpenStack sont capables de scale out également

Guide HA :

<http://docs.openstack.org/high-availability-guide/content/>



# Haute disponibilité de l'agent L3 de Neutron

Formation OpenStack

  └ Déployer OpenStack

    └ Bonnes pratiques pour un déploiement en production

      └ Haute disponibilité de l'agent L3 de Neutron

• Plusieurs solutions possibles

2014-08-11

- Plusieurs solutions possibles



# Considérations pour une environnement de production

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

##### Considérations pour une environnement de production

2014-08-11

- Des URLs uniformes pour toutes les APIs : utiliser un reverse proxy
- Utilisation des quotas
- Prévoir les bonnes volumétries, notamment pour les données Ceilometer
- Monitoring

Guide Operations :

<http://docs.openstack.org/trunk/openstack-ops/content/>

- Des URLs uniformes pour toutes les APIs : utiliser un reverse proxy
- Utilisation des quotas
- Prévoir les bonnes volumétries, notamment pour les données Ceilometer
- Monitoring

Guide Operations :  
<http://docs.openstack.org/trunk/openstack-ops/content/>



# Découpage réseau

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Découpage réseau

2014-08-11

- Management network : réseau d'administration
- Data network : réseau pour la communication inter instances
- External network : réseau externe, dans l'infrastructure réseau existante
- API network : réseau contenant les endpoints API



# Considérations liées à la sécurité

## Formation OpenStack

### Déployer OpenStack

- └ Bonnes pratiques pour un déploiement en production
  - └ Considérations liées à la sécurité

2014-08-11

- Indispensable : HTTPS sur l'accès des APIs à l'extérieur
- Sécurisation des communications MySQL et RabbitMQ
- Un accès MySQL par base et par service
- Un utilisateur Keystone par service
- Limiter l'accès en lecture des fichiers de configuration (mots de passe, token)

- Indispensable : HTTPS sur l'accès des APIs à l'extérieur
- Sécurisation des communications MySQL et RabbitMQ
- Un accès MySQL par base et par service
- Un utilisateur Keystone par service
- Limiter l'accès en lecture des fichiers de configuration (mots de passe, token)



# Segmenter son cloud

- Host aggregates : machines physiques avec des caractéristiques similaires
- Availability zones : machines dépendantes d'une même source électrique, d'un même switch, d'un même DC, etc.
- Regions : chaque région a son API
- Cells : permet de regrouper plusieurs cloud différents sous une même API

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Segmenter son cloud

2014-08-11

- Host aggregates : machines physiques avec des caractéristiques similaires
- Availability zones : machines dépendantes d'une même source électrique, d'un même switch, d'un même DC, etc.
- Regions : chaque région a son API
- Cells : permet de regrouper plusieurs cloud différents sous une même API



# Host aggregates / agrégats d'hôtes

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

##### Host aggregates / agrégats d'hôtes

2014-08-11

- L'administrateur définit des agrégats via l'API
- 1 agrégat ≡ 1 point commun, ex : GPU
- L'utilisateur peut choisir un agrégat à la création d'instance



# Availability zones / zones de disponibilité

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Availability zones / zones de disponibilité

2014-08-11

- Groupe d'hôtes
- Découpage en termes de disponibilité : Rack, Datacenter, etc.
- L'utilisateur peut choisir une zone de disponibilité
- L'utilisateur peut demander à ce que des instances soient démarrées dans une même zone, ou au contraire dans des zones différentes

- Groupe d'hôtes
- Découpage en termes de disponibilité : Rack, Datacenter, etc.
- L'utilisateur peut choisir une zone de disponibilité
- L'utilisateur peut demander à ce que des instances soient démarrées dans une même zone, ou au contraire dans des zones différentes



- Équivalent des régions d'AWS
- Un service peut avoir différents endpoints dans différentes régions
- Chaque région est autonome
- Cas d'usage : cloud de grande ampleur (comme certains clouds publics)

# Régions

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

- Régions

2014-08-11



# Cellules / Cells

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

##### Cellules / Cells

2014-08-11

- Fonctionnalité de Nova uniquement
- Un seul nova-api devant plusieurs cellules
- Chaque cellule a sa propre BDD et file de messages
- Ajoute un niveau de scheduling (choix de la cellule)



2014-08-11

- Le packaging est fait dans de multiples distributions, RPM, DEB et autres



2014-08-11

- Le packaging est fait dans de multiples distributions, RPM, DEB et autres
- Ubuntu est historiquement la plateforme de référence pour le développement d'OpenStack
- Le packaging dans Ubuntu suit de près le développement d'OpenStack, et des tests automatisés sont réalisés
- Canonical fournit la Ubuntu Cloud Archive, qui met à disposition la dernière version d'OpenStack pour la dernière Ubuntu LTS

- Le packaging est fait dans de multiples distributions, RPM, DEB et autres
- Ubuntu est historiquement la plateforme de référence pour le développement d'OpenStack
- Le packaging dans Ubuntu suit de près le développement d'OpenStack, et des tests automatisés sont réalisés
- Canonical fournit la Ubuntu Cloud Archive, qui met à disposition la dernière version d'OpenStack pour la dernière Ubuntu LTS



# OpenS

LTS support 5 years, non LTS support 18 mon

12.04      12.10      13.04      13.10      14.04      14.10      15.04      15

Ubuntu 12.04 LTS

ESSEX

FOLSOM

21 mo

GRIZZLY

15 mo

HAVANA

9 mo

2014-08-11

- OpenStack est intégré dans les dépôts officiels de Debian
- Red Hat propose RHOS
- Comme Ubuntu, le cycle de release de Fedora est synchronisé avec celui d'OpenStack

- OpenStack est intégré dans les dépôts officiels de Debian
- Red Hat propose RHOS
- Comme Ubuntu, le cycle de release de Fedora est synchronisé avec celui d'OpenStack



# Les distributions OpenStack

Formation OpenStack

└ Déployer OpenStack

  └ Bonnes pratiques pour un déploiement en production

    └ Les distributions OpenStack

2014-08-11

- StackOps
- Mirantis
- HP
- etc.

- StackOps
- Mirantis
- HP
- etc.



# Déploiement bare metal

- Le déploiement des hôtes physiques OpenStack peut se faire à l'aide d'outils dédiés

Formation OpenStack

└ Déployer OpenStack

  └ Bonnes pratiques pour un déploiement en production

    └ Déploiement bare metal

2014-08-11

Déploiement bare metal

● Le déploiement des hôtes physiques OpenStack peut se faire à l'aide d'outils dédiés



2014-08-11

- Le déploiement des hôtes physiques OpenStack peut se faire à l'aide d'outils dédiés
- Canonical/Ubuntu propose MaaS : Metal as a Service
- Dell propose Crowbar
- eDeploy (eNovance)

- Le déploiement des hôtes physiques OpenStack peut se faire à l'aide d'outils dédiés
- Canonical/Ubuntu propose MaaS : Metal as a Service
- Dell propose Crowbar
- eDeploy (eNovance)



# Gestion de configuration

Formation OpenStack

  └ Déployer OpenStack

    └ Bonnes pratiques pour un déploiement en production

      └ Gestion de configuration

2014-08-11

- Puppet, Chef, CFEngine, Saltstack, Ansible, etc.



# Gestion de configuration

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

##### Gestion de configuration

2014-08-11

- Puppet, Chef, CFEngine, Saltstack, Ansible, etc.
- Ces outils peuvent aider à déployer le cloud OpenStack
- ... mais aussi à gérer les instances (section suivante)

# Modules Puppet

## Formation OpenStack

### Déployer OpenStack

#### Bonnes pratiques pour un déploiement en production

##### Modules Puppet

2014-08-11

- PuppetLabs maintient (avec d'autres) des modules pour déployer OpenStack
- <https://forge.puppetlabs.com/puppetlabs/openstack>

- PuppetLabs maintient (avec d'autres) des modules pour déployer OpenStack
- <https://forge.puppetlabs.com/puppetlabs/openstack>





# Formation OpenStack

## Déployer OpenStack

### Faire face aux problèmes

#### Plan

2014-08-11

# Plan

## 1 Virtualisation

## 2 Le Cloud : vue d'ensemble

## 3 OpenStack : projet, logiciel et utilisation

## 4 Déployer OpenStack

- Les briques nécessaires
- Keystone : Authentification, autorisation et catalogue de services
- Nova : Compute
- Glance : Registre d'images
- Neutron : Réseau en tant que service
- Cinder : Stockage block
- Horizon : Dashboard web
- Swift : Stockage objet
- Ceilometer : Collecte de métriques
- Heat : Orchestration des ressources
- Trove : Database as a Service
- Designate : DNS as a Service
- Quelques autres composants intéressants
- Bonnes pratiques pour un déploiement en production
- Faire face aux problèmes

## 5 Tirer partie de l'IaaS

# Les réflexes en cas d'erreur ou de comportement erroné

- Travaille-t-on sur le bon tenant ?
- Est-ce que l'API renvoie une erreur ? (le dashboard peut cacher certaines informations)
- Si nécessaire d'aller plus loin :
  - ▶ Regarder les logs sur le cloud controller (/var/log/<composant>/\*.log)
  - ▶ Regarder les logs sur le compute node et le network node si le problème est spécifique réseau/instance
  - ▶ Éventuellement modifier la verbosité des logs dans la configuration

## Formation OpenStack

### Déployer OpenStack

#### Faire face aux problèmes

##### Les réflexes en cas d'erreur ou de comportement erroné

2014-08-11

- Travaille-t-on sur le bon tenant ?
- Est-ce que l'API renvoie une erreur ? (le dashboard peut cacher certaines informations)
- Si nécessaire d'aller plus loin :
  - ▶ Regarder les logs sur le cloud controller (/var/log/<composant>/\*.log)
  - ▶ Regarder les logs sur le compute node et le network node si le problème est spécifique réseau/instance
  - ▶ Éventuellement modifier la verbosité des logs dans la configuration



# Est-ce un bug ?

Formation OpenStack

└ Déployer OpenStack

  └ Faire face aux problèmes

    └ Est-ce un bug ?

2014-08-11

- Si le client CLI crash, c'est un bug



- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug

# Est-ce un bug ?

## Formation OpenStack

### Déployer OpenStack

#### Faire face aux problèmes

##### Est-ce un bug ?

2014-08-11

- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug



# Est-ce un bug ?

- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug
- Si les logs montrent une stacktrace Python, c'est un bug

## Formation OpenStack

### Déployer OpenStack

#### Faire face aux problèmes

##### Est-ce un bug ?

2014-08-11

- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug
- Si les logs montrent une stacktrace Python, c'est un bug



# Est-ce un bug ?

- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug
- Si les logs montrent une stacktrace Python, c'est un bug
- Sinon, à vous d'en juger

## Formation OpenStack

### Déployer OpenStack

#### Faire face aux problèmes

##### Est-ce un bug ?

2014-08-11

- Si le client CLI crash, c'est un bug
- Si le dashboard web affiche une erreur 500, c'est peut-être un bug
- Si les logs montrent une stacktrace Python, c'est un bug
- Sinon, à vous d'en juger



# Plan

1 Virtualisation

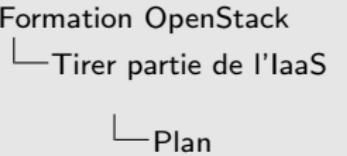
2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

5 Tirer partie de l'IaaS

- Côté applications
- Côté système



2014-08-11



# Deux visions

Formation OpenStack  
└ Tirer partie de l'IaaS

2014-08-11

└ Deux visions

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
  - Faire évoluer ses pratiquer, tant côté applicatif que système
- "Pets vs Cattle"

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
- Faire évoluer ses pratiquer, tant côté applicatif que système

"Pets vs Cattle"



# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

5 Tirer partie de l'IaaS

- Côté applications
- Côté système

Formation OpenStack  
└ Tirer partie de l'IaaS  
  └ Côté applications  
    └ Plan

2014-08-11



# Adapter ou faire ses applications "cloud ready"

## Formation OpenStack

└ Tirer partie de l'IaaS

  └ Côté applications

    └ Adapter ou faire ses applications "cloud ready"

2014-08-11

- Stateless : permet de multiplier les routes d'accès à l'application
- Ne pas stocker les données en local, mais plutôt :
  - ▶ Base de données
  - ▶ Stockage objet
- Utiliser des outils standards de journalisation

- Stateless : permet de multiplier les routes d'accès à l'application
- Ne pas stocker les données en local, mais plutôt :
  - ▶ Base de données
  - ▶ Stockage objet
- Utiliser des outils standards de journalisation



2014-08-11

# Plan

1 Virtualisation

2 Le Cloud : vue d'ensemble

3 OpenStack : projet, logiciel et utilisation

4 Déployer OpenStack

5 Tirer partie de l'IaaS

● Côté applications

● Côté système



# Adopter une philosophie DevOps

- Infrastructure as Code
- Scale out au lieu de scale up
- HA niveau application plutôt qu'infrastructure

## Formation OpenStack

└ Tirer partie de l'IaaS

  └ Côté système

    └ Adopter une philosophie DevOps

2014-08-11

- Infrastructure as Code
- Scale out au lieu de scale up
- HA niveau application plutôt qu'infrastructure



# Utiliser des images cloud

## Formation OpenStack

└ Tirer partie de l'IaaS

  └ Côté système

    └ Utiliser des images cloud

2014-08-11

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)

La plupart des distributions fournissent aujourd'hui 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)

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>

Formation OpenStack  
└ Tirer partie de l'IaaS  
  └ Côté système  
    └ Cirros

2014-08-11

- Cirros est l'image cloud par excellence
- OS minimaliste
- Contient cloud-init
- <https://launchpad.net/cirros>



# Faire ou modifier une image cloud

## Formation OpenStack

└ Tirer partie de l'IaaS

  └ Côté système

    └ Faire ou modifier une image cloud

• Utilisation de libguestfs

2014-08-11

- Utilisation de libguestfs



- Cloud-init est un moyen de tirer partie 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

## Formation OpenStack

- └ Tirer partie de l'IaaS
  - └ Côté système
    - └ Exemple cloud-config

2014-08-11

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

## Exemple cloud-config

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

# Configurer et orchestrer ses instances

## Formation OpenStack

└ Tirer partie de l'IaaS

  └ Côté système

    └ Configurer et orchestrer ses instances

2014-08-11

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

- 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

# Conclusion

## Formation OpenStack

### └ Conclusion

#### └ Conclusion

2014-08-11



## Références

- [http://www.antoinebenkemoun.fr/data/PPT\\_AC.pdf](http://www.antoinebenkemoun.fr/data/PPT_AC.pdf)

