

Cloud Computing

Guillaume Urvoy-Keller

August 24, 2018

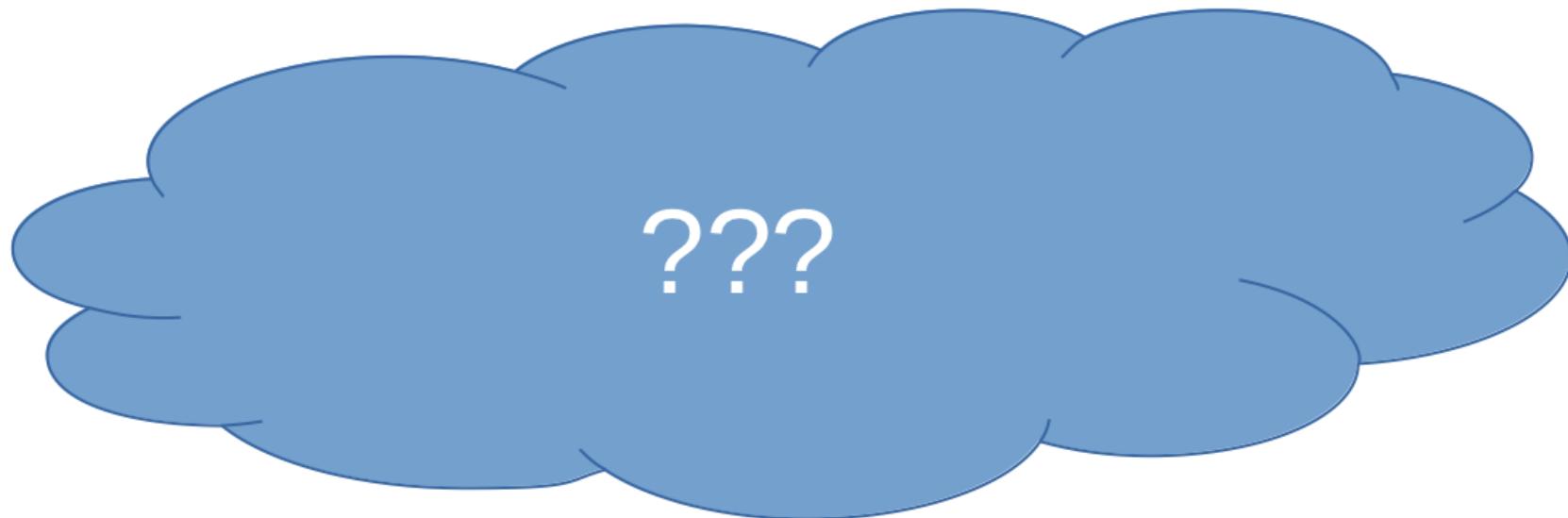
Outline

1 PaaS, SaaS, IaaS

2 Cloud Privé et Hybride

3 Data Center

Qu'est-ce que le Cloud pour vous?



Cloud Public

Utiliser un service dont on ne gère pas le matériel (= serveurs physiques)

- Un opérateur (Google Cloud, Amazon Web Services, OVH, etc.) gère le matériel et multiplexe les utilisateurs → hyperviseur (ex: VMware, KVM, VirtualBox) sur machine physique et une/plusieurs machines virtuelles par client

Cloud Public

Utiliser un service dont on ne gère pas le matériel (= serveurs physiques)

- Un opérateur (Google Cloud, Amazon Web Services, OVH, etc.) gère le matériel et multiplexe les utilisateurs → hyperviseur (ex: VMware, KVM, VirtualBox) sur machine physique et une/plusieurs machines virtuelles par client
- Paiement en fonction de la quantité (CPU, RAM, Réseau) consommée

Cloud Public

Utiliser un service dont on ne gère pas le matériel (= serveurs physiques)

- Un opérateur (Google Cloud, Amazon Web Services, OVH, etc.) gère le matériel et multiplexe les utilisateurs → hyperviseur (ex: VMware, KVM, VirtualBox) sur machine physique et une/plusieurs machines virtuelles par client
- Paiement en fonction de la quantité (CPU, RAM, Réseau) consommée
- Passage à l'échelle facile : création rapide de nouvelle machine virtuelle à la demande

Cloud Public

Utiliser un service dont on ne gère pas le matériel (= serveurs physiques)

- Un opérateur (Google Cloud, Amazon Web Services, OVH, etc.) gère le matériel et multiplexe les utilisateurs → hyperviseur (ex: VMware, KVM, VirtualBox) sur machine physique et une/plusieurs machines virtuelles par client
- Paiement en fonction de la quantité (CPU, RAM, Réseau) consommée
- Passage à l'échelle facile : création rapide de nouvelle machine virtuelle à la demande
- Accès à distance

SaaS, IaaS, PaaS

Software/Infrastructure/Platform as a Service

SaaS : Software as a Service

- Accès à un logiciel via interface Web
- Pas de gestion des mises-à-jour
- Interaction possible avec une API ou installation d'un client

Ex : Google Mail, Dropbox.

SaaS : API Google Mail

The screenshot shows the official Google Developers page for the Gmail API. At the top, there's a navigation bar with links for HOME, GUIDES, REFERENCE, SAMPLES, SUPPORT, and SEND FEEDBACK. Below the navigation is a main heading "Flexible, RESTful access to the user's inbox". To the left, there's a section titled "Gmail API overview" featuring an image of three devices (a smartphone, a tablet, and a desktop screen) displaying the Gmail interface. To the right, there's a "Quickstart guides" section with a video thumbnail titled "Revolutionizing Email Access with the Gmail API" and a portrait of a smiling man. At the bottom, there are links for ANDROID, iOS, JAVASCRIPT, and APPS SCRIPT, along with a set of small navigation icons.

Modern, easy, fast, RESTful.

HOME GUIDES REFERENCE SAMPLES SUPPORT SEND FEEDBACK

Flexible, RESTful access to the user's inbox

Gmail API overview

Read and send messages, work with labels, and search for specific threads.

OVERVIEW

LAUNCHPAD online

Revolutionizing Email Access with the Gmail API

Quickstart guides

Create a simple app that makes requests to the Gmail API in just 5 to 10 minutes.

ANDROID iOS JAVASCRIPT APPS SCRIPT

SaaS : API Google Mail



Users.messages

For Users.messages Resource details, see the [resource representation](#) page.

Method	HTTP request	Description
URIs relative to https://www.googleapis.com/gmail/v1/users , unless otherwise noted		
delete	<code>DELETE /userId/messages/<i>id</i></code>	Immediately and permanently deletes the specified message. This operation cannot be undone. Prefer <code>messages.trash</code> instead.
get	<code>GET /userId/messages/<i>id</i></code>	Gets the specified message.
insert	<code>POST https://www.googleapis.com/upload/gmail/v1/users/<i>userId</i>/messages</code> and <code>POST /userId/messages</code>	Directly inserts a message into only this user's mailbox similar to <code>IMAP APPEND</code> , bypassing most scanning and classification. Does not send a message.
list	<code>GET /userId/messages</code>	Lists the messages in the user's mailbox.
modify	<code>POST /userId/messages/<i>id</i>/modify</code>	Modifies the labels on the

IaaS : Infrastructure as a Service

- Location d'une machine virtuelle
- Contrôle complet de la machine : déploiement arbitraire d'applications
- Création des VMs, des réseaux au travers d'une interface Web ou via API

IaaS : Elastic Cloud Computing (EC2) d'Amazon Web Services (AWS)

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar navigation bar includes links for EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES (with Instances selected), Launch Templates, Spot Requests, Reserved Instances, Dedicated Hosts, Scheduled Instances, IMAGES (with AMIs selected), Bundle Tasks, ELASTIC BLOCK STORE (with Volumes, Snapshots, Lifecycle Manager selected), and NETWORK & SECURITY (with Security Groups, Elastic IPs selected). The main content area displays a table of instances. A search bar at the top of the table results in one result: i-06cfa61a01db47178. The instance details show it is a t2.micro type in us-east-1a, currently running. Below the table, the instance's public IP (54.227.179.201) is listed, along with its configuration details such as Instance ID, Instance state, Instance type, Availability zone, Security groups, Scheduled events, Platform, IAM role, and Kev pair name. The Public DNS (IPv4) is listed as -.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-06cfa61a01db47178	t2.micro	us-east-1a	running	Initializing	None	-

Instance: i-06cfa61a01db47178 Public IP: 54.227.179.201

Description	Status Checks	Monitoring	Tags
Instance ID: i-06cfa61a01db47178			
Instance state: running			
Instance type: t2.micro			
Elastic IPs:			
Availability zone: us-east-1a			
Security groups: launch-wizard-2 . view inbound rules . view outbound rules			
Scheduled events: No scheduled events			
AMI ID: amzn2-ami-hvm-2.0.20180810-x86_64-gp2 (ami-04681a1dbd79675a5)			
Platform: -			
IAM role: -			
Kev pair name: Virginia			
Public DNS (IPv4): -			
IPv4 Public IP: 54.227.179.201			
IPv6 IPs: -			
Private DNS: ip-172-30-0-45.ec2.internal			
Private IPs: 172.30.0.45			
Secondary private IPs: -			
VPC ID: vpc-794a4a01			
Subnet ID: subnet-028fa249			
Network interfaces: eth0			
Source/dest. check: True			
T2/T3 Unlimited: Disabled			

IaaS : accès API unix à EC2

```
mkdir "$IPSEC_CLIENT_FOLDER" > /dev/null
fi
if [ ! -d "$OPENVPN_SERVER_FOLDER" ]; then
  mkdir "$OPENVPN_SERVER_FOLDER" > /dev/null
fi
if [ ! -d "$IPSEC_SERVER_FOLDER" ]; then
  mkdir "$IPSEC_SERVER_FOLDER" > /dev/null
fi
echo -e '*****'
echo '* Provisioning: Amazon AWS'
echo -e '*****\n'

### Configure Network on Amazon
echo -e "Configuring AWS\n"
# Create new Virtual Private Cloud (VPC)

VPC_ID=$(aws ec2 create-vpc --cidr-block $AWS_IP_ADDR | jq -r '.Vpc.VpcId')
if [ -z "$VPC_ID" ]; then
  exit 1 # Error VPC limit exceeded
fi
echo -e "New VPC $VPC_ID\n"
echo "$VPC_ID" >> "$VPC_INDEX"

# Create new subnet of the VPC
SUBNET_ID=$(aws ec2 create-subnet --vpc-id "$VPC_ID" --cidr-block $AWS_VPC_SUBNET_IP_ADDR | jq -r '.Subnet.SubnetId')
echo -e "New Subnet $SUBNET_ID\n"
echo "$SUBNET_ID" >> "$SUBN_INDEX"

# Create new Internet Gateways
IGW_ID=$(aws ec2 create-internet-gateway | jq -r '.InternetGateway.InternetGatewayId')
if [ -z "$IGW_ID" ]; then
  exit 2 # Error gateways limit exceeded
fi
echo -e "New Internet Gateway $IGW_ID\n"
echo "$IGW_ID" >> "$IGW_INDEX"
aws ec2 attach-internet-gateway --vpc-id "$VPC_ID" --internet-gateway-id "$IGW_ID" > /dev/null
```

PaaS : Platform as a Service

- Vous gérez le code de l'application
- Vous poussez vers le code vers le service géré par le gestionnaire de PaaS
- Le gestionnaire de PaaS instancie votre application sur ses serveurs.
- Exemples : Elastic Beantalk d'AWS, Heroku (au dessus d'EC2), Windows Azure

PaaS : Heroku



Download Heroku Toolkit for Mac OS X ▾

```
$ heroku login
...
$ git clone https://github.com/heroku/java-getting-started.git
$ cd java-getting-started

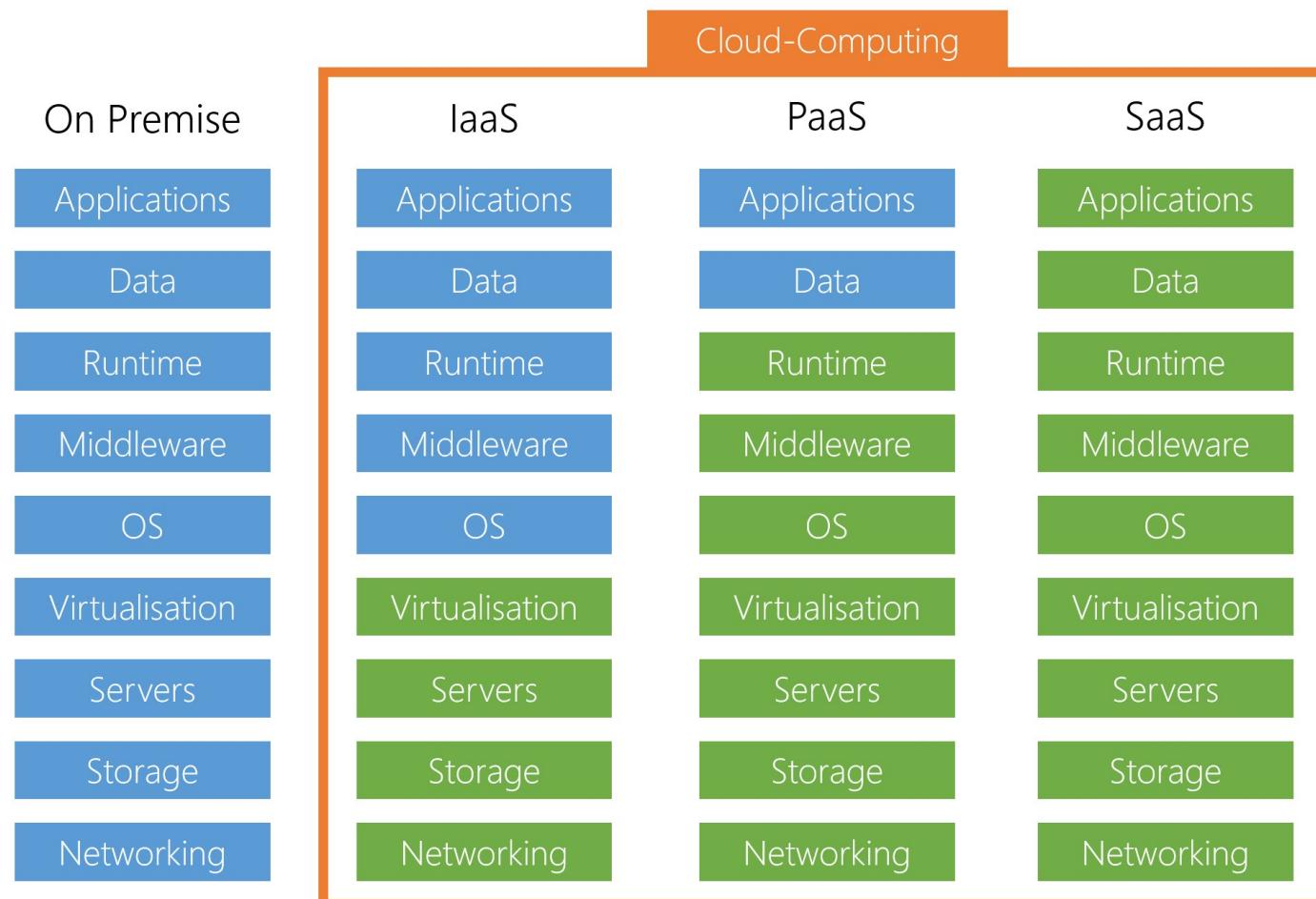
$ heroku create
Creating warm-eyrie-9006... done, stack is cedar-14
http://warm-eyrie-9006.herokuapp.com/ | git@heroku.com:warm-eyrie-9006.git
Git remote heroku added

$ git push heroku master
...
http://warm-eyrie-9006.herokuapp.com/ deployed to Heroku

$ heroku ps:scale web=1
```

IaaS vs. SaaS vs. PaaS

- En **vert** : géré par un vendeur
- En **bleu** : géré par vous



Cloud Privé, Hybride

- Ré-utilisation des techniques développées pour le cloud public dans un usage privé
- Certaines entreprises veulent la souplesse du cloud mais garder le contrôle de leurs **données**
- Cela signifie : traiter les resources informatiques comme un *pool* (ensemble) unique et les partager entre les groupes internes à l'entreprise
 - Possibilité de créer des VMs à la demande
 - PaaS interne, ex: Openshift

Exemple de PaaS privé : Openshift

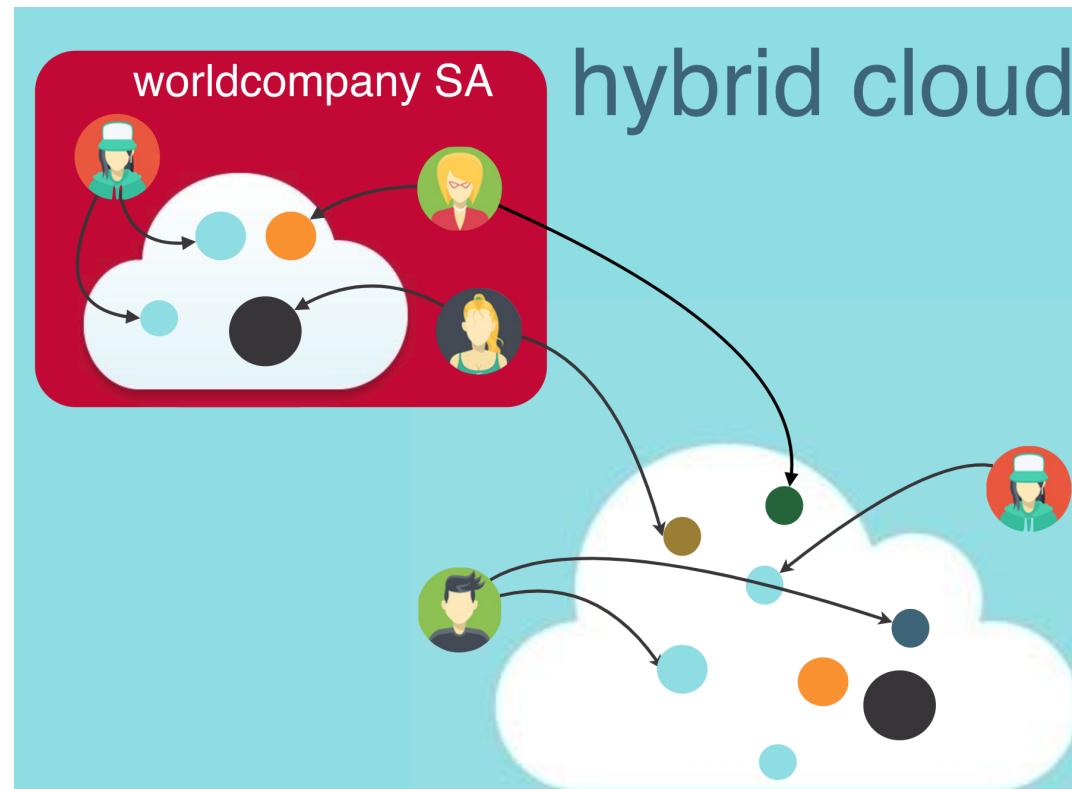
The screenshot shows a web browser displaying the Red Hat OpenShift website at <https://www.openshift.com/learn/what-is-openshift/>. The page has a red background with white text. At the top, there is a navigation bar with links for PRODUCTS, LEARN, COMMUNITY, SUPPORT, FREE TRIAL, and SIGN IN. The main heading is "What is OpenShift". Below it, a subtext states: "Red Hat® OpenShift® is a comprehensive enterprise-grade application platform, built for containers with Kubernetes." A button labeled "WATCH 2 MIN OVERVIEW" is visible. The overall design is clean and professional.

Build, deploy, and scale

Automate the build, deployment, and management of applications so that you can focus on writing the code for your next big idea.

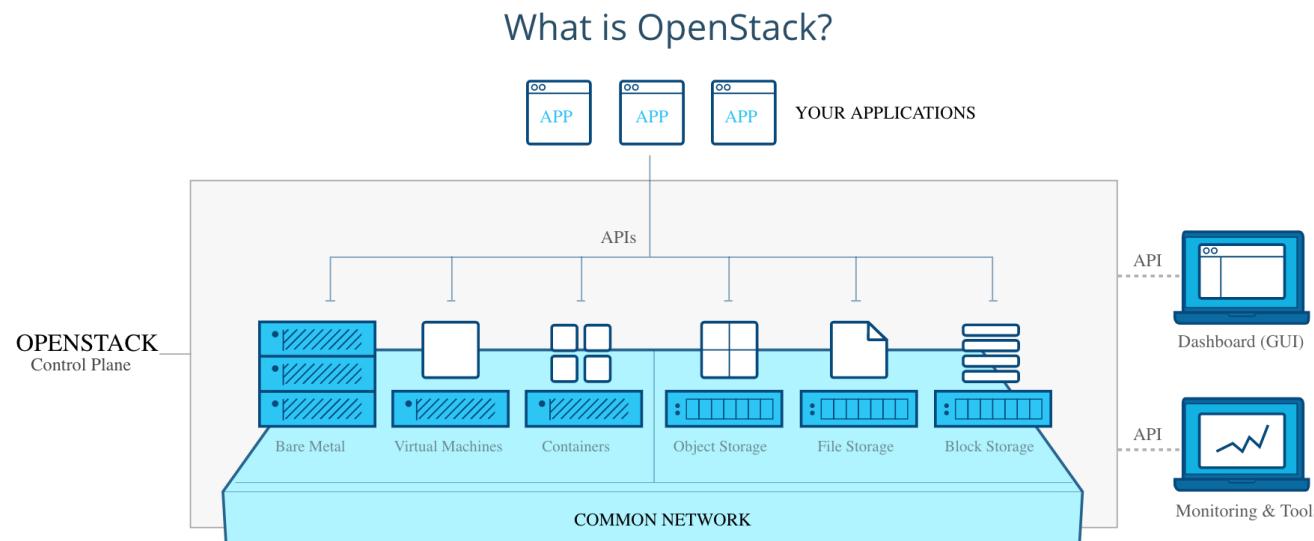
Hybrid Cloud

- Notion de Cloud Hybride : utilisation simultanée de ressources locales (serveurs physiques de l'entreprise) et de resources dans un ou plusieurs cloud publics
- Exemple typique d'utilisation : création de resources dans le cloud



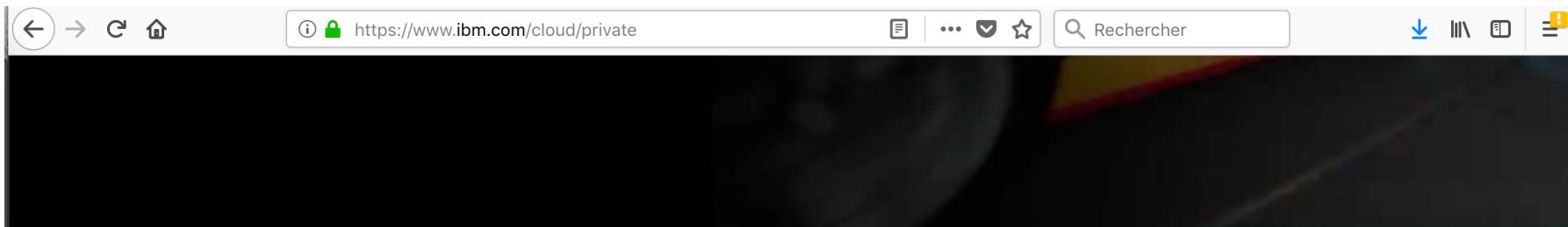
Gestionnaire du cloud privé et ou hybride

- Suite logicielle de gestion de cloud privé : Openstack



- De nombreuses sociétés de services proposent des services de gestion de cloud privé et hybride : IBM, Nutanix

Service *private cloud* d'IBM



Why IBM Cloud Private

Extend your investments and build for the future with IBM Cloud™ Private.

IBM Cloud Private enables you to sensibly manage enterprise workloads with the benefit of extra controls for security. A Kubernetes-based container platform, IBM Cloud Private can help you quickly move, modernize and automate workloads or build new cloud-native applications. Development and deployment take place on your own infrastructure and in your data center, mitigating risk and keeping you out of the headlines.

Modernize traditional applications, boosting scalability and resilience. The IBM Cloud Private catalog includes containerized IBM middleware that's ready to deploy into the cloud. Containerization erases concerns about application-specific breakage points when modernizing monolithic, heritage applications. This enables you to reduce downtime by resolving a single issue without taking down the entire system. IBM also provides capabilities like Transformation Advisor and expert advice from IBM Cloud Garage Services to help you isolate application interdependencies and facilitate the modernization process.

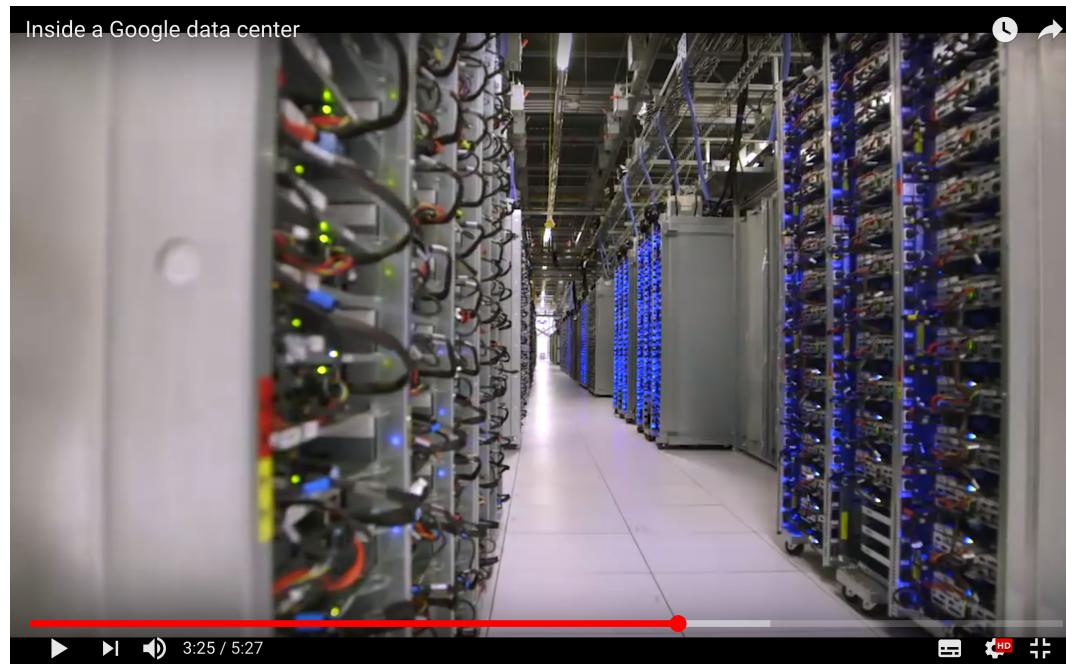


IBM Cloud Private fills the missing link in the Application Modernization journey

Let's talk

Data Center

- Le cœur des clouds publics, mais également des clouds privés
- Des milliers de machines dans le cas de Google ou Amazon



- Quelques racks pour une entreprise privée.
Warning Ce n'est pas parce que c'est plus petit que ce n'est pas puissant. Un DC typique géré par Nutanix pour un client va avoir 6 serveurs, chacun avec 300 Go de RAM et beaucoup de processeurs

Data Centers : organisation interne

- Serveurs organisés en rack
- Switch haut débit sert un rack de 40-80 serveurs physiques → ToR (Top of the Rack) switch
- Chaque serveur physique va héberger quelques dizaines de machines virtuelles.
- Racks interconnectés par réseau haut débit
- Architecturation du réseau est important pour les performances

Data Centers : organisation interne

- Combien de machines virtuelles par serveur physique ?
1 serveur avec :
 - P processeurs
 - C coeurs par processeur
 - Technique d'hyperthreading : technique (implémentée sur les processeurs) permettant de faire tourner 2 processus simultanément
$$P * C * 2 \text{ processus pouvant s'exécuter simultanément} \rightarrow \text{PxCx2 machines virtuelles}$$
- Au total : 1 rack de 40 serveurs physiques quadri-processeurs quadri-cores $40 * 4 * 4 * 2 = 1280 \text{ VM}$ par rack \rightarrow gros besoin réseau

Data Center : quelles applications?

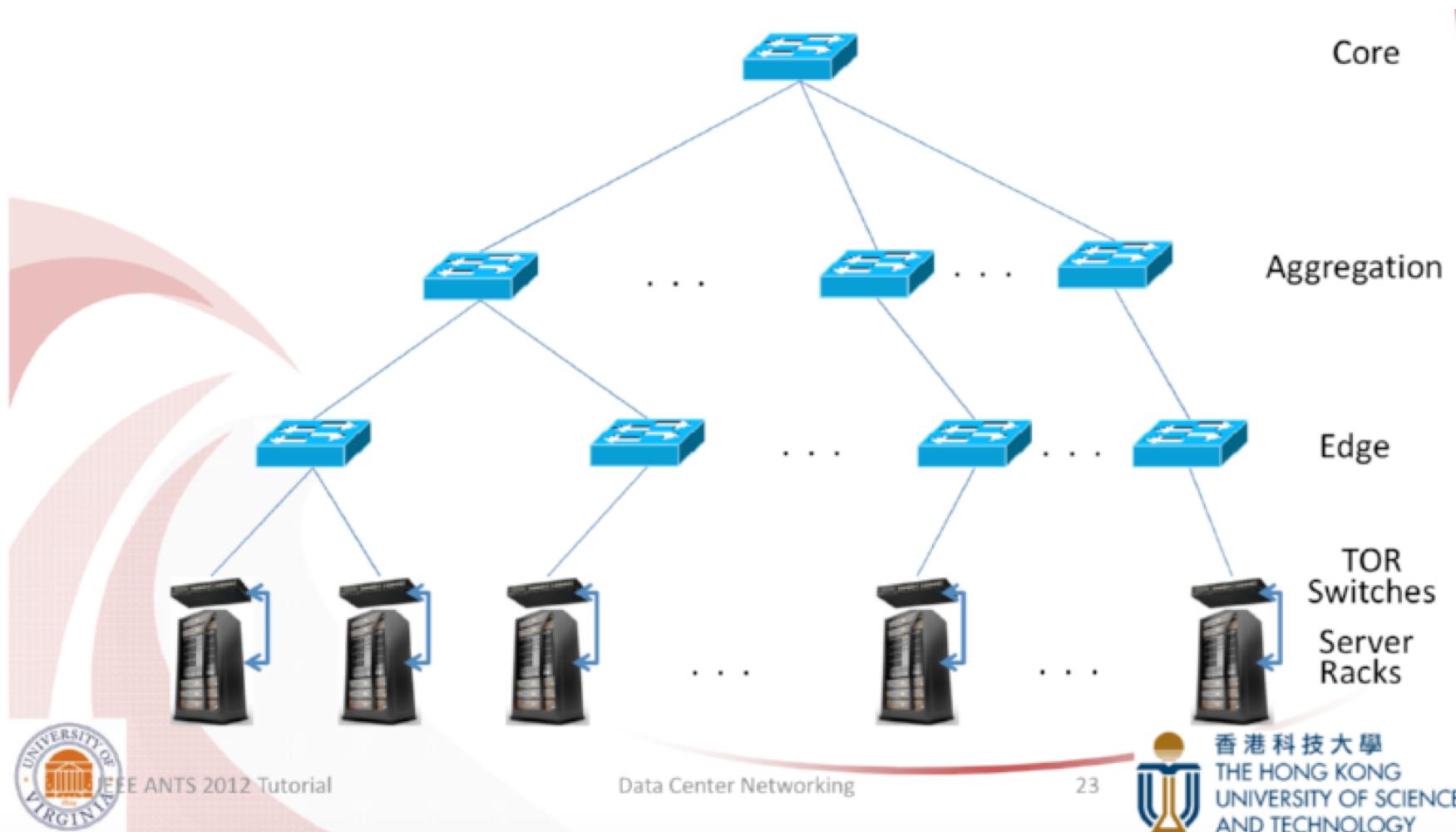
- DC privé
 - les services de l'entreprises (Mail, DNS, LDAP, etc)
 - les services offerts aux utilisateurs, par ex. Openstack pour créer des VMs pour les développeurs.
- DC public : services offerts aux utilisateurs
 - Ex typique : service Web Gmail
 - Attention, une simple recherche par mot-clef chez Google est envoyée sur des clusters de quelques milliers de CPU
- DC public ou privé : Applications de type Big Data
 - Grande masse de données réparties sur les serveurs
 - Tâches de calcul envoyées vers les données ...
 - car trop massives pour être envoyées à un seul ordinateur qui ferait tous les calculs

Besoin Réseaux

Typiquement, même débit entre 2 serveurs (ou VM) quelle que soit leur position !!! → très forte contrainte !!!

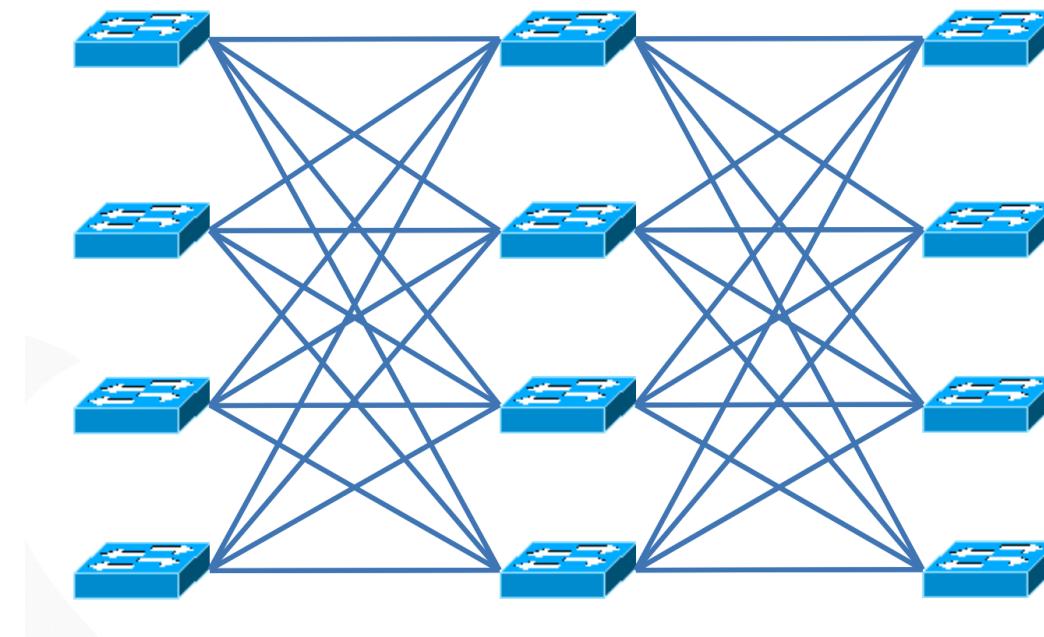
Data centers : réseau d'interconnexion

Ce qui ne marche pas : un arbre car nombre de sauts (et donc débit) entre 2 serveurs dépend de la position



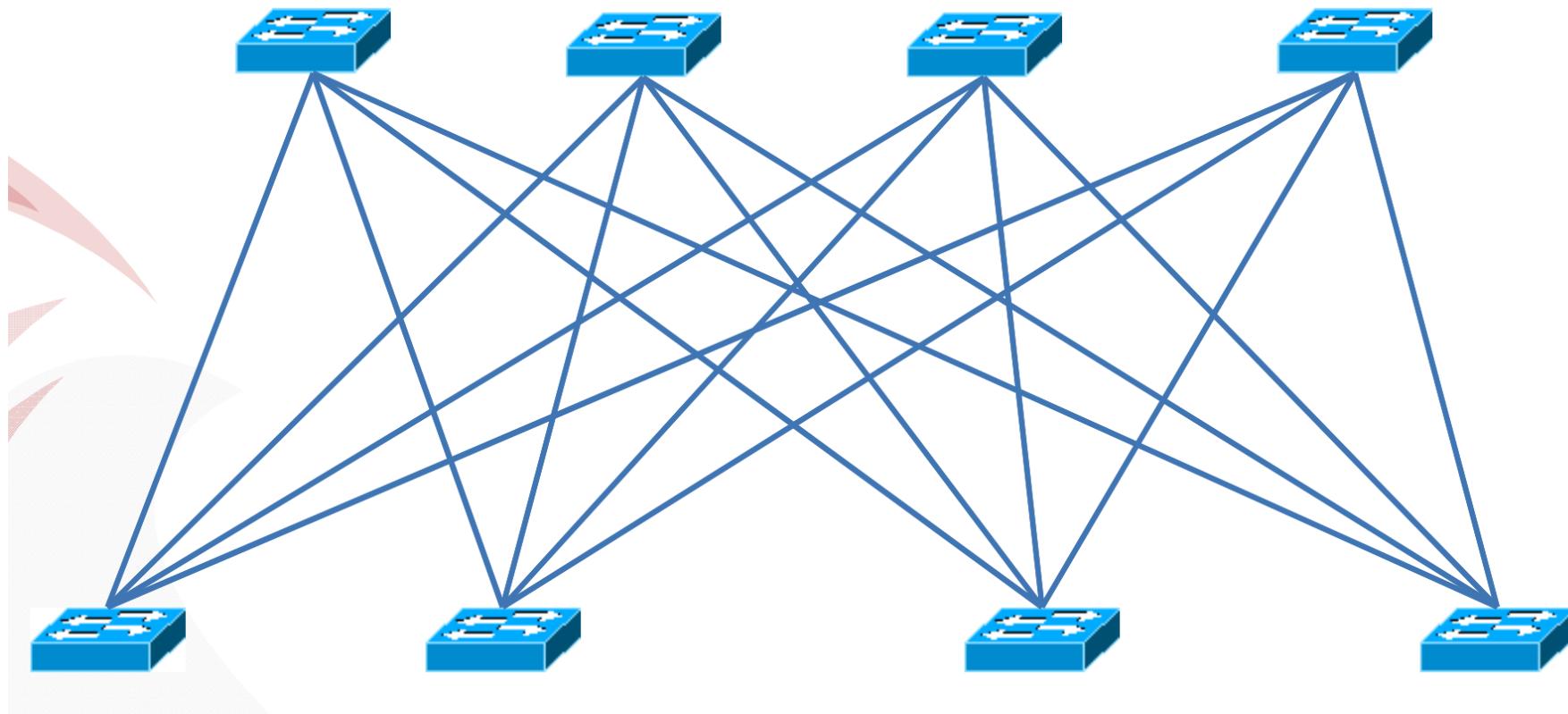
Data centers : réseau d'interconnexion

- Architecture de Clos
- Charles Clos, 1953
- Pour les réseaux à commutation de circuits (voix) à cette époque
- Des chemins multiples de même longueur entre chaque couple de serveurs
- Ex : réseau à 3 étages



Data centers : réseau d'interconnexion

Variante fréquente : leaf and spine (feuille et dorsale). Recommandée par CISCO, utilisée par Google



Data centers : réseau d'interconnexion - L'exemple de Google

Datacenter Generation	First Deployed	Merchant Silicon	ToR Config	Aggregation Block Config	Spine Block Config	Fabric Speed	Host Speed	Bisection BW
Four-Post CRs	2004	vendor	48x1G	-	-	10G	1G	2T
Firehose 1.0	2005	8x10G 4x10G (ToR)	2x10G up 24x1G down	2x32x10G (B)	32x10G (NB)	10G	1G	10T
Firehose 1.1	2006	8x10G	4x10G up 48x1G down	64x10G (B)	32x10G (NB)	10G	1G	10T
Watchtower	2008	16x10G	4x10G up 48x1G down	4x128x10G (NB)	128x10G (NB)	10G	nx1G	82T
Saturn	2009	24x10G	24x10G	4x288x10G (NB)	288x10G (NB)	10G	nx10G	207T
Jupiter	2012	16x40G	16x40G	8x128x40G (B)	128x40G (NB)	10/40G	nx10G/ nx40G	1.3P

Table 2: Multiple generations of datacenter networks. (B) indicates blocking, (NB) indicates Nonblocking.

Data centers : réseau d'interconnexion - L'exemple de Google

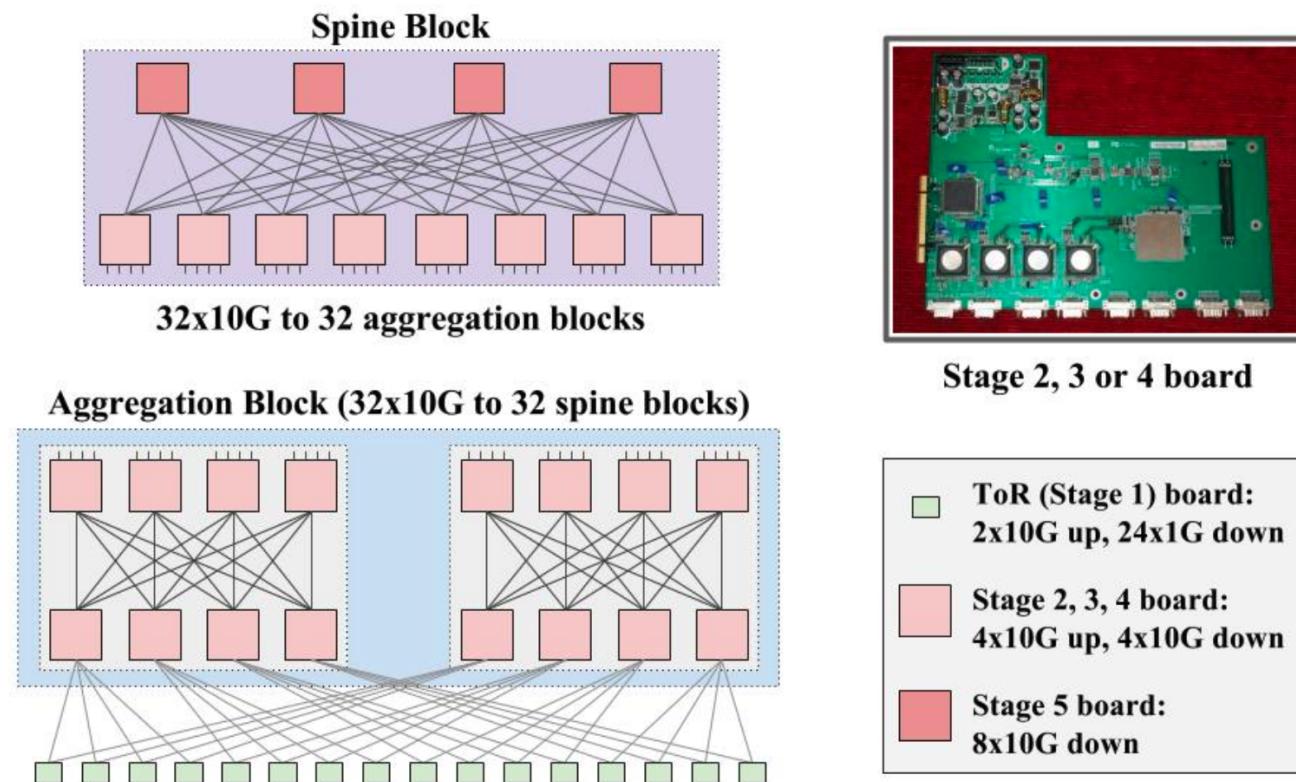


Figure 5: Firehose 1.0 topology. Top right shows a sample 8x10G port fabric board in Firehose 1.0, which formed Stages 2, 3 or 4 of the topology.

Introduction à la virtualisation système

G. Urvoy-Keller

LP

Les 2 types de virtualisation

➤ Virtualisation lourde

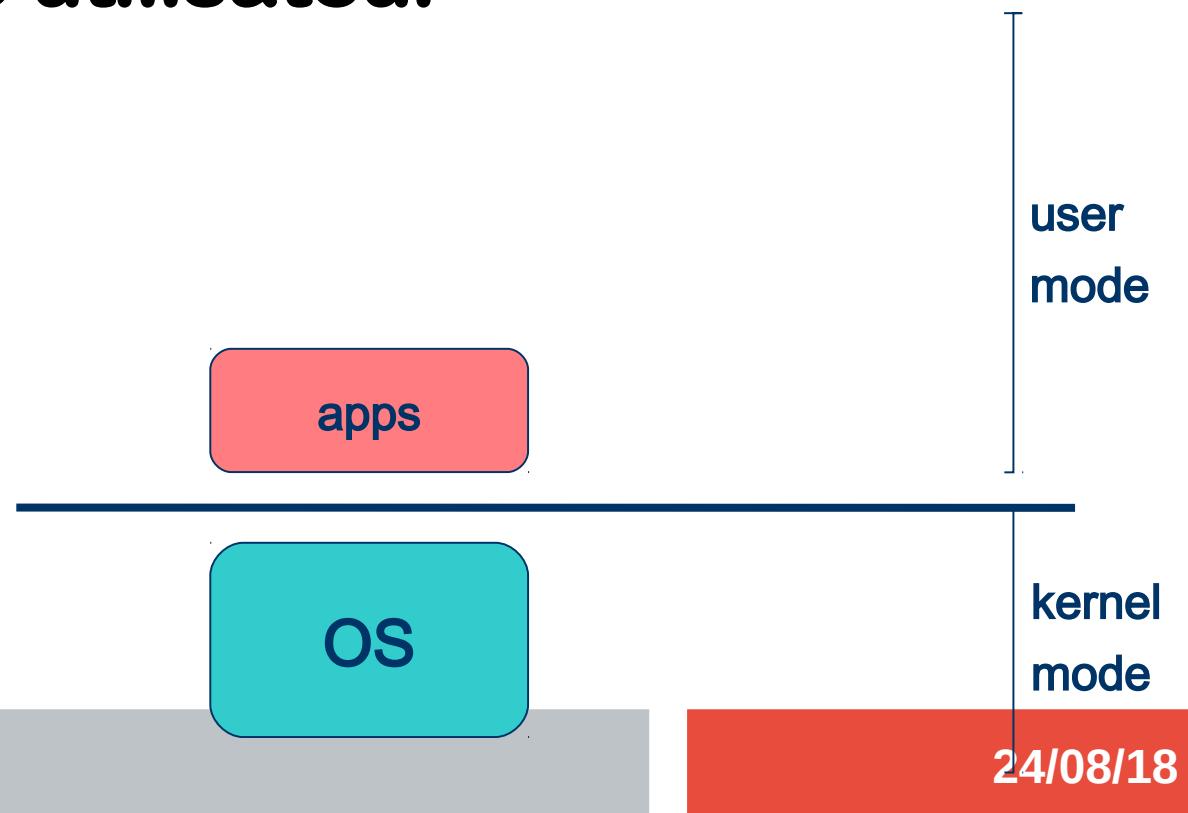
- Ce que vous faites avec virtualbox en salle de TP ;-)
- OS + applications

➤ Virtualisation légère

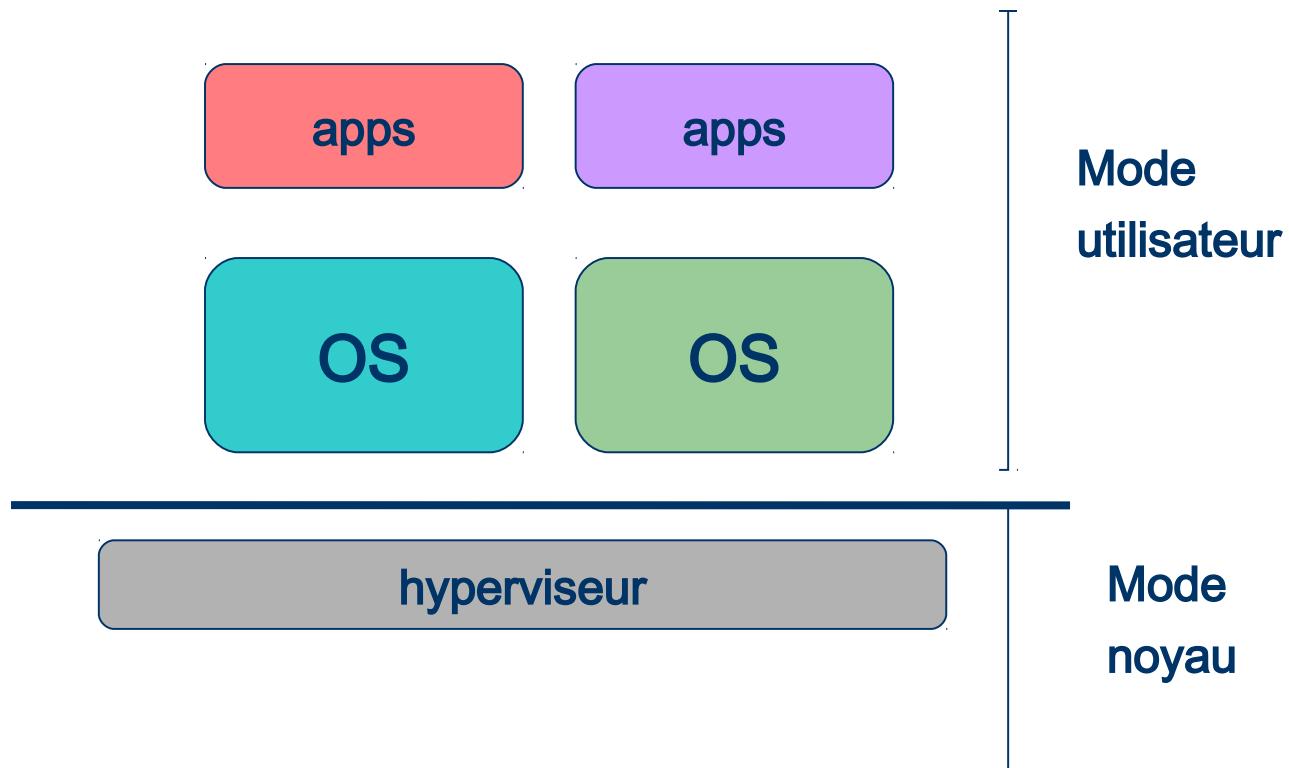
- Containers
- Ce que vous ferez en TP
- Application seule

Virtualisation lourde : dé-privilégier un OS

- **2 modes de fonctionnement d'un processeur (classiquement) : mode noyau et mode utilisateur**



Virtualiser : dé-privilégier un OS



Le zoo des hyperviseurs

- **Vsphere de VmWare - 60% du marché**
- **Hyper-V de Microsoft - 20 %**
- **Xen Server de Citrix – 4 % du marché**
- **Virtualbox d'Oracle**
- **QEMU/KVM, etc.**

Pourquoi virtualiser?

- **Dans les années 1990-2000, le coût des serveurs décroît**
 - ➔ En valeur absolue du fait de la concurrence
 - ➔ Par rapport au coût des mainframes encore très présents
- **Les éditeurs (Microsoft, distribution Linux) recommandent une application/service par système d'exploitation** →
 - ➔ Une machine pour le DNS
 - ➔ Une machine pour le mail
 - ➔ Une machine pour NFS, etc.

Pourquoi virtualiser?

➤ **Au final:**

- ➔ Une multiplicité de serveurs dans les datacenters des entreprises
- ➔ 80% ont une utilisation moyenne inférieure à 10%
- ➔ Des coûts d'exploitation/maintenance (personnel du service informatique) qui croissent avec le nombre de serveurs
- ➔ Des coûts en place : les salles serveurs ne sont pas indéfiniment extensibles
- ➔ Des coûts en climatisation/électricité élevés → NE JAMAIS NEGLIGER CES COÛTS

Pourquoi virtualiser?

- **Les serveurs deviennent si puissants qu'une seule application par serveur n'est plus justifiable**
 - ➔ Processeurs 64 bits multi-coeurs
 - ➔ Un serveur en 2009 est estimé, en moyenne, 10 à 12 fois plus puissant qu'un serveur en 2004
- **Remplacer les serveurs à raison de un pour un n'est plus possible!!**

Avantages de la virtualisation

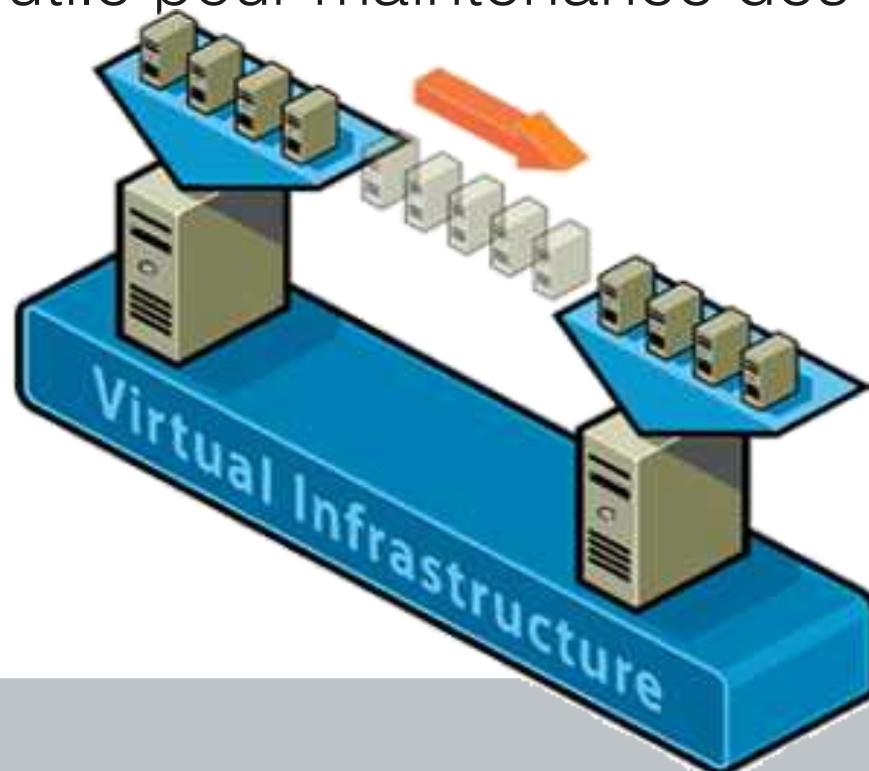
- **Plaçons-nous dans le cas où la virtualisation est effective**
- **Vous avez acheté :**
 - ➔ Deux gros serveurs
 - ➔ Des licences d'un outil de virtualisation, par exemple VMWARE
 - ➔ Les formations pour votre personnel

Avantages de la virtualisation

- **Réduction des coûts**
 - ➔ 20 à 40% en général
- **Avantages additionnels :**
 - ➔ Gain de place
 - ➔ De nouvelles fonctionnalités

Avantages de la virtualisation

- **Migration des machines virtuelles d'un serveur physique à l'autre**
 - ➔ Utile si panne → notion de disponibilité
 - ➔ TRES utile pour maintenance des serveurs physiques



Avantages de la virtualisation

- **Mise en service quasi-instantanée d'une nouvelle machine**
- **En général, avec une interface graphique, vous:**
 - ➔ Spécifiez le nombre de CPU, la quantité de mémoire, de disque, les accès réseaux, le système d'exploitation
 - ➔ Indiquez où se trouve l'ISO de l'OS
 - ➔ Démarrez l'installation
- **Mieux encore, vous utilisez un « template », une machine quasi-finalisée qui permet de démarrer en 1 minute une nouvelle machine**
 - Ce que fait createvm en salle de TP
 - Ce qu'offre Amazon Web Service, VMware, Openstack

Avantages de la virtualisation

- **Possibilité de faire des instantanés (snapshots) des machines**
- **Exemple :**
 - ➔ vous voulez installer une nouvelle fonctionnalité sur une machine, faire une mise à jour
 - ➔ Vous n'êtes pas sûr du résultat.
 - ➔ Vous :
 - * Faites un instantané
 - * Effectuez le test
 - * Si l'installation échoue, vous ... revenez dans le temps!!!

Virtualisation légère : les containers Linux

Container versus hyperviseur

- Partage du noyau entre toutes les VMs (ou containers)
- Sur un serveur typique:
 - 10-100 machines virtuelles
 - 100-1000 containers
- Un containter est un groupe de processus dans une machine Linux, isolés des autres processus qui tournent sur la machine
 - Utilisation des **namespaces** (du noyau Linux) pour assigner à un groupe de processus : isolation, leur propre pile réseau (interfaces, sockets, routage), volumes
 - Utilisation du **cgroups** (du noyau Linux) pour assigner des ressources à ces processus par exemple de la CPU, de la mémoire
 - Similaire à ce que vous faites sous Virtualbox

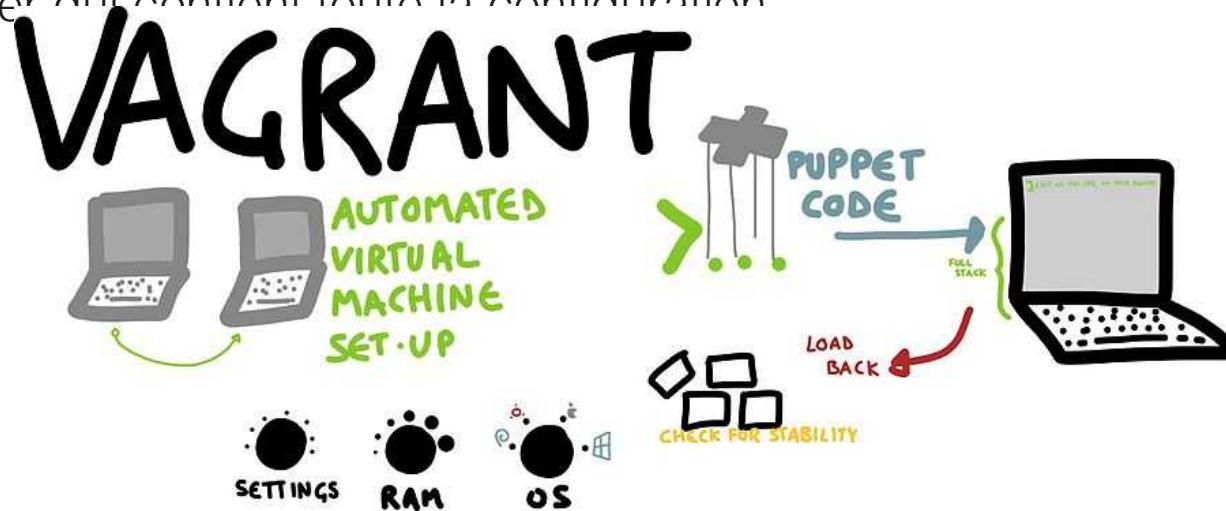
Container versus Hyperviseur

- De l'intérieur, ressemble à une VM
 - De l'extérieur, ressemble à des processus normaux
 - Un container peut être une VM complète ou un groupe de processus par exemple un serveur Apache ou MySQL
-
- Moteurs de gestion de containers:
 - LXC (LinuX Containers – August 2008)
 - Docker (started in March 2013)
 - OpenVZ (started in 2005)

Management des VMs et containers

Gestion de VMs

- › Vmware Vsphere, Citrix Xen permettent de gérer quelques serveurs physiques
- › Vagrant: Gestion de VMs indépendamment des hyperviseurs
 - › Notion d'images (boxes de Vagrant)
 - › Configuration automatique de VM: support de Puppet, Chef, Ansible
 - › Un fichier qui contient toute la configuration



Fichier de configuration Vagrant

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure(2) do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://atlas.hashicorp.com/search.
  config.vm.box = "ubuntu/vivid64"

  # Disable automatic box update checking. If you disable this, then
  # boxes will only be checked for updates when the user runs
  # `vagrant box outdated`. This is not recommended.
  # config.vm.box_check_update = false

  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine. In the example below,
  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 5001, host: 5001

  # Create a private network, which allows host-only access to the machine
  # using a specific IP.
  # config.vm.network "private_network", ip: "192.168.33.10"

  # Create a public network, which generally matched to bridged network.
  # Bridged networks make the machine appear as another physical device on
  # your network.
  config.vm.network "public_network"
```

Gestion de VMs

➤ Plateforme de gestion de clouds

➤ Openstack

➤ Chaque fonction (managementdef VM, réseaux, volumes, identités) est un composant (au final un service Linux)

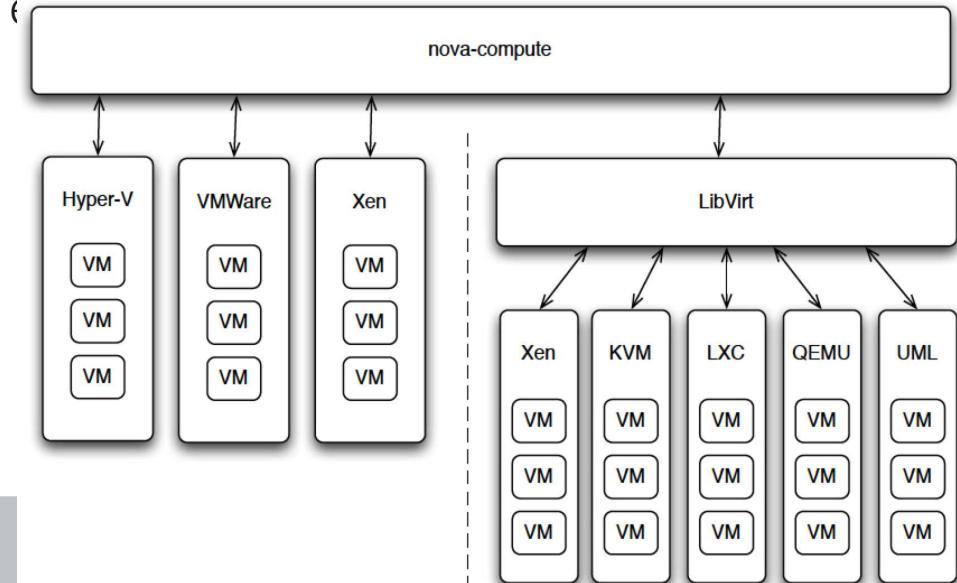
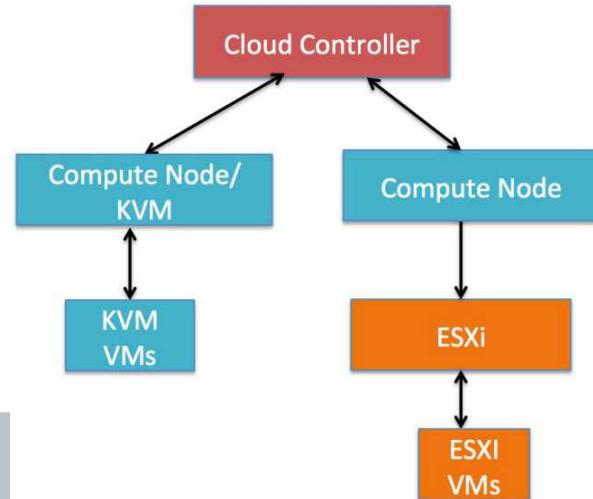
➤ Nova: compute nodes (hyperviseurs)

➤ Cinder : volumes

➤ Neutron : réseaux

➤ Les composants interagissent via une API REST

➤ Les nœuds Compute (serveurs physiques) peuvent faire tourner des hyperviseurs différents : KVM, Xen, Citrix, ...



Gestion de containers

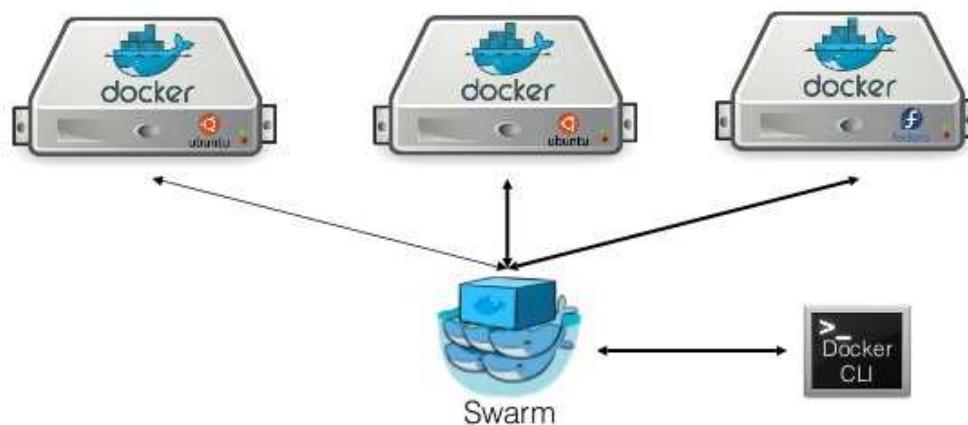
➤ Orchestration de containers

- Serveur unique: Docker, LXC
- Several multiplesl: Docker Swarm

➤ Orchestration avancés : Kubernetes, Swarm, Mesos



With Docker Swarm



Ce dont on a pas le temps de parler....

➤ **SDN : Software Defined Network**

- Le réseau version Le Seigneur des Anneaux avec
 - des contrôleurs idiots (enfin...)
 - Un contrôleur unique pour les contrôler tous !

➤ **NFV : network virtualisation function**

- Idée : mettre les fonctions réseaux du type NAT, répartiteur de charge, pare-feu dans des VMs

➤ **La virtualisation dans les réseaux mobiles**

- CloudRAN (RAN = Radio Access Network)



La virtualisation de serveurs avec VMWare Infrastructure

-

Retour d'expérience

Rodérick Petetin – CRI INSA Rennes



- ❖ Le contexte INSA Rennes
- ❖ Objectifs du projet
- ❖ Travail préparatoire
- ❖ Architecture mise en place
- ❖ Processus de migration des serveurs
- ❖ Bilan
- ❖ Observations
- ❖ Evolutions prévues



- ☒ 1600 étudiants + 450 personnels
 - ☒ 35 serveurs gérés par le CRI
 - ☒ CRI : 11 “techniciens” + 1 directeur
- Equipe système : 2 personnes



Forte demande de nouveaux services

- + 1 service majeur par serveur (+ 1 ou 2 mineurs)
- = **Doublement du parc serveurs (en 3 ans)**

Augmentation du nombre de :

- ➡ problèmes matériels
- ➡ raccordements électriques
- ➡ raccordements au réseau
- ➡ prolongations des garanties du matériel



- ☒ Moins de matériel
- ☒ Maximiser l'utilisation du matériel
- ☒ Simplifier l'administration (install, upgrades, ...)
- ☒ Premier pas vers un Plan de Reprise d'Activités



- ❖ Etude d'éligibilité (société extérieure)
 - Indicateurs sur l'util. réelle des ressources
 - Tout n'est pas virtualisable
 - ➡ Scénarios + propositions d'architecture
- ❖ Assainissement du parc serveurs
 - Elimination des serveurs "historiques"
 - Regroupement de services mineurs
 - Ecriture du cahier des charges

Travail préparatoire

Criadm3 (Microsoft Windows 2000 Advanced Server)
1023 Mo RAM
Conso CPU moyenne: 2.7% - RAM moyenne: 467 Mo

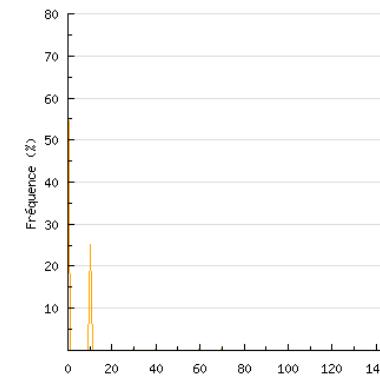


Criadm3 (Microsoft Windows 2000 Advanced Server)
1023 Mo RAM
Conso CPU moyenne: 2.7% - RAM moyenne: 467 Mo

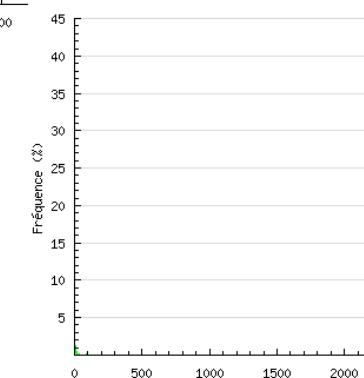
Distribution mémoire
(1023 Mo)

— Mémoire (Mo)

Criadm3 (Microsoft Windows 2000 Advanced Server)
1023 Mo RAM
Conso CPU moyenne: 2.7% - RAM moyenne: 467 Mo



Criadm3 (Microsoft Windows 2000 Advanced Server)
1023 Mo RAM
Conso CPU moyenne: 2.7% - RAM moyenne: 467 Mo



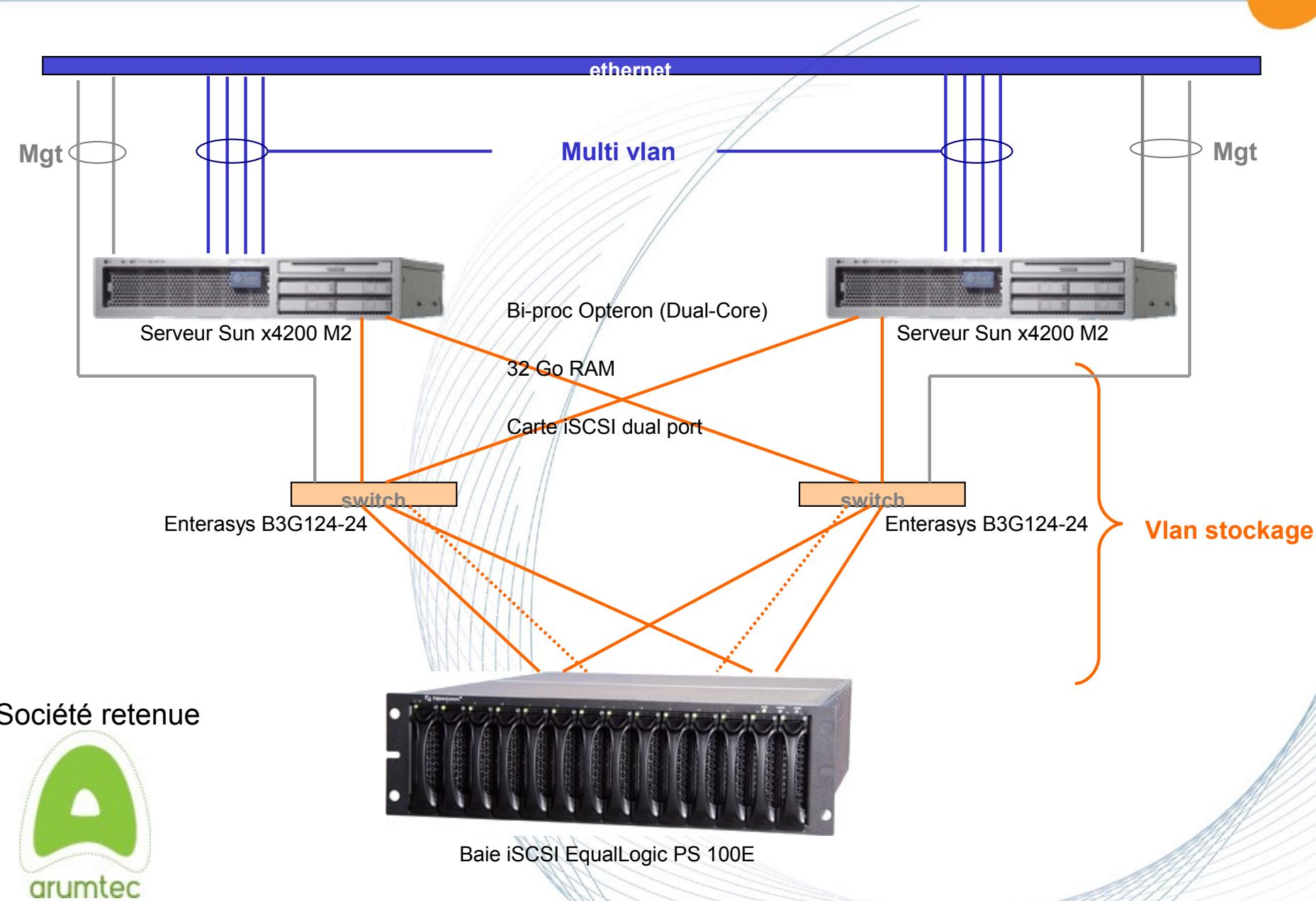
Travail préparatoire

		CPU Avg %	CPU Max %	RAM %	IO /s Avg %	IO /s Max %	IO Mbs %	NET Mbs %	Conso	STATUS	Domaine	Fonction
	acrisios	0,75	1,12	26,67	0,00	0,00	7,32	0,01	0,6	o	Commun	Serveur Oracle
	annuaire	0,16	0,63	13,33	0,00	0,00	5,57	6,93	0,6	o	Commun	Serveur LDAP / DNS
	anubis	0,30	0,90	13,33	0,00	0,00	0,62	0,48	0,4	o	Commun	Serveur Appli java
	anuter	0,31	2,82	10,00	0,00	0,00	0,41	0,06	0,2	o	Commun	Serveur LDAP / portail de test
	barbu	0,29	1,35	7,50	0,00	0,00	8,03	1,20	0,2	o	Recherche	Serveur mail recherche
	Biblio	5,30	7,33	4,17	0,00	0,25	2,47	1,72	0,0	o	Recherche	Serveur appli biblio
	cerbere	0,16	0,47	6,67	0,00	0,00	0,75	0,01	0,2	o	Ext	Serveur SSO
	chapi	0,16	0,63	10,00	0,00	0,00	0,31	1,86	0,2	o	Ext	Serveur appli portail
	chapo	0,16	0,31	9,17	0,00	0,00	0,04	0,00	0,2	o	Ext	Serveur appli portail
	Criadm1	2,41	4,82	5,00	0,00	0,18	2,83	2,03	0,0	o	Recherche	Controleur domaine personnels
	Criadm2	0,16	0,47	4,17	0,00	0,13	1,38	0,00	0,0	o	Recherche	Serveur intranet
	Criadm3	0,18	0,54	6,67	0,01	0,44	2,44	1,39	0,2	o	Recherche	Controleur domaine personnels
	Criadm5	0,13	1,46	3,33	0,00	0,76	1,67	0,03	0,0	o	Recherche	Serveur appli Oracle
	cubitus	1,10	2,82	3,33	0,00	0,00	0,84	0,01	0,0	o	Commun	Serveur Radius
	cyrano	0,14	0,29	3,33	0,00	0,00	0,75	0,03	0,0	o	Commun	Serveur test mail
	cyrus	1,72	10,18	14,17	0,00	0,00	11,40	14,01	1,0	o	Commun	Serveur Mail étudiants
	Educ1	0,24	2,17	8,33	0,01	1,86	8,77	6,52	0,6	o	Enseignement	Controleur AD étudiants
	Educ2	0,24	0,72	6,67	0,00	0,16	0,44	0,19	0,2	N	Enseignement	Replica AD
	geronimo	0,16	1,10	5,83	0,00	0,00	0,06	3,31	0,0	o	Ext	Frontal Apache portail
	hugh	2,11	9,34	13,33	0,00	0,00	7,00	3,06	0,6	o	DMZ	Serveur Web
	metro	4,07	17,54	13,33	0,00	0,00	0,54	0,01	0,4	o	Commun	Métrologie
	Nte2	0,12	0,48	1,67	0,00	0,03	0,03	0,02	0,0	o	Enseignement	Serveur WEBCT / Clefs réseau
	Nte3	0,07	0,22	2,50	0,00	0,03	0,03	0,00	0,0	o	Enseignement	?
	Pc-dir15	0,14	0,29	2,50	0,00	0,16	1,23	0,01	0,0	o	Recherche	Appli gestion infirmerie
	RER1	0,07	0,22	5,00	0,00	0,45	2,41	2,46	0,0	N	Enseignement	Universalis / iReef
	rocco	0,63	0,94	5,00	0,00	0,25	0,62	0,00	0,0	o	Commun	Serveur appli cartes magnetiques
	satanas	1,20	8,43	13,33	0,00	0,00	9,08	13,26	1,0	o	Commun	Serveur stockage étudiants

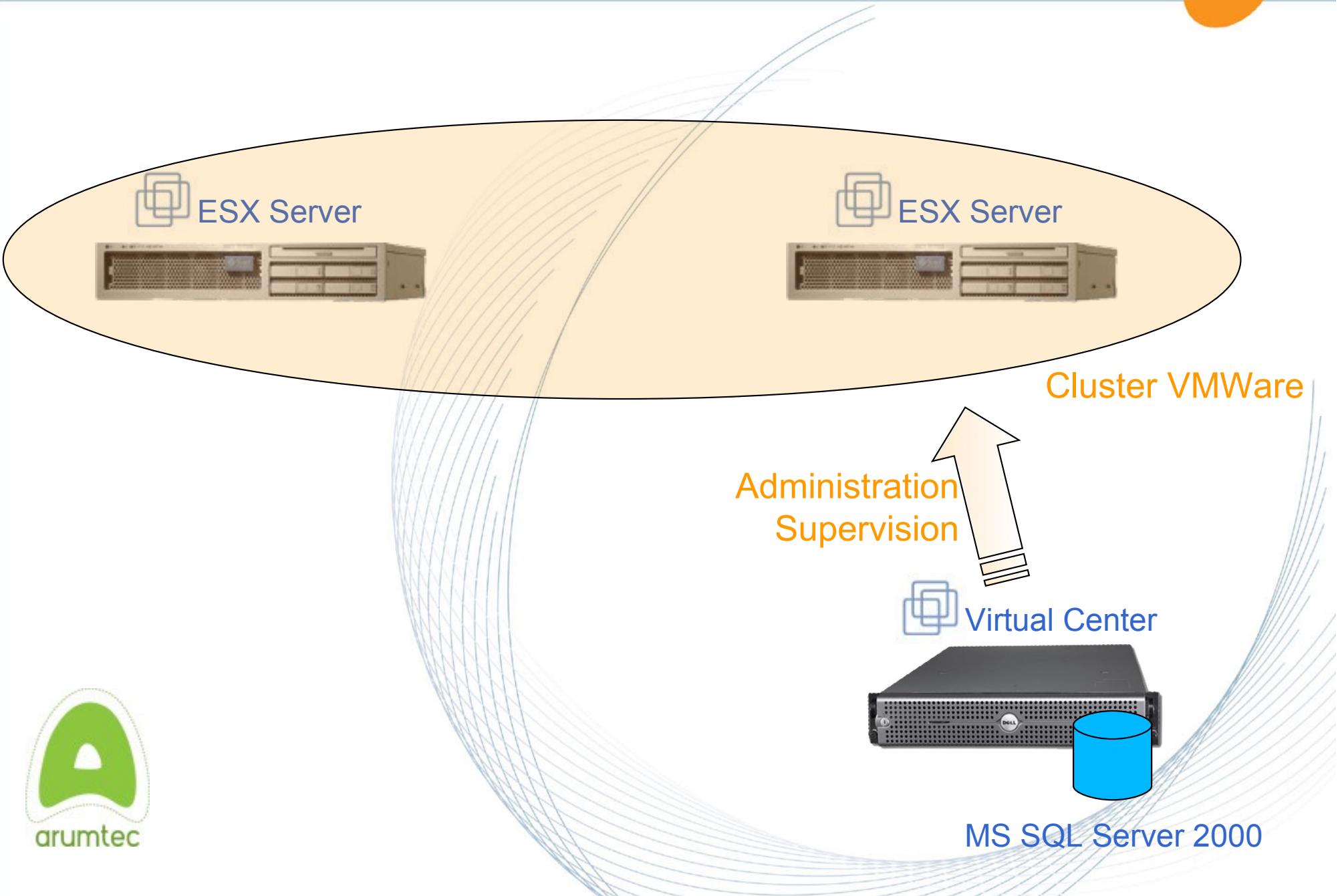


- ❖ Etude d'éligibilité (société extérieure)
 - Indicateurs sur l'util. réelle des ressources
 - Tout n'est pas virtualisable
 - ➡ Scénarios + propositions d'architecture
- ❖ Assainissement du parc serveurs
 - Elimination des serveurs "historiques"
 - Regroupement de services mineurs
 - Ecriture du cahier des charges

Architecture matérielle



Architecture logicielle





Licences achetées

- **2 licences VMWare Infrastructure 3 Ent (2 CPU)**
 - ESX : système installé sur les machines physiques qui accueillent les VMs (RedHat Linux adapté)
 - HA : Haute disponibilité des VMs. Redémarrage des VMs sur incident.
 - DRS : Attribution dynamique des ressources
 - VMotion : Migration à chaud des VMs
- **1 licence VMWare Virtual Center** : Administration et supervision de l'infrastructure.



- ☒ Parc serveurs : 35 serveurs
- ☒ Périmètre pré-défini : 30 serveurs
- ☒ Migrations planifiées : 24 serveurs
- ☒ Migrations réalisées : 23 serveurs



- ☒ Cahier des charges : 4j.homme
- ☒ Étude dossiers : 2j.homme
- ☒ Mise en place socle : 6j.homme
- ☒ Migrations : 20j.homme
- ☒ Suivi de projet : 7j.homme

Charge de travail globale : 39j.homme



- ☒ Période 10 jours de migrations
- ☒ 1 à 4 serveur(s) par jour à planifier
- ☒ Durée de migration d'un serveur : 2 heures à 12 heures
- ☒ Informations d'arrêt de services auprès des utilisateurs



• <u>Matériel (support 3ans)</u>	<u>62 384 €</u>
• 2 serveurs Sun X4200 M2 32Go RAM	
• Baie EqualLogic PS100E 3,5To	
• 2 switchs Enterasys B3G124-24	
• <u>Logiciel (support 3 ans)</u>	<u>11 602 €</u>
• 2 licences VMWare Infrastructure 3 Enterprise	
• 1 licence VirtualCenter	
• <u>Service</u>	<u>26 300 €</u>
• Pré-install	
• Migration	
• Docs	
• Support + audit + préconisations 1 an	
• <u>Formation</u>	<u>7381 €</u>
• 3 personnes x 4 jours	
TOTAL	107 667 € HT

Observations



Positif

- ❖ C'est drôlement bien !
- ❖ Plateforme évolutive
- ❖ Administration réellement simplifiée
- ❖ Nouveaux outils (migration à chaud, templates d'install, monitoring, planification,...)
- ❖ Assainissement parc

Négatif

- ❖ Mais ça fait peur !
- ❖ Migration = recopie et adaptation bas niveau
- ❖ Opération extrêmement lourde
- ❖ Nécessité d'arrêter chaque serveur
- ❖ Temps de transfert variable
- ❖ Pas de retour arrière après mise en service des serveurs virtuels



💡 Système / Réseau :

- Recenser très tôt tous les services existants sur les serveurs
- Faire le ménage sur les serveurs avant la migration
- Attention aux licences produits basées sur l' $@$ MAC
- Attention à ne pas sous-dimensionner les disques qui accueillent les VMs
- Prévoir la redondance maximum de l'infrastructure
- Prévoir un grand nb de ports réseau

💡 Plateforme :

- Console d'administration très ergonomique
- Plutôt stable (1 seul bug sérieux depuis la mise en production)
- Réelle isolation entre les machines
- Le DNS est très important dès qu'il y a plus d'un ESX
- DRS et VMotion sont des options indispensables dès qu'il y a plus d'un ESX
- Avoir une assistance technique est un plus indéniable (au début au moins)



💡 Gestion de projet :

- Prévoir formation des administrateurs avant ET après la migration
- Prévoir transfert de compétences aux collègues qui interviennent sur les serveurs (utilisation console entre autres)
- Se poser la question de la diffusion d'informations aux utilisateurs concernant la migration

Machines Virtuelles

Guillaume Urvoy-Keller

October 7, 2019

Rôle de l'hyperviseur

- Offrir à l'OS invité (de la VM) un environnement virtuel tel que l'OS invité s'exécute **sans** modification.
- Assurer une correspondance efficace entre CPU/RAM/périphériques virtuel(le)(s) et physique(s)

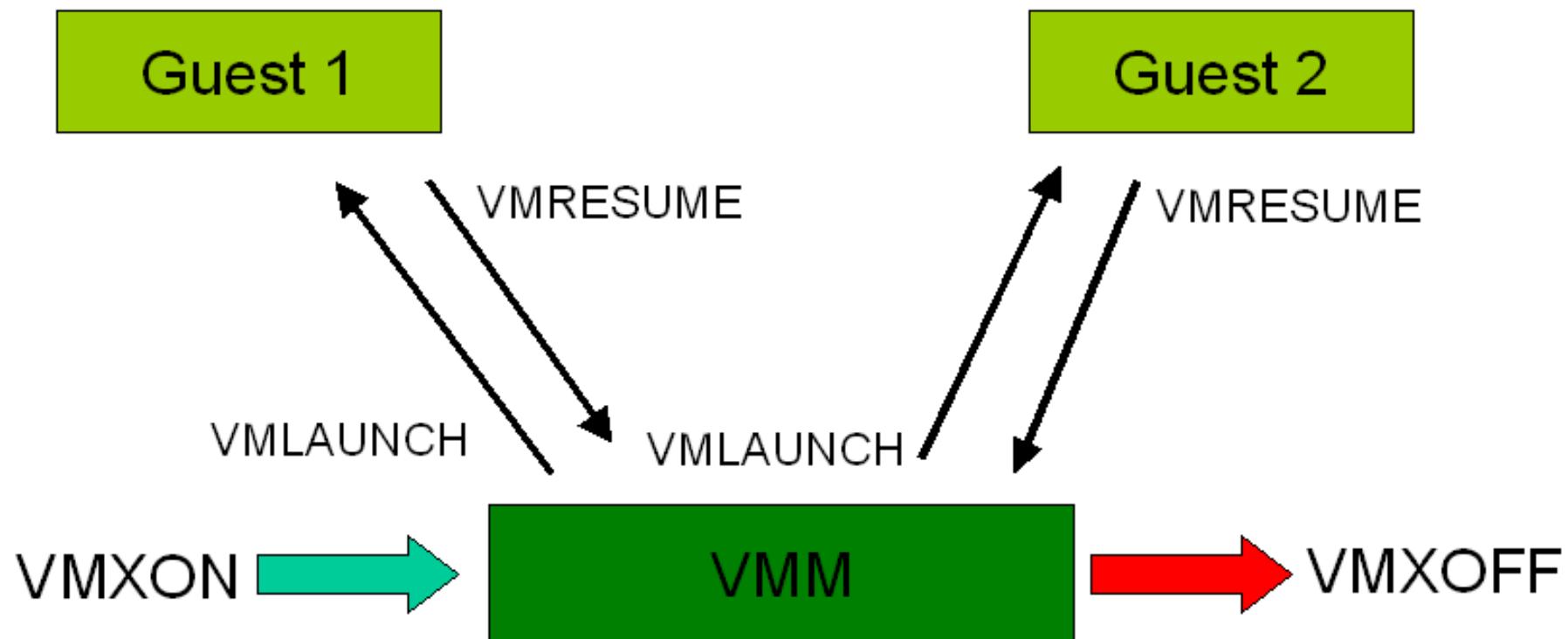
- Architecture x86 commune à Intel et AMD
 - On parle d'ISA: Instruction Set Architectures
 - ISA: Jeux d'instruction et architecture des registres
 - Instructions en mode noyau (accès RAM, périphériques) pour l'OS hôte (ou l'hyperiseur) seulement
 - Instructions en mode utilisateur: opérations sur les données du programme
- Virtualisation bas niveau:
Techno Intel Virtualization Technology (Intel VT) et AMD Virtualization (AMD-V) introduites en 2006... puis enrichies
- Intel-VT et AMD-V aident l'hyperviseur dans sa gestion des VMs

Virtualisation processeur

- Le processeur permet un multiplexage temporels des processus
⇒ un seul processus à la fois : l'hyperviseur ou l'OS invité ou le programme invité
- Question : comment l'OS invité récupère la main sur le programme invité? Et comment l'hyperviseur récupère la main sur l'OS invité
- Lorsqu'il y a un système non virtualisé - OS+programmes - l'OS modifie le timer de temps (dans un registre système) qui repasse le processeur en mode noyau
⇒ Conflit : l'OS invité va l'utiliser pour ses programmes et l'hyperviseur pour ses OS invités.

Activation du mode

- Intel-VT et AMD-v permettent d'avoir deux timers de temps, un pour la VM et un pour l'hyperviseur.
- Pour cela, instructions spécifiques: VMXON/OFF: on va activer des VMs et VM launch/resume pour donner la mains aux VMs



- Cas non virtualisé:
 - Il existe des pages virtuelles (de 1 à 2^{64}) pour chaque programme et des pages réelles vues par l'OS
 - Une table de correspondance est maintenue en RAM avec un cache maintenu par le processeur (TLB)
- Cas virtualisé → 2 niveaux de traductions :
 - Programme invité → OS invité
 - OS invité → hyperviseur
- Intel offre l'extended page tables (EPT), connue sous le nom *SLAT, Second-Level Address Translation* chez AMD
- Chez Intel, depuis le Core i3

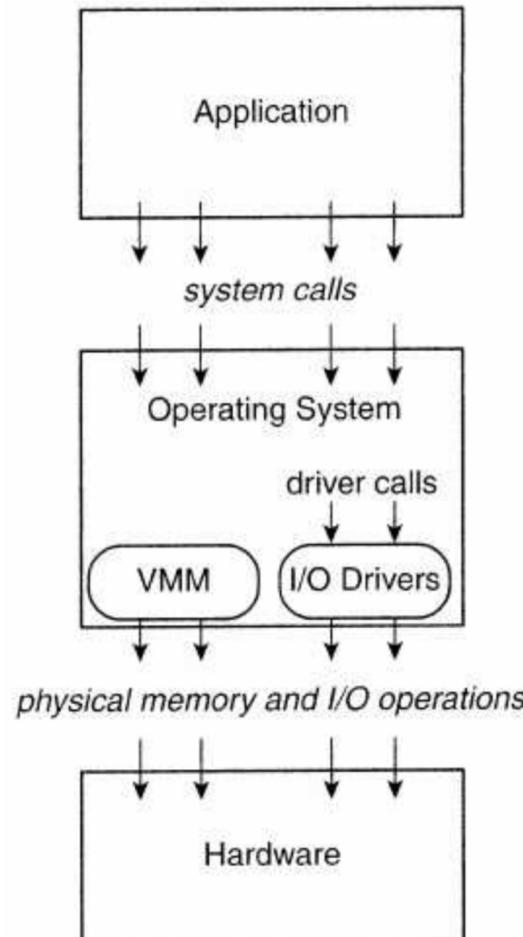
- E/S lentes. Il existe des techniques typiques pour améliorer les performances.
 - déportage (*offload*) du calcul de la somme de contrôle sur la carte réseau
 - Accès direct mémoire (DMA) : OS indique au contrôleur du périphérique une zone mémoire physique où écrire ses données.

- Une des tâches les plus complexes du fait de la prolifération des périphériques
- Technique générale : offrir une interface virtualisée à l'OS invité

Typologie des périphériques

- Dédiés. Ex : écran, clavier.
- Partitionnés. Ex : disque
- Partagés. Ex: carte réseau
- Spoolés. Ex : imprimante. Partagé, mais à un niveau de granularité élevé (fichiers dans spooler). En pratique, imprimantes souvent accédées par le réseau → retour au cas précédent
- Inexistants. Ex : switch virtuel qui permet la communication inter-VM.

A quel niveau intercepter les E/S?

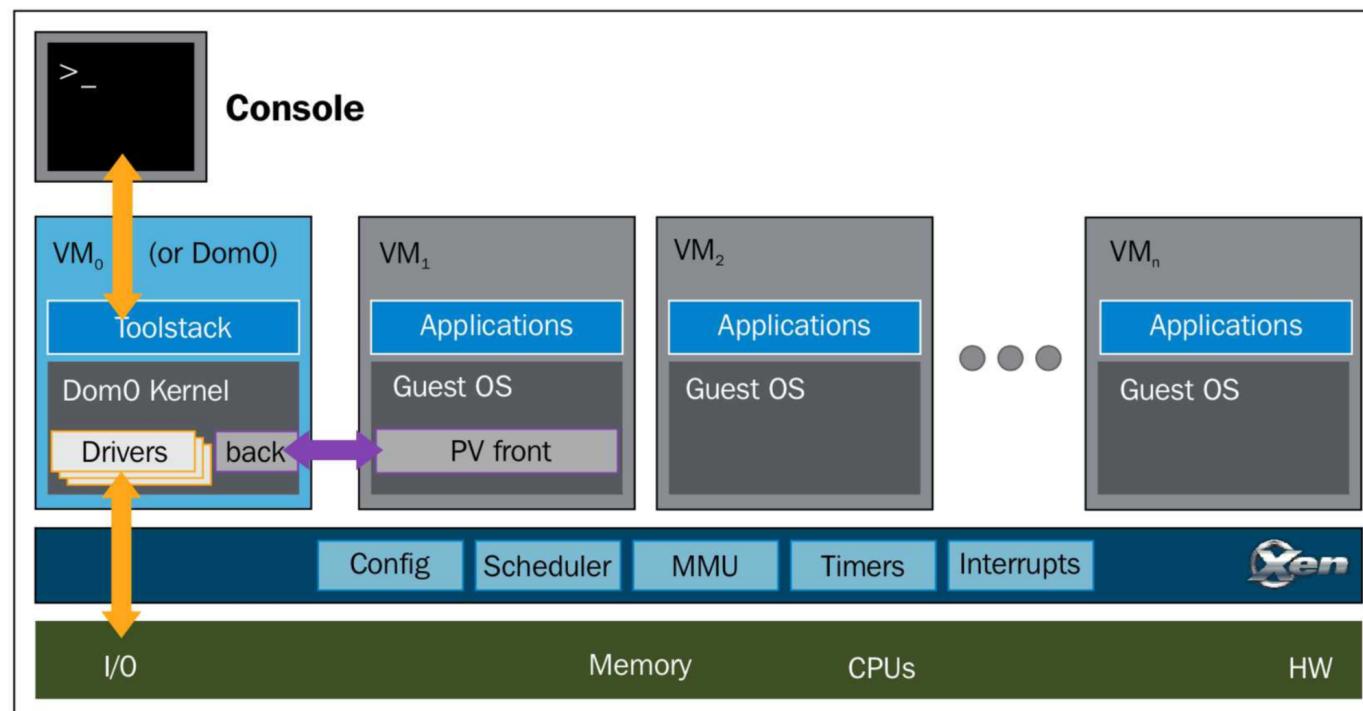


A quel niveau intercepter les E/S?

- Appels système? Compliqué
- Instruction d'E/S du processeur? Compliqué mais des technos Intel le permettent
- Driver? Souvent utilisé, ex : Virtualbox additions invité, VMware Tools, Xen.
 - On offre à la VM un driver spécial à la VM qui permet une discussion directe avec l'hyperviseur.

Exemple de Xen

- Driver a un *front-end* dans la VM et un *back-end* dans Dom0
Dom0 : une VM spéciale, qui peut parler aux périphériques physiques
 - Dom0 est séparée de l'hyperviseur pour ne pas exposer l'hyperviseur aux bugs (fréquents) dans les drivers.



Source: <http://www.xenproject.org/>

Support processeur

Idée : donner un accès direct à l'OS invité sans médiation de l'hyperviseur

- Intel® Virtualization Technology for Directed I/O (VT-d)
- Virtual Machine Device Queues (VMDQ),
- Single Root I/O Virtualization (SR-IOV, a PCI-SIG standard)
- Intel® Data Direct I/O Technology (Intel® DDIO) enhancements

Exemple DMA

- Système non virtualisé OS invité "voudrait" utiliser du DMA pour ses programmes
- Problème: OS invité ne voit pas les pages physiques (seul l'hyperviseur les voit)
- Utilisation d'une I/O MMU (memory management unit - sur le processeur) → qui fait la correspondance entre adresse physique de l'hôte et adresses physiques (réelles) de l'hyperviseur.

Containers

Guillaume Urvoy-Keller

November 26, 2019

Agenda

1 Introduction

2 Docker

- Les bases
- Images

3 Réseaux et Volumes

4 Swarm

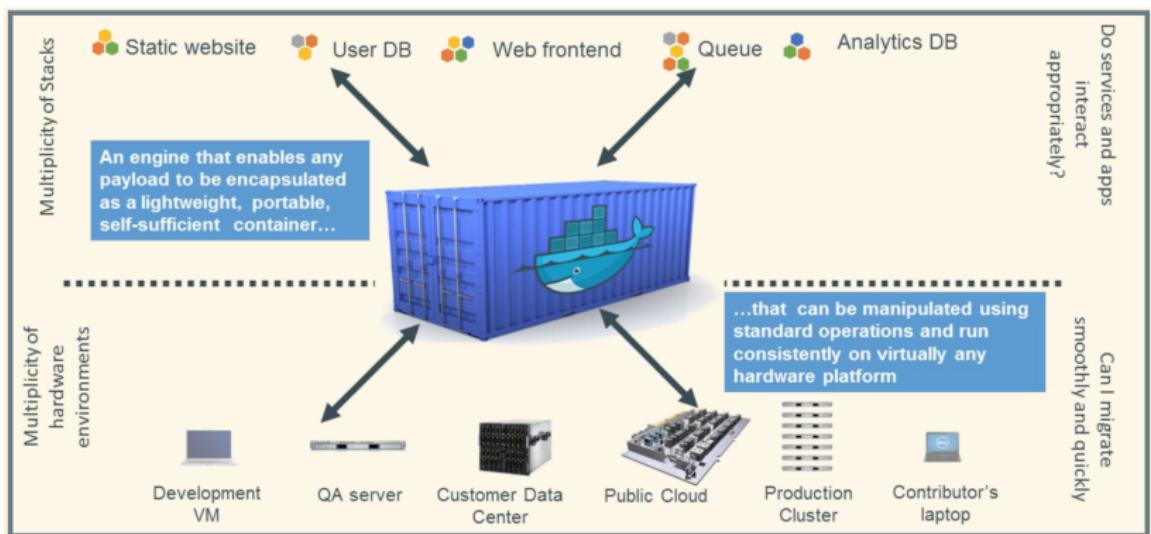
Source documents

- Remerciement à Christian Benedetti, administrateur système, Université Côte d'Azur pour son cours et ses articles dans Linux Magazine sur les containers (notamment)

Les promesses des containers

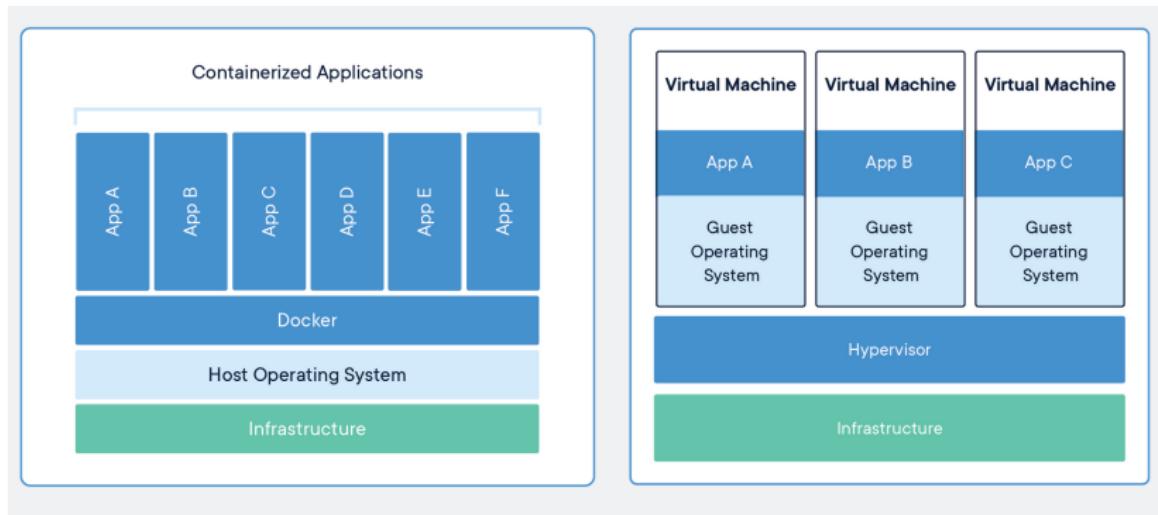
- Les installations (librairies) des différentes applications ne créent pas des problèmes de configuration mutuels
- Portabilité du container: déploiement possible sur des architectures variées.

Les promesses des containers



Source : docker.com

VMs vs. container



Source : docker.com

VMs vs. container

VM :

- +
 - OS complet
 - Isolation "parfaite" entre VM
- -
 - Usage disque important
 - Impact sur les performances

Container :

- +
 - Faible espace disque utilisé
 - Impact quasi nul sur les performances CPU, réseau et I/O
- -
 - Fortement lié au kernel hôte
 - Sécurité

Container Linux

(Groupe de) processus isolé(s) des mécanismes noyaux.

- Namespaces
- Cgroups

Namespaces

- Apparue dans le noyau 2.4.19 en 2002, réellement utiles en 2013 dans le noyau 3.8
- Limite ce qu'un processus peut voir du système:
- 6 namespaces : pid, net, mnt, uts, ipc, user
- Chaque processus est dans un namespace de chaque type
- **Les namespaces sont natifs en Linux:** même si il n'y a pas de container créé ou Docker installé, chaque processus créé est associé à 6 namespaces (namespaces par défaut)

Namespace Réseau (NET)

Le processus ne voit que la pile réseau du namespace dont il fait partie:

- Ses interfaces (eth0, l0, différentes de l'hôte)
- Sa table de routage séparée.
- Ses règles iptables (firewall, NAT, etc)
- Ses sockets

Namespaces

UTS

détermine le nom de l'hôte (get/set hostname)

IPC (Inter-Process Communication)

Permettent à un groupe de processus d'un même namespace d'avoir leurs propres

sémaphores, files de messages et mémoire partagée

... sans risque de conflit avec d'autres groupes d'autres namespaces

USER

mappe uid/gid vers différents utilisateurs de l'hôte

uid 0 ⇒ 9999 du container C1 correspond à uid 10000 ⇒ 119999 sur l'hôte

uid 0 ⇒ 9999 du container C2 correspond à uid 12000 ⇒ 139999 sur l'hôte

Namespaces

MOUNT

Un namespace:

- dispose de son propre rootfs (conceptuellement proche d'un chroot)
- peut disposer de son propre /proc, /sys
- peut aussi avoir ses montages "privés"
- son /tmp (par utilisateur, par service)

PID

- Un processus ne voit que les processus de son namespace PID
- Chaque namespace pid a sa propre numérotation, débutant à 1
- Si le PID 1 disparait, le namespace est détruit.
- Un processus dans un namespace PID est aussi visible dans le namespace par défaut: il a un numéro dans les 2 (pas le même).

Manipulation Namespaces

Créés à l'aide des commandes clone() ou unshare()

Matérialisés par des pseudo-files dans /proc/\$PID/ns fichiers dans /proc/{pid}/ns

Exemple pour le processus ayant le PID 16 :

```
root@debian:/proc/16/ns# ls -al
total 0
dr-x---x--x 2 root root 0 Nov 25 15:01 .
dr-xr-xr-x 9 root root 0 Nov 14 12:18 ..
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 cgroup -> 'cgroup:[4026531835]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 ipc -> 'ipc:[4026531839]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 mnt -> 'mnt:[4026531840]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 net -> 'net:[4026531992]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 pid -> 'pid:[4026531836]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 pid_for_children -> 'pid:[4026531836]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 user -> 'user:[4026531837]'
lrxwxrwxrwx 1 root root 0 Nov 25 15:01 uts -> 'uts:[4026531838]'
```

Control groups

- Fonctionnalité du noyau, apparue en 2008 (noyau 2.6.24)
- Contrôle les ressources d'un processus en terme de CPU, mémoire, réseau, I/O

Namespace vs. cgroups

Namepaces permettent une séparation logique des containers alors que les cgroups permettent de spécifier quelles resources physiques ils peuvent consommer

Cgroups

- Une hiérarchie par resource (ou groupe de resources): hiérarchie CPU, Mémoire
- Les groupes sont matérialisés par des pseudos systèmes de fichiers: généralement montés dans /sys/fs/cgroup
- Pid 1 est placé à la racine de chaque hiérarchie
- Les nouveaux processus sont démarrés dans le groupe de leur parent ⇒ hiérarchie

Pour placer un processus dans un cgroup: On écrit le numéro du processus dans un fichier spécial

Docker - Les bases

Important

- Docker utilise les 6 namespaces et les cgroups pour construire des containers
- Docker est un moteur de gestion des containers.

Les bases

Démarre un container basé sur l'image debian et lance un processus, /bin/echo

```
$ docker run debian /bin/echo "Salut"  
Salut
```

Démarre un terminal en interactif relié à /bin/bash (dans l'image debian)

```
$ docker run -it debian /bin/bash  
root@2c666d3ae783:# ps -a  
PID TTY TIME CMD  
6 ? 00:00:00 ps
```

Exécuter une commande dans le namespace du container

```
$ docker exec -it happy_mietner /bin/bash  
root@2c666d3ae783:# exit
```

Les bases

Se détacher d'un container :

```
root@2c666d3ae783:/# ^P^Q
```

Se rattacher à un container existant :

```
$ docker attach happy_mietner
root@2c666d3ae783:/#
```

Gestion des containers

Lister les containers avec ps

```
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS
2c666d3ae783 debian "/bin/bash" 44 seconds ago Up 42 seconds
```

Lister les processus d'un container

```
$ docker top happy_mietner
UID PID PPID C STIME TTY TIM
root 23338 867 0 15:06 pts/15 00:00
```

Voir ce qui se passe dans un namespace depuis l'extérieur : le log

```
$ docker logs hungry_visvesvaraya Salut
Salut
```

Gestion des containers

Cycle de vie d'un container :

- Démarré
- Arrêté (par exemple suite à un exit) ⇔ équivalent d'une VM suspendue
- Redémarrage ou Destruction

Exemple

On démarre un container, puis on s'en détache, puis on s'y rattache puis on le suspend avec un exit.

A ce moment, le container n'est plus visible avec ps mais avec ps -a
Puis on le démarre avec un start

```
sh-3.2# docker run -it ubuntu /bin/bash
root@8cbe7248c918:/# ^P^Q
sh-3.2# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8cbe7248c918 ubuntu "/bin/bash" 23 seconds ago Up 22 seconds affectionate_dhawan
sh-3.2# docker attach 8cbe7248c918
root@8cbe7248c918:/# exit
sh-3.2# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
sh-3.2# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8cbe7248c918 ubuntu "/bin/bash" 44 seconds ago Exited (0) 6 seconds ago
    affectionate_dhawan
sh-3.2# docker start -i 8cbe7248c918
root@8cbe7248c918:/#
```

Images

- Ensemble de fichiers à partir desquels les containers sont construits
- Image composée de couches
- Des images peuvent partager des couches pour optimiser

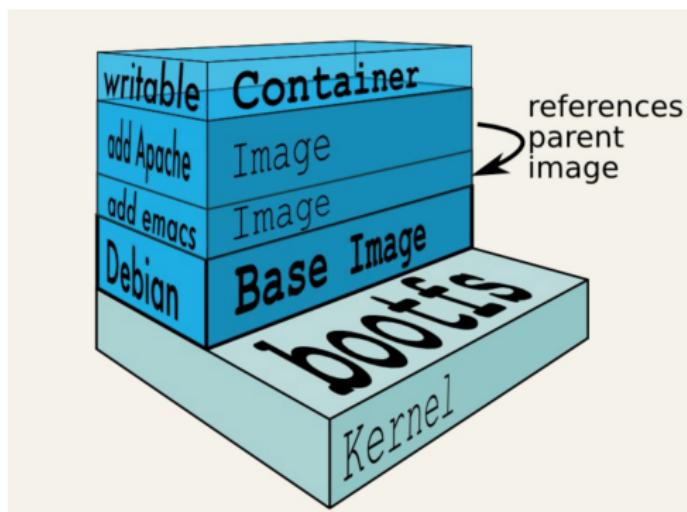


Image - Exemple

```
sh-3.2# docker pull ubuntu # telechargement ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
Digest: sha256:6e9f67fa63b0323e9a1e587fd71c561ba48a034504fb804fd26fd8800039835d
Status: Image is up to date for ubuntu:latest
docker.io/library/ubuntu:latest
sh-3.2# docker pull nginx # telechargement nginx
Using default tag: latest
latest: Pulling from library/nginx
000eee12ec04: Pull complete # couche 1
eb22865337de: Pull complete # couche 2
bee5d581ef8b: Pull complete # couche 3
Digest: sha256:50cf965a6e08ec5784009d0fccb380fc479826b6e0e65684d9879170a9df8566
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
sh-3.2# docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest 231d40e811cd 2 days ago 126MB
ubuntu latest 775349758637 3 weeks ago 64.2MB
```

Images

- Les images sont téléchargées depuis le hub docker
- Possibilité d'avoir un hub privé
- Nom de l'image :
 - un nom simple, ex: ubuntu, nginx → images officielles
 - un nom du type benedetti/nginx: image d'un utilisateur
 - tag: numéro de version

```
$ docker tag debian benedetti/debian:8.6
$ docker images | grep debian
debian latest 140f9bdfeb97 4 days ago 123 MB
benedetti/debian 8.6 93a2e30f1000 4 days ago 123 MB
debian 8.5 93a2e30f1000 3 months ago 125.1 MB
debian 8.4 f854eed3f31f 5 months ago 125 MB
debian 8.2 32f2a4cccab8 8 months ago 125 MB
```

Création Images

A partir d'un fichier Dockerfile :

```
FROM debian # depuis l'image debian
RUN apt-get update # on execute des commandes
RUN apt-get install -y nginx
CMD nginx -v # commande par defaut qui sera execute
```

Construction de l'image avec le nom benedetti/nginx et le tag 0.2:

```
$ docker build -t benedetti/nginx:0.2 .
Sending build context to Docker daemon 2.048 kB Step 1 : FROM debian
--> 93a2e30f1000
Step 2 : RUN apt-get update
--> Running in 20f50a8284f5
Get:1 http://security.debian.org jessie/updates InRelease [63.1 kB] ...
Processing triggers for sgml-base (1.26+nmu4) ... --> 95f9e15fa8d2
Removing intermediate container e99f0c6f5bd2 Successfully built 95f9e15fa8d2
```

Réseaux et Volumes

Gestion Réseaux

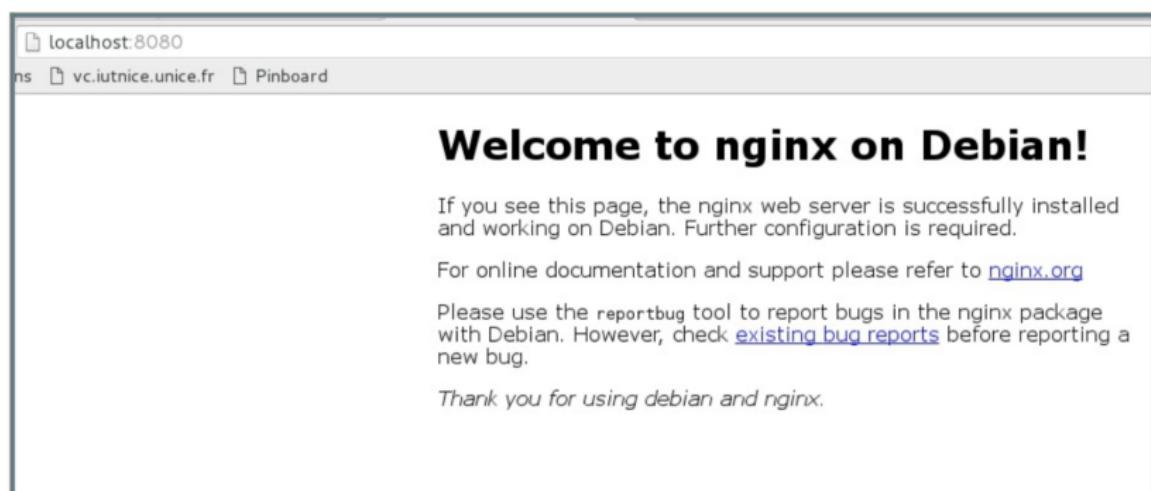
- Tous les ports sont privés par défaut
- Il faut les rendre explicitement accessibles depuis l'extérieur → accessibles comme un port de la machine hôte
- Exemple : le port 80 du container va correspondre au port 8080 sur l'hôte

```
$ docker run -d -p 8080:80 nginx
```

- L'adresse IP du container n'est **jamais exposé** à l'extérieur: on expose une application seulement!

Réseaux

```
$ docker run -d -p 8080:80 nginx
```



Volumes

On peut monter un volume de l'hôte dans le container, par exemple ici, le répertoire courant (pwd) depuis lequel on lance le container, qui est monté dans /usr/share/nginx/html.

```
$ docker run -d -v $(pwd):/usr/share/nginx/html -P nginx
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
eca9f876dc7f nginx "nginx -g 'daemon off;'" 5 seconds ago Up 3 seconds 0.0.0.0:32769->80/
      tcp confident_hellman
$ echo "Bienvenue sur mon image Nginx" > index.html
$ curl http://localhost:32769
Bienvenue sur mon image Nginx
```

Volumes

Création d'un volume nommé

```
$ docker volume create --name=logs logs
$ docker run -P -v logs:/var/log/nginx -d benedetti/nginx:0.7 $ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
eca9f876dc7f nginx "nginx -g 'daemon of..." 5 seconds ago Up 3 seconds rapid_joe
$ docker run -it --volumes-from rapid_joe debian ls /var/log/nginx access.log error.log
```

Les volumes sont persistants

```
$ docker rm -f rapid_joe
$ docker volume ls DRIVER
local local
VOLUME NAME 57a0848c5e5f2924be84e830757922c7b5b856aa5bac12e494da495 logs
```

Docker Swarm

Docker Swarm

Problème

Vous avez plusieurs hôtes Docker.

Vous désirez les utiliser sous forme de cluster, et répartir de manière transparente l'exécution de conteneur.

Solution

Docker Engine 1.12 inclut le mode swarm pour nativement gérer un cluster de Docker Engines qu'on nomme un swarm (essaim).

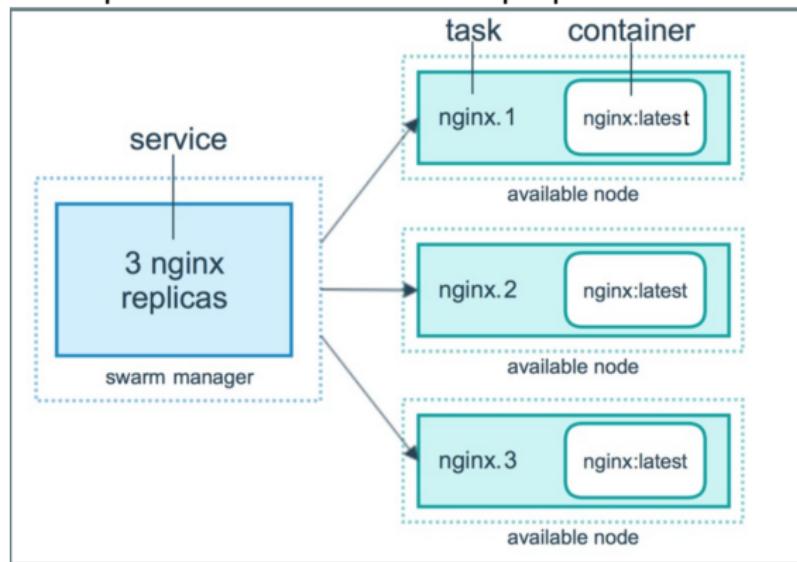
CLI Docker utilisée pour créer un swarm, déployer des service d'application sur un swarm

Docker Swarm

- Un swarm consiste en un ou plusieurs **managers** qui permettent de déployer les containers
- Au niveau swarm, on parle d'un **service** et non d'un container
- Un service contient un ou plusieurs containers
- Container exécutés sur les noeuds
- **Modes d'exécution:**
 - Répliqué : un container du service par noeud
 - Global: les N containers du services sont répartis sur le cluster

Docker Swarm

Exemple : service en mode répliqué avec container nginx



Création Swarm

Sur le manager dont l'IP est 172.31.4.182:

```
$ docker swarm init --advertise-addr <MANAGER-IP>
Swarm initialized: current node (8jud)
is now a manager. To add a worker to this swarm, run the following command:
docker swarm join --token SWMTKN-1-59fl4ak4nq4eprhrola2l87... 172.31.4.182:2377
```

Sur les noeuds :

```
docker swarm join --token SWMTKN-1-59fl4ak4nq4eprhrola2l87... 172.31.4.182:2377
```

On fait un ls niveau swarm et on trouve 2 noeuds actifs

```
docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS
8jud...ox4b * ip-172-31-4-182 Ready Active Leader
ehb0...4fvx ip-172-31-4-180 Ready Active
```

Lancement tâche sur Swarm

Sur le manager, on lance le service helloworld :

```
$ docker service create --replicas 1 --name helloworld alpine ping docker.com  
9uk4639qpg7npwf3fn2aasksr
```

On fait un ls niveau service (le service n'a qu'un replica ici):

```
$ docker service ls  
ID NAME SCALE IMAGE COMMAND  
9uk4639qpg7n helloworld 1/1 alpine ping docker.com
```