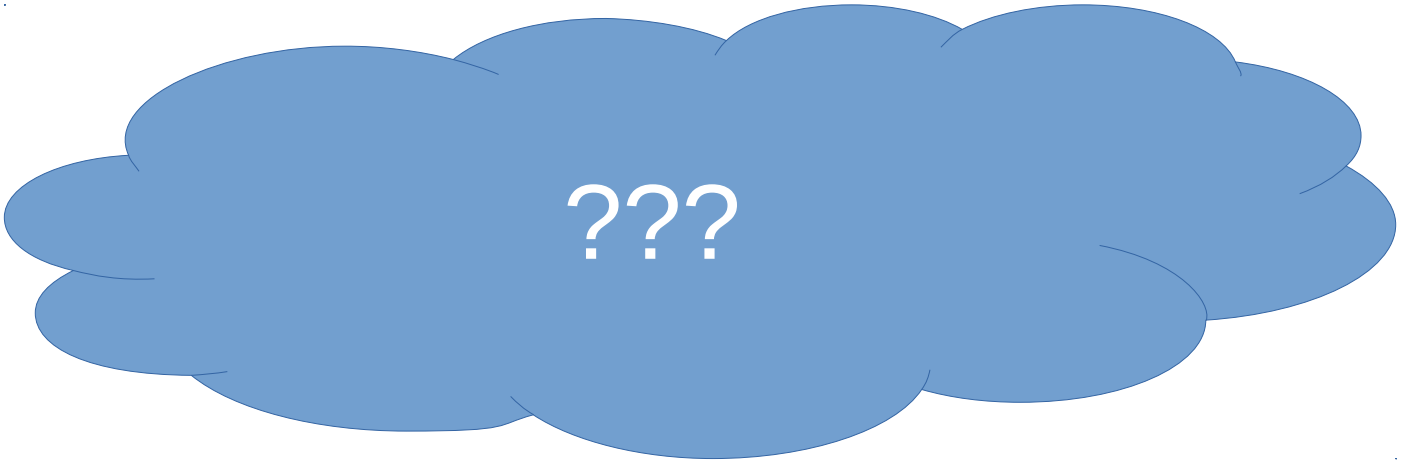
Introduction au cloud computing

G. Urvoy-Keller

DUT R&T

Qu'est-ce le cloud computing pour vous ?



Des exemples

- **Google :**
  - Mail/Agenda
  - Drive
- **Dropbox**
- **Amazon Web services**

- **Idées de base du cloud computing : vous utilisez un service dont vous ne gérez pas le matériel (= serveurs physiques)**
  - Un opérateur gère le matériel et multiplexe les utilisateurs
  - Les utilisateurs sont des particuliers ou des entreprises

Mots-clefs à connaître

- **SaaS : Software as a Service**
- Ex : Google *

- **PaaS : Platform as a Service**
- Plus difficile : vous gérez du code et c'est un service extérieur qui instancie les machines sur lesquelles votre code tourne
- Ex : Heroku

- **IaaS : Infrastructure as a Service**
- Vous louez des machines virtuelles
- Ex : Amazon Web Services, Google Cloud, Windows Asur

Mots-clefs liés au cloud computing

- **Virtualisation**
  - Outil de base pour multiplexer les utilisateurs
- **Vitualisation lourde**
  - Des machines virtuelles complètes
    - Ex : ce que vous utilisez en TP, Virtualbox mais aussi : VMware, Xen
- **Virtualisation légère**
  - Ensemble de processus séparés des autres processus d'une machine
  - Ex : Docker, Rackspace

Mots-clefs liés au cloud computings

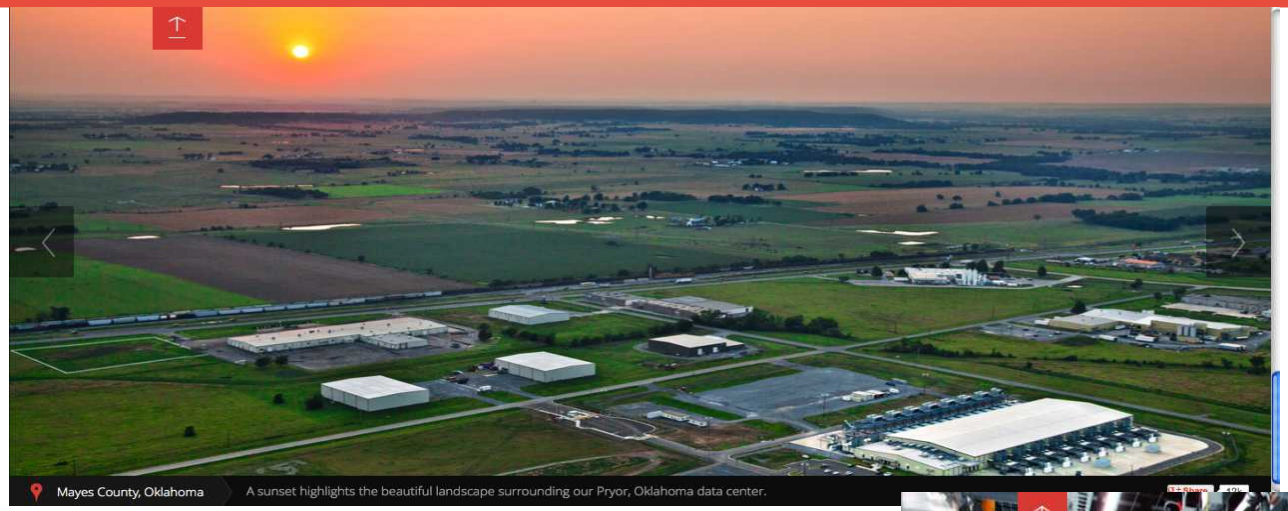
- **Clouds privés, clouds publics**
- Privé : on utilise les techniques de cloud public, comme la virtualisation sur un ensemble de serveurs privés

- **Ensemble de serveurs = data centers**

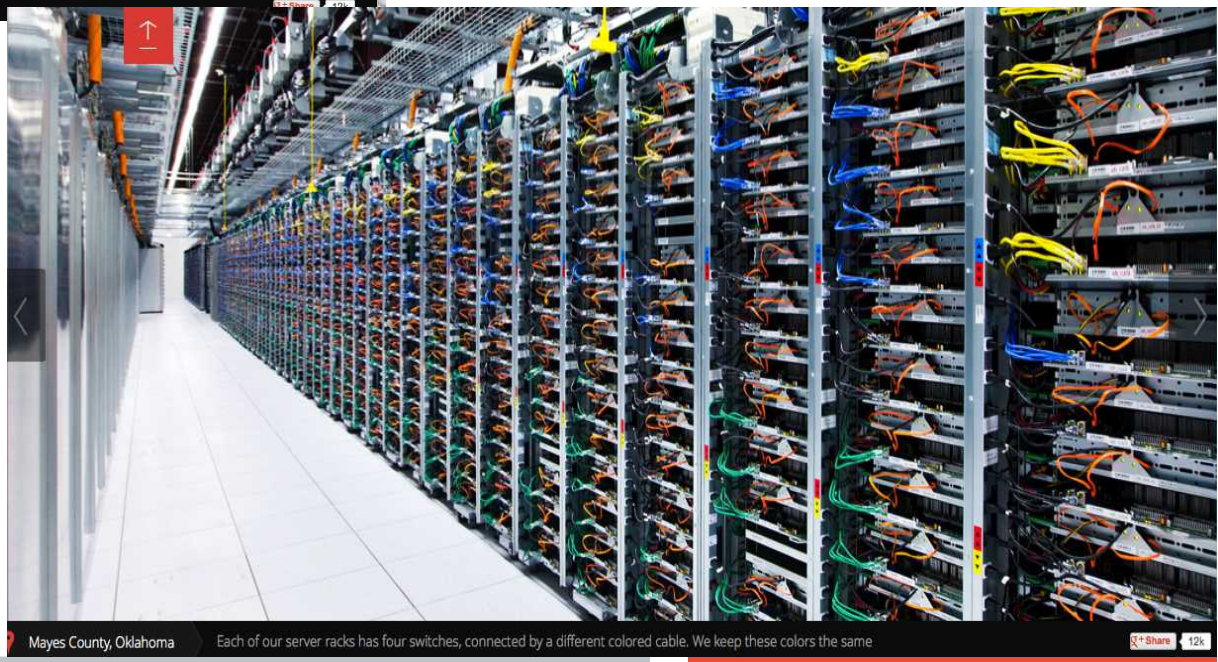
- **Virtualisation réseau** → besoin de configurer à la demande de nombreux réseaux pour interconnecter des serveurs virtuels
- Ex : VXLAN, SDN (Software Defined Networking)

PARTIE 1 : les Data Centers (DC)

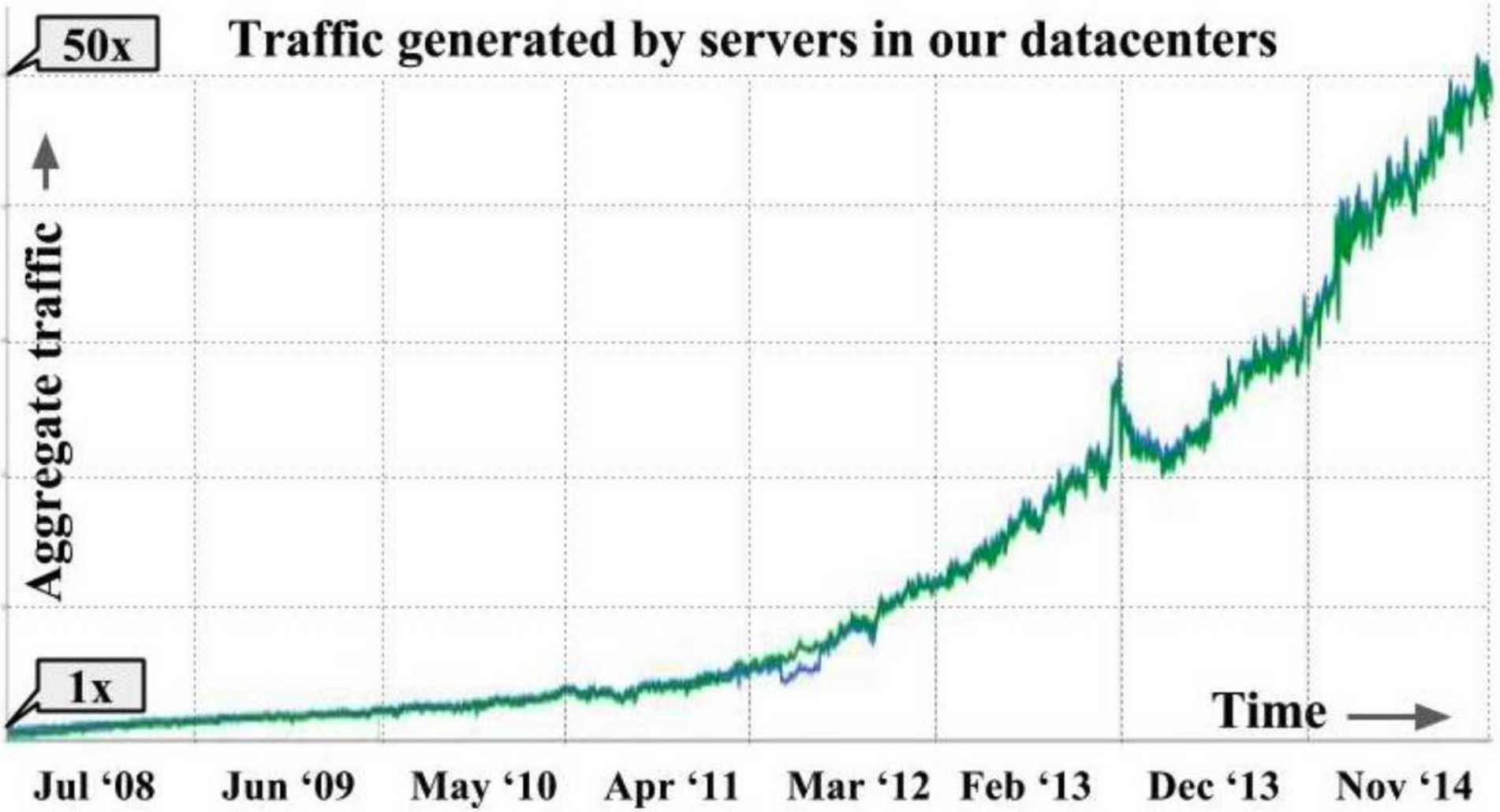
Data centers



Google Data centers



Data centers : Google – demande x 2 tous les 12/15 mois



Data centers

- **Centres de calcul**
- **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 : trafic et objectif

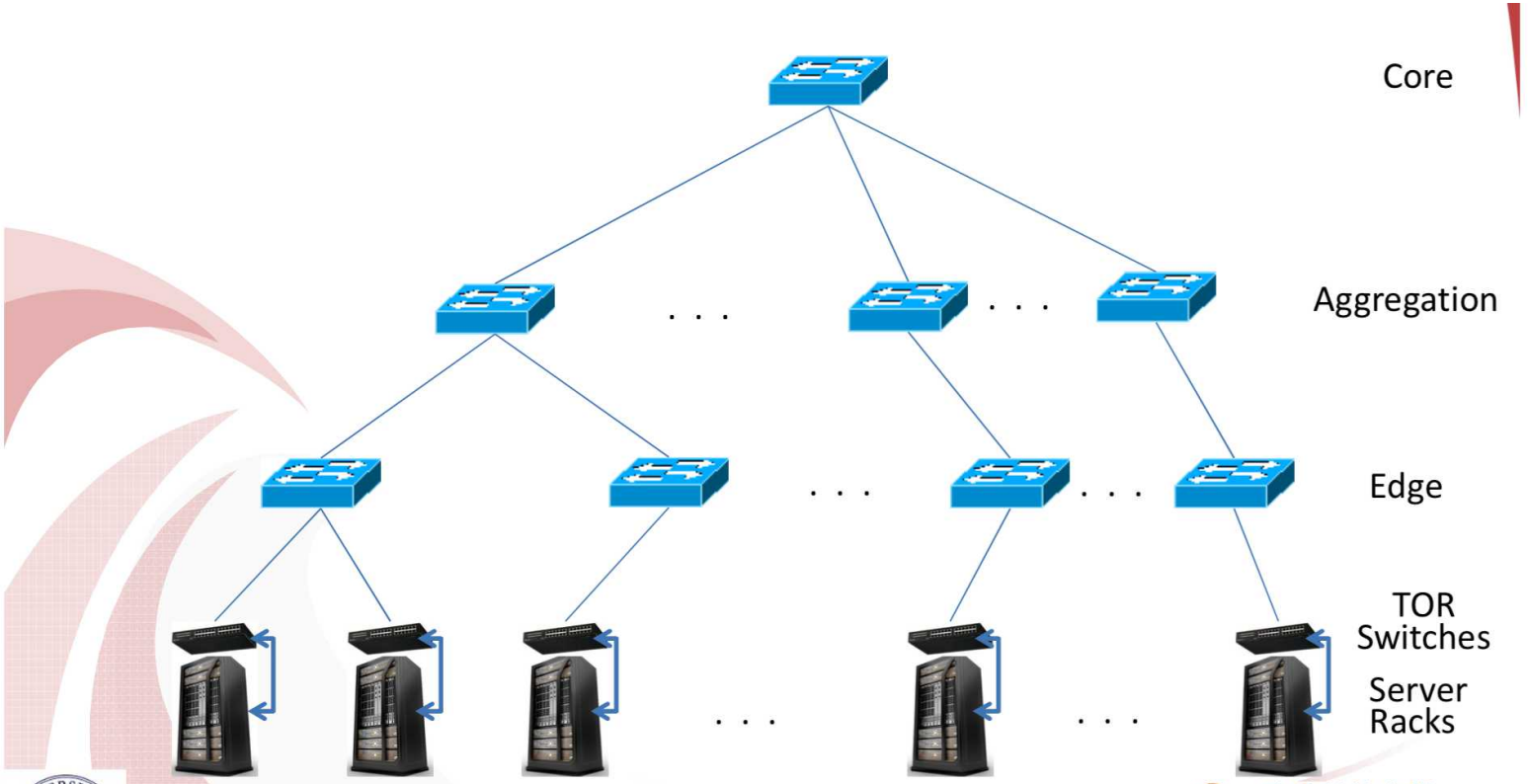
- **Deux types d'applications dominant :**
  - Services ouverts vers l'extérieur du DC
    - Ex typique : service Web
    - Attention, une simple recherche par mot-clef chez Google est envoyée sur des clusters de quelques milliers de CPU
  - Applications de type Big Data
    - Grande masse de données réparties sur les serveurs
    - Tâches de calcul envoyées vers les données car elles sont trop massives pour être envoyées à un seul ordinateur qui ferait tous les calculs
- **Objectif : on veut le même débit entre 2 serveurs (ou VM) quelle que soit leur position !!! → très forte contrainte !!!**

Data centers

- **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
  - PxCx2 processus pouvant s'exécuter simultanément → PxCx2 machines virtuelles
- **Au total : 1 rack de 40 serveurs physiques quadri-processeurs quadri-cores**
  - 40 x 4 x 4 x 2 = 1280 VM par rack → gros besoin réseau

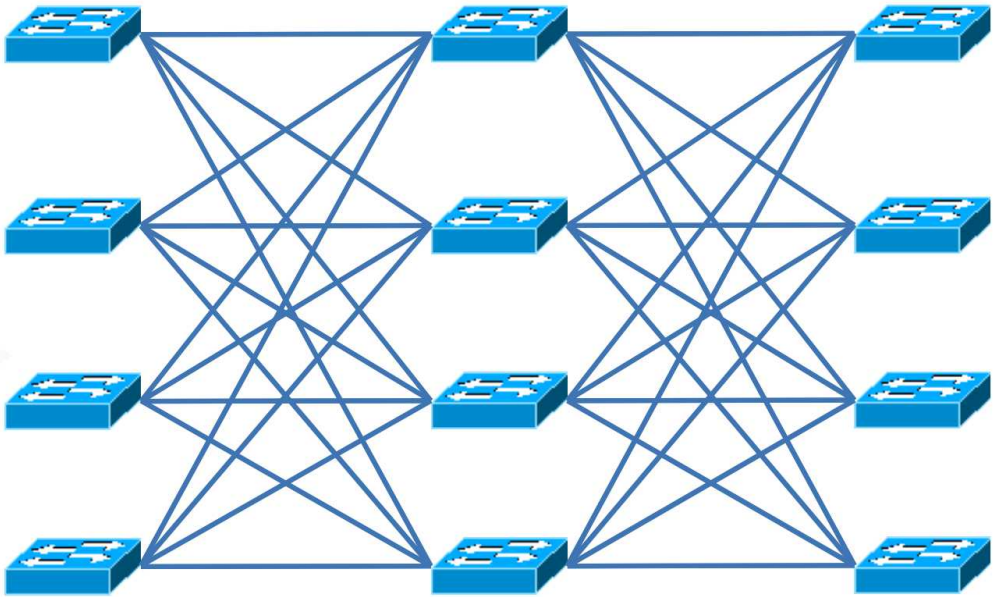
Data centers : architecture

- **Ce qui ne marche pas : l'arbre simple**



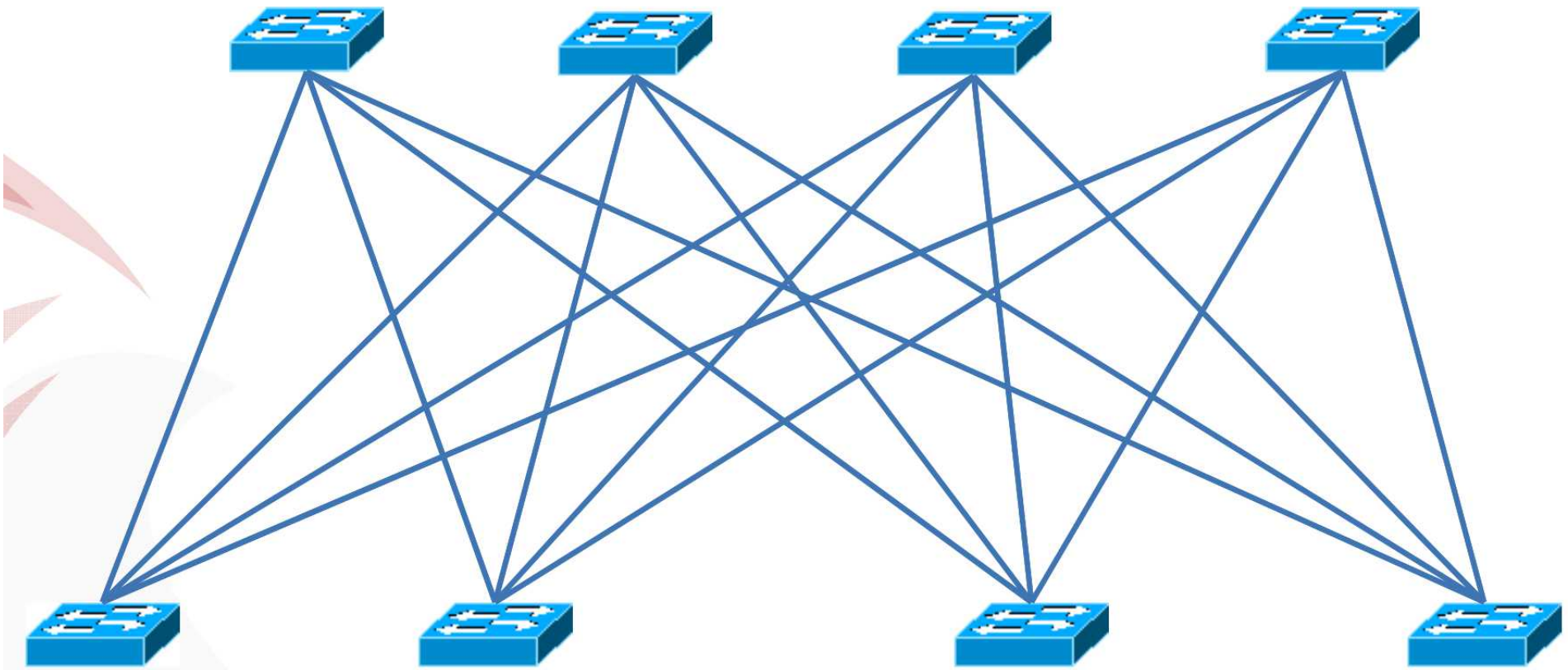
Data centers - architectures

- **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 - architectures

- **Variante fréquente : leaf and spine (feuille et dorsale)**



Data centers : générations de DC 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.

Google DC : firehose 1.0 generation

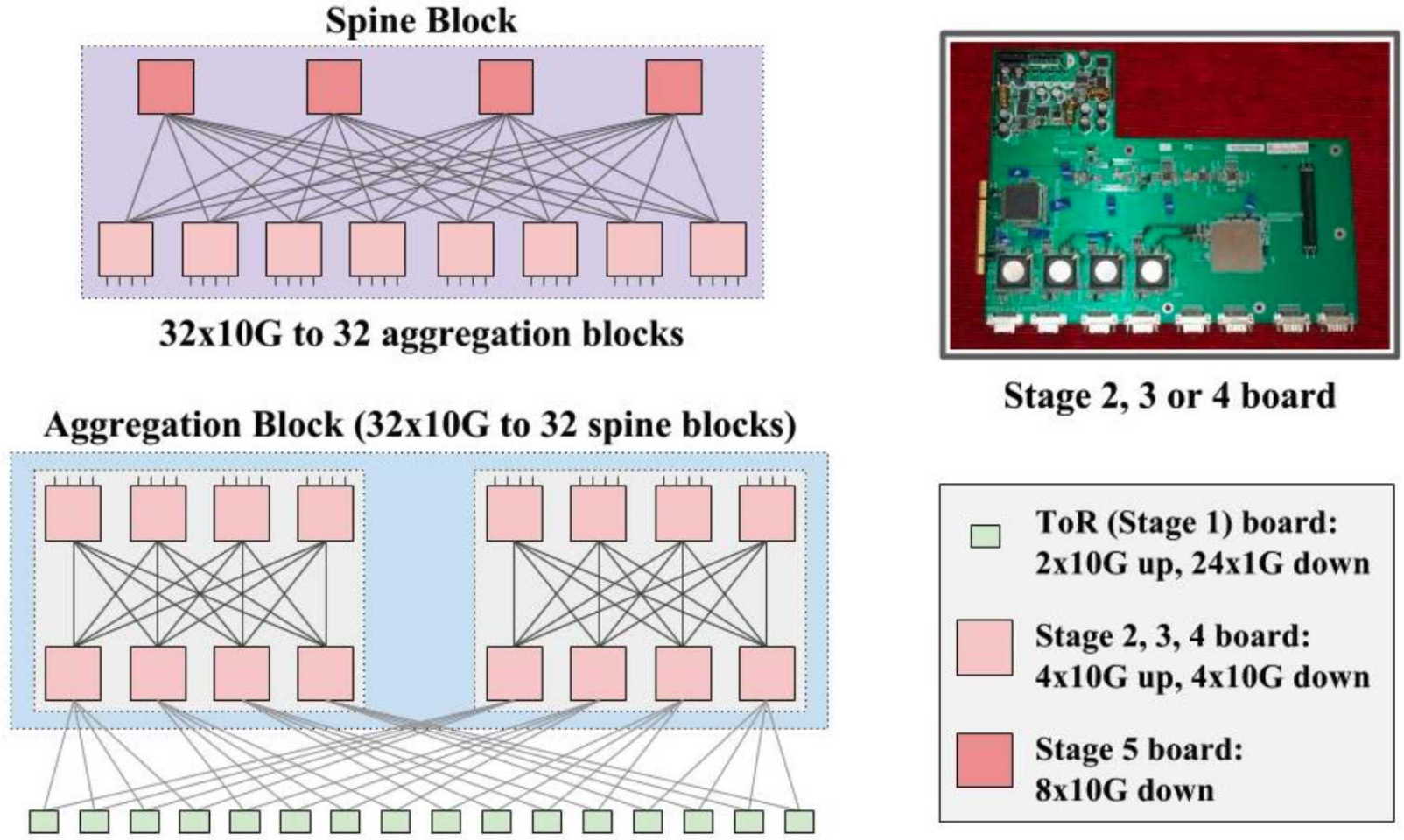


Figure 5: Firehose 1.0 topology. Top right shows a sample 8x10G port fabric board in Firehose 1.0, which formed Stages 2, 3 or 4 of the topology.

Data centers : quelques réflexion de Google

- **Article conférence Sigcomm 2015 (disponible en ligne)**

Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network

Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat
Google, Inc.
jupiter-sigcomm@google.com

Data centers : quelques réflexion de Google

- **Pour construire des réseaux de DC, les switches utilisées pour les WAN (très haute fiabilité) ne sont pas adaptés**
  - Approche :
    - Plus de switchs de base
    - En fait, plus de cartes obtenues directement auprès des constructeurs, mais pas des switchs
- **Une raison pour laquelle on veut qu'une application soit distribuée sur plusieurs racks (et donc que les VMs entre ces racks communiquent entre elles) est pour le cas où il y a des pannes !!!**

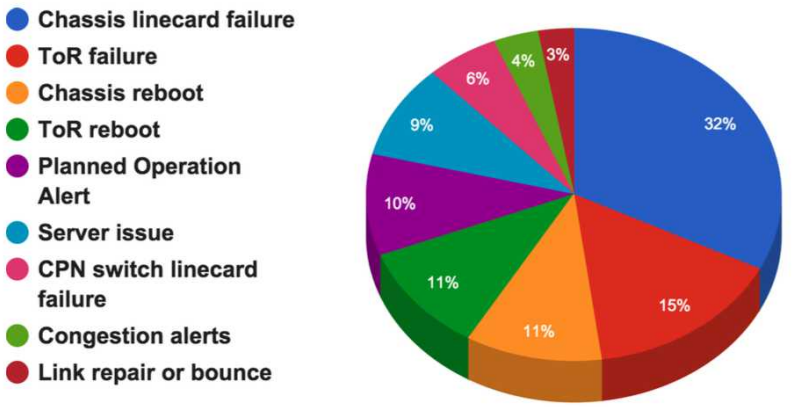


Figure 19: Alerts in Firehose/Watchtower fabrics over 9 months in '08-'09.

PARTIE 2 : la virtualisation système

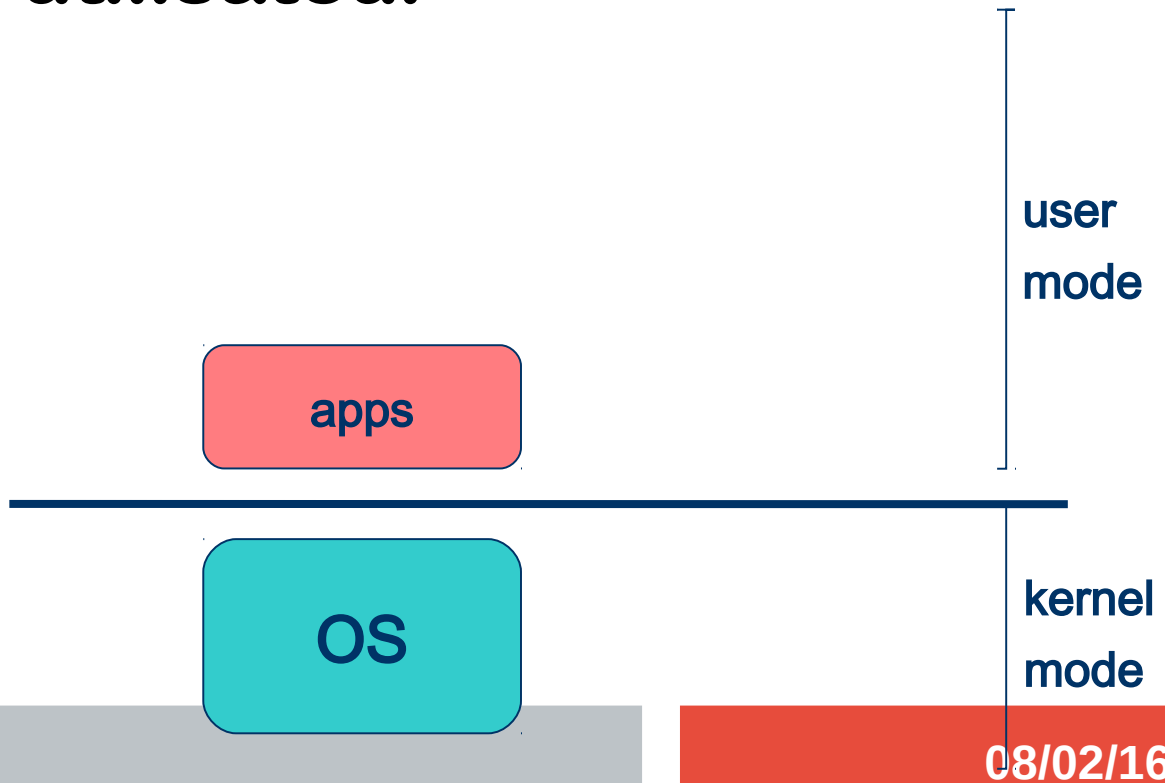
Les 2 types de virtualisation

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

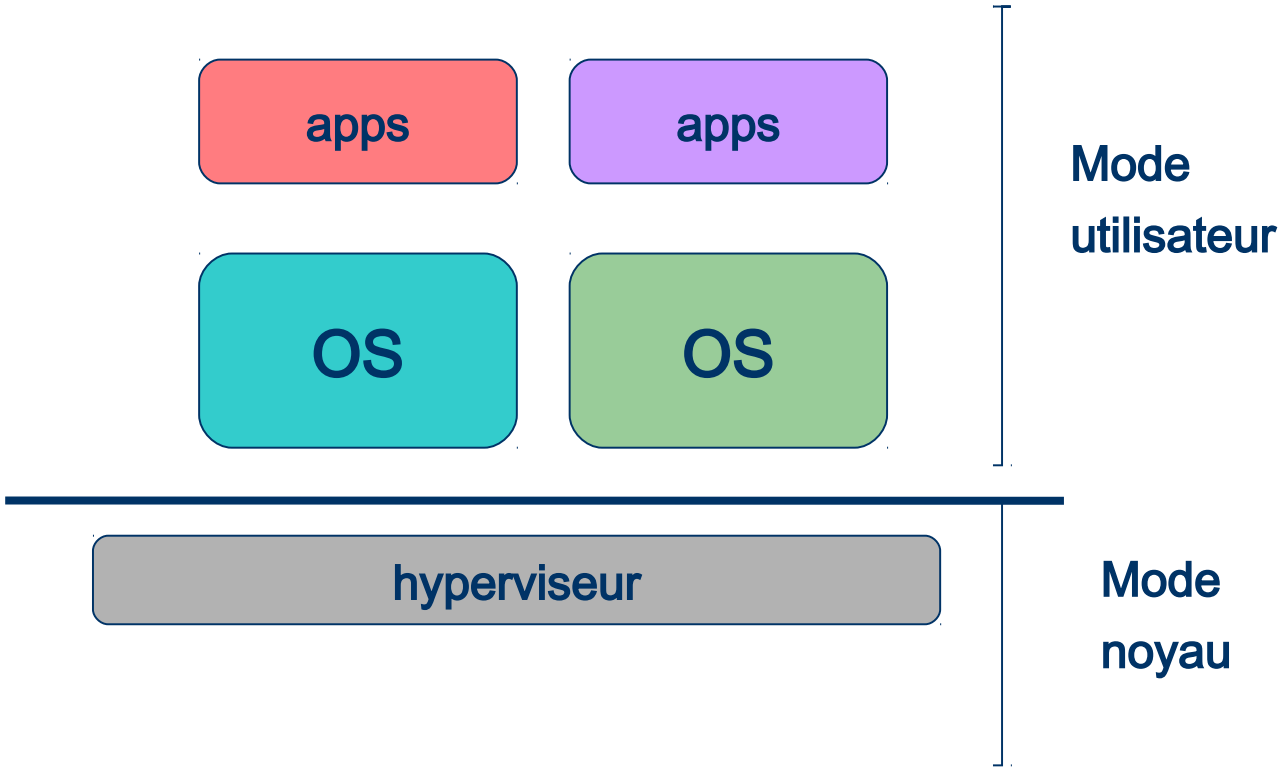
- **Virtualisation légère**
  - Containers
  - Ce que vous ferez en TP
  - Application seule

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 décroit**
  - 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

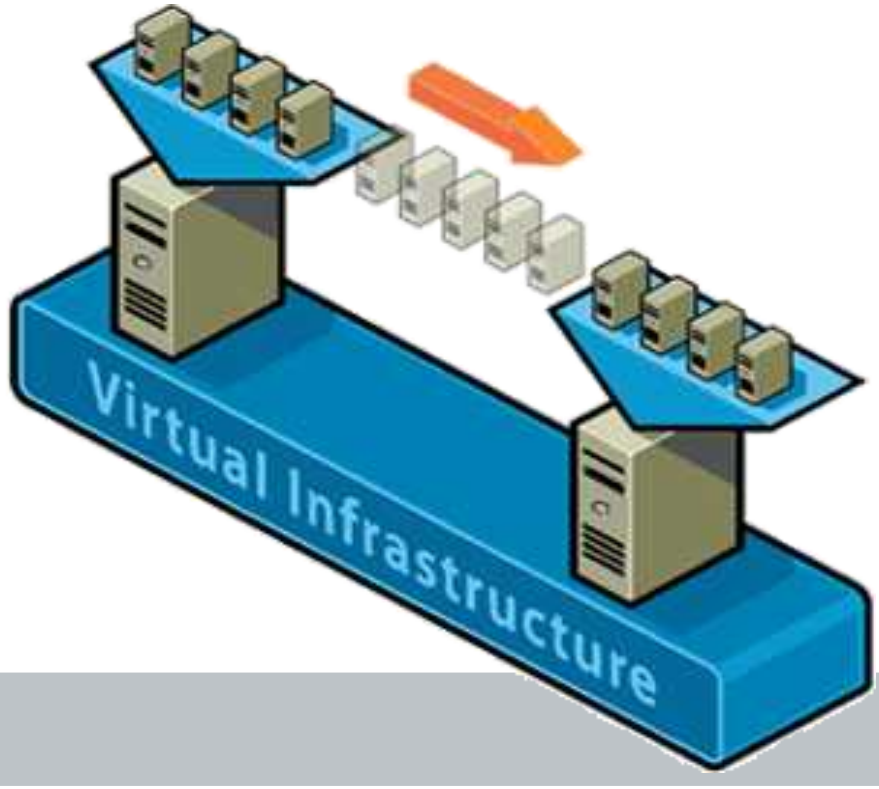
- **Plaçons-nous dans le cas où la virtualisation est effective**
- **Vous avez acheté :**
  - Deux gros serveurs
  - Des licences d'un outil de virtualisation, par exemple VMWARE
  - Les formations pour votre personnel

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 resources à ces processus par exemple de la CPU, de la mémoire
    - Similaire à ce que vous faites sous Virtualbox

Container versus Hyperviseur

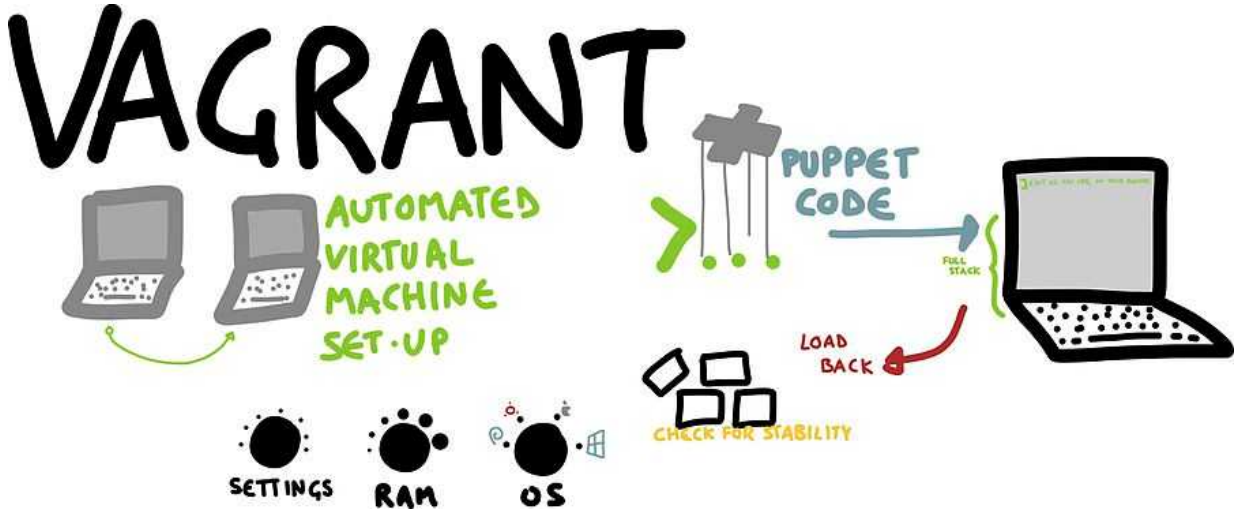
- **De l'intérieur, ressemble à une VM**
- **De l'extérieur, ressemble à des processus normaux**
- **Un container peut être une VM complète ou un groupe de processus par exemple un serveur Apache ou MySQL**

- **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

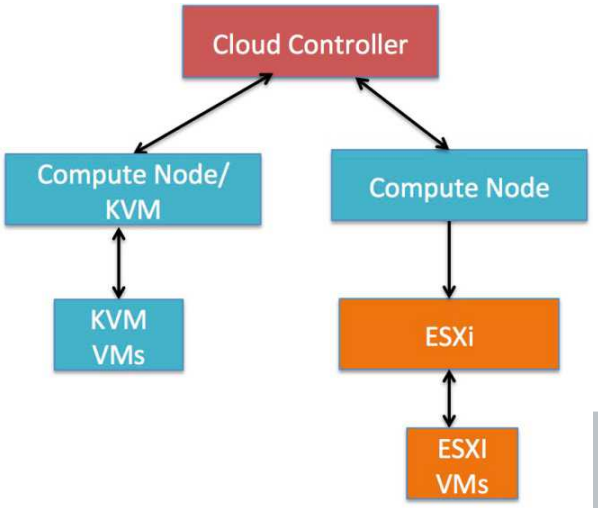


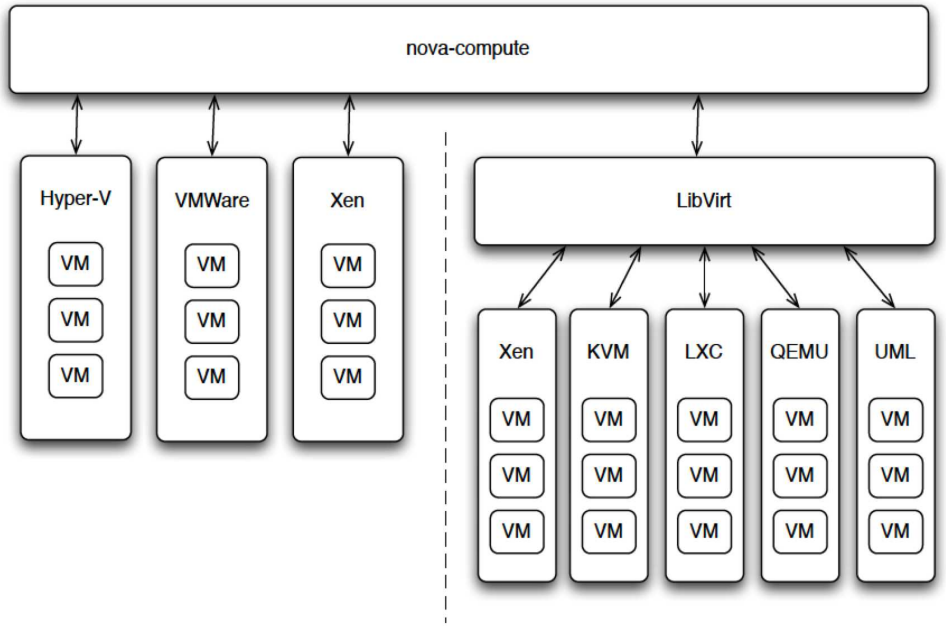
Fichier de configuration Vagrant

```
# -*- 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

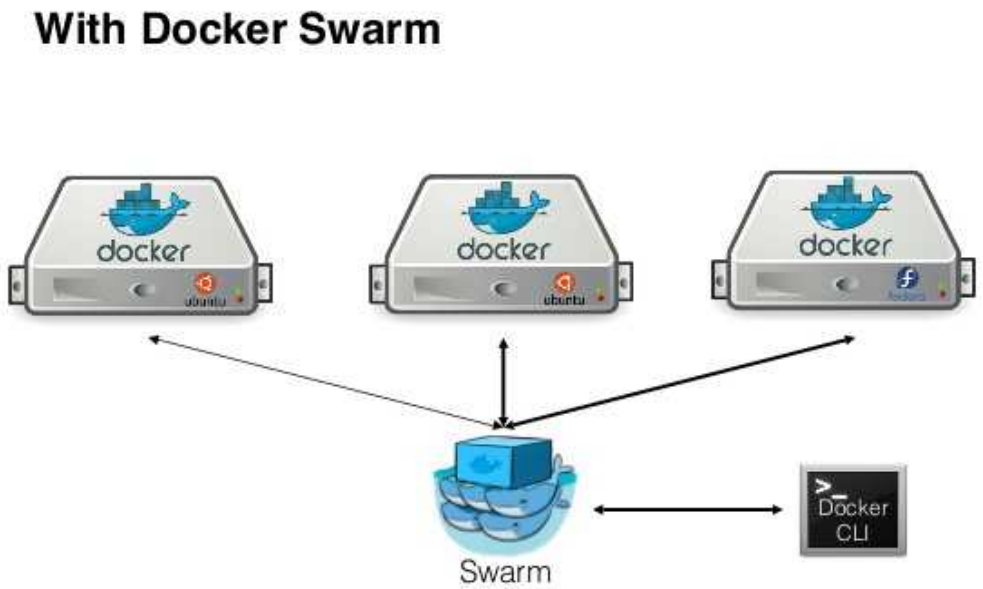
- Plateforme de gestion de clouds
  - Openstack
    - Chaque fonction (managementdef VM, réseaux, volumes, identités) est un composant (au final un service Linux)
      - Nova: compute nodes (hyperviseurs)
      - Cinder : volumes
      - Neutron : réseaux
    - Les composants interagissent via une API REST
    - Les nœuds Compute (serveurs physiques) peuvent faire tourner des hyperviseurs différents : KVM, Xen, Citrix, etc





Gestion de containers

- **Orchestration de containers**
  - Serveur unique: Docker, LXC
  - Several multiplesl: Docker Swarm
- **Orchestration avancés : Kubernetes, Swarm, Mesos**



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 vitualisation function**
  - Idée : mettre les functions réseaux du type NAT, répartiteur de charge, pare-feu dans des VMs
- **La virtualisation dans les réseaux mobiles**
  - CloudRAN (RAN = Radio Access Network)