

TABLE DES MATIERES

Virtualisation	1
I- Introduction	1
II- Virtualbox	2
III- Proxmox.....	4
IV- Conteneurisation.....	4

VIRTUALISATION

I- INTRODUCTION

1) Réponse à un problème

La virtualisation consiste à exécuter sur une machine hôte, dans un environnement isolé, des systèmes d'exploitation (virtualisation système) ou des applications (virtualisation applicative).

Liste non exhaustive des usages :

- tester un OS sans l'installer sur une machine physique
- s'assurer de la compatibilité d'une application sur différents systèmes
- utiliser une application dans un environnement particulier
- utiliser les fonctionnalités spécifiques d'un OS
- simuler une 2e machine et faire des tests de communication
- créer un petit réseau de plusieurs machines
- simuler des environnements multi-OS
- mutualiser des serveurs physiques et faire des économies
- partager un serveur physique en plusieurs serveurs
- rendre une architecture hautement disponible en améliorant sa tolérance aux pannes
- migrer facilement des serveurs
- déployer des serveurs virtuels à la volée
- faciliter les opérations de maintenance

2) Machine virtuelle

Une machine virtuelle est une machine qui n'est pas physique. Elle utilise les ressources de la machine physique (hôte) sur laquelle elle est installée.

Le principe de la virtualisation repose sur le partage des ressources : processeur, mémoire, stockage, périphériques.

Une VM est un ordinateur dans un ordinateur !

3) Hyperviseur

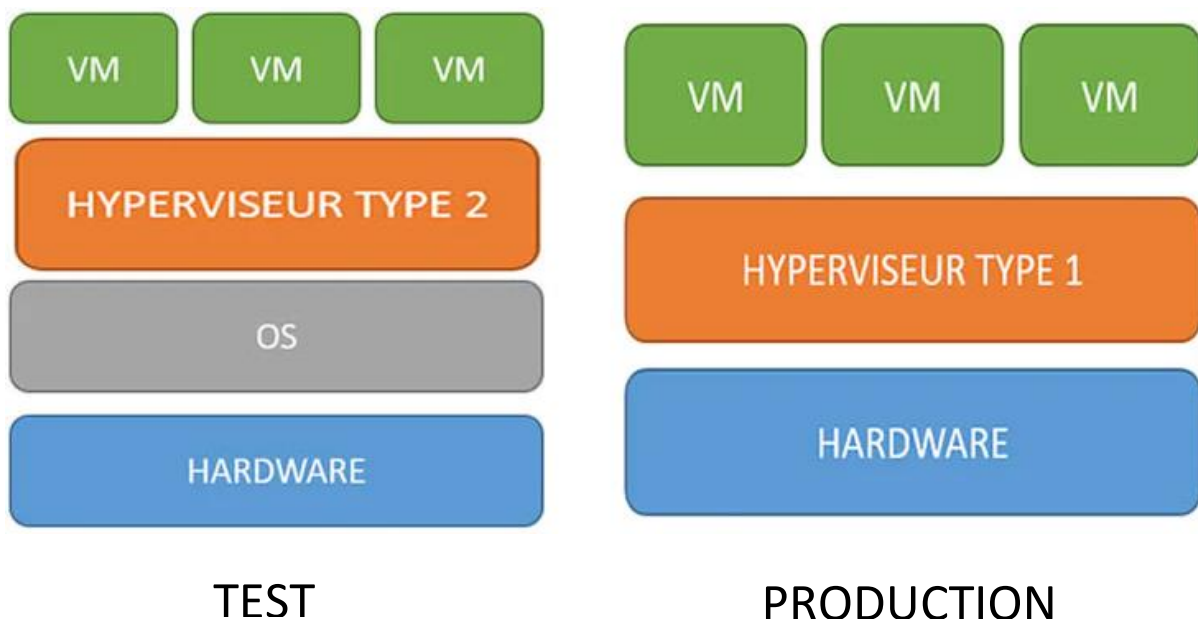
Nativement, les machines ne sont pas capable de partager leur ressources physiques : c'est le rôle de l'hyperviseur.

Il a deux rôles majeurs :

- créer des ressources virtuelles pour chaque VM,
- répartir ses propres ressources physique

Deux types d'hyperviseurs :

- Type1 : natif (bare métal), c'est un logiciel qui s'exécute directement sur une plateforme matérielle. Il contient un système d'exploitation allégé et optimisé.
Exemples : Xen(libre), Proxmox ou KVM(libre), Hyper-V(Microsoft), ESX(VMware).
- Type 2 : Hébergé (hosted), c'est un logiciel qui s'exécute au-dessus d'un système d'exploitation.
Exemples : Virtual PC et Virtual Server(Microsoft), VirtualBox(Libre).



II- VIRTUALBOX

1) Prérequis

- Une machine (Windows, Linux, Mac)
- Minimum 4Go (dont 512Mo pour Virtualbox)
- Minimum 30 Go stockage (20 pour windows)

- Un processeur qui prend en charge la virtualisation et penser à activer dans le BIOS AMD-V ou VT-X pour intel).

2) Créer une VM

Avant tout, il faut se poser quelques questions :

- Quel OS ?
- Combien de RAM ?
- Combien de vCPU ?
- Quelle taille de disque ?
- Quelle connexion réseau ?

Deux moyens de créer une VM :

- Créer une VM nue sans OS et installer un système à partir d'une image ISO.(long à installer et paramétrer)
- Télécharger directement une VM avec son OS

<https://virtualboxes.org/images>

<https://sourceforge.net/projects/virtualboximage/files>

<https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/>

Les disques durs virtuels sont aux formats :

- vhd (historique Microsoft VirtualPC)
- vdmk (historique VMware)
- vdi (Virtualbox)

Ces formats contiennent uniquement le disque dur, mais pas la configuration de la VM. Ils sont au format dynamique (thin provisioning) : leur taille augmente ou diminue jusqu'au maximum autorisé.

Les fichiers d'export/import de VM :

- OVF ou vbox : uniquement le descriptif de la configuration.
- OVA : contient le descriptif et le disque.

3) Réseau

Il existe plusieurs modes réseau :

NAT - Pour utiliser une VM seule avec internet, mais pas de communication possible avec d'autre VM : elles ont toutes la même IP.

Accès par pont – Pour que la VM soit accessible sur le réseau avec sa propre IP et puisse communiquer avec les autres. [Penser à les mettre en automatique pour quelles prennent une adresse IP sur le même LAN que l'hôte.](#)

Réseau interne – Pour utiliser plusieurs VM pour simuler des réseaux privés sont accès à l'hôte et à l'extérieur. [Toutes les machines d'un sous réseau doivent être sur le même nom de réseau et leur mettre une IP fixe si il n'y a pas de serveur DHCP parmi elles.](#)

Réseau NAT – Même utilisation que le NAT mais avec plusieurs VM qui peuvent communiquer.

Réseau privé d'hôtes - Semblable au réseau interne mais avec communication avec l'hôtes.

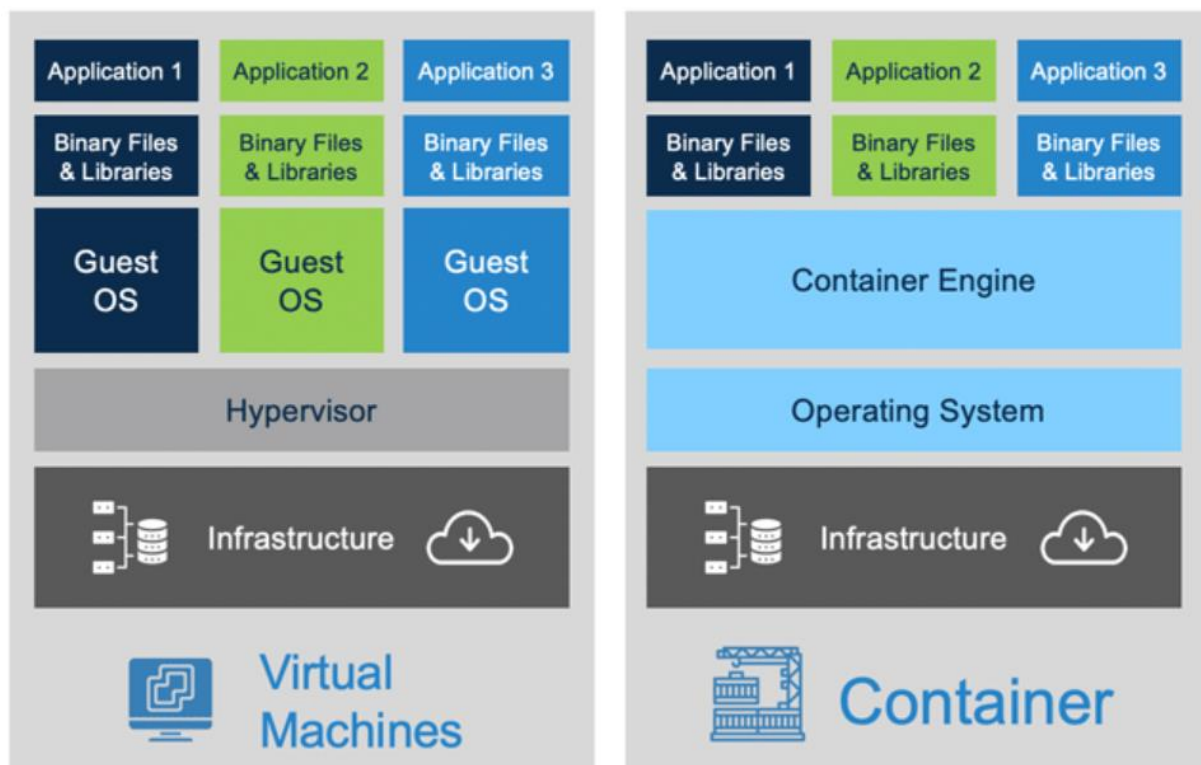
Une VM peut avoir plusieurs cartes réseau et chacune dans des modes différents.

III- PROXMOX

IV- CONTENEURISATION

1) Conteneurs

Un conteneur linux est un processus ou un ensemble de processus isolés du reste du système. C'est de la virtualisation légère : on ne virtualise pas les ressources, on isole simplement les processus. Le conteneur partage donc les ressources avec le système hôte.



Les conteneurs, comme OpenVZ, LXC, apportent une isolation importante des processus systèmes. Ils partagent entre eux le noyau linux (pas Windows ni mac).

Avantages :

- N'utilise que les ressources nécessaires (le reste n'est pas verrouillé)
- Démarre rapidement : pas de réservation des ressources
- Donne plus d'autonomie aux développeurs
- Réduits les coûts
- Améliore le cycle de déploiement
- Répond au besoin de scalabilité

2) Docker

Crée par la société DotCloud pour une PAAS. Finalement en mars 2013, création de l'entreprise Docker IMC et placement de Docker en open source. Dans la vision de Docker, un conteneur ne fait tourner qu'un seul processeur. Exemple : 3 conteneurs pour un serveur web avec apache, Mysql et php. Avec Docker, les conteneurs deviennent immuables. Et stateless. Une base de données est statefuls : elle stocke son état ; si on la redémarre, on la retrouve dans le même état. Stateless, c'est l'inverse : le protocole http est stateless ; si on redémarre un serveur apache, les requêtes précédentes on était oubliées...

Le système Docker est sur une base debian.

3) Installation

Docker Inc distribue 3 versions différentes de Docker :

- Docker Community Edition (Linux Only, gratuite)
- Docker Desktop (Mac ou Windows, gratuite)
- Docker Entreprise (Linux only, payante)

Le Docker Hub est un service fourni par Docker Inc pour stocker les images (comme Github). Il faut créer un compte (pas obligatoire pour Linux).

Les versions Edge sont des versions bêta.

Sur Windows, l'utilitaire va créer une machine virtuelle Hyper-V pour installer Docker. Il faut obligatoirement Windows Pro ou Entreprise.

4) Utilisation

En créant un compte DockerHub, on a accès à la Registry officielle de Docker : logiciel qui permet de partager des images à d'autres personnes. C'est un composant majeur qui permet :

- aux développeurs de distribuer des images prêtes à l'emploi et de le versionner avec un système de tags ;
- à des outils d'intégration en contenu de jouer une suite de tests ;
- à des systèmes automatisés de déployer les applications sur les environnements de développement et de production.

Commande pour démarrer un premier conteneur :

Docker run hello-world

Quand on lance cette commande, le démon Docker va chercher l'image hello-world en local, sinon sur la registry officielle.

On commence par télécharger la VM Debian, on lui attribue un nom et une adresse ip avec les ports :

`docker run name serwebstatique -it debian:stretch`

J'en ai également profiter pour mettre a jour la time zone

```
Invite de commandes - docker run --name servwebstatic -it debian:stretch

C:\Users\alexis>docker run --name servwebstatic -it debian:stretch
root@66bf921f1e32:/# dpkg-reconfigure tzdata
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (Can't locate Term/ReadLine.pm in @INC (you may need to install the Term::ReadLine module) (@INC contains: /etc/perl /usr/local/lib/x86_64-linux-gnu/perl/5.24.1 /usr/local/share/perl/5.24.1 /usr/lib/x86_64-linux-gnu/perl/5.24 /usr/share/perl5 /usr/lib/x86_64-linux-gnu/perl/5.24 /usr/share/perl/5.24 /usr/local/lib/site_perl /usr/lib/x86_64-linux-gnu/perl-base) at /usr/share/perl5/Debconf/FrontEnd/Readline.pm line 7.)
debconf: falling back to frontend: Teletype
Configuring tzdata
*****

Please select the geographic area in which you live. Subsequent configuration questions will narrow this down by
presenting a list of cities, representing the time zones in which they are located.

 1. Africa      3. Antarctica  5. Arctic    7. Atlantic  9. Indian    11. SystemV  13. Etc
 2. America    4. Australia  6. Asia      8. Europe    10. Pacific  12. US
Geographic area: 8

Please select the city or region corresponding to your time zone.

 1. Amsterdam  10. Bucharest  19. Isle_of_Man  28. Luxembourg  37. Paris      46. Simferopol  55. Vaduz
 2. Andorra    11. Budapest  20. Istanbul    29. Madrid      38. Podgorica   47. Skopje      56. Vatican
 3. Astrakhan  12. Busingen  21. Jersey      30. Malta       39. Prague     48. Sofia       57. Vienna
 4. Athens     13. Chisinau  22. Kaliningrad 31. Mariehamn   40. Riga        49. Stockholm  58. Vilnius
 5. Belfast    14. Copenhagen 23. Kiev        32. Minsk       41. Rome        50. Tallinn     59. Volgograd
 6. Belgrade   15. Dublin    24. Kirov       33. Monaco      42. Samara      51. Tirane      60. Warsaw
 7. Berlin     16. Gibraltar 25. Lisbon      34. Moscow      43. San_Marino  52. Tiraspol    61. Zagreb
 8. Bratislava 17. Guernsey   26. Ljubljana   35. Nicosia     44. Sarajevo    53. Ulyanovsk   62. Zaporozhye
 9. Brussels   18. Helsinki  27. London      36. Oslo        45. Saratov     54. Uzhgorod    63. Zurich
Time zone: 37

Current default time zone: 'Europe/Paris'
Local time is now:      Mon Dec 12 11:08:31 CET 2022.
Universal Time is now:  Mon Dec 12 10:08:31 UTC 2022.

root@66bf921f1e32:/#
```

Maintenant je lance ma machine et je la mets à jour

```
Invite de commandes - docker exec -it servwebstatic /bin/bash

C:\Users\alexis>docker commit servwebstatic af/debian:stretch-apache2
sha256:5b647508081392cf815376a670b926d569b7a289fce41c2631a9d4ac7d233848

C:\Users\alexis>docker exec -it servwebstatic /bin/bash
root@66bf921f1e32:/# apt update
Ign:1 http://deb.debian.org/debian stretch InRelease
Get:2 http://security.debian.org/debian-security stretch/updates InRelease [59.1 kB]
Hit:3 http://deb.debian.org/debian stretch-updates InRelease
Hit:4 http://deb.debian.org/debian stretch Release
Fetched 59.1 kB in 0s (114 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
root@66bf921f1e32:/# apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2 is already the newest version (2.4.25-3+deb9u13).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@66bf921f1e32:/#
```

Maintenant, il nous faut télécharger les outils nécessaires pour la gestion des paquets :

apt-get install apt-transport-https ca-certificates curl gnupg2 software-properties-common

Suite à quoi nous sommes prêts à télécharger apache 2 avec la commande suivante :

apt-get install apache2

Une fois cela fait, on sort du shell Debian et on met dans le cmd la commande suivante :

docker commit sweet_austin np/debian:stretch-apache2

On se dirige maintenant vers /etc/apache2 et le fichier ports.conf pour modifier le port d'écoute de notre serveur apache.

On veut le port 80, donc on garde la valeur du fichier ports.conf

```
# cat ports.conf

# If you just change the port or add more ports here, you
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Nous pouvons maintenant lancer le serveur Apache2 :

```
# service apache2 start
Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message

# service apache2 status
apache2 is running.
```

5) Le réseau dans Docker

Prérequis : démarrer Docker Desktop (Mac ou Windows) ou Docker Engine (Linux).

Faire un test dans un terminal : docker version

Quelle est votre version ?

C'est la 4.15.0

Connectivité : par défaut, il y a trois réseaux préconfigurés par Docker

`docker network ls`

Lesquels ?

Bridge / host / none

Bridge : réseau par défaut

Vérification :

`docker run -d --name=nginx nginx:alpine`

`docker inspect nginx | grep -A 1 Networks`

Quelle est votre adresse IP ? Réseau ? Masque ?

On utilise `docker inspect nginx > temp.txt` pour sortir dans un fichier puis on l'ouvre

```
"Networks": {
  "bridge": {
    "IPAMConfig": null,
    "Links": null,
    "Aliases": null,
    "NetworkID": "6b6197f898874d542a44048e6e8207a436c89664eebf821a703e6263c8edefff",
    "EndpointID": "3c1ebebe8e6c43d962fe587ffa624c8c56e1928bf98be5b21a790380d4f27ef1",
    "Gateway": "172.17.0.1",
    "IPAddress": "172.17.0.2",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "MacAddress": "02:42:ac:11:00:02",
    "DriverOpts": null
  }
}
```

On démarre un deuxième conteneur :

`docker run -it --name=alpine alpine`

`ip a show eth0`

Quelle adresse IP ?

172.17.0.3

Ping du nginx

Le ping marche

```
/ # ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2): 56 data bytes
64 bytes from 172.17.0.2: seq=0 ttl=64 time=0.195 ms
64 bytes from 172.17.0.2: seq=1 ttl=64 time=0.136 ms
64 bytes from 172.17.0.2: seq=2 ttl=64 time=0.124 ms
64 bytes from 172.17.0.2: seq=3 ttl=64 time=0.130 ms
64 bytes from 172.17.0.2: seq=4 ttl=64 time=0.109 ms
64 bytes from 172.17.0.2: seq=5 ttl=64 time=0.130 ms
64 bytes from 172.17.0.2: seq=6 ttl=64 time=0.130 ms
64 bytes from 172.17.0.2: seq=7 ttl=64 time=0.124 ms
^C
--- 172.17.0.2 ping statistics ---
8 packets transmitted, 8 packets received, 0% packet loss
round-trip min/avg/max = 0.109/0.134/0.195 ms
```

[Récupérer la page d'accueil de l'autre](#)

`wget -O ip_nginx`

Commande pour démarrer un premier conteneur :

`docker run hello-world`

Quand on lance cette commande, démon docker va chercher l'image hello-world en local, sinon sur la registry officielle.

Dans cet exemple, le conteneur a démarré, affiché du contenu et s'est arrêté. Si vous souhaitez que le conteneur reste allumé, ajouter l'argument : `--detach` ou `-d`

Pour l'arrêter :

`Docker stop id_retourné_par_le_docker_num`

Pour le supprimer :

`Docker rm id_retourné_par_le_docker_num`

Pour récupérer une image sans lancer le conteneur : `docker pull hello-world`

Pour afficher l'ensemble des conteneurs :

`Docker ps -a`

Pour lister les images présentes en local :

`Docker image -a`

```
C:\Users\fruct>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
6036ce78990c	hello-world	"/hello"	2 minutes ago	Exited (0) 2 minutes ago	
magical_napier					
3a3252201c4d	debian	":stretch"	24 minutes ago	Created	
servWebStatic					
89b5e72698cb	alpine	"/bin/sh"	28 hours ago	Exited (255) 22 hours ago	
alpine					
118c12913661	nginx:alpine	"/docker-entrypoint...."	28 hours ago	Exited (255) 22 hours ago	80/tcp
nginx					
8ceb6f4ad1ef	phpmyadmin	"/docker-entrypoint...."	13 days ago	Up 5 hours	0.0.0.0:8000->80/tcp
phpMyAdmin_sio23					
8458f4900a06	initiationphpsql-web	"docker-php-entrypoi..."	13 days ago	Up 5 hours	0.0.0.0:8080->80/tcp
cours_php_sio23					
f8ff861d79e4	mysql	"docker-entrypoint.s..."	13 days ago	Up 5 hours	3306/tcp, 33060/tcp
dataBase_sio23					

Pour faire le ménage :

`Docker system prune`