

Cloud Computing : Adaptability and Autonomic Management

Claire Gageot - Arnaud Guiller



Lab 1 : Introduction to Cloud Hypervisors

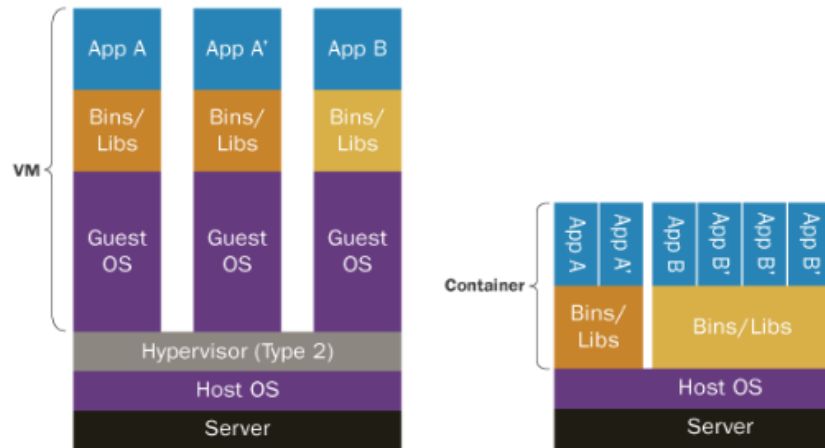
Partie Théorique :

1. Différences fondamentales entre les types d'hôtes de virtualisation (VM et CT)

Le principe de **Machine Virtuelle (VM)** repose sur l'utilisation d'un hyperviseur. Les **hyperviseurs** sont des plateformes de virtualisation qui permettent de faire fonctionner plusieurs systèmes d'exploitation sur une même machine physique, et en parallèle. C'est un logiciel (appelé Virtual Machine Manager) qui permet la création, la gestion et la fermeture des VMs. Les ressources matérielles de la machine sont alors simulées, notamment le processeur, le disque dur ou encore la mémoire. Avec ce mode de fonctionnement, les applications exécutées sont totalement isolées, et exécutées indépendamment du matériel et les logiciels sur lesquels repose la machine physique.

Dans le cas d'un **conteneur** (en anglais : container ; on abrégera **CT**), le principe est différent. On ne parle plus de virtualisation mais de conteneurisation : il n'y a pas de simulation du

système d'exploitation, et chaque conteneur est directement exécuté sur l'OS de la machine hôte, à partir de différentes instances. Il existe une isolation entre les processus. Enfin, on retiendra que l'hyperviseur est assurée par un moteur de conteneurisation, le plus célèbre étant Docker.



Différences d'ordre général :

Grâce à son mode de fonctionnement, un conteneur entraîne l'exécution d'un seul kernel sur la machine physique : la conteneurisation présente donc l'avantage de n'entraîner qu'un léger overhead en comparaison aux VMs qui sont plus lourdes, et consomment plus de ressources de manière générale. Si les VMs et la conteneurisation offrent tous les deux une isolation entre les exécutions, celle offerte par les conteneurs est en revanche plus faible. La sécurité garantie par les VMs est plus importante que celle des conteneurs. Enfin, on notera que la conteneurisation est moins gourmande en espace sur un disque et que la migration est donc plus rapide que pour les VMs.

Différences du point de vue d'un développeur :

	VM	CT
Utilisation des ressources	Ressources allouées donc plus grande liberté d'utilisation, mais peut rapidement atteindre 100% d'utilisation	Ne consomme que les ressources nécessaires
Sécurité	Meilleure isolation donc sécurité élevée	Plus faible
Performances	Le changement entre les différents OS des VMs peut entraîner des latences	Meilleur temps de réponse car utilisation d'un seul kernel
Environnement de développement	Identique à l'environnement original	Nécessite la mise en place de fichiers additionnels, et la configuration avec des dépendances et des bibliothèques

Différences du point de vue d'un administrateur d'infrastructures :

	VM	CT
Coûts	Elevés, car nécessite des ressources dédiées.	Plus accessible.
Sécurité	Niveau élevé	Nécessite la mise en place de mesures de sécurité plus importantes car le CT peut accéder à des fichiers binaires et des librairies partagées avec des applications du système
Performances	Nécessité l'allocation de ressources : plus compliqué à gérer	Plus facile à gérer et utilisation moindre d'espace et de ressources
Administration	Complexe	Administration simplifiée grâce à une interface d'orchestration

2. Similarités et différences entre les types de CTs

Il existe différents types de conteneurs sur le marché, dont les caractéristiques sont différentes. Ils peuvent être comparés sur la base des critères suivants :

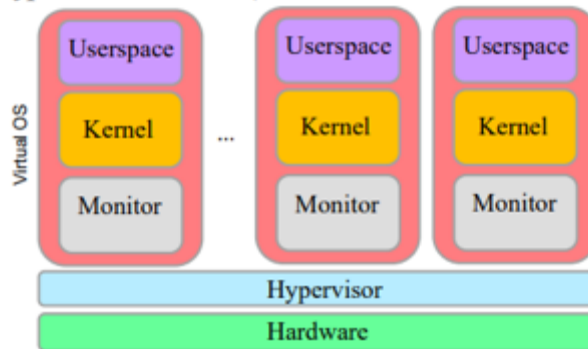
- **Niveau de conteneurisation**, qui caractérise la nature de ce qui sera conteneurisé : les conteneurs peuvent englober seulement les applications ou carrément le système complet avec plusieurs applications.
- **Panel d'outils** : différents niveaux de fonctionnalités sont offerts aux développeurs et aux administrateurs, de la simple ligne de commande à l'interface graphique avancée. Les fonctionnalités peuvent notamment concerner la migration, les sauvegardes, etc.
- **Isolation application / ressources** : impactera le nombre de fichiers partagés entre les applications, et donc le niveau de sécurité. Plus de fichiers sont partagés, plus les risques de vulnérabilité sont élevés, et les appels externes sont aussi multipliés ce qui entraîne une plus grande consommation de ressources.

3. Différences fondamentales entre les architectures des types d'hyperviseurs

Il existe 2 types d'hyperviseurs :

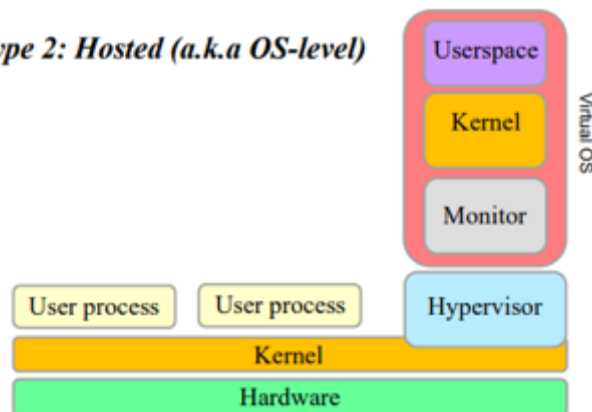
- L'hyperviseur de **type 1**, appelé hyperviseur “bare-metal” ou “natif” : il est directement exécuté sur le hardware de la machine et est optimisé pour.

– *Type 1: Bare-metal (a.k.a Native or Hardware-level)*



- L'hyperviseur de **type 2**, dit “hosted” ou “OS-level” : il est exécuté au sein d'un autre système d'exploitation et héberge des OSs appelés “guests”.

– *Type 2: Hosted (a.k.a OS-level)*



Promox est un hyperviseur de type 1 alors que Virtualbox est un hyperviseur de type 2.

4. Différences entre les deux principaux modes de connexion au réseau, pour les VMs et les CTs

Pour le **mode “bridge”**, les VMs et les CTs sont connectés au même réseau local que le PC ; ils peuvent accéder à internet et peuvent être trouvés sur le réseau de la même manière que des machines classiques.

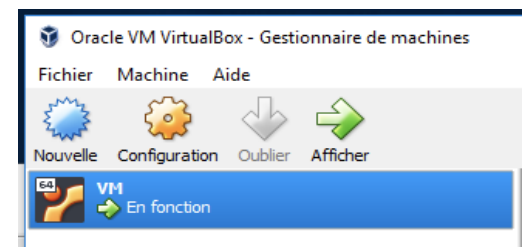
Pour le **mode “NAT”**, le conteneur se trouvent sur des réseaux IP privés, et disposent d'un routeur virtuel au sein du PC hôte, ce dernier se charge des traductions d'adresse. Les machines virtuelles et les conteneurs ne sont pas visibles depuis l'extérieur.

Partie Pratique:

1- Tâches reliées aux objectifs 4 et 5

Partie 1: Création et configuration d'une machine virtuelle

La création et la configuration de la VM ont fonctionné correctement.



Partie 2: Tester la connectivité de la VM

Dans un premier temps on identifie les adresses IP suivantes :

- VM: 10.0.2.5

```
user@tutorial-vm:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe1d:32a7 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:1d:32:a7 txqueuelen 1000 (Ethernet)
    RX packets 5247 bytes 7489827 (7.4 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 439 bytes 43543 (43.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Boucle locale)
    RX packets 56 bytes 5162 (5.1 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 56 bytes 5162 (5.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

user@tutorial-vm:~$
```

-Hôte: 10.1.1.55

```
Carte Ethernet Ethernet :  
  
Suffixe DNS propre à la connexion. . . : insa-toulouse.fr  
Adresse IPv6 de liaison locale. . . . : fe80::9152:6828:12a7:ec9%6  
Adresse IPv4. . . . . : 10.1.1.55  
Masque de sous-réseau. . . . . : 255.255.0.0  
Passerelle par défaut. . . . . : 10.1.0.254  
  
Carte Ethernet VirtualBox Host-Only Network :  
  
Suffixe DNS propre à la connexion. . . :  
Adresse IPv6 de liaison locale. . . . : fe80::11d0:f072:f815:9419%4  
Adresse IPv4. . . . . : 192.168.56.1  
Masque de sous-réseau. . . . . : 255.255.255.0  
Passerelle par défaut. . . . . :
```

Ensuite, avec la commande ping, on vérifie la connectivité :

-de la VM vers l'extérieur : cela fonctionne

```
user@tutorial-vm:~$ ping 8.8.8.8  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=51 time=7.98 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=51 time=7.97 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=51 time=8.00 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=51 time=9.17 ms  
^C  
--- 8.8.8.8 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3004ms  
rtt min/avg/max/mdev = 7.978/8.284/9.172/0.520 ms
```

-de l'ordinateur d'un voisin vers la VM: cela ne fonctionne pas

-de l'hôte vers la VM: cela ne fonctionne pas (mais fonctionne lorsque l'on utilise l'adresse IP de la carte Ethernet "VirtualBox Host-Only Network")

```
U:\>ping 10.0.2.15  
  
Envoi d'une requête 'Ping' 10.0.2.15 avec 32 octets de données :  
Délai d'attente de la demande dépassé.  
Délai d'attente de la demande dépassé.  
Délai d'attente de la demande dépassé.  
Délai d'attente de la demande dépassé.  
  
Statistiques Ping pour 10.0.2.15:  
    Paquets : envoyés = 4, reçus = 0, perdus = 4 (perte 100%),  
U:\>
```

```

user@tutorial-vm:~$ ping 192.168.56.1
PING 192.168.56.1 (192.168.56.1) 56(84) bytes of data.
64 bytes from 192.168.56.1: icmp_seq=1 ttl=127 time=0.695 ms
64 bytes from 192.168.56.1: icmp_seq=2 ttl=127 time=1.37 ms
64 bytes from 192.168.56.1: icmp_seq=3 ttl=127 time=1.52 ms
^C
--- 192.168.56.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2013ms
rtt min/avg/max/mdev = 0.695/1.198/1.522/0.361 ms

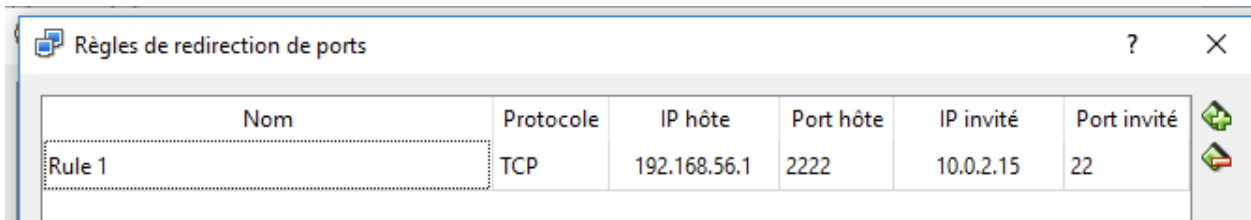
```

Nous avons une connectivité unidirectionnelle de la VM vers l'extérieur. Cette connectivité est due au fait que l'on utilise un réseau NAT. L'adresse IP publique de l'hôte ne peut être traduite en une adresse IP privée compréhensible par la VM.

Partie 3: Configurer la connectivité manquante

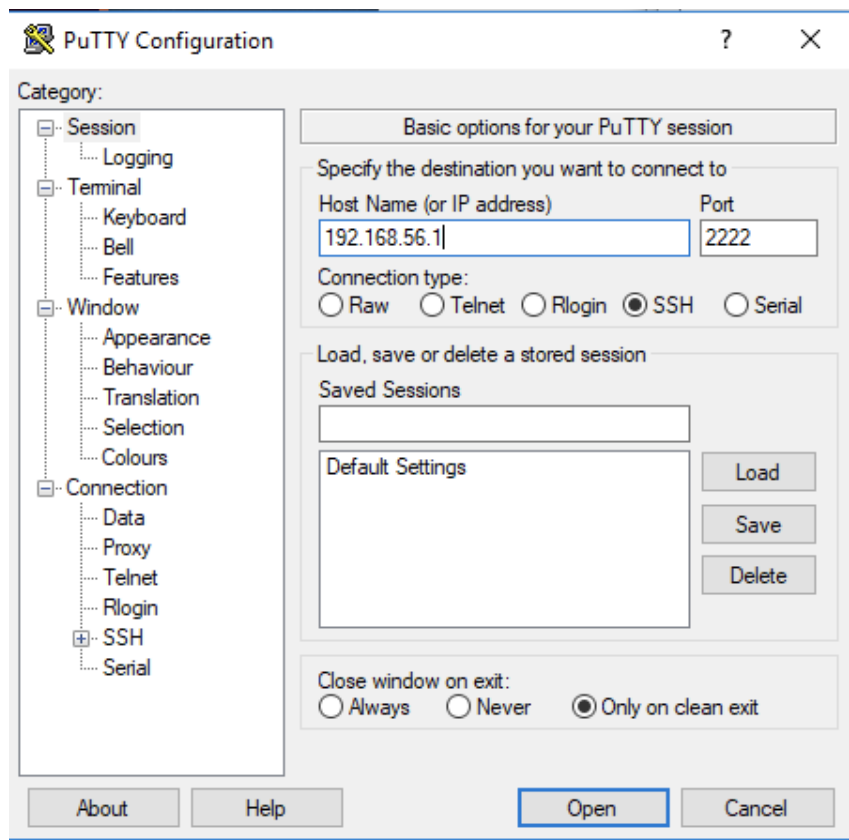
Afin de remédier à ce défaut de connexion, et obtenir une communication bidirectionnelle, nous allons ajouter une règle à la table de routage de la VM. Cette règle permet de rediriger la connexion de l'hôte (port 2222) lorsque qu'il essaye de se connecter à la VM. Lorsque l'hôte tente de se connecter, il y a une redirection automatique sur l'adresse IP de la VM port 22 (SSH).

Dans un premier temps, on définit une règle de redirection de ports.



Nom	Protocole	IP hôte	Port hôte	IP invité	Port invité
Rule 1	TCP	192.168.56.1	2222	10.0.2.15	22

Ensuite, sur l'ordinateur hôte, on utilise PuTTY comme un client SSH.



```
user@tutorial-vm: ~
login as: user
user@192.168.56.1's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-65-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Mon Jan 20 18:47:34 CET 2020

System load:  0.0          Processes:      176
Usage of /:   21.6% of 17.59GB   Users logged in:  1
Memory usage: 60%          IP address for enp0s3: 10.0.2.15
Swap usage:  23%           IP address for docker0: 172.17.0.1

 * Overhead at KubeCon: "microk8s.status just blew my mind".

   https://microk8s.io/docs/commands#microk8s.status

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

144 paquets peuvent être mis à jour.
74 mises à jour de sécurité.

Last login: Fri Oct  4 01:38:33 2019 from 10.0.2.2
user@tutorial-vm:~$
```


Partie 4: Duplication de la VM

La duplication de la VM a fonctionné et le nouveau clone fonctionne correctement.

```
U:\>"C:\Program Files\Oracle\VirtualBox\VBoxManage.exe" clonemedium "U:\Windows\Bureau\disk.vmdk" "U:\Windows\Bureau\disk-copy.vmdk"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Clone medium created in format 'VMDK'. UUID: 271bb6eb-0789-4643-9b5f-e1d50ea2b3d6
U:\>
```

Partie 5: Provisionnement de containers Docker

L'instanciation d'un nouveau Docker ubuntu a fonctionné (CT1). Son adresse IP est: 172.17.0.2

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.0.2 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:ac:11:00:02 txqueuelen 0 (Ethernet)
    RX packets 7243 bytes 18196846 (18.1 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6665 bytes 365156 (365.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

On va maintenant vérifier la connectivité:

-du Docker vers une ressource internet: cela fonctionne

```
root@ea46ca0ca3ba:/# ping www.google.com
PING www.google.com (172.217.171.228) 56(84) bytes of data.
64 bytes from mrs09s07-in-f4.1e100.net (172.217.171.228): icmp_seq=1 ttl=50 time
=8.64 ms
64 bytes from mrs09s07-in-f4.1e100.net (172.217.171.228): icmp_seq=2 ttl=50 time
=7.99 ms
64 bytes from mrs09s07-in-f4.1e100.net (172.217.171.228): icmp_seq=3 ttl=50 time
=8.67 ms
64 bytes from mrs09s07-in-f4.1e100.net (172.217.171.228): icmp_seq=4 ttl=50 time
=8.95 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 7.999/8.568/8.959/0.357 ms
root@ea46ca0ca3ba:/#
```

-du Docker vers la VM: cela ne fonctionne pas

```
root@ea46ca0ca3ba:/# ping 10.0.2.5
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data.
From 10.0.2.15 icmp_seq=1 Destination Host Unreachable
From 10.0.2.15 icmp_seq=2 Destination Host Unreachable
From 10.0.2.15 icmp_seq=3 Destination Host Unreachable
From 10.0.2.15 icmp_seq=4 Destination Host Unreachable
From 10.0.2.15 icmp_seq=5 Destination Host Unreachable
From 10.0.2.15 icmp_seq=6 Destination Host Unreachable
^C
--- 10.0.2.5 ping statistics ---
9 packets transmitted, 0 received, +6 errors, 100% packet loss, time 8198ms
pipe 4
```

-de la VM vers le Docker: cela fonctionne

```
user@tutorial-vm:~$ ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.030 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=64 time=0.284 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=64 time=0.069 ms
^C
--- 172.17.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2049ms
rtt min/avg/max/mdev = 0.030/0.127/0.284/0.112 ms
user@tutorial-vm:~$
```

Au final, on voit qu'il y a une connexion bidirectionnelle entre le Docker et l'extérieur. Cependant, la communication est unidirectionnelle entre le Docker et la VM (le Docker ne peut pas communiquer avec la VM).

L'exécution d'une nouvelle instance (CT2) du Docker ubuntu a fonctionné. Nous y avons installé nano. Ensuite nous avons réalisé un snapshot de CT2 et vérifié son ID.

```
user@tutorial-vm:~$ sudo docker ps
[sudo] Mot de passe de user :
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS             PORTS              NAMES
e65d0bde2284       ubuntu             "/bin/bash"        3 minutes ago
Up 3 minutes       0.0.0.0:2223->22/tcp ct2
ea46ca0ca3ba       ubuntu             "/bin/bash"        34 minutes ago
Up 34 minutes      ct1
```

Nous avons alors arrêté CT2.

```
user@tutorial-vm:~$ sudo docker stop e65d0bde2284
e65d0bde2284
user@tutorial-vm:~$ sudo docker rm e65d0bde2284
e65d0bde2284
user@tutorial-vm:~$
```

Puis nous avons listé les images du Docker disponibles dans la VM.

```
user@tutorial-vm:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
home/user            version1           f38bf70c4f63       6 minutes ago
93.1MB
ubuntu              latest             ccc6e87d482b       4 days ago
64.2MB
user@tutorial-vm:~$
```

Nous avons exécuté une nouvelle instance (CT3) à partir du snapshot précédemment créée à partir de CT2.

```
user@tutorial-vm:~$ sudo docker run --name ct3 -it home/user:version1
root@9cefd70ecd63:/#
```

Nous avons ensuite vérifié si nano était déjà installé sur CT3. Et nano est bien installé sur CT3.

Docker nous permet de partager des “recettes” pour créer des images persistantes (alternative aux snapshots). Nous avons donc réalisé le fichier: *myDocker.dockerfile*

```
GNU nano 2.9.3 myDocker.dockerfile
FROM ubuntu
RUN apt update -y
RUN apt install -y nano
CMD ["/bin/bash"]
```

Et nous avons ensuite “build” l’image dans la VM. Nous obtenons finalement un container CT4 avec les mêmes configurations que CT3.

```
user@tutorial-vm:~$ sudo docker build -t home/user:version2 -f myDocker.dockerfile .
Sending build context to Docker daemon 1.688MB
Step 1/4 : FROM ubuntu
--> ccc6e87d482b
Step 2/4 : RUN apt update -y
--> Running in 1c2677f54068

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic InRelease [242 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:5 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [6779 B]
Get:6 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [780 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/main amd64 Packages [1344 kB]
Get:8 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [804 kB]
Get:9 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [23.7 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [11.3 MB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [186 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic/restricted amd64 Packages [13.5 kB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [37.4 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [1078 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [10.8 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [1338 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic-backports/main amd64 Packages [2496 B]
Get:18 http://archive.ubuntu.com/ubuntu bionic-backports/universe amd64 Packages [4241 B]
Fetched 17.5 MB in 8s (2227 kB/s)
Reading package lists...
Building dependency tree...
Reading state information...
1 package can be upgraded. Run 'apt list --upgradable' to see it.
Removing intermediate container 1c2677f54068
--> e0a127a03dff
Step 3/4 : RUN apt install -y nano
--> Running in 2cef4b911960

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Reading package lists...
Building dependency tree...
Reading state information...
Suggested packages:
  spell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 231 kB of archives.
After this operation, 778 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 nano amd64 2.9.3-2 [231 kB]
debconf: delaying package configuration, since apt-utils is not installed
Fetched 231 kB in 0s (496 kB/s)
Selecting previously unselected package nano.
(Reading database ... 4046 files and directories currently installed.)
Preparing to unpack .../nano_2.9.3-2_amd64.deb ...
Unpacking nano (2.9.3-2) ...
Setting up nano (2.9.3-2) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
Removing intermediate container 2cef4b911960
--> 48dcfbc5637b
Step 4/4 : CMD ["/bin/bash"]
--> Running in c4e2662af362
Removing intermediate container c4e2662af362
--> 9990bf1175ed
Successfully built 9990bf1175ed
Successfully tagged home/user:version2
user@tutorial-vm:~$
```

Conclusion

Au cours de cette session, nous avons acquis des connaissances sur les machines virtuelles les conteneurs, leurs caractéristiques, ce qui les différencient, et dans quels usages ils étaient adaptés. Nous avons aussi acquis une expérience technique pour leur mise en place.

Lab 2: Provisioning End-user Application in Cloud Platforms

Partie Théorique:

Ce TP permet une bonne compréhension de la gestion du cycle de vie des applications basées sur les services dans le contexte du cloud computing. Nous y étudions différentes phases du processus de provisionnement et la façon dont ils devraient être mis en œuvre lors du provisionnement : (i) exemple de code HelloWorld sur les plates-formes cloud existantes et diverses (niveau PaaS) (à savoir Google Cloud Platform, Cloud Foundry et Jelastic) et, (ii) le générateur et gestionnaire de ressources

Voici les différentes phases et étapes qui composent le processus de provisionnement des applications cloud:

- Développement : développement des ressources, compilation, testing...
- Déploiement: allocations des ressources, Téléchargement des exécutables sur ces ressources, Activation de l'application selon les plans/SLA spécifiés.
- Management: Exécuter l'application, Effectuer les opérations de gestion appropriées visant à optimiser les performances et réduire les coûts

Savoir développer une application cloud complexe

Voici les concepts clés pour designer et développer de manière optimale des applications cloud:

- Concevoir l'application comme une collection de (micro) services: les applications cloud sont mieux déployées sous la forme de collection de services cloud ou API.
- Découper les données: les clouds sont des systèmes distribués complexes qui fonctionnent mieux avec des architectures d'application qui répartissent le traitement et les données en composants séparés.
- Tenir compte des communications entre les composants d'application: les composants d'application qui communiquent constamment entre eux réduisent les performances de l'application globale.
- Modéliser, concevoir et coder pour optimiser les performances et la scalabilité
- Rendre la sécurité systémique au sein de l'application

Partie Pratique :

Cloud Foundry

Nous avons provisionné le Servelet HelloWorld sur Cloud Foundry.

```
Last login: Wed Jan 22 22:30:25 on console
MacBook-Pro-de-Claire-2:~ clairegaigeot$ brew install cloudfoundry/tap/cf-cli

Updating Homebrew...
==> Auto-updated Homebrew!
Updated 3 taps (homebrew/core, homebrew/cask and adoptopenjdk/openjdk).
==> New Formulae
apollo-cli      awscli          dsdpn           eureka          findomain       glow            mmctl           wasmer
==> Updated Formulae
ack              dependency-check homebank         minitest         rocksdb
acpica           devspace        httslib         minio            ruby-build
alp             dhall           httpie          minio-mc         sbt
alpine          dhall-json      hub             mk-config        scc
ammonite-repl   dnscontrol      imagemagick     elpack           scummvm
angular-cli     docker-compose  imagemagick@6   mongo-c-driver  serverless
ansible         doctl           imgproxy        mono             shadowssocks-libev
antlr           duplicity        interactive-rebase-tool monolith         shfmt
armadillo       elasticsearch   jenkins         mpd              sile
artifactory     ensmallen       jetty           msitools         sk
ask-cli         erlang          jfrog-cli-go    multimarkdown    skaffold

byobu           gitlab-runner  libxkbcommon    pdsh              tmux-resume
byteman         gitleaks       libxslt++3       pds             topgrade
cake            gmic           logstash         perl tidy         traefik
calicoctl       gmt            lxc              petsc            translate-toolkit
cedille         gmt@5          mafft            petsc-complex    triton
cfn-lint        gnu-getopt     magic-wormhole   php              tundra
cfr-decompiler  gnu-sed        mailutils        phpstan          twtxt
cgreep          golang-migrate make             pipx             typescript
cheat           govc           makedepend       plantuml         uftp
chronograf      gradle         man-db           pnetcdf          vault
cim             grafana        mariadb          pnpm             verilator
closures        grakn          mariadb@10.2    pspp             vnstat
composer        groovy         mariadb@10.3    pulumi           vtk
conan           gssdp          mbedtls          purescript       vulkan-headers
contentful-cli  h3             mda-lv2          pyinvoke         wabt
convex          haproxy        mdcats           python-yq         whistle
cppunit         hdf5           mdcats           q                wireguard-go
cromwell        helmshman      metricbeat       qmp              wireshark
csound          hey            mg               quickjs          xdotool
csvq            hey            micronaut        radare2          xmrig
cypher-shell    hledger        midnight-commander rhino             yaegi
deno            hint           mill             rke              youtube-dl

==> Deleted Formulae
apel            auto-scaling    aws-elasticache aws-sns-cli     lastfmfpclient trr

==> Tapping cloudfoundry/tap
Cloning into '/usr/local/Homebrew/Library/Taps/cloudfoundry/homebrew-tap'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 14 (delta 1), reused 6 (delta 0), pack-reused 0
Unpacking objects: 100% (14/14), done.
Tapped 7 formulae (49 files, 55.3KB).
==> Installing cf-cli from cloudfoundry/tap
Warning: Your Xcode (10.3) is outdated.
Please update to Xcode 11.3 (or delete it).
Xcode can be updated from the App Store.

==> Downloading https://packages.cloudfoundry.org/homebrew/cf-6.49.0.tgz
==> Downloading from https://s3-us-west-1.amazonaws.com/cf-cli-releases/releases/v6.49.0/cf-cli_6.49.0_osx.tgz
##### 100.0%
==> Caveats
Bash completion has been installed to:
  /usr/local/etc/bash_completion.d
==> Summary
📦 /usr/local/Cellar/cf-cli/6.49.0: 6 files, 23.2MB, built in 7 seconds
MacBook-Pro-de-Claire-2:~ clairegaigeot$
MacBook-Pro-de-Claire-2:~ clairegaigeot$ cf api api.run.pivotal.io
Définition du noeud final d'API api.run.pivotal.io...
OK

noeud final d'API : https://api.run.pivotal.io
version d'API : 2.144.0
Not logged in. Use 'cf login' or 'cf login --sso' to log in.
MacBook-Pro-de-Claire-2:~ clairegaigeot$
```

```

Mot de passe: MacBook-Pro-de-Claire-2:~ clairegaigeot$ cf login
Noeud final d'API : https://api.run.pivotal.io

Email: MacBook-Pro-de-Claire-2:~ clairegaigeot$ cf login -a https://api.run.pivotal.io
Noeud final d'API : https://api.run.pivotal.io

Email: gaigeot@etud.insa-toulouse.fr

Mot de passe:
Authentification...
OK

Targeted org gaigeot-org

Targeted space development

```

Pivotal Web Services

Search apps, services, spaces, & orgs

press **?**

gaigeot@etud.insa-toulouse.fr

Home <<

Marketplace

Recently Accessed Apps

No apps recently accessed

Orgs CREATE ORG

Org Name	Quota	Spaces	Domains
gaigeot-org	0% (0 Bytes / 2 GB)	1	0

```

manifest.yml
1  ---
2  applications:
3  - name: hello-java-gaigeot
4    memory: 240M
5    instances: 1
6    host: gaigeot-org
7    path: target/hello-java.war
8

```

```

MacBook-Pro-de-Claire-2:cf-example-hello-java-master clairegaigeot$ cf push
Envoi par commande push du manifeste à l'organisation gaigeot-org/l'espace development en tant que gaigeot@etud.insa-toulouse.fr...
Utilisation du fichier manifeste /Users/clairegaigeot/Desktop/Cloud/cf-example-hello-java-master/manifest.yml

Deprecation warning: Route component attributes 'domain', 'domains', 'host', 'hosts' and 'no-hostname' are deprecated. Found host.
Please see https://docs.cloudfoundry.org/devguide/deploy-apps/manifest-attributes.html#deprecated for the currently supported syntax and other app manifest deprecations. This feature will be removed in the future.

Utilisation du fichier manifeste /Users/clairegaigeot/Desktop/Cloud/cf-example-hello-java-master/manifest.yml
Mise à jour de l'application hello-java-gaigeot dans l'organisation gaigeot-org/l'espace development en tant que gaigeot@etud.insa-toulouse.fr...
OK
Création de la route gaigeot-org.cfapps.io...
OK
Liaison de gaigeot-org.cfapps.io à hello-java-gaigeot...
OK
Téléchargement de hello-java-gaigeot...
Téléchargement des fichiers d'application depuis : /var/folders/r4/wjbtwqydn8sq12wdfc63dtrc9888gn/7/unzipped-app554212562
Téléchargement de 3.3K, 13 fichier(s)
Done uploading
OK

Démarrage de l'application hello-java-gaigeot dans l'organisation gaigeot-org/l'espace development en tant que gaigeot@etud.insa-toulouse.fr...
Downloading web_config_transform_buildpack...
Downloading nodejs_buildpack...

```




```
0 instance(s) en cours d'exécution sur 1, 1 en cours de démarrage
0 instance(s) en cours d'exécution sur 1, 1 en cours de démarrage
0 instance(s) en cours d'exécution sur 1, 1 en panne
ECHEC
Erreur lors du redémarrage de l'application : Echec du démarrage

ASTUCE : utilisez 'cf logs hello-java-gaigeot --recent' pour plus d'informations
MacBook-Pro-de-Claire-2:cf-example-hello-java-master clairegaigeot$
```

```
2020-06-23T13:29:30.94+0100 [APP/PROD/M03/W] DWR Consul calculate DWR memory configuration: there is insufficient memory remaining for heap. Memory available for allocation 512M is less than allocated memory 57942M 1-00:ReservedCodeSize
2020-06-23T13:29:30.94+0100 [APP/PROD/M03/W] OUT Exit status 1
2020-06-23T13:29:30.94+0100 [CELL/006/W] OUT Exit status 0
2020-06-23T13:29:30.94+0100 [CELL/0] OUT Cell 78d73e58-8a79-4317-b061-c56223af688e stopping instance e8bc7bd2-8d86-4a82-c08e-8782
2020-06-23T13:29:30.94+0100 [CELL/0] OUT Cell 78d73e58-8a79-4317-b061-c56223af688e destroying container for instance e8bc7bd2-8d86-4a82-c08e-8782
2020-06-23T13:29:30.94+0100 [APP/PROD/M03/W] OUT Process has crashed with type: "web"
2020-06-23T13:29:30.94+0100 [APP/PROD/M03/W] OUT App Instance exited with guid 8003e0ea-f1aa-43a3-80ad-3582f5c8a88e payload: {"instance":"e8bc7bd2-8d86-4a82-c08e-8782", "index":0, "cell_id":"78d73e58-8a79-4317-b061-c56223af688e", "reason":"OAS402", "exit_description":"APP/PROD/M03: Exited with status 1", "crash_count":3, "crash_timestamp":"2020-06-23T13:29:30.94+0100", "version":"97f11a1-e88e-46c3-a367-e21786ae088e"}
2020-06-23T13:29:30.94+0100 [PROD/W0] OUT Exit status 137
2020-06-23T13:29:30.94+0100 [CELL/0] OUT Cell 78d73e58-8a79-4317-b061-c56223af688e successfully destroyed container for instance e8bc7bd2-8d86-4a82-c08e-8782
MacBook-Pro-de-Claire-2:cf-example-hello-java-master clairegaigeot$
```

Le déploiement a échoué car la mémoire allouée n'était pas suffisante. Nous avons donc augmenté l'allocation de mémoire.

press  gaigeot@etud.insa-toulouse.fr ▾

×

Scale app

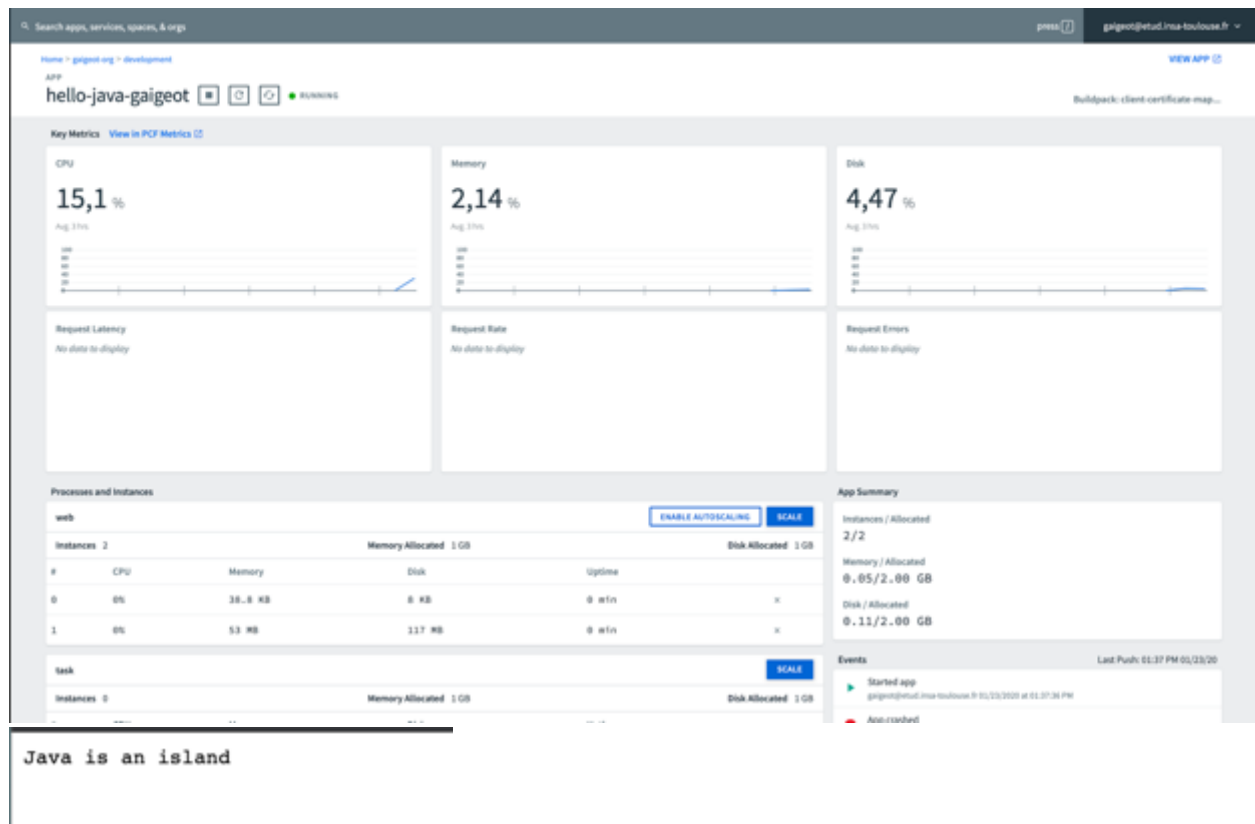
web

Instances	Memory Limit	Disk Limit
<input type="text" value="1"/>	<input type="text" value="512 MB"/>	<input type="text" value="1 GB"/>

Usage Total

Instances	Memory Limit	Disk Limit
1	0.50 GB	1.00 GB

APPLY CHANGES



Le déploiement a fonctionné et nous pouvons donc ouvrir notre application dans un moteur de recherche.

APP **hello-java-gaigeot** ■ □ □ ■ RUNNING

Buildpack: client-certificate-map...

[BIND SERVICE](#) [NEW SERVICE](#)

Service	Plan	Instance Name	Binding Name
ElephantSQL	free - Tiny Turtle	gaigeot_SQL	gaigeot_SQL

Nous avons ensuite lié un service, ici ElephantSQL, qui permettra de stocker des données.

Jelastic

Nous avons provisionné le même Servelet HelloWorld sur Jelastic. Pour ce faire, nous avons utilisé Hidora.

