# TP3 – QEMU

## Configuration de la machine de base

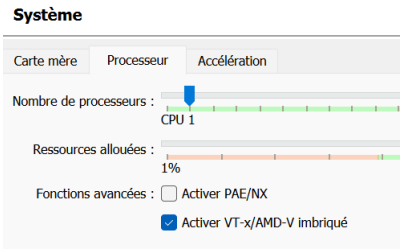
Pour ce TP j'ai utilisé Virtual Box.

## Question 1 : installer les paquets nécessaires à l'exécution d'une machine virtuelle QEMU

⚠ Attention sur **VirtualBox** il faut changer la configuration des machines pour qu'elles puissent supporter la virtualisation. Dans un terminal taper la commande suivante :

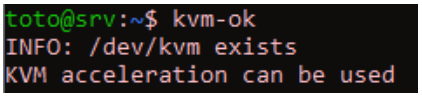
```
C:\'Program Files'\Oracle\VirtualBox\VBoxManage.exe modifyvm Clone_Ubuntu --nested-hw-virt on
```

Mettre le nom de sa machine à la place du paramètre orange. Pour vérifier que la commande est fonctionnée, se rendre dans l'onglet Système de la machine concernée et vérifier que la case « Activer VT-x/AMD-V imbriqué » soit cochée.




Dans un premier temps il faut faire une mise à jour de sa machine ainsi que vérifier si l'option « `kvm` » est disponible. Voici les commandes faites :

```
sudo apt-get update
kvm-ok
```




Ensuite, j'ai pu installer les paquets nécessaires à l'utilisation de QEMU. Certains paquets téléchargés ne seront pas utiles puisqu'ils servent uniquement en mode graphique. Nous ne pouvons pas utiliser ces paquets graphiques puisque nous sommes dans une machine Ubuntu Server.

```
sudo apt install qemu-kvm bridge-utils
```

## Question 2 : installer une machine Alpine dans QEMU. Réaliser une installation système. Utiliser un disque virtuel de 2Go.

Dans un premier temps j'ai téléchargé l'iso d'une machine Alpine.

⚠ Attention ne pas utiliser l'iso « `alpine-standard-3.17.0-x86_64.iso` » qui était la dernière version disponible au moment de la rédaction de ce TP. En effet, lors du boot un bug rend la fin de l'installation impossible.

J'ai donc téléchargé une iso antérieure avec la ligne de commande :

```
wget https://dl-cdn.alpinelinux.org/alpine/v3.10/releases/x86_64/alpine-extended-3.10.0-x86_64.iso
```

Puis, j'ai créé un modèle compressé de cette image dans un format nommé `qcow2` :

```
qemu-img create -f qcow2 alpine-extended-3.10.0-x86_64.qcow2 2G
```

## Question 3 : Démarrer la machine virtuelle selon la configuration suivante :

| |
|---|
| **128** ou 256 Mo de mémoire |
| Clavier français |
| Mode réseau user |

Afin de démarrer la machine virtuelle j'ai créé un script avec les commandes suivantes :

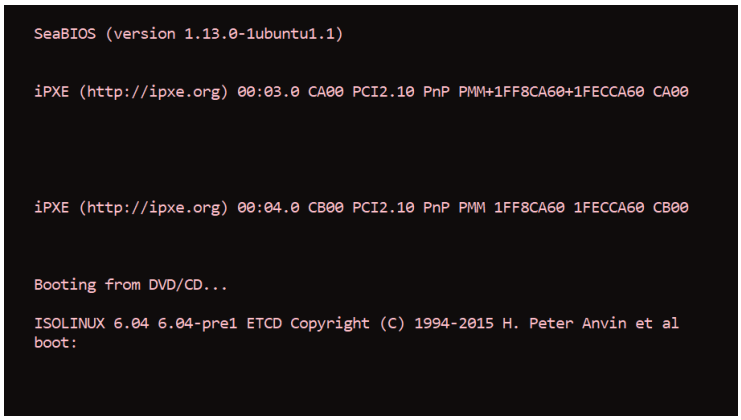
```
nano script.sh
chmod a+x script.sh
qemu-system-x86_64 \
  -k fr \
  -vga virtio \
  -m 128 \
  -nic user \
  -boot d \
  -cdrom alpine-extended-3.10.0-x86_64.iso \
  -drive file=alpine-extended-3.10.0-x86_64.qcow2,if=virtio \
  -net nic \
  -net user,hostfwd=tcp::2223-:22 \
  -curses
^X
```

Explication des différents champs (sur ce lien se trouve l'explication de tous les champs possibles) :

| *Champ :* | *Explication :* |
|---|---|
| `-k fr` | Permet de mettre le clavier en français. Sinon, même pendant le boot avec un choix de clavier français le clavier sera Qwerty. |
| `-vga virtio` | Sans cette option il ne sera pas possible d'afficher en mode terminal la console de la machine. |
| `-m 512` | Choisir la taille de la RAM |
| `-net user,hostfwd=tcp::2223-:22` | Permet le forward de port |
| `-cdrom` | Permet de booter la machine |
| `-curses` | Obligatoire pour le mode non graphique. En effet le mode `-nographic` ne renverra rien à l'écran. Aussi, si vous ne mettez aucune option il est impossible d'exécuter le script puisqu'il n'est pas possible de lancer QEMU en mode graphique, et l'option de base est `gtk`. |

Il a ensuite fallu l'exécuter avec la commande : `sudo ./script.sh`

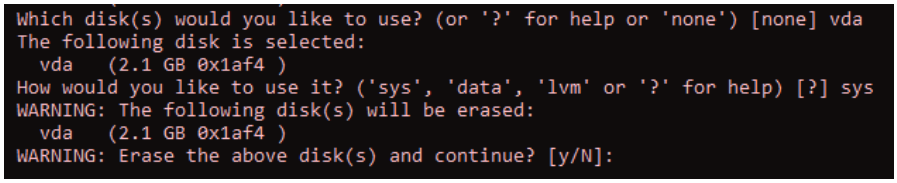
Apparaît alors à l'écran le lancement de l'installation de la machine Alpine :




La machine boot et nous demande de rentrer de nom d'un utilisateur qui est `root`. La première commande a rentré par la suite est « `setup-alpine` ».

Concernant les choix de configurations :

- Choisir `fr` puis `fr` pour le clavier
- Choisir `vda` comme choix de disque, puis `sys` à la question de comment l'utiliser et enfin répondre `yes` pour supprimer les disques.




Pour le reste des configurations il est possible de laisser le choix par défaut.

Une fois l'installation terminée éteindre la machine avec « `poweroff` ».

Pour relancer la machine j'ai créé un nouveau script. Celui-là ne comprend pas l'option « » puisque la machine est déjà installée. Ce script permet donc de se connecter/ d'avoir accès au terminal de la machine.

```
nano alpine.sh
qemu-system-x86_64 \
  -k fr \
  -vga virtio \
  -m 128 \
  -nic user \
  -boot d \
  -drive file=alpine-extended-3.10.0-x86_64.qcow2,if=virtio \
  -net nic \
  -net user,hostfwd=tcp::2223-:22 \
  -curses
^X
chmod a+x alpine.sh
```

## Tester la connexion réseau

Pour tester la connexion j'ai utilisé la commande « `ping 8.8.8.8` ». J'ai remarqué que la machine avait donc accès à internet.

```
localhost:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=255 time=18.946 ms
64 bytes from 8.8.8.8: seq=1 ttl=255 time=5.628 ms
64 bytes from 8.8.8.8: seq=2 ttl=255 time=5.413 ms
64 bytes from 8.8.8.8: seq=3 ttl=255 time=5.674 ms
64 bytes from 8.8.8.8: seq=4 ttl=255 time=5.686 ms
64 bytes from 8.8.8.8: seq=5 ttl=255 time=5.035 ms
64 bytes from 8.8.8.8: seq=6 ttl=255 time=5.283 ms
64 bytes from 8.8.8.8: seq=7 ttl=255 time=5.799 ms
^C
--- 8.8.8.8 ping statistics ---
8 packets transmitted, 8 packets received, 0% packet loss
round-trip min/avg/max = 5.035/7.183/18.946 ms
localhost:~#
```

### Donner l'adresse IP de la machine

L'adresse IP de la machine est 10.0.2.15 comme j'ai pu le remarquer avec la commande « `ip a show` »

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP ql
en 1000
    link/ether 52:54:00:12:34:56 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fec0::5054:ff:fe12:3456/64 scope site dynamic
       valid_lft 86139sec preferred_lft 14139sec
    inet6 fe80::5054:ff:fe12:3456/64 scope link
       valid_lft forever preferred_lft forever
```

## Question 4 : Installer dans la machine…

### Un serveur SSH

J'ai utilisé les commandes suivantes pour installer le serveur SSH dans la machine Alpine :

```
apk add openssh
rc-update add sshd
/etc/init.d/sshd start
```

J'ai également décommenté les lignes (qui se trouvent dans la section Port 22) :

- `AddressFamily any`
- `ListenAddress 0.0.0.0`
- `ListenAddress ::`

Qui se trouvent dans le fichier de configuration « `/etc/ssh/sshd_config` ». Même si ce n'est pas recommandé normalement, j'ai donné l'autorisation de se connecter en tant que super utilisateur :

```
PermitRootLogin yes
```

Pour que les modifications soient prises en compte j'ai terminé avec les commandes :

```
service sshd restart
poweroff
```

### Un serveur Apache

Voici les commandes :

```
apk add apache2
rc-service apache2 start
```

## Proposer une commande qui permet de relayer :

### *Un port de l'hôte vers le port 22 de l'invité*

Cette configuration a déjà été faite dans les scripts, cf. question 3. Avec les modifications faites au fichier de configuration en question 4 il est bien possible de se connecter depuis l'hôte sur l'invité avec la commande « `ssh root@127.0.0.1 -p 2223` ».

```
root@127.0.0.1's password:
Welcome to Alpine!

The Alpine Wiki contains a large amount of how-to guides and general
information about administrating Alpine systems.
See <http://wiki.alpinelinux.org/>.

You can setup the system with the command: setup-alpine

You may change this message by editing /etc/motd.

localhost:~#
```

### *Un port de l'hôte vers le port 80 de l'invité*

Pour cela il suffit de modifier le script en rajoutant les valeurs en gras à la ligne de commande déjà présente :

```
net user,hostfwd=tcp::2223-:22,hostfwd=tcp::2224-:80 \
```

En relançant la machine avec ce script, je remarque que l'hôte écoute sur le port 2224 et que l'invité écoute sur le port 80. Ceci a donc fonctionné.

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:2223            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:2224            0.0.0.0:*               LISTEN
```

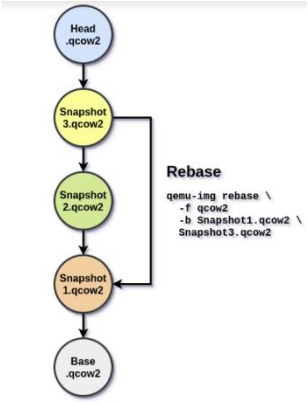
*Capture de l'hôte, commande netstat -a*

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:http            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
```

*Capture de l'invité, commande netstat -a*

## Question 5 : Réaliser deux clones liés (rebase) de la machine virtuelle

Pour information :




- `qemu-img rebase` : permet de modifier le fichier de sauvegarde actuel vers un autre fichier de sauvegarde
- `qemu-img create` : permet de créer une nouvelle image disque sur le système de fichiers

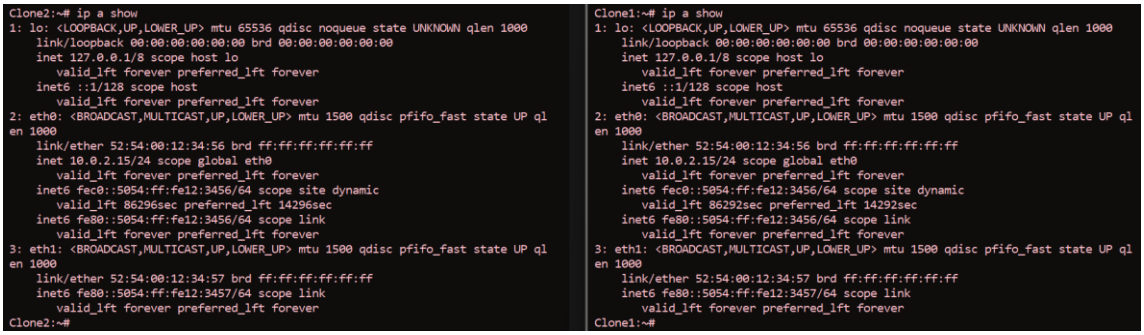
Ici, « `qemu-img rebase` » n'est donc pas nécessaire pour créer des clones. Il faut créer un snap shot de la machine Alpine, puis créer avec ce snap shot des nouvelles machines. Les commandes sont donc les suivantes :

| *Clone 1* | *Clone 2* |
|---|---|
| `sudo qemu-img create -f qcow2 -o backing_file=alpine-extended-3.10.0-x86_64.qcow2 clone1.qcow2 -F qcow2`<br><br>`nano clone1.sh` | `sudo qemu-img create -f qcow2 -o backing_file=alpine-extended-3.10.0-x86_64.qcow2 clone2.qcow2 -F qcow2`<br><br>`nano clone2.sh` |

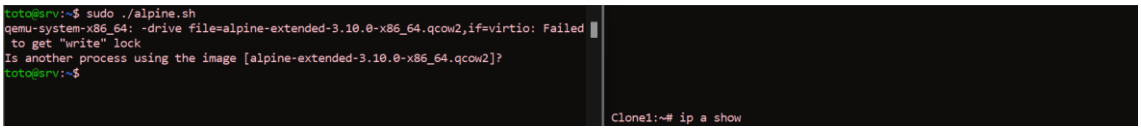
| | |
|---|---|
| qemu-system-x86_64 \<br>-k fr \<br>-vga virtio \<br>-m 512 \<br>-nic user \<br>-boot d \<br>-drive file=clone1.qcow2,if=virtio \<br>-net nic \<br>-net tap \<br>-display curses<br><br>chmod a+x clone1.sh | qemu-system-x86_64 \<br>-k fr \<br>-vga virtio \<br>-m 512 \<br>-nic user \<br>-boot d \<br>-drive file=clone2.qcow2,if=virtio \<br>-net nic \<br>-net tap \<br>-display curses<br><br>chmod a+x clone2.sh |

### *Quelles machines peuvent être lancées en même temps ?*

J'ai remarqué qu'on ne pouvait pas lancer l'Alpine Master en même temps que ses clones. Cependant, il est possible de lancer ses deux clones en même temps.




*Capture d'écran des deux clones lancés en même temps*




*Quand un clone est démarré le master ne peut pas être démarré*

## Question 6 : lancer deux machines QEMU en mode SLIRP par défaut.

Cette configuration par défaut permet à vos machines virtuelles d'accéder facilement à Internet, à condition que l'hôte y soit connecté, mais les machines virtuelles ne seront pas directement visibles sur le réseau externe, et les machines virtuelles ne pourront pas communiquer entre elles si vous démarrez plusieurs simultanément. (cf. ce lien partie 6.2)

### *Récupérer leur adresses IP*

Leur adresse IP sont les mêmes comme nous pouvons le remarquer sur une des captures d'écran précédentes : 10.0.2.15. Elles ne pourront donc pas communiquer ainsi.

## Question 6b : relancer deux machines QEMU en les plaçant sur le même réseau

Voir le TP4 pour cette question