Virtualisation des systèmes

Florent Glück

March 31, 2025

QEMU avancé

Introduction

Le but de ce travail pratique est d'approfondir votre compréhension et votre pratique de l'hyperviseur QEMU, notamment en manipulant conversions et inspection d'images disques, snapshots et enfin en pratiquant la virtualisation de bureau (VDI).

(1) Conversion d'images disque

Le site https://www.osboxes.org est un site où l'on peut trouver des images disques d'OS déjà toutes préparées pour les hyperviseurs "grand public". Contrairement à une image ISO, les OS dans ces images disque sont déjà installés, donc directement utilisables sans installation préalable requise.

- Inspectez les types d'images disponibles sur le site. Quels sont les formats d'images disque téléchargeables et leurs extensions respectives ? Pour chaque type d'image, indiquez pour quel hyperviseur celle-ci est destinée.

Téléchargez l'image de la distribution Linux Debian 11 Bullseye **server** version 64-bits dans chaque format d'image disponible. La raison pour laquelle on désire la version serveur est que l'image (compressée) est de petite taille, moins de 400MB, alors que la version desktop fait > 1GB. De plus, il est toujours utile de se faire la main dans un terminal ! Malheureusement, il n'y a pas d'image Debian serveur en version 12, d'où le choix de la version 11.

- Exécutez ensuite une VM QEMU sur chaque format d'image disque téléchargé. Est-ce que QEMU gère correctement chaque image comme s'il s'agissait d'une image "native" au format qcow2 ? Déterminez pour chaque type d'image disque, si celle-ci est supportée par QEMU.

Pour information, vous pouvez éteindre un système Debian avec la commande `sudo shutdown -h now` et rebooter avec `sudo reboot`.

Procédez par éteindre proprement vos VM, donc sans fermer les fenêtres QEMU, mais en exécutant un *shutdown* depuis l'OS guest. Ensuite, convertissez une des images disque ci-dessus de votre choix en une image au format qcow2 native à QEMU, en spécifiant une taille de cluster de 2MB au lieu de la taille par défaut de 64KB, afin de maximiser les performances disque (entrées/sorties - I/O).

Exécutez ensuite une VM sur l'image qcow2 que vous venez de créer et loggez-vous afin de vérifier que le système est fonctionnel. Vous pouvez changer le mot de passe de l'utilisateur courant avec la commande `passwd`. Le système est configuré avec un clavier américain QWERTY. Si celui-ci ne vous convient pas, vous pouvez le changer avec :

```
sudo dpkg-reconfigure keyboard-configuration
sudo dpkg-reconfigure console-setup
```

(2) Snapshots

Vous allez ici explorer les possibilités offertes par QEMU en matière de snapshots via l'utilisation de `qemu-img`. Lisez la syntaxe des snapshots avec `man qemu-img` ou `qemu-img --help`.

Partie 1

En premier lieu, vous manipulerez les snapshots externes. Commencez par faire une copie de l'image du disque de votre VM Debian et nommez cette copie `disk-base.qcow`. Cette image sera l'image de base (*backed*) sur laquelle se baseront les snapshots (overlays) qui vont suivre.

Démarrez une VM utilisant le disque `disk-base.qcow`.

- Démarrez ensuite une deuxième VM en lui spécifiant la même image disque. Que remarquez-vous ? Quelle est la raison de ce comportement à votre avis ?

Éteignez votre VM, puis similairement au point précédent, démarrez 2 VMs utilisant le même disque `disk-base.qcow`, mais en spécifiant l'argument `-snapshot` à QEMU.

- Est-ce que QEMU se plaint toujours ? Que se passe-t-il exactement maintenant et pourquoi est-ce que cela fonctionne ?

Éteignez vos VMs, puis redémarrez une nouvelle VM sur `disk-base.qcow` (sans l'argument `-snapshot`). Dans celle-ci, créez le script `createfile` qui prend en argument un nom de fichier et y créé un fichier de 100MB. Pour rappel, `dd` permet de créer un fichier à partir d'une source : `dd if=source_file of=dest_file bs=block_size count=block_count`. Dans votre script, utilisez `dd` avec la source `/dev/urandom` pour remplir les 100MB du fichier.

Soit la séquence de snapshots illustrée en Figure 1 où l'ordre des opérations réalisées est indiqué par les disques numérotés. Le script `createfile` précédent a été utilisé pour ajouter les fichiers `fileA, fileB, fileC`.

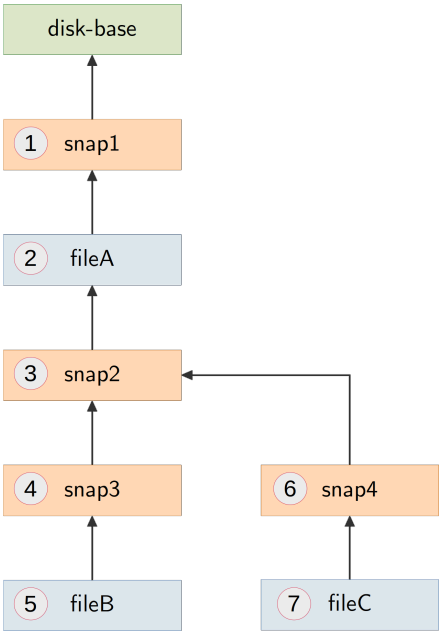

Figure 1: L'image de base (*backed*) est en vert, les snapshots (*overlays*) en orange et les fichiers ajoutés en bleu. L'ordre des opérations réalisées est indiqué par les disques numérotés.

On part du principe que les 6 étapes illustrées en Figure 1 ont été réalisées.

Reproduisez le scénario de la Figure 1 afin de confirmer votre réponse.

- Vous exécutez une VM sur l'image `snap3`. Quel(s) fichier(s) `fileX` seront/sera présent(s) dans cette VM ?
- Vous exécutez ensuite une VM sur l'image `snap4`. Quel(s) fichier(s) `fileX` seront/sera présent(s) dans cette VM ?

Utilisez `qemu-img` pour afficher la chaîne d'images ayant mené à `snap3` et vérifiez qu'il s'agit de la même séquence qu'en Figure 1. Réalisez pareil pour `snap4`.

Exécutez une première VM avec le disque `snap3.qcow`. Exécutez ensuite une deuxième VM avec le disque `snap1.qcow`.

- Que va-t-il se passer à votre avis ? Vérifiez votre réponse empiriquement et déduisez-en ce que réalise QEMU.

Réalisez un **merge** de type "commit" de l'image `snap4` dans l'image `snap1`.

- Quels fichiers `fileX` seront présents dans `snap1` ? Confirmez votre hypothèse empiriquement.

Comme indiqué dans la documentation de QEMU, il est alors fortement conseillé de supprimer les images intermédaires qui pourraient être incohérentes.

- Quelle(s) image(s) pourrait/pourraient être incohérente(s) ?

Au début de la partie 1, vous aviez réalisé une copie de votre image Debian dans `disk-base.qcow` afin de ne pas modifier la première image.

- Qu'auriez-vous pu faire pour éviter de réaliser cette copie de plusieurs GB tout en évitant de modifier le fichier d'origine ?

Partie 2

Vous allez maintenant utiliser un jeu graphique. Reprenez alors l'image disque `lubuntu.qcow` du labo précédent, puis exécutez une VM dessus. Dans l'OS guest, installez le jeu `chromium-bsu`, puis lancez une partie.

En cours de jeu, réalisez un snapshot de VM (cf. slides de cours).

- Pourquoi est-ce que la création de ce snapshot prend significativement plus de temps qu'un snapshot de disque ?

Jouez un petit peu, puis réalisez un nouveau snapshot de VM. Listez ensuite les snapshots depuis le *monitor*.

- Voyez-vous les snapshots réalisés ?

Chargez chaque snapshot de VM afin de vérifier que vous obtenez bien le comportement attendu.

Éteignez votre VM.

- Comment pouvez-vous listez les snapshots de VM que vous venez d'effectuer sans utiliser le QEMU *monitor* ?

Exécutez à nouveau votre VM, mais en indiquant à QEMU de charger le dernier snapshot réalisé depuis la ligne de commande avec l'argument `-loadvm`.

Éteignez votre VM, puis exécutez en une nouvelle avec la même image disque, mais avec des périphériques différents (p.ex. 1 seul CPU au lieu de 2, carte graphique différente, quantité de RAM différente, etc.).

- Chargez ensuite un des snapshots effectués précédemment. Que remarquez-vous ?

(3) Virtualisation de desktop (VDI)

Jusqu'à maintenant, le VMM et l'utilisateur de la VM utilisent et interagissent avec la même machine physique. On s'intéresse ici à **découpler** l'exécution de la VM des périphériques nécessaires à son interaction, c'est à dire l'affichage du bureau, la clavier, la souris et l'audio. Ainsi, la VM s'exécutera sur un serveur distant et votre PC (local) sera uniquement utilisé comme système d'interaction (affichage, clavier, etc.). La communication entre VMM ↔ VM et périphériques d'interaction (affichage, etc.) utilisera le protocole Spice au-dessus du protocole réseau tcp/ip.

Machine distante

A partir de maintenant, vous réaliserez la suite du labo sur un VMM se trouvant sur une machine distante (en réalité une VM) localisée au sein de l'infrastructure ISC. Dans le fichier `VMs_etudiants.ods` disponible sur le git du cours, vous pouvez trouver l'ip de la VM distante qui vous est attribuée ainsi que les crédentials pour vous y connecter. Encore une fois, **cette VM distante aura le rôle de VMM !**

Attention : votre utilisateur possède un **accès root** via `sudo`. Vous pouvez donc complètement **briquer** votre système et le rendre **non bootable** si vous faites n'importe quoi !

En premier lieu, sécurisez votre système en changeant le mot de passe de votre utilisateur (`passwd`) afin que personne d'autre ne puisse accéder à votre machine distante.

Ensuite, mettez en place une paire de clés ssh pour vous connecter à votre machine distante sans avoir à entrer de mot de passe. La clé privée doit résider sur la machine source et la clé publique sur la machine distante. Un moyen simple pour mettre en place vos clés est d'utiliser la commande `ssh-keygen` qui génère une paire de clés ssh. Une clé RSA de 2048 bits fait très bien l'affaire. La commande `ssh-copy-id` copie une clé publique sur une machine de destination (celle-ci est copiée dans le répertoire `.ssh/` de l'utilisateur spécifié). Pensez à copier votre clé publique pour les utilisateurs `student` et `root`.

Enfin, il est possible de configurer, via le fichier `~/.ssh/config`, les crédentials que le client ssh utilise lors de l'établissement d'une connexion. Voici un exemple de configuration où `id_rsa_virt` dénote la clé privée (la clé publique porte l'extension `.pub`) :

```
Host some_host_name 192.168.100.100
        User student
        Port 22
        IdentityFile ~/.ssh/id_rsa_virt
```

Aussi, ssh permet de se connecter à une machine distante en passant par d'autres machines intermédiaires. Cela s'appelle un saut ssh (*jump* ou *hop*). Voici un exemple qui permet de se connecter sur la machine distante 192.168.100.10 (nommée "hyperion") en faisant un saut sur une machine intermédiaire 192.168.100.20 (nommée "endimyon") :

```
Host endimyon
    User dave
    HostName 192.168.100.20

Host hyperion
    User student
    HostName 192.168.100.10
    ProxyJump endimyon
```

Ainsi, en tapant simplement `ssh hyperion`, on peut se connecter à la machine distante.

Partie 1

Vous allez maintenant vous familiariser avec la virtualisation de bureau (VDI) grâce à votre machine distante qui aura le rôle de VMM et au protocole Spice.

Afin de rendre cette partie un peu plus ludique, téléchargez l'image disque compressée `doom.qcow.gz` se trouvant sur https://drive.switch.ch/index.php/s/rHScyU6IVNyoZYD. Une fois décompressée, cette image aura une taille d'environ 15 GB ce qui est trop volumineux pour tenir sur le disque de votre machine distante !

Heureusement, celle-ci contient deux disques supplémentaires de 20GB chacun. Vous pouvez les lister avec la commande `lsblk`. Vous allez placer `doom.qcow` sur le deuxième disque, à savoir `sdb`. Pour cela, créez un système de fichiers de type `ext4` pour y stocker `doom.qcow`. Pour rappel, pour accéder à un système de fichiers, vous devez au préalable le monter avec la commande `mount`. Aussi, étant donné que l'espace sur le deuxième disque n'est pas suffisant pour y stocker l'image disque compressée et l'image décompressée, vous pouvez vous servir du premier disque comme espace de stockage temporaire pour l'image compressée, puis la décompresser sur le deuxième disque.

Copiez le script `vm_run` du travail pratique précédent en `vm_run_spice`, puis modifiez ce dernier afin d'ajouter un serveur Spice à QEMU. On désire également que cette VM présente une carte audio afin de pouvoir profiter du son dans le jeu Doom. Voici les arguments à passer à QEMU pour ajouter une carte son (émulée) Intel 82801AA AC97 :

```
-device ac97
```

Sur votre machine distante, démarrez alors une VM sur l'image disque `doom.qcow`. A noter que le mot de passe pour se connecter est... "doom".

Partie 2

Sur votre machine locale, installez un client spice comme `remote-viewer` ou `spicy`, puis connectez-vous à votre VM `doom` distante.

Dans l'OS guest de la VM, exécutez `doom` afin d'observer les performance de l'affichage distant. Dans le cas où votre machine locale se trouverait à l'extérieur du réseau HES, vous pouvez y accéder via le VPN de l'école.

- Si vous vous trouvez sur le réseau interne de l'école (depuis le wifi HES-GE ou mieux, en câblé depuis les salles de cours au 4ème étage), le jeu Doom est-il jouable ?
- Même question que ci-dessus mais depuis chez vous, via le VPN ?
- Sur quelle machine physique s'exécute le jeu exactement ?

- Vous devriez entendre la bande sonore du jeu. Le serveur distant ne possède pourtant physiquement aucune carte audio. Comment est-il alors possible que vous entendiez le son du jeu ?
- Si vous fermez la fenêtre du client Spice durant l'utilisation de la VM, quelle est la conséquence sur l'exécution de la VM ?
- Vous est-il possible de copier/coller de texte entre la VM et votre machine hôte ? Comment est-ce possible ?