

Lab 1 : Installation de KVM sur UBUNTU et création d'une VM

Objectif du Lab :

Dans ce laboratoire, nous allons installer KVM (Kernel-based Virtual Machine) avec tous ses composants nécessaires pour permettre la virtualisation complète. Ensuite, nous procéderons à la création d'une machine virtuelle (VM) afin de démontrer le fonctionnement de KVM et ses capacités de gestion des ressources. Ce guide couvre les étapes d'installation, la configuration de l'environnement, et les commandes pour créer et gérer une VM.

Prérequis :

Avant de commencer ce laboratoire, assurez-vous que vous avez les éléments suivants :

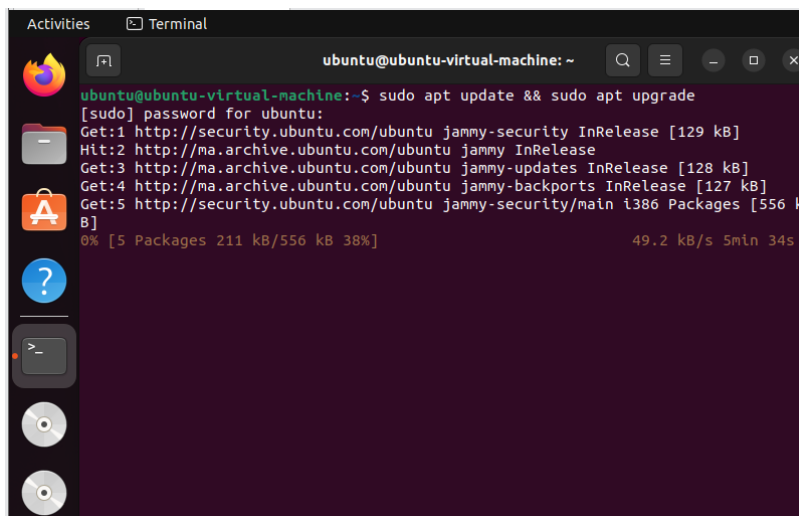
- Une machine Ubuntu (Ubuntu 20.04 ou version plus récente)
- Accès root ou sudo pour installer des paquets.

I. Installation de KVM sur Ubuntu

1. Mettez à jour les paquets de votre système Ubuntu :

```
$ sudo apt update && sudo apt upgrade -y
```

2. Vérifier la compatibilité de votre CPU avec KVM :



```
ubuntu@ubuntu-virtual-machine: ~  
ubuntu@ubuntu-virtual-machine:~$ sudo apt update && sudo apt upgrade  
[sudo] password for ubuntu:  
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]  
Hit:2 http://ma.archive.ubuntu.com/ubuntu jammy InRelease  
Get:3 http://ma.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]  
Get:4 http://ma.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]  
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [556 k  
B]  
0% [5 Packages 211 kB/556 kB 38%] 49.2 kB/s 5min 34s
```

Dans cette étape, il est important de vérifier si votre CPU prend en charge KVM (Kernel-based Virtual Machine).

Pour cela, utilisez la commande suivante :

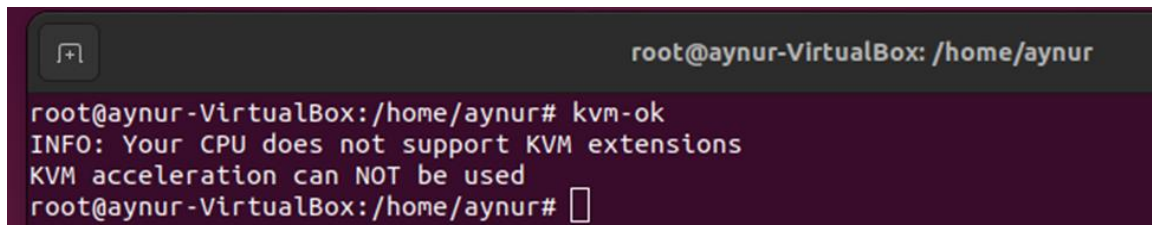
```
$egrep -c '(vmx|svm)' /proc/cpuinfo
```

Si le résultat est 0, cela signifie que votre CPU ne prend pas en charge la virtualisation matérielle (cette technologie permet de séparer les ressources matérielles physiques, comme le processeur et le stockage, du reste du système). Si le résultat est 1 ou plus, cela signifie que votre CPU est compatible.

Une autre méthode pour vérifier la compatibilité est d'utiliser les deux commandes suivantes :

```
$ sudo apt install cpu-checker
```

```
$ kvm-ok
```

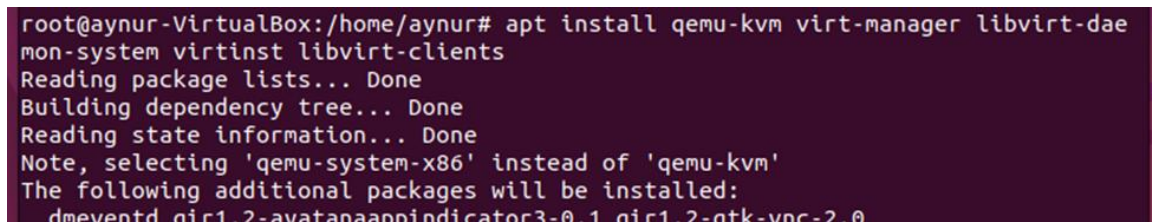


```
root@aynur-VirtualBox: /home/aynur# kvm-ok
INFO: Your CPU does not support KVM extensions
KVM acceleration can NOT be used
root@aynur-VirtualBox: /home/aynur#
```

Note: Si vous voyez cela, vous pouvez toujours exécuter des machines virtuelles, mais elles seront beaucoup plus lentes sans les extensions KVM.

3. Installez KVM par la commande :

```
$apt install qemu-kvm virt-manager libvirt-daemon-system virtinst libvirt-clients
```



```
root@aynur-VirtualBox: /home/aynur# apt install qemu-kvm virt-manager libvirt-dae
mon-system virtinst libvirt-clients
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'qemu-system-x86' instead of 'qemu-kvm'
The following additional packages will be installed:
  dmideventd gir1.2-ayatanaappindicator3-0.1 gir1.2-gtk-vnc-2.0
```

4. Activez le service de libvirt :

```
$systemctl enable --now libvirtd
```

```
$Systemctl start libvirtd
```

ces commandes sont essentielles pour gérer le service libvirtd, qui assure la gestion des machines virtuelles et permet d'utiliser les fonctionnalités de KVM.

```
root@aynur-VirtualBox:/home/aynur# systemctl status libvirtd
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-11-08 16:10:31 +04; 6min ago
     TriggeredBy: ● libvirtd-admin.socket
                  ● libvirtd-ro.socket
                  ● libvirtd.socket
        Docs: man:libvirtd(8)
              https://libvirt.org
    Main PID: 5633 (libvirtd)
      Tasks: 21 (limit: 32768)
     Memory: 9.7M
        CPU: 561ms
    CGroup: /system.slice/libvirtd.service
            └─5633 /usr/sbin/libvirtd
              └─5766 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default
                └─5767 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default

Noy 08 16:10:31 aynur-VirtualBox dnsmasq-dhcp[5766]: DHCP, IP range 192.168.122.0/24
Noy 08 16:10:31 aynur-VirtualBox dnsmasq-dhcp[5766]: DHCP, sockets bound exclusively
Noy 08 16:10:31 aynur-VirtualBox dnsmasq[5766]: reading /etc/resolv.conf
Noy 08 16:10:31 aynur-VirtualBox dnsmasq[5766]: using nameserver 127.0.0.53#53
```

II. Création d'une machine virtuelle sur KVM:

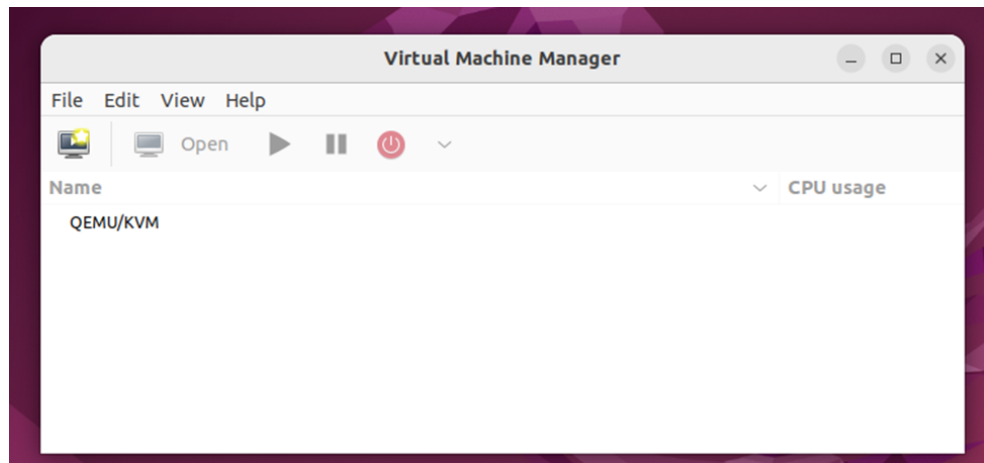
Pour créer une machine virtuelle, on doit ouvrir d'abord Virtual Machine Manager, et on peut faire ceci par 2 méthodes différents :

1. Des commandes shell :

Tapez la commande :

```
$virt-manager
```

L'interface de VMM va être ouverte comme ceci :

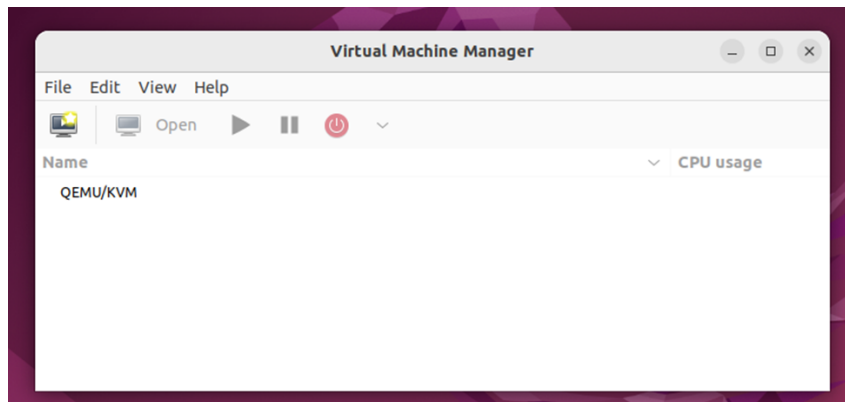


2. Méthode manuelle :

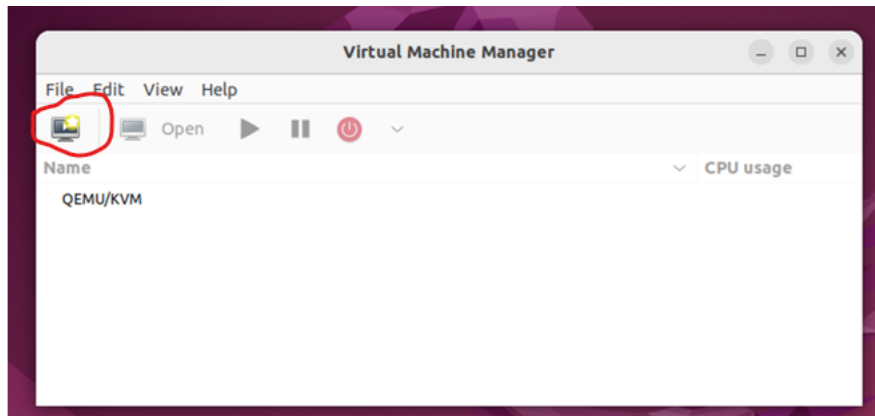
Dans la barre de recherche, tapez Virtual Machine Manager :



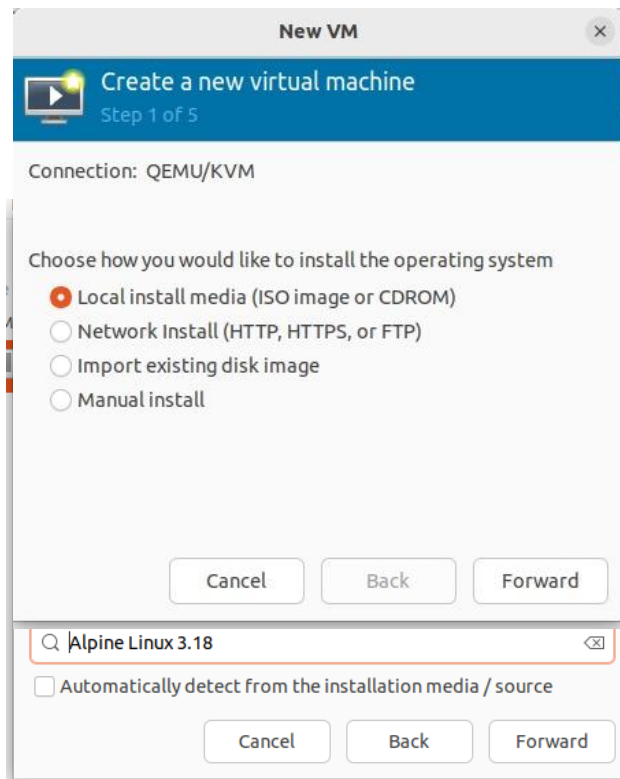
D'où l'ouverture de Virtual Machine manager :



Maintenant on clique ici pour sélectionner notre VM :



Et on sélectionne ISO Image :



On parcourt vers l'emplacement notre VM, on a choisi pour tester Alpine Linux (Alpine Linux est une distribution Linux légère et sécurisée, conçue principalement pour être utilisée dans des environnements nécessitant peu de ressources, comme les conteneurs Docker) :

Notre VM est en cours d'exécution :

```
OpenRC 0.54 is starting up Linux 6.6.49-0-lts (x86_64)

* /proc is already mounted
* Mounting /run ... [ ok ]
* /run/openrc: creating directory
* /run/lock: creating directory
* /run/lock: correcting owner
* Caching service dependencies ... [ ok ]
* Remounting devtmpfs on /dev ... [ ok ]
* Mounting /dev/mqueue ... [ ok ]
* Mounting modloop ... [ ok ]
* Verifying modloop [ ok ]
* Mounting security filesystem ... [ ok ]
* Mounting debug filesystem ... [ ok ]
* Mounting persistent storage (pstore) filesystem ... [ ok ]
* Starting busybox mdev ... [ ok ]
* Scanning hardware for mdev ... [ ok ]
* Loading hardware drivers ...
* Loading modules ...
* Setting system clock using the hardware clock [UTC] ...
* Checking local filesystems ...
* Remounting filesystems ...
* Mounting local filesystems ...
* Configuring kernel parameters ...
* Migrating /var/lock to /run/lock ...
* Creating user login records ...
* Cleaning /tmp directory ...
* Setting hostname ...
* Starting busybox syslog ...
* Starting firstboot ...

Welcome to Alpine Linux 3.20
Kernel 6.6.49-0-lts on an x86_64 (/dev/tty1)

localhost login:
```