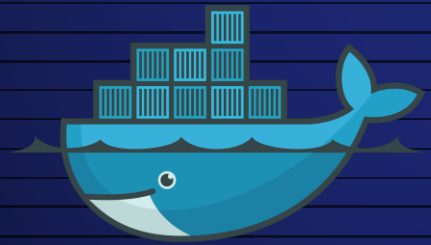
Rapport d’installation de Docker et Kubernetes sous Debian12



docker

kubernetes

Présenté par

1. Mariama DIACK
2. Néné MBAYE
3. Mouhamed SECK

Sous la direction de:
M.LO Massamba

Année académique: 2023-2024

INTRODUCTION

Docker et Kubernetes sont deux technologies essentielles pour le déploiement et la gestion d'applications conteneurisées.

Docker est une plateforme de conteneurisation qui permet de regrouper une application et ses dépendances dans un package léger et portable appelé conteneur. Les conteneurs Docker sont isolés les uns des autres, ce qui signifie qu'ils n'interfèrent pas les uns avec les autres et qu'ils peuvent s'exécuter de manière fiable sur différents environnements. Kubernetes est une plateforme d'orchestration de conteneurs qui permet d'automatiser le déploiement, la mise à l'échelle et la gestion des applications conteneurisées.

Kubernetes peut gérer des clusters de conteneurs, ce qui signifie qu'il peut gérer plusieurs conteneurs répartis sur plusieurs machines.

Ensemble, Docker et Kubernetes offrent un moyen puissant et flexible de déployer et de gérer des applications dans des environnements cloud, hybrides ou sur site.

Installation de Docker

Installer les dépendances

Premièrement, nous devons installer les dépendances nécessaires au bon fonctionnement de Docker. Commençons par mettre à jour le cache des paquets :

```
root@server1:~# sudo apt-get update
```

Puis, exécutons la commande ci-dessous pour installer les paquets :

```
root@server1:~# sudo apt-get install ca-certificates curl gnupg
```

```
root@server1:~# sudo install -m 0755 -d /etc/apt/keyrings
```

Une fois cette étape effectuée, passons à la suite

Ajouter le dépôt officiel Docker

Deuxièmement, nous devons ajouter le dépôt officiel de Docker à notre machine Debian afin de pouvoir récupérer les sources. Commençons par récupérer la clé GPG qui nous permettra de valider les paquets récupérés depuis le dépôt Docker :

```
root@server1:~# curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --de
armor -o /etc/apt/keyrings/docker.gpg
```

```
root@server1:~# sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

Ensuite, on ajoute le dépôt Docker à la liste des sources de notre machine :

```
root@server1:~# echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https
://download.docker.com/linux/debian \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
> sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
root@server1:~# sudo apt-get update
```

Installation des paquets Docker

Troisièmement, c'est l'installation de Docker qui doit être réalisée. Trois paquets sont à installer sur notre hôte pour bénéficier de l'ensemble des composants. Voici la commande à exécuter :

```
root@server1:~# sudo apt-get install docker-ce docker-ce-cli containerd.io docker-build
x-plugin docker-compose-plugin
```

Docker est-il bien installé?

L'installation des paquets est terminée, mais Docker est-il correctement installé ? Pour répondre à cette question, vous pouvez regarder le statut de Docker, ce qui sera une première indication si le service est identifié sur la machine.

```
root@server1:~# sudo service docker status
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Wed 2024-05-01 01:21:16 UTC; 32s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 10421 (dockerd)
      Tasks: 7
     Memory: 39.8M
        CPU: 622ms
     CGroup: /system.slice/docker.service
             └─10421 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd>
```

Docker est-il bien installé?

Ensuite, le meilleur moyen de vérifier si Docker est installé, c'est d'exécuter le container nommé "hello-world". La commande ci-dessous permettra de télécharger l'image de ce container et de l'exécuter.

```
root@server1:~# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:a26bff933ddc26d5cdf7faa98b4ae1e3ec20c4985e6f87ac0973052224d24302
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
```

Quelle est la version de Docker installée

Pour finir avec la phase d'installation, sachez qu'à tout moment vous pouvez voir quelle est la version de Docker que vous utilisez grâce à la commande suivante :

```
root@server1:~# docker compose version
Docker Compose version v2.27.0
```

Quelques commandes Docker

Docker est installé sur notre serveur Debian 11, mais comment faire pour l'utiliser ? Pour finir, nous allons voir quelques commandes utiles qui vous permettront de débuter avec Docker.

- Lister les containers Docker en cours d'exécution

```
root@server1:~# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Le résultat de la commande permet d'avoir une liste avec différentes informations dont l'ID unique du container, le nom de l'image, et le statut.

- Lister tous les containers Docker enregistrés sur votre machine, peu importe l'état

```
root@server1:~# docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED              STATUS
      PORTS     NAMES
f4e37c075525   hello-world   "/hello"   About a minute ago   Exited (0) About a minute
ago              gracious_shtern
```

Quelques commandes Docker

- Lister les images Docker (disponibles en local)

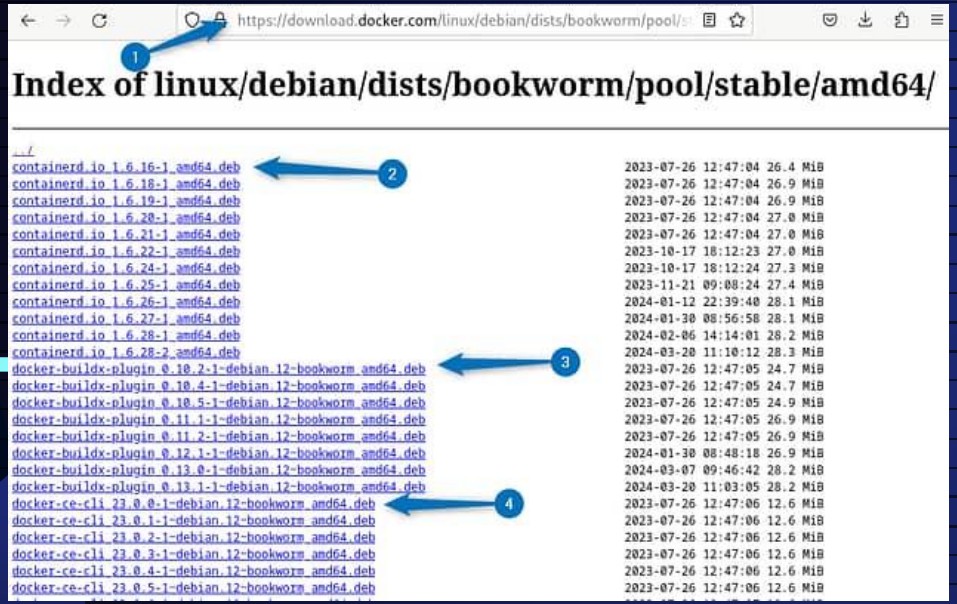
```
root@server1:~# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    d2c94e258dcb   12 months ago   13.3kB
```

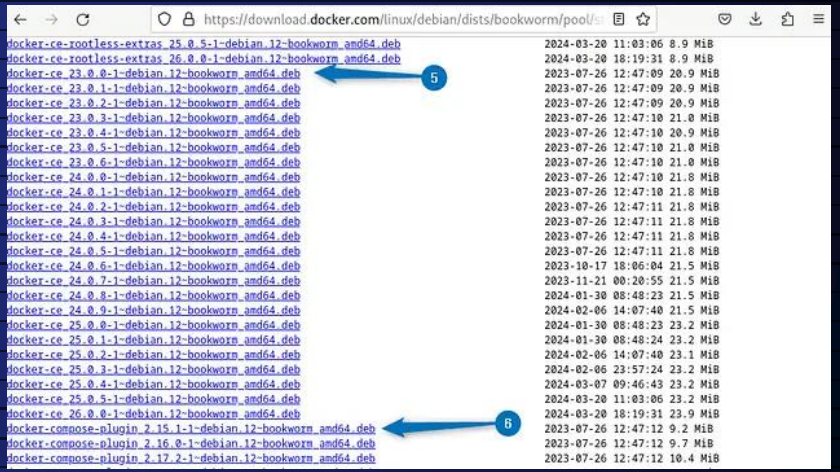
- Télécharger une image Docker à partir de Docker Hub

Le site Docker Hub référence les images Docker, et il est possible de télécharger une image à partir de cette source. Voici un exemple où l'image d'un container Docker Apache est téléchargée :

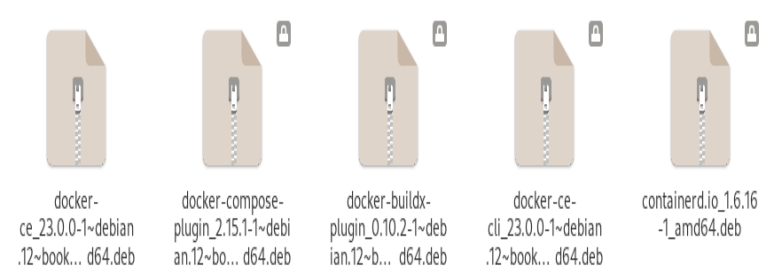
```
root@server1:~# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
b0a0cf830b12: Downloading  5.626MB/29.15MB
851c52adaa9b: Download complete
4f4fb700ef54: Download complete
39d9f60535a6: Downloading  2.61MB/4.007MB
943a2b3cf551: Downloading  5.834MB/26.03MB
ea83e81966d6: Waiting
```

Pour installer Docker sur Debian 12 à l'aide du fichier deb, il est possible d'installer sa dernière version, mais le problème est que nous devons télécharger chaque paquet, puis l'installer manuellement. Pour télécharger tous les paquets, visitez la **https://download.docker.com/linux/debian/dists/bookworm/pool/stable/amd64/** , puis cliquez sur la version souhaitée de chaque module pour obtenir leurs fichiers deb :





Vous devez avoir ces cinq fichiers .deb après téléchargements



Une fois que nous avons téléchargé tous les fichiers deb, nous pouvons vérifier en vérifiant le dossier de téléchargement correspondant avec la commande **ls**.

```
root@debian:/home/mariamd/Téléchargements# ls
containerd.io_1.6.16-1_amd64.deb
docker-buildx-plugin_0.10.2-1~debian.12~bookworm_amd64.deb
docker-ce_23.0.0-1~debian.12~bookworm_amd64.deb
docker-ce-cli_23.0.0-1~debian.12~bookworm_amd64.deb
docker-compose-plugin_2.15.1-1~debian.12~bookworm_amd64.deb
```

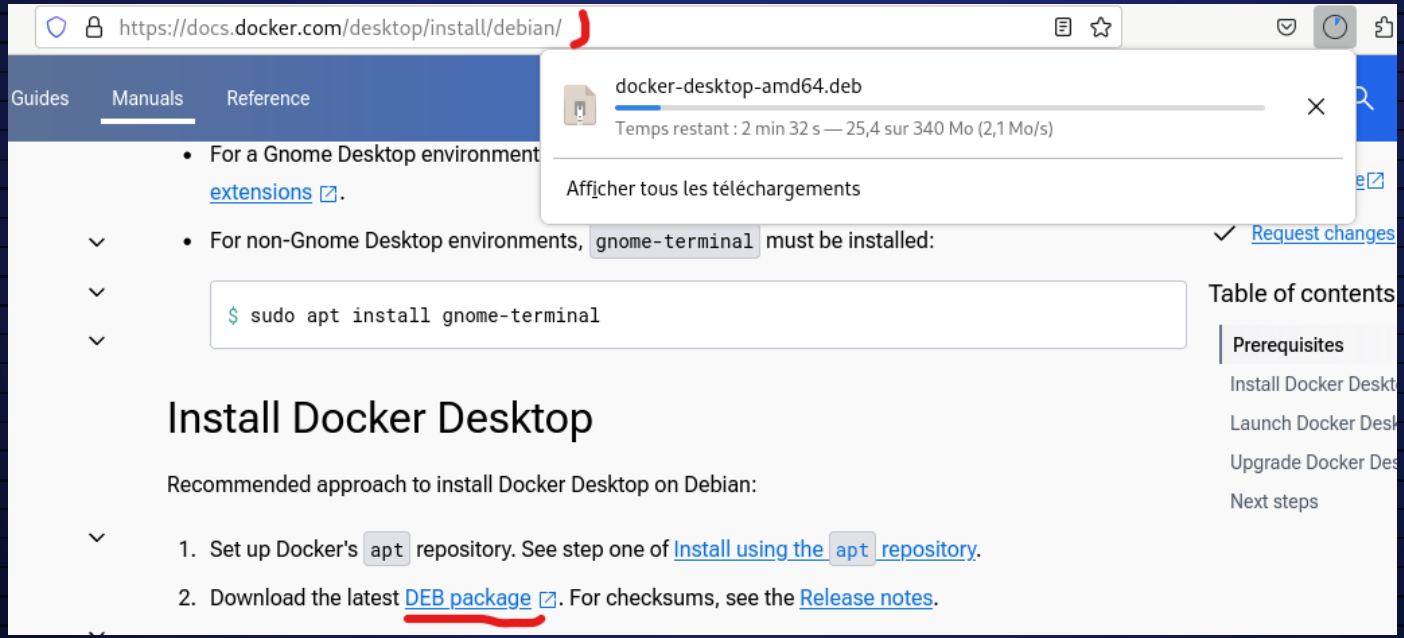
Utilisons le gestionnaire de paquets Debian pour installer tous ces modules, nous pouvons définir la commande ci-dessous si nous avez téléchargé différentes versions :

```
sudo dpkg -i ./containerd.io_1.6.16-1_amd64.deb \ ./docker-ce_23.0.0-1~debian.12~bookworm_amd64.deb \ ./docker-ce-cli_23.0.0-1~debian.12~bookworm_amd64.deb \ ./docker-buildx-plugin_0.10.2-1~debian.12~bookworm_amd64.deb \ ./docker-compose-plugin_2.15.1-1~debian.12~bookworm_amd64.deb
```

Démarrons maintenant le service Docker, puis vérifions-le en vérifiant son état à l'aide de l'utilitaire systemctl :

```
root@debian:/home/mariamd/Téléchargements# sudo systemctl start docker
sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Wed 2024-08-14 22:19:12 CEST; 1min 18s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 3347 (dockerd)
      Tasks: 7
     Memory: 95.8M
        CPU: 225ms
     CGroup: /system.slice/docker.service
             └─3347 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

Docker, comme mentionné précédemment, est fourni avec une version GUI qui est idéale pour les utilisateurs qui doivent se familiariser avec les commandes sur Debian 12 ou d'autres distributions Linux. De plus, la version GUI facilite également l'utilisation de Docker. Docker Desktop peut être installé via son fichier deb, donc pour le télécharger, visitons la page de téléchargement du site officiel via ce lien: **https://docs.docker.com/desktop/install/debian/**

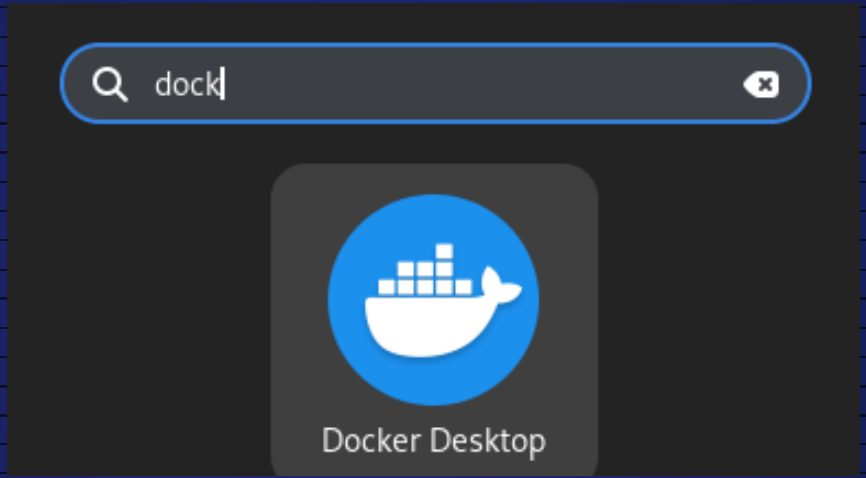


```
root@debian:/home/mariamd/Téléchargements# ls
containerd.io_1.6.16-1_amd64.deb
docker-buildx-plugin_0.10.2-1~debian.12~bookworm_amd64.deb
docker-ce_23.0.0-1~debian.12~bookworm_amd64.deb
docker-ce-cli_23.0.0-1~debian.12~bookworm_amd64.deb
docker-compose-plugin_2.15.1-1~debian.12~bookworm_amd64.deb
docker-desktop-amd64.deb
```

Utilisons maintenant le programme d’installation de paquets par défaut de Debian pour installer le bureau Docker à l’aide de son fichier deb :

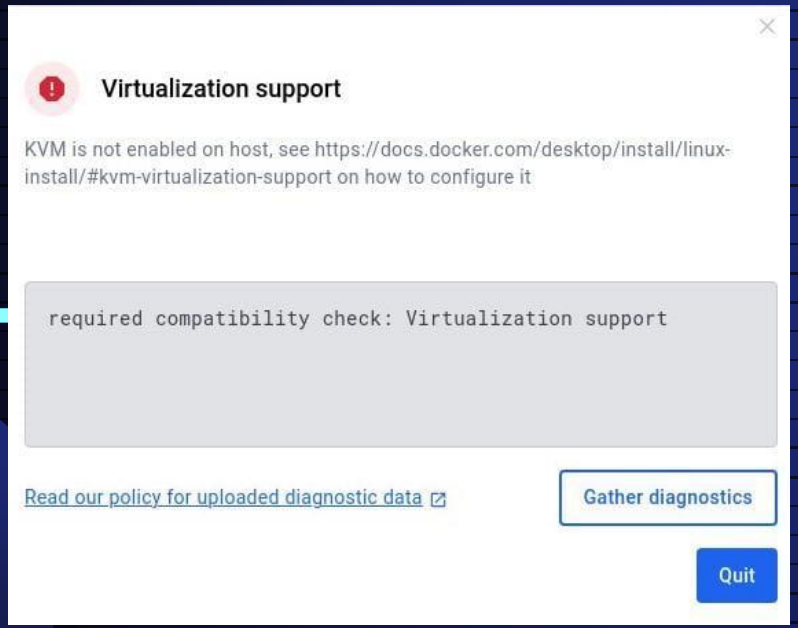
sudo apt install ./docker-desktop-amd64.deb

Lançons l'application de bureau Docker en la recherchant dans le menu de l'application :

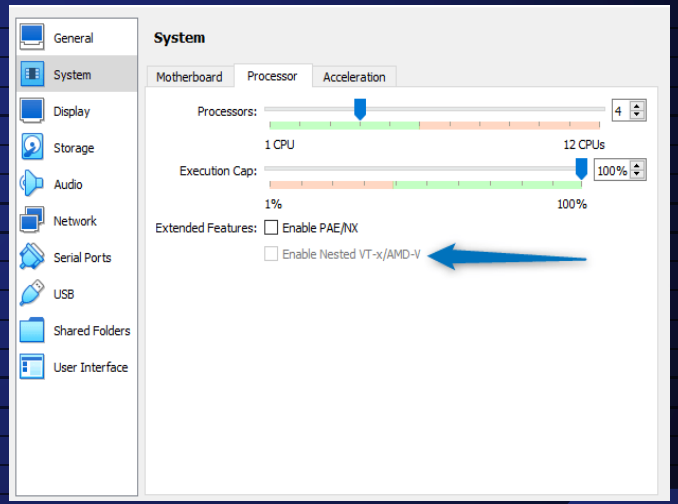


Comment résoudre le problème KVM non activé sur l'hôte dans Debian 12 ?

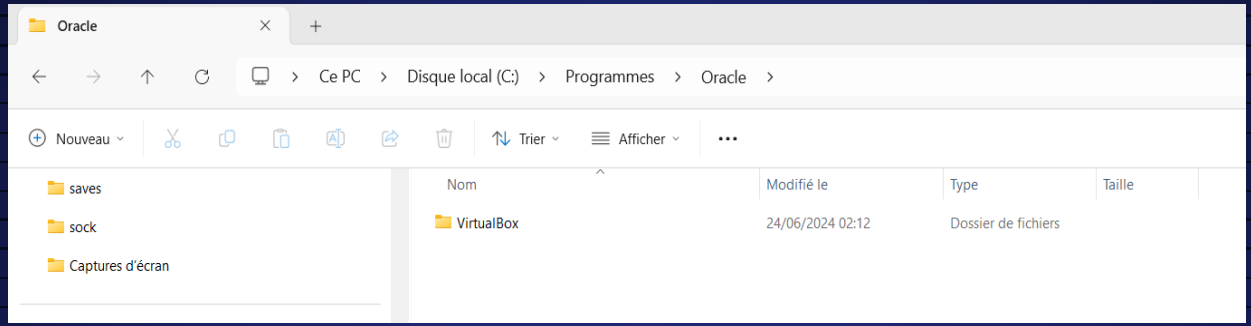
Si nous voyons une erreur KVM non activé, cela signifie que la virtualisation n'est pas activée sur notre système :

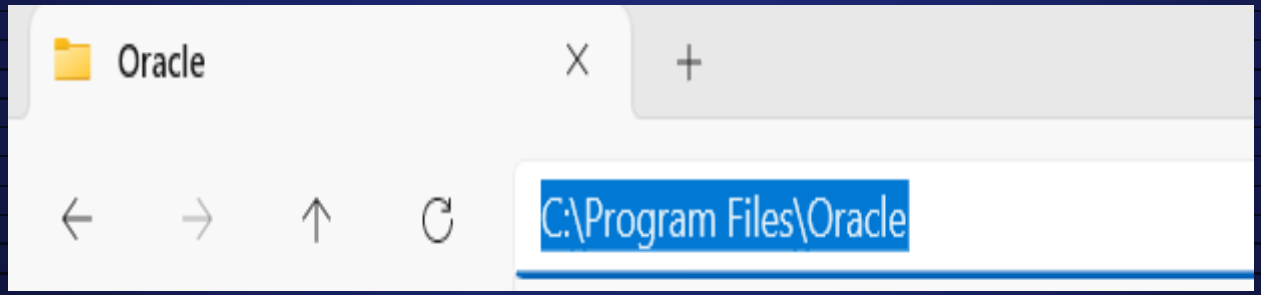


Nous avons exécuté Debian 12 sur Virtualbox, donc la méthode pour activer la prise en charge de la virtualisation est différente. Cependant, si nous avons installé Debian sur le système, il nous suffit d'activer la virtualisation à partir du menu du BIOS. L'image ci-dessous montre l'option que nous devons activer pour activer la prise en charge de la virtualisation pour le bureau Docker :



Pour activer la virtualisation, nous devons d'abord copier le chemin du répertoire où la boîte virtuelle est installée sur notre système :





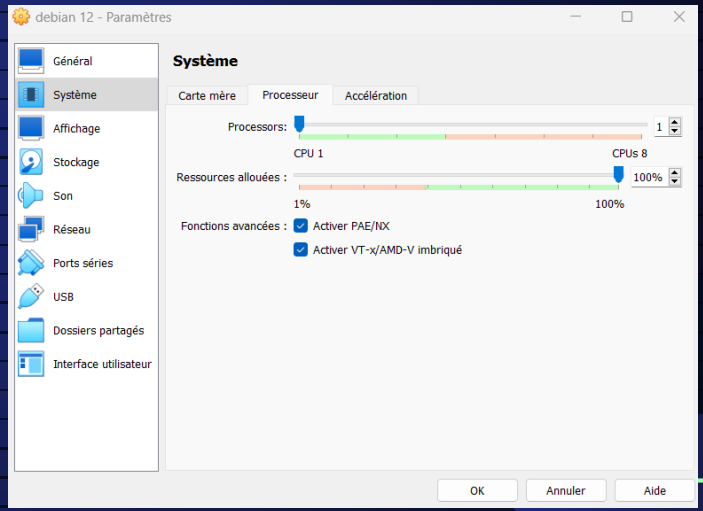
Lançons maintenant PowerShell avec les droits d'administrateur et accédons au répertoire de VirtualBox :

```
PS C:\Users\AAA> cd "C:\Program Files\Oracle\VirtualBox"
```

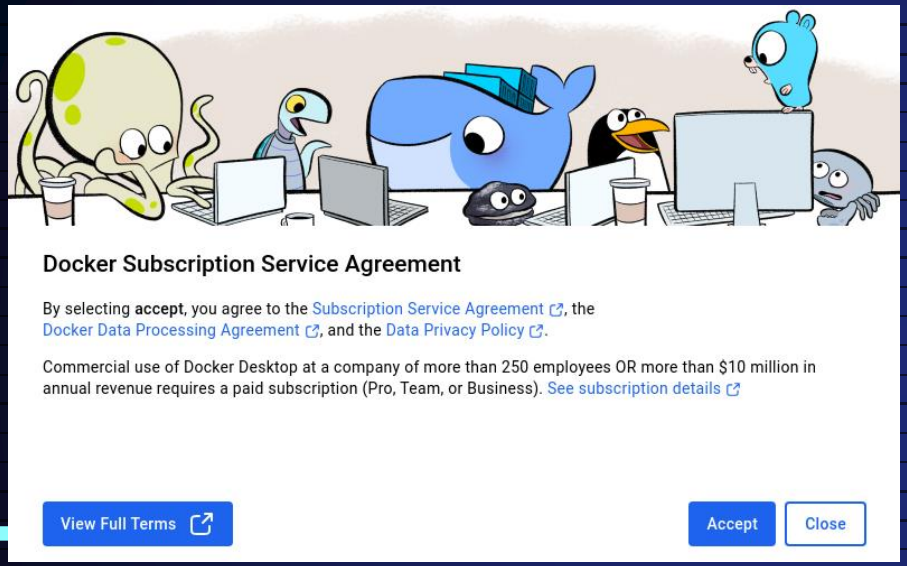
Utilisons maintenant la commande modifyvm avec le nom de la machine et activons la virtualisation en utilisant :

```
PS C:\Program Files\Oracle\VirtualBox> .\VBoxManage modifyvm "debian 12" --nested-hw-virt on
PS C:\Program Files\Oracle\VirtualBox>
```

Ensuite, vérifions les modifications en accédant aux paramètres du processeur sous l'option système de VirtualBox :



Lançons maintenant Docker sur Debian 12 et le problème sera résolu :

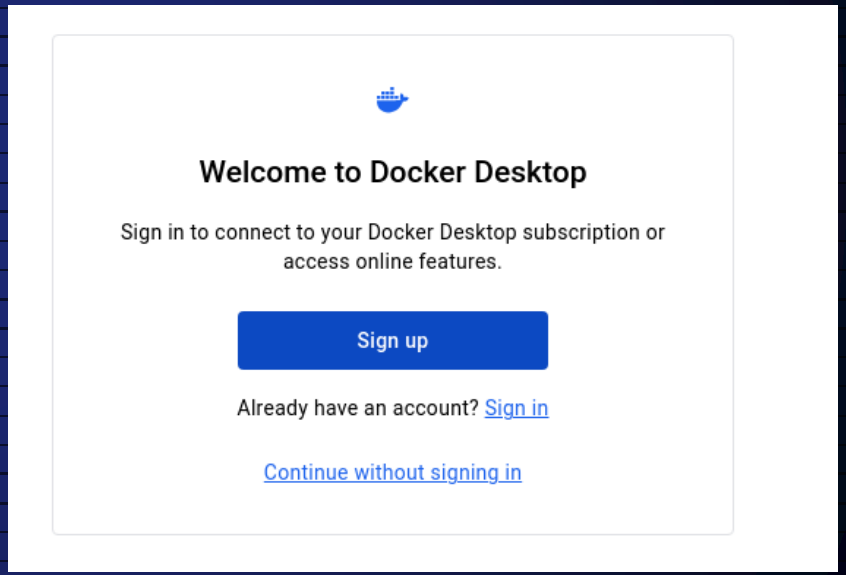


Docker Subscription Service Agreement

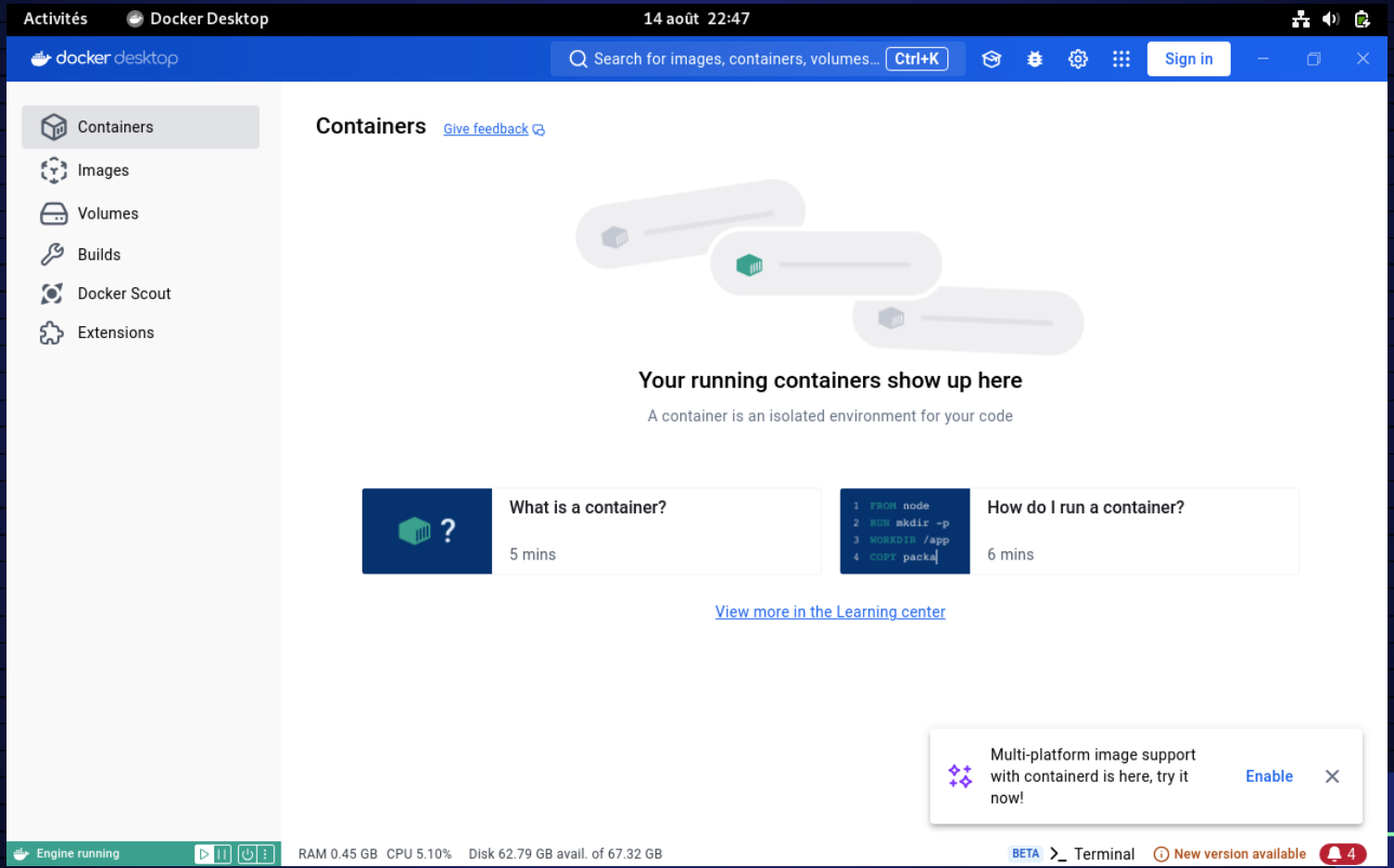
By selecting **accept**, you agree to the Subscription Service Agreement, the Docker Data Processing Agreement, and the Data Privacy Policy.

Commercial use of Docker Desktop at a company of more than 250 employees OR more than $10 million in annual revenue requires a paid subscription (Pro, Team, or Business). See subscription details

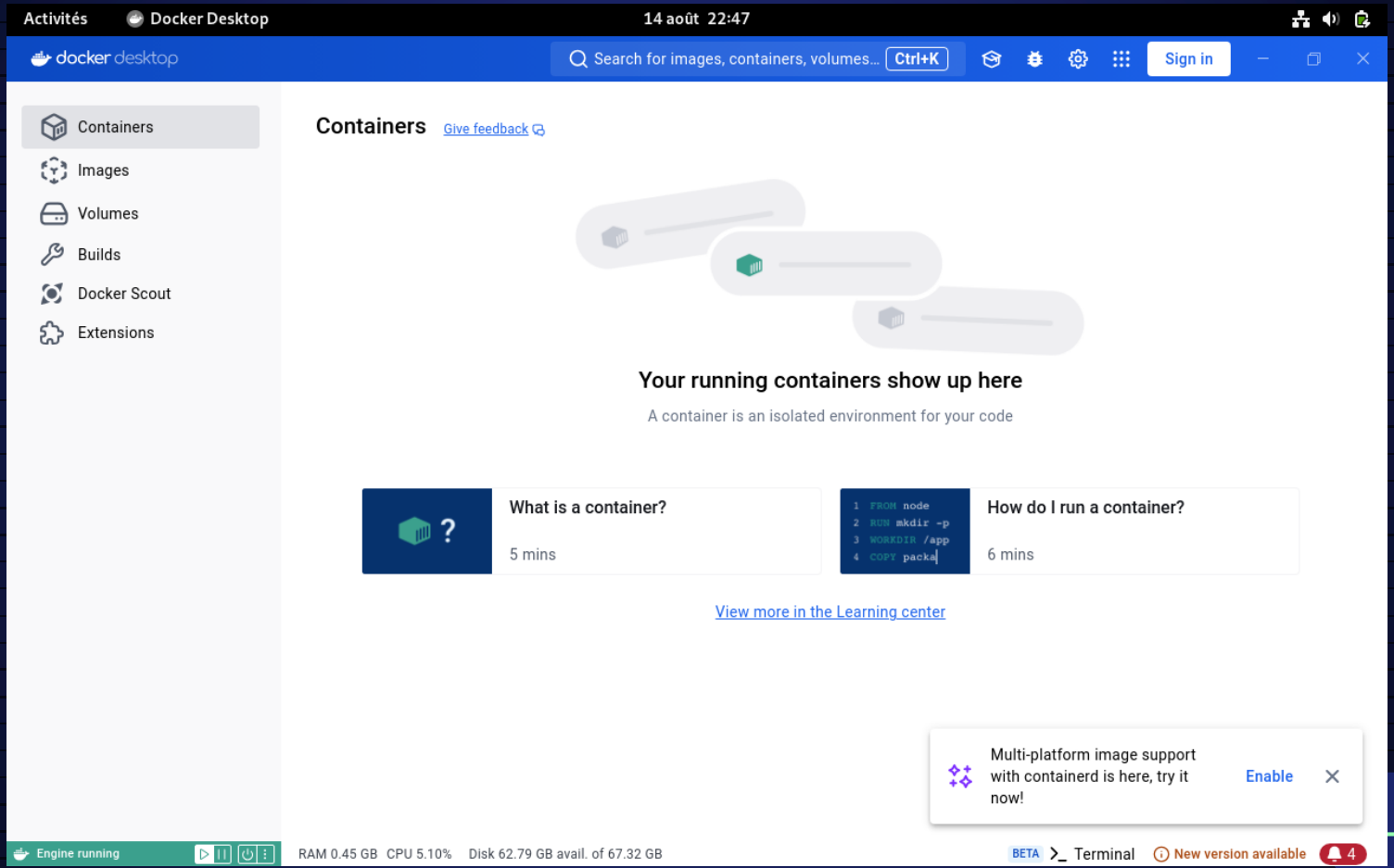
View Full Terms | Accept | Close



Interface de Docker



Interface de Docker



Installation de Kubernetes

Installation du binaire kubectl avec curl

```
root@server1:~#  curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/rel
ease/stable.txt)/bin/linux/amd64/kubectl.sha256"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0    419      0 --:--:-- --:--:-- --:--:--   419
100    64  100    64    0     0    112      0 --:--:-- --:--:-- --:--:--   112
```

Validation du binaire kubectl par rapport au fichier de somme de contrôle

```
root@server1:~# echo "$(cat kubectl.sha256)  kubectl" | sha256sum --check
kubectl: OK
```

Installation de kubectl

```
root@server1:~# sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
```

Testons pour nous assurer que la version que nous avons installée est à jour

```
root@server1:~# kubectl version --client
Client Version: v1.30.0
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

Ou utilisons ceci pour une vue détaillée de la version :

```
root@server1:~# kubectl version --client --output=yaml
clientVersion:
  buildDate: "2024-04-17T17:36:05Z"
  compiler: gc
  gitCommit: 7c48c2bd72b9bf5c44d21d7338cc7bea77d0ad2a
  gitTreeState: clean
  gitVersion: v1.30.0
  goVersion: go1.22.2
  major: "1"
  minor: "30"
  platform: linux/amd64
kustomizeVersion: v5.0.4-0.20230601165947-6ce0bf390ce3
```

Mettons à jour l' apt index des packages et installons les packages nécessaires pour utiliser le apt référentiel Kubernetes :

```
root@server1:~# sudo apt-get update
```

```
root@server1:~# sudo apt-get install -y apt-transport-https ca-certificates curl
```

Téléchargeons la clé de signature publique pour les référentiels de packages Kubernetes. La même clé de signature est utilisée pour tous les référentiels, nous pouvons donc ignorer la version dans l'URL :

```
root@server1:~# echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] http
s://pkgs.k8s.io/core:/stable:/v1.30/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernet
es.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/
stable:/v1.30/deb/ /
root@server1:~# sudo chmod 644 /etc/apt/sources.list.d/kubernetes.list
```

Mettons à jour apt l’index du package, puis installons kubectl

```
root@server1:~# sudo apt-get update
```

```
root@server1:~# sudo apt-get install -y kubectl
```

installons kubectl à l'aide du gestionnaire de packages snap prenant en charge snap.

```
root@server1:~# apt install snapd
```

```
root@server1:~# snap install kubectl --classic
```

CONCLUSION

Docker et Kubernetes sont des outils puissants qui peuvent transformer la façon dont les applications sont développées, déployées et gérées. Si nous envisageons d'utiliser des conteneurs pour nos applications, Docker et Kubernetes sont des technologies essentielles à prendre en compte.

Merci de votre aimable attention!