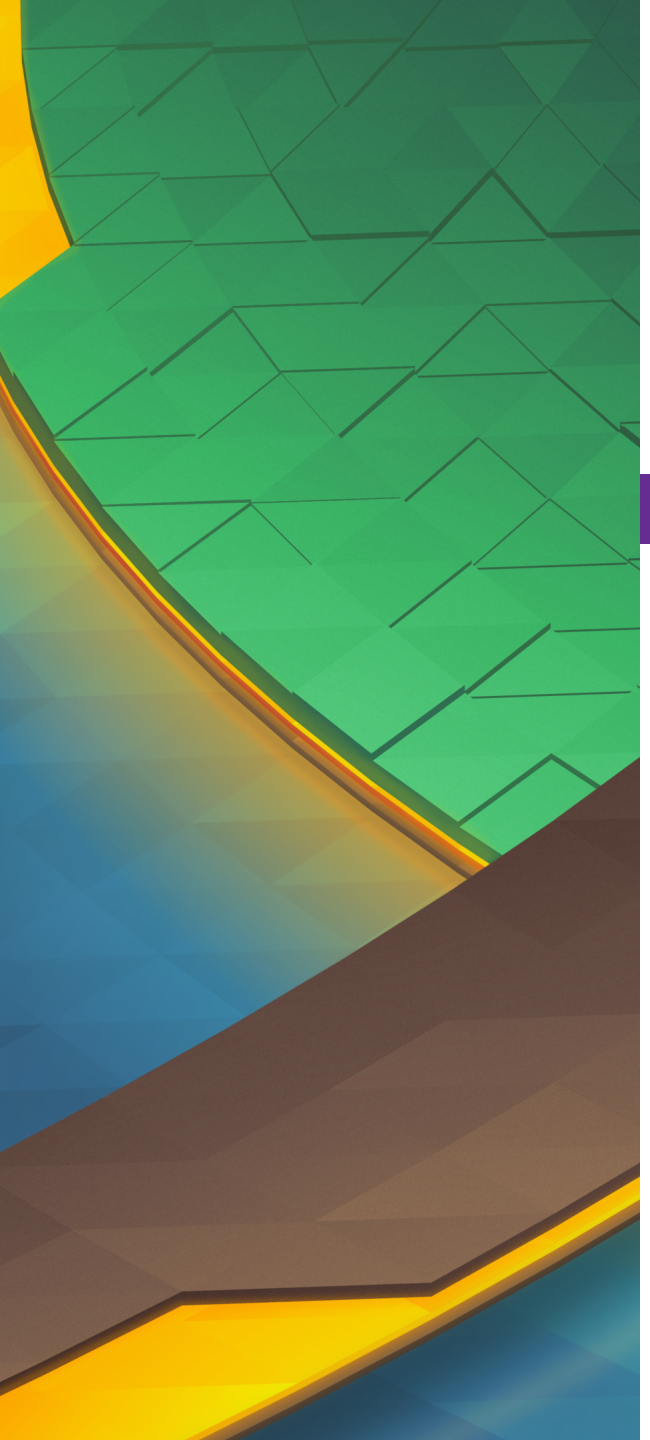
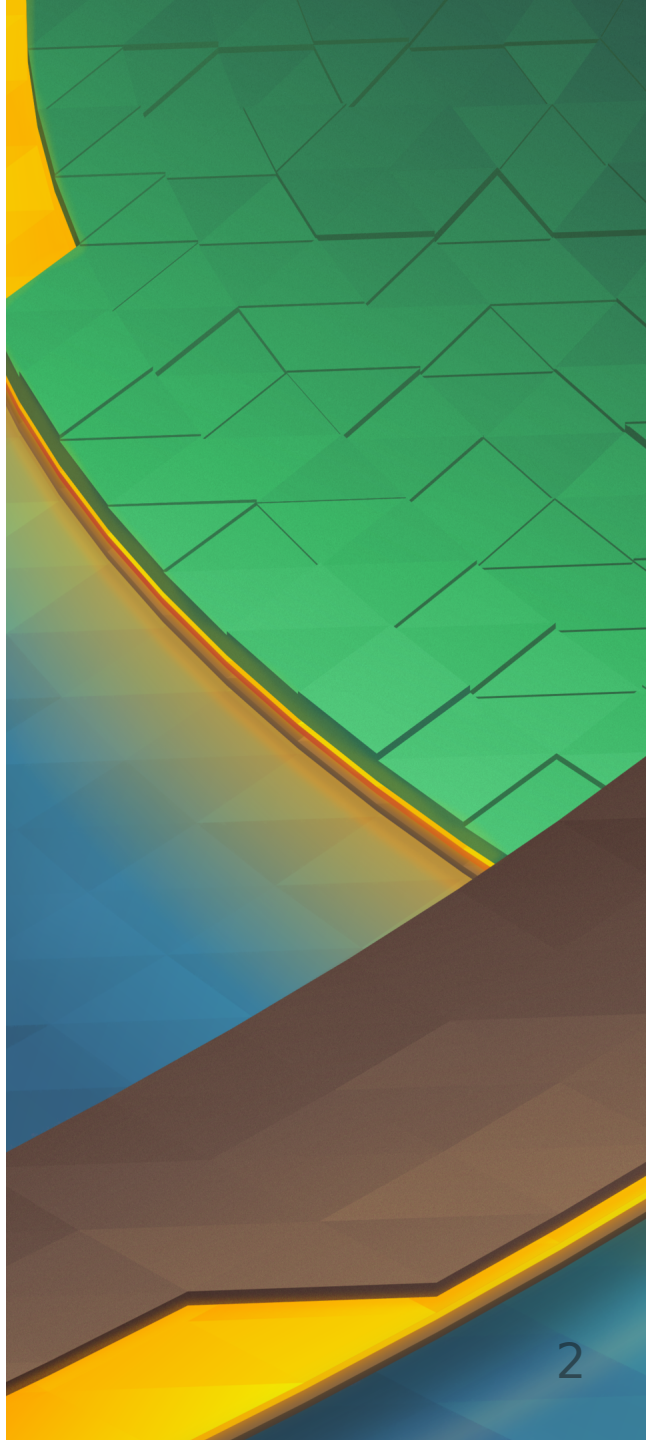


Prérequis pour la formation

Sur l'environnement *Debian* ou *Ubuntu* (Linux)

Glenn ROLLAND / glenux@glenux.net

Prérequis matériels et systèmes



Vue d'ensemble

Les pages suivantes fournissent un guide détaillé pour la préparation des postes de travail des participants en vue d'une formation dans la thématique DevOps.

Ce document liste :

- **les prérequis de configuration matérielle et système**,
- **les étapes d'installation** des logiciels nécessaires,
- **les étapes de vérification** pour assurer un environnement de formation optimal.

Caractéristiques matérielles

- **Processeur récent, compatible Intel x86**:
  - ... en 64 bits,
  - ... 8 coeurs mininum (octo-core),
  - ... cadencé à 2.5 Ghz minimum ;
- **Plus de 24 Go de mémoire RAM** au total ;
- **Plus de 50 Go d'espace libre** sur le disque dur (SSD ou NVMe si possible).

Caractéristiques du réseau

- **Un connexion réseau stable**, si possible filaire pour chacun des postes de travail ;
- **Un accès à internet adapté à la formation**, c'est à dire:
  - sans proxy filtrant, transparent ou non,
  - sans VPN, et sans routage dynamique sur les postes,
  - avec un débit descendant supérieur à 10 Mb/s dédié à chacun des postes ;
- **Être isolé des réseaux et systèmes de production** de l'organisation.

Configuration système

- **Un système d'exploitation hôte et non-virtualisé,** parmi les suivants :
  - Linux Debian 12 *(Bookworm)* ou plus récent,
  - Ubuntu 22.04 *(Jammy Jellyfish)* ou plus récent,
- **Les droits d'administration** sur la machine:
  - pour installer des logiciel,
  - pour autoriser l'ouverture et le NAT de ports dans le firewall,
  - pour les autres permissions (périphériques, etc.) nécessaires à l'exécution de machines virtuelles ;
- **La compatibilité avec Qemu + KVM**:
  - Activer la virtualisation activée dans le BIOS / EFI,
  - Désactiver les autres logiciels de virtualisation (VMWare, Hyper-V, etc.), tant coté drivers que démarrage système, car ces outils manipulent tous les mêmes ressources du système et entrent en conflit.

Configuration logicielle

- Installer Git
- Installer QEmu + KVM + Libvirt
- Installer Vagrant
- Installer Vagrant-Libvirt (plugin)
- Installer Visual Studio Code

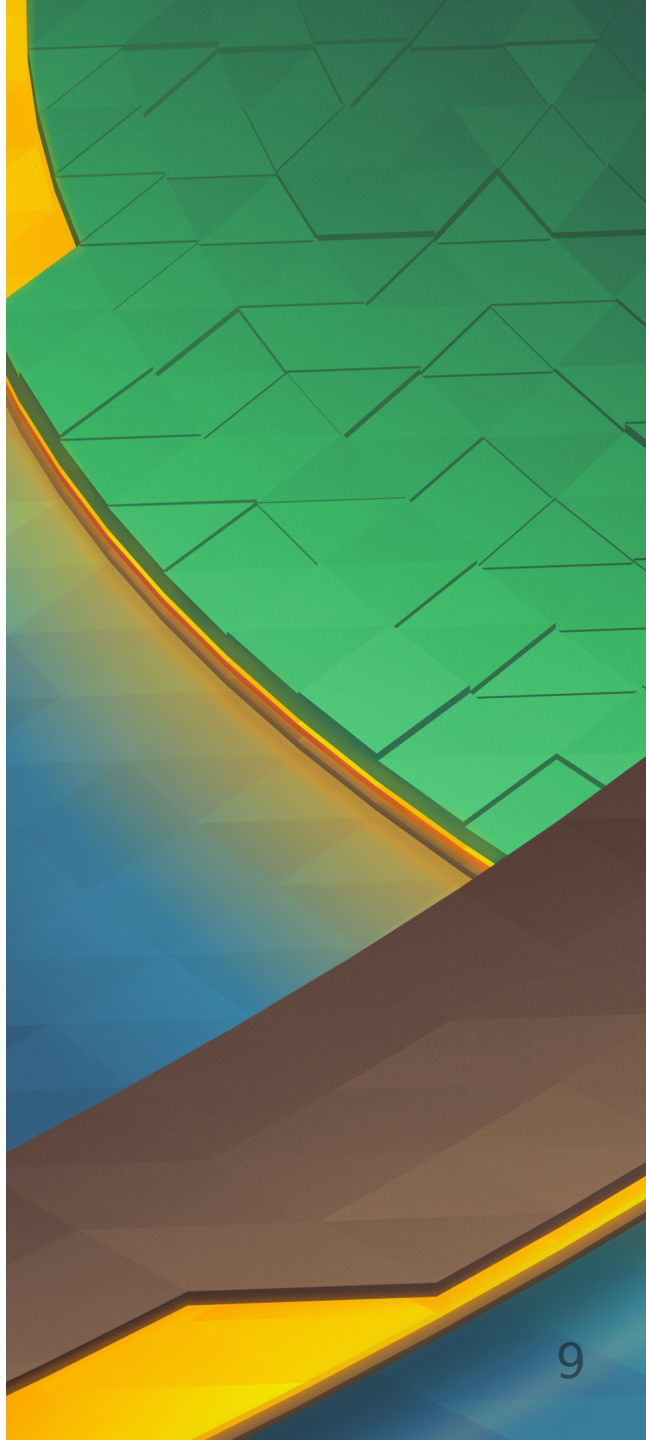
Le détail des étapes d'installation et de configuration est expliqué dans la partie Installation et configuration de ce document.

Vérification du bon fonctionnement

- Vérifier que le firewall ne gène pas
- Vérifier que le NAT interne de Qemu + KVM fonctionne correctement

Le détail de ces étapes est expliqué dans la partie Vérification du bon fonctionnement de ce document.

Installation et configuration



Avant l'installation

Les systèmes de virtualisation ne cohabitent pas bien entre eux car ils nécessitent d'interférer avec le fonctionnement normal du noyau: *scheduler*, périphériques réseaux et firewall.

Pour cela ils utilisent des pilotes du noyau qui essayent de prendre la main sur ces aspects... et s'ils sont plusieurs à faire la même chose, cela dysfonctionne.

Il est donc nécessaire de désactiver les pilotes du noyau et le service système des autres logiciels de virtualisation, tels que VMWare ou Microsoft Hyper-V pour éviter les conflits logiciels qui pourraient survenir lors de l'utilisation des outils de la formation (Qemu + KVM + Libvirt + Vagrant).

La méthode pour désactiver les autres systèmes de virtualisation présents sur le système dépend des logiciels et des versions utilisées. Référez-vous à la documentation des logiciels concernés.

Exemple: désactiver VMware

Taper dans un terminal:

```
$ sudo update-rc.d name disable vmware
```

Et ensuite redémarrer.

Installer Git

Présentation

Git permet de versionner les différentes versions du code source et de configuration que les participants produiront.

Procédure d'installation

1. Lancer un terminal,
2. Taper ensuite les commandes suivantes :

```
$ sudo apt update
$ sudo apt install git
```

Installer QEmu + KVM + Libvirt

Présentation

Qemu est un hyperviseur. C'est à dire qu'il gère la simulation et le bon fonctionnement de machines virtuelles.

KVM est un module de para-virtualisation, faisant partie du noyau Linux, et qui accélère le fonctionnement de Qemu.

Libvirt fournit un *daemon* et une bibliothèque d'abstraction facilisant l'utilisation de Qemu par d'autres programmes (ex: Vagrant).

Les environnements de travail et les exercices manipulés par les participants durant la formation ont été préparés pour fonctionner sur Qemu + KVM + Libvirt.

Installer QEmu + KVM + Libvirt (2)

Procédure d'installation

1. Lancer un terminal
2. Taper ensuite les commandes suivantes :

```
$ sudo apt-get install -y qemu-utils \
    libvirt-daemon-system \
    libvirt-dev ebtables cpu-checker \
    libguestfs-tools bridge-utils \
    ruby-fog-libvirt

$ sudo adduser vagrant libvirt
```

Installer Vagrant

Présentation de l'outil

Vagrant est un outil de gestion automatisée d'hyperviseur et de configuration automatique (*provisioning*)

Vagrant permet de controler les différents hyperviseurs pour créer et charger automatiquement des images virtuelles avec différents systèmes et paramètres prédéfinis depuis un fichier de configuration, de simuler un topologie réseau avec plusieurs VM, d'automatiser leur installation, ainsi qu'accéder aux machines virtuelles en réseau via le protocole SSH.

Installer Vagrant (2)

Procédure d'installation

1. Lancer un terminal
2. Taper ensuite les commandes suivantes :

```
$ curl -fsSL https://apt.releases.hashicorp.com/gpg \
      | sudo apt-key add -
$ sudo apt-add-repository \
      "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb_release -cs) main"
$ sudo apt-get update
$ sudo apt-get install vagrant
```

Installer Vagrant-Libvirt (plugin)

Procédure d'installation

1. Lancer un terminal
2. Taper ensuite la commande suivantes pour installer les prérequis :

```
$ sudo apt-get install -y libxslt-dev libxml2-dev zlib1g-dev ruby-dev gcc make
```

3. Taper enfin la commande suivantes (en tant qu'utilisateur non-root) pour installer le plugin :

```
$ vagrant plugin install vagrant-libvirt
```

Éditeur Visual Studio Code

Présentation

Visual Studio Code (VSCode) est l'éditeur de code proposé en open-source par Microsoft.

VSCode intègre de nombreux plugins qui permettent de manipuler et vérifier facilement la syntaxe de différents langages.

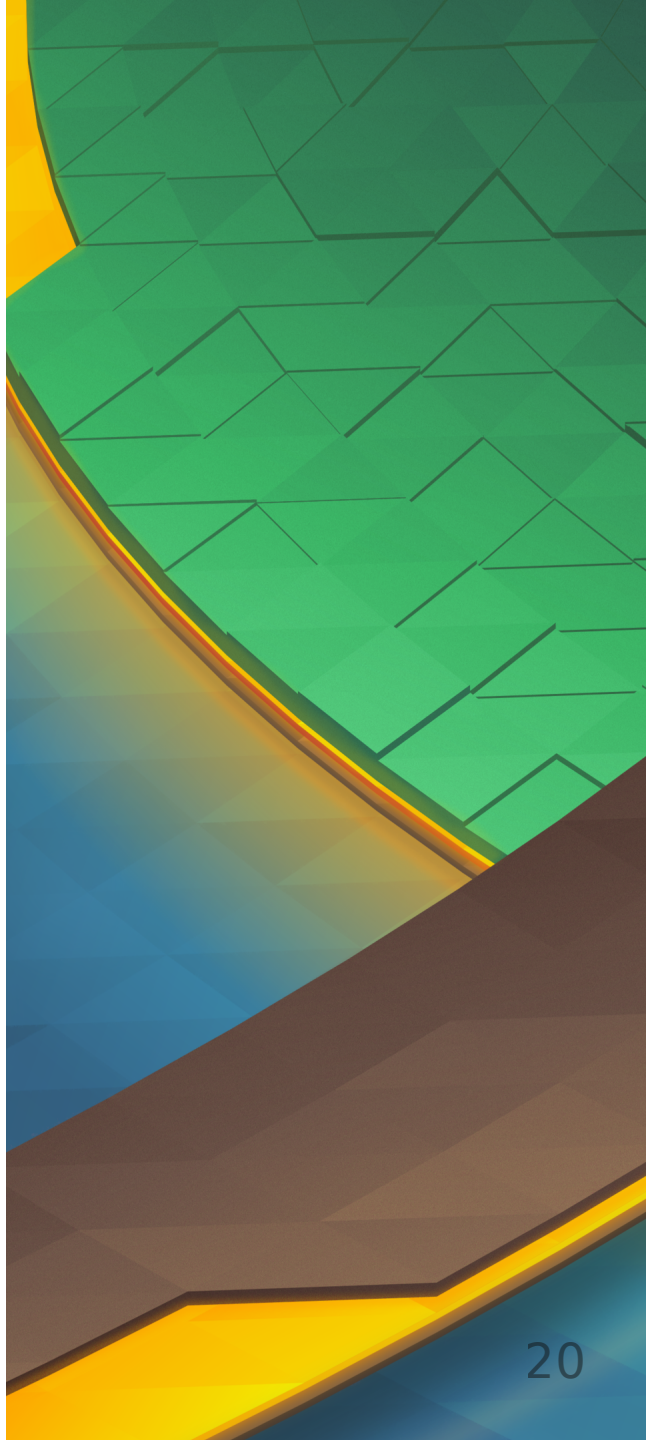
Il permet en outre de manipuler facilement (avec le plugin *remote-ssh*) des fichiers qui seraient dans une VM où sur une machine distante comme s'il étaient en local.

Éditeurs Visual Studio Code (2)

Procédure d'installation

```
$ curl https://packages.microsoft.com/keys/microsoft.asc \
    | gpg --dearmor > microsoft.gpg
$ sudo install -o root -g root -m 644 microsoft.gpg \
    /usr/share/keyrings/microsoft-archive-keyring.gpg
$ sudo sh -c 'echo "deb [arch=amd64,arm64,armhf \
    signed-by=/usr/share/keyrings/microsoft-archive-keyring.gpg] \
    https://packages.microsoft.com/repos/vscode stable main" \
    | sudo tee /etc/apt/sources.list.d/vscode.list'
$ sudo apt-get update
$ sudo apt-get install code
```

Vérifier le bon fonctionnement



Vérifier l'installation de git

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
git version
```

3. Le resultat suivant devrait s'afficher:

```
git version 2.39.0
```

Note : la version de Git peut être plus récente, ce n'est pas gênant pour la formation.

Vérifier l'installation de Qemu

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
$ qemu-system-x86_64 --version
```

3. Le resultat suivant devrait s'afficher:

```
QEMU emulator version 8.2.1 (Debian 1:8.2.1+ds-2)
Copyright (c) 2003-2023 Fabrice Bellard and the QEMU Project developers
```

Note : la version de Qemu peut être plus récente, ce n'est pas gênant pour la formation.

Vérifier l'installation de KVM

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
$ lsmod |grep ^kvm
```

3. Le resultat suivant devrait s'afficher:

```
kvm_intel             413696  0
kvm                  1363968  1 kvm_intel
```

Note : si vous avez un processeur AMD, vous verrez `kvm_amd` à la place de `kvm_intel`.

Vérifier l'installation de KVM (2)

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
$ kvm-ok
```

3. Le resultat suivant devrait s'afficher:

```
INFO: /dev/kvm exists
KVM acceleration can be used
```

Vérifier l'installation de Libvirt

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
$ sudo systemctl status libvirtd
```

3. Le resultat suivant devrait s'afficher:

```
○ libvirtd.service - Virtualization daemon
     Loaded: loaded (/lib/systemd/system/libvirtd.service; enabled; preset: enabled)
     Active: inactive (dead) since Tue 2024-03-12 15:04:23 UTC; 29s ago
   Duration: 2min 105ms
TriggeredBy: ● libvirtd-ro.socket
             ● libvirtd.socket
             ● libvirtd-admin.socket
       Docs: man:libvirtd(8)
             https://libvirt.org
    Process: 19922 ExecStart=/usr/sbin/libvirtd $LIBVIRTD_ARGS (code=exited, status=0/SUCCESS)
   Main PID: 19922 (code=exited, status=0/SUCCESS)
        CPU: 140ms
```

Vérifier l'installation de vagrant

1. Lancer un terminal
2. Taper ensuite la commande suivante :

```
vagrant version
```

3. Le resultat suivant devrait s'afficher:

```
Installed Version: 2.3.4
...
```

Note : la version de Vagrant peut être plus récente, ce n'est pas gênant pour la formation.

Vérifier le bon fonctionnement de l'environnement

Étape 1 : préparation d'un dossier de travail

1. Ouvrir un terminal
2. Taper ensuite les commandes suivantes :

```
$ mkdir -p tmp/cours-test
$ cd tmp/cours-test
$ curl -fsSL -o Vagrantfile https://bit.ly/3ephXn6
```

Vérifier le bon fonctionnement de l'environnement (2)

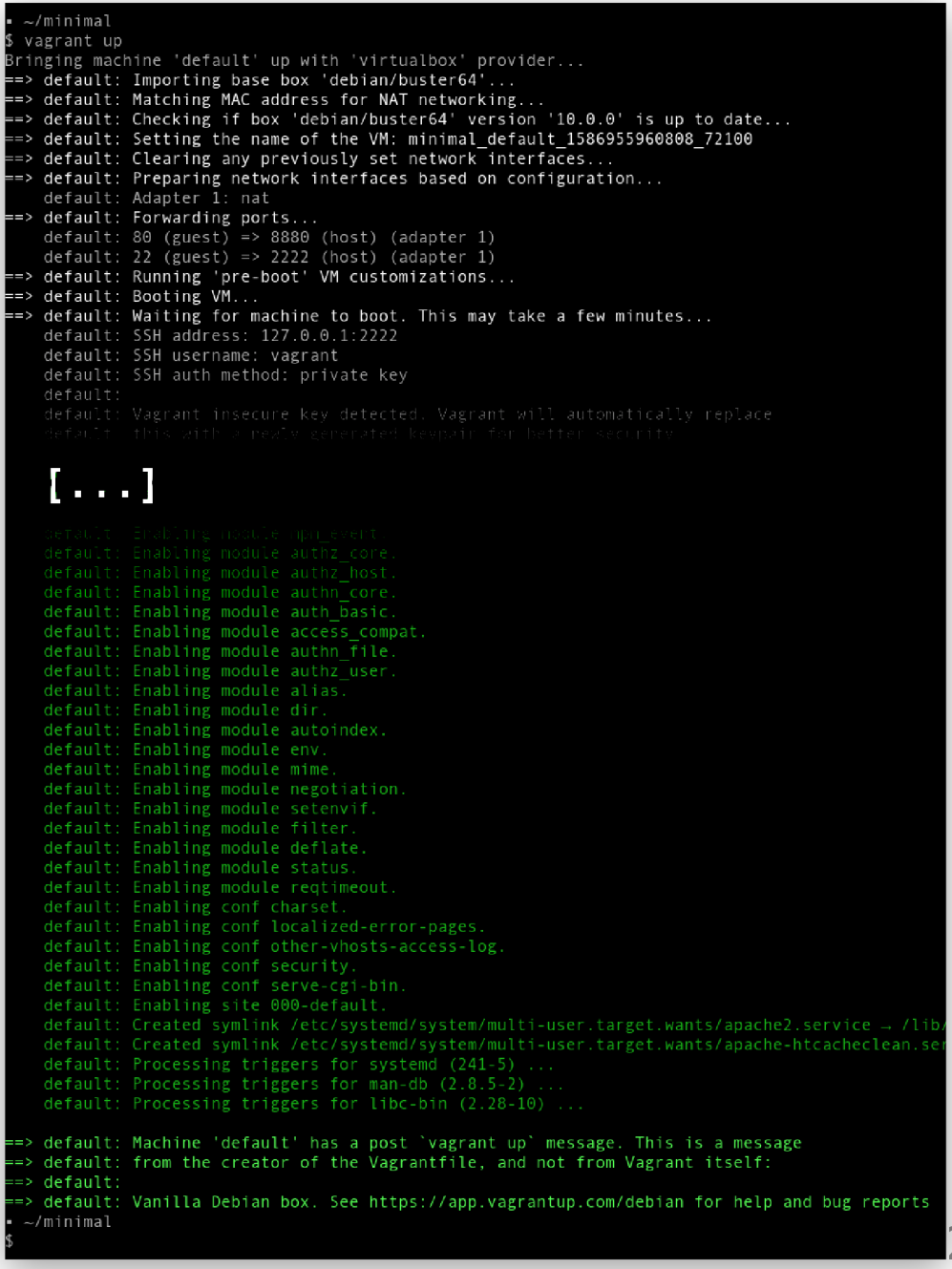
Étape 2 : lancement des machines virtuelles

1. Dans le même terminal que précédemment, taper ensuite la commande :

```
$ vagrant up --provider=libvirt
```

2. Un certain nombre de lignes s'affichent...

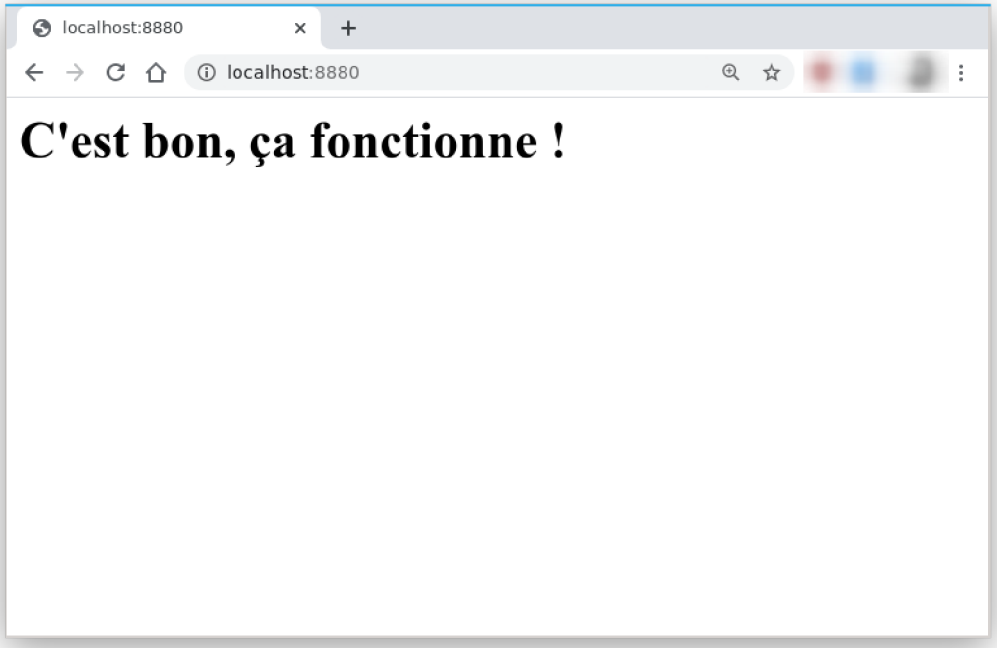
```
Bringing machine 'host' up with 'libvirt' provider...
==> host: Checking if box 'xxxxxx' version 'xxxxxx' is up to date...
==> host: Creating image (snapshot of base box volume).
==> host: Creating domain with the following settings...
==> host:  -- Name:              xxxxxxxxxx
==> host:  -- Description:       Source: xxxxxxxxxxxxxxxxx/Vagrantfile
==> host:  -- Domain type:       kvm
[...]
```

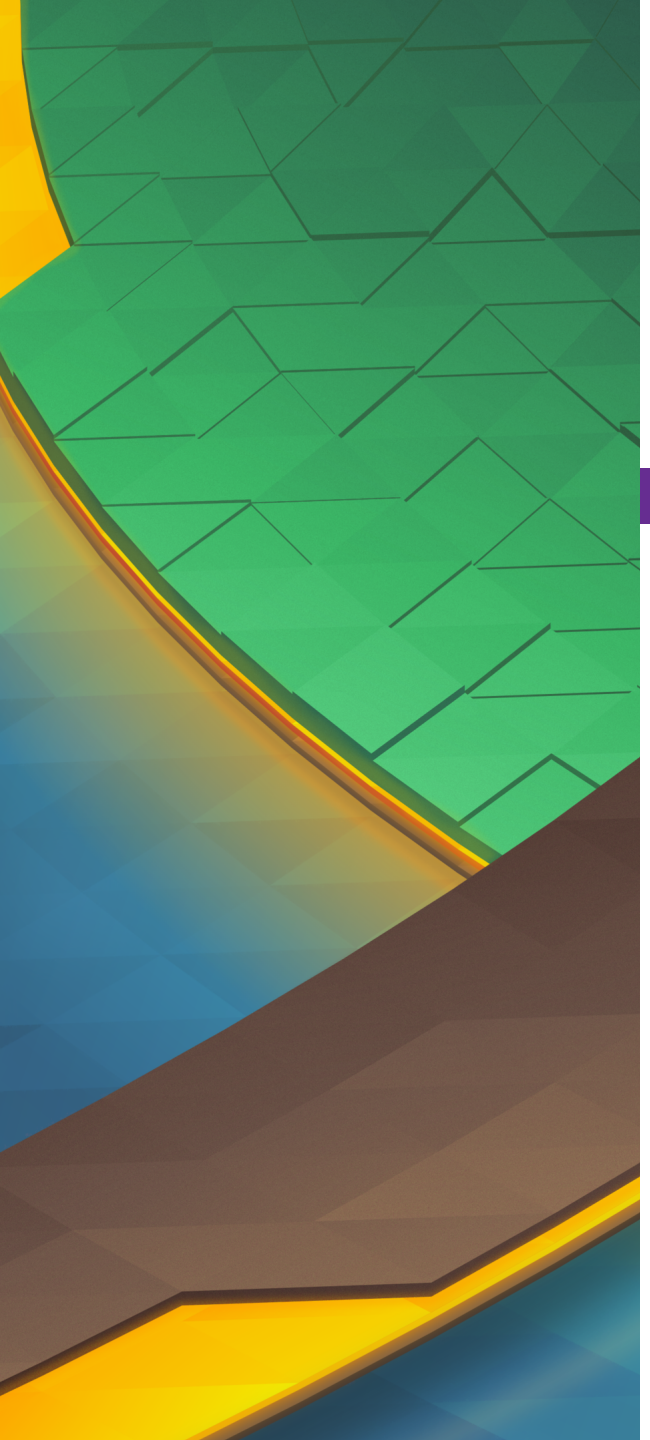


Vérifier le bon fonctionnement de l'environnement (3)

Étape 3 : vérification du pare-feu et du NAT interne

1. Ouvrir le lien http://127.0.0.1:8880 dans un navigateur.
2. Cela affiche le texte *« C'est bon, ça fonctionne ! »*





Merci pour la préparation !

Pour toute question, contacter :

Glenn ROLLAND

Email : glenux@glenux.net

Tel : +33 6 739 839 56