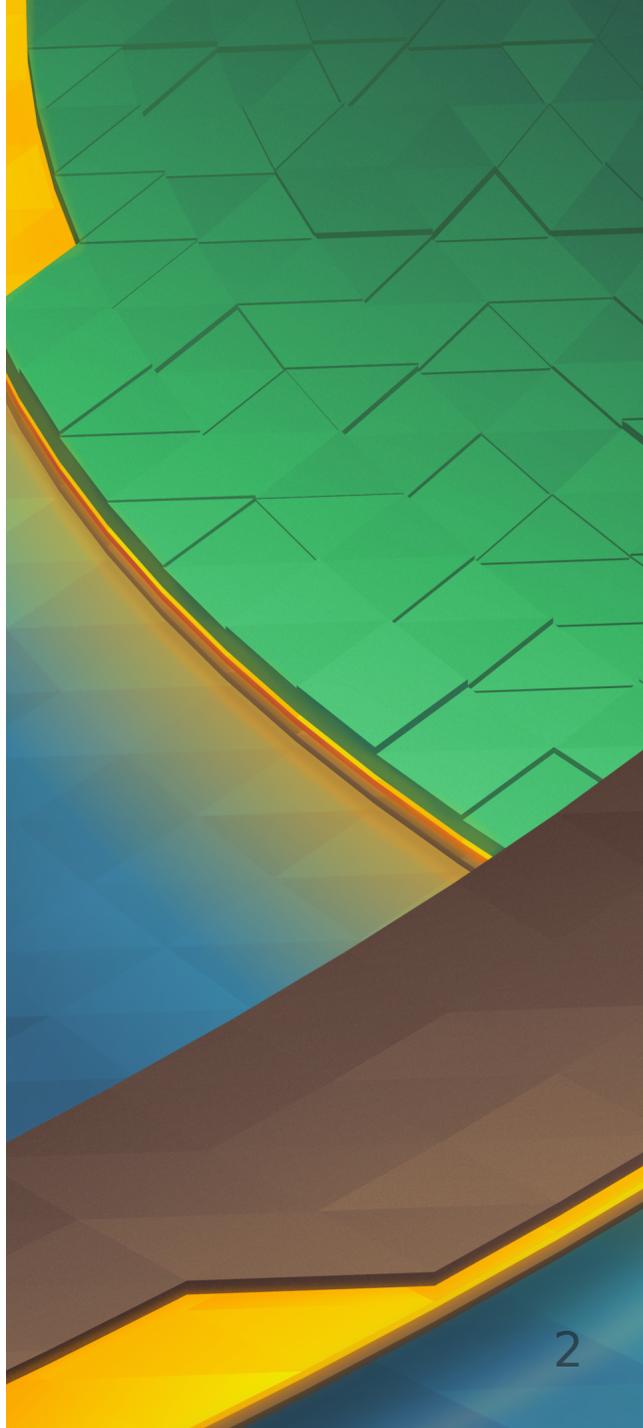


■ Prérequis pour la formation

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,
 - ... quad-core minimum,
 - ... cadencé à 2.4 Ghz minimum ;
- **Plus de 16 Go de mémoire RAM** au total ;
- **Plus de 50 Go d'espace libre** sur le disque dur.

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** parmi les suivants :
 - Linux Debian 12 (*Bookworm*) ou plus récent,
 - Ubuntu 22.04 (*Jammy Jellyfish*) ou plus récent,
 - Microsoft Windows 10 ou plus récent (voir le document "prérequis" prévu pour Microsoft Windows) ;
- **Les droits d'administration** sur la machine:
 - pour installer des logiciels,
 - 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 VirtualBox:**
 - Activer la virtualisation activée dans le BIOS / EFI,
 - Désactiver les autres systèmes de virtualisation (VMWare, Hyper-V, etc.) tant côté drivers que démarrage système car les conflits sont fréquents. conflits.

Configuration logicielle

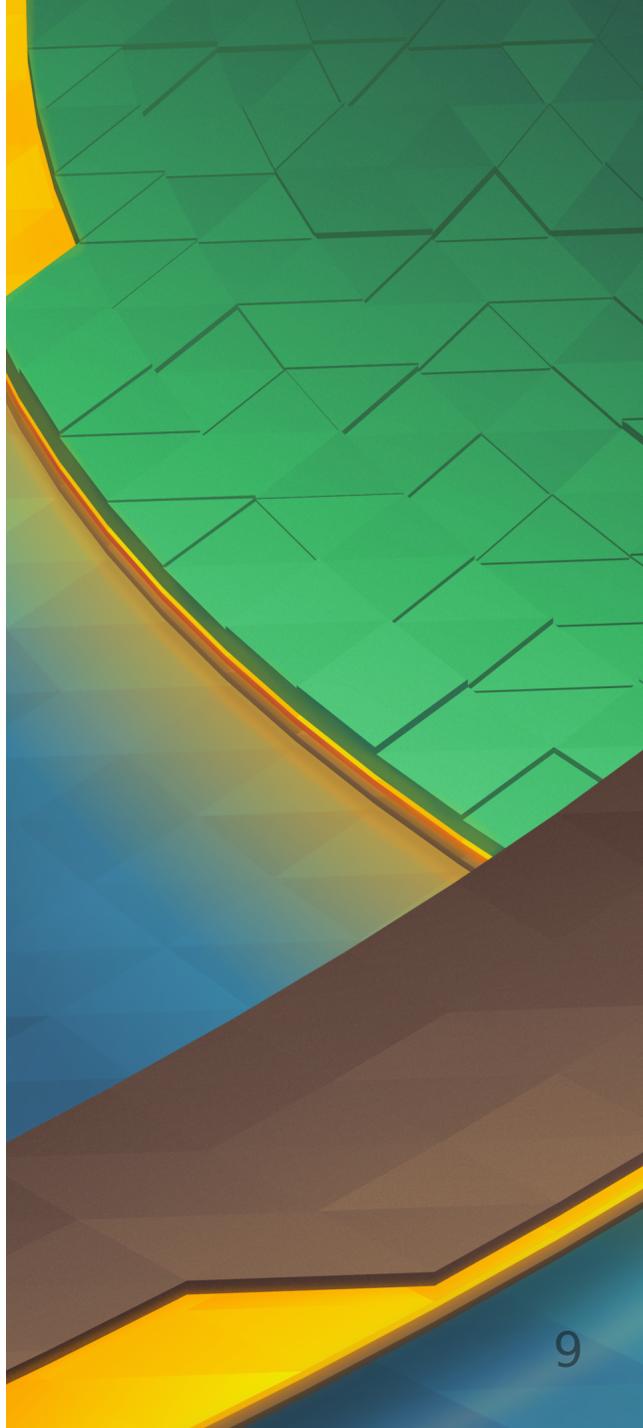
- Installer Git
- Installer Virtualbox
- Installer Vagrant
- Installer Visual Studio Code

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 gêne pas
- Vérifier que le NAT interne de Virtualbox 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 (VirtualBox, 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.

Avant l'installation

Exemple: désactiver VMware

Taper dans un terminal:

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

Et ensuite redémarrer.

 [Vagrant Documentation: Windows, VirtualBox, and Hyper-V](#)

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 Virtualbox

Présentation

VirtualBox est un hyperviseur de machines virtuelles. C'est à dire qu'il gère le bon fonctionnement de machines virtuelles.

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

Installer Virtualbox (2)

Procédure d'installation

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

```
$ echo "deb [arch=amd64 signed-by=/usr/share/keyrings/oracle-virtualbox-2016.gpg] \  
    https://download.virtualbox.org/virtualbox/debian $(lsb_release -cs) contrib" \  
    | sudo tee /etc/apt/sources.list.d/virtualbox.list  
$ wget -O- https://www.virtualbox.org/download/oracle_vbox_2016.asc \  
    | sudo gpg --yes --output /usr/share/keyrings/oracle-virtualbox-2016.gpg --dearmor  
$ sudo apt update  
$ sudo apt-get install virtualbox-7.0
```

Installer Vagrant

Présentation de l'outil

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

Vagrant permet de contrôler VirtualBox pour créer et charger automatiquement des images virtuelles avec différents systèmes et paramètres prédéfinis, de simuler une topologie réseau avec plusieurs VM depuis un fichier de configuration, 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
```

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 VirtualBox

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

```
$ VBoxManage --version
```

3. Le resultat suivant devrait s'afficher:

```
7.0.14r161095
```

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

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
```

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

```
~/minimal
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'debian/buster64'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'debian/buster64' version '10.0.0' is up to date...
==> default: Setting the name of the VM: minimal_default_1586955960808_72100
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
default: Adapter 1: nat
==> default: Forwarding ports...
default: 80 (guest) => 8880 (host) (adapter 1)
default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
default: SSH address: 127.0.0.1:2222
default: SSH username: vagrant
default: SSH auth method: private key
default:
default: Vagrant insecure key detected. Vagrant will automatically replace
default: this with a newly generated keypair for better security.

[... ]

default: Enabling module ssl.
default: Enabling module authz_core.
default: Enabling module authz_host.
default: Enabling module authn_core.
default: Enabling module auth_basic.
default: Enabling module access_compat.
default: Enabling module authn_file.
default: Enabling module authz_user.
default: Enabling module alias.
default: Enabling module dir.
default: Enabling module autoindex.
default: Enabling module env.
default: Enabling module mime.
default: Enabling module negotiation.
default: Enabling module setenvif.
default: Enabling module filter.
default: Enabling module deflate.
default: Enabling module status.
default: Enabling module reqtimeout.
default: Enabling conf charset.
default: Enabling conf localized-error-pages.
default: Enabling conf other-vhosts-access-log.
default: Enabling conf security.
default: Enabling conf serve-cgi-bin.
default: Enabling site 000-default.
default: Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib
default: Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.ser
default: Processing triggers for systemd (241-5) ...
default: Processing triggers for man-db (2.8.5-2) ...
default: Processing triggers for libc-bin (2.28-10) ...

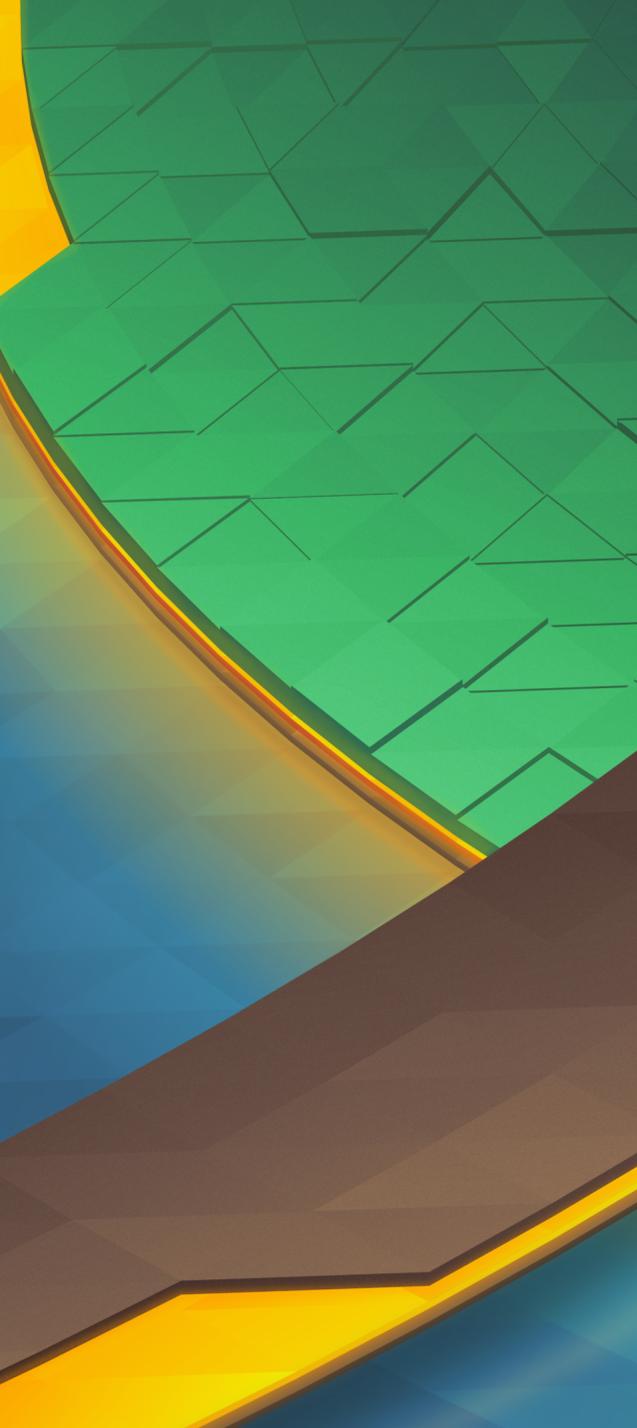
==> default: Machine 'default' has a post `vagrant up` message. This is a message
==> default: from the creator of the Vagrantfile, and not from Vagrant itself:
==> default:
==> default: Vanilla Debian box. See https://app.vagrantup.com/debian for help and bug reports
~/minimal
$
```

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](tel:+33673983956)