

Procédure d'utilisation de Vagrant dans le cadre de projets BTS SIO

Introduction à Vagrant

Vagrant est un outil open-source permettant de créer, configurer et gérer des machines virtuelles de manière reproductible. Il utilise des fichiers de configuration, appelés **Vagrant Files**, pour automatiser le déploiement d'environnements de développement sur des hyperviseurs comme **VirtualBox**, **VMware**, ou **Docker**. Vagrant permet de définir des environnements isolés, facilitant ainsi le travail collaboratif entre développeurs en assurant que tous les membres utilisent la même configuration et les mêmes dépendances. Il offre également la possibilité d'intégrer facilement des outils de provisioning comme **Ansible**, **Chef**, ou **Puppet** pour automatiser la configuration des machines. Un **script Vagrant** est un fichier de configuration utilisé pour automatiser la création et la gestion d'une machine virtuelle. Ce script contient des instructions qui définissent des paramètres pour la machine, comme son image (box), ses ressources (mémoire, CPU) et des actions à réaliser à son démarrage, comme l'installation de logiciels ou la configuration de services.

Par exemple, un script Vagrant peut :

- Choisir l'image du système d'exploitation à utiliser (comme Ubuntu).
- Configurer la mémoire, le nombre de processeurs et d'autres ressources de la machine virtuelle.
- Partager des dossiers entre l'ordinateur hôte et la machine virtuelle pour échanger des fichiers facilement.
- Automatiser des actions comme l'installation de logiciels (par exemple, Nginx ou Docker) dès que la machine virtuelle démarre.

Le but d'un script Vagrant est de rendre le processus de création et de configuration des machines virtuelles plus rapide, facile et reproductible, sans avoir à faire toutes ces étapes manuellement à chaque fois.

Exemples de réalisations avec Vagrant

1. Déploiement de Nginx sur Debian 12

Dans le cadre de nos travaux pratiques, l'un des projets réalisés avec Vagrant consistait à déployer un serveur **Nginx** sur une machine virtuelle exécutant **Debian 12**. Ce projet nous a permis de configurer un environnement de développement dédié pour tester et configurer un serveur web.

Le Script Vagrant :

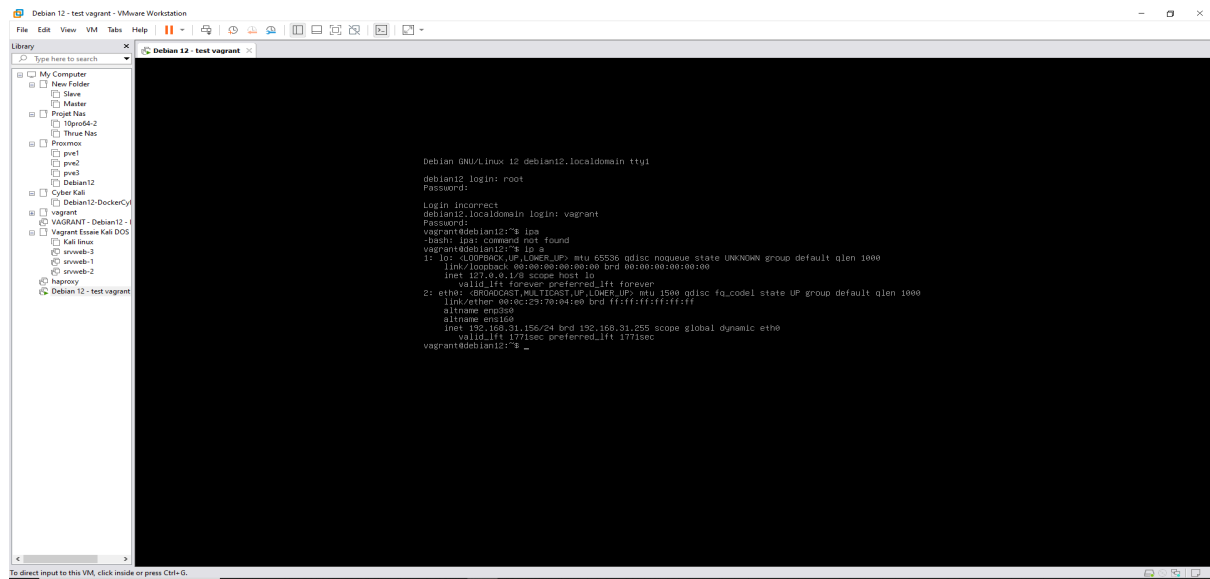
```
1 #Vagrant.configure("2") do |config|
2   # Nom de la box
3   config.vm.box = "generic/debian12"
4   # Version de la box
5   config.vm.box_version = "202112.20.0"
6   # Hostname de la VM
7   config.vm.hostname = "Debian12"
8   # Configuration de l'hyperviseur
9   config.vm.provider "vmware_desktop" do |v|
10    # Nom de la VM dans l'hyperviseur
11    v.name[displayname] = "Debian 12 - test vagrant"
12    # Affichage de l'ui de l'hyperviseur
13    v.gui = true
14    # Définition des ressources CPU RAM pour la VM
15    v.vmx["memory"] = "4096"
16    v.vmx["numvcpus"] = "4"
17  end
18  #Dossier partagé
19  config.vm.synced_folder ".", "/vagrant"
20  #Provisionnement via docker
21  config.vm.provision "docker" do |d|
22    # définition des conteneurs & des arguments
23    d.run "nginx",
24      cmd: "bash -l",
25      args: "-p '80:80'"
26  end
27 end
```

Le bash pour start les VM avec vagrant :

```
C:\Windows\System32\cmd.exe - vagrant up
Microsoft Windows [version 10.0.19045.5487]
(c) Microsoft Corporation. Tous droits réservés.

D:\Kylia Cheroret\Vagrant\vagrant Nginx conteneur>vagrant up
Bringing machine 'default' up with 'vmware_desktop' provider...
==> default: Cloning VMWare VM: 'generic/debian12'. This can take some time...
==> default: Checking if box 'generic/debian12' version '4.3.12' is up to date...
==> default: Verifying vmnet devices are healthy...
==> default: Preparing network adapters...
==> default: Fixed port collision for 22 => 2222. Now on port 2203.
==> default: Starting the VMWare VM...
```

Les Machines Virtuelles :



```
Debian GNU/Linux 12 debian12.localdomain tty1
debian12 login: root
Password:
Login incorrect
debian12.localdomain login: vagrant
Password:
vagrant@debian12:~$ ipa
-bash: ipa: command not found
vagrant@debian12:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:29:70:04:00 brd ff:ff:ff:ff:ff:ff
    altname ens160
    inet 192.168.31.156/24 brd 192.168.31.255 scope global dynamic eth0
        valid_lft 1771sec preferred_lft 1771sec
vagrant@debian12:~$ _
```

2. Laboratoire Cyber - Attaque MITM avec Vagrant

Une autre réalisation importante a été la mise en place d'un laboratoire en cybersécurité pour tester une attaque **Man-in-the-Middle (MITM)**. Ce laboratoire a été configuré à l'aide de Vagrant pour créer plusieurs machines virtuelles interconnectées, où nous avons simulé l'attaque.

Le Script Vagrant :

Ici un script trouvé sur le net qui sert à créer l'environnement

```
D:\Kylan Cheroret\Vagrant\cyber docker\Vagrantfile - Notepad++
Fichier Edition Recherche Affichage Outils Paramètres Outils Macro Exécution Modules d'extension Documents ?
Vagrantfile Vagrantfile Vagrantfile Vagrantfile
1 # -*- mode: ruby -*-
2 # vi: set ft=ruby :
3
4 # All Vagrant configuration is done below. The "2" in Vagrant.configure
5 # configures the configuration version (we support older styles for
6 # backwards compatibility). Please don't change it unless you know what
7 # you're doing.
8 Vagrant.configure("2") do |config|
9   # The most common configuration options are documented and commented below.
10   # For a complete reference, please see the online documentation at
11   # https://docs.vagrantup.com.
12
13   # Every Vagrant development environment requires a box. You can search for
14   # boxes at https://vagrantcloud.com/search.
15   config.vm.box = "generic/debian12"
16   # Disable automatic box update checking. If you disable this, then
17   # boxes will only be checked for updates when the user runs
18   # 'vagrant box outdated'. This is not recommended.
19   config.vm.box_check_update = false
20   config.vm.provider "vmware_desktop" do |v|
21     v.guest = true
22     v.vmx["memory"] = "8192"
23     v.vmx["numvcpus"] = "4"
24   end
25
26   # Create a forwarded port mapping which allows access to a specific port
27   # within the machine from a port on the host machine. In the example below,
28   # accessing "localhost:8080" will access port 80 on the guest machine.
29   # NOTE! This will enable public access to the opened port
30   # config.vm.network "forwarded_port", guest: 80, host: 8080
31
32   $script = <<-SCRIPT
33   echo Configuration de la VM en cours...
34   date > /etc/vagrant_provisioned_at
35   echo -----Mise à jour des dépôts-----
36   sleep 3s
37   sudo apt update
38   sleep 3s
39   echo -----Installation Serveur Web apache2-----
40   sleep 3s
41   sudo apt install -y apache2 docker.io docker-compose git
42   sleep 3s
43   sudo git clone https://forge.apps.education.fr/reseau-certe/bts-sio/labo-kali-docker/lab2.git
44   sleep 3s
45   sudo bash /home/vagrant/lab2/gestion_lab2.sh -c
46   sleep 3s
47   ip a | grep ens33
48
49   SCRIPT
50
51   config.vm.provision "shell", inline: $script
52
53   # Create a forwarded port mapping which allows access to a specific port
54   # within the machine from a port on the host machine and only allow access
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
length: 3.827 lines: 96 Ln:1 Col:1 Pos:1 Windows (CR LF) UTF-8 RNS
Ruby file
```

Le bash pour start les VM avec vagrant :

Ici les VM et l'environnement vagrant se crée grâce à la commande Vagrant UP en bash

```
Selection C:\Windows\System32\cmd.exe - vagrant up
Microsoft Windows [version 10.0.19045.5487]
(c) Microsoft Corporation. Tous droits réservés.

D:\Kylvian Cheroret\Vagrant\Vagrant cyber docker>vagrant up
Building machine 'default' up with 'vmware_desktop' provider...
--> default: Cloning VMware VM: 'generic/debian12'. This can take some time...
--> default: Checking if box 'generic/debian12' version '4.3.12' is up to date...
--> default: Verifying vmnet devices are healthy...
--> default: Preparing network adapters...
--> default: Fixed port collision for 22 -> 2222. Now on port 2204.
--> default: Starting the VMware VM...
--> default: Waiting for the VM to receive an address...
--> default: Forwarding ports...
--> default: -- 22 -> 2204
--> default: Waiting for machine to boot. This may take a few minutes...
--> default: SSH address: 127.0.0.1:2204
--> default: SSH username: vagrant
--> default: SSH auth method: private key
--> default: Vagrant insecure key detected. Vagrant will automatically replace
--> default: this with a newly generated keypair for better security.
--> default: Inserting generated public key within guest...
--> default: Removing insecure key from the guest if it's present...
--> default: Key inserted! Disconnecting and reconnecting using new SSH key...
--> default: Machine booted and ready!
--> default: Configuring network adapters within the VM...
--> default: Running provisioner: shell...
--> default: Running: inline script
--> default: Configuration de la VM en cours...
--> default: ---Mise à jour des dépôts---
--> default: WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
--> default:
--> default: Get:1 http://security.debian.org/debian-security bookworm-security InRelease [48.0 kB]
--> default: Get:2 http://deb.debian.org/debian bookworm InRelease [151 kB]
--> default: Get:3 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
--> default: Get:4 http://security.debian.org/debian-security bookworm-security/main amd64 Packages [245 kB]
--> default: Get:5 http://security.debian.org/debian-security bookworm-security/main Translation-en [147 kB]
--> default: Get:6 http://deb.debian.org/debian bookworm/main Sources [9,496 kB]
--> default: Get:7 http://deb.debian.org/debian bookworm/non-free-firmware Sources [6,436 B]
--> default: Get:8 http://deb.debian.org/debian bookworm/main Sources [9,496 kB]
--> default: Get:9 http://deb.debian.org/debian bookworm/main amd64 Packages [8,792 kB]
--> default: Get:10 http://deb.debian.org/debian bookworm/main Translation-en [6,109 kB]
--> default: Get:11 http://deb.debian.org/debian bookworm/non-free-firmware amd64 Packages [6,248 B]
--> default: Get:12 http://deb.debian.org/debian bookworm/non-free-firmware Translation-en [20.9 kB]
--> default: Get:13 http://deb.debian.org/debian bookworm-updates/main Sources.diff/Index [15.1 kB]
--> default: Ign:13 http://deb.debian.org/debian bookworm-updates/main Sources.diff/Index
--> default: Get:14 http://deb.debian.org/debian bookworm-updates/main amd64 Packages.diff/Index [15.1 kB]
--> default: Ign:14 http://deb.debian.org/debian bookworm-updates/main amd64 Packages.diff/Index
--> default: Get:15 http://deb.debian.org/debian bookworm-updates/main Translation-en.diff/Index [15.1 kB]
--> default: Ign:15 http://deb.debian.org/debian bookworm-updates/main Translation-en.diff/Index
--> default: Get:16 http://deb.debian.org/debian bookworm-updates/non-free-firmware Sources [2,076 B]
--> default: Get:17 http://deb.debian.org/debian bookworm-updates/non-free-firmware amd64 Packages [616 B]
--> default: Get:18 http://deb.debian.org/debian bookworm-updates/non-free-firmware Translation-en [384 B]
--> default: Get:19 http://deb.debian.org/debian bookworm-updates/main Sources [16.2 kB]
--> default: Get:20 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [13.5 kB]
--> default: Get:21 http://deb.debian.org/debian bookworm-updates/main Translation-en [10.0 kB]
--> default: Fetched 25.3 MB in 6s (4,463 kB/s)
--> default: Reading package lists...
--> default: Building dependency tree...
--> default: Reading state information...
--> default: 79 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Les Machines Virtuelles utilisées :

Conclusion

L'utilisation de **Vagrant** a grandement facilité la gestion de machines virtuelles dans un environnement contrôlé, permettant de simuler différents scénarios de développement et de cybersécurité. Grâce à la configuration automatisée des VMs et à la possibilité d'intégrer des outils comme **Docker**, **Vagrant** s'est révélé être un outil indispensable pour la réalisation de projets en BTS SIO.