Workshop 1

Creación de una máquina virtual con Vagrant

Pasos a seguir

1. Descarga e instalación de VirtualBox

1. Descarga VirtualBox desde la página oficial.
2. Sigue las instrucciones de instalación para tu sistema operativo.

2. Descarga e instalación de Vagrant

1. Descarga Vagrant desde la página oficial.
2. Sigue las instrucciones de instalación para tu sistema operativo.

3. Creando el entorno de trabajo

1. Crear una carpeta llamada `Workshop 1`:

```
mkdir Workshop1
cd Workshop1
```

2. Dentro de esta carpeta, crea un archivo README (este archivo que estás leyendo).

3. **Ejemplo de estructura de carpetas**:

e\Workshops\Workshop I

4. Creando el archivo de configuración de la máquina virtual

1. Inicializa un nuevo proyecto de Vagrant:

```
vagrant init debian/bookworm64
```

2. Esto creará un archivo `Vagrantfile` en tu directorio de trabajo.

3. **Ejemplo de archivo Vagrantfile inicial**:

```
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
```

```ruby
 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 = "debian/bookworm64"
16
17      # Disable automatic box update checking. If you disable this, then
18      # boxes will only be checked for updates when the user runs
19      # `vagrant box outdated`. This is not recommended.
20      # config.vm.box_check_update = false
21
22      # Create a forwarded port mapping which allows access to a specific port
23      # within the machine from a port on the host machine. In the example below,
24      # accessing "localhost:8080" will access port 80 on the guest machine.
25      # NOTE: This will enable public access to the opened port
26      # config.vm.network "forwarded_port", guest: 80, host: 8080
27
28      # Create a forwarded port mapping which allows access to a specific port
29      # within the machine from a port on the host machine and only allow access
30      # via 127.0.0.1 to disable public access
31      # config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"
32
33      # Create a private network, which allows host-only access to the machine
34      # using a specific IP.
35      # config.vm.network "private_network", ip: "192.168.33.10"
36
37      # Create a public network, which generally matched to bridged network.
38      # Bridged networks make the machine appear as another physical device on
39      # your network.
40      # config.vm.network "public_network"
41
42      # Share an additional folder to the guest VM. The first argument is
43      # the path on the host to the actual folder. The second argument is
44      # the path on the guest to mount the folder. And the optional third
45      # argument is a set of non-required options.
46      # config.vm.synced_folder "../data", "/vagrant_data"
47
48      # Disable the default share of the current code directory. Doing this
49      # provides improved isolation between the vagrant box and your host
50      # by making sure your Vagrantfile isn't accessible to the vagrant box.
51      # If you use this you may want to enable additional shared subfolders as
52      # shown above.
53      # config.vm.synced_folder ".", "/vagrant", disabled: true
54
55      # Provider-specific configuration so you can fine-tune various
56      # backing providers for Vagrant. These expose provider-specific options.
57      # Example for VirtualBox:
58      #
59      # config.vm.provider "virtualbox" do |vb|
60      #   # Display the VirtualBox GUI when booting the machine
61      #   vb.gui = true
62      #
63      #   # Customize the amount of memory on the VM:
64      #   vb.memory = "1024"
65      # end
66      #
67      # View the documentation for the provider you are using for more
68      # information on available options.
69
70      # Enable provisioning with a shell script. Additional provisioners such as
71      # Ansible, Chef, Docker, Puppet and Salt are also available. Please see the
72      # documentation for more information about their specific syntax and use.
73      # config.vm.provision "shell", inline: <<-SHELL
74      #   apt-get update
75      #   apt-get install -y apache2
76      # SHELL
77    end
78
```

5. Buscar la IP de red en VirtualBox

1. Abre VirtualBox y busca la IP que tiene por defecto para conectarse a internet.

2. **Ejemplo de configuración de red en VirtualBox**:
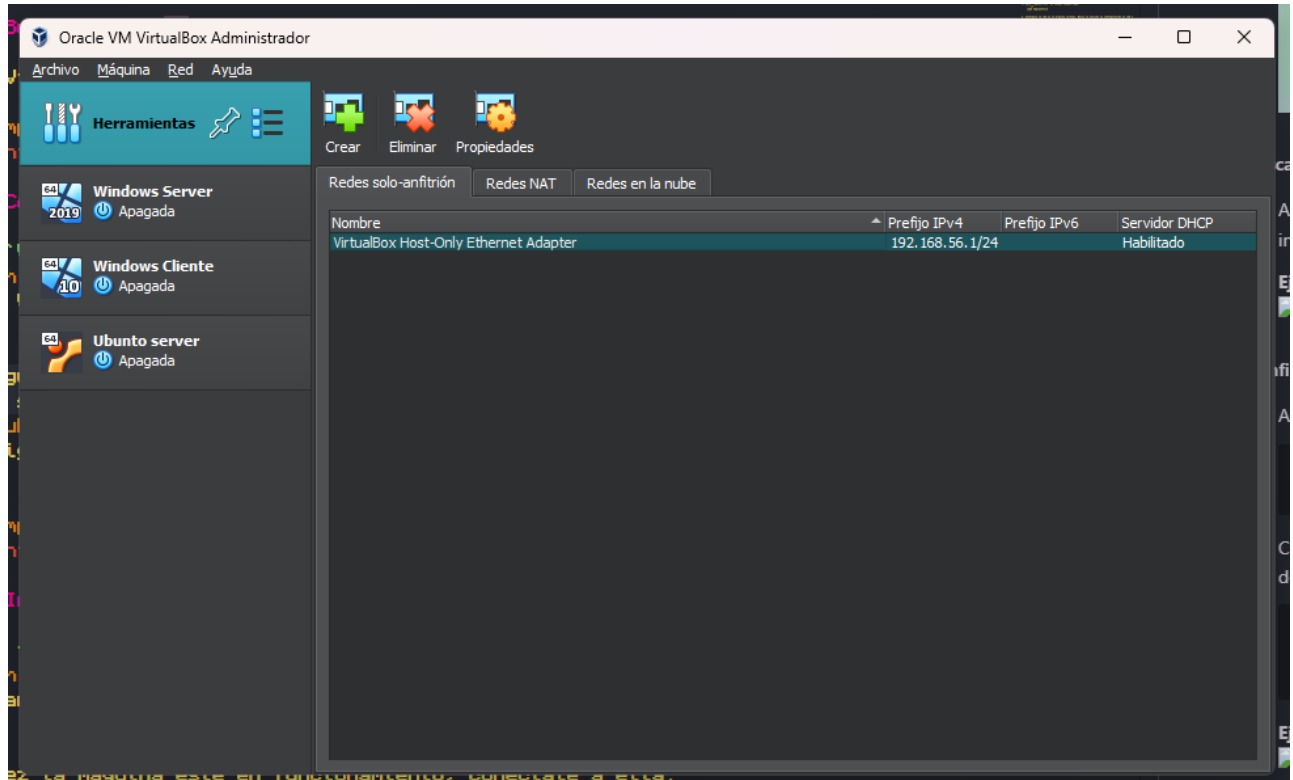
6. Configurando la máquina virtual

1. Abre Vagrantfile en Visual Studio Code:

```
code Vagrantfile
```

2. Configura la red de la máquina virtual. Busca la sección de configuración de red y añade la siguiente línea:

```
config.vm.network "private_network", ip: "192.168.56.1"
```

3. **Ejemplo de configuración de red en Vagrantfile**:

```
1   # via 127.0.0.1 to disable public access
2   # config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"
3
4   # Create a private network, which allows host-only access to the machine
5   # using a specific IP.
6   config.vm.network "private_network", ip: "192.168.56.1"
7
8   # Create a public network, which generally matched to bridged network.
9   # Bridged networks make the machine appear as another physical device on
10  # your network.
11  # config.vm.network "public_network"
```

7. Iniciando la máquina virtual

1. En tu terminal, dentro del directorio `Workshop1`, ejecuta:

```
vagrant up
```

2. Una vez la máquina esté en funcionamiento, conéctate a ella:

```
vagrant ssh
```

3. **Ejemplo de conexión exitosa a la máquina virtual**:

```
Admin@FierceSpectrum MINGW64 ~/Documents/Cursos/Software Libre/Workshops/Worksho
p I/maquinas
$ vagrant ssh
Linux bookworm 6.1.0-18-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.76-1 (2024-02-01) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
vagrant@bookworm:~$
```

8. Instalando paquetes básicos

1. Actualiza la lista de paquetes y el sistema:

```
sudo apt-get update
sudo apt-get upgrade
```

2. Instala algunos paquetes básicos. Por ejemplo:

```
sudo apt-get install -y git
```

3. Verifica que los paquetes se han instalado correctamente:

```
git --version
```

4. **Ejemplo de instalación y verificación de paquetes**:

```
vagrant@bookworm:~$ sudo apt-get install -y git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl patch
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn ed
  diffutils-doc
The following NEW packages will be installed:
  git git-man liberror-perl patch
0 upgraded, 4 newly installed, 0 to remove and 1 not upgraded.
Need to get 9377 kB of archives.
After this operation, 48.0 MB of additional disk space will be used.
Get:1 https://deb.debian.org/debian bookworm/main amd64 liberror-perl all 0.17029-2 [29.0 kB]
Get:2 https://deb.debian.org/debian bookworm/main amd64 git-man all 1:2.39.2-1.1 [2049 kB]
Get:3 https://deb.debian.org/debian bookworm/main amd64 git amd64 1:2.39.2-1.1 [7171 kB]
Get:4 https://deb.debian.org/debian bookworm/main amd64 patch amd64 2.7.6-7 [128 kB]
Fetched 9377 kB in 1s (6564 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 25485 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-2_all.deb ...
Unpacking liberror-perl (0.17029-2) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.39.2-1.1_all.deb ...
Unpacking git-man (1:2.39.2-1.1) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.39.2-1.1_amd64.deb ...
Unpacking git (1:2.39.2-1.1) ...
Selecting previously unselected package patch.
Preparing to unpack .../patch_2.7.6-7_amd64.deb ...
Unpacking patch (2.7.6-7) ...
Setting up liberror-perl (0.17029-2) ...
Setting up patch (2.7.6-7) ...
Setting up git-man (1:2.39.2-1.1) ...
Setting up git (1:2.39.2-1.1) ...
Processing triggers for man-db (2.11.2-2) ...
vagrant@bookworm:~$ git --version
git version 2.39.2
vagrant@bookworm:~$
```

9. Finalizando

1. Para salir de la máquina virtual:

```
exit
```

2. Para detener la máquina virtual:

```
vagrant halt
```

3. **Ejemplo de comandos para finalizar la máquina virtual**:

```
Admin@FierceSpectrum MINGW64 ~/Documents/Cursos/Software Libre/Workshops/Workshop I/maquinas
$ vagrant halt
==> default: Attempting graceful shutdown of VM...
```

¡Y eso es todo! Ahora has configurado y trabajado con una máquina virtual Debian usando Vagrant.

[My GitHub](#)