Vagrant essentials

Build portable environments

Vagrant network configuration

Introduction

- I hope you’ve seen so far the power of Vagrant and Vagrant-automation: in a few minutes you were able to setup, configure, install necessary prerequisites, and actually deploy a bare-bones PHP web application.
- You can pick the OS of your choice (Windows, Linux, MAC, or Solaris), work from the comfort of your own desktop tools, browser(s) and editor to work with the application through the shared directory.
- But Vagrant is capable of doing more than that. Till this point, we've been accessing the web application internally though port 8080 on the host. This is just one form on networking that Vagrant supports. Let’s have a look at Vagrant networking modes:
  - Forwarded ports
  - Private network
  - Bridged networking

Forwarded ports

- This is the most basic way of accessing network services on the guest system. You instruct Vagrant (through the Vagrantfile) to forward traffic arriving at a specific port on the host to another port on the guest. This line should be now familiar to you:
  ```
  config.vm.network "forwarded_port", guest: 80, host: 8080
  ```
- This allows your web application to be accessible from your own machine via `localhost:8080`. It also allows anyone in your network to access this application if he/she knows the IP address of your machine (this might pose a potential security threat).
- You should use this approach when:
  - You have very few ports to be opened (remember each port needs its own rule).
  - If you don't want (or care) to use ports that are less than 1024. Ports less than 1024 are reserved for system processes and are not allowed to be used by non administrative users (like Vagrant).
- By default, Vagrant will detect if the forwarded port is already used (perhaps by another guest) and will attempt to use another free port.
- Vagrant assumes that you need to forward ports for the TCP protocol. If you intend to forward a port that uses UDP, you must declare that in the Vagrant file. For example:
  ```
  config.vm.network "forwarded_port", guest: 123, host: 8080, protocol: "udp"
  ```

Private network

- As the name suggests, this option makes the guest machine accessible only from the host on which it is operating.
- Other guests on the same host can access each other if they are configured on this network type, as long as they are on the same subnet. However, none of them can be reached from outside the host.
- You can communicate with the guest machine directly through it's IP address, accessing any ports directly on it rather than using forwarded ports on localhost. For example. our web application can be reached through http://172.16.0.10/index.php (172.16.0.10 refers to the guest machine's IP address).
- Guests in this mode can also communicate with services running on the host. In forwarded-ports mode, this type of access is not available.
- Use this option if:
  - You are more concerned about exposing your applications and services to your local network, perhaps for security or privacy.
  - You have a large number of ports that need to be accessed on the guest and you don't care (or want) external access to those ports outside your host machine.
  - The project you are working on do not need collaboration from your coworkers on the local network so that needn't access it.
- Note that when using the private network mode, you can either assign a static IP to the VM (you are in charge of not assigning it an IP that is already taken on the same network), or use a DHCP server to automatically assign an available IP address. Both options can be added to the Vagrantfile as follows:

```
config.vm.network "private_network", type: "dhcp"
config.vm.network "private_network", ip: "172.16.0.10"
```

Public network

- This option makes your VM act as if it was a totally separate device on the network. It has its own public IP address. It can be accessed from the local network as well as from external networks and the Internet (if routers were configured to allow this).
- You can communicate with the VM through its IP address, which will be in the same range as the host's IP address.
- Use this mode if:
  - You need the machine to be publicly accessible not only from the local network, but also from external networks and/or the Internet.
  - You want any application to be published through its native port(s) rather than forwarded ports.
  - You are not concerned about machine network isolation (although this can be achieved using network configuration options like firewalls, vLans…, etc.)
- Like the private network mode, you have the option to set the IP address of your machine manually (static IP) or automatically (through DHCP). This can be done in the Vagrantfile by adding one of the following lines:

```
config.vm.network "public_network" #To use DHCP
config.vm.network "public_network", ip: "192.168.0.111" # To use a static IP
```

- The public network works by “bridging” the virtual network interface on the VM the physical network interface on the host machine.

- If you have more than interface on the machine (almost all modern computers now have a wired and a wireless network interface card), you will have to specify which interface you are going to bridge the VM virtual interface to. If you don’t, Vagrant will ask you to choose one when it boots.

- To assign a network interface in the public network mode, you modify the Vagrantfile line to be as follows:
  `config.vm.network "public_network", bridge: "en1: Wi-Fi (AirPort)"`

- Notice the name of the network card in the above line much match exactly the name displayed for your network interface. Otherwise, Vagrant will ask you to choose one of the available interfaces on boot.

- If you use DHCP to assign an IP to the machine, you will have to login to the VM, issue `ifconfig -a` to view the available interfaces and their assigned IP addresses. Vagrant always assigns the bridged network to the second interface.

Using multiple network options

- Let’s say that you have an application running on your host and on some other computers on your network that is configured to communicate with a web application running on a Vagrant VM on your machine on the forwarded port 8080 mapped to the application port 80.

- Now because you have a lot of other applications running on the VM, each with its own set of ports, forwarded ports are no longer an efficient option so you want to change the configuration to be “public network”

- However, the clients installed on your machine and on other devices on the network are already configured to communicate with the machine on port 8080. Changing this is a lengthy operation. The end result is: you want to keep the forwarded port as is and at the same time use public network.

- Then soon your network requirements get even more complex: you need your VM to be accessible from your local Ethernet network as well on the WiFi one; so that mobile devices can be used to communicate with your application.

- Fortunately, Vagrant has got your back. You can instruct Vagrant to operate on forwarded ports, and also bridge interfaces on any number of physical network interfaces. This is limited only by the physical number of network cards available and the maximum number of interfacees the virtualization platform supports. VirtualBox currently supports up to eight interfaces.

- Let’s modify the Vagrantfile to achieve the desired result:
```
config.vm.network "forwarded_port", guest: 80, host: 8080
config.vm.network "public_network", bridge: "en1: Wi-Fi (AirPort)"
config.vm.network "public_network", bridge: "en0: Ethernet"
```