MODUL PEMBELAJARAN
TEKNOLOGI KOMPUTASI AWAN
VAGRANT

Modul 1 Vagrant

Dosen pengampu :
Henning Titi Ciptaningtyas, S.Kom, M.Kom.

Disusun oleh :
Banabil Fawazaim Muhammad
Haffif Rasya Fauzi
Sharira Saniane

Daftar isi

Dasar Teori

1. **Virtualisasi**
   Virtualisasi adalah suatu teknologi yang memungkinkan pengoperasian beberapa sistem operasi secara bersamaan pada satu komputer fisik. Dengan menggunakan virtualisasi, setiap sistem operasi yang berjalan dalam lingkungan virtual terisolasi sehingga jika terjadi masalah pada salah satu sistem operasi, hal tersebut tidak akan mempengaruhi sistem operasi lainnya maupun komputer fisik/hostnya. Contoh teknologi virtualisasi yang umum digunakan meliputi VirtualBox, VMware, Vagrant, dan Docker.
2. **Provisioning**
   Proses provisioning dalam komputasi awan melibatkan penyediaan aplikasi atau layanan dengan menggunakan virtualisasi. Proses ini meliputi pembuatan instance virtual, konfigurasi sumber daya, instalasi sistem operasi, pemasangan aplikasi atau layanan, serta konfigurasi yang diperlukan. Penggunaan virtualisasi dalam proses provisioning sangat penting dalam komputasi awan karena harus memastikan bahwa tidak ada waktu henti atau layanan yang mati dalam jangka waktu yang lama. Dengan menggunakan virtualisasi, proses provisioning dapat dilakukan dengan cepat dan mengurangi waktu henti sehingga layanan cloud dapat dijalankan tanpa terganggu.
3. **Vagrant**
   Vagrant adalah sebuah framework yang digunakan untuk mengelola lingkungan virtualisasi. Dengan Vagrant, dapat dibuat lingkungan virtual yang terisolasi. Banyak pengembang yang menggunakan Vagrant dalam tim pengembangan mereka. Keberadaan Vagrant memastikan keseragaman dan konsistensi lingkungan pengembangan di antara para pengembang, sehingga menghindari masalah seperti "Ini berfungsi di sistem saya". Selain itu, Vagrant juga digunakan dalam proses penyediaan layanan. Vagrant mendukung berbagai provider virtualisasi seperti VirtualBox, VMware, AWS, dan Docker.

Instalasi

1. **Instalasi VirtualBox**
   - Debian based: apt-get install virtualbox.
   - MacOS: https://download.virtualbox.org/virtualbox/7.0.10/VirtualBox-7.0.10a-158379-OSX.dmg.
   - Windows: https://download.virtualbox.org/virtualbox/7.0.10/VirtualBox-7.0.10-158379-Win.exe.
2. **Instalasi Vagrant**
   - Debian based: apt-get install vagrant.
   - MacOs: brew install hashicorp/tap/hashicorp-vagrant atau download pada link berikut: https://developer.hashicorp.com/vagrant/downloads.

Membuat Virtualisasi

Setelah berhasil menginstall vagrant, selanjutnya kita akan mencoba membuat virtualisasi baru menggunakan provider virtualbox. Langkah-langkah pembuatan virtualisasi baru dijelaskan sebagai berikut:

1. Buatlah folder baru untuk meletakkan konfigurasi.

mkdir vagrant-example

2. Masuk ke dalam folder yang telah dibuat.

cd vagrant-example

3. Melakukan Inisialisasi pada projek vagrant.

vagrant init

Setelah menjalankan perintah diatas akan dibuat file baru bernama Vagrantfile.

4. Isi dari Vagrantfile yaitu sebagai berikut:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
  Vagrant.configure("2") do |config|
# The most common configuration options are documented and commented below.
# For a complete reference, please see the online documentation at
# https://docs.vagrantup.com.

# Every Vagrant development environment requires a box. You can search for
# boxes at https://vagrantcloud.com/search.
  config.vm.box = "base"

# Disable automatic box update checking. If you disable this, then
# boxes will only be checked for updates when the user runs
# `vagrant box outdated`. This is not recommended.
# config.vm.box_check_update = false

# Create a forwarded port mapping which allows access to a specific port
# within the machine from a port on the host machine. In the example below,
# accessing "localhost:8080" will access port 80 on the guest machine.
# NOTE: This will enable public access to the opened port
# config.vm.network "forwarded_port", guest: 80, host: 8080

# Create a forwarded port mapping which allows access to a specific port
# within the machine from a port on the host machine and only allow access
# via 127.0.0.1 to disable public access
# config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"

# Create a private network, which allows host-only access to the machine
# using a specific IP.
# config.vm.network "private_network", ip: "192.168.33.10"

# Create a public network, which generally matched to bridged network.
# Bridged networks make the machine appear as another physical device on
# your network.
```

```
# config.vm.network "public_network"

# Share an additional folder to the guest VM. The first argument is
# the path on the host to the actual folder. The second argument is
# the path on the guest to mount the folder. And the optional third
# argument is a set of non-required options.
# config.vm.synced_folder "../data", "/vagrant_data"

# Provider-specific configuration so you can fine-tune various
# backing providers for Vagrant. These expose provider-specific options.
# Example for VirtualBox:
#
# config.vm.provider "virtualbox" do |vb|
#   # Display the VirtualBox GUI when booting the machine
#   vb.gui = true
#
#   # Customize the amount of memory on the VM:
#   vb.memory = "1024"
# end
#
# View the documentation for the provider you are using for more
# information on available options.

# Enable provisioning with a shell script. Additional provisioners such as
# Puppet, Chef, Ansible, Salt, and Docker are also available. Please see the
# documentation for more information about their specific syntax and use.
# config.vm.provision "shell", inline: <<-SHELL
#   apt-get update
#   apt-get install -y apache2
# SHELL
  end
```

5. Pada contoh kasus ini kita ingin menggunakan os Ubuntu (20.04) Focal 64 bit. Maka dari itu kita perlu download Vagrant Box terlebih dahulu. Dengan cara:

   vagrant box add ubuntu/focal64

6. Kemudian jika box mendukung lebih dari satu provider akan ditanyakan provider yang akan digunakan. Pilih provider virtualbox.
7. Setting box pada Vagrant file dengan cara ganti vm.box yang awalnya base menjadi ubuntu/focal64.

   config.vm.box = "base"

   menjadi seperti berikut:

   config.vm.box = "ubuntu/focal64"

Konfigurasi Resource Virtual Machine

Layaknya komputer fisik, virtual machine terdapat memory dan core cpu. Pada Vagrant, resource core dan memory virtual machine diatur pada Vagrantfile. Untuk mengatur resource virtual machine dapat dilakukan dengan langkah berikut:

1. Uncomment konfigurasi:

```
config.vm.provider "virtualbox" do |vb|
  # Display the VirtualBox GUI when booting the machine
  # vb.gui = true

  # Customize the amount of memory on the VM:
  vb.memory = "1024"
end
```

2. Untuk menentukan resource memory yang diperlukan ganti value dari **vb.memory**.
3. Untuk mengatur core cpu, tambahkan baris sebelum end.

```
config.vm.provider "virtualbox" do |vb|
  # Display the VirtualBox GUI when booting the machine
  # vb.gui = true

  # Customize the amount of memory on the VM:
  vb.memory = "1024"
     vb.cpus = 2
end
```

4. Untuk storage tidak dapat diatur oleh vagrant. Mungkin bisa diatur melalui Hypervisor yang digunakan.

Cara Bermain

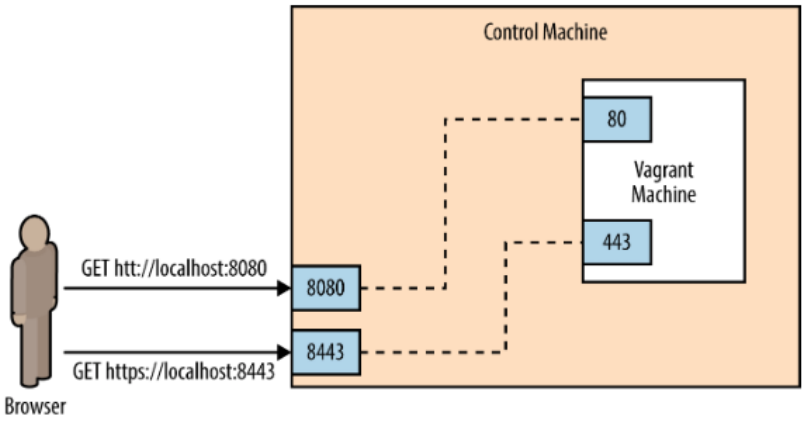
1. Menjalankan vagrant : **vagrant up**.
2. Untuk masuk ke virtual machine : **vagrant ssh**.
3. Untuk mematikan virtual machine : **vagrant halt**.
4. Untuk menghapus virtual machine : **vagrant destroy**.
5. Untuk merestart virtual machine dan akan memuat ulang konfigurasi Vagrantfile : **vagrant reload**.
6. Untuk menjalankan provisioning : **vagrant provision**.

Konfigurasi Internet pada Virtual Machine

Pada vagrant terdapat 3 jenis konfigurasi agar virtual machine dapat diakses dari host maupun dari luar host.

1. Port Forwarders

   Dalam lingkungan produksi, untuk mengakses aplikasi yang berjalan di dalam virtualisasi, perlu dilakukan mekanisme port forwarders. Mekanisme ini berfungsi untuk meneruskan akses dari port komputer host ke port khusus di dalam virtualisasi. Untuk memahami konsep ini secara visual, berikut adalah ilustrasi gambar yang menjelaskan penggunaan port forwarders dalam lingkungan tersebut.



Klien mengakses port 8080/8443 pada komputer host, kemudian akan diteruskan menuju port 80/443 pada virtualisasi vagrant. Untuk mengaktifkan port forwarders, terdapat step-step sebagai berikut:

- Uncomment baris agar port 80 VM dapat diakses melalui port 8080 host

config.vm.network "forwarded_port", guest: 80, host: 8080, host_ip: "127.0.0.1"

Dan hapus bagian:

host_ip: "127.0.0.1

Agar bisa diakses dari luar host. Sehingga menjadi seperti berikut:

config.vm.network "forwarded_port", guest: 80, host: 8080

- Tambahkan baris

config.vm.network "forwarded_port", guest: 443, host: 8443

Agar port 443 pada virtual machine dapat diakses melalui port 8443

2. Static Local IP

Ketika banyak virtualisasi yang dijalankan, kita membutuhkan alamat IP lokal agar antar virtualisasi dapat saling berkomunikasi. IP lokal hanya dapat diakses oleh komputer host dan virtualisasi pada komputer host yang sama. Untuk menentukan alamat IP pada virtualisasi lakukan step-step berikut:

- Uncomment baris :

config.vm.network "private_network", ip: "192.168.33.10"

Virtual machine dapat kita akses melalui ip 192.168.33.10

Untuk pemilihan IP, pilihlah ip dengan angka belakang 2-254, karena ip dengan angka belakang 1 sudah digunakan oleh komputer host sebagai router antar virtualisasi. Alamat IP lokal antar virtualisasi tidak perlu harus dalam subnet yang sama. Alamat IP pada subnet yang berbeda tetap bisa berkomunikasi, karena mekanisme routing telah diatur oleh vagrant.

3. Bridged IP

Pada kondisi tertentu dibutuhkan virtualisasi yang dapat diakses dari luar, contohnya pada layanan cloud vps(virtual private server). Untuk membuat virtualisasi dapat diakses dari luar, dibutuhkan mekanisme yang disebut bridging . Mekanisme bridging ini telah ditangani oleh vagrant. Untuk mengaktifkan fungsi bridged ikuti langkah-langkah berikut:

- Uncomment baris :

config.vm.network "public_network"

- Untuk konfigurasi IP static pada virtualisasi gunakan konfigurasi berikut:

config.vm.network "public_network", ip: "10.151.36.225"

Ip tergantung subnet dari host

Sinkronisasi Folder

Ketika kita ingin mengembangkan aplikasi menggunakan editor favorit seperti Sublime Text, NetBeans, atau lainnya, namun ingin agar kode aplikasi tersebut berada di dalam virtualisasi, terdapat beberapa cara untuk mencapai hal tersebut. Salah satunya adalah dengan menggunakan fitur sinkronisasi pada Vagrant. Sinkronisasi folder memungkinkan folder di dalam virtualisasi dapat diakses melalui komputer host. Fitur ini memastikan bahwa perubahan yang terjadi pada kode melalui komputer host atau dalam virtualisasi akan secara otomatis terjadi sinkronisasi, sehingga data dalam folder tetap sama.

Untuk mengaktifkan sinkronisasi folder di Vagrant, ikuti langkah-langkah berikut:

1. Buka file Vagrantfile ubah baris berikut.
   # config.vm.synced_folder "../data", "/vagrant_data"
2. Menjadi seperti berikut:
   config.vm.synced_folder "src/", "/var/www"
3. Simpan file Vagrantfile. src/ adalah folder pada komputer host, sedangkan /var/www adalah folder pada virtual machine.
4. Buat folder src di dalam folder projek vagrant example kemudian tambahkan file index.html

```
mkdir src
echo "hello world" >> src/index.html
```

5. Jalankan virtualisasi
   i. **vagrant up**
6. Masuk ke dalam virtualisasi
   ii. **vagrant ssh**
7. Lakukan perubahan pada file src/index.html di komputer host, kemudian cek file index.html yang berada pada folder /var/www di virtual machine. Kedua file akan berisi data yang sama, karena telah tersinkronisasi.

Provisioning Aplikasi Pada Virtual Machine

Kita dapat dengan mudah melakukan instalasi dan konfigurasi aplikasi yang dibutuhkan di komputer virtual dengan menggunakan metode yang disebut sebagai provisioning. Dalam Vagrant, provisioning dapat dilakukan dengan mudah melalui pembuatan skrip menggunakan bahasa bash. Berikut ini adalah langkah-langkah yang dapat diikuti untuk melakukan provisioning:

1. Menggunakan File Bootstrap
   - Buat bash script dengan nama bootsrap.sh pada folder yang sama dengan vagrant file.
   - Untuk menginstall apache tuliskan baris berikut pada file bootstrap.sh.

```
#!/usr/bin/env bash
apt-get update
apt-get install -y apache2
```

   - Pada file Vagrantfile diatas end terakhir, tambahkan baris
     config.vm.provision :shell, path: "bootstrap.sh".

Sehingga menjadi seperti berikut:

```
config.vm.provision "shell", path: "bootstrap.sh"
end
```

- Simpan file Vagrantfile kemudian nyalakan virtualisasi.
  vagrant up
- Jika virtualisasi sudah dibuat dan sedang menyala maka jalankan fungsi reload dengan menambahkan flag --provision untuk memaksa vagrant restart virtualisasi dan menjalankan script provisioning ketika mesin virtual sedang aktif.
  vagrant reload --provision
  atau tanpa melakukan restart vagrant:
  vagrant provision
- Cek apakah provisioning berhasil dengan masuk ke dalam virtualisasi menggunakan ssh.
  vagrant ssh
- Cek apakah apache telah berhasil terinstall
  service apache2 status

2. Atau dengan Menambahkan command pada Vagrantfile
   Uncomment baris:

```
config.vm.provision "shell", inline: <<-SHELL
apt-get update
apt-get install -y apache2
SHELL
```

Proses provisioning dapat juga menggunakan configuration management seperti ansible, chef, atau puppet. Proses provisioning otomatis menjalankan menggunakan superuser. Jika ingin mematikan super user dapat menambahkan opsi: **privileged:false**

Contoh:

config.vm.provision "shell", path: "bootstrap.sh", privileged:false

Demo Tutorial

Video tutorial

Soal Latihan

1. Buat vagrant virtualbox dan install nginx. Nginx dapat diakses pada port 9000 host.
2. Buat vagrant virtualbox dan lakukan provisioning install Flask Web Framework

Referensi

1. Modul Komputasi Awan 2017 oleh Thiar Hasbiya S.Kom, M.Kom
2. Modul Awan: https://github.com/fathoniadi/cloud-2018/tree/master