

casos de uso

3 nodos vagrant + docker

la idea es desplegar un cluster swarm

```
Vagrant.configure("2") do |config|
  config.vm.define "nodo1" do |nodo1|
    nodo1.vm.box = "debian/jessie64"
    nodo1.vm.hostname = "nodo1"
#    nodo1.vm.network "public_network", bridge: "eth0"
    nodo1.vm.network "private_network", ip: "10.0.100.101"
    nodo1.vm.provision "shell", path: "install-docker.sh"
    nodo1.vm.provider "virtualbox" do |vb|
      vb.name = "debian-swarm-nodo1"
      vb.memory = 1024
      vb.cpus = 1
      vb.linked_clone = true
    end
  end
  config.vm.define "nodo2" do |nodo2|
    nodo2.vm.box = "debian/jessie64"
    nodo2.vm.hostname = "nodo2"
    nodo2.vm.network "private_network", ip: "10.0.100.102"
    nodo2.vm.provider "virtualbox" do |vb|
      vb.name = "debian-swarm-nodo2"
      vb.memory = 1024
      vb.cpus = 1
      vb.linked_clone = true
    end
  end
  config.vm.define "nodo3" do |nodo3|
    nodo3.vm.box = "debian/jessie64"
    nodo3.vm.hostname = "nodo3"
    nodo3.vm.network "private_network", ip: "10.0.100.103"
    nodo3.vm.provider "virtualbox" do |vb|
      vb.name = "debian-swarm-nodo3"
      vb.memory = 1024
      vb.cpus = 1
      vb.linked_clone = true
    end
  end
end
```

```
#!/bin/bash
sudo apt-get remove -y docker docker-engine docker.io containerd runc
sudo apt-get update
sudo apt-get install -y \
  apt-transport-https \
  ca-certificates \
```

```
curl \
gnupg2 \
software-properties-common \
vim
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/debian \
    $(lsb_release -cs) \
    stable"
sudo apt-get update
sudo apt-get install -y docker-ce
sudo usermod -aG docker vagrant
#newgrp docker
#docker run hello-world
```

vagrant + docker

```
Vagrant.configure(2) do |config|
  config.vm.box = "debian/jessie64"
  config.vm.hostname = "debivan"
  config.vm.network "forwarded_port", guest: 80, host: 1080
  config.vm.network "public_network", bridge: "enp0s31f6"
  config.vm.synced_folder "./data", "/vagrant"
#  config.vm.provision "shell", path: "install.sh"
  config.vm.provision "docker" do |d|
    d.pull_images "nginx"
    d.run "nginx",
      args: "-v '/vagrant:/usr/share/nginx/html' -p '80:80'"
  end
  config.vm.provider "virtualbox" do |vb|
    vb.name = "debivan-vm"
    vb.memory = "512"
    vb.cpus = 2
    vb.linked_clone = true
  end
end
```

vagrant + docker

```
Vagrant.configure(2) do |config|
  config.vm.box = "debian/jessie64"
  config.vm.hostname = "debivan"
  config.vm.network "forwarded_port", guest: 80, host: 1080
  config.vm.network "public_network", bridge: "enp0s31f6"
  config.vm.synced_folder "./data", "/vagrant"
#  config.vm.provision "shell", path: "install.sh"
  config.vm.provision "docker" do |d|
    d.pull_images "nginx"
    d.run "nginx",
```

```
    args: "-v '/vagrant:/usr/share/nginx/html' -p '80:80'"
end
config.vm.provider "virtualbox" do |vb|
  vb.name = "debivan-vm"
  vb.memory = "512"
  vb.cpus = 2
  vb.linked_clone = true
end
end
```

configuraciones simples

```
Vagrant.configure(2) do |config|
  config.vm.box = "debian/jessie64"
  config.vm.hostname = "prueba1"
  config.vm.provision "shell", path: "actualizar.sh"

  config.vm.network :private_network, ip: "192.168.100.10" # only host
  config.vm.network :public_network, ip: "192.168.100.20"
  config.vm.network :public_network, :bridge=>"eth0"
  config.vm.network :forwarded_port, guest: 80, host: 8080
  config.vm.synced_folder "./www", "/www"

  config.vm.provider "virtualbox" do |vb|
    vb.name = "VM-Prueba1"
    vb.memory = 1024
    vb.cpus = 3
    vb.linked_clone = true
    vb.gui = true
  end
end
```

```
Vagrant.configure("2") do |config|
  config.vm.define "web" do |nodo1|
    nodo1.vm.box = "debian/jessie64"
    nodo1.vm.hostname = "web"
    nodo1.vm.network "public_network", bridge: "eth0"
    nodo1.vm.network "private_network", ip: "10.0.100.101"
  end
  config.vm.define "db" do |nodo2|
    nodo2.vm.box = "debian/jessie64"
    nodo2.vm.hostname = "db"
    nodo2.vm.network "private_network", ip: "10.0.100.102"
  end
end
```

master (Dani)

- la configuración de las VM está en fichero separado

```
API_VERSION = '2'
VMS_FILE_PATH = 'vagrant/vms.yml'

hosts = YAML.load_file(VMS_FILE_PATH)

Vagrant.configure(API_VERSION) do |config|
  hosts.each do |host|
    config.vm.define host['name'] do |node|
      node.vm.box = host['box']
      node.vm.hostname = host['hostname']
      node.vm.network :private_network, ip: host['ip']

      host['sync']&.each { |sync| config.vm.synced_folder sync['host'],
sync['guest'], type: sync['type'] }
      host['ports']&.each { |port| node.vm.network :forwarded_port, guest:
port['guest'], host: port['host'] }

      if host['ansible_local']
        host['ansible_local']['playbooks']&.each { |pb|
          config.vm.provision :ansible_local do |ansible|
            ansible.limit = pb['limit']
            ansible.become = pb['become']
            ansible.playbook = pb['playbook']
            ansible.inventory_path = pb['inventory']
            ansible.version = host['ansible_local']['config']['version']
            ansible.install_mode = host['ansible_local']['config']['mode']
          end
        }
      end

      config.vm.provider :virtualbox do |v|
        v.linked_clone = true
        v.name = host['name']
        v.cpus = host['cpus']
        v.memory = host['memory']
      end
    end
  end
end
```

```
- name: openshift-master
  box: centos/7
  hostname: 10.0.0.10
  memory: 2048
  cpus: 2
  ip: 10.0.0.10
  ports:
```

```
- host: 8443
  guest: 8443
sync:
- host: .
  guest: /vagrant
  type: nfs

- name: openshift-node1
  box: centos/7
  hostname: 10.0.0.11
  memory: 2048
  cpus: 2
  ip: 10.0.0.11
  ports:
    - host: 8081
      guest: 80
    - host: 8481
      guest: 443
  sync:
    - host: .
      guest: /vagrant
      type: nfs

- name: openshift-node2
  box: centos/7
  hostname: 10.0.0.12
  memory: 2048
  cpus: 2
  ip: 10.0.0.12
  ports:
    - host: 8082
      guest: 80
    - host: 8482
      guest: 443
  sync:
    - host: .
      guest: /vagrant
      type: nfs

- name: bastion
  box: centos/7
  memory: 512
  cpus: 1
  ip: 10.0.0.2
  sync:
    - host: .
      guest: /vagrant
      type: nfs
  ansible_local:
    config:
      mode: pip
      version: 2.4.1.0
    playbooks:
      - limit: "all"
        become: true
        playbook: ./ansible/prerequisites.yml
```

```
inventory: ./ansible/inventory/hosts
- limit: "all"
  become: true
  playbook: ./openshift-ansible/playbooks/byo/config.yml
  inventory: ./ansible/inventory/hosts
```

master

```
VAGRANTFILE_API_VERSION = '2'
DEFAULT_PROVIDER_MEMORY = '256'
DEFAULT_PROVIDER_CPUS = '2'
DEFAULT_LINKED_CLONE = true

def do_ports(vm, host)
  if host.has_key?('ports')
    host['ports'].each do |port|
      vm.network :forwarded_port, guest: port['guest'], host: port['host']
    end
  end
end

def do_provision(vm, host)
  if host.has_key?('shell')
    vm.provision "shell", path: host['shell']
  end
end

def do_syncfolder(vm, host)
  if host.has_key?('syncfolder')
    host['syncfolder'].each do |syncf|
      vm.synced_folder syncf['host'], syncf['guest']
    end
  end
end

def do_provider(vm, host)
  memory = DEFAULT_PROVIDER_MEMORY;
  cpus = DEFAULT_PROVIDER_CPUS;

  chars = { :memory => DEFAULT_PROVIDER_MEMORY, :cpus => DEFAULT_PROVIDER_CPUS,
:linked_clone => DEFAULT_LINKED_CLONE }

  if host.has_key?('provider')
    host['provider'].each do |singlechar|
      #memory = host['memory'] ? host['memory'] : DEFAULT_PROVIDER_MEMORY;
      #memory = singlechar['memory'] ? singlechar['memory'] :
DEFAULT_PROVIDER_MEMORY;
      #cpus = singlechar['cpus'] ? singlechar['cpus'] : DEFAULT_PROVIDER_CPUS;
      chars[:memory] = singlechar['memory'] ? singlechar['memory'] :
DEFAULT_PROVIDER_MEMORY;
      chars[:cpus] = singlechar['cpus'] ? singlechar['cpus'] :
```

```
DEFAULT_PROVIDER_CPUS;
  end # singlechar
end

vm.provider :virtualbox do |v|
  v.name = host['name']
  v.memory = chars[:memory]
  v.cpus = chars[:cpus]
  v.linked_clone = chars[:linked_clone]
end

# características.each_with_index do |v,k|
#   #puts "#{k}"
#   vm.provider.k = v[k]
# end

end

hosts = YAML.load_file('Vagranthosts.yml')

Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|

  hosts.each do |host|
    config.vm.define host['name'] do |node|
      node.vm.box = host['box']
      node.vm.hostname = host['name']
      node.vm.network :private_network, ip: host['ip']

      do_ports(node.vm,host)
      do_provision(node.vm,host)
      do_syncfolder(node.vm,host)
      do_provider(node.vm,host)

      end # node
    end # host
  end # config
```

```
- name: c7-gitlab
  ip: 192.168.56.130
  box: centos/7
  shell: c7-gitlab.sh
  ports:
    - host: 8080
      guest: 80
  syncfolder:
    - host: './data'
      guest: '/vagrant'
  provider:
    - cpus: 2
      memory: 512
```

From:

<https://miguelangel.torresegea.es/wiki/> - **miguel angel torres egea**

Permanent link:

<https://miguelangel.torresegea.es/wiki/tech:vagrant:casosuso?rev=1550618183>

Last update: **19/02/2019 15:16**

