Ansible Puppet Vagrant - 3 Jours

Whiteboard : http://bit.ly/383Yy8c

Philippe @ nboost.eu

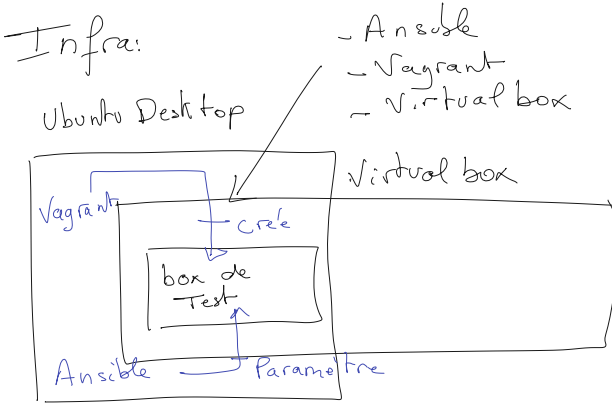
Infra :
Ubuntu 18 Desktop (Pour puppet client, Ansible, Vagrant)
2 x (ou plus) Ubuntu Srv
Le tout est préconfiguré au niveau DNS (IP et /etc/hosts)

Il est possible de créer d'autres serveurs en :
1) décompressant ubuntu srv1 ou srv2 en tant que nouvelles VMs
2) modifiant leur MAC address comme uniques
3) les booter
4) modifier leurs ip interne
5) contrôler depuis ubuntu desktop que les nouveaux srv puissent être pingées
6) ajouter de nouvelles ip dans le desktop et serveurs si nécessaire

Mise en oeuvre :
1) Installer VMWAre Workstation en éval 30 Jours
2) déplier Ubuntu avec ses 3 VMs
3) modifiez la VM Desktop pour autoriser la virtualisation imbriquée :
Modifier la conf / paramètres du CPU / Cocher les cases
Virtualize Intel Vt
Virtualize Perf counter

Ajouter les lignes dans le VMx : (respecter parfaitement ce contenu)
monitor.virtual_mmu = "automatic"
monitor.virtual_exec = "automatic"
hypervisor.cpuid.v0 = "FALSE"
mce.enable = "TRUE"

3) démarrer la VM Desktop (modifier sa mémoire si nécessaire) - le mot de passe est dans le titre de la VM



Exercice Vagrant autonome :

basé sur l'image nboost/u18srv (Ubuntu server en Français)
Créez un projet avec 2 VMs ayant deux ip fixes privées
fixez leur mémoire a 2Go / VM (a paramétrer via le provider virtualbox)
nommez ces VM srv01 et srv02 (leur hostname, et leur nom sous virtualbox, et leur référence Vagrant)

config.vm.define "desktop", primary: true -> le nom de la VM sous Vagrant
master.vm.provider :virtualbox do |vbox|
                vbox.name = "master" -> le nom de la VM sous VirtualBox

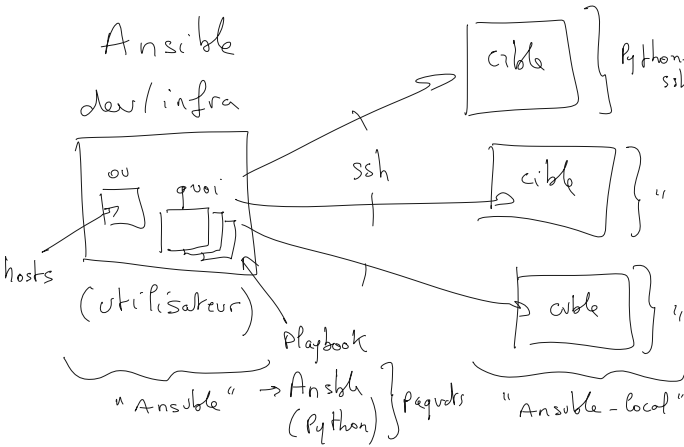
master.vm.hostname = "master" -> le hostname de la VM

Multi serveurs Vagrant :

```
Vagrant.configure("2") do |config|

  (1..2).each do |i|
    config.vm.define "srv0#{i}" do |node|
      node.vm.box = "ubuntu/trusty64"
      node.vm.hostname = "srv0#{i}"
      node.vm.network :private_network, ip: "192.168.33.1#{i}"
      node.vm.provider "virtualbox" do |vbox|
        vbox.name = "srv0#{i}"
        vbox.memory = 2048
      end
    end
  end

end
```

multi :

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/trusty64"

  config.vm.define "srv01", primary: true do |srv01|
    srv01.vm.network :private_network, ip: "192.168.33.10"
    srv01.vm.hostname = "srv01"
    srv01.vm.provider :virtualbox do |vbox1|
      vbox1.name = "srv01"
      vbox1.memory = 2048
    end
  end

  config.vm.define "srv02" do |srv02|
    srv02.vm.network :private_network, ip: "192.168.33.11"
    srv02.vm.hostname = "srv02"
    srv02.vm.provider :virtualbox do |vbox2|
      vbox2.name = "srv02"
      vbox2.memory = 2048
    end
    srv02.vm.provision "shell", inline: <<-SHELL
            echo "Hello world!"
    SHELL
  end

end
```



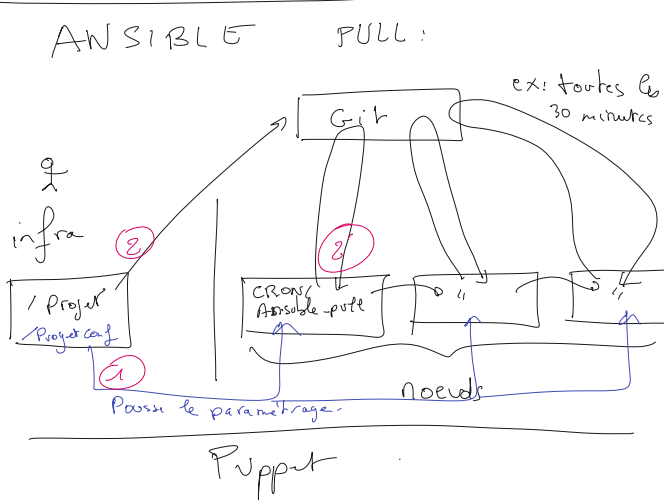
Index des modules ansible :
https://docs.ansible.com/ansible/latest/modules/modules_by_category.html

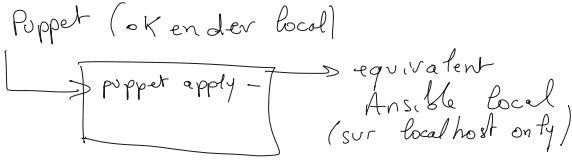
Exercice : créer un fichier et afficher un message si il n'existe pas

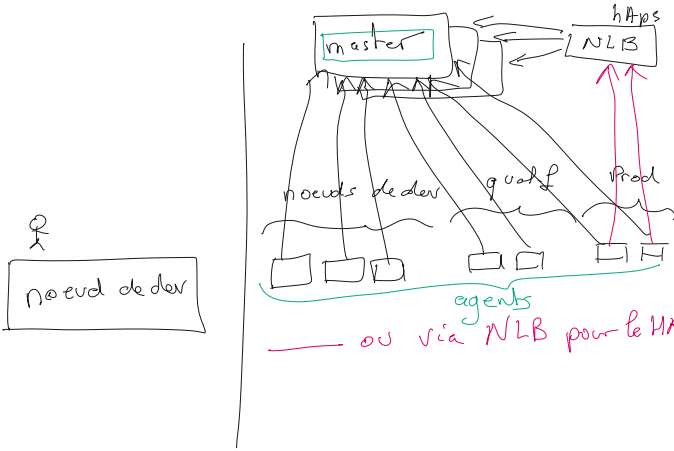
```
---
- hosts: all
  tasks:
    - name: Créer un fichier
      copy: src=myfile dest=/tmp/myfile
      notify: file_ok
  handlers:
    - name: file_ok
      debug: msg="ok fichier créé"
```

```
root@master:/opt/ansible# ansible-playbook  file_playbook.yml

PLAY [all]
*********************************************************************************************
*****************************************

TASK [Gathering Facts]
*********************************************************************************************
****************************
ok: [srv1.stage.lan]
ok: [srv2.stage.lan]

TASK [Créer un fichier]
*********************************************************************************************
****************************
changed: [srv2.stage.lan]
changed: [srv1.stage.lan]

RUNNING HANDLER [file_ok]
*********************************************************************************************
***********************
ok: [srv2.stage.lan] => {
    "msg": "ok fichier créé"
}
ok: [srv1.stage.lan] => {
    "msg": "ok fichier créé"
}

PLAY RECAP
*********************************************************************************************
*****************************************
srv1.stage.lan             : ok=3    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
srv2.stage.lan             : ok=3    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0

root@master:/opt/ansible# ansible-playbook  file_playbook.yml

PLAY [all]
*********************************************************************************************
*****************************************

TASK [Gathering Facts]
*********************************************************************************************
****************************
ok: [srv1.stage.lan]
ok: [srv2.stage.lan]

TASK [Créer un fichier]
*********************************************************************************************
****************************
ok: [srv2.stage.lan]
ok: [srv1.stage.lan]

PLAY RECAP
*********************************************************************************************
*****************************************
srv1.stage.lan             : ok=2    changed=0    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
srv2.stage.lan             : ok=2    changed=0    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

Création de roles pour ansible :
https://docs.ansible.com/ansible/latest/galaxy/dev_guide.html#creating-roles-for-galaxy







wget https://apt.puppetlabs.com/puppet6-release-bionic.deb
dpkg -i puppet6-release-bionic.deb
apt update

sur le serveur :
apt install puppetserver

sur l'agent :
apt install puppet-agent

server :
Tuner la mémoire du serveur web:
nano /etc/default/puppetserver

# Modify this if you'd like to change the memory allocation, enable JMX, etc
JAVA_ARGS="-Xms2g -Xmx2g -Djruby.logger.class=com.puppetlabs.jruby_utils.jruby.Slf4jLogger"

Xms : mémoire minimal automatiquement allouée
Xmx : mémoire maximale

pour ubuntu 16 par ex; ajouter le path :
export PATH=$PATH:/opt/puppetlabs/bin

dans /etc/bash.bashrc par exemple

générer les certificats :
puppetserver ca setup
Generation succeeded. Find your files in /etc/puppetlabs/puppet/ssl/ca

activer le serveur :
systemctl start puppetserver
systemctl enable puppetserver

puis l'agent :
systemctl enable puppet
systemctl start puppet

Si le hostname du serveur n'est pas standard, alors le forcer côté agents et serveur :
root@srv1:/tmp# puppet config set server srv1.stage.lan
root@srv1:/tmp# systemctl restart puppetserver

Définir le serveur sur lequel l'agent doit se connecter :
puppet config set server srv1.stage.lan

Si tout est ok, l'agent a du faire une demande au serveur de requête de certificat :
root@srv1:/tmp# puppetserver ca list
Requested Certificates:
srv2.stage.lan (SHA256) 5B:0C:04:45:23:F5:5B:4C:C6:89:56:E6:5A:1F:6E:BA:04:52:8F:16:16:94:BB:AA:DD:E8:E5:14:3C:76:B5:7E

Réinitialiser l'agent :
https://puppet.com/docs/pe/2017.3/regenerate_puppet_agent_certificates.html

Accepter le certificat de l'agent sur le serveur :
puppetserver ca sign --certname srv2.stage.lan

de fait :
root@srv1:/tmp# puppetserver ca list
No certificates to list

On spécifie un manifeste sur le serveur :
root@srv1:/etc/puppetlabs/code/environments/production/manifests# cat site.pp
group { 'testgroup':
ensure => present,
gid => 2000,
}

que l'on peut tester localement :
puppet apply site.pp

ou attendre que l'agent récupère et applique ce manifest en 30 minutes maximum

définition d'une classe et test :
cat nginx.pp :
```
class nginx {
 package { 'nginx':
  ensure => installed,
 }

 service { 'nginx':
  ensure  => true,
  enable  => true,
  require => Package['nginx'],
 }
}
```

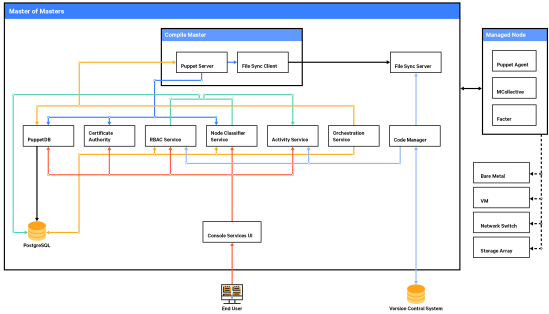
root@srv1:/etc/puppetlabs/code/environments/production/manifests# cat site.pp
```
group { 'testgroup':
   ensure => present,
   gid    => 2000,
}

node default {
  class { 'nginx': }   # utiliser le module nginx défini dans nginx.pp
}
```

test :
root@srv1:/etc/puppetlabs/code/environments/production/manifests# puppet apply .
Notice: Compiled catalog for srv1.stage.lan in environment production in 0.50 seconds
Notice: /Stage[main]/Nginx/Package[nginx]/ensure: created
Notice: Applied catalog in 4.41 seconds

Activation sur le client :

```
root@srv2:/tmp# systemctl status nginx
● nginx.service
   Loaded: not-found (Reason: No such file or directory)
   Active: inactive (dead)
root@srv2:/tmp# puppet agent status
Error: Could not prepare for execution: The puppet agent command does not take parameters
root@srv2:/tmp# puppet agent -t
Info: Using configured environment 'production'
Info: Retrieving pluginfacts
Info: Retrieving plugin
Info: Retrieving locales
Info: Caching catalog for srv2.stage.lan
Info: Applying configuration version '1575644207'
Notice: /Stage[main]/Nginx/Package[nginx]/ensure: created
Notice: Applied catalog in 4.84 seconds
root@srv2:/tmp# systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since ven. 2019-12-06 15:56:51 CET; 5s ago
 Main PID: 2611 (nginx)
   CGroup: /system.slice/nginx.service
           ├─2611 nginx: master process /usr/sbin/nginx -g daemon on; master_process on
           ├─2612 nginx: worker process
           ├─2613 nginx: worker process
           ├─2614 nginx: worker process
           └─2615 nginx: worker process

déc. 06 15:56:51 srv2 systemd[1]: Starting A high performance web server and a reverse proxy server...
déc. 06 15:56:51 srv2 systemd[1]: Started A high performance web server and a reverse proxy server.
root@srv2:/tmp#
```

Architecture complète :



Transférer un fichier vers les agents :

Sur le master, aller dans le dossier de conf :
root@srv1:/etc/puppetlabs/puppetserver/conf.d

créer un fichier fileserver.conf ,
y spécifier une section INI portant le nom que vous voulez, sa localisation, ses droits :
[extra_files]
  path /etc/puppet/files
  allow *

créer le dossier qui contiendra les fichiers :
mkdir -p /etc/puppet/files

y mettre un fichier
root@srv1:/etc/puppetlabs/puppetserver/conf.d# mkdir -p /etc/puppet/files
root@srv1:/etc/puppetlabs/puppetserver/conf.d# echo "ok" > /etc/puppet/files/ok
root@srv1:/etc/puppetlabs/puppetserver/conf.d# cat /etc/puppet/files/ok
ok

Ajouter l'envoi du fichier en ressource (/etc/puppetlabs/code/environments/production/manifests/site.pp) :
```
file { '/home/testfile.txt':
  ensure => file,
  owner  => 'root',
  group  => 'root',
  mode   => 644,
  source => 'puppet://srv1.stage.lan/extra_files/ok',
}
```