1 Ansible

1.1 Sumario

1.2 Introdución

Ansible é unha plataforma de *software* libre para configurar e administrar computadoras. Combina instalación multi-nodo, execucións de tarefas *ad hoc* e administración de configuracións. Adicionalmente, Ansible é categorizado como unha ferramenta de orquestación.

- Manexa nodos a través de SSH e non require ningún *software* remoto adicional (excepto Python 2.4 o posterior). Dispone de módulos que traballan sobre JSON e a saída estándar pode ser escrita en calquera linguaxe. Nativamente utiliza YAML para describir configuracións reusables dos sistemas.
- A plataforma foi creada por Michael DeHaan, tamén autor da aplicación de aprovisionamento Cobbler e co-autor do *framework* para administración remota *Func.4? Es* incluido como parte da distribución de Linux Fedora. (FonteWikipedia)

1.3 Instalación

A administración dos nodos Linux realízase a través do protocolo SSH e dos nodos Windows con WinRM. Non é necesario instalar un axente neles, só é necesario instalar Ansible nun nodo onde se iniciará a comunicación e se realizarán as tarefas especificadas.

Este nodo, onde se instala Ansible, denomínase máquina de control (*Control Machine*). Neste nodo non se instalará *software* adicional, como podería ser base de datos, servidor web, servidor ftp,...

Os requisitos para a máquina de control son os seguintes: Python 2 (Versións 2.6 ou 2.7) ou Python 3 (Versións 3.5 ou posteriores). O Windows non se soporta como máquina de control.

Os nodos a automatizar denomínanse nodos xestionados (*Managed Node*) e só é preciso ter instalado Python 2.5 ou posterior e, por defecto, utilízase sftp para copiar os ficheiros precisos a ese nodo.

1.3.1 Configurar Debian como *Control Machine*

O primeiro é ter a máquina:

- Con Python instalado.
- Actualizada.
- Cunha configuración IP estática: 192.168.1.200.
- Cun nome axeitado: CMAnsible.

Tamén é interesante ter instalado *Open SSH* para conectarse ao equipo remotamente:

```
$ apt install openssh-server
```

Para Debian instalamos Ansible utilizando **pip**:

```
$ apt install python-pip
$ apt install python3-pip
$ pip install ansible
```

1.4 Primeiros pasos

1.4.1 Primeiros pasos *Managed node* Linux

Unha vez instalado Ansible, detallaremos agora os primeiros pasos que podemos realizar para comprender o funcionamento e como conectarse aos nodos que imos administrar.

O primeiro será restaurar unha máquina Debian que fará de nodo cliente (*Managed node*):

- Como NON imos ter servidor DNS configurámoslle unha IP estática: 192.168.1.111
- Instalamos o **openssh-server**:

```
$ apt install openssh-server
```

- Instalamos a ferramenta ***sudo*** para poder administrar o nodo sen empregar o usuario *root* (se queremos empregar *root* faise o mesmo pero saltando todo o que ten que ver con *sudo* e habilitando a conexión SSH con *root*):

```
$ apt install sudo
```

- Creamos o usuario **lansible** e dámoslle permisos:

```
$ adduser lansible
...

#O usuario "lansible" non precisará poñer contrasinal con sudo
$ cat /etc/sudoers
...
# User privilege specification
root      ALL=(ALL:ALL) ALL
lansible  ALL=(ALL) NOPASSWD:ALL

# Allow members of group sudo to execute any command
%sudo    ALL=(ALL:ALL) ALL
...

$ reboot
```

Na máquina **CMAnsible** (*Control Machine*) creamos, ou modificamos, o arquivo ***/etc/ansible/hosts*** indicando os equipos que imos a administrar (de momento só este Debian - 192.168.1.111).

```
$ nano /etc/ansible/hosts
#Clientes Linux
[clientesLinux]
192.168.1.111
```

Para listar os *Managed nodes* que queremos administrar empregamos o comando:

```
$ ansible --list-host all
  hosts (1):
    192.168.1.111
```

Para comprobar a conexión aos *Managed nodes* que imos administrar:

```
#Primeiro facemos un ping a los nodos para ver que todo vai ben
$ for nodo in `cat /etc/ansible/hosts | grep -Ei "^[a-z1-9].*"`; do ping -c 1 $nodo; done

#Logo empregamos o seguinte comando de Ansible
$ ansible all -m ping
192.168.1.111 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ssh: connect to host 192.168.1.111 port 22: Connection refused\r\n",
    "unreachable": true
}
```

Vemos que non é posible conectarse por SSH, isto é porque a clave pública do usuario que está executando o comando **ansible** non foi incluida dentro das claves autorizadas de dito nodo. Podemos copiar a clave pública empregando o comando *ssh-copy-id*:

```
# Xeramos as chaves SSH - Deixamos a passphrase en branco
$ ssh-keygen
```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
...

# Copiamos a chave pública de root para o usuario "lansible" dos nodos a administrar
## Antes instalamos a ferramenta "sshpass"
$ apt install sshpass
## Xa podemos automatizar a copia das chaves nos equipos cliente (o password do lansible é "abc123..")
$ for nodo in `cat /etc/ansible/hosts | grep -Ei "^[a-z1-9].*"`; do echo "abc123.." | sshpass ssh-copy-id lansible@$nodo; done
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'lansible@192.168.1.111'"
and check to make sure that only the key(s) you wanted were added.
```

Xa podemos comprobar a conexión aos *Managed nodes* que imos administrar:

```
$ ansible all -m ping -u lansible
192.168.1.111 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Podemos agora executar comandos en todos eses equipos:

```
$ ansible all -a "uptime" -u lansible
192.168.1.111 | CHANGED | rc=0 >>
 23:52:13 up 56 min,  2 users,  load average: 0,67, 0,43, 0,20
```

Podemos executar comandos con *sudo* neses equipos:

```
$ ansible all -a "fdisk -l" -u lansible -b
...
```

Se queremos só executar os comandos nun "grupo" de equipos:

```
$ ansible clientesLinux -a "ip a" -u lansible
...
```

1.4.2 Primeiros pasos *Managed node* Windows

É tamén posible administrar *hosts* que teñan o sistema operativo Windows instalado. Neste caso non se emprega SSH, emprégase WinRM.

1.- O primeiro que debemos facer é instalar o paquete *pywinrm* na *Control Machine*:

```
$ pip install pywinrm
```

2.- Logo restauramos unha máquina Windows (10, Server 2016, Server 2019):

- Actualizamos a máquina.
- Configuramos a IP: 192.168.1.110
- Instalamos o chocolatey.

3.- Instalamos na *Managed node* Windows unha versión de Python 3.0 ou superior.

```
PS> choco install python3
```

4.- Descargamos o *script* de Powershell ConfigureRemotingForAnsible.ps1 e o executamos con permisos de Administrador.

5.- Creamos o usuario local "wansible", e dende Powershell como Administrador executamos o comando:

```
PS>winrm configSDDL default
# E engadimos ese usuario "wansible" con "Control Total (All Operations)" e "Aceptar"
```

6.- Na máquina *Control machine* modificamos o arquivo *host* do seguinte xeito:

```
$ cat /etc/ansible/hosts
[linux]
192.168.1.118

[windows]
192.168.1.110
[windows:vars]
ansible_user = wansible
ansible_password = abc123..
ansible_port = 5986
ansible_connection = winrm
ansible_winrm_server_cert_validation = ignore
```

7.- Xa estamos preparados para probar algún módulo *ansible* para Windows (para executar os comandos só nos equipos Windows poñeremos "ansible windows..."):

```
$ansible windows -m win_ping -u wansible
192.168.1.110 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}

$ansible windows -m win_disk_facts -u wansible
192.168.1.110 | SUCCESS => {
    "ansible_facts": {
        "ansible_disks": [
            {
                "bootable": true,
                "bus_type": "SATA",
                "clustered": false,
                "firmware_version": "1.0",
                "friendly_name": "VBOX HARDDISK",
                "guid": null,
                "location": "Integrated : Adapter 0 : Port 0",
                "manufacturer": null,
                "model": "VBOX HARDDISK",
                "number": 0,
                "operational_status": "Online",
                "partition_count": 2,
                "partition_style": "MBR",
                "partitions": [
                    {
                        "access_paths": [
                            "\\\\?\\Volume{be3c284e-0000-0000-0000-100000000000}\\"
                        ],
                        "active": true,
...
```