

Ansible Homework

Erick Montes Bedolla

Abstract ▾

Ansible is a software that facilitate the management of multiple servers by remotely accessing them and coordinating the automation of tasks, the use of applications, updating, administration, etc. In this way you can standardize the characteristics necessary to develop a task and there is no need to perform these same steps from one machine to one, but all at the same time.

• Connecting the servers ▾

For this task I have an ubuntu subsystem on windows that I use as a local server. To use as a remote server, I installed with **vagrant** an ubuntu 20.04 virtual machine. When creating the machine, I modified the vagrantfile in the **config.vm.network** option so it appears as another physical device on the network.

```
# Every Vagrant development environment requires a box. You can search for
# boxes at https://vagrantcloud.com/search.
config.vm.box = "ubuntu/focal64"

# 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# `vagrant box outdated`.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"
```

Once the machine is created, check its ip by executing

```
ip --brief addr show
```

```
vagrant@ubuntu-focal:~$ ip --brief addr show
lo                UNKNOWN        127.0.0.1/8 ::1/128
enp0s3            UP             10.0.2.15/24 fe80::9a:94ff:fec9:30e2/64
enp0s8            UP             192.168.1.15/24 2806:266:485:10e2:a00:27ff:fed8:5931/64 fe80::a00:27ff:fed8:5931/64
```

Modify the properties of the ssh connection by entering the sshd_config file, which is the system-wide configuration file for OpenSSH.

```
sudo nano /etc/ssh/sshd_config
```

It opens the file where we have to set the following options as:

```
PasswordAuthentication yes  
ChallengeResponseAuthentication no
```

```
# To disable tunneled clear text passwords, change to no here!  
PasswordAuthentication no  
#PermitEmptyPasswords no  
  
# Change to yes to enable challenge-response passwords (beware issues with  
# some PAM modules and threads)  
ChallengeResponseAuthentication no
```

After modified the file, restart the ssh service with the following command, exit the virtual machine and re-enter in the machine

```
sudo systemctl restart sshd
```

```
vagrant@ubuntu-focal:~$ sudo nano /etc/ssh/sshd_config  
vagrant@ubuntu-focal:~$ sudo systemctl restart sshd  
vagrant@ubuntu-focal:~$ exit  
logout
```

Now in our local server, we generate a ssh key with

```
ssh-keygen -o
```

Leaving blank each of the options that will appear. Once created, we copy the key to the remote server with

```
ssh-copy-id -i ~/.ssh/id_rsa.pub <username>@<remote ip>
```

In my case, the default user is **vagrant** and the ip is 192.168.1.15, so I must run the command as:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub vagrant@192.168.1.15
```

```
erickdknight@LAPTOP-0N3OKRHS:~$ ssh-copy-id -i ~/.ssh/id_rsa.pub vagrant@192.168.1.15  
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/erickdknight/.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  
vagrant@192.168.1.15's password:  
  
Number of key(s) added: 1  
  
Now try logging into the machine, with: "ssh 'vagrant@192.168.1.15'"  
and check to make sure that only the key(s) you wanted were added.
```

Now with this we are able to connect from the local to the remote server with the command

```
ssh <user>@<remote ip>
```

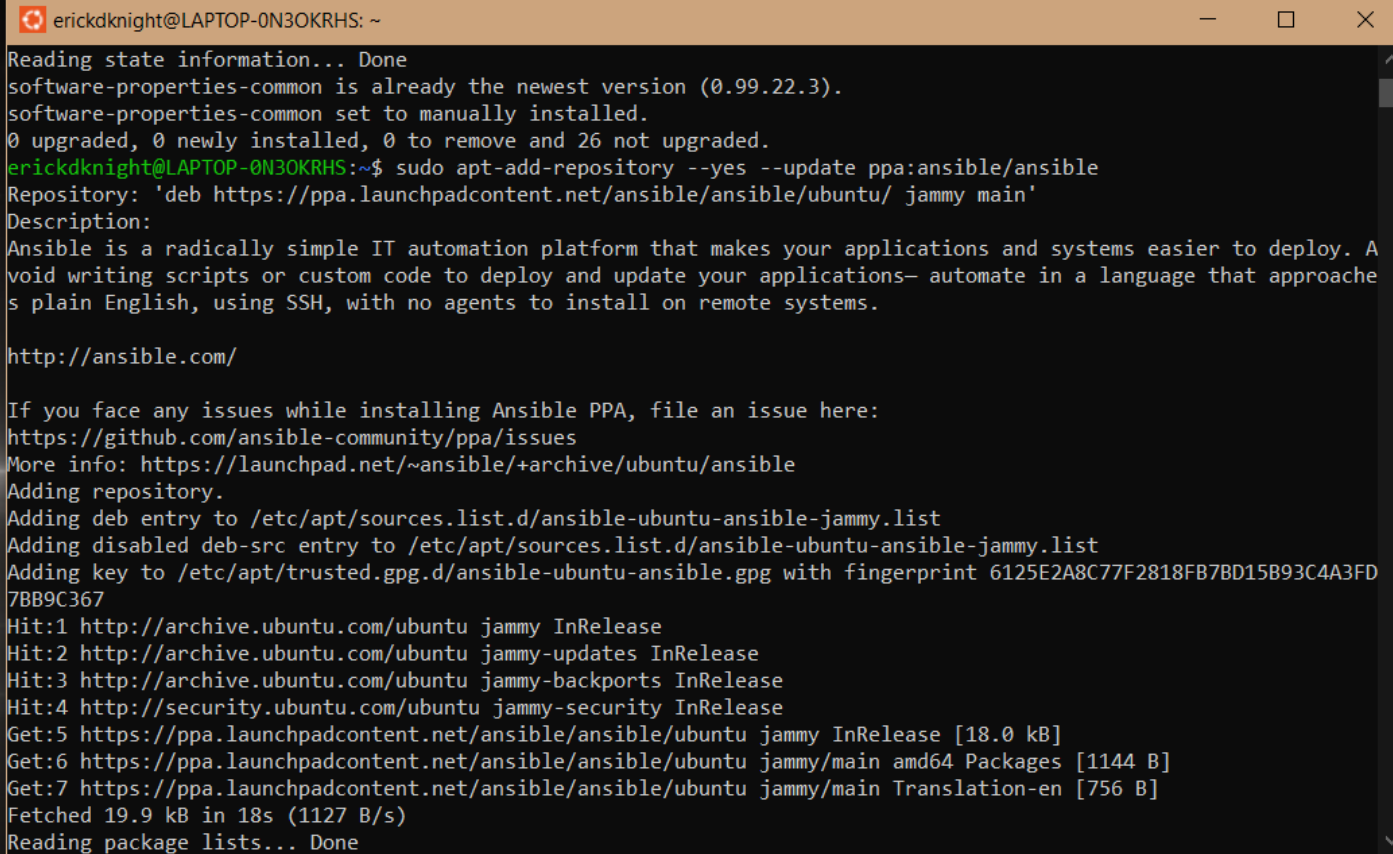
```
erickdknight@LAPTOP-0N3OKRHS:~$ ssh vagrant@192.168.1.15  
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-131-generic x86_64)
```

Exit from the connection

Installing ansible

In the local server, add the Ansible PPA repository with the command:

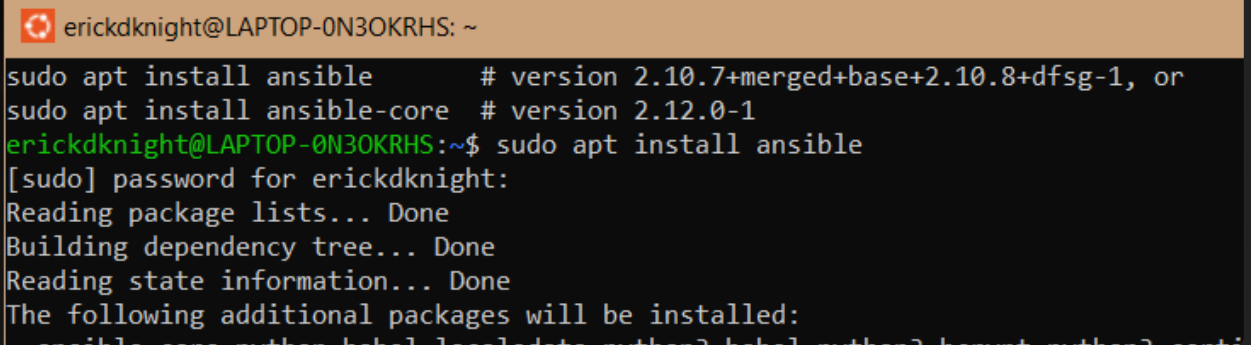
```
sudo apt-add-repository --yes --update ppa:ansible/ansible
```



```
erickdknight@LAPTOP-0N3OKRHS: ~  
Reading state information... Done  
software-properties-common is already the newest version (0.99.22.3).  
software-properties-common set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 26 not upgraded.  
erickdknight@LAPTOP-0N3OKRHS:~$ sudo apt-add-repository --yes --update ppa:ansible/ansible  
Repository: 'deb https://ppa.launchpadcontent.net/ansible/ansible/ubuntu/ jammy main'  
Description:  
Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy. A  
void writing scripts or custom code to deploy and update your applications- automate in a language that approach  
s plain English, using SSH, with no agents to install on remote systems.  
  
http://ansible.com/  
  
If you face any issues while installing Ansible PPA, file an issue here:  
https://github.com/ansible-community/ppa/issues  
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible  
Adding repository.  
Adding deb entry to /etc/apt/sources.list.d/ansible-ubuntu-ansible-jammy.list  
Adding disabled deb-src entry to /etc/apt/sources.list.d/ansible-ubuntu-ansible-jammy.list  
Adding key to /etc/apt/trusted.gpg.d/ansible-ubuntu-ansible.gpg with fingerprint 6125E2A8C77F2818FB7BD15B93C4A3FD  
7BB9C367  
Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease  
Hit:2 http://archive.ubuntu.com/ubuntu jammy-updates InRelease  
Hit:3 http://archive.ubuntu.com/ubuntu jammy-backports InRelease  
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease  
Get:5 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy InRelease [18.0 kB]  
Get:6 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy/main amd64 Packages [1144 B]  
Get:7 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy/main Translation-en [756 B]  
Fetched 19.9 kB in 18s (1127 B/s)  
Reading package lists... Done
```

Then, install the program running:

```
sudo apt install ansible
```



```
erickdknight@LAPTOP-0N3OKRHS: ~  
sudo apt install ansible # version 2.10.7+merged+base+2.10.8+dfsg-1, or  
sudo apt install ansible-core # version 2.12.0-1  
erickdknight@LAPTOP-0N3OKRHS:~$ sudo apt install ansible  
[sudo] password for erickdknight:  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  ansible-core ansible-lint ansible-playbook ansible-runner ansible-test ansible-vault ansible-zsh-plugins
```

Ansible should be installed by now. We can check it running the command:

```
ansible --version
```

```
erickdknight@LAPTOP-0N30KRHS:~$ ansible --version
ansible [core 2.13.6]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/erickdknight/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/erickdknight/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.10.6 (main, Aug 10 2022, 11:40:04) [GCC 11.3.0]
  jinja version = 3.0.3
  libyaml = True
```

• Creating and running hosts file and playbook.yml ✓

Create two files located on the local server, hosts file and playbook.yml.

Hosts file

In **ansible_host** we write the ip of the remote server.

In **ansible_user** we define the name of the remote server user

```
[servers]
server1 ansible_host=192.168.1.15
[all:vars]
ansible_user=vagrant
ansible_python_interpreter=/usr/bin/python3
```

Playbook.yml

In this file we describe the servers we are going to be working with and the process we need to run. In this case, we need to install VIM in the remote server.

```
- hosts: all
  become: true
  tasks:
    - name: Installing VIM
      apt: name=vim state=latest
    - name: Sending Hello World!
      shell: echo Hello World!
```

Enter the location of both files

```
erickdknight@LAPTOP-0N30KRHS:~/to-vm_ubuntu-ansible$ ls
hosts  playbook.yml
```

We can verify the information in the hosts file with:

```
ansible-inventory -i hosts --list -y
```

```
erickdknight@LAPTOP-0N30KRHS:~/to-vm_ubuntu-ansible$ ansible-inventory -i hosts --list -y
all:
  children:
    servers:
      hosts:
        server1:
          ansible_host: 192.168.1.15
          ansible_python_interpreter: /usr/bin/python3
          ansible_user: vagrant
      ungrouped: {}
```

To run the playbook, we write:

```
ansible-playbook -i hosts playbook.yml
```

```
erickdknight@LAPTOP-0N30KRHS:~/to-vm_ubuntu-ansible$ ansible-playbook -i hosts playbook.yml

PLAY [all] *****
TASK [Gathering Facts] *****ok: [server1]

TASK [Installing VIM] *****ok: [server1]

TASK [Sending Hello World!] *****changed: [server1]

PLAY RECAP *****server1
: ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

With this, we installed VIM in the remote server from the local server using Ansible