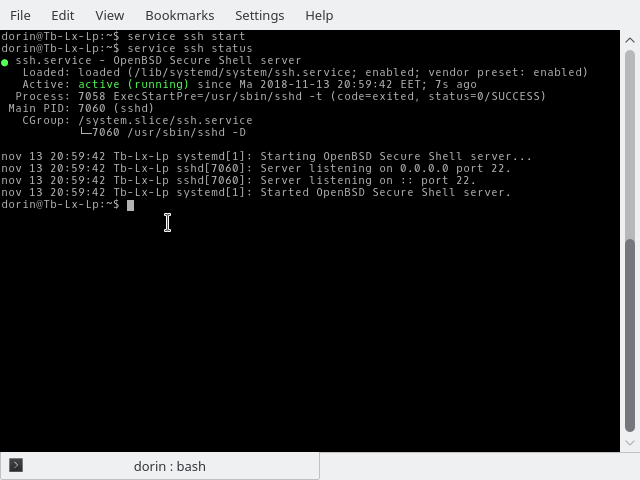
Understand **Ansible** better, by making it work

Prerequisites:

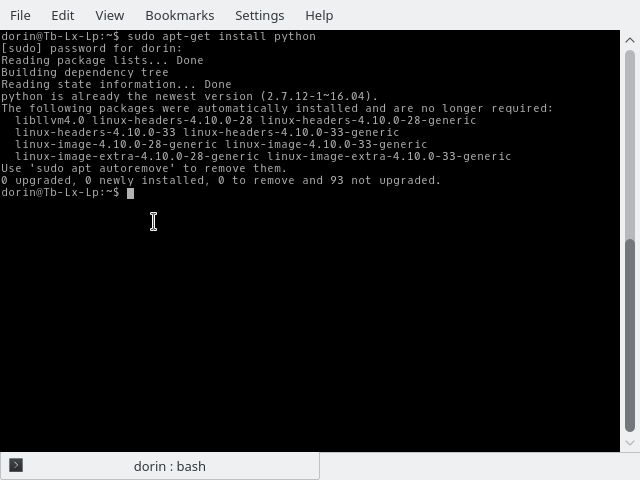
* Two or more Linux stations, in my case I had three all different distributions just to show how great Ansible is.
* One station will be the Ansible server (with Lubuntu 18.04) and thus will have Ansible installed on it
* The other two stations (a Kubuntu 16.04 and a Kali 2.0) will be hosts and all they need to have is:
  + SSH service installed / enabled and
  + Python installed (which all Linux distros do these days, we still have to check it though)
  + A non-root Sudo-er user
  + Also we will need to know their IP address
* SSH keys generated for the non-**root** user on the Ansible server.

So, the plan is to first check the prerequisites and then to actually install Ansible on the server and make it work, get the other stations executes commands.

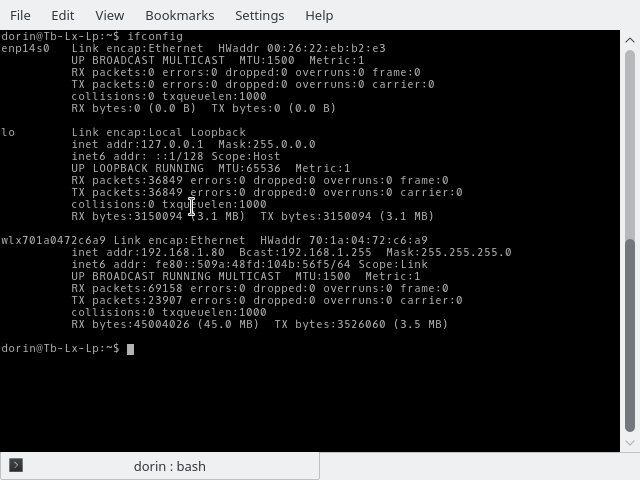
Let’s start with host1, SSH status:



Everything is peachy, we move on to check host1 Python status

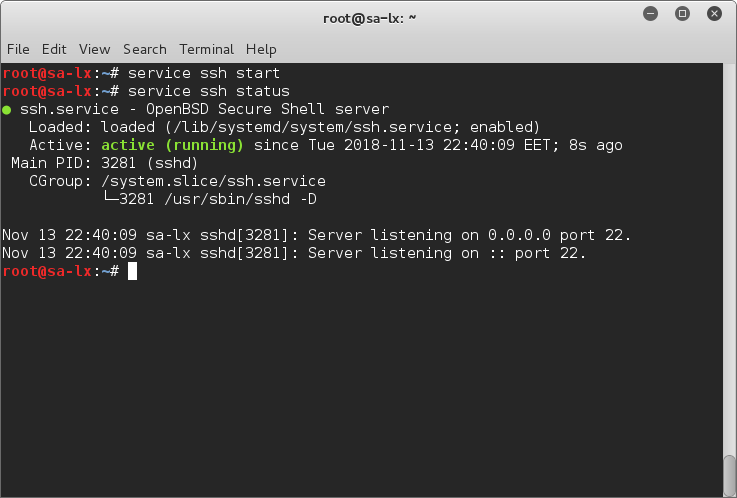


Looks great it has it installed, next let’s have a look at the IP v4 address of host1

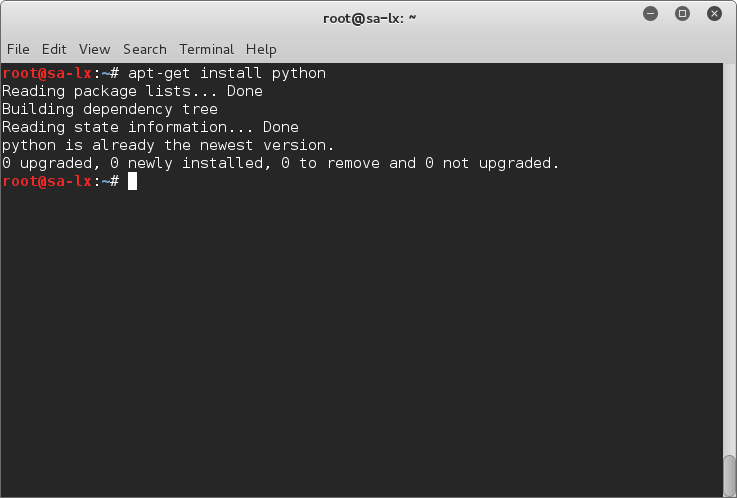


Since it’s an Ubuntu distro and is the only user after installation it is a SUDO user and is not root.

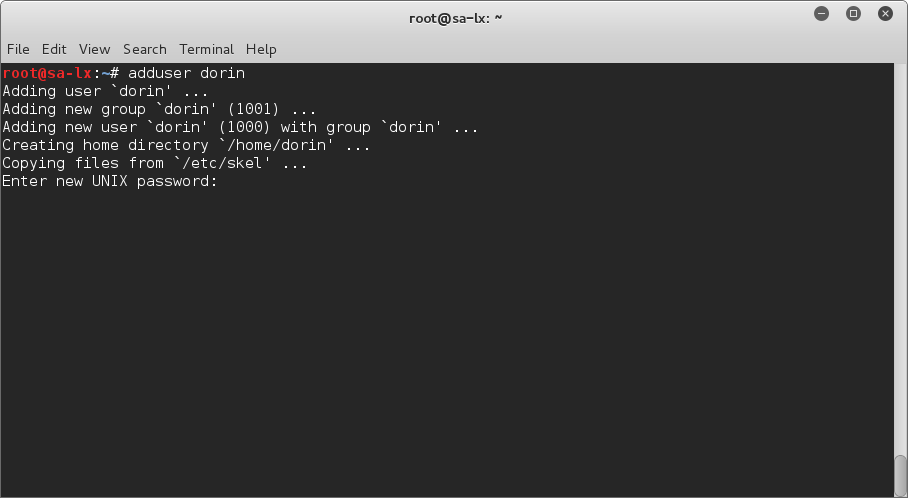
We got everything checked on host1, now let’s check host2 SSH status

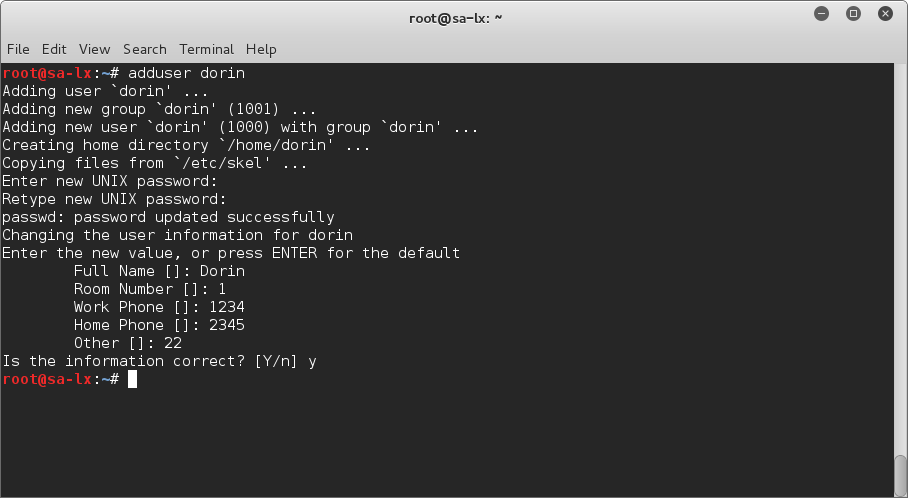


SSH is up and running, Python is also installed as we are seeing bellow:

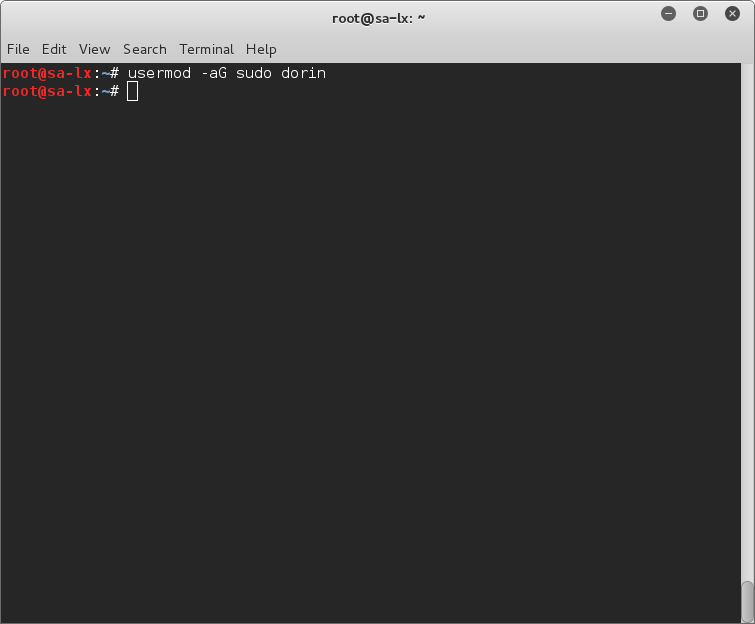


Now, we need to create the Non-root user:

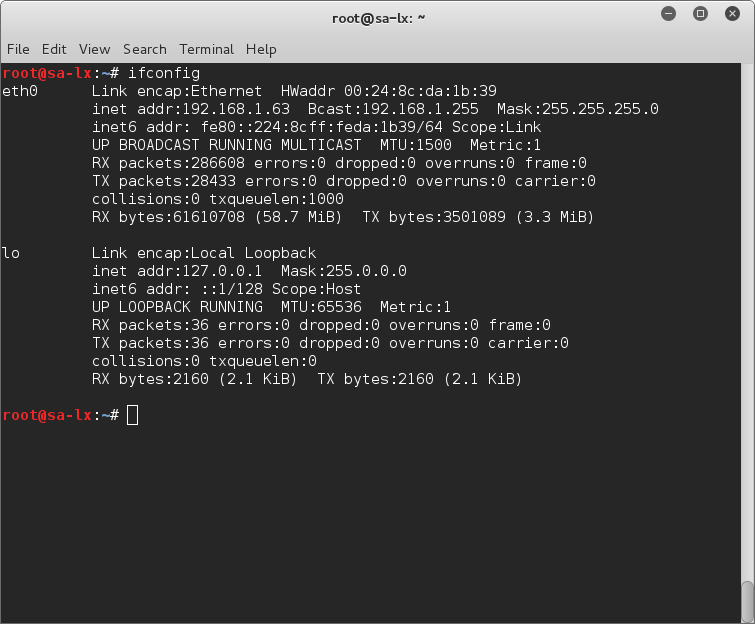




Make that user a so called Sudo-er



Let’s have a look at host2 IP v4 address:

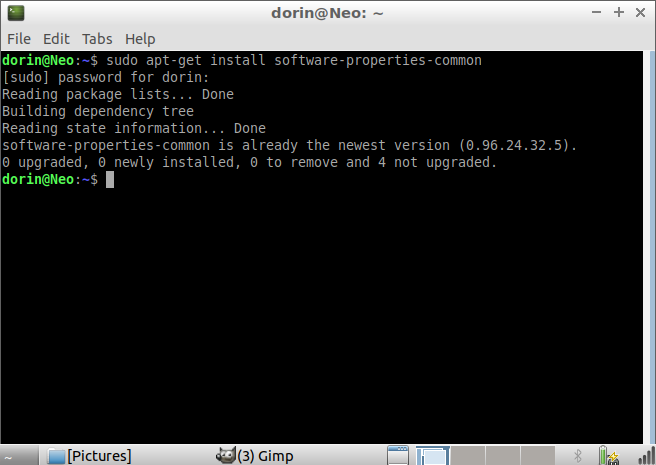


We got all the info and verifications on host1 and host2 let’s move to the server

Installing Ansible:

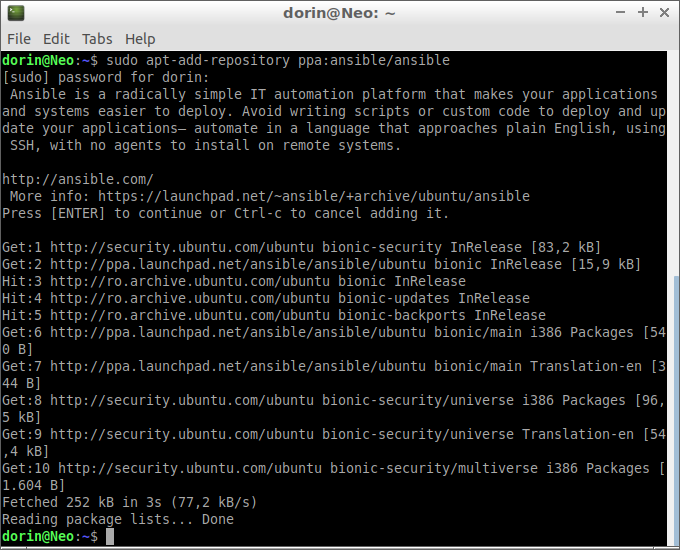
To begin using Ansible as a means of managing your various servers, you need to install the Ansible software on at least one machine.

To get the latest version of Ansible for Ubuntu, you can add the project's PPA (personal package archive) to your system. Before doing this, though, you should first update your package index and install the software-properties-common package. This software will make it easier to manage this and other independent software repositories:



Then add the Ansible PPA by typing the following command:

sudo apt-add-repository ppa:ansible/ansible



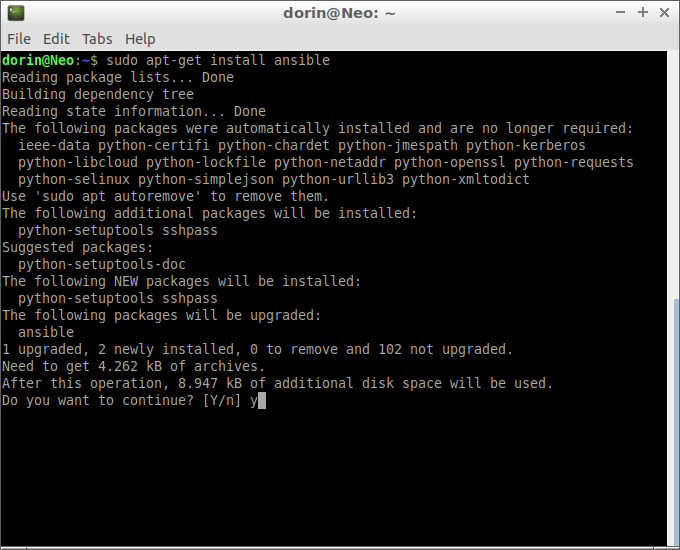
Press ENTER to accept the PPA addition.

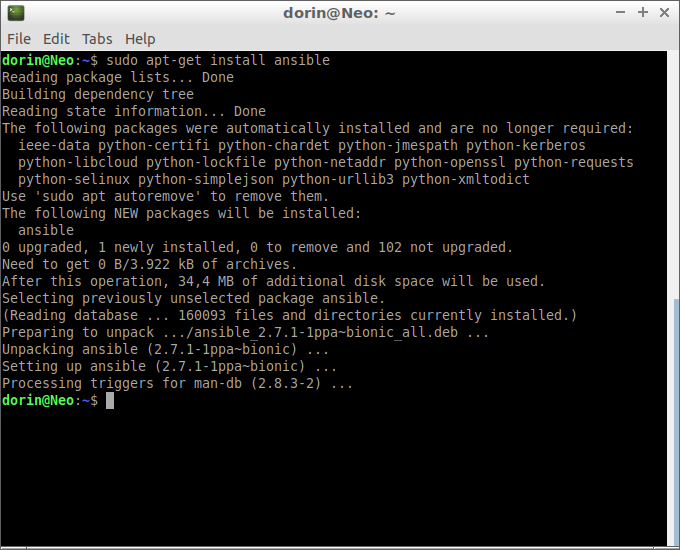
Next, refresh your system's package index once again so that it is aware of the packages available in the PPA:

sudo apt update

Following this update, you can install the Ansible software:

sudo apt install ansible





Your Ansible server now has all of the software required to administer your hosts.

Setting Up Ansible Hosts

Ansible keeps track of all of the servers that it knows about through a hosts file. We need to set up this file first before we can begin to communicate with our other computers.

Open the file with sudo privileges, like this:

sudo nano /etc/ansible/hosts

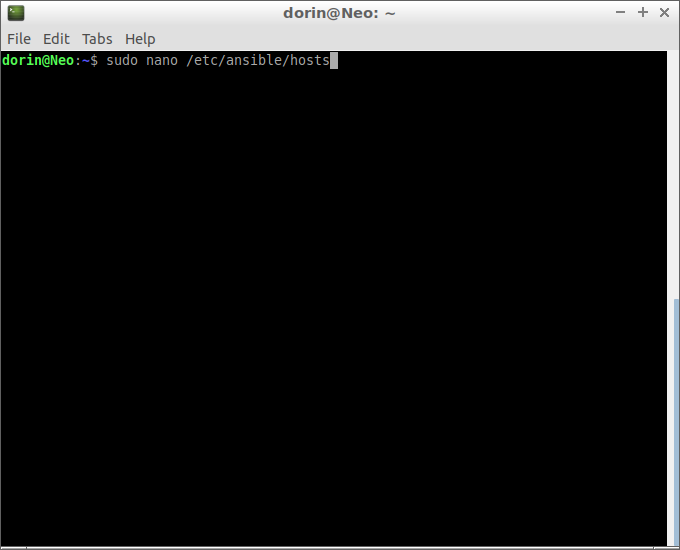
Inside the file, you will see a number of example configurations that have been commented out (with a # preceding each line). These examples won’t actually work for us since the hosts listed in each one are made up. We will, however, keep these examples in the file to help us with configuration if we want to implement more complex scenarios in the future.

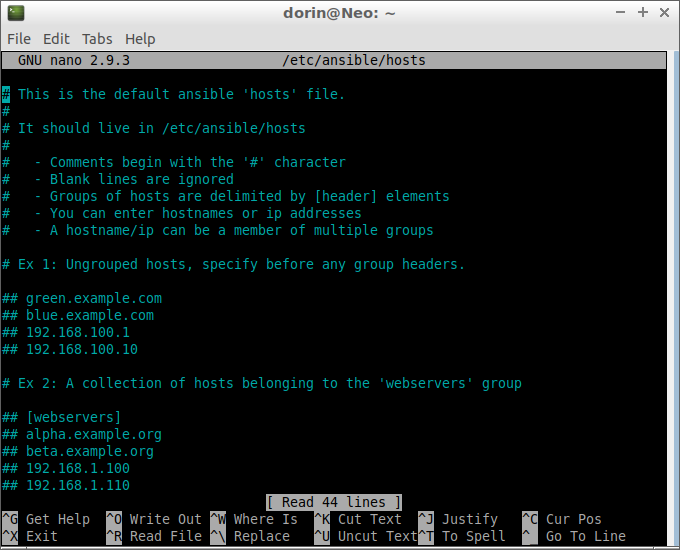
The hosts file is fairly flexible and can be configured in a few different ways. The syntax we are going to use, though, looks like this:

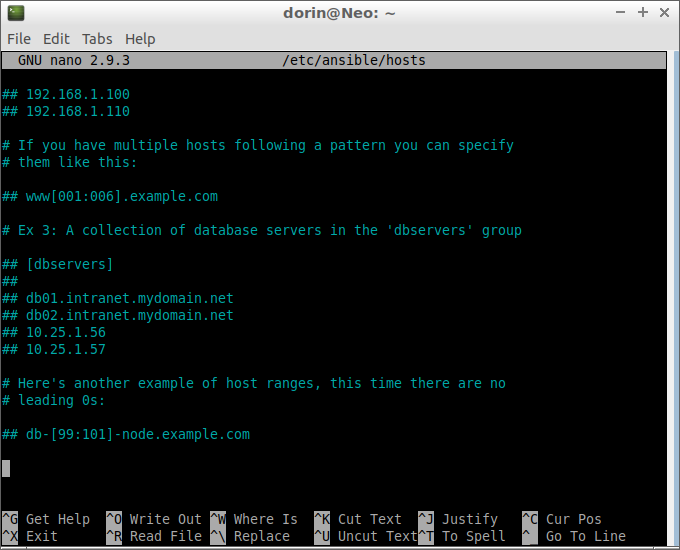
[group\_name]

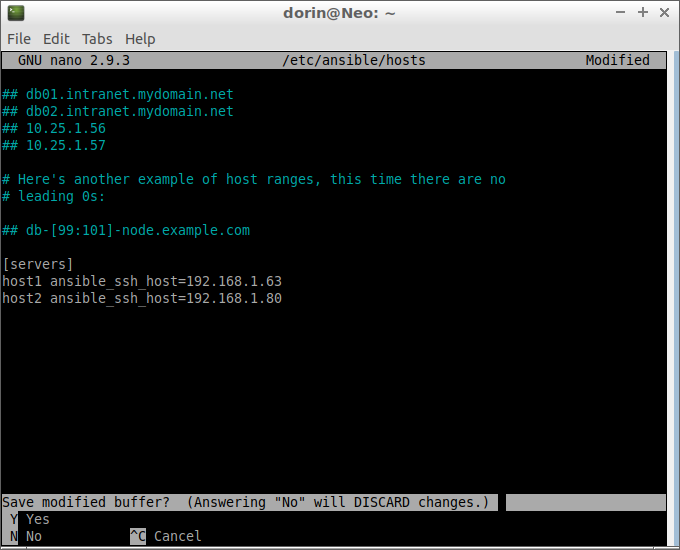
alias ansible\_ssh\_host=your\_server\_ip

In this example, group\_name is an organizational tag that lets you refer to any servers listed under it with one word, while alias is just a name to refer to one specific server.







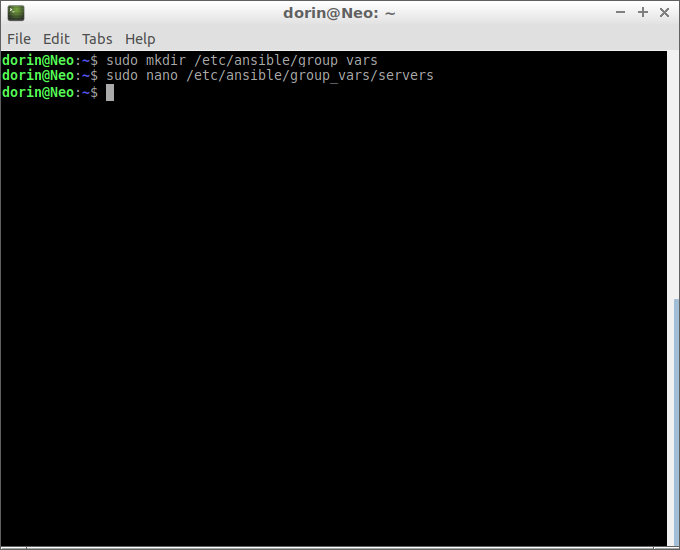


We can create a file that tells all of the servers in the "servers" group to connect as the root or any other user, in my case I’ll chose the user dorin.

To do this, we will create a directory in the Ansible configuration structure called group\_vars. Within this folder, we can create YAML-formatted files for each group we want to configure:

sudo mkdir /etc/ansible/group\_vars

sudo nano /etc/ansible/group\_vars/servers

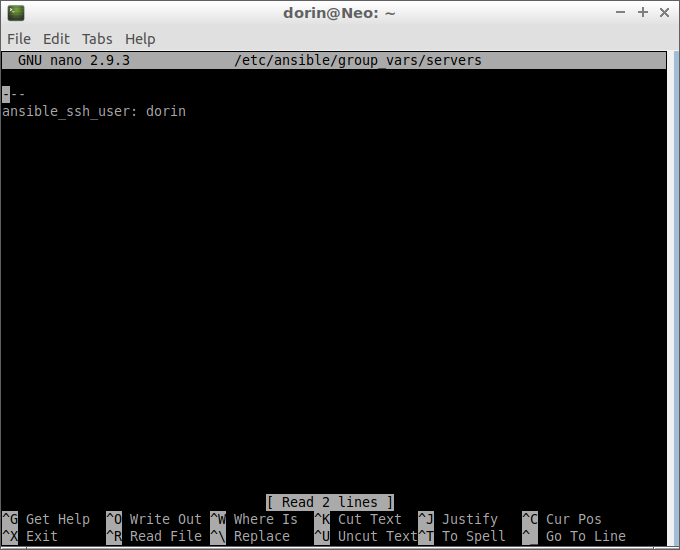


We can put our configuration in here. YAML files start with "---", so make sure you don't forget that part.

/etc/ansible/group\_vars/servers

---

ansible\_ssh\_user: dorin

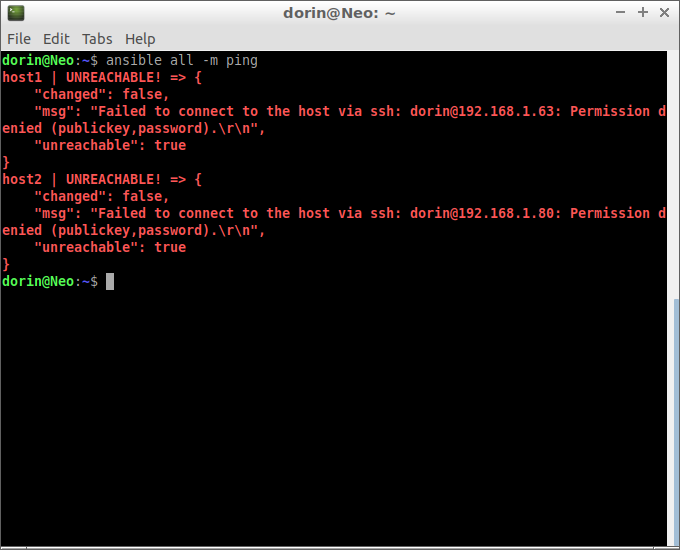


Save and close this file when you are finished.

Now that we have our hosts set up and enough configuration details to allow us to successfully connect to our hosts, we can try out our very first command.

Ping all of the servers you configured by typing:

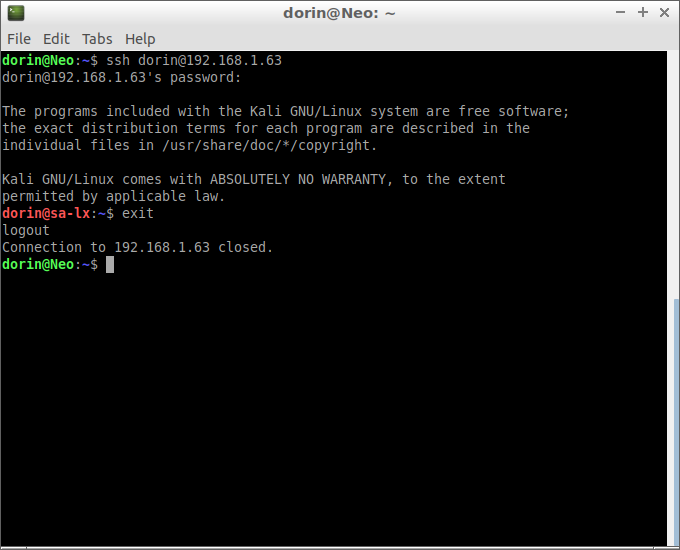
ansible -m ping all

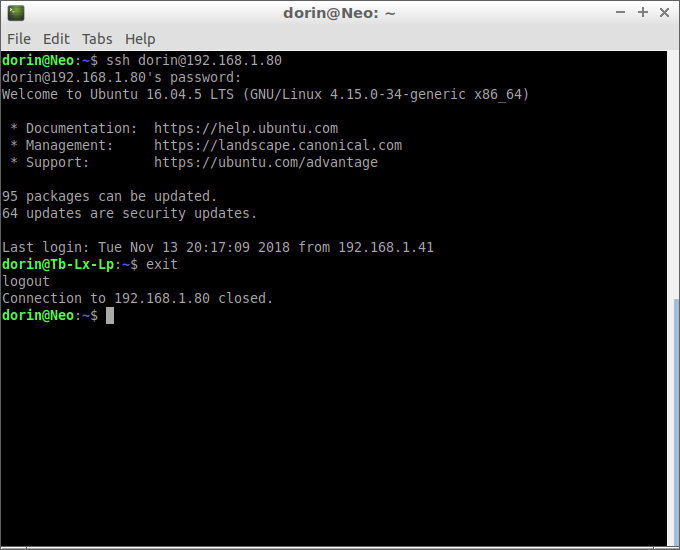


And boom access denied ☺

I intentionally left the adding of the public key afterwards so you can realize better that if you don’t do it you won’t get access to the hosts.

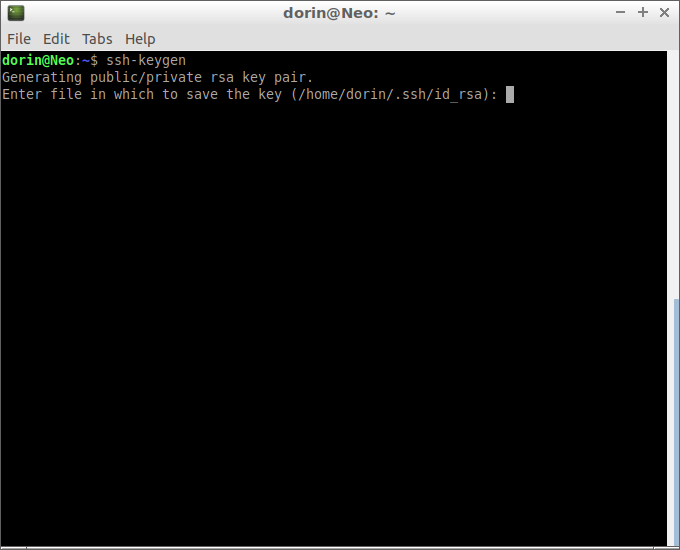
First let’s try a simple SSH login to each host:



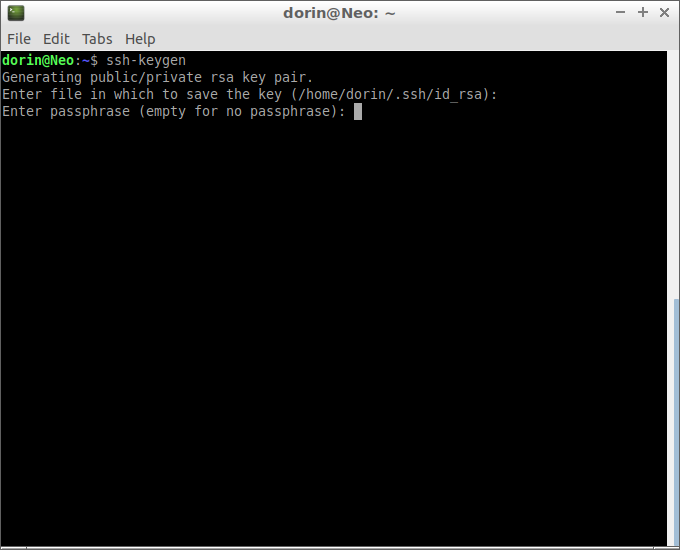


And now we will generate a public/private rsa key pair, in order to add the public key to the hosts.

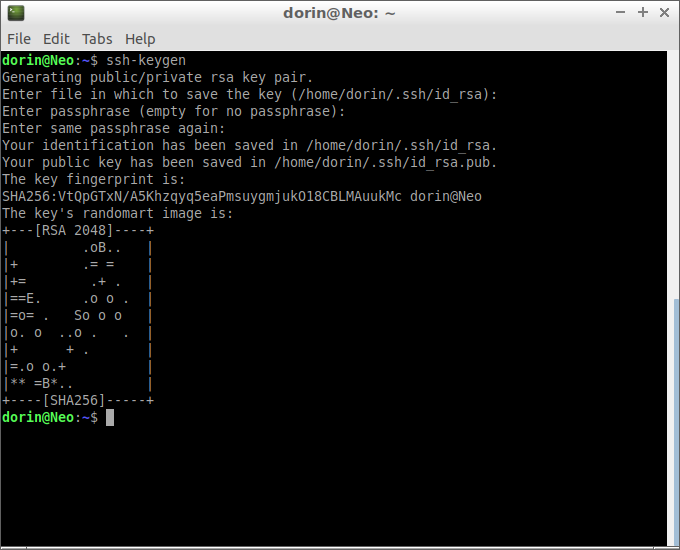
First we generate the key pair



In this example I’ll choose to leave the password blank, but you can choose to enter a password but you will be prompted for it at each command (connection).

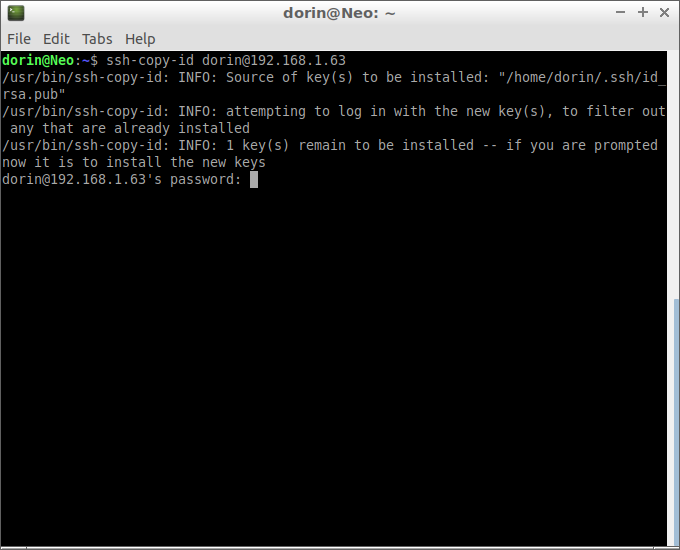


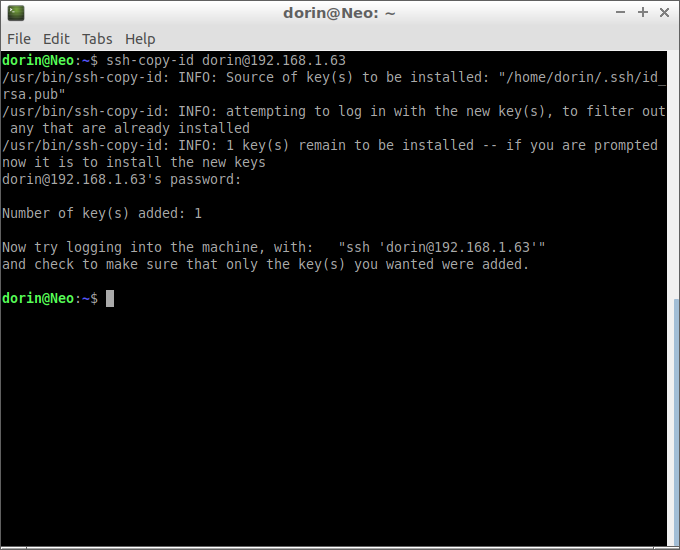
Just press Enter,



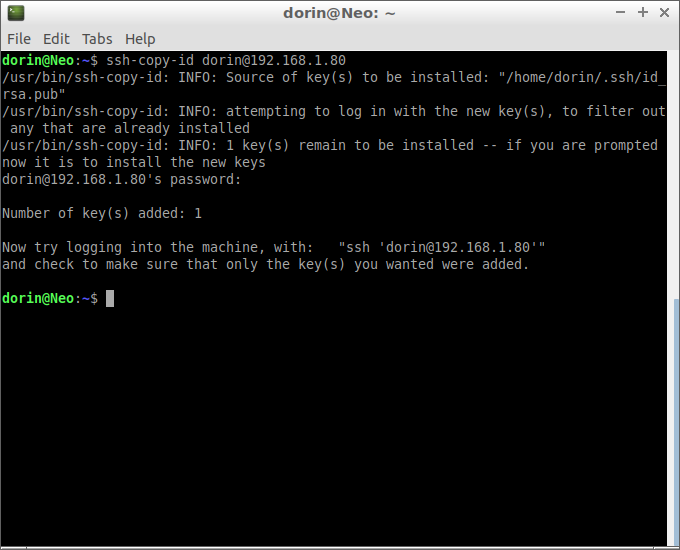
I got the key pair generated, now I’ll add the public key to both hosts

Host1:





And host2:

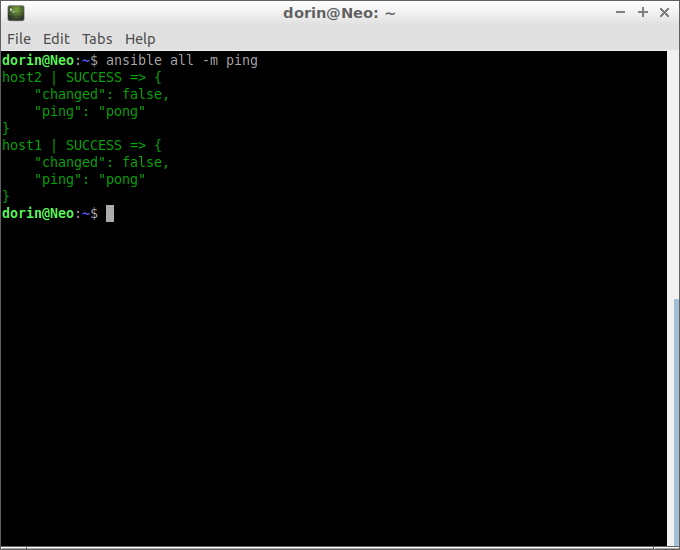


Done.

Now It won’t be a problem connecting to my hosts next time I’ll try.

So let’s give it a go:

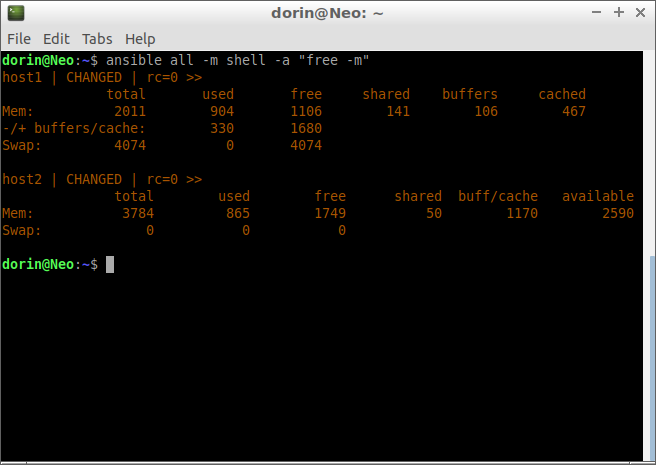
ansible -m ping all:



And is working, you see how great it is?

Let’s try another command:

Ansible all –m shell –a “free -m”



It tells us the memory status of host1 and host2 in MB.

You could do lot more with Ansible but this is a simple case just to see it working.