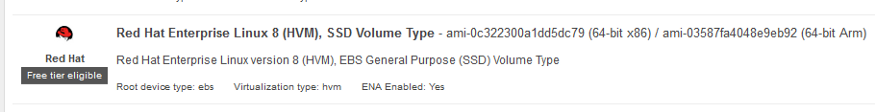
**Step 1:**

Go to your AWS management console and launch an RHEL instance. Use Putty/MobaXterm to SSH into the instances once they start running.



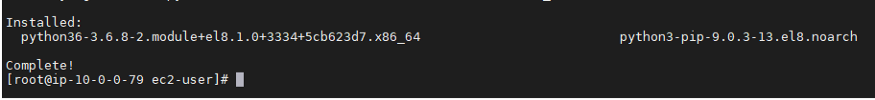
Selecting the AWS AMI

**Step 2:**

Next, we will be installing Python3 in our RHEL 8 EC2 instances. If you have a special requirement of Python2, mention Python2 in the command, otherwise use the command below as is.

*#yum install python3 -y*

https://miro.medium.com/max/60/1*4BZm9Ix9JKf9J_rYaKa5EQ.png?q=20

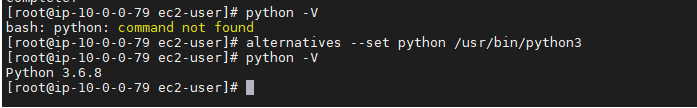


Installation of python3 completed

Next, we should setup alternatives for our Python3 commands to run using Python. Use the command below for both instances:

*#alternatives — set python /usr/bin/python3*

https://miro.medium.com/max/60/1*48Ge5N7UMSD6xrMkXHCSHw.png?q=20



Set alternatives

**Step 3:**

Now, you must install Python3-pip on the instance. Pip is a package manager for Python that allows you to install and manage additional Python packages which are not part of the standard python library.

*#yum -y install python3-pip*

https://miro.medium.com/max/60/1*NdaBwlUt2WY8lX7qCSbsPA.png?q=20



Installing python3-pip

**Step 4:**

We wouldn’t be able to install Ansible as a root user here, because in RHEL 8, this operation is not allowed. So, we are going to create a new user and setup a password for it.

*#useradd ansiblecontrol*

*#passwd ansiblecontrol*

For the second instance, create a user named, “managedhost” and setup a password for it as well.

Once completed, we should add the user ‘ansiblecontrol’ to sudoers file for sudo access. Use the command below to do that.

*#echo “ansiblecontrol ALL=(ALL) NOPASSWD: ALL” >> /etc/sudoers*

Similarly, repeat the above with the user “managedhost” in the second instance.

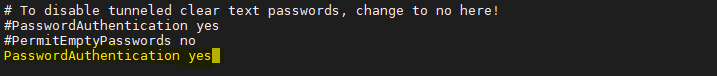
**Step 5:**

The control node, also referred to as Ansible Master, connects to the managed host using SSH. Though using key-based authentication is recommended, when you are at a learning stage, use password-based authentication.

Edit the sshd\_config file in /etc/ssh directory using vi.

*#vi /etc/ssh/sshd\_config*

https://miro.medium.com/max/60/1*2uK1jKwHuR4Ov4EteHwcvg.png?q=20



Editing enabling password authentication

Find the PasswordAuthentication string in that file and change the value against it to ‘yes’. Save and quit the file.

Reload sshd service using the command below:

*#service sshd reload*

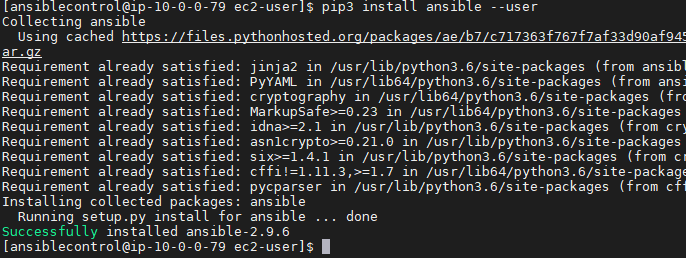
**Step 6:**

In RHEL 8, Ansible cannot be installed as a root user. So, we are going to install Ansible after switching user to the one which we created earlier in Step 4. Use the commands provided below for the same.

*#su ansiblecontrol*

*$pip3 install ansible — user*

https://miro.medium.com/max/60/1*FOH_3DB_Tfc-oSydkQUTUw.png?q=20



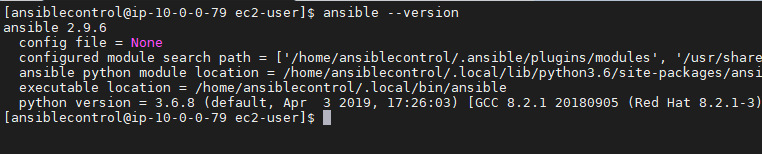
Installation of Ansible

**Step 7:**

Let’s check the Ansible version and its config file.

*$ansible — version*

https://miro.medium.com/max/60/1*j1cWpXP3fQviIN9ebOhD7w.png?q=20



Checking Ansible version

As evident from the above screenshot, we have installed Ansible 2.9.6 and there is no config file present. In older versions of Ansible, it used to be present in the directory /etc/ansible/ansible.cfg, but in the newer versions, we are supposed to create one manually. We also need a hosts file to be able to run playbooks, so we are going to create the required directory and files as well.

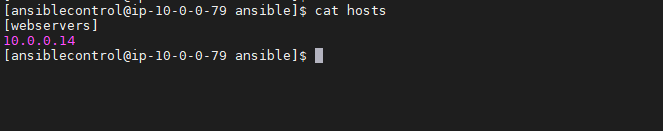
*$sudo mkdir /etc/ansible*

*$cd /etc/ansible/*

*$sudo vi hosts*

We will now create a file named ‘hosts’ in the Ansible directory and input the private IPs of the managed hosts. In our case, we have 1 instance, so I will give that IP alone.

https://miro.medium.com/max/60/1*Ug29J4BxAiCCZ0-8QFZQjg.png?q=20



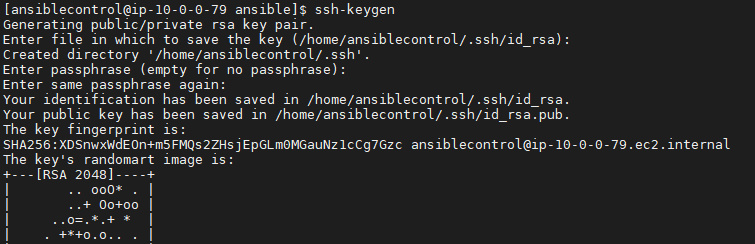
Configuring the hosts file for Ansible

In the above screenshot, notice that I have mentioned the private IP of the managed host under [webservers]. This is called creating a group. I created a group named ‘webservers’ so that I can call all the servers at once, and thus avoid using individual IPs in my Ansible commands.

Now let me show you, how we can configure password-less authentication between the ansible-control host and managed hosts. Run the command below in the Ansible directory:

*$ssh-keygen*

https://miro.medium.com/max/60/1*TUgqmzkoD8l_e4yNhsdG-Q.png?q=20



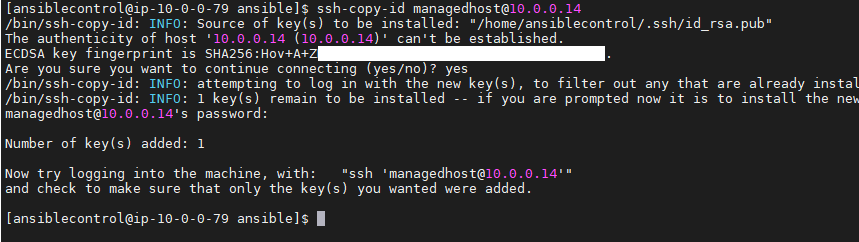
ssh-keygen command

Note here, that your key’s randomart image is confidential information and do not share it with anyone else. The above image has been cropped for this reason.

You need to copy the public key to the target instances, for Ansible to be able to connect to them. Note that you should allow the security group to allow SSH connection from the control node to managed host. Copy the public key to the managed host using the following command:

*$ssh-copy-id*[*managedhost@*](mailto:managedhost@172.31.90.161)*10.0.0.14*

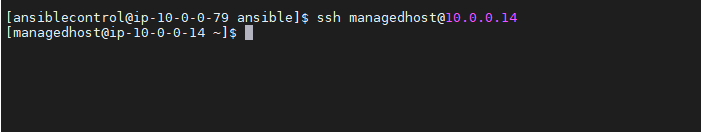
https://miro.medium.com/max/60/1*cQ7J7ZGDP-B38mQwQ0D2lw.png?q=20



Adding the key to the managed host

It will ask for the password configured for the managed host. I just copied the public key from the control machine to the managed host, so that the former can connect to the latter using that key via SSH protocol.

https://miro.medium.com/max/60/1*H6ujaIcJde-d9b8RCdbI1w.png?q=20



Checking the SSH connection

Above screenshot shows how I logged in to the managed host from the Ansible control machine.

Once this is done, you can go the sshd\_config file and disable the password authentication, which is a hardening technique you can use. It will make the instance secure as well. Path for the sshd\_config file is:

*/etc/ssh/sshdconfig*

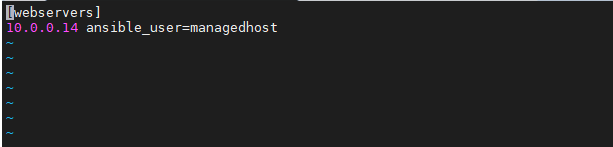
I know this step seems to be a tricky one, but trust me, it doesn’t take much time if you follow the steps correctly and Ansible can be configured easily.

**Step 8:**

Go to the hosts file in the /etc/ansible directory and edit the file. Add the username for the managed instance as ‘managedhost’ and then save the file.

This will tell Ansible to use ‘managedhost’ as username while connecting to it.

https://miro.medium.com/max/60/1*D7j3N53L9F-_gbQdZub0Eg.png?q=20

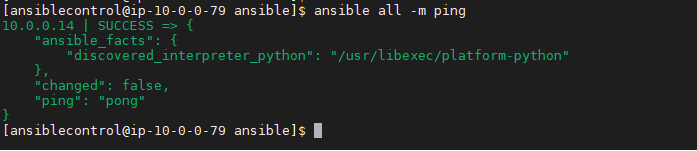


Adding IPs to the hosts file

Let’s check the Ansible connection now. Run the command below to run ping module:

*$ansible all -m ping*

https://miro.medium.com/max/60/1*eufx6tgZBkgxLqK8B6J_dA.png?q=20



Execute Ansible ping command to check connectivity

We just ran the ping module, which means we pinged the instances in the group ‘webservers’ mentioned in the ‘hosts’ file created earlier. The above screenshot implies the same, where it mentions *10.0.0.14 | success;* and at the end*“ping” : “pong”* which is Ansible’s way of saying the ping has been successful.

Hope the steps mentioned above are clear. Share your thoughts in the comments section below. Watch this space for more blogs on Ansible.