We will enumerate the steps required to setup a password less ssh connection between two AWS EC2 instances. With this approach, you will be able to connect to any instance using SSH.

To do any automation using Ansible,

- You need the SSH connection between instances.
- Do the initial setup
- We do not want to login to each of the servers and create first user and setup the authorized_keys manually.

Solution:

- We will create an ansible playbook which will setup a user
- We will use the .pem file which we have associated while launching the instances to connect to the server initially.

Agenda:

1. Create and Setup AWS EC2 instances
2. SSH to the Ansible master node
3. Setup a new user devops on the Ansible master node manually
4. Run the playbook to setup a devops user on all other nodes
5. If you do not want to create a new user and use the default user like `ec2-user,ubuntu` then you can skip the creation of user.

Launch two AWS EC2 instances

1. Login to AWS Console
2. Search for service EC2 ->Click on EC2 -> Instances ->Launch Instance -> Linux AMI2 -> select default instance t2.micro -> configure security group Review and Launch -> create a key to connect to the instance

Connect to Ansible Master Node using SSH

- Run the below command using git bash
- You need to use the pem file which you have downloaded while launching an instance

```
ssh -i "ansiblepem.pem" ec2-user@ec2-3-18-106-15.us-east-2.compute.amazonaws.com
```

Now you are connected to your master Ansible node

- Run the yum update command to get all system updates

```
sudo yum update
```

Prerequisite

1. Python should be installed

2. Install Ansible

```
sudo amazon-linux-extras install ansible2
[ec2-user@ip-172-31-22-242 ~]$ ansible --version
ansible 2.9.9
```

Setup a devops user on Master Node

- Create a user devops
- Set a password

```
sudo -i
useradd -m -s /bin/bash devops
passwd devops
```

- Add the user in sudoers.d file, this allow user to run any command using sudo without passing their password

```
echo -e 'devops\tALL=(ALL)\tNOPASSWD:\tALL' > /etc/sudoers.d/devops
```

Encrypt the password

```
sudo yum whatprovides */mkpasswd
sudo yum install expect
[root@ip-172-31-22-242 ~]# mkpasswd devops
Xphw>97Wt
```

User devops has created successfully.

Now we will generate the SSH keys for the devops user

Generate a SSH Key

1. Login as a devops user and follow the prompts

ssh-keygen -t rsa

It will generate the public and private key file for the devops user.

Now we have to add this public key to all the remote hosts.

- copy the id_rsa.pub file to your git repo or anywhere on the master server so that you can refer that in your playbook

How we will connect initially to our other nodes ?

- If you try to run the below command as ec2-user and devops you will get the error "Permission Denied" because we have not copied the public key to the remote hosts yet

ssh -i ~/.ssh/id_rsa ipoftheserver

Install git and clone the git repo

sudo yum install git

git clone https://github.com/DevenderMusukula/Devops-EC2-SSH-KEYS-setup.git

- Write a playbook to create a new user, set a password, add it to the sudoers file.
- lookup command will try to find the .pub file on the master ansible node for devops user and put that public key in the authorized_keys on the remote servers. Put the .pub file either on your git repo or anywhere on the master node

```
- name: Add a new user named devops
user:

name=devops

password={{ devops_password }}


- name: Add devops user to the sudoers

copy:
```

```
dest: "/etc/sudoers.d/devops"

content: "devops  ALL=(ALL)  NOPASSWD: ALL"


- name: Deploy SSH Key

authorized_key: user=devops

key="{{ lookup('file', 'devops_id_rsa.pub') }}"

state=present
```

- Playbook to call the above role

```
- hosts: all
become: true

become_user: root

gather_facts: false

tasks:

- include_role:

name: add_devops_user

tasks_from: add_user.yml
```

How to run the playbook

- You need to provide the user ec2-user and the key to connect to the remote host.
- I am assuming all the remote hosts have same keys
- You need to use the .pem file to connect initially
- PEM file need to have specific permission before you can use it directly. If the permission is not set properly you will see the error “It is required that your private key files are NOT accessible by others. This private key will be ignored.”

```
ansible-playbook main.yml -i inventories/dev/hosts --user ec2-user --key-file ansible_auth.pem -e
'@configs/dev.yml'
```

Now change the permission of the pem file and then re-run the playbook

```
sudo chmod 600 ansible_auth.pem
```

devops user has created successfully and the public key also get copied to the remote servers

- Now try to do the ssh using ec2-user you will still see the “Permission Denied” error, because we have set the devops user for ssh connectivity

- Now try to ssh using devops user

You have successfully setup the ssh key between two servers.

- Once you setup the devops user then you can use the devops key and run the playbook using devops user

```
[root@ip-172-31-22-242 Ansible-Sample-Application-Deployment]#
ansible-playbook main.yml -i inventories/dev/hosts --user devops --key-file /home/devops/.ssh/id_rsa  -e
'@configs/dev.yml'
```

Congratulations, you have successfully Setup SSH between two AWS EC2 instances using Ansible.