Day 5 - AWS CLI - How to connect to EC2 Instance from UI & Terminal

Using UI : Open AWS Console, go to EC2 Dashboard, then click on Instances (Running Instances), click on the running instance ID, click on CONNECT , scroll down and click CONNECT. this is ho wto connect AWS VM using Console UI Successfully.

- **As a role & responsibly of DevOps Engineer, one should open N Number of VM's daily for specific reasons.**
- **to sort this issue, there is a efficient way instaed of 3rd party terminals.**
- **For MAC users iTerm is suggested and for Windows users command propmt, Putty, Mobaxterm terminal(in Mobaxterm we can save upto 10 VM's passwords easily) or NOMACHINE.**

Using CLI (CMD) : Copy the public IP Address from the Instance, use command "ssh ubuntu@<IP Address>" and click enter.

NOTE : at very first time it asks for fingerprint, type "YES", automatically the cionnection is denied. because we doesnt given the RSA key value pair

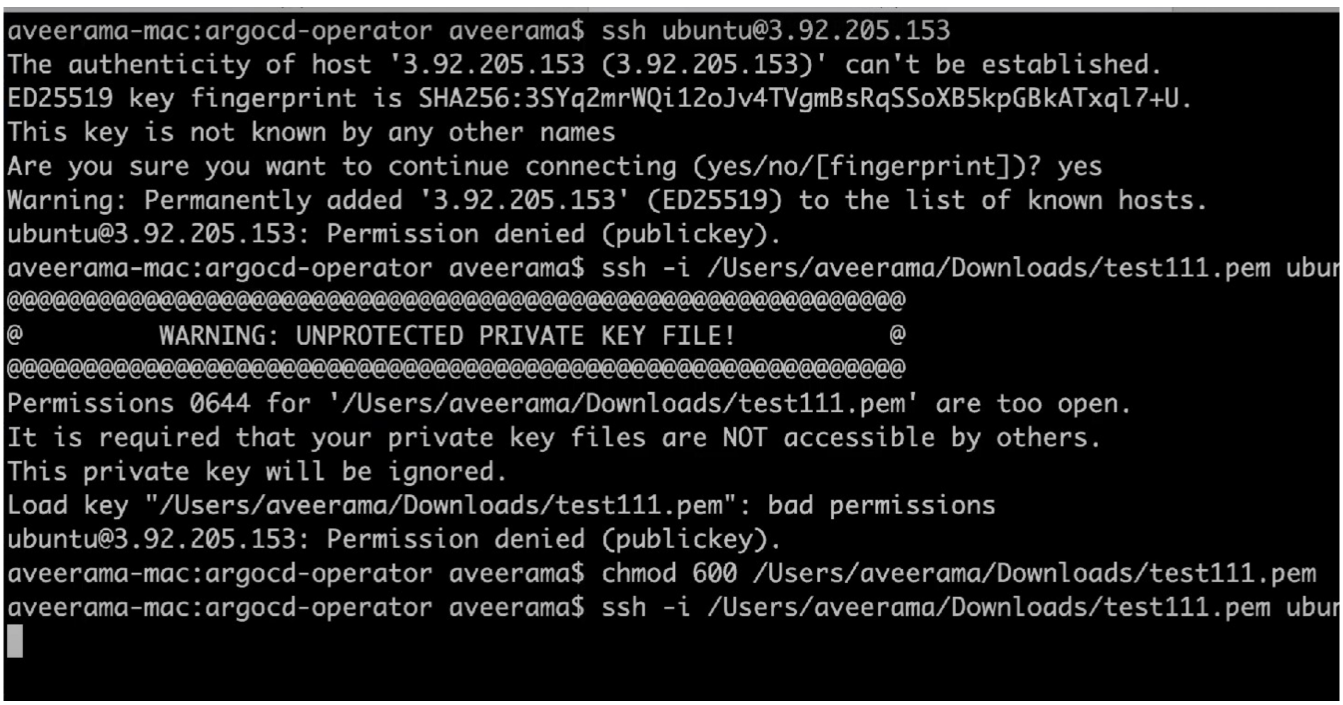
- **So to sort this, use the following command :**
"ssh -i <Identity File Location(Pem file)> ubuntu@<IP Address>"

NOTE : again the VM gonna disconnect, because of the permissions are to open according to the PEM File, its a secret file which has sensitive information. someone can easily open the VM using this PEM File.

- **to sort this use the following commands :**
"chmod 600 <Identity File Location(Pem file)>"   Then use this command

"ssh -i <Identity File Location(Pem file)> ubuntu@<IP Address>"

- **You will be succesfully loggedin through the AWS VM using CLI**



- **Best Practice : Stop the Instance and then Terminate the Instance completely.**

Using AWS CLI : Download AWS CLI Terminal from web and install it

- **check that AWS CLI is installed properly aren't by using the command "aws version"**
- **even though the AWS CLI is installed we cant access anything because we just installed the AWS Binary(check "which aws"), which doesnt have our user information / account information.**
- **so we have to authenticate this, open AWS Console, go to Security Credentials from the profile icon, scroll down, click on create access keys, agree to the risks and terms.(Never share the access keys to anyone)**
- **copy the access keys and store in the secret loaction safely.**
- **go back to CMD, use command "aws configure", enter**
- **paste the AWS Access key , then enter**
- **paste the secret access key, then enter**
- **let it be be the default region-us-east-1 and output format-json**
- **to check that the AWS CLI Terminal is working properly, lets use this command to see the existing S3 buckets "aws s3 ls"(Access key ID required here)**
- **paste the required AWS Access key ID here**
- **paste the required AWS Secret Access key ID here**

• let it be be the default region-us-east-1 and output format-json

• use command "aws s3 ls", where the s3 buckets in the AWS Console UI must match with the AWS CLI Buckets list.

• we can create AWS S3 Bucket, using command "aws s3 mb <unique bucket name>"

• To create AWS Instance, simply navigate to the AWS Documentations and copy the Instance Creation Code and run in the AWS CLI.(which includes image info, type of Instance, key value location, security group and subnet ID)

• we can create AWS Instances very simply.

Using AWS CFT (Cloud Formation Template) : the AWS CFT and Terraform are clusted under the IAC (Infrastructure as Code)

• in GitHub, from awslabs/aws-cloudformation-templates, we have wide range of examples

• click on servies, go to EC2 Service and click on EC2InstanceWithSecurityGroupSample

• its an example, the entire script to create AWS VM through AWS CFT

• to save the entire code, click on CFT, go to Cloud formation, we can create own stack here

Template is ready - used by github code
use sample template- pre built templates are given her by aws
create template in desinger-we can create a customized template

• in another way, you can save the template locally and upload here.

Using Script Automation : using the aws api, one should write script in shell or favorite language. here we are automating python using Boto3(support module).

• Boto3 can pick the information from the configuration files too.

• its a very simple script.

• it has beginner friendly documentation.

Example using boto3 to list running EC2 instances

boto3_listinstances_example.py

```python
import boto3

ec2 = boto3.resource('ec2')

def lambda_handler(event, context):
    # create filter for instances in running state
    filters = [
        {
            'Name': 'instance-state-name',
            'Values': ['running']
        }
    ]

    # filter the instances based on filters() above
    instances = ec2.instances.filter(Filters=filters)

    # instantiate empty array
    RunningInstances = []

    for instance in instances:
        # for each instance, append to array and print instance id
        RunningInstances.append(instance.id)
        print instance.id
```

Assignment 1 : Install AWS CLI package, create security credentials, authenticate anything like terminal, Putty or Mobaxterm. make a call with your AWS

• Create S3 bucket, List S3 Bucket, List EC2 Instances.

• follow AWS Documentation.(Open in any browser - AWS CLI Doc)