Cloud Computing Exercise #22

Launching an EC2 instance from CLI

A. Preparation

1. Sign in to your AWS account as the non-root admin user.
2. In this lab, you will launch an EC2 instance from the CLI. First, you will launch an EC2 instance, the master instance, manually and log into it using ssh. Then, using the CLI of the master instance, you will launch another EC2 instance, the slave EC2 instance. You will retrieve various system and instance properties using the CLI, and finally, terminate the slave EC2 instance.

B. Launch an EC2 instance using the Management Console

1. Before Launching the EC2 Instance, Make sure you have the access keys downloaded on your machine. You should have this if you have followed the previous exercises. If not, go to IAM -> Security Credentials -> and create access keys. Now go to EC2 and launch a new instance using the “Amazon Linux 2023 AMI”, Add a tag with key field “Name” and value field “Master EC2 instance” and the t2.micro instance type. You can use the default 8 GB EBS volume settings and the default security group settings (port 22 open).. Launch the EC2 instance, and connect to it using your local SSH client/Putty (the user name is “ec2-user”). This AMI has the AWS CLI client already installed, so there is no need to manually install it.
2. Make note of the Instance ID, Key pair attached to the instance, and the availability zone where the instance was launched.

C. Configure the AWS CLI

1. SSH into the instance, Start the CLI configuration process by entering the following command on the EC2 ssh console:aws configure

Copy and paste the access key ID and the secret access key when prompted.

**[Note: The commands are written in blue. Any text highlighted in black indicates that you should replace it with your specific IDs and details for your scenario.]**

D. Query the AWS EC2 service

1. aws ec2 describe-instances --instance-ids i-03dd0d585ca8b84d9 --query "Reservations[0].Instances[0].ImageId" --output text



Note down this AMI ID as this will be used later.

1. Use the AWS CLI in your ssh client to list the available VPCs in your default region. Unless you are working on some other projects, you should see only one VPC listed – your default VPC. However, to make the command more robust and easier to create a bash script later, add the following functionality to the EC2 CLI command:

* It should use server-side filtering (--filters option) to retrieve only the default VPC’s information.
* It should use client-side filtering (--query option) to only output the VPC ID of the VPC.
* It should format the output as text (--output option).

You should see the VPC ID of the default VPC printed on the screen (without quotes). Note the VPC ID of the default VPC.

aws ec2 describe-vpcs --filters "Name=isDefault,Values=true" --query "Vpcs[0].VpcId" --output text



1. Use the AWS CLI to list the available subnets in the default VPC identified in the previous step. You should see multiple subnets listed – each belonging to a different availability zone. Add the following functionality to the CLI command:

* It should use server-side filtering (--filters option) to only retrieve the subnet information for the subnet in the us-east-1a/us-east-2a availability zone of the default VPC. The easiest way to do this is to use the –filters option twice: once for the VPC ID of the default VPC and once for the availability zone.
* It should use client-side filtering (--query option) to only output the subnet ID of the subnet.
* It should format the output as text (--output option).

You should see the subnet ID of the subnet printed on the screen (without quotes). Note the subnet ID of this subnet.

aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-058370a0a3813af7b" "Name=availability-zone,Values=us-east-2a" --query "Subnets[0].SubnetId" --output text



E. Create and configure a security group

1. Create a security group using the CLI for the default VPC. Provide a reasonable name and description for the security group. The command will return the security group ID of the newly created security group, so use the text output format (--output option) to print only the security group ID without JSON formatting and quotes. This will be useful when we write a bash script and want to save the value of the security group ID in a variable.

aws ec2 create-security-group --group-name "MySecurityGroup" --description "Security group for SSH access" --vpc-id vpc-058370a0a3813af7b --query "GroupId" --

sg-070a0990d6b3be854



1. Configure the newly created security group to allow incoming TCP traffic on port 22 (SSH) from anywhere (use the 0.0.0.0/0 CIDR IP range).

aws ec2 authorize-security-group-ingress --group-id sg-070a0990d6b3be854 --protocol tcp --port 22 --cidr 0.0.0.0/0

A black screen with white text

Description automatically generated

F. Launch the EC2 instance from the CLI

1. Launch an EC2 instance using the following parameters:
   * Image ID: the Amazon Linux 2 AMI’s ID that you noted previously
   * Key name: the name of the keypair you are using to access the EC2 instances (e.g. mykey)
   * Instance type: t2.micro
   * Security group ID: the ID of the security group you just created
   * Subnet ID: the ID of the subnet you have previously output
   * Block device mapping parameters:
     + Device name: /dev/xvda
     + The EBS volume should be deleted when the EC2 instance is terminated
     + EBS volume size: 10 GBytes
   * Add the following tag: Tag key: “Name”, Tag value: “Slave EC2 instance”
   * Filter the output (--query option) in such a way that only the instance ID of the newly launched instance is displayed (Hint: Look at the unfiltered output and determine the appropriate JMESPath expression. You may have to launch multiple instances to get it right, but that is OK – you can terminate the surplus instances immediately using the Management Console.)
   * Set the output format (--output option) to text.

You should see the instance ID of the new instance being launched displayed on the screen. If you go to the Management Console (EC2/Instances/Instances), you should see the new instance in the list of instances, first in “pending” state, and then in “running state”, tagged with the tag “Slave EC2 instance”.

aws ec2 run-instances \

--image-id ami-0fae88c1e6794aa17 \

--key-name Aditya02 \

--instance-type t2.micro \

--security-group-ids sg-070a0990d6b3be854 \

--subnet-id subnet-0a7ef8c26af2a2b48 \

--block-device-mappings "[{ \"DeviceName\": \"/dev/xvda\", \"Ebs\": { \"VolumeSize\": 10, \"DeleteOnTermination\": true }}]" \

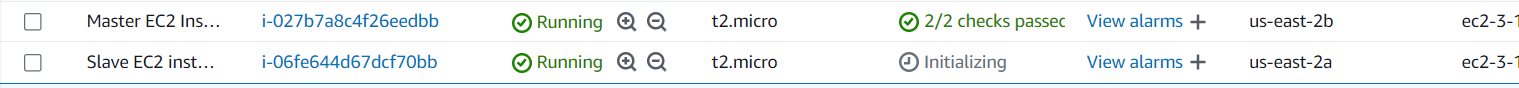
--tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value="Slave EC2 instance"}]' \

--query "Instances[0].InstanceId" \

--output text

A computer code on a black background

Description automatically generated



G. Retrieve information from the running instance

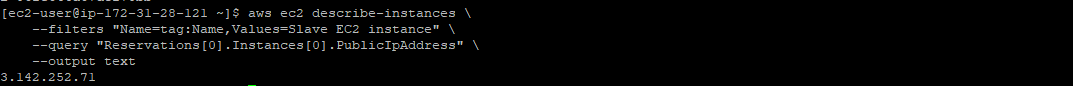
1. Retrieve the properties of your running EC2 instance (the one tagged with the “Slave EC2 instance” tag) using the describe-instances CLI command. Filter the output (--query option) to display only the public IP address of the instance, and use the text output format (--output option). You should see the public IP address of the device displayed on the screen.

aws ec2 describe-instances \

--filters "Name=tag:Name,Values=Slave EC2 instance" \

--query "Reservations[0].Instances[0].PublicIpAddress" \

--output text



H. Terminate the running instance using CLI

1. Terminate your running EC2 instance (the one tagged with the “Slave EC2 instance” tag) from the CLI. If you go to the Management Console (EC2/Instances/Instances), you should see the slave instance transition to the “shutting down” and then the “terminated” state.

aws ec2 describe-instances \

--filters "Name=tag:Name,Values=Slave EC2 instance" \

--query "Reservations[0].Instances[0].InstanceId" \

--output text

aws ec2 terminate-instances \

--instance-ids i-06fe644d67dcf70bb

A screen shot of a computer

Description automatically generated



I. Clean up after yourself

1. Close the ssh window and terminate the master EC2 instance (the one tagged with the “Master Ec2 instance” tag. Delete the security groups you created. Check and make sure that there are no EBS volumes that survived the instance termination operations.
2. Log out of AWS.