Enabling GUI interface on Amazon linux 2 ec2 instance

Perform the below tasks using AWS CLI commands

Tasks :

Create a group and attach below policies

//Policies

- VPC access.
- Full access to IAM.
- Full access to S3.
- Access to CloudWatch.
- Access to SQS (Simple Queue Service).
- Access to Lambda.

//Attaching policies to the existing group
//Policy arn [amazon resource name] for each services

arn:aws:iam::aws:policy/AmazonVPCFullAccess

arn:aws:iam::aws:policy/IAMFullAccess

arn:aws:iam::aws:policy/AmazonS3FullAccess

arn:aws:iam::aws:policy/CloudWatchFullAccess

arn:aws:iam::aws:policy/CloudWatchFullAccess

arn:aws:iam::aws:policy/AWSLambda_FullAccess

Commands:

//Creating group named Cloud_shell_deployment

Commands:

aws iam create-group --group-name Cloud_shell_deployment

aws iam attach-group-policies --group-name Cloud_shell_deployment --policy-arn arn:aws:iam::aws:policy/AmazonVPCFullAccess

//repeat the above process for rest of the services that you want to give access

Creating a user and adding to the existing group

//User-creation commands

aws iam create-user --username username

//giving the user a password

aws iam create-login-profile --user-name username --password YourPassword

//Adding user to existing group

aws add-user-to-group --user-name username --group-name Cloud_shell_deployment

[Optional]

Giving the user a programmatic access , only if the user want to manage AWS via CLI

//Note: we need to create access keys and make sure to note down both **YourAccessKeyId** and **YourSecretAccessKey**

Command:

aws iam create-access-key --user-name username

//In order to perform the tasks as user you have created in aws via cli , you need to configure aws cli

Command:

aws configure

//give your **YourAccessKeyId** and **YourSecretAccessKey**

//Now you have entered into aws service as user you have created

//Creating custom VPC

Commands:

aws ec2 create-vpc --cidr-block 10.0.0.0/16 --region us-east-1

//creating network access control list [NACL] which can be filter traffic at network level

commands:

```
aws ec2 create-network-acl --vpc-id givevpcid --network-acl-name youraclname
```

//Editing inbound and outbound rule for NACL [optional, if you want to create HTTPs repeat the process by changing the port]

```
# Create inbound rule for HTTP (port 80)
aws ec2 create-network-acl-entry \
  --network-acl-id your-nacl-id \
  --rule-number 100 \
  --protocol tcp \
  --rule-action allow \
  --egress false \
  --cidr-block 0.0.0.0/0 \
  --port-range From=80,To=80
```

Create a default outbound rule allowing all traffic

commands:

```
aws ec2 create-network-acl-entry \
  --network-acl-id your-nacl-id \
  --rule-number 100 \
  --protocol -1 \
  --rule-action allow \
  --egress true \
  --cidr-block 0.0.0.0/0
```

Create a default inbound rule allowing all traffic

//windows terminal we use **^** for line continuation

```
aws ec2 create-network-acl-entry \
  --network-acl-id your-nacl-id \
  --rule-number 100 \
  --protocol -1 \
  --rule-action allow \
```

--egress false \

--cidr-block 0.0.0.0/0

//create a private and public subnet

commands:

aws ec2 create-subnet --vpc-id givevpc-id --cidr-block 10.0.1.0/24, Name publicsubnet_01

aws ec2 create-subnet --vpc-id givevpc-id --cidr-block 10.0.2.0/24 , Name privatesubnet_01

Now we will associate the NACL with public subnet

commands:

aws ec2 associate-network-acl-subnet \

--subnet-id your-public-subnet-id \

--network-acl-id your-nacl-id

//Creating public and private route tables

//private routetable:

command:

aws ec2 create-route-table --vpc-id givevpc-id --region us-east-1

//associate with private subnet

command:

aws ec2 associate-route-table --subnet-id givesubnetid --route-table-id giveroutetableid --region us-east-1

//public routetable:

command:

aws ec2 create-route-table --vpc-id givevpc-id --region us-east-1

//associate with public subnet

command:

aws ec2 associate-route-table --subnet-id givesubnetid --route-table-id giveroutetableid --region us-east-1

//create an Internet gateway for public subnet and NAT gateway for private subnet

//Internet gateway creation

command:

aws ec2 create-internet-gateway --region us-east-1

//Now we need to attach it with public route table

commands:

aws ec2 create-route --route-table givepublicroutetableid --destination-cidr-block 0.0.0.0/0 --gateway-id giveinternetgatewayid

//NATgateway creation

//for NAT gateway we need to manually define elastic IP

//Creation of elastic IP

aws ec2 allocate-address --region us-east-1

//creation of NATgateway and attaching to privatesubnet

aws ec2 create-nat-gateway --subnet-id giveprivatesubnetid --allocation-id giveelasticip --region us-east-1

//Now we need to attach with private route table

aws ec2 create-route --route-table giveprivateroutetableid --destination-cidr-block 0.0.0.0/0 target:nat-gateway-id givenatgatewayid

//Creating security groups

//we need to create 1 public security groups for public subnet and 1 private security group for private subnet

//Public security group creation

commands:

aws ec2 create-security-group --group-name publicsg_01 --vpc-id --region us-east-1

//Private security group creation

aws ec2 create-security-group --group-name privatesg_01 --vpc-id --region us-east-1

//Now we need to edit the inbound rules for security groups

//Editing the rules for public security groups

//For public security group we need to allow below ports

- HTTP port 80
- HTTPS port 443
- RDP port 3389
- SSH port 22

//note: inbound rules also called ingress in networking level

//commands:

//for HTTP

aws ec2 authorize-security-group-ingress --group-id givepublicsgname --protocol tcp --port 80 --cidr-block 0.0.0.0/0

//for HTTPS

aws ec2 authorize-security-group-ingress --group-id givepublicsgname --protocol tcp --port 443 --cidr-block 0.0.0.0/0

//for RDP

aws ec2 authorize-security-group-ingress --group-id givepublicsgname --protocol tcp --port 3389 --cidr-block 0.0.0.0/0

//for SSH

aws ec2 authorize-security-group-ingress --group-id givepublicsgname --protocol tcp --port 22 --cidr-block 0.0.0.0/0

Now we will launch Amazon Linux 2 bastion [all public]

IDs of services:

VPC id: vpc-03992d62a403f97de
NACLid: acl-03eb561edb6b20f57
pubsubnet-id: subnet-040bf3ab0b762ab94
privatesubid: subnet-093dd2c47afd7b5a3
pubsg: "GroupId": " sg-0bd8036c9a4601f9a "
privatesg: "GroupId": "sg-0493d6e49a74b87f6"
privatert ID: rtb-01ec55e05c6eede3b
publicrt id: rtb-0226fbbb1a07837d6
igw id: "igw-0a997dd59c3852da3"
elastic ip: "eipalloc-0fe7f992fd01ad3a5"
natgatway: nat-0e7f8f797201af9a9
keyname: virginiaroot
instance type: t2.micro

Amazon linux 2023 : ami-067d1e60475437da2

commands:

```
aws ec2 run-instances\
--image-id {imageid}\
--instance-type {instancetype}\
--key-name{keyname}\
--subnet-id{pubsubnetid}\
--security-group-ids{pubsg}\
--region us-east-1 \
--associate-public-ip-address     //allocating public ip for instance
```

//windows terminal we use ^ for line continuation

```
aws ec2 run-instances ^
--image-id ami-005b11f8b84489615 ^
--instance-type t2.micro ^
--key-name virginiaroot ^
--subnet-id subnet-040bf3ab0b762ab94 ^
--security-group-ids sg-0bd8036c9a4601f9a ^
--associate-public-ip-address ^
--region us-east-1 ^
```

To get the running instances details

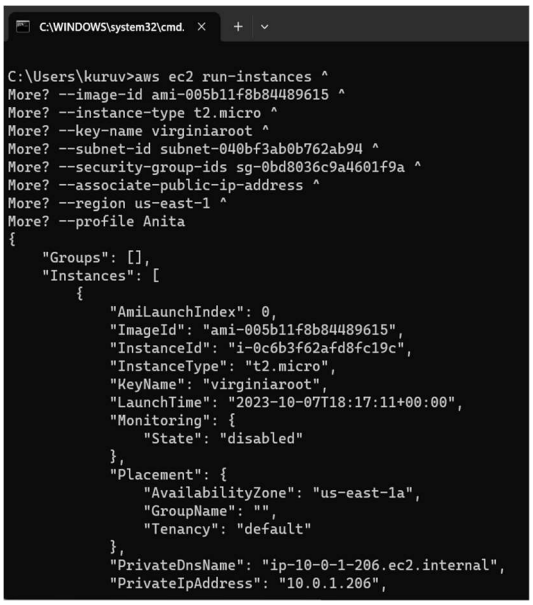
aws ec2 describe-instances --filters "Name=instance-state-name, Values= running" --region us-east-1

To get the short details

aws ec2 describe-instance-status --region us-east-1

//bash terminal

```
aws ec2 run-instances \
--image-id ami-005b11f8b84489615 \
--instance-type t2.micro \
--key-name virginiaroot \
--subnet-id subnet-040bf3ab0b762ab94 \
--security-group-ids sg-0bd8036c9a4601f9a \
--associate-public-ip-address \
--region us-east-1 \
--profile Anita
```



//Now we will enter the Linux kernel and start the process for enabling desktop GUI to make it OS

//Updating the kernel

commands:

yum clean all -y

yum update -y

yum upgrade -y

//now we will install the desktop GUI environment

commands:

yum group list [To list the available environments]

yum groupinstall “server with GUI” [for installing default environment] or we can use below command to install desired environment

yum groupinstall “Mate-desktop-environment”

//Creating a user in Linux with administrative privileges

commands:

useradd -m -s /bin/bash Anita *[-m: This option tells the system to create a home directory for the new user. and -s is to make bash as default shell]*

cat /etc/passwd | cut -d: -f1 *[To get all users]*

cat /etc/passwd | cut -d: -f1 | grep username *[To search for particular username]*

passwd Anita *[adding password ‘anita@123’]*

usermod -aG wheel Anita *[To add user to the current group ‘wheel ‘ -aG: These options are used to add a user to a group.*

wheel: This is the name of the group to which the user is being added, the "wheel" group often has administrative privileges.

<username>: This should be replaced with the actual username of the user you want to add to the "wheel" group.]

Amazon Linux 2 have pre install RDP services , now we connect with RDP tool to our public instance linux machine.

the default RDP service for amazonlinux2 is xvnc service

//We just only need to change the default booting process from multi-user to GUI and reboot

commands:

sudo systemctl get-default [*To get the default booting target]*

sudo systemctl list-units --type target *[To see the list of booting targets]*

sudo systemctl start graphical.target *[To load and activate the graphical target]*

systemctl set-default graphical.target *[To make the booting as default GUI]*

sudo systemctl reboot *[To reboot to reflect the settings]*

//connecting linux via RDP tool

open RDP tool and connect using public ip, username and password

GUI Desktop