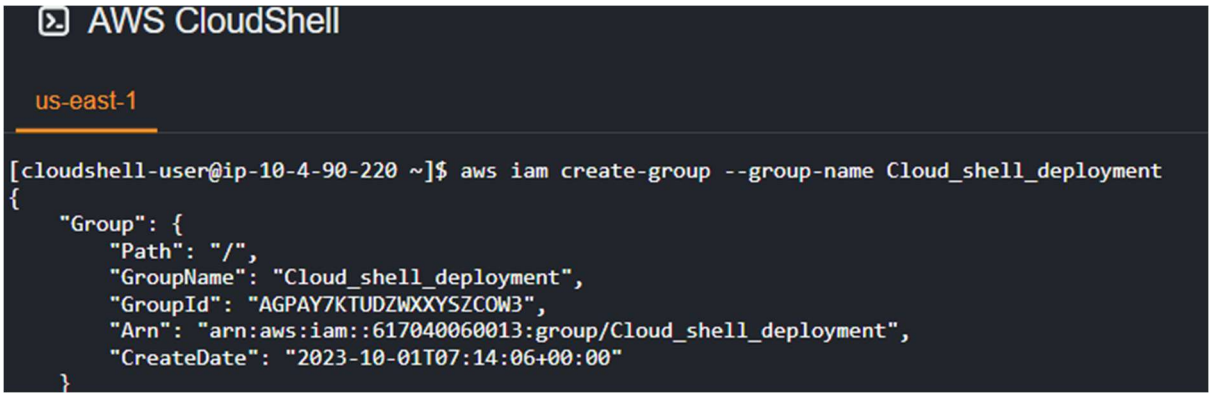
IDs of services:

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

Perform the below tasks using AWS CLI commands

Tasks :

1. **Create an IAM User Group: 'cloud_shell_deployment'**
   $aws iam create-group --group-name Cloud_shell_deployment



- Establish an Identity and Access Management (IAM) user group named 'cloud_shell_deployment' with the following permissions:

- VPC access.

- Full access to IAM.

- Full access to S3.

- Access to CloudWatch.

- Access to SQS (Simple Queue Service).

- Access to Lambda.

Command to add one by one :

```
}
[cloudshell-user@ip-10-4-90-220 ~]$ aws iam attach-group-policy --policy-arn arn:aws:iam:policy/AmazonVPCFullAccess --group-name Cloud_shell_deployment
```

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

Command to add all at a time :

```
[cloudshell-user@ip-10-4-90-220 ~]$ {
> aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/IAMFullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess --group-name Cloud
_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/CloudWatchFullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonSQSFull
Access --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AWSLambda_FullAccess --group-name Cloud_shell_deployment;
> }
[cloudshell-user@ip-10-4-90-220 ~]$
CloudShell   Feedback   © 2023, Amazon Web Services India Private Limited or its affiliates.   Privacy   Terms   Cookie preference
```

${

aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/IAMFullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/CloudWatchFullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonSQSFullAccess --group-name Cloud_shell_deployment; aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AWSLambda_FullAccess --group-name Cloud_shell_deployment;
}

2. **Create an IAM User and Add to the 'cloud_shell_deployment' Group**

- Create an IAM user and include them in the 'cloud_shell_deployment' IAM group with the appropriate permissions.

$aws iam add-user-to-group --user-name sanju --group-name Cloud_shell_deployment

```
}
[cloudshell-user@ip-10-4-90-220 ~]$ aws iam add-user-to-group --user-name sanju --group-name Cloud_shell_deployment
```

3. Create AWS VPC at useast-1 *[note down VPC ID]*

- VPC features: *["VpcId": "vpc-03992d62a403f97de"]*

a. The AWS VPC should be at useast-1 (N.Virginia) with CIDR rage : 10.0.0.0/16

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

4. 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
```

5. create 1 private and 1 public subnet

- Subnet features

a. Public subnet should contain CIDR range 10.0.1.0/24, Name Publicsubnet_01 *[SubnetId": "subnet-040bf3ab0b762ab94]*

$aws ec2 create-subnet --vpc-id *vpc-03992d62a403f97de* --cidr-block 10.0.1.0/24 --availability-zone us-east-1a --region us-east-1

```
[cloudshell-user@ip-10-2-2-89 ~]$ aws ec2 create-subnet --vpc-id vpc-03992d62a403f97de --cidr-block 10.0.1.0/24 --availability-zone us-east-1a --region us-east-1
{
    "Subnet": {
        "AvailabilityZone": "us-east-1a",
        "AvailabilityZoneId": "use1-az4",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.1.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-040bf3ab0b762ab94",
        "VpcId": "vpc-03992d62a403f97de",
        "OwnerId": "617040060013",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": []
```

b. Private subnet should contain CIDR range 10.0.2.0/24, Name privatesubnet_01 *[SubnetId": "subnet-093dd2c47afd7b5a3]*

$aws ec2 create-subnet --vpc-id *vpc-03992d62a403f97de* --cidr-block 10.0.2.0/24 --availability-zone us-east-1b --region us-east-1

```
[cloudshell-user@ip-10-2-2-89 ~]$ aws ec2 create-subnet --vpc-id vpc-03992d62a403f97de --cidr-block 10.0.2.0/24 --availability-zone us-east-1b --region us-east-1
{
    "Subnet": {
        "AvailabilityZone": "us-east-1b",
        "AvailabilityZoneId": "use1-az6",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "10.0.2.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-093dd2c47afd7b5a3",
        "VpcId": "vpc-03992d62a403f97de",
        "OwnerId": "617040060013",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "SubnetArn": "arn:aws:ec2:us-east-1:617040060013:subnet/subnet-093dd2c47afd7b5a3",
```

6. create 1 public security group and 1 private security group

- Securitygroup features:
  - *a.* Public security group name = “Pubsg_01” *["GroupId": "sg-0bd8036c9a4601f9a"]*
    $aws ec2 create-security-group --group-name Pubsg_01 --description "Public Security Group" --vpc-id *vpc-03992d62a403f97de* --region us-east-1

```
AWS CloudShell
us-east-1 ✕   us-east-1 ✕   us-east-1 ✕
cloudshell-user@ip-10-2-2-89 ~]$ $aws ec2 create-security-group --group-name Pubsg_01 --description "Public Security Group" --vpc-id vpc-03992d62a403f97de --region us-east-1
bash: ec2: command not found
cloudshell-user@ip-10-2-2-89 ~]$ aws ec2 create-security-group --group-name Pubsg_01 --description "Public Security Group" --vpc-id vpc-03992d62a403f97de --region us-east-1
    "GroupId": "sg-0bd8036c9a4601f9a"
cloudshell-user@ip-10-2-2-89 ~]$
```

  - b. Private security group name = “privatesg_01” *["GroupId": "sg-0ca85264b61df3ded"]*
    $aws ec2 create-security-group --group-name privatesg_01 --description "Private Security Group" --vpc-id *vpc-03992d62a403f97de* --region us-east-1

```
}
[cloudshell-user@ip-10-2-2-89 ~]$ aws ec2 create-security-group --group-name privatesg_01 --description "Private Security Group" --vpc-id vpc-03992d62a403f97de --region us-east-1
{
    "GroupId": "sg-0ca85264b61df3ded"
}
[cloudshell-user@ip-10-2-2-89 ~]$
```

  - c. The “Pubsg_01” and “privatesg_01” security group should contain inbound rules that allows Ports for HTTP , HTTPS, RDP and Oracle database for everywhere from IPv4.
    Pubsg_01:

#Note: repeat the below commands individually and refresh the inbound rules to refelect at AWS Console

$aws ec2 authorize-security-group-ingress --group-id *sg-0bd8036c9a4601f9a* --protocol tcp --port 80 --cidr 0.0.0.0/0
$aws ec2 authorize-security-group-ingress --group-id *sg-0bd8036c9a4601f9a* --protocol tcp --port 443 --cidr 0.0.0.0/0
$aws ec2 authorize-security-group-ingress --group-id *sg-0bd8036c9a4601f9a* --protocol tcp --port 3389 --cidr 0.0.0.0/0
$aws ec2 authorize-security-group-ingress --group-id *sg-0bd8036c9a4601f9a* --protocol tcp --port 1521 --cidr 0.0.0.0/0
$aws ec2 authorize-security-group-ingress --group-id *sg-0bd8036c9a4601f9a* --protocol tcp --port 22 --cidr 0.0.0.0/0

```
[cloudshell-user@ip-10-2-2-89 ~]$ aws ec2 authorize-security-group-ingress --group-id sg-0bd8036c9a4601f9a --protocol tcp --port 1521 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0ae8693b9c0211fd3",
            "GroupId": "sg-0bd8036c9a4601f9a",
            "GroupOwnerId": "617040060013",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 1521,
            "ToPort": 1521,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Privatesg_01:
$aws ec2 create-security-group --group-name privatesg_01 --description "Private Security Group" --vpc-id <YOUR_VPC_ID> --region us-east-1

#Please replace **<YOUR_VPC_ID>**, **<PUBSG_01_GROUP_ID>**, and **<PRIVATESG_01_GROUP_ID>** with the appropriate values or IDs.

7. **Create 1 Private Route Table** *[rtb-01ec55e05c6eede3b]*

- Configure a dedicated private route table for the private subnet named Privatert_01.

$aws ec2 create-route-table --vpc-id *vpc-03992d62a403f97de* --region us-east-1

```
[cloudshell-user@ip-10-4-33-4 ~]$ aws ec2 create-route-table --vpc-id vpc-03992d62a403f97de --region us-east-1
{
    "RouteTable": {
        "Associations": [],
        "PropagatingVgws": [],
        "RouteTableId": "rtb-01ec55e05c6eede3b",
        "Routes": [
            {
                "DestinationCidrBlock": "10.0.0.0/16",
                "GatewayId": "local",
                "Origin": "CreateRouteTable",
                "State": "active"
            }
```

a. Associating private subnet with private route table

$aws ec2 associate-route-table --subnet-id subnet-093dd2c47afd7b5a3 --route-table-id rtb-01ec55e05c6eede3b --region us-east-1

```
[cloudshell-user@ip-10-4-33-4 ~]$ aws ec2 associate-route-table --subnet-id subnet-093dd2c47afd7b5a3 --route-table-id rtb-01ec55e05c6eede3b --region us-east-1
{
    "AssociationId": "rtbassoc-0a21dbe5736235fa5",
    "AssociationState": {
        "State": "associated"
    }
}
```

8. **Create 1 Public Route Table** *[ID: rtb-0226fbbb1a07837d6]*

- Configure a dedicated public route table for the public subnet named Publicrt_01.

$aws ec2 create-route-table --vpc-id *vpc-03992d62a403f97de* --region us-east-1

a. Associating public subnet with public route table

$aws ec2 associate-route-table --subnet-id subnet-040bf3ab0b762ab94 --route-table-id rtb-0226fbbb1a07837d6 --region us-east-1

```
[cloudshell-user@ip-10-4-33-4 ~]$ aws ec2 associate-route-table --subnet-id subnet-040bf3ab0b762ab94 --route-table-id rtb-0226fbbb1a07837d6 --region us-east-1
{
    "AssociationId": "rtbassoc-05f9d06c9aba252f4",
    "AssociationState": {
        "State": "associated"
    }
}
```

9. **Create 1 Internet Gateway with Public IP for Public Subnet “Publicsubnet_01” and Associate it with Public Route Table “Publicrt_01”** *[igw id: "igw-0a997dd59c3852da3"]*

- Create an Internet Gateway with a public IP for the public subnet and associate it with the public route table.

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

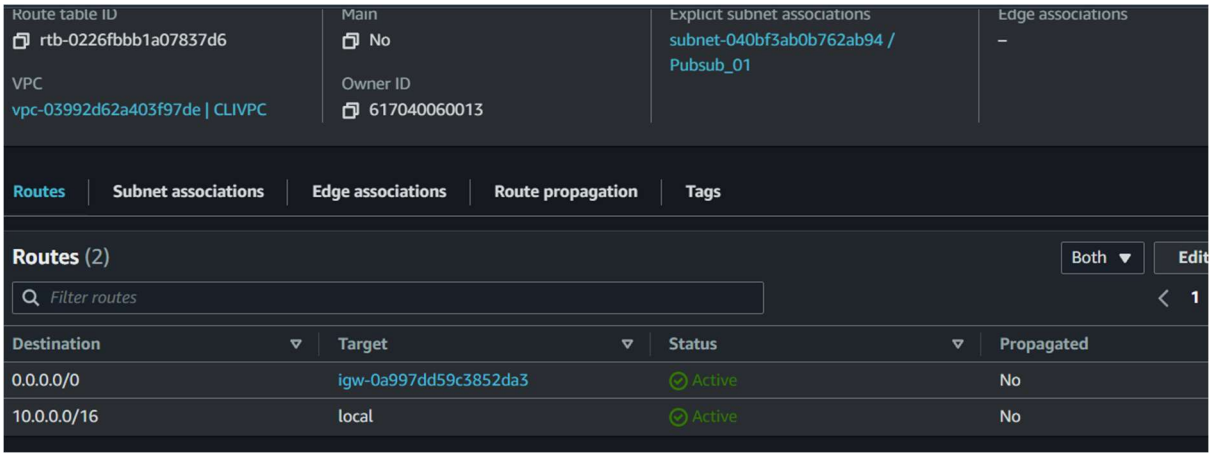
```
[cloudshell-user@ip-10-4-33-4 ~]$ aws ec2 create-internet-gateway --region us-east-1
{
    "InternetGateway": {
        "Attachments": [],
        "InternetGatewayId": "igw-0a997dd59c3852da3",
        "OwnerId": "617040060013",
        "Tags": []
    }
}
```

Edit the public routable routes and attach the internetgateway with destination: 0.0.0.0/0 target : internetgateway

$ aws ec2 create-route --route-table-id *rtb-0226fbbb1a07837d6* --destination-cidr-block 0.0.0.0/0 --gateway-id *igw-0a997dd59c3852da3*

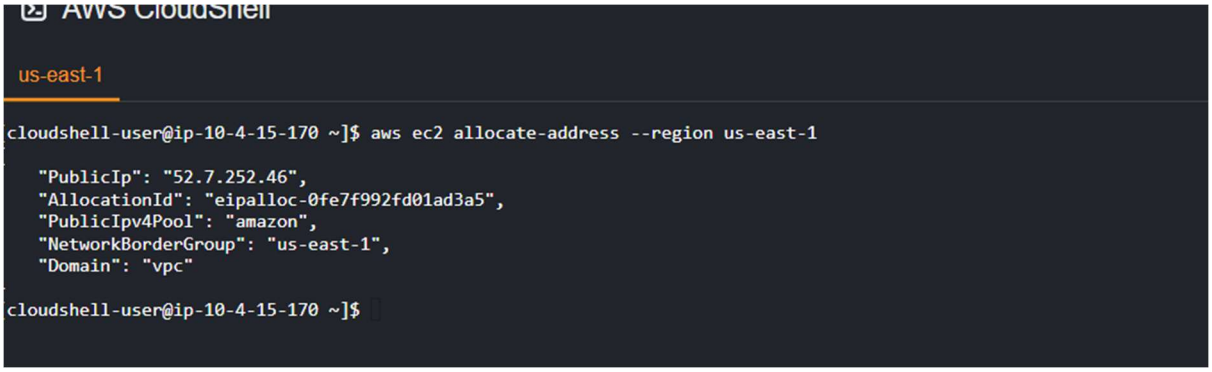
```
cloudshell-user@ip-10-4-15-170 ~]$ aws ec2 create-route --route-table-id rtb-0226fbbb1a07837d6 --destination-cidr-block 0.0.0.0/0 --gateway-id igw-0a997dd59c3852da3

  "Return": true
```



10. Create an Elastic IP for NAT Gateway *[elastic ip: "eipalloc-0fe7f992fd01ad3a5"]*

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



11. **Create 1 NAT Gateway for private subnet and Associate it with elastic IP

- Deploy a Network Address Translation (NAT) gateway and associate it with the private route table for the private subnet.

$aws ec2 create-nat-gateway --subnet-id *subnet-093dd2c47afd7b5a3* --allocation-id *eipalloc-0fe7f992fd01ad3a5* --region us-east-1 *[natgatewayid: nat-0e7f8f797201af9a9 #ignore id on image]*

Edit the private routable routes and attach the natgateway with destination: 0.0.0.0/0 target : natgateway

$ aws ec2 create-route --route-table-id *rtb-01ec55e05c6eede3b* --destination-cidr-block 0.0.0.0/0 --nat-gateway-id *nat-0e7f8f797201af9a9*

```
[cloudshell-user@ip-10-4-15-170 ~]$ aws ec2 create-route --route-table-id rtb-01ec55e05c6eede3b --destination-cidr-block 0.0.0.0/0 --nat-gateway-id nat-0e7f8f797201af9a9
{
    "Return": true
}
[cloudshell-user@ip-10-4-15-170 ~]$
```

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-01ec55e05c6eede3b	No	subnet-093dd2c47afd7b5a3 / prisub_01	–
VPC	Owner ID		
vpc-03992d62a403f97de | CLIVPC	617040060013		

Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2) — Both — Edit routes

Filter routes

Destination	Target	Status	Propagated
0.0.0.0/0	nat-0e7f8f797201af9a9	Active	No
10.0.0.0/16	local	Active	No

12. **Create EC2 Instances at Public subnet “Publicsubnet_01”**

- a. redhat linux (RHEL 9)
- b. Rockylinux
- c. CentOS

RHEL9 Bastion server :
// Replace these placeholders with your actual values

ImageId=ami-026ebd4cfe2c043b2

InstanceType=t2.micro

KeyName=virginiaroot

SubnetId=subnet-040bf3ab0b762ab94

VpcId=vpc-03992d62a403f97de

SecurityGroupId=sg-0bd8036c9a4601f9a

//Create the EC2 instance *[id: i-0b4306c81783f625a]*

```
$aws ec2 run-instances \
  --image-id ami-026ebd4cfe2c043b2 \
  --instance-type t2.micro \
  --key-name virginiaroot \
  --vcp-id  vpc-03992d62a403f97de \
  --subnet-id subnet-040bf3ab0b762ab94 \
  --security-group-ids sg-0bd8036c9a4601f9a \
```

--associate-public-ip-address

```
AWS CloudShell
us-east-1
[cloudshell-user@ip-10-4-15-170 ~]$ aws ec2 run-instances \
>     --image-id ami-026ebd4cfe2c043b2 \
>     --instance-type t2.micro \
>     --key-name virginiaroot \
>     --subnet-id subnet-040bf3ab0b762ab94 \
>     --security-group-ids sg-0bd8036c9a4601f9a \
>     --associate-public-ip-address
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-026ebd4cfe2c043b2",
            "InstanceId": "i-0b4306c81783f625a",
            "InstanceType": "t2.micro",
:...skipping...
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-026ebd4cfe2c043b2",
            "InstanceId": "i-0b4306c81783f625a",
            "InstanceType": "t2.micro",
            "KeyName": "virginiaroot",
            "LaunchTime": "2023-10-01T12:29:34+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-10-0-1-70.ec2.internal",
CloudShell    Feedback
```

Instances (1/2) Info | Connect | Instance state | Actions | Launch instances

Find instance by attribute or tag (case-sensitive)

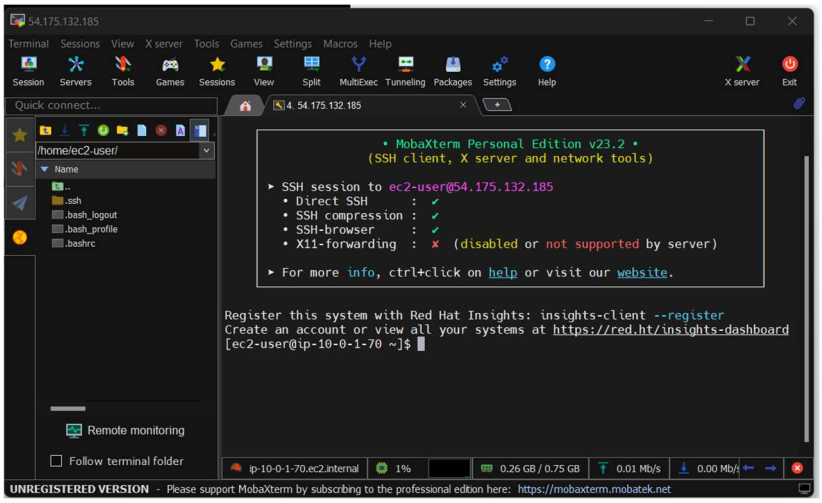
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
RHEl9CLI	i-0b4306c81783f625a	Running	t2.micro	Initializing	No alarms	us-east-1a	–
Bastion amzl2	i-0c0a53eab3df7760b	Stopped	t2.micro	–	No alarms	us-east-1b	–

Unable to connect via EC2 instance connect:

Failed to connect to your instance
Error establishing SSH connection to your instance. Try again later.

SSH Client: IP: 54.175.132.185, ssh i “virginiaroot.pem” ec2-user@54.175.132.185

Mobaxterm:



13. **Install Packages Related to Desktop or MATE Package/Distro** *[somehow for RHEL9 Mate desktop is not available go with default GUI "server with GUI" [GNOME]]*

- Install packages relevant to the desktop or MATE package/distribution for RHEL.

$ sudo su –

You need to register with subscription manager or make sure u have subscribed to redhat to perform below commands

subscription-manager register *[Provide username and password]*

#dnf clean all *[To clean all unreleated or broken installation files and dependencies]*

#dnf update -y *[To update the services and depenencies]*

14. Adding user and adding the user to wheel group

//We need to add user and make bash as default for the user

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

passwd sanju *[adding password 'sanju@123']*

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

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

usermod -aG wheel *Sanju* *[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.]

15. Installing the services required for enabling RDP connection

//Try to start xrdp service and tigervnc-server service

#systemctl start xrdp.service *[To start xrdp service]*

#systemctl status xrdp.service *[To get status of xrdp service]*

If above both commands are throwing error saying the service not found, try below commands

#dnf install epel-release *[epel is the extra package for redhat where rdp services stored]*

//if the above commands throw error , try below command

#dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm *[we are installing epel-release package manually]*

#rpm -ql epel-release *[To find if package available in our instance]*

// now if the package is available continue the below steps to install xrdp and tigervnc server.

#dnf install xrdp tigervnc-server -y *[Installing the xrdp and tigervnc server]*

#systemctl start xrdp.service *[To start xrdp service]*

#systemctl status xrdp.service *[To get status of xrdp service]*

#systemctl enable xrdp.service *[To start the service whenever the systems boots up [syslink]]*

//Enable and open port 3389 at firewall in linux server

#systemctl start firewalld.service *[Starting the firewall service]*

#systemctl status firewalld.service *[To get the status of firewall]*

#systemctl enable firewalld.service *[To start the service whenever the systems boots up]*

#firewall-cmd --permanent --add-port=3389/tcp *[Adding custom port 3389 as TCP protocol]*

#firewall-cmd –reload *[restarting the firewall]*

16. **Enable GUI Package for RHEL9**

- Configure the Graphical User Interface (GUI) package for the Linux environment.

#dnf groupinstall "server with GUI" -y *[To install default GUI for RHEL9]*

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

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

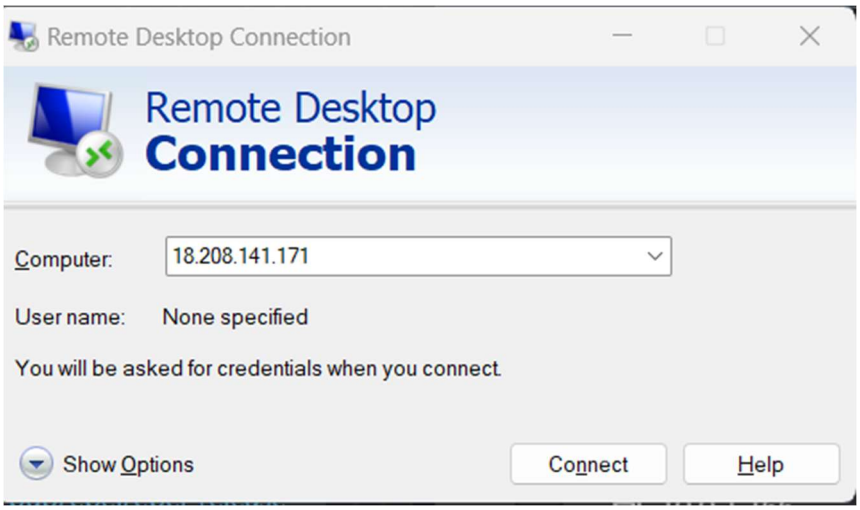
sudo systemctl start graphical.target *[To load and activate the graphical target]*

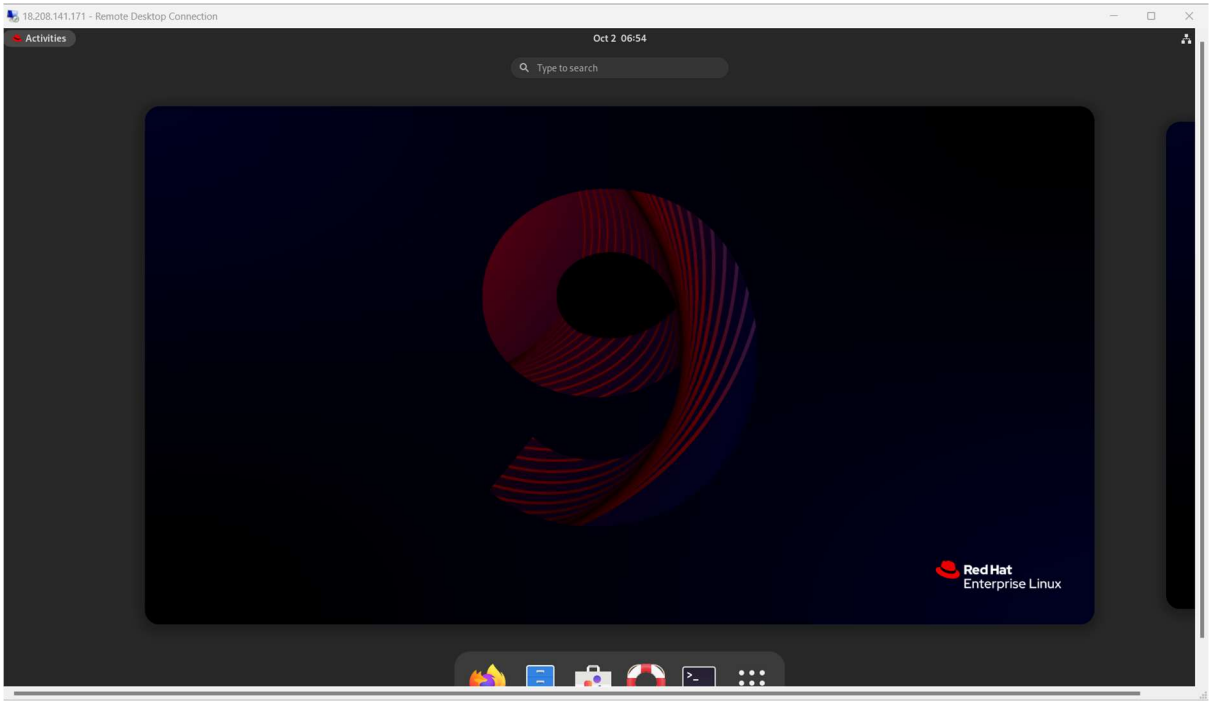
systemctl set-default graphical.target *[To make the booting as default GUI]*

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

17. RDP connection to the RHEL 9 server as GUI.

Now open RDP tool on windows and provide public IPv4 address of instance and login using the user you have created by providing username and password *[In my case username : Sanju, wait for sometime it loads very slowly]*




18.208.141.171 - Remote Desktop Connection
Activities
Oct 2 06:54
Type to search
Red Hat
Enterprise Linux