



PRÁCTICA 7 AWS II

Paloma Pérez de Madrid

ÍNDICE

- 3. Instalación del cliente AWS CLI
- 4. Amazon EC2 (Elastic Computing Cloud)
- 5. Amazon VPC (Virtual Private Cloud)
- 6. Boto3. AWS SDK para Python

ÍNDICE

- 3. Instalación del cliente AWS CLI**
- 4. Amazon EC2 (Elastic Computing Cloud)
- 5. Amazon VPC (Virtual Private Cloud)
- 6. Boto3. AWS SDK para Python

3. INSTALACIÓN DEL CLIENTE AWS CLI

A) Instalar la última versión de la interfaz de línea de comandos (CLI) de AWS en un sistema Linux.

```
Warning: -FILE- to save to a file.  
ubuntupaloma@ubuntu:~/Desktop/aws$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awsxliv2.zip"  
% Total    % Received % Xferd  Average Speed   Time     Time      Current  
          Dload  Upload Total   Spent    Left  Speed  
100 57.6M  100 57.6M    0     0  6126k      0  0:00:09  0:00:09  --:--:-- 6115k
```

```
ubuntupaloma@ubuntu:~/Desktop/aws$ sudo ./aws/install  
You can now run: /usr/local/bin/aws --version  
ubuntupaloma@ubuntu:~/Desktop/aws$
```

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws --version  
aws-cli/2.15.40 Python/3.11.8 Linux/6.5.0-27-generic exe/x86_64/ubuntu.22 prompt/off  
ubuntupaloma@ubuntu:~/Desktop/aws$ which aws  
/usr/local/bin/aws
```

3. INSTALACIÓN DEL CLIENTE AWS CLI

- A) Instalar la última versión de la interfaz de línea de comandos (CLI) de AWS en un sistema Linux.
- B) Asociar las credenciales de un usuario programático a la configuración de AWS CLI. Establecer la región AWS de trabajo como predeterminada y json como formato de salida (archivos credentials y config).
- C) Verificar que se tiene acceso a la infraestructura AWS con el cliente configurado (por ejemplo, listar las zonas de disponibilidad de la región configurada. Nota: En caso de detectar problemas de autentificación, comprobar en primer lugar que la fecha y hora del equipo cliente están sincronizadas (para la sincronización puede usar el protocolo ntp) o el comando date -s “aaaa-mm-dd hh:mm”. Nota: Si se utiliza una cuenta de AWS Academy, es necesario añadir el aws_session_token al archivo credentials (las credenciales pueden obtenerse en AWS Details).

3. INSTALACIÓN DEL CLIENTE AWS CLI

B) Asociar las credenciales de un usuario programático a la configuración de AWS CLI. Establecer la región AWS de trabajo como predeterminada y json como formato de salida (archivos credentials y config).

Entramos en el Lab de academy :

The screenshot shows the AWS Academy Lab interface. At the top, there are buttons for 'Start Lab' (disabled), 'End Lab' (disabled), 'AWS Details' (selected), 'Readme', and 'Reset'. Below this, under 'AWS CLI:', there is a 'Show' button followed by a long horizontal arrow pointing right. A table below lists 'AWSAccountId' as 958486390477 and 'Region' as us-east-1.

AWSAccountId	958486390477
Region	us-east-1

Copy and paste the following into ~/.aws/credentials

```
[default]
aws_access_key_id=ASIA56KR5U3G2YIKK5UM
aws_secret_access_key=xN98kb66pqOOTkvbWKy7SFyN3D+PH6wgDUbjQ9rw
aws_session_token=lQoJb3JpZ2luX2VjEA8aCXVzLXdIc3QtMiJGMEQCIHCFNSi
pb2MmbMtj8J4IiyqevxWg8uFDMmRS3WTvr+/aAiAuR/HxNBmeaX9qohxPI3tY
kulFC12xj8ZxTvXuPdWXCiq5AghXEAAaDDk1ODQ4NjM5MDQ3NyIMonKX+Sfp
XnZ+zs5OKpYCzBsQZmZhneZBOfRGqHcmJMWY/r77b1rtAxNpm6lY8AaaQq/H
h4wH1ZYhPSnsI/gxOch59cB1ZXtQIFESH6FCBLRd4ljS9+5OuC/YfTr98tqJx++66
2uACUWJfUk8lljQRPP5l7XtsGUcH2Q6btJEftoidAx1Fz4UihsI4So0wALP+Tx9a
V+IbPggUtgfTP4GF+Ib7S2YC97gY46k/JDDbExO5G3/ZP97ThvCjJkoT1jqrxDEB/T
jfwrPBoxEBh3+dFqzFXZqvLeI2waYjhKocz02BNqD+ba7aX30AqtY/wHQSaPFzp
EIV4LKyuV3nQ+UodnCGleHNZCTXVyVDZanygnGdRxzdj3P/SNIFYKpj9S91wWS
1Yw6IOYsQY6ngGcOyyvipTnh+ayJR9YxUO3ICTF46WejU2iVtJvF8iOiw7eCbWy/
Og1rJh54q43dL+y+OjL9w5f//f9iKakFHy5FZCr8R0G86qDG5wxVES8JP4zH02T
yk4vtudrz1AB3LJTF4soglAOULsic83Fcr9akUirn5A86bDsjP+cL5fR2+zL1SmUW
qyNtClwU7E7hTNZSQFKQIHdClf2PcA==
```

3. INSTALACIÓN DEL CLIENTE AWS CLI

B) Asociar las credenciales de un usuario programático a la configuración de AWS CLI. Establecer la región AWS de trabajo como predeterminada y json como formato de salida (archivos credentials y config).

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws configure
AWS Access Key ID [None]: ASIA56KR5U3G2YIKK5UM
AWS Secret Access Key [None]: xN98kb66pq00TkvbWKy7SFyN3D+PH6wgDUbjQ9rw
Default region name [None]: us-east-1
Default output format [table]: json
```

Nota: Si se utiliza una cuenta de AWS Academy, es necesario añadir el aws_session_token al archivo credentials (las credenciales pueden obtenerse en AWS Details).

Añadimos el token de autenticación manualmente: `vim ~/.aws/credentials`

```
[default]
aws_access_key_id = ASIA56KR5U3G2YIKK5UM
aws_secret_access_key = xN98kb66pq00TkvbWKy7SFyN3D+PH6wgDUbjQ9rw
aws_session_token=IQoJb3JpZ2luX2VjEA8aCXVzLXdIc3QtMiJGMEQC1HCFNSipb2MmbMtj8J4IiyqevxWg8uFDMmRS3WTv+r+/aAiAuR/HxNBmeaX9qohxPI3tYkulFC
12xj8ZxTvXuPdWCi5AgfXEAAaDDk10DQ4NjM5MDQ3NyIMonKX+SfpXnZ+z50KpYCzBsQZmZhneZB0fRGqHcmJMwY/r77b1rtAxNpm6LY8AaaQq/Hh4Wh1ZYhPSnsI/gx
0ch59cb1ZxtQLFESH6FCBLRd4ljS9+50uC/YfTr98tqJx++662uACUWJfUk8lljQRPP5l7XtsGUcH2Q6btJEftoidAx1Fz4Uihsf4So0wALP+Tx9aV+IbPggUtgfTP4GF+
Ib7S2YC97gY46k/JDDbEx05G3/ZP97ThvCjJkoT1jqrxDEB/TjfwrPBoxEBh3+dFqzFXZqvLe12waYjhKocz02BNqD+ba7aX30AqtY/wHQSaPFzpElV4LKyuV3nQ+UodnCG
leHNZCTXvyVDZanygnGdRxzdj3P/SN1FYKpj9S91wWS1Yw6I0YsQY6ngGcOyyvipTnh+ayJR9YxU03ICTF46WejU2iVtJvF8i0iw7eCbwy/0g1rJh54q43dL+y+0jL9w5f/
/f9iKakFHy5FZCr8R0G86qDG5wxVES8JZP4zH02Tyk4vtudrz1AB3LJTF4sogLA0ULsic83Fcr9akUirn5A86bDsJP+cL5fR2+zL1SmUWqyYnEtClwU7E7hTNZSQFKQlHdC
lf2PcA==
```

3. INSTALACIÓN DEL CLIENTE AWS CLI

C) Verificar que se tiene acceso a la infraestructura AWS con el cliente configurado (por ejemplo, listar las zonas de disponibilidad de la región configurada).

Comprobamos que se puede acceder a los servicios de AWS:

```
$ aws ec2 describe-regions
```

```
$ aws ec2 describe-instances
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 describe-regions
{
  "Regions": [
    {
      "Endpoint": "ec2.ap-south-1.amazonaws.com",
      "RegionName": "ap-south-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-north-1.amazonaws.com",
      "RegionName": "eu-north-1",
      "OptInStatus": "opt-in-not-required"
    },
    {
      "Endpoint": "ec2.eu-west-3.amazonaws.com",
      "RegionName": "eu-west-3"
    }
  ]
}
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 describe-instances
{
  "Reservations": []
}
```

Ahora mismo no tengo ninguna

ÍNDICE

- 3. Instalación del cliente AWS CLI
- 4. **Amazon EC2 (Elastic Computing Cloud)**
- 5. Amazon VPC (Virtual Private Cloud)
- 6. Boto3. AWS SDK para Python

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

- A) Crear un nuevo grupo de seguridad (ssh-http-sg) que asociaremos a nuestras instancias EC2. Permitirá las conexiones entrantes por los puertos TCP 22 y 80. El grupo permitirá cualquier conexión saliente (es la configuración predeterminada).
- B) Crear una pareja de claves pública / privada con nombre CEU-Keys y guardar la clave privada (en formato .pem) en el directorio .ssh de nuestro usuario Linux. Cambiar los permisos de este archivo para que pueda ser utilizado por el cliente ssh.
- C) Obtener el ID (ami-xxxxxx) de la última imagen disponible para x86_64 cuya descripción comienza con la cadena Amazon Linux 2023 AMI. Lanzar una primera instancia EC2 usando el ID de esta imagen. Será del tipo t2.micro, usará el grupo de seguridad y las claves obtenidas en los apartados anteriores, y se ubicará en la subred por defecto de la primera zona de disponibilidad (1a).
- D) Verificar que se cuenta con una nueva instancia iniciada y obtener su dirección IP pública (elástica) asociada. A continuación, abrir una sesión SSH usando la clave privada y el usuario predeterminado de definido en la AMI (ec2-user en esta imagen).
- E) Detener la instancia EC2, verificar su nuevo estado y a continuación volver a arrancarla.

Nota: Indicar el proceso seguido con la consola CLI para obtener el ID de la imagen, y verificar que ese ID coincide con el que nos proponen como primera opción a través de la consola web. Si no se obtiene el ID esperado con AWS CLI, usar el propuesto por la consola web

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

- A) Crear un nuevo grupo de seguridad (ssh-http-sg) que asociaremos a nuestras instancias EC2. Permitirá las conexiones entrantes por los puertos TCP 22 y 80. El grupo permitirá cualquier conexión saliente (es la configuración predeterminada).

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 create-security-group --group-name ssh-http-sg --description "Grupo de Seguridad que permite conexiones entrantes TCP 22 y 80"
{
    "GroupId": "sg-0d9f0888a02ec59f4"
}
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 authorize-security-group-ingress --group-name ssh-http-sg --protocol
tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-039f74271eccff444",
            "GroupId": "sg-0d9f0888a02ec59f4",
            "GroupOwnerId": "958486390477",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Nota: las credenciales cambian cada vez que apagas el lab

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

- A) Crear un nuevo grupo de seguridad (ssh-http-sg) que asociaremos a nuestras instancias EC2. Permitirá las conexiones entrantes por los puertos TCP 22 y 80. El grupo permitirá cualquier conexión saliente (es la configuración predeterminada).

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 authorize-security-group-ingress --group-name ssh-http-sg --protocol
tcp --port 80 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0f9bb59b4d963b8a8",
            "GroupId": "sg-0d9f0888a02ec59f4",
            "GroupOwnerId": "958486390477",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Nota: las credenciales cambian cada vez que apagas el lab

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

B) Crear una pareja de claves pública / privada con nombre CEU-Keys y guardar la clave privada (en formato .pem) en el directorio .ssh de nuestro usuario Linux. Cambiar los permisos de este archivo para que pueda ser utilizado por el cliente ssh.

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 create-key-pair --key-name CEU-Keys --query 'KeyMaterial' --output text > CEU-Keys.pem
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ ls  
CEU-Keys.pem  dist  install  README.md  THIRD_PARTY LICENSES
```

```
-----BEGIN RSA PRIVATE KEY-----  
MIIEogIBAAQCAQEAvJaVkJGeCTWAEQZ2Jm973tjEMzxBJWlnTpxrF6345uS+6VK34  
XYSk2uvDBGk8hMeb7sutUp3FKTPrmXKpb+lc03lcttaBtqOoNthLKR5lx+YPUJ  
x8emNkj5U68MuIH/sRTT4Kayw08WA0Z+0$/fjZ7huTHMROteG0k0fwnp0184pv4  
Kt9xTTBW9wsBv/iimS0cUYQ0xbCOPp1A1sQ+wOn4+jj2LVmEJ8YdI2aTP+t83me  
KUPm13/l56fdBm9m6YwMH/2RQ56lVhf9prX1qgJEYChJugNbdaFptrPcwMmqdxr  
70enJsfpEQv6Zk90+EBLAGTo3HVsM3LdoPw+IDAQABAOIBAEEbfkDkoVFRsrq04a  
Am56Pu5s4TuBMWqWU1AwgZyoB6FY9Wk1XUC120nsl4vi509LfqDNRadp1n9h8  
VLZMTC1kBFWR4E0fn34vi0NnuluZy6ZeRR1vGzMCpnLR1vTUGICN7z+Tha9HLC  
zwE90f/Lcd01xtJfE14E8kd4TvMLyyhh/n9uyfchGRkwxxqF9hHcz/di3LXKQ3Cns8  
Mzwr/4T/9KvCzRiv5+l0wELF9fyRjJ9MRJo73LvvRwrGYo/sR4u0XciqOXXJuiEb  
8b5rIxixnxp0jNql/Uh09VxSmkWvSLsspMH0J1Bq03MSyadbfbeQ6HloUXBWR0zN  
Kth8r4EcgYEAB8jQ14DEb1FqqNFzUQ0LlvXnBdA2F8n6saULygebKcvS7PFUGtflk  
teCpiY9/E3yMANU02ZnlpBp3jv8dy/X2xqn2pnwgD9PioZH-CUIh2T4DRefdr5Cw  
7n3SRyc5iZv+2q4YnkIT3wY1d/Hj1gnVZKu0Fe9ZWds4k1lA/CIMcgYEax1s0  
N3tQ5fT0+0/38PGVI0NG+S5xYcCeFXKelvpinaVxCQw6uBFn+e7Kz/IPx8ZpeU8  
03SX6rttoy/nqT7Bb3pkafwHsRmk1Ln50Un6Yr5dHzpG4E0krdaDgn0g7j67sY7  
T8ypM//LRemiyq7lVlPMKMnl713AE49GRCK20kCgYBxyzPw1EHWA77tMPwMPmQ  
S6qHYY8m0nsgfRKQs6U0Y2VA0FxP02lVkgS3FTNlYD9txxhAiXoogK/agvu04R  
orMevzy1Ewdsl1H7eZw7opYZFvohzWJDetv70CMJu0Bm79rjJ/p8E0e1xfEcK9wt  
gzzvg00puhLw0h9Bdsy/bwKBgCnECdi00j0Eznlsrr0jozo16pujqhY1aepMjfZM  
mIU3uSuQKG2RBGBxCdhFhvTlc3RTS92eq5wfLDFGL2B0n4dyQD7w1ltNJF71HTF  
RUumFwbVdKKR7Byh14YZh+ZhyBjN7t2VnRZLjsbAAAGGAnrXPn0wto3RZUqaDXKh2  
0WIhAoGAR+SLBks+GKyp5m2l2exPdnRlkv8SxmxFGLbh1o/dw2UB7fbH8BulQovhGd  
ap17CaAW2k0LIMKyp5m2l2exPdnRlkv8SxmxFGLbh1o/dw2UB7fbH8BulQovhGd  
7dwIMfhGF2tARVkcjk1PoB1ERiVuutSPtHT7mhGISG+IQ4BP/Y=
```

-----END RSA PRIVATE KEY-----

vim CEU-Keys

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ chmod 400 CEU-Keys.pem
```

Copiarlo en el directorio .ssh:

```
mkdir -p ~/.ssh  
cp CEU-Keys.pem ~/.ssh/  
chmod 400 ~/.ssh/CEU-Keys.pem
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ mkdir -p ~/.ssh  
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ cp CEU-Keys.pem ~/.ssh/  
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ chmod 400 ~/.ssh/CEU-Keys.pem  
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ ls ~/.ssh/  
CEU-Keys.pem
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

C) Obtener el ID (ami-xxxxxx) de la última imagen disponible para x86_64 cuya descripción comienza con la cadena Amazon Linux 2023 AMI. Lanzar una primera instancia EC2 usando el ID de esta imagen. Será del tipo t2.micro, usará el grupo de seguridad y las claves obtenidas en los apartados anteriores, y se ubicará en la subred por defecto de la primera zona de disponibilidad (1a).

Buscar las AMIs cuya descripción comienza con “Amazon Linux 2023 AMI” y seleccionará la última según la fecha de creación:

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 describe-images --filters "Name=name,Values=Amazon Linux 2023 AMI*" --query "Images | sort_by(@, &CreationDate) | [-1].ImageId" --output text  
ami-092ab41080623aaaa
```

```
$ aws ec2 describe-images \  
--filters "Name=name,Values=Amazon Linux 2023 AMI*" \  
--query "Images | sort_by(images, &CreationDate) | [-1].ImageId" \  
--output text
```

Esa imagen en concreto falla con la cuenta de Lab, para elegir una imagen correcta hemos entrado en la consola y hemos seleccionado la imagen ami de t2.micro

ami-04e5276ebb8451442

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

C) Obtener el ID (ami-xxxxxx) de la última imagen disponible para x86_64 cuya descripción comienza con la cadena Amazon Linux 2023 AMI. Lanzar una primera instancia EC2 usando el ID de esta imagen. Será del tipo t2.micro, usará el grupo de seguridad y las claves obtenidas en los apartados anteriores, y se ubicará en la subred por defecto de la primera zona de disponibilidad (1a).

Primero averiguamos la subnet de la zona de disponibilidad y la región:

```
$ aws ec2 describe-subnets
```

```
{  
    "AvailabilityZone": "us-east-1a",  
    "AvailabilityZoneId": "use1-az1",  
    "AvailableIpAddressCount": 4091,  
    "CidrBlock": "172.31.0.0/20",  
    "DefaultForAz": true,  
    "MapPublicIpOnLaunch": true,  
    "MapCustomerOwnedIpOnLaunch": false,  
    "State": "available",  
    "SubnetId": "subnet-0096e44e33f1d04e8",  
    "VpcId": "vpc-00a5dfc0664ac615a",  
    "OwnerId": "958486390477",  
    "AssignIpv6AddressOnCreation": false,  
    "Ipv6CidrBlockAssociationSet": [],  
    "SubnetArn": "arn:aws:ec2:us-east-1:958486390477:subnet/subnet-0096e44e33f1d04e8",  
    "EnableDns64": false,  
    "Ipv6Native": false,  
}
```

subnet-0096e44e33f1d04e8

```
$ aws ec2 describe-regions us-east-1
```

```
{  
    "Endpoint": "ec2.us-east-1.amazonaws.com",  
    "RegionName": "us-east-1",  
    "OptInStatus": "opt-in-not-required"  
},
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

C) Obtener el ID (ami-xxxxxx) de la última imagen disponible para x86_64 cuya descripción comienza con la cadena Amazon Linux 2023 AMI. Lanzar una primera instancia EC2 usando el ID de esta imagen. Será del tipo t2.micro, usará el grupo de seguridad y las claves obtenidas en los apartados anteriores, y se ubicará en la subred por defecto de la primera zona de disponibilidad (1a).

Lanzamos la instancia

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 run-instances --image-id ami-04e5276ebb8451442 --count 1 --instance-type t2.micro --key-name CEU-Keys --security-group-ids sg-0d9f0888a02ec59f4 --placement AvailabilityZone=us-east-1a --query 'Instances[0].InstanceId' --region us-east-1  
"i-058372ee5476e0bd6"  
  
"i-058372ee5476e0bd6"
```

```
aws ec2 run-instances --image-id ami-04e5276ebb8451442 --count 1 --instance-type t2.micro --key-name CEU-Keys --security-group-ids sg-0d9f0888a02ec59f4 --placement AvailabilityZone=us-east-1a --query 'Instances[0].InstanceId' --region us-east-1
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

D) Verificar que se cuenta con una nueva instancia iniciada y obtener su dirección IP pública (elástica) asociada. A continuación, abrir una sesión SSH usando la clave privada y el usuario predeterminado de definido en la AMI (ec2-user en esta imagen).

Verificar instancias:

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].[InstanceId, PublicIpAddress]' --output text
i-0a51649ff43d637ba    None      Instancias de pruebas terminadas
i-0609b5576fa536a1c    None
i-058372ee5476e0bd6    44.200.195.60    IP Pública de nuestra instancia
```

```
$ aws ec2 describe-instances --query 'Reservations[*].Instances[*].[InstanceId, PublicIpAddress]' --output text
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

D) Verificar que se cuenta con una nueva instancia iniciada y obtener su dirección IP pública (elástica) asociada. A continuación, abrir una sesión SSH usando la clave privada y el usuario predeterminado de definido en la AMI (ec2-user en esta imagen).

Abrir sesión ssh:

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ ssh ec2-user@44.200.195.60 -i CEU-Keys.pem
The authenticity of host '44.200.195.60 (44.200.195.60)' can't be established.
ED25519 key fingerprint is SHA256:FP2yzXQvZAwdcnRFqUhFUM7TAgv/hcNV4Z6x2XqzAsA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '44.200.195.60' (ED25519) to the list of known hosts.

,
#_
~\_\_ #####_          Amazon Linux 2023
~~ \_\#\#\#\#
~~  \#\#\#
~~      \#/ _--> https://aws.amazon.com/linux/amazon-linux-2023
~~      V~' '-'>
~~~           /
~~_. /_/
/_m/'_/
[ec2-user@ip-172-31-4-90 ~]$
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

E) Detener la instancia EC2, verificar su nuevo estado y a continuación volver a arrancarla

ID de nuestra instancia: i-058372ee5476e0bd6

Parar la instancia:

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 stop-instances --instance-ids i-058372ee5476e0bd6
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-058372ee5476e0bd6",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

E) Detener la instancia EC2, verificar su nuevo estado y a continuación volver a arrancarla

Comprobar su estado:

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 describe-instances --query 'Reservations[].Instances[][Placement.AvailabilityZone,State.Name,InstanceId]' --output text
us-east-1b      terminated      i-0a51649ff43d637ba
us-east-1b      terminated      i-0609b5576fa536a1c
us-east-1a      stopped        i-058372ee5476e0bd6
ubuntupaloma@ubuntu:~/Desktop/aws/aws$
```

4. AMAZON EC2 (ELASTIC COMPUTING CLOUD)

E) Detener la instancia EC2, verificar su nuevo estado y a continuación volver a arrancarla

Volver a arrancarla:

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 start-instances --instance-ids i-058372ee5476e0bd6
{
    "StartingInstances": [
        {
            "CurrentState": {
                "Code": 0,
                "Name": "pending"
            },
            "InstanceId": "i-058372ee5476e0bd6",
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 describe-instances --query 'Reservations[].Instances[].[Placement.AvailabilityZone,State.Name,InstanceId]' --output text
us-east-1b      terminated      i-0a51649ff43d637ba
us-east-1b      terminated      i-0609b5576fa536a1c
us-east-1a      pending        i-058372ee5476e0bd6
```

ÍNDICE

3. Instalación del cliente AWS CLI
4. Amazon EC2 (Elastic Computing Cloud)
- 5. Amazon VPC (Virtual Private Cloud)**
6. Boto3. AWS SDK para Python

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

- A. Identificar las características de la red VPC de nuestra cuenta y de las subredes presentes en las zonas de disponibilidad de la región en la que nos encontramos (identificador de red y subredes, y bloque de direcciones IP asociadas).
- B. Crear una nueva subred en la tercera zona de disponibilidad (1c) de nuestra región. Tendrá asociado un bloque de direcciones /24 válido (dentro del espacio de direcciones de la VPC, y aún no asignado). Visualizar las propiedades de la subred creada ¿Qué valor tiene la propiedad map-public-ip-on-launch? ¿Qué implica ese valor?
- C. Lanzar una nueva instancia con las mismas características de la anterior pero ahora ubicada en la nueva subred. La instancia no tendrá dirección IP pública (elástica) asociada.
- D. Obtener la dirección IP privada de la nueva instancia creada. Comprobar que es posible abrir una conexión SSH a la nueva instancia desde la anterior (será necesario contar con la clave SSH privada en la primera instancia).
- E. La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

A) Identificar las características de la red VPC de nuestra cuenta y de las subredes presentes en las zonas de disponibilidad de la región en la que nos encontramos (identificador de red y subredes, y bloque de direcciones IP asociadas).

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 describe-vpcs
{
    "Vpcs": [
        {
            "CidrBlock": "172.31.0.0/16",
            "DhcpOptionsId": "dopt-08516077413f59bf6",
            "State": "available",
            "VpcId": "vpc-00a5dfc0664ac615a",
            "OwnerId": "958486390477",
            "InstanceTenancy": "default",
            "CidrBlockAssociationSet": [
                {
                    "AssociationId": "vpc-cidr-assoc-07f5621a7dccbcb28",
                    "CidrBlock": "172.31.0.0/16",
                    "CidrBlockState": {
                        "State": "associated"
                    }
                }
            ],
            "IsDefault": true
        }
    ]
}
```

ID VPC: vpc-00a5dfc0664ac615a

Bloque de direcciones: 172.31.0.0/16

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-00a5dfc0664ac615a" --query "Subnets[*].[SubnetId,CidrBlock,AvailabilityZone]"
[
    [
        "subnet-0abe696b516920447",
        "172.31.32.0/20",
        "us-east-1d"
    ],
    [
        "subnet-06590d11e9b6c0295",
        "172.31.96.0/24",
        "us-east-1c"
    ],
    [
        "subnet-0096e44e33f1d04e8",
        "172.31.0.0/20",
        "us-east-1a"
    ],
    [
        "subnet-0d183fe04df17c9dd",
        "172.31.80.0/20",
        "us-east-1b"
    ],
    [
        "subnet-098d6c5fbeae93c0f",
        "172.31.48.0/20",
        "us-east-1e"
    ],
    [
        "subnet-061ad182da4aa73cb",
        "172.31.16.0/20",
        "us-east-1c"
    ],
    [
        "subnet-041f0345f41404651",
        "172.31.64.0/20",
        "us-east-1f"
    ]
]
```

A) Identificar las características de la red VPC de nuestra cuenta y de las subredes presentes en las zonas de disponibilidad de la región en la que nos encontramos (identificador de red y subredes, y bloque de direcciones IP asociadas).

```
$ aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-00a5dfc0664ac615a"--query "Subnets[*].[SubnetId,CidrBlockl, AvailabilityZone]"
```

```
[  
    "subnet-06590d11e9b6c0295",  
    "172.31.96.0/24",  
    "us-east-1c"  
],
```

```
[  
    "subnet-061ad182da4aa73cb",  
    "172.31.16.0/20",  
    "us-east-1c"  
],
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

B) Crear una nueva subred en la tercera zona de disponibilidad (1c) de nuestra región. Tendrá asociado un bloque de direcciones /24 válido (dentro del espacio de direcciones de la VPC, y aún no asignado). Visualizar las propiedades de la subred creada ¿Qué valor tiene la propiedad map-public-ip-on-launch? ¿Qué implica ese valor?

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 create-subnet --vpc-id vpc-00a5dfc0664ac615a --cidr-block
172.31.96.0/24 --availability-zone us-east-1c
{
    "Subnet": {
        "AvailabilityZone": "us-east-1c",
        "AvailabilityZoneId": "use1-az4",
        "AvailableIpAddressCount": 251,
        "CidrBlock": "172.31.96.0/24",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false,
        "State": "available",
        "SubnetId": "subnet-06590d11e9b6c0295",
        "VpcId": "vpc-00a5dfc0664ac615a",
        "OwnerId": "958486390477",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "SubnetArn": "arn:aws:ec2:us-east-1:958486390477:subnet/subnet-06590d11e9b6c0295",
        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        }
    }
}
```

Bloque de direcciones elegido: 172.31.96.0/24
ID_SUBNET: subnet-06590d11e9b6c0295

MapPublicInAddressCount: false

Indica si las instancias lanzadas en esta subred recibirán automáticamente una dirección IPv4 pública.

- TRUE → significa que las instancias recibirán una dirección IP pública al ser lanzadas, lo que les permite ser accesibles desde internet.
- FALSE → las instancias no recibirán una dirección IP pública automáticamente, lo que las hace inaccesibles desde internet a menos que se configure una IP pública manualmente o se utilice un dispositivo NAT.

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

C) Lanzar una nueva instancia con las mismas características de la anterior pero ahora ubicada en la nueva subred. La instancia no tendrá dirección IP pública (elástica) asociada.

```
ubuntupaloma@ubuntu:~$ aws ec2 run-instances --image-id ami-04e5276ebb8451442 --count 1
--instance-type t2.micro --key-name CEU-Keys --security-group-ids sg-0d9f0888a02ec59f4
--subnet-id subnet-06590d11e9b6c0295 --no-associate-public-ip-address
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-04e5276ebb8451442",
            "InstanceId": "i-03f8f0fbc26fe7b77",
            "InstanceType": "t2.micro",
            "KeyName": "CEU-Keys",
            "LaunchTime": "2024-04-24T16:27:07+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1c",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-96-200.ec2.internal",
            "PrivateIpAddress": "172.31.96.200",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "Status": "pending"
        }
    ]
}
```

ID AMI anterior:
ami-04e5276ebb8451442

SubnetID:
subnet-06590d11e9b6c0295

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

C) Lanzar una nueva instancia con las mismas características de la anterior pero ahora ubicada en la nueva subred. La instancia no tendrá dirección IP pública (elástica) asociada.

```
"PublicDnsName": "",  
"AvailabilityZone": "us-east-1c",  
"PrivateIpAddress": "172.31.96.200",  
"InstanceId": "i-03f8f0fbc26fe7b77",  
"SubnetId": "subnet-06590d11e9b6c0295",  
"VpcId": "vpc-00a5dfc0664ac615a",
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

D) Obtener la dirección IP privada de la nueva instancia creada. Comprobar que es posible abrir una conexión SSH a la nueva instancia desde la anterior (será necesario contar con la clave SSH privada en la primera instancia).

Descargamos las claves en la instancia con la dirección pública:

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ sftp -i CEU-Keys.pem ec2-user@3.234.250.77:/home/ec2-user
Connected to 3.234.250.77.
Changing to: /home/ec2-user
sftp> put CEU-Keys.pem /home/ec2-user
Uploading CEU-Keys.pem to /home/ec2-user/CEU-Keys.pem
CEU-Keys.pem                                         100% 1675      7.7KB/s   00:00
%
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

D) Obtener la dirección IP privada de la nueva instancia creada. Comprobar que es posible abrir una conexión SSH a la nueva instancia desde la anterior (será necesario contar con la clave SSH privada en la primera instancia).

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ ssh -i CEU-Keys.pem ec2-user@3.80.162.148
The authenticity of host '3.80.162.148 (3.80.162.148)' can't be established.
ED25519 key fingerprint is SHA256:FP2yzXQvZAwdcnRFqUhFUM7TAgv/hcNV4Z6x2XqzAsA.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:1: [hashed name]
  ~/.ssh/known_hosts:3: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.80.162.148' (ED25519) to the list of known hosts.
,
  #_
~\_\ #####_          Amazon Linux 2023
~~ \_\#####\
~~  \###|
~~   \#/ __ https://aws.amazon.com/linux/amazon-linux-2023
~~   V~' '-'>
~~~   / /
~~~.._. / /
~~~ / / -/
~~~ /m/ ' / /
Last login: Tue Apr 23 17:00:38 2024 from 81.41.169.152
[ec2-user@ip-172-31-4-90 ~]$ ls
CEU-Keys.pem
```

```
Last login: Tue Apr 23 17:00:38 2024 from 81.41.169.152
[ec2-user@ip-172-31-4-90 ~]$ ls
CEU-Keys.pem
[ec2-user@ip-172-31-4-90 ~]$ ssh -i CEU-Keys.pem ec2-user@172.31.96.200
The authenticity of host '172.31.96.200 (172.31.96.200)' can't be established.
ED25519 key fingerprint is SHA256:+0JElAiADLrkEBrNBu/Xr+5rSrtzqCNde5QNxU88KGY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.31.96.200' (ED25519) to the list of known hosts.
,
  #_
~\_\ #####_          Amazon Linux 2023
~~ \_\#####\
~~  \###|
~~   \#/ __ https://aws.amazon.com/linux/amazon-linux-2023
~~   V~' '-'>
~~~   / /
~~~.._. / /
~~~ / / -/
~~~ /m/ ' / /
[ec2-user@ip-172-31-96-200 ~]$
```

Nota: La IP pública es distinta porque lo hice otro dia

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

Reservar una IP Pública elástica:

```
$aws ec2 allocate-address --domain vpc
```

```
ubuntu@ubuntuloma: ~/Desktop/aws/aws$ aws ec2 allocate-address --domain vpc
{
    "PublicIp": "44.222.14.124",
    "AllocationId": "eipalloc-0a133010edf734849",
    "PublicIpv4Pool": "amazon",
    "NetworkBorderGroup": "us-east-1",
    "Domain": "vpc"
}
ubuntu@ubuntuloma: ~/Desktop/aws/aws$
```

```
"PublicIp": "44.222.14.124",
"AllocationId": "eipalloc-0a133010edf734849",
"PublicIpv4Pool": "amazon",
"NetworkBorderGroup": "us-east-1",
"Domain": "vpc"
```

Nota: La IP pública es distinta porque lo hice otro dia

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

Crear un Nat Gateway

```
$ aws ec2 create-nat-gateway --subnet-id subnet-xxxxxx --allocation-id eipalloc-xxxxxxxx  
aws ec2 create-nat-gateway --subnet-id SUBRED US EAST 1-C --allocation-id eipalloc-0a133010edf734849
```

```
aws ec2 create-nat-gateway --subnet-id subnet-061ad182da4aa73cb--allocation-id eipalloc-  
0a133010edf734849
```

```
$ aws ec2 create-nat-gateway --subnet-id subnet-061ad182da4aa73cb --allocation-id eipalloc-  
0a133010edf734849 --tag-specifications  
'ResourceType=natgateway,Tags=[{Key=Name,Value=NATGatewayVPC}]'
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 create-nat-gateway --subnet-id subnet-061ad182da4aa73cb --allocation-id eipalloc-0a133010edf734849 --tag-specifications 'ResourceType=natgateway,Tags=[{Key=Name,Value=NATGatewayVPC}]'  
{  
    "ClientToken": "d46c6690-1685-45d1-814b-40a5173d281f",  
    "NatGateway": {  
        "CreateTime": "2024-04-25T10:25:48+00:00",  
        "NatGatewayAddresses": [  
            {  
                "AllocationId": "eipalloc-0a133010edf734849",  
                "IsPrimary": true,  
                "Status": "associating"  
            }  
        ],  
        "NatGatewayId": "nat-0a019353025d300ce",  
        "State": "pending",  
        "SubnetId": "subnet-061ad182da4aa73cb",  
        "VpcId": "vpc-00a5dfc0664ac615a",  
        "Tags": [  
            {  
                "Key": "Name",  
                "Value": "NATGatewayVPC"  
            }  
        ],  
        "ConnectivityType": "public"  
    }  
}
```

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

Crear una tabla de rutas para la subred privada → crear una tabla de rutas :

```
$ aws ec2 create-route-table --vpc-id  
vpc-xxxxxx  
$ aws ec2 create-route-table --vpc-id  
vpc-00a5dfc0664ac615a
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 create-route-table --vpc-id vpc-00a5dfc0  
664ac615a  
{  
    "RouteTable": {  
        "Associations": [],  
        "PropagatingVgws": [],  
        "RouteTableId": "rtb-06c437e841672f249",  
        "Routes": [  
            {  
                "DestinationCidrBlock": "172.31.0.0/16",  
                "GatewayId": "local",  
                "Origin": "CreateRouteTable",  
                "State": "active"  
            }  
        ],  
        "Tags": [],  
        "VpcId": "vpc-00a5dfc0664ac615a",  
        "OwnerId": "958486390477"  
    },  
    "ClientToken": "60a72cd5-e2fe-4487-82fd-de5d0d06f28b"  
}
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

Asociar la tabla de rutas a la subred privada:

```
$ aws ec2 associate-route-table --route-table-id rtb-xxxxxx --subnet-id subnet-xxxxxx  
$ aws ec2 associate-route-table --route-table-id rtb-06c437e841672f249 --subnet-id subnet-06590d11e9b6c0295
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 associate-route-table --route-table-id rtb-06c437e841672f249 --subnet-id subnet-06590d11e9b6c0295  
{  
    "AssociationId": "rtbassoc-0db1479e14975bed9",  
    "AssociationState": {  
        "State": "associated"  
    }  
}
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

Crear una ruta en la tabla de rutas para dirigir el tráfico hacia el NAT Gateway:

```
$ aws ec2 create-route --route-table-id rtb-xxxxxx --destination-cidr-block 0.0.0.0/0 --gateway-id nat-xxxxxx  
$ aws ec2 create-route --route-table-id rtb-06c437e841672f249 --destination-cidr-block 0.0.0.0/0 --gateway-id  
nat-0a019353025d300ce
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 create-route --route-table-id rtb-06c437  
e841672f249 --destination-cidr-block 0.0.0.0/0 --gateway-id nat-0a019353025d300ce  
{  
    "Return": true  
}
```

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

E) La nueva instancia creada no tiene salida a Internet. Para facilitar la salida de esta máquina reservar en primer lugar una IP pública (elástica) y a continuación crear un nat-gateway que tendrá asociada la IP pública reservada y estará ubicado en la subred pública de la tercera zona de disponibilidad. Crear una tabla de rutas para la subred privada y una nueva ruta que usará el nat-gateway para las conexiones externas desde la subred privada. Verificar que la nueva instancia tiene conectividad exterior.

IP pública de la instancia: 3.235.155.115

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ ssh -i CEU-Keys.pem ec2-user@3.235.155.115
The authenticity of host '3.235.155.115 (3.235.155.115)' can't be established.
ED25519 key fingerprint is SHA256:FP2yzXQvZAwdcnRFqUhFUM7TAgv/hcNV4Z6x2XqzAsA.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:1: [hashed name]
  ~/.ssh/known_hosts:3: [hashed name]
  ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.235.155.115' (ED25519) to the list of known hosts.
,
#_
~\_\ #####
~~ \_\#####\ Amazon Linux 2023
~~   \###|
~~     \|/|--- https://aws.amazon.com/linux/amazon-linux-2023
~~       \|-'-->
~~~      /|_
~~~.-./|-/
~-/m/`-/
Last login: Wed Apr 24 17:10:08 2024 from 81.41.169.152
[ec2-user@ip-172-31-4-90 ~]$ ssh -i CEU-Keys.pem ec2-user@172.31.96.200
```

Tiene conectividad
al exterior

5. AMAZON VPC (VIRTUAL PRIVATE CLOUD)

```
Last login: Wed Apr 24 17:10:08 2024 from 81.41.169.152
[ec2-user@ip-172-31-4-90 ~]$ ssh -i CEU-Keys.pem ec2-user@172.31.96.200
      , #
      ~\_\####_
      ~~ \_\#####\
      ~~   \###|
      ~~     \#/ _-- https://aws.amazon.com/linux/amazon-linux-2023
      ~~       V~' '-->
      ~~~         /
      ~~.~.        /
      _/_\_\_/
      _/m/'^C

Last login: Wed Apr 24 17:10:10 2024 from 172.31.4.90
[ec2-user@ip-172-31-96-200 ~]$ ping google.com
PING google.com (172.253.122.113) 56(84) bytes of data.
64 bytes from bh-in-f113.1e100.net (172.253.122.113): icmp_seq=1 ttl=55 time=2.48 ms
64 bytes from bh-in-f113.1e100.net (172.253.122.113): icmp_seq=2 ttl=55 time=2.23 ms
64 bytes from bh-in-f113.1e100.net (172.253.122.113): icmp_seq=3 ttl=55 time=1.93 ms
64 bytes from bh-in-f113.1e100.net (172.253.122.113): icmp_seq=4 ttl=55 time=1.98 ms
64 bytes from bh-in-f113.1e100.net (172.253.122.113): icmp_seq=5 ttl=55 time=1.99 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 1.929/2.120/2.475/0.204 ms
```

ÍNDICE

- 3. Instalación del cliente AWS CLI
- 4. Amazon EC2 (Elastic Computing Cloud)
- 5. Amazon VPC (Virtual Private Cloud)
- 6. Boto3. AWS SDK para Python**

6. BOTO3. AWS SDK PARA PYTHON

- A) Instalar el SDK Boto3 para AWS. Utilizar las credenciales de usuario creadas al comienzo de la práctica. Verificar su funcionamiento con el intérprete de Python.
- B) Crear un script en Python para listar todas las instancias EC2 no terminadas presentes en la región predeterminada. Devolverá su ID, estado (solo el “Name”), ID de la subred donde se encuentra, dirección IP privada y, en caso de tenerla, dirección IP pública (elástica).
- C) Crear un script en Python para listar todas las subredes presentes en las VPC existentes en la región. Para cada subred devolverá su ID, el ID de la VPC donde se encuentra, el bloque de direcciones IP asociado a la subred, y si está activada la asignación automática de dirección IP pública (yes|no).
- D) Crear un script en Python para lanzar una nueva instancia EC2 en la región predeterminada. Recibirá como parámetros el ID de la imagen (AMI), el tipo de instancia, el ID de la subred a la que se conectará, si tendrá asociada una IP pública o elástica (parámetro yes|no), id del grupo de seguridad, y nombre de la clave pública a injectar. Devolverá el ID de la nueva instancia y su estado.
- E) Crear un script en Python que nos indique el número de volúmenes, instantáneas, grupos de seguridad, balanceadores de carga y parejas de claves asignados a nuestra cuenta en la región en la que nos encontramos.
- F) Crear un script en Python para eliminar (Terminate) la instancia o instancias cuyos ID se pasan como parámetro.
- G) Crear un script en Python para arrancar todas las instancias de la región que se encuentren detenidas (Stopped) y tengan asociada una etiqueta (Tag) que se pasará como parámetro.

6. BOTO3. AWS SDK PARA PYTHON

A) Instalar el SDK Boto3 para AWS. Utilizar las credenciales de usuario creadas al comienzo de la práctica. Verificar su funcionamiento con el intérprete de Python.

```
sudo apt install python3-pip  
$ pip install boto3
```

```
Defaulting to user installation because normal site-packages is not writeable  
Collecting boto3  
  Downloading boto3-1.34.91-py3-none-any.whl (139 kB)  
          139.3/139.3 KB 3.0 MB/s eta 0:00:00  
Collecting jmespath<2.0.0,>=0.7.1  
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)  
Collecting botocore<1.35.0,>=1.34.91  
  Downloading botocore-1.34.91-py3-none-any.whl (12.2 MB)  
          12.2/12.2 MB 17.3 MB/s eta 0:00:00  
Collecting s3transfer<0.11.0,>=0.10.0  
  Downloading s3transfer-0.10.1-py3-none-any.whl (82 kB)  
          82.2/82.2 KB 11.5 MB/s eta 0:00:00  
Requirement already satisfied: urllib3!=2.2.0,<3,>=1.25.4 in /usr/lib/python3/dist-packages (from botocore<1.35.0,>=1.34.91->boto3) (1.26.5)  
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /usr/lib/python3/dist-packages (from botocore<1.35.0,>=1.34.91->boto3) (2.8.1)  
Installing collected packages: jmespath, botocore, s3transfer, boto3  
Successfully installed boto3-1.34.91 botocore-1.34.91 jmespath-1.0.1 s3transfer-0.10.1
```

Comprobar que funciona con Python:

```
import boto3  
print(boto3.__version__)
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ vim botoVersion.py  
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 botoVersion.py  
1.34.91
```

Comprobar que funciona con las credenciales:

```
import boto3  
ec2 = boto3.resource('ec2')  
for instance in ec2.instances.all():  
    print(instance.id, instance.state)
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ vim pruebaCredenciales.py  
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 pruebaCredenciales.py  
i-03f8f0fb26fe7b77 {'Code': 16, 'Name': 'running'}  
i-058372ee5476e0bd6 {'Code': 16, 'Name': 'running'}
```

6. BOTO3. AWS SDK PARA PYTHON

B) Crear un script en Python para listar todas las instancias EC2 no terminadas presentes en la región predeterminada. Devolverá su ID, estado (solo el “Name”), ID de la subred donde se encuentra, dirección IP privada y, en caso de tenerla, dirección IP pública (elástica).

```
import boto3

ec2 = boto3.client('ec2')

def lista_de_instancias():
    instancias = ec2.describe_instances(
        Filters=[
            {'Name': 'instance-state-name', 'Values': ['pending', 'running', 'stopping', 'stopped']}
        ]
    ) # Todas menos terminated

    for reservations in instancias['Reservations']:
        for instancia in reservations['Instances']:
            ip_publica = instancia.get('PublicIpAddress', 'None') # dirección IP pública sino 'None'

            print(f"ID: {instancia['InstanceId']}")
            print(f"Estado: {instancia['State']['Name']}")
            print(f"ID de Subred: {instancia['SubnetId']}")
            print(f"Dirección IP Privada: {instancia['PrivateIpAddress']}")
            print(f"Dirección IP Pública: {ip_publica}")
            print("-" * 60)

lista_de_instancias()
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ vim instancias.py
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 instancias.py
ID: i-03f8f0fbc26fe7b77
Estado: running
ID de Subred: subnet-06590d11e9b6c0295
Dirección IP Privada: 172.31.96.200
Dirección IP Pública: None
-----
ID: i-058372ee5476e0bd6
Estado: running
ID de Subred: subnet-0096e44e33f1d04e8
Dirección IP Privada: 172.31.4.90
Dirección IP Pública: 34.200.223.166
-----
```

6. BOTO3. AWS SDK PARA PYTHON

C) Crear un script en Python para listar todas las subredes presentes en las VPC existentes en la región. Para cada subred devolverá su ID, el ID de la VPC donde se encuentra, el bloque de direcciones IP asociado a la subred, y si está activada la asignación automática de dirección IP pública (yes|no).

```
$ aws ec2 describe-subnets
```

```
{  
    "Subnets": [  
        {  
            "AvailabilityZone": "us-east-1d",  
            "AvailabilityZoneId": "use1-az6",  
            "AvailableIpAddressCount": 4091,  
            "CidrBlock": "172.31.32.0/20",  
            "DefaultForAz": true,  
            "MapPublicIpOnLaunch": true,  
            "MapCustomerOwnedIpOnLaunch": false,  
            "State": "available",  
            "SubnetId": "subnet-0abe696b516920447",  
            "VpcId": "vpc-00a5dfc0664ac615a",  
            "OwnerId": "958486390477",  
            "AssignIpv6AddressOnCreation": false,  
            "Ipv6CidrBlockAssociationSet": [],  
            "SubnetArn": "arn:aws:ec2:us-east-1:958486390477:subnet/subnet-0abe696b5169  
20447",  
            "EnableDns64": false,  
            "Ipv6Native": false,  
            "PrivateDnsNameOptionsOnLaunch": {  
                "HostnameType": "ip-name",  
                "EnableResourceNameDnsARecord": false,  
                "EnableResourceNameDnsAAAARecord": false  
            }  
        }  
    ]  
}
```

En base a la salida de `describe-subnets` cogemos:

- SubnetId
- VpcId
- CidrBlock
- MapPublicIpOnLaunch (si es True → Yes)

6. BOTO3. AWS SDK PARA PYTHON

C) Crear un script en Python para listar todas las subredes presentes en las VPC existentes en la región. Para cada subred devolverá su ID, el ID de la VPC donde se encuentra, el bloque de direcciones IP asociado a la subred, y si está activada la asignación automática de dirección IP pública (yes|no).

```
import boto3

ec2 = boto3.client('ec2', region_name='us-east-1')

subnets = ec2.describe_subnets()

for subnet in subnets['Subnets']:
    auto_assign_ip = 'yes' if subnet['MapPublicIpOnLaunch'] else 'no'

    print(f"Subnet ID: {subnet['SubnetId']}")
    print(f"VPC ID: {subnet['VpcId']}")
    print(f"CIDR Block: {subnet['CidrBlock']}")
    print(f"Auto-assign Public IP: {auto_assign_ip}")
    print("-" * 60)
```

6. BOTO3. AWS SDK PARA PYTHON

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 subredes.py
Subnet ID: subnet-0abe696b516920447
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.32.0/20
Auto-assign Public IP: yes
-----
Subnet ID: subnet-06590d11e9b6c0295
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.96.0/24
Auto-assign Public IP: no
-----
Subnet ID: subnet-0096e44e33f1d04e8
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.0.0/20
Auto-assign Public IP: yes
-----
Subnet ID: subnet-0d183fe04df17c9dd
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.80.0/20
Auto-assign Public IP: yes
-----
Subnet ID: subnet-098d6c5fbeae93c0f
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.48.0/20
Auto-assign Public IP: yes
-----
Subnet ID: subnet-061ad182da4aa73cb
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.16.0/20
Auto-assign Public IP: yes
-----
Subnet ID: subnet-041f0345f41404651
VPC ID: vpc-00a5dfc0664ac615a
CIDR Block: 172.31.64.0/20
Auto-assign Public IP: yes
-----
```

C) Crear un script en Python para listar todas las subredes presentes en las VPC existentes en la región. Para cada subred devolverá su ID, el ID de la VPC donde se encuentra, el bloque de direcciones IP asociado a la subred, y si está activada la asignación automática de dirección IP pública (yes|no).

6. BOTO3. AWS SDK PARA PYTHON

D) Crear un script en Python para lanzar una nueva instancia EC2 en la región predeterminada. Recibirá como parámetros el ID de la imagen (AMI), el tipo de instancia, el ID de la subred a la que se conectará, si tendrá asociada una IP pública o elástica (parámetro yes|no), id del grupo de seguridad, y nombre de la clave pública a injectar. Devolverá el ID de la nueva instancia y su estado.

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 lanzarInstancia4.py  
ID de la instancia: i-0f4bace2083f21770  
Estado de la instancia: pending
```

```
import boto3

ec2_client = boto3.client('ec2')

def lanzar_instancia_ec2(ami_id, instance_type, subnet_id, associate_public_ip, security_group_id, key_name):

    public_ip = associate_public_ip.lower() == 'yes'

    response = ec2_client.run_instances(
        ImageId=ami_id,
        InstanceType=instance_type,
        NetworkInterfaces=[
            {
                'AssociatePublicIpAddress': public_ip,
                'SubnetId': subnet_id,
                'Groups': [security_group_id],
                'DeviceIndex': 0
            }
        ],
        KeyName=key_name,
        MinCount=1,
        MaxCount=1
    )

    instance_id = response['Instances'][0]['InstanceId']
    state = response['Instances'][0]['State']['Name']

    return instance_id, state

ami_id = 'ami-04e5276ebb8451442'
instance_type = 't2.micro'
subnet_id = 'subnet-061ad182da4aa73cb'
associate_public_ip = 'yes'
security_group_id = 'sg-0d9f0888a02ec59f4'
key_name = 'CEU-Keys'

instance_id, instance_state = lanzar_instancia_ec2(ami_id, instance_type, subnet_id, associate_public_ip, security_group_id, key_name)

print(f"ID de la instancia: {instance_id}")
print(f"Estado de la instancia: {instance_state}")
```

6. BOTO3. AWS SDK PARA PYTHON

D) Crear un script en Python para lanzar una nueva instancia EC2 en la región predeterminada. Recibirá como parámetros el ID de la imagen (AMI), el tipo de instancia, el ID de la subred a la que se conectará, si tendrá asociada una IP pública o elástica (parámetro yes|no), id del grupo de seguridad, y nombre de la clave pública a injectar. Devolverá el ID de la nueva instancia y su estado.

Para comprobar que cumple con las características:

```
aws ec2 describe-instances --instance-ids i0f4bace2083f21770
```

```
"Groups": [  
    {  
        "GroupName": "ssh-http-sg",  
        "GroupId": "sg-0d9f0888a02ec59f4"  
    }]
```

```
{  
    "Reservations": [  
        {  
            "Groups": [],  
            "Instances": [  
                {  
                    "AmiLaunchIndex": 0,  
                    "ImageId": "ami-04e5276ebb8451442",  
                    "InstanceId": "i-0f4bace2083f21770",  
                    "InstanceType": "t2.micro",  
                    "KeyName": "CEU-Keys",  
                    "LaunchTime": "2024-04-28T16:00:53+00:00",  
                    "Monitoring": {  
                        "State": "disabled"  
                    },  
                    "Placement": {  
                        "AvailabilityZone": "us-east-1c",  
                        "GroupName": "",  
                        "Tenancy": "default"  
                    },  
                    "PrivateDnsName": "ip-172-31-21-125.ec2.internal",  
                    "PrivateIpAddress": "172.31.21.125",  
                    "ProductCodes": [],  
                    "PublicDnsName": "ec2-3-83-248-180.compute-1.amazonaws.com",  
                    "PublicIpAddress": "3.83.248.180",  
                    "State": {  
                        "Code": 16,  
                        "Name": "running"  
                    },  
                    "StateTransitionReason": "",  
                    "SubnetId": "subnet-061ad182da4aa73cb",  
                    "VpcId": "vpc-00a5dfc0664ac615a",  
                    "Architecture": "x86_64",  
                    "BlockDeviceMappings": [  
                        {  
                            "DeviceName": "/dev/xvda",  
                            "Ebs": {  
                                "AttachTime": "2024-04-28T16:00:53+00:00",  
                                "DeleteOnTermination": true,  
                                "Status": "attached",  
                                "VolumeId": "vol-0e8fcf21ca82b8a9c"  
                            }  
                        }  
                    ]  
                }  
            ]  
        }  
    ]  
}
```

6. BOTO3. AWS SDK PARA PYTHON

D) Crear un script en Python para lanzar una nueva instancia EC2 en la región predeterminada. Recibirá como parámetros el ID de la imagen (AMI), el tipo de instancia, el ID de la subred a la que se conectará, si tendrá asociada una IP pública o elástica (parámetro yes|no), id del grupo de seguridad, y nombre de la clave pública a injectar. Devolverá el ID de la nueva instancia y su estado.

Vamos a comprobar que no añade una IP Pública si se indica

```
ami_id = 'ami-04e5276ebb8451442'
instance_type = 't2.micro'
subnet_id = 'subnet-061ad182da4aa73cb'
associate_public_ip = 'no'
security_group_id = 'sg-0d9f0888a02ec59f4'
key_name = 'CEU-Keys'
```

```
root@lapc01a:~/Desktop/aws$ python
ID de la instancia: i-0b8279113ca3210ba
Estado de la instancia: pending
```

```
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-04e5276ebb8451442",
                    "InstanceId": "i-0b8279113ca3210ba",
                    "InstanceType": "t2.micro",
                    "KeyName": "CEU-Keys",
                    "LaunchTime": "2024-04-28T16:10:06+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1c",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-31-223.ec2.internal",
                    "PrivateIpAddress": "172.31.31.223",
                    "ProductCodes": [],
                    "PublicDnsName": "",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "StateTransitionReason": "",
                    "SubnetId": "subnet-061ad182da4aa73cb",
                    "VpcId": "vpc-00a5dfc0664ac615a",
                    "Architecture": "x86_64",
                    "BlockDeviceMappings": [
                        {
                            "DeviceName": "/dev/xvda",
                            "Ebs": {
                                "AttachTime": "2024-04-28T16:10:06+00:00",
                                "DeleteOnTermination": true,
                                "Status": "attached",
                                "VolumeId": "vol-072fe2c26085cf6f6"
                            }
                        }
                    ]
                }
            ]
        }
    ]
}
```

6. BOTO3. AWS SDK PARA PYTHON

E) Crear un script en Python que nos indique el número de volúmenes, instantáneas, grupos de seguridad, balanceadores de carga y parejas de claves asignados a nuestra cuenta en la región en la que nos encontramos.

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 contar.py
Volúmenes: 2
Instantáneas: 1
Grupos de Seguridad: 4
Balanceadores de Carga: 0
Parejas de Claves: 3
```

```
import boto3

ec2_client = boto3.client('ec2')
elb_client = boto3.client('elbv2')

def contar_recursos_aws():

    volumenes = ec2_client.describe_volumes()
    cuenta_volumenes = len(volumenes['Volumes'])

    instantaneas = ec2_client.describe_snapshots(OwnerIds=['self'])
    cuenta_instantaneas = len(instantaneas['Snapshots'])

    grupos_seguridad = ec2_client.describe_security_groups()
    cuenta_grupos_seguridad = len(grupos_seguridad['SecurityGroups'])

    balanceadores_carga = elb_client.describe_load_balancers()
    cuenta_balanceadores_carga = len(balanceadores_carga['LoadBalancers'])

    parejas_claves = ec2_client.describe_key_pairs()
    cuenta_parejas_claves = len(parejas_claves['KeyPairs'])

    return {
        'Volúmenes': cuenta_volumenes,
        'Instantáneas': cuenta_instantaneas,
        'Grupos de Seguridad': cuenta_grupos_seguridad,
        'Balanceadores de Carga': cuenta_balanceadores_carga,
        'Parejas de Claves': cuenta_parejas_claves
    }

cuenta_recursos = contar_recursos_aws()

for recurso, cuenta in cuenta_recursos.items():
    print(f"{recurso}: {cuenta}")
```

6. BOTO3. AWS SDK PARA PYTHON

F) Crear un script en Python para eliminar (Terminate) la instancia o instancias cuyos ID se pasan como parámetro.

Creamos una instancia:

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 run-instances --image-id ami-04e5276ebb8451442 --count 1 --instance-type t2.micro --key-name CEU-Keys --security-group-ids sg-0d9f0888a02ec59f4 --placement AvailabilityZone=us-east-1a --query 'Instances[0].InstanceId' --region us-east-1
"i-08bf5112a50d6a91d"
```

```
ubuntupaloma@ubuntu:~/Desktop/aws$ aws ec2 describe-instances --instance-id i-08bf5112a50d6a91
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-04e5276ebb8451442",
                    "InstanceId": "i-08bf5112a50d6a91d",
                    "InstanceType": "t2.micro",
                    "KeyName": "CEU-Keys",
                    "LaunchTime": "2024-04-25T18:15:34+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-15-97.ec2.internal",
                    "PrivateIpAddress": "172.31.15.97",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-34-201-26-48.compute-1.amazonaws.com",
                    "PublicIpAddress": "34.201.26.48",
                    "State": {
                        "Code": 16,
                        "Name": "running"
                    },
                    "SubnetId": "subnet-00000000000000000000000000000000",
                    "VpcId": "vpc-00000000000000000000000000000000"
                }
            ]
        }
    ]
}
```

La instancia está activa

6. BOTO3. AWS SDK PARA PYTHON

F) Crear un script en Python para eliminar (Terminate) la instancia o instancias cuyos ID se pasan como parámetro.

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ python3 terminarInstancias.py
Terminating instance i-08bf5112a50d6a91d
```

```
ubuntupaloma@ubuntu:~/Desktop/aws/aws$ aws ec2 describe-instances --instance-id i-08bf5112a50d6a91d
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-04e5276ebb8451442",
                    "InstanceId": "i-08bf5112a50d6a91d",
                    "InstanceType": "t2.micro",
                    "KeyName": "CEU-Keys",
                    "LaunchTime": "2024-04-25T18:15:34+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-15-97.ec2.internal",
                    "PrivateIpAddress": "172.31.15.97",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-34-201-26-48.compute-1.amazonaws.com",
                    "PublicIpAddress": "34.201.26.48",
                    "State": {
                        "Code": 32,
                        "Name": "shutting-down"
                    }
                }
            ]
        }
    ]
}
```

La instancia está terminada