Gestión de Usuarios IAM con Python

Objetivos de la práctica

1. Crear un usuario IAM
2. Crear claves de acceso para el usuario
3. Adjuntar una política gestionada al usuario
4. Crear un grupo IAM y agregar el usuario
5. Crear una política personalizada y adjuntarla al grupo
6. Listar usuarios, grupos y políticas

In [ ]:
```python
import boto3
import os

# Crear una sesión de boto3
session = boto3.Session(
    aws_access_key_id = os.getenv('AWS_ACCESS_KEY_ID'),
    aws_secret_access_key = os.getenv("AWS_SECRET_ACCESS_KEY"),
    region_name = 'us-east-1',
)

# Crear un cliente de IAM
iam_client = session.client('iam')
```

1. Crear un usuario IAM

In [ ]:
```python
# Crear un usuario de IAM
nuevo_usuario = 'nuevo_usuario'
response = iam_client.create_user(UserName = nuevo_usuario)
print(f"Usuario {nuevo_usuario} creado: {response}")
```

```
Usuario nuevo_usuario creado: {'User': {'Path': '/', 'UserName': 'nuevo_usuario', 'UserId': 'AIDAQ3EG
UCWSX5PY3EQIT', 'Arn': 'arn:aws:iam::058264393125:user/nuevo_usuario', 'CreateDate': datetime.datetim
e(2024, 6, 20, 17, 13, 18, tzinfo=tzutc())}, 'ResponseMetadata': {'RequestId': '64380a21-c566-473a-91
40-41421efff515', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:13:17 GMT', 'x-
amzn-requestid': '64380a21-c566-473a-9140-41421efff515', 'content-type': 'text/xml', 'content-lengt
h': '487'}, 'RetryAttempts': 0}}
```

2. Crear Claves de Acceso para el Usuario

In [ ]:
```python
# Crear claves de acceso para nuevo_usuario
access_key_response = iam_client.create_access_key(UserName = nuevo_usuario)
access_key_id = access_key_response['AccessKey']['AccessKeyId']
secret_access_key = access_key_response['AccessKey']['SecretAccessKey']

print('Access Key Id: {}'.format(access_key_id))
print('Secret Access Key: {}'.format(secret_access_key))
```

```
Access Key Id: AKIAQ3EGUCWSSCJ6FG5E
Secret Access Key: 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
```

3. Adjuntar una política gestionada al usuario

In [ ]:
```
# Adjuntar una política gestionada al usuario
policy_arn = 'arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess'
response = iam_client.attach_user_policy(
    UserName = nuevo_usuario,
    PolicyArn = policy_arn
)

print("Política {} adjuntada al usuario {}: {}".format(policy_arn,
                                                       nuevo_usuario,
                                                       response))
```

```
Política arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess adjuntada al usuario nuevo_usuario: {'Respons
eMetadata': {'RequestId': '7967e5fa-a24d-4c0e-8381-1f8eaa9d4b58', 'HTTPStatusCode': 200, 'HTTPHeader
s': {'date': 'Thu, 20 Jun 2024 17:13:26 GMT', 'x-amzn-requestid': '7967e5fa-a24d-4c0e-8381-1f8eaa9d4b
58', 'content-type': 'text/xml', 'content-length': '212'}, 'RetryAttempts': 0}}
```

4. Crear un grupo IAM y agregar el usuario

In [ ]:
```
# Crear un nuevo grupo
group_name = "nuevo_grupo"
response = iam_client.create_group(GroupName = group_name)
print("Grupo {} creado: {}".format(group_name, response))
```

```
Grupo nuevo_grupo creado: {'Group': {'Path': '/', 'GroupName': 'nuevo_grupo', 'GroupId': 'AGPAQ3EGUCW
SYIMJEUIUX', 'Arn': 'arn:aws:iam::058264393125:group/nuevo_grupo', 'CreateDate': datetime.datetime(20
24, 6, 20, 17, 13, 37, tzinfo=tzutc())}, 'ResponseMetadata': {'RequestId': 'f379c1bb-1d45-46ac-9a70-8
969849f5f51', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:13:37 GMT', 'x-amzn
-requestid': 'f379c1bb-1d45-46ac-9a70-8969849f5f51', 'content-type': 'text/xml', 'content-length': '4
94'}, 'RetryAttempts': 0}}
```

In [ ]:
```
# Adjuntar al usuario al nuevo grupo
response = iam_client.add_user_to_group(
    GroupName = group_name,
    UserName = nuevo_usuario
)

print("Usuario {} agregado al grupo {}: {}".format(nuevo_usuario,
                                                   group_name,
                                                   response))
```

```
Usuario nuevo_usuario agregado al grupo nuevo_grupo: {'ResponseMetadata': {'RequestId': '6c874b7e-ee3
d-4916-86cd-e934e9a7d9a9', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:13:40
GMT', 'x-amzn-requestid': '6c874b7e-ee3d-4916-86cd-e934e9a7d9a9', 'content-type': 'text/xml', 'conten
t-length': '208'}, 'RetryAttempts': 0}}
```

5. Crear una política personalizada y adjuntarla al grupo

Se definirá una política simple que permita listar los usuarios de IAM

In [ ]:
```
# Definir el documento de política para listar usuarios de IAM
policy_document = {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

In [ ]:
```
# Crear la política personalizada
import json

policy_name = 'CustomIAMListPolicy'
response = iam_client.create_policy(
    PolicyName = policy_name,
    PolicyDocument = json.dumps(policy_document)
)
policy_arn = response['Policy']['Arn']
print("Política personalizada {} creada: {}".format(policy_arn,
                                                    response))
```

```
Política personalizada arn:aws:iam::058264393125:policy/CustomIAMListPolicy creada: {'Policy': {'Poli
cyName': 'CustomIAMListPolicy', 'PolicyId': 'ANPAQ3EGUCWS35MAJVTAY', 'Arn': 'arn:aws:iam::05826439312
5:policy/CustomIAMListPolicy', 'Path': '/', 'DefaultVersionId': 'v1', 'AttachmentCount': 0, 'Permissi
onsBoundaryUsageCount': 0, 'IsAttachable': True, 'CreateDate': datetime.datetime(2024, 6, 20, 17, 13,
48, tzinfo=tzutc()), 'UpdateDate': datetime.datetime(2024, 6, 20, 17, 13, 48, tzinfo=tzutc())}, 'Resp
onseMetadata': {'RequestId': 'b531f6f0-c699-4119-8186-16f7abe0c378', 'HTTPStatusCode': 200, 'HTTPHead
ers': {'date': 'Thu, 20 Jun 2024 17:13:47 GMT', 'x-amzn-requestid': 'b531f6f0-c699-4119-8186-16f7abe0
c378', 'content-type': 'text/xml', 'content-length': '773'}, 'RetryAttempts': 0}}
```

In [ ]:
```
# Adjuntar la política personalizada al grupo
response = iam_client.attach_group_policy(
    GroupName = group_name,
    PolicyArn = policy_arn
)

print("Política {} adjuntada al grupo {}: {}".format(policy_arn,
                                                     group_name,
                                                     response))
```

```
Política arn:aws:iam::058264393125:policy/CustomIAMListPolicy adjuntada al grupo nuevo_grupo: {'Respo
nseMetadata': {'RequestId': '69e6f9fb-dd05-4597-8e0f-d33c5aa49bab', 'HTTPStatusCode': 200, 'HTTPHeade
rs': {'date': 'Thu, 20 Jun 2024 17:13:51 GMT', 'x-amzn-requestid': '69e6f9fb-dd05-4597-8e0f-d33c5aa49
bab', 'content-type': 'text/xml', 'content-length': '214'}, 'RetryAttempts': 0}}
```

6. Listar usuarios, grupos y políticas

In [ ]:
```
# Listar usuarios
response = iam_client.list_users()
print("Usuarios IAM {}".format(response['Users']))
```

```
Usuarios IAM [{'Path': '/', 'UserName': 'nuevo_usuario', 'UserId': 'AIDAQ3EGUCWSX5PY3EQIT', 'Arn': 'a
rn:aws:iam::058264393125:user/nuevo_usuario', 'CreateDate': datetime.datetime(2024, 6, 20, 17, 13, 1
8, tzinfo=tzutc())}]
```

In [ ]:
```
# Listar grupos
response = iam_client.list_groups()
print("Grupos IAM: {}".format(response['Groups']))
```

```
Grupos IAM: [{'Path': '/', 'GroupName': 'nuevo_grupo', 'GroupId': 'AGPAQ3EGUCWSYIMJEUIUX', 'Arn': 'ar
n:aws:iam::058264393125:group/nuevo_grupo', 'CreateDate': datetime.datetime(2024, 6, 20, 17, 13, 37,
tzinfo=tzutc())}]
```

In [ ]:
```
# Listar políticas
response = iam_client.list_policies(Scope = 'Local')
print("Políticas personalizadas: {}".format(response['Policies']))
```

```
Políticas personalizadas: [{'PolicyName': 'CustomIAMListPolicy', 'PolicyId': 'ANPAQ3EGUCWS35MAJVTAY',
'Arn': 'arn:aws:iam::058264393125:policy/CustomIAMListPolicy', 'Path': '/', 'DefaultVersionId': 'v1',
'AttachmentCount': 1, 'PermissionsBoundaryUsageCount': 0, 'IsAttachable': True, 'CreateDate': datetim
e.datetime(2024, 6, 20, 17, 13, 48, tzinfo=tzutc()), 'UpdateDate': datetime.datetime(2024, 6, 20, 17,
13, 48, tzinfo=tzutc())}]
```

Pasos finales

Se procedera a eliminar las políticas, grupos y usuarios creados.

In [ ]:
```
# Eliminar la política del grupo
iam_client.detach_group_policy(
    GroupName = group_name,
    PolicyArn = policy_arn
)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '4fc579aa-d792-41e9-a492-3c95a8c05790',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:06 GMT',
   'x-amzn-requestid': '4fc579aa-d792-41e9-a492-3c95a8c05790',
   'content-type': 'text/xml',
   'content-length': '214'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar la política
iam_client.delete_policy(PolicyArn = policy_arn)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '3656d9b2-40e5-41dd-a5f6-0ec1a5b436d7',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:11 GMT',
   'x-amzn-requestid': '3656d9b2-40e5-41dd-a5f6-0ec1a5b436d7',
   'content-type': 'text/xml',
   'content-length': '204'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar al usuario del grupo
iam_client.remove_user_from_group(
    GroupName = group_name,
    UserName = nuevo_usuario
)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '13074312-d70d-451c-a5f6-ad01300c9313',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:13 GMT',
   'x-amzn-requestid': '13074312-d70d-451c-a5f6-ad01300c9313',
   'content-type': 'text/xml',
   'content-length': '218'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar la política gestionada del usuario
iam_client.detach_user_policy(
    UserName = nuevo_usuario,
    PolicyArn = 'arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess'
)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '088be5cd-4c9d-429c-a8ec-543d6c3a94b2',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:16 GMT',
   'x-amzn-requestid': '088be5cd-4c9d-429c-a8ec-543d6c3a94b2',
   'content-type': 'text/xml',
   'content-length': '212'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar claves de acceso del usuario
iam_client.delete_access_key(
    UserName = nuevo_usuario,
    AccessKeyId = access_key_id
)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '2840540f-d849-41fd-9b92-d6ddfa8cea66',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:18 GMT',
   'x-amzn-requestid': '2840540f-d849-41fd-9b92-d6ddfa8cea66',
   'content-type': 'text/xml',
   'content-length': '210'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar al usuario
iam_client.delete_user(UserName = nuevo_usuario)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '368c0c47-63bc-4626-9bbd-db04a46645b7',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:22 GMT',
   'x-amzn-requestid': '368c0c47-63bc-4626-9bbd-db04a46645b7',
   'content-type': 'text/xml',
   'content-length': '200'},
  'RetryAttempts': 0}}
```

In [ ]:
```
# Eliminar al grupo
iam_client.delete_group(GroupName = group_name)
```

Out[ ]:
```
{'ResponseMetadata': {'RequestId': '6332ed28-93a9-4239-957a-c2d73faf57c7',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'date': 'Thu, 20 Jun 2024 17:14:25 GMT',
   'x-amzn-requestid': '6332ed28-93a9-4239-957a-c2d73faf57c7',
   'content-type': 'text/xml',
   'content-length': '202'},
  'RetryAttempts': 0}}
```