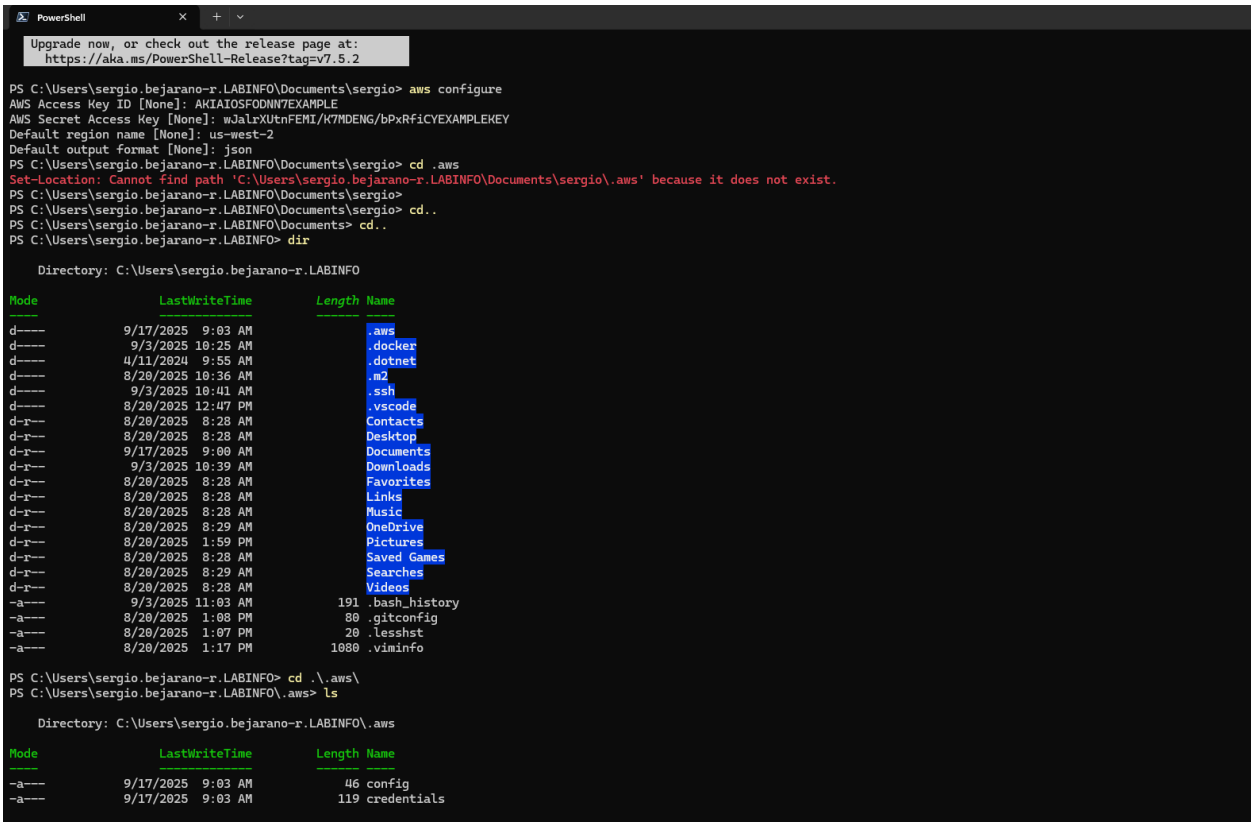
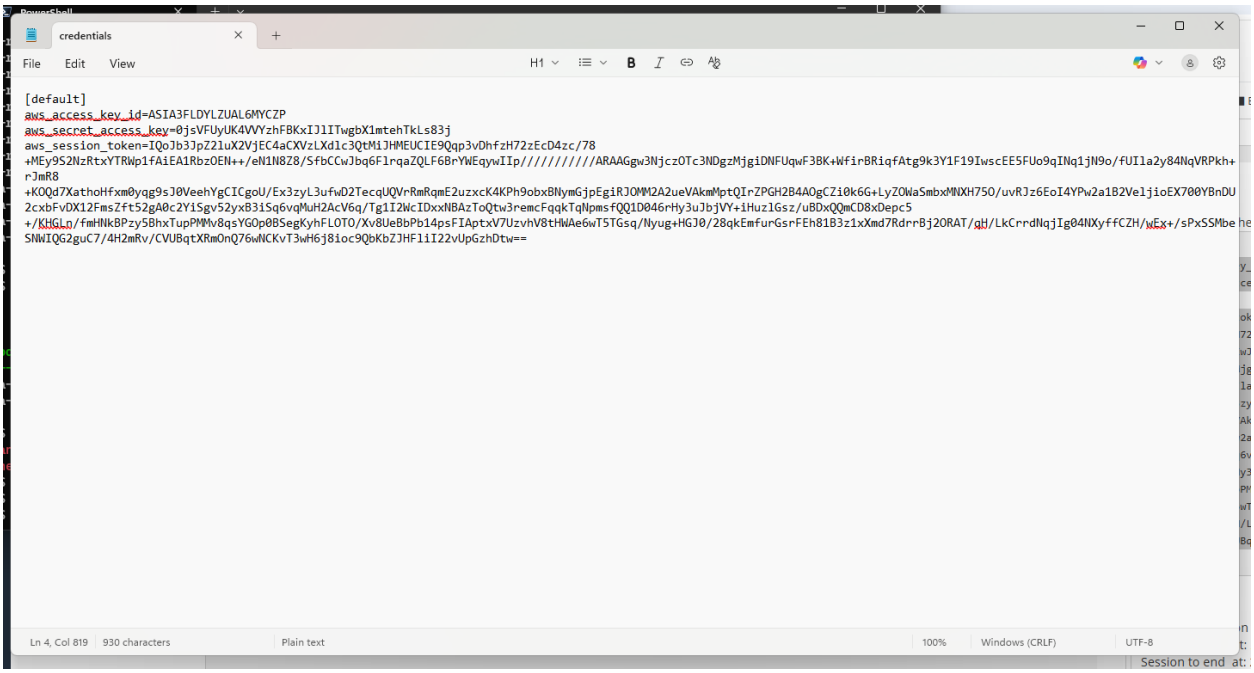
Report - How to use AWS-CLI to create an EC2 instance?

Author: Sergio Bejarano

First, locate the hidden aws folder and edit the respective files to define the correct credentials and configuration.

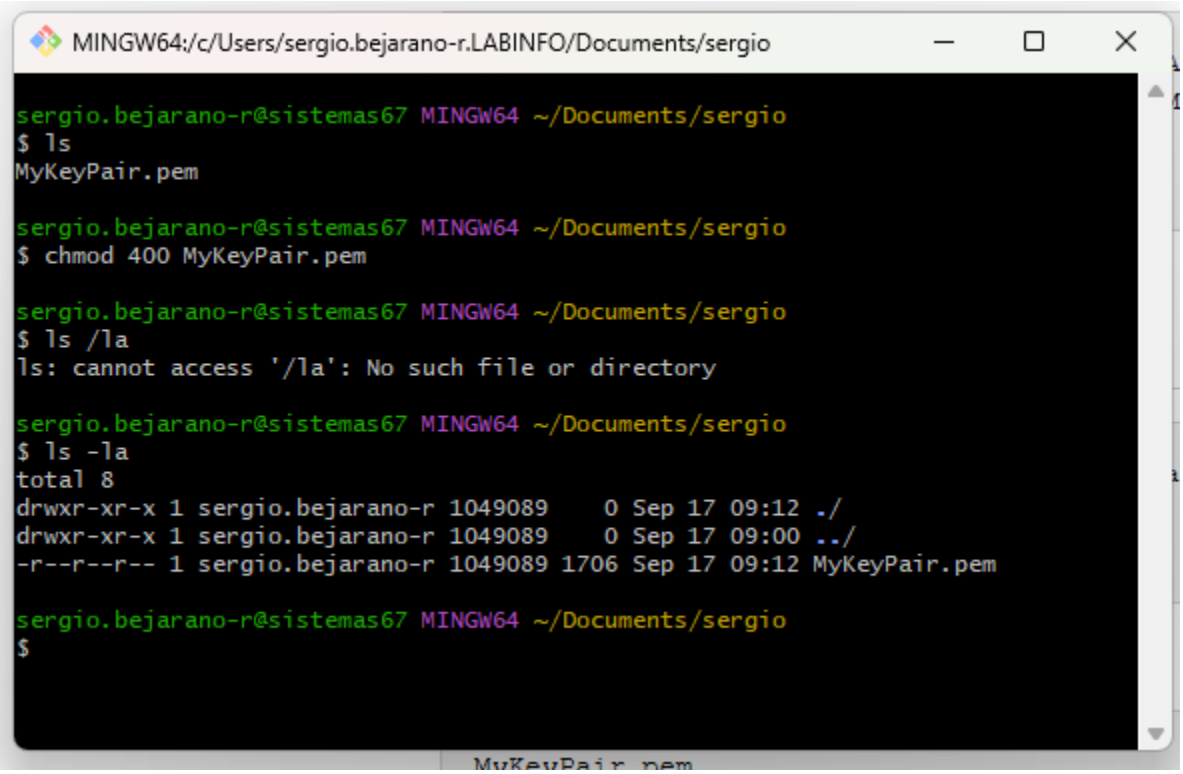




Step 1: Create a Key Pair for EC2

```
PS C:\Users\sergio.bejarano-r.LABINFO\Documents\sergio> aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text > MyKeyPair.pem
PS C:\Users\sergio.bejarano-r.LABINFO\Documents\sergio> ls

    Directory: C:\Users\sergio.bejarano-r.LABINFO\Documents\sergio

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a---           9/17/2025  9:12 AM           1706 MyKeyPair.pem

PS C:\Users\sergio.bejarano-r.LABINFO\Documents\sergio>
```

Make the private key readable only by myself:

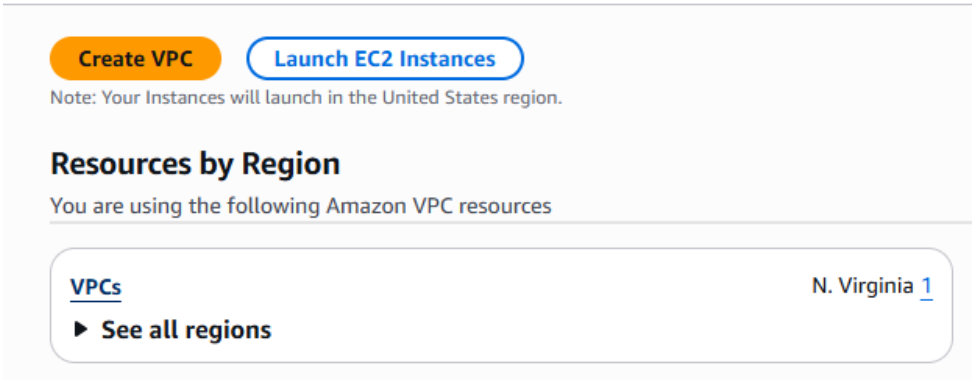


Check the fingerprint:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ ls -la
total 8
drwxr-xr-x 1 sergio.bejarano-r 1049089    0 Sep 17 09:12 ./
drwxr-xr-x 1 sergio.bejarano-r 1049089    0 Sep 17 09:00 ../
-r--r--r-- 1 sergio.bejarano-r 1049089 1706 Sep 17 09:12 MyKeyPair.pem

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 describe-key-pairs --key-name MyKeyPair
{
    "KeyPairs": [
        {
            "KeyPairId": "key-01faad2c8e72eba38",
            "KeyFingerprint": "f0:76:30:86:a2:64:dc:75:17:6f:a5:36:18:a2:64:fb:3
2:df:bf:fe",
            "KeyName": "MyKeyPair",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2025-09-17T14:12:44.239000+00:00"
        }
    ]
}


sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$
```

Step 2: Create a Security Group

First, check for VPCs configured in the account.




```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 create-security-group --group-name my-sg-cli --description "My security group" --vpc-id vpc-050895a691e406167
{
    "GroupId": "sg-0af0eb3614538d6f6"
}


sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$
```

List security groups:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 describe-security-groups --group-ids sg-0af0eb3614538d6f6
{
    "SecurityGroups": [
        {
            "Description": "My security group",
            "GroupName": "my-sg-cli",
            "IpPermissions": [],
            "OwnerId": "767397748328",
            "GroupId": "sg-0af0eb3614538d6f6",
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "Ipv6Ranges": [],
                    "PrefixListIds": [],
                    "UserIdGroupPairs": []
                }
            ],
            "VpcId": "vpc-050895a691e406167"
        }
    ]
}

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
```

Add Ingress Rules

Check your public IP address (optional for restricted access):

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ curl https://checkip.amazonaws.com
45.239.88.82
```

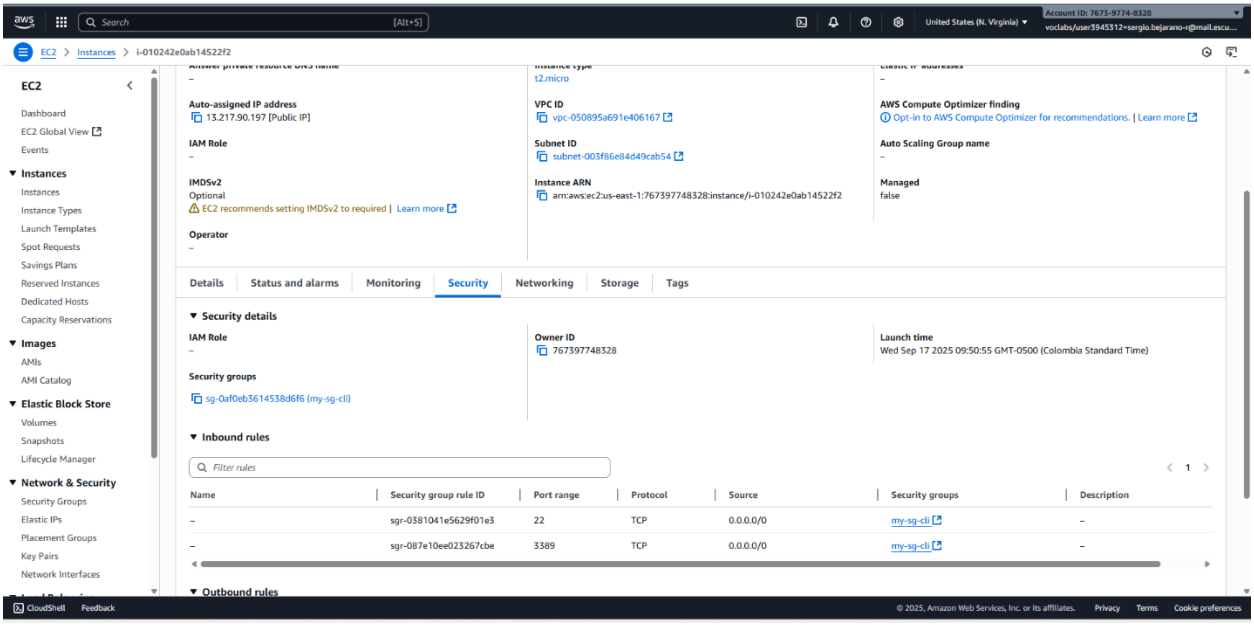
Allow RDP (port 3389):

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 authorize-security-group-ingress --group-id sg-0af0eb3614538d6f6 --protocol tcp --port 3389 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-087e10ee023267cbe",
            "GroupId": "sg-0af0eb3614538d6f6",
            "GroupOwnerId": "767397748328",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 3389,
            "ToPort": 3389,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Allow SSH (port 22):

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 authorize-security-group-ingress --group-id sg-0af0eb3614538d6f6 --protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0381041e5629f01e3",
            "GroupId": "sg-0af0eb3614538d6f6",
            "GroupOwnerId": "767397748328",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
```

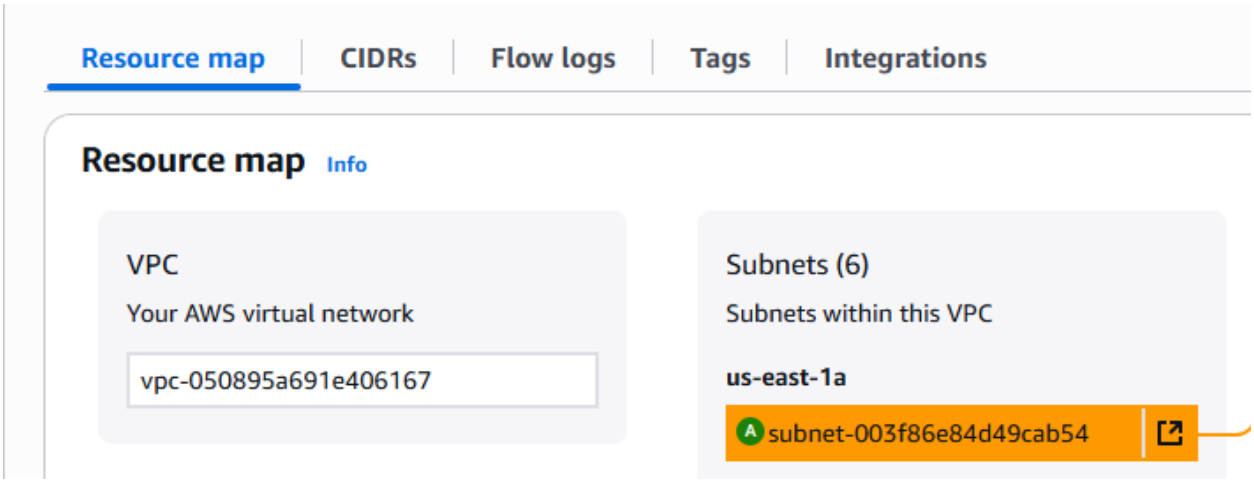
Verifying:



Step 3: Create the Instance

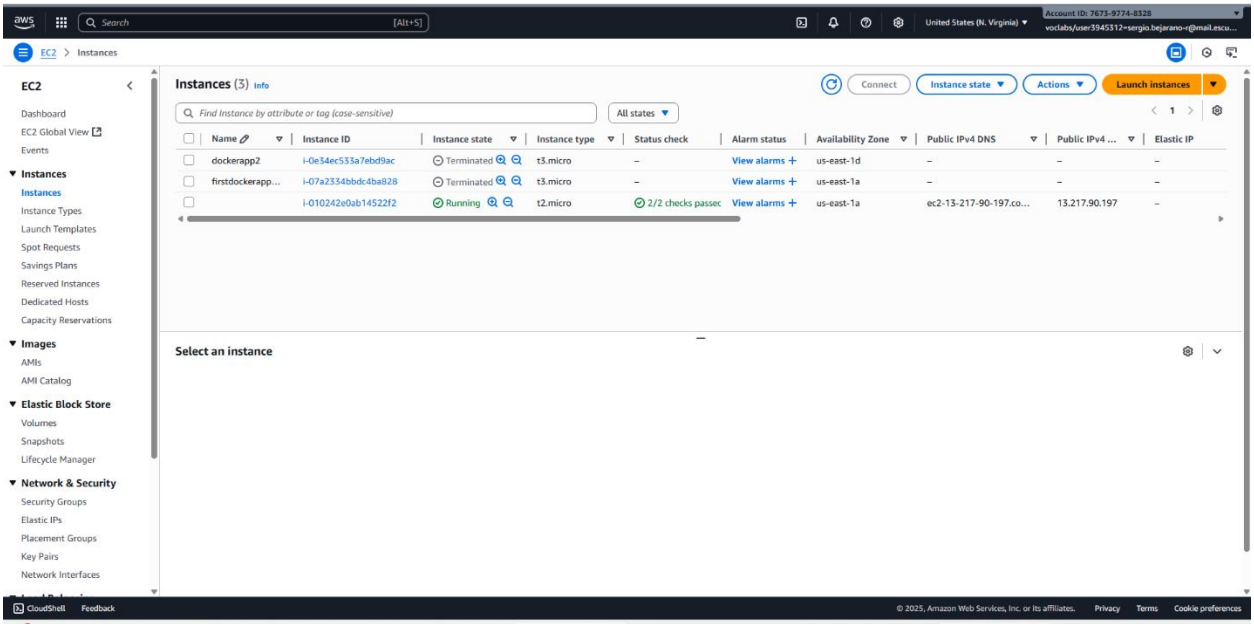
Before creating the instance, ensure you have a subnet configured.

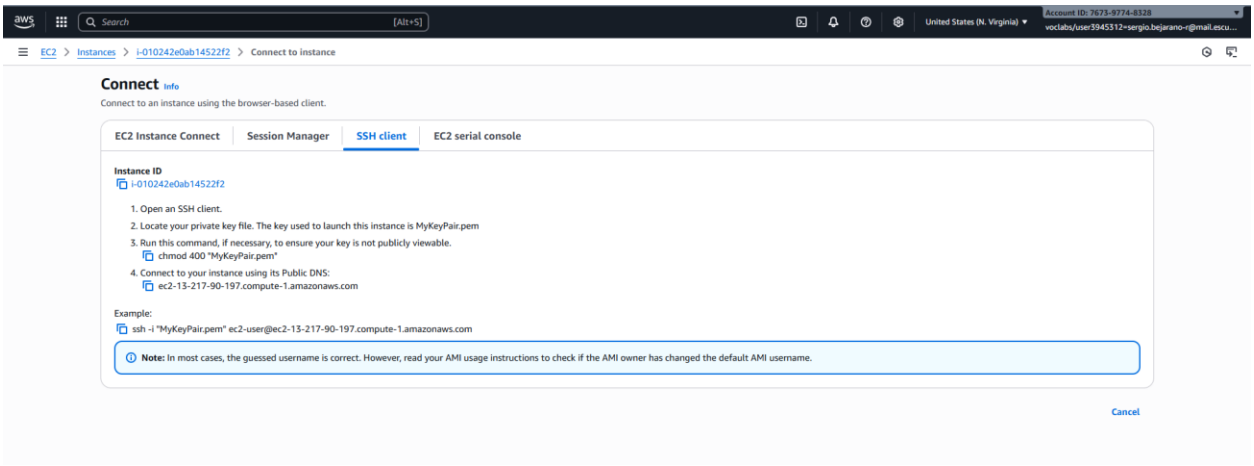
Run the following command to launch a **t2.micro** instance:



```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 run-instances --image-id ami-032930428bf1abbff --count 1 --instance-type t2.micro --key-name MyKeyPair --security-group-ids
sg-0af0eb3614538d6f6 --subnet-id subnet-003f86e84d49cab54
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-032930428bf1abbff",
            "InstanceId": "i-010242e0ab14522f2",
            "InstanceType": "t2.micro",
            "KeyName": "MyKeyPair",
            "LaunchTime": "2025-09-17T14:50:55+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-east-1a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-172-31-47-63.ec2.internal",
            "PrivateIpAddress": "172.31.47.63",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "StateTransitionReason": "",
            "SubnetId": "subnet-003f86e84d49cab54",
            "VpcId": "vpc-050895a691e406167",
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "d95b5af4-deb3-41e1-9748-9b9f99eab116",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2025-09-17T14:50:55+00:00",
                        "AttachmentId": "eni-attach-06721f36c623cd80c",
                        "DeleteOnTermination": true,
```

Step 4: Connect to the Instance






```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ ssh -i "MyKeyPair.pem" ec2-user@ec2-13-217-90-197.compute-1.amazonaws.com
The authenticity of host 'ec2-13-217-90-197.compute-1.amazonaws.com (13.217.90.197)' can't be established.
ED25519 key fingerprint is SHA256:4EqJ/EmpeirKKmJjc/jPJpffMvGoJUCd2Fl+7r49zyY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-217-90-197.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
27 package(s) needed for security, out of 44 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-47-63 ~]$
```

Step 5: List Your Instances

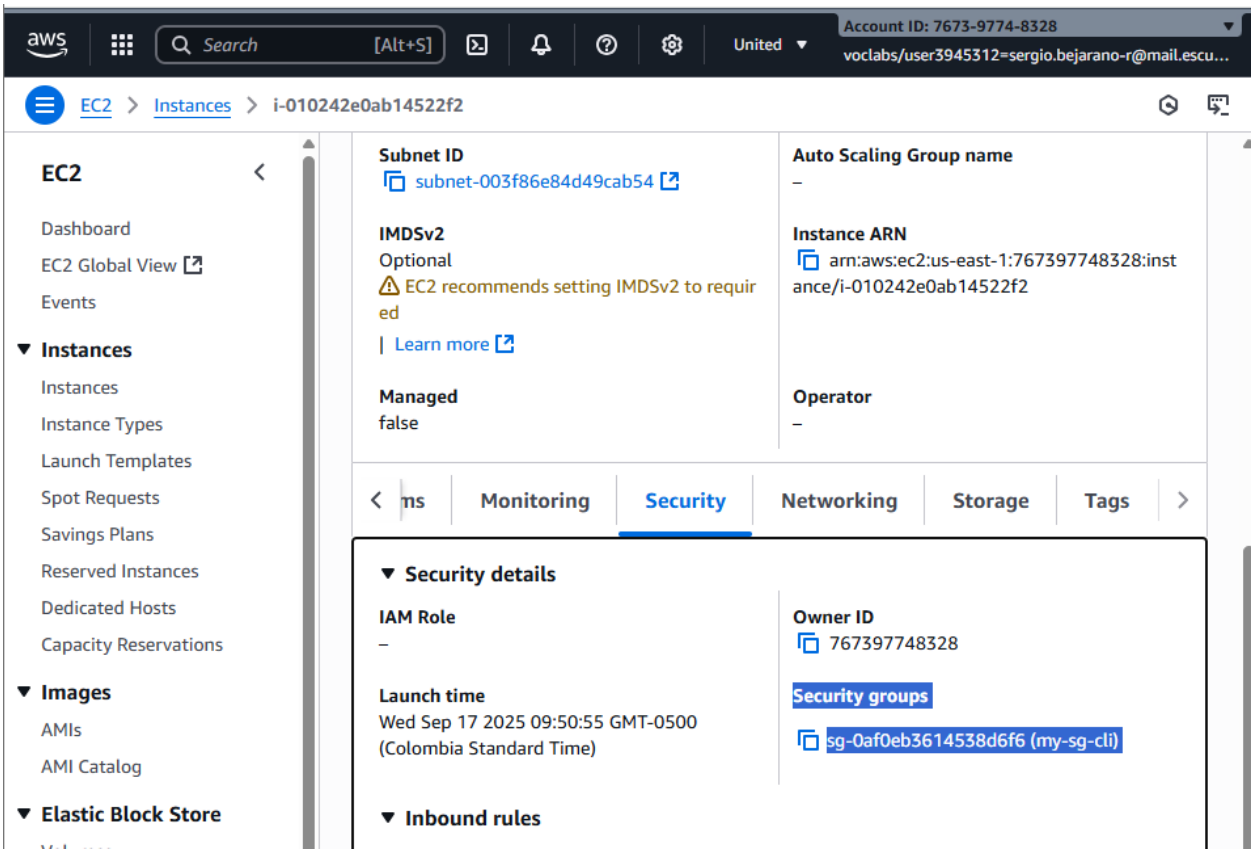
```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query "Reservations[].Instances[].InstanceId"
[
    "i-010242e0ab14522f2"
]


sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$
```

Step 6: Clean Up

Delete the key pair:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 delete-key-pair --key-name MyKeyPair
{
    "Return": true,
    "KeyPairId": "key-01faad2c8e72eba38"
}
```

Delete the security group:



An error occurred:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 delete-security-group --group-id sg-0af0eb3614538d6f6

An error occurred (DependencyViolation) when calling the DeleteSecurityGroup operation: resource sg-0af0eb3614538d6f6 has a dependent
object

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ ^C
```

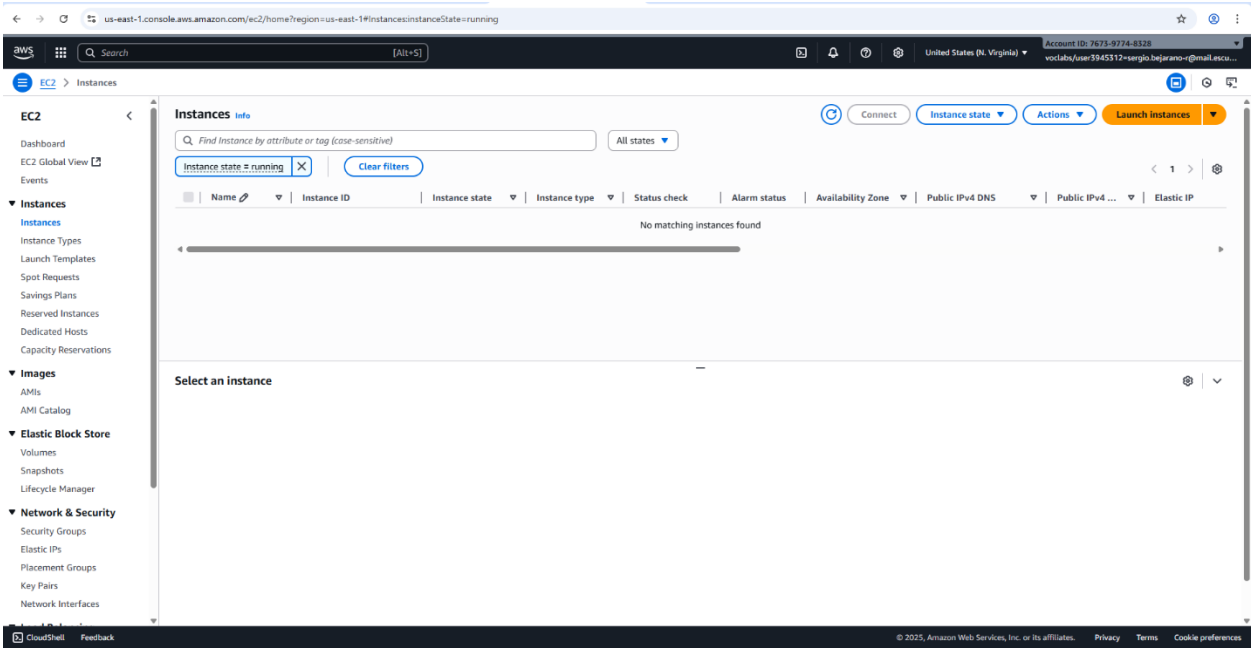
Delete the instance:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 terminate-instances --instance-ids i-010242e0ab14522f2
{
    "TerminatingInstances": [
        {
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "InstanceId": "i-010242e0ab14522f2",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$
```

Now the security group is deleted:

```
sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$ aws ec2 delete-security-group --group-id sg-0af0eb3614538d6f6

sergio.bejarano-r@sistemas67 MINGW64 ~/Documents/sergio
$
```

Verifying:


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#instances:instanceState=running
Search
[Alt+S]
United States (N. Virginia)
Account ID: 7673-9774-8328
voclabs/user3945312=sergio.bejarano-r@mail.escu...
EC2 > Instances
EC2
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Instances Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Instance state = running
Clear filters
1
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
No matching instances found
Select an instance
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences