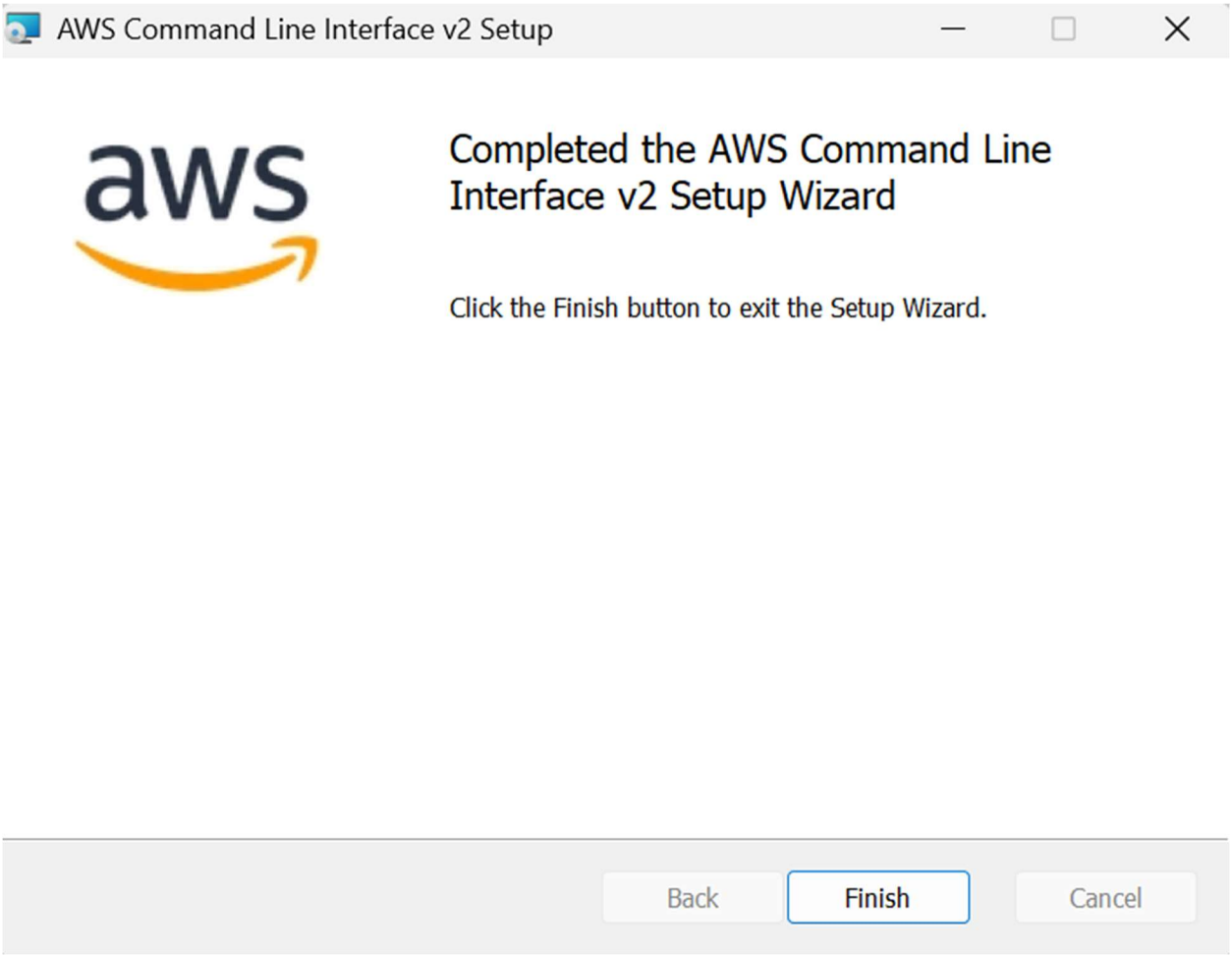
AWS-CLI Primer Workshop

Student: Cristian David Polo Garrido

In this workshop, you will learn how to use AWS-CLI to create an EC2 instance.

Prerequisites

1. Install the AWS-CLI application.



2. Configure the AWS-CLI application:

aws configure

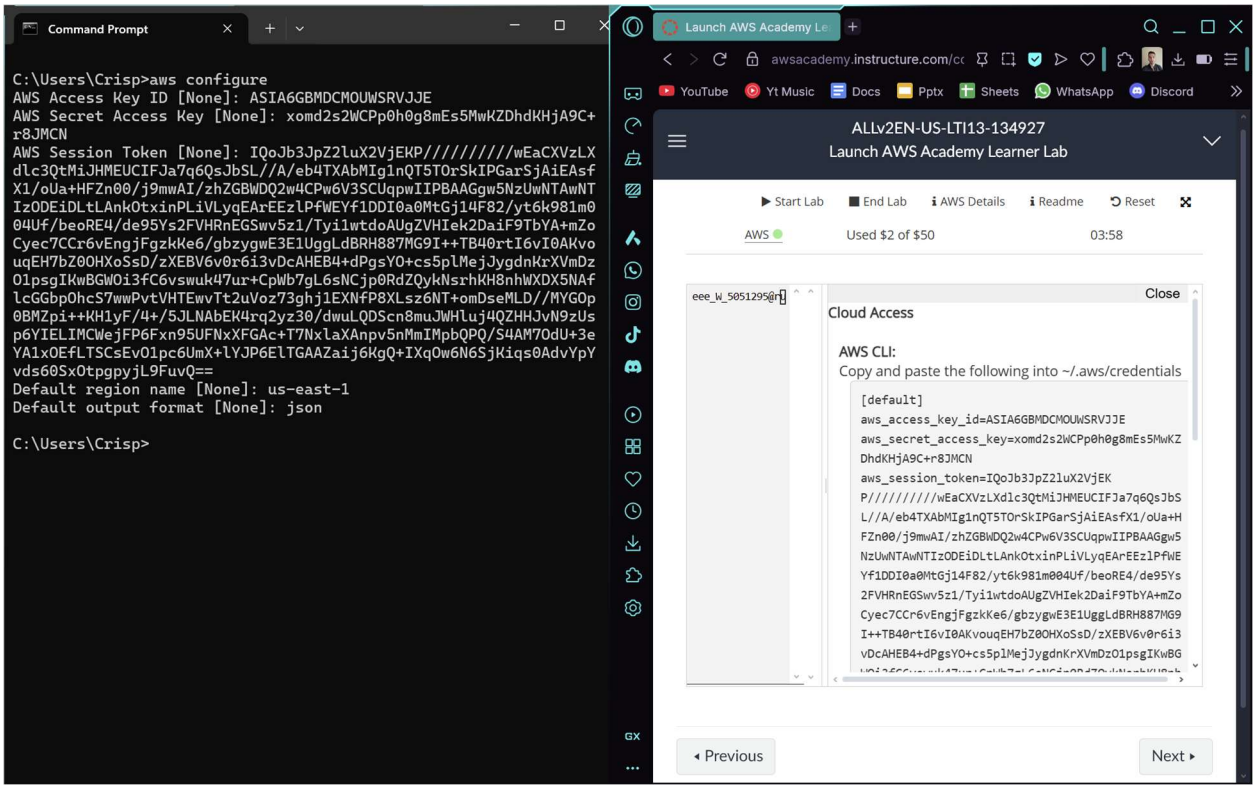
Example configuration:

AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE

AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

Default region name [None]: us-west-2

Default output format [None]: json



Step 1: Create a Key Pair for EC2

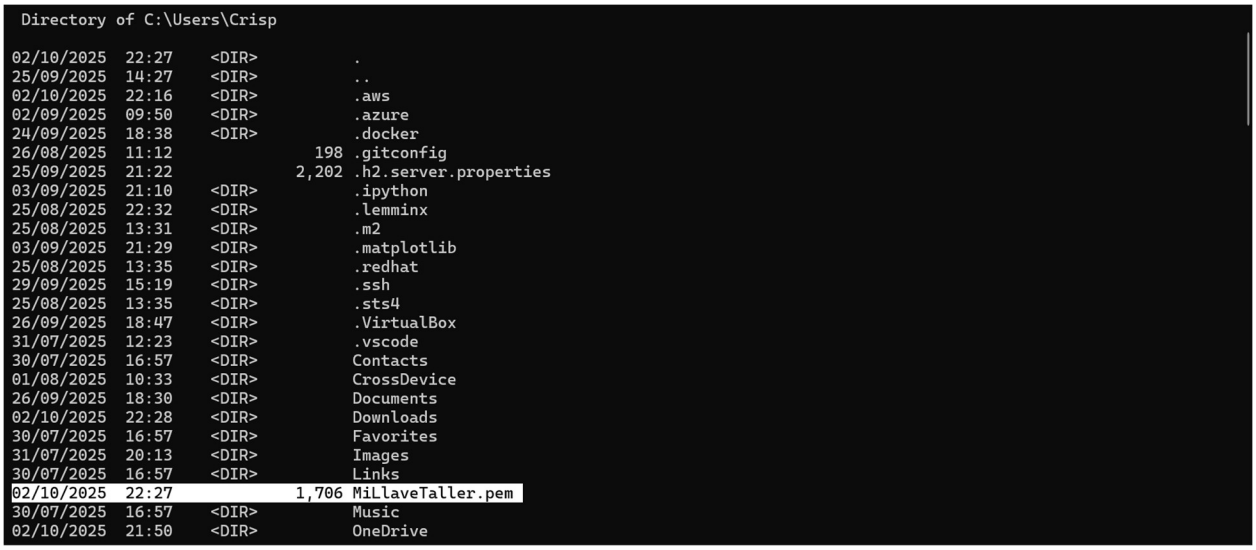
aws ec2 create-key-pair --key-name MyKeyPair --query 'KeyMaterial' --output text > MyKeyPair.pem

```
C:\Users\Crisp>aws ec2 create-key-pair --key-name MiLlaveTaller --query "KeyMaterial" --output text > MiLlaveTaller.pem
```

ls

Output:

MyKeyPair.pem



Make the private key readable only by you:

chmod 400 MyKeyPair.pem

ls -la

Output:

-r-------- 1 user staff 1675 Oct 9 20:39 MyKeyPair.pem

Check the fingerprint:

aws ec2 describe-key-pairs --key-name MyKeyPair

Step 2: Create a Security Group

First, check for VPCs configured in your account.

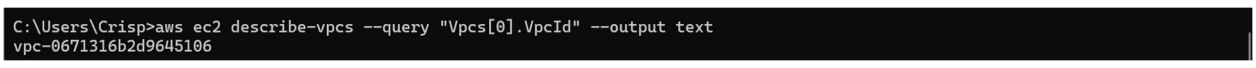
aws ec2 create-security-group --group-name my-sg-cli --description "My security group" --vpc-id vpc-xxxxxxxx

Example output:

{

"GroupId": "sg-01f4c77b81e9dc434"

}



```
C:\Users\Crisp>aws ec2 create-security-group --group-name taller --description "Grupo para taller" --vpc-id vpc-0671316b2d964510
6
{
    "GroupId": "sg-06515a1bacbfac7dd",
    "SecurityGroupArn": "arn:aws:ec2:us-east-1:975050052381:security-group/sg-06515a1bacbfac7dd"
}
```

List security groups:

aws ec2 describe-security-groups --group-ids sg-01f4c77b81e9dc434

Add Ingress Rules

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

curl https://checkip.amazonaws.com

Example output:

186.96.109.58

Allow RDP (port 3389):

aws ec2 authorize-security-group-ingress --group-id sg-01f4c77b81e9dc434 --protocol tcp --port 3389 --cidr 0.0.0.0/0

```
C:\Users\Crisp>aws ec2 authorize-security-group-ingress --group-id sg-06515a1bacbfac7dd --protocol tcp --port 80 --cidr 0.0.0.0/
0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-02d044de5f847d54d",
            "GroupId": "sg-06515a1bacbfac7dd",
            "GroupOwnerId": "975050052381",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0",
            "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:975050052381:security-group-rule/sgr-02d044de5f847d54d"
        }
    ]
}
```

Allow SSH (port 22):

aws ec2 authorize-security-group-ingress --group-id sg-01f4c77b81e9dc434 --protocol tcp --port 22 --cidr 0.0.0.0/0

```
C:\Users\Crisp>aws ec2 authorize-security-group-ingress --group-name taller --protocol tcp --port 22 --cidr 0.0.0.0/0
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-0188905552bdf33e0",
            "GroupId": "sg-06515a1bacbfac7dd",
            "GroupOwnerId": "975050052381",
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIpv4": "0.0.0.0/0",
            "SecurityGroupRuleArn": "arn:aws:ec2:us-east-1:975050052381:security-group-rule/sgr-0188905552bdf33e0"
        }
    ]
}
```

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:

aws ec2 run-instances --image-id ami-032930428bf1abbff --count 1 --instance-type t2.micro --key-name MyKeyPair --security-group-ids sg-01f4c77b81e9dc434 --subnet-id subnet-1175cf1d

```
C:\Users\Crisp>aws ec2 run-instances --image-id ami-0c02fb55956c7d316 --count 1 --instance-type t2.micro --key-name MiLlaveTalle
r --security-group-ids sg-06515a1bacbfac7dd --subnet-id subnet-024fd8f5538e3d0f3
{
    "ReservationId": "r-0efcfa5cc6e7d2e56",
    "OwnerId": "975050052381",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "63ca39ac-6d6f-4ddc-b97b-a3eef26ad945",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachTime": "2025-10-03T03:36:32+00:00",
                        "AttachmentId": "eni-attach-0e3df54a2ae1e2c6d",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupId": "sg-06515a1bacbfac7dd",
                            "GroupName": "taller"
                        }
                    ],
```

To know the subnets, I typed this command:

```
C:\Users\Crisp>aws ec2 describe-subnets --query "Subnets[*].SubnetId" --output text
subnet-024fd8f5538e3d0f3        subnet-081fe58c630ffbf46        subnet-0bbd850af021a8535        subnet-0f33da10e0639be46
subnet-04a51381d560e534e        subnet-0b62e0b25a34c9c15
```

Now I list the instances that are running, seeing that indeed the one I just created is up and running:

```
C:\Users\Crisp>aws ec2 describe-instances --filters "Name=instance-state-name,Values=running" --query "Reservations[*].Instances
[*].[InstanceId,PublicIpAddress]" --output table
-----------------------------------------
|           DescribeInstances           |
+----------------------+----------------+
|  i-0390c0612aef2adf0 |  54.226.199.38 |
|  i-0edaa4badf79523ca |  54.174.31.98  |
|  i-0a8914ab6985c8962 |  23.22.49.189  |
|  i-0b3c36b99856ac69e |  98.84.162.28  |
+----------------------+----------------+
```

Step 4: Connect to the Instance

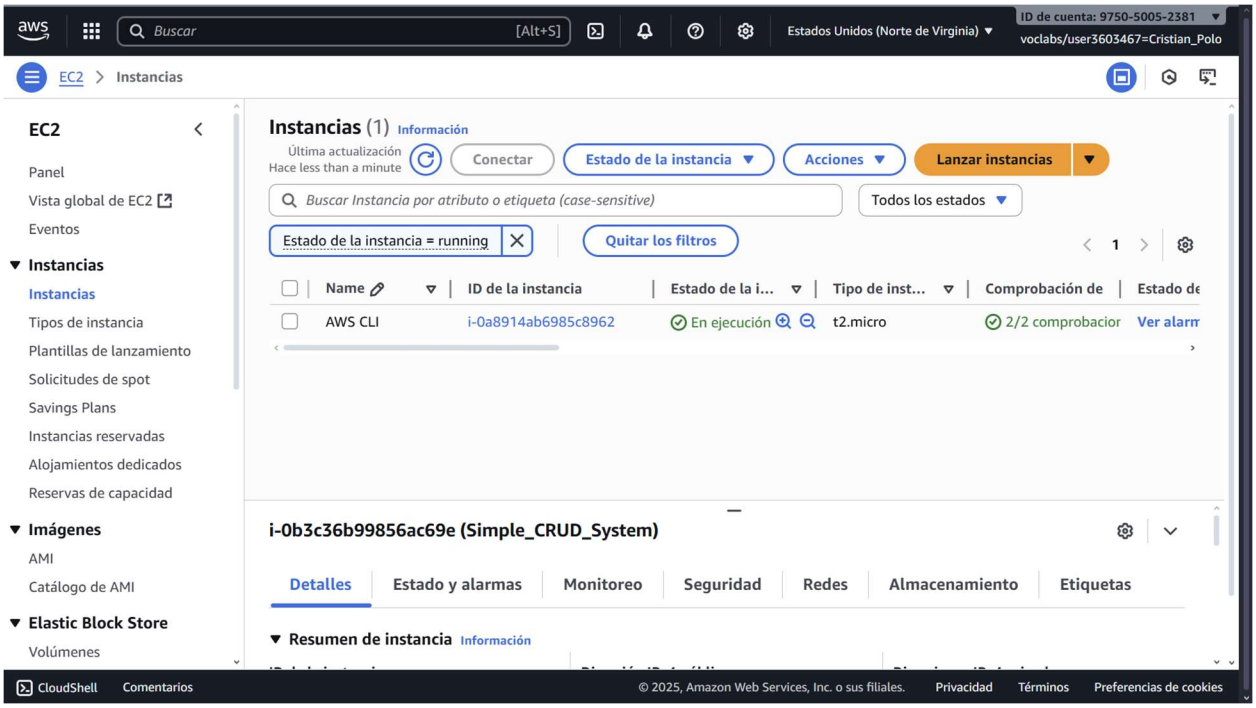
ssh -i "MyKeyPair.pem" ec2-user@ec2-34-204-197-22.compute-1.amazonaws.com

```
C:\Users\Crisp>ssh -i "MiLlaveTaller.pem" ec2-user@ec2-23-22-49-189.compute-1.amazonaws.com
The authenticity of host 'ec2-23-22-49-189.compute-1.amazonaws.com (23.22.49.189)' can't be established.
ED25519 key fingerprint is SHA256:rmB14tJZe4xoTtlNfd+GohVKZIqteRxMQFUu6R3IX78.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-23-22-49-189.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
54 package(s) needed for security, out of 97 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-25-38 ~]$
```

Step 5: List Your Instances

aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query "Reservations[].Instances[].InstanceId"

```
C:\Users\Crisp>aws ec2 describe-instances --filters "Name=instance-type,Values=t2.micro" --query "Reservations[].Instances[].Ins
tanceId"
[
    "i-0edaa4badf79523ca",
    "i-0a8914ab6985c8962"
]
```



Step 6: Clean Up

Delete the key pair:

aws ec2 delete-key-pair --key-name MyKeyPair

```
C:\Users\Crisp>aws ec2 delete-key-pair --key-name MiLlaveTaller
{
    "Return": true,
    "KeyPairId": "key-08033b001beb5852d"
}
```

Delete the security group:

aws ec2 delete-security-group --group-id sg-903004f8

```
C:\Users\Crisp>aws ec2 delete-security-group --group-id sg-06515a1bacbfac7dd
{
    "Return": true,
    "GroupId": "sg-06515a1bacbfac7dd"
}
```

Terminate the instance:

aws ec2 terminate-instances --instance-ids i-07d0ddb36ea3e65a4

```
C:\Users\Crisp>aws ec2 terminate-instances --instance-ids i-0a8914ab6985c8962
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0a8914ab6985c8962",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

Conclusion

Congratulations! You just learned how to automatically deploy an EC2 instance on AWS.

References

- AWS CLI User Guide