# Task 13

# How to Configure, Maintain and Access a Cloud Virtual Machine on AWS

REPORT SUBMITTED BY

MALAVIKA.MS

# CONTENTS

## 1. Overview

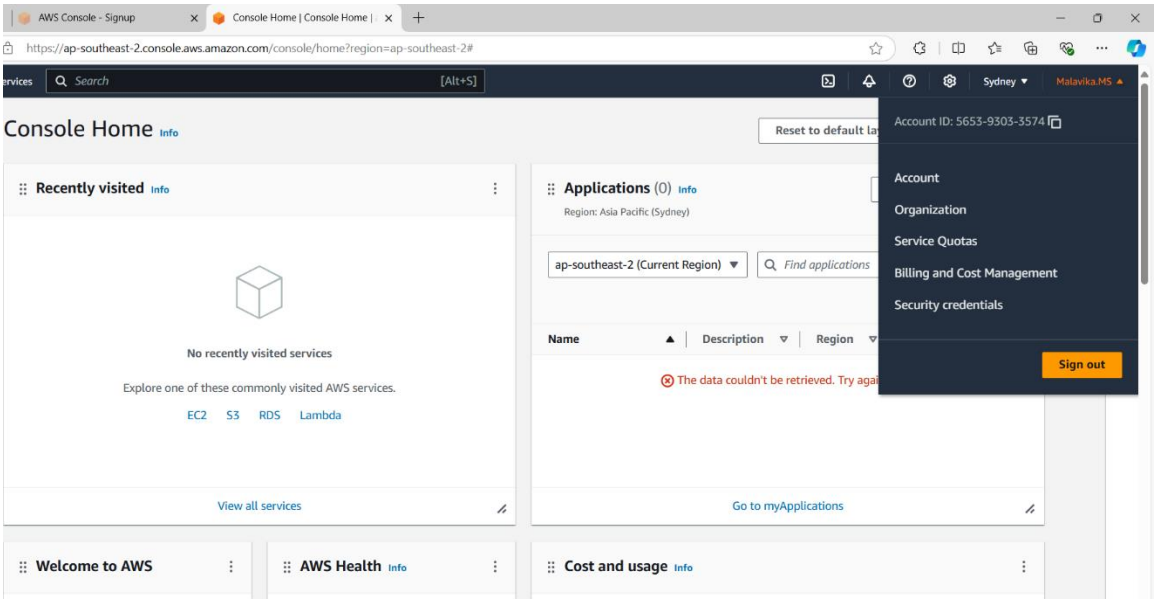
**Task Description**: The objective of this task is to research and deploy a free Virtual Machine (VM) in a cloud environment. The tutorial covers the steps involved in setting up the VM, obtaining its IP address, testing connectivity using ping, and accessing the VM via SSH. Additional tasks include performing network tests and exploring the VM's features, followed by shutting down the VM properly after the work is completed. For this demonstration, I chose **AWS** as the cloud service provider.

## 2. EC2 Instance Launch Process

### 2.1 Accessing the EC2 Console

The process begins by accessing the AWS EC2 dashboard through the following steps:

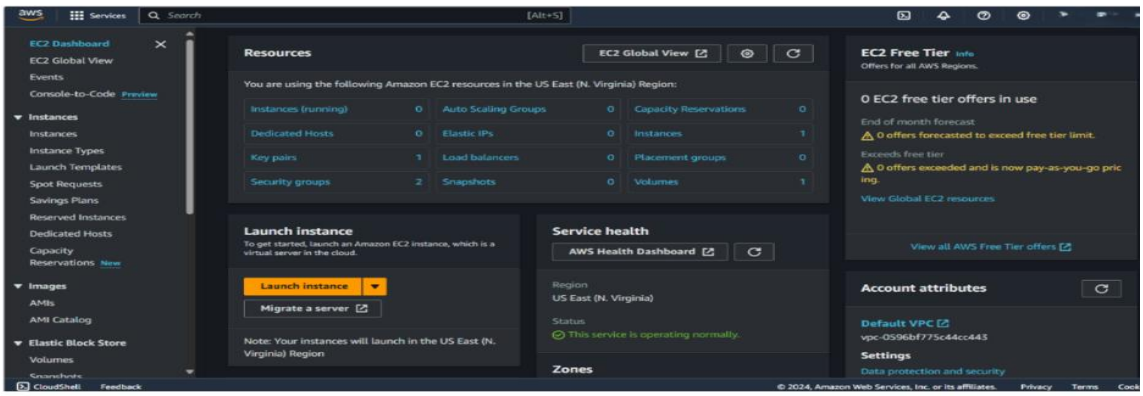
1. Logged into the **AWS Management Console** and navigated to the **EC2 Dashboard** by searching for EC2.
2. Selecting EC2 from the search results took me directly to the EC2 dashboard.




### 2.2 Launching an EC2 Instance

To start the process of creating an EC2 instance:

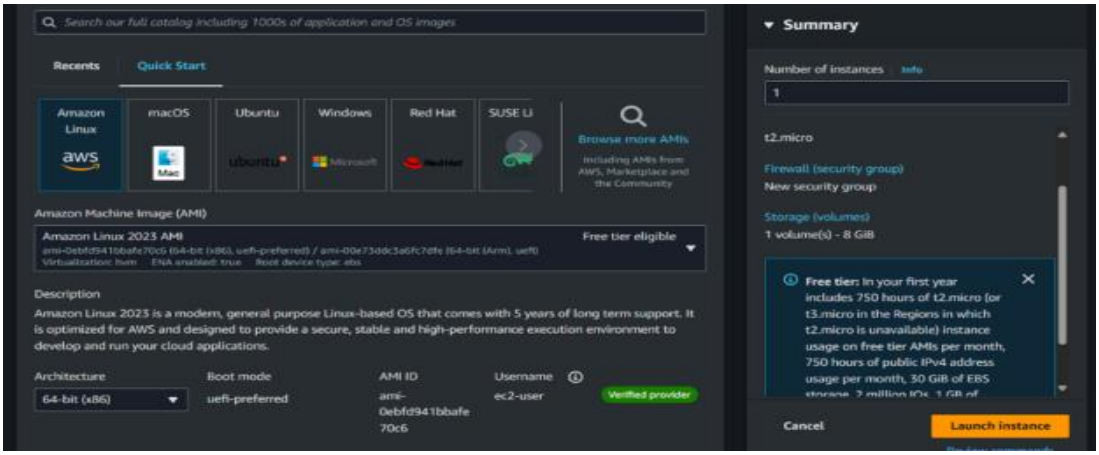
1. I selected the **Launch Instance** option from the EC2 dashboard.
2. This directed me to the **Launch an Instance** page, where I was able to customize the VM's configuration settings.




### 2.3 Choosing the Amazon Machine Image (AMI)

The operating system is determined by the selected Amazon Machine Image (AMI):

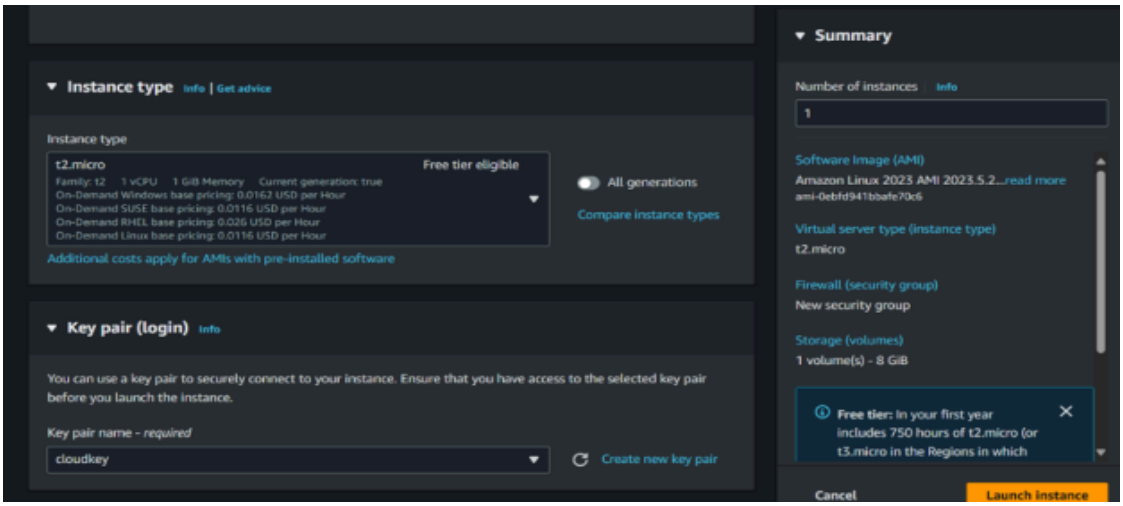
1. I reviewed the available AMIs and opted for Amazon Linux, which qualifies for the free tier.
2. While alternatives like macOS, Ubuntu, and Windows Server were available, I decided to go with Amazon Linux for this setup.




### 2.4 Selecting an Instance Type

The instance type determines the resource allocation (CPU, memory, etc.):

1. I chose the **t2.micro** instance type, which is ideal for small workloads and falls within the AWS free-tier limits.
2. For secure access to the EC2 instance, I selected a key pair named **cloudkey** (a key pair consists of a public key and a private key, used for secure authentication).




### 2.5 Configuring Instance Details

I proceeded to configure additional settings for the instance:

1. I chose the default **VPC** for networking, leaving most of the settings unchanged.
2. This was sufficient, as I did not need any advanced networking configurations.

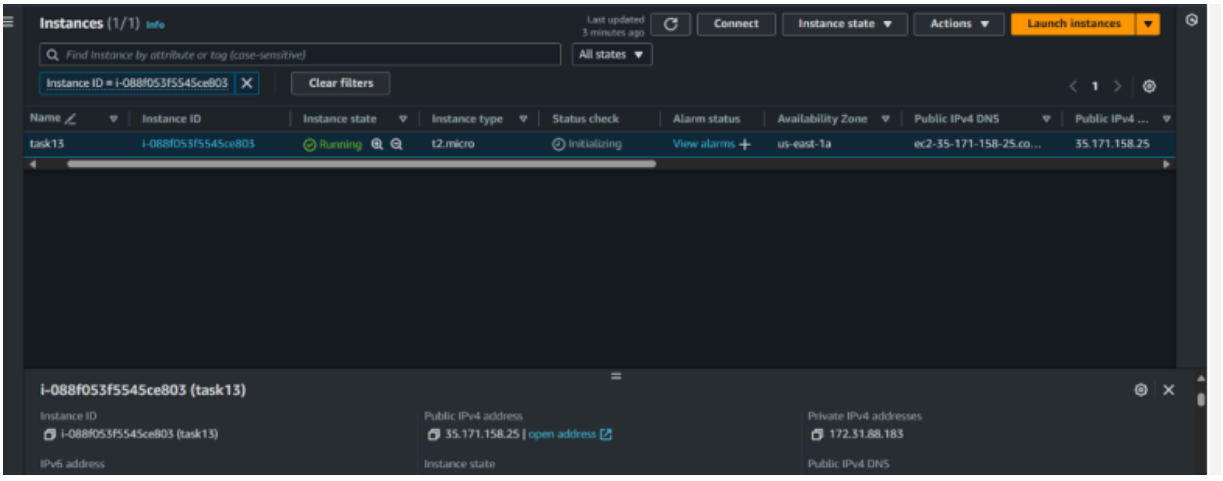
### 2.6 Configuring Security Group Rules

The Security Group acts as a virtual firewall that manages incoming and outgoing traffic for the instance:

1. Before starting, I ensured that SSH (port 22) was enabled for remote access to the Linux instance.
2. I then modified the inbound rules to permit specific traffic. For example, I added a rule to allow ICMP, which is necessary for testing the server's connectivity via ping (see screenshot 8).

### 2.7 Final Review and Launch

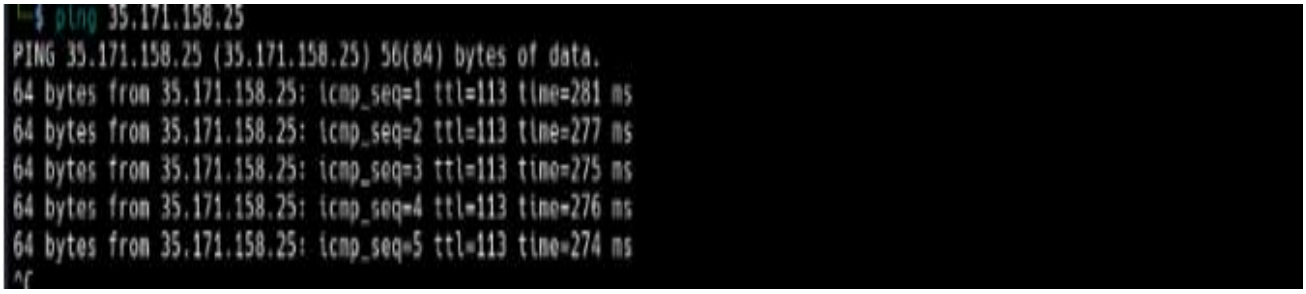
1. I verified that all configuration settings were accurate.
2. After confirming that everything was correctly set up, I clicked the **Launch Instance** button.



### 3. Connectivity Testing and Instance Access

#### 3.1 Checking Public IP Connectivity Using the Ping Command

After the instance was up and running, I verified the connectivity by checking the public IP with the ping command.




#### 3.2 SSH Access to the EC2 Instance

To connect to the virtual machine from a remote system, I used the SSH command as follows:

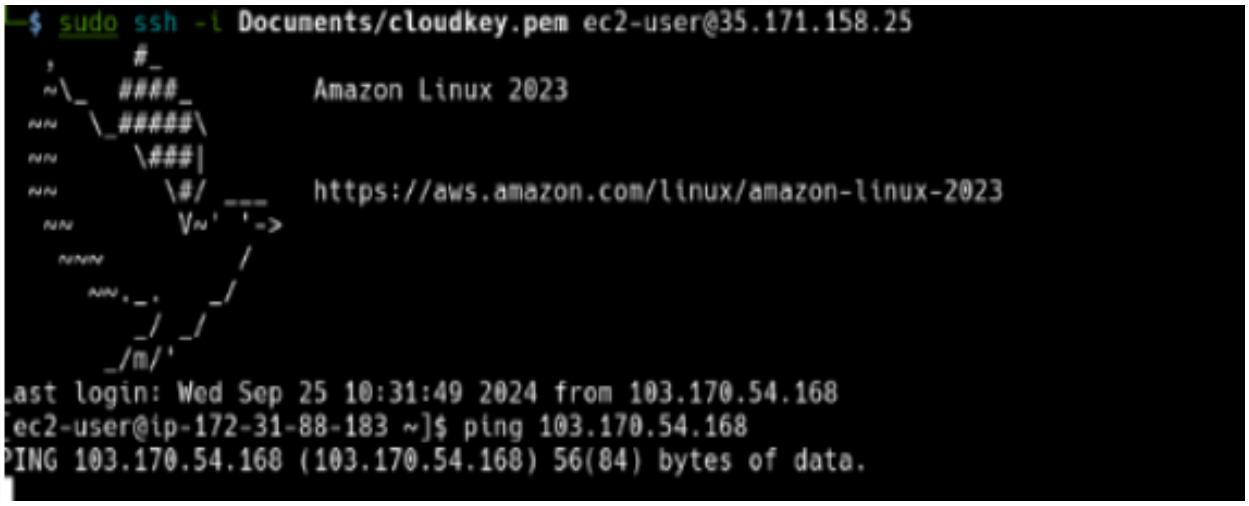
1. I utilized the generated security key **cloudkey** from the instance setup to securely log in to the instance from my local machine.
2. I executed the following command on my Kali Linux system:
   sudo ssh -i Documents/cloudkey.pem ec2-user@52.91.114.191

### 4. Testing Connectivity to the Public IP from the VM in the Cloud

#### 4.1 Ping Test from the EC2 Instance

I attempted to check the public IP connectivity by pinging my public IP from within the EC2 instance:

1. I was unable to ping the public IP due to restrictions imposed by my ISP, which blocks ICMP (ping) packets.
2. This is a common practice among Internet Service Providers to restrict ICMP traffic for security or network performance reasons.



**5. Terminating the VM**

I navigated back to the AWS EC2 dashboard to terminate the VM, ensuring that it wouldn't continue running and incur charges or consume resources.

**Results Summary**

1. I successfully launched a free-tier VM on AWS and obtained its public IP address.
2. After modifying the security settings, I was able to ping the VM from my Kali Linux machine.
3. I accessed the VM via SSH and executed the command ip a, capturing a screenshot of the output.
4. I attempted to ping the public IP from within the cloud VM, but the request was blocked due to restrictions imposed by my ISP.
5. After completing all these tasks, I properly shut down and terminated the VM in accordance with the requirements.