Launching, Configuring, and Accessing a Free Cloud VM (AWS Example)

VAISHNAV P

CONTENTS

1. Overview

The purpose of this task is to research and start a free Virtual Machine (VM) in a cloud environment (preferably Azure but AWS or GCP can also be used). The process includes configuring the VM, obtaining its IP address, verifying connectivity through ping, and accessing the VM via SSH. Additionally, tasks include performing network tests, exploring the VM, and properly shutting it down after use.

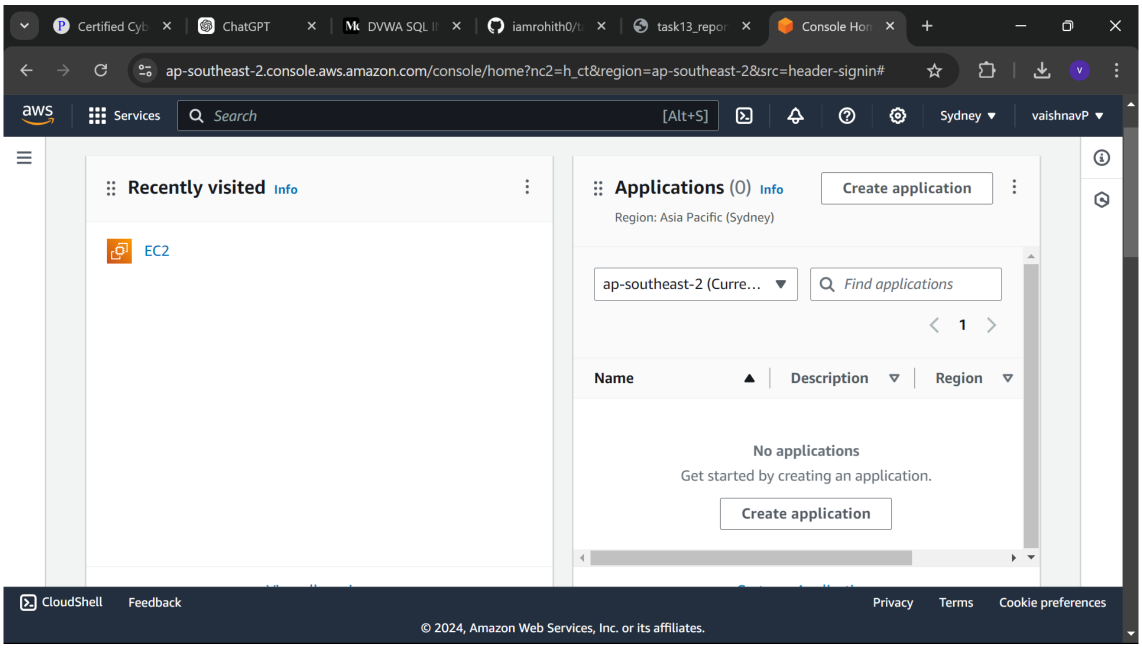
For this report, I have used AWS as the cloud provider.

2. EC2 Instance Launch Process

2.1 Accessing the EC2 Console

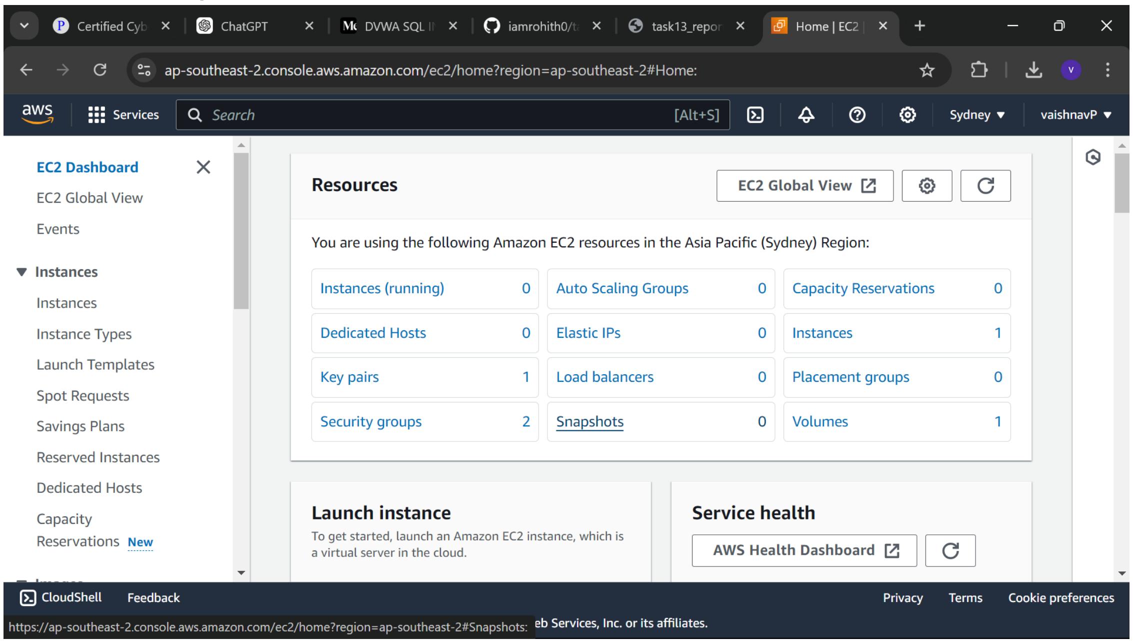
To begin the setup process, I accessed the **EC2 Dashboard** by performing the following actions:

1. I logged into the AWS Management Console and used the search bar to locate **EC2**.



Screenshot 1

2. Selecting **EC2** from the results redirected me to the EC2 dashboard.



ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#Home:

Resources

You are using the following Amazon EC2 resources in the Asia Pacific (Sydney) Region:

Instances (running)	0	Auto Scaling Groups	0	Capacity Reservations	0
Dedicated Hosts	0	Elastic IPs	0	Instances	1
Key pairs	1	Load balancers	0	Placement groups	0
Security groups	2	Snapshots	0	Volumes	1

Launch instance

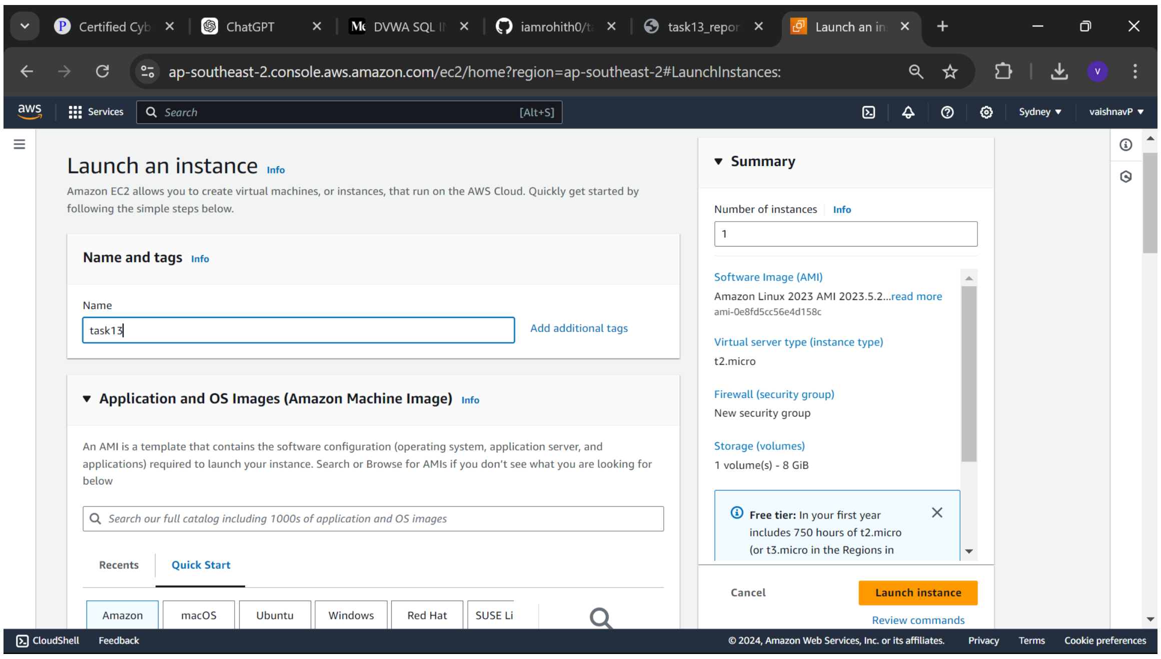
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Service health

2.2 Starting the EC2 Instance Launch

To initiate the creation of an EC2 instance:

1. I clicked the **Launch Instance** button on the EC2 dashboard.
2. This brought me to the **Launch an Instance** page, where I could configure the settings for the virtual machine (VM).



ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#LaunchInstances:

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: task13

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Summary

Number of instances: 1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.5.2...read more
ami-0e8fd5cc56e4d158c

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in

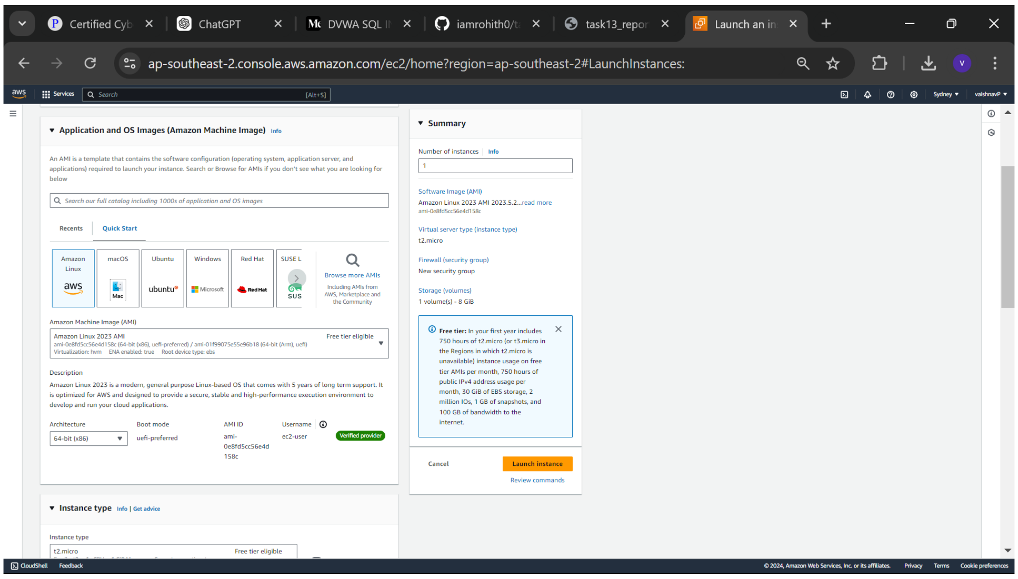
© 2024, Amazon Web Services, Inc. or its affiliates.

Screenshot 3

2.3 Selecting the Amazon Machine Image (AMI)

The Amazon Machine Image (AMI) determines the operating system for the EC2 instance:

1. I browsed through the available AMIs and chose **Amazon Linux**, which is **free-tier eligible**.
2. Other options like **macOS**,**Ubuntu** or **Windows Server** are also available, but I selected Amazon Linux for this setup.



▾ Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Amazon Machine Image (AMI)
Amazon Linux 2023 AMI — Free tier eligible
ami-0e8fd5cc56e4d158c (64-bit (x86), uefi-preferred) / ami-01f99075e55e96b18 (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Architecture	Boot mode	AMI ID	Username
64-bit (x86)	uefi-preferred	ami-0e8fd5cc56e4d158c	ec2-user — Verified provider

▾ Instance type Info | Get advice

Instance type
t2.micro — Free tier eligible

▾ Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.5.2...read more
ami-0e8fd5cc56e4d158c

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

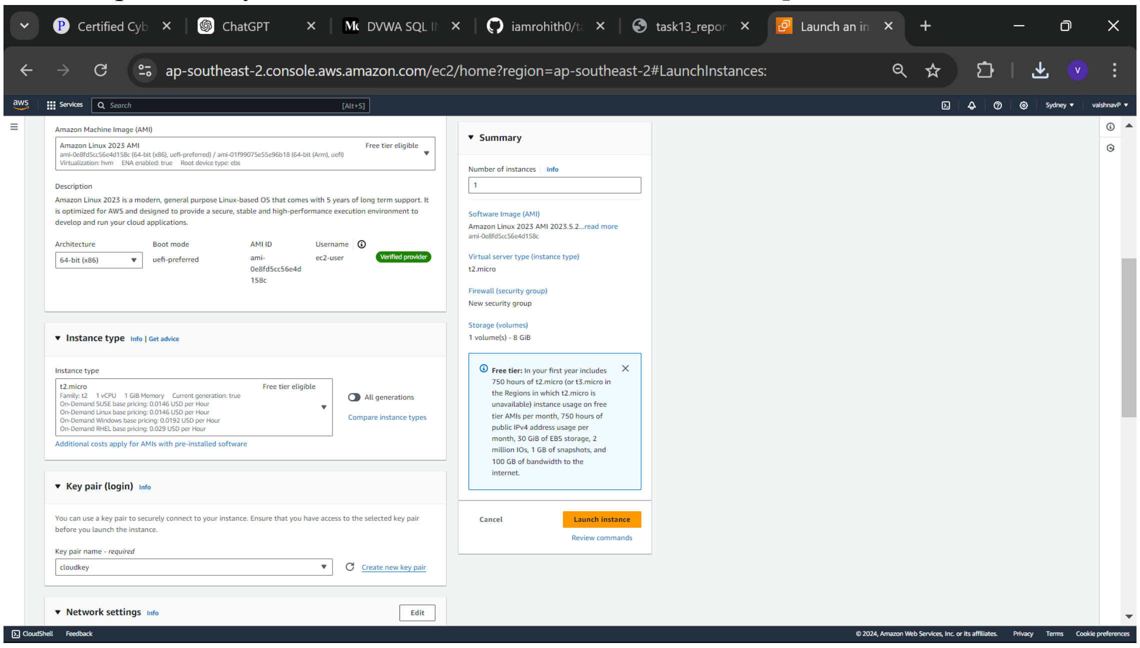
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

SCRENNSHOT 4

2.4 Choosing an Instance Type

Instance types define the resource configuration (CPU, memory, etc.):

1. I selected the **t2.micro** instance type, which is suitable for light workloads and falls under the **AWS free-tier** offering.
2. Selected **key pair** renamed to **cloudkey** [a key pair consists of a public key and a private key used for secure access to EC2 instances.]



Amazon Machine Image (AMI)
Amazon Linux 2023 AMI — Free tier eligible
ami-0e8fd5cc56e4d158c (64-bit (x86), uefi-preferred) / ami-01f99075e55e96b18 (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Architecture	Boot mode	AMI ID	Username
64-bit (x86)	uefi-preferred	ami-0e8fd5cc56e4d158c	ec2-user — Verified provider

▾ Instance type Info | Get advice

Instance type
t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand SUSE base pricing: 0.0146 USD per Hour
On-Demand Linux base pricing: 0.0146 USD per Hour
On-Demand Windows base pricing: 0.0192 USD per Hour
On-Demand RHEL base pricing: 0.029 USD per Hour
Additional costs apply for AMIs with pre-installed software

▾ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*
cloudkey

▾ Network settings Info

▾ Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.5.2...read more
ami-0e8fd5cc56e4d158c

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Screenshot 5

2.5 Configuring Instance Details

At this stage, I configured additional settings for the instance:

1. I used the **default VPC (Virtual Private Cloud)** for networking, leaving most of the default options unchanged.
2. This configuration was adequate for my needs, as I didn't require any advanced networking configurations.

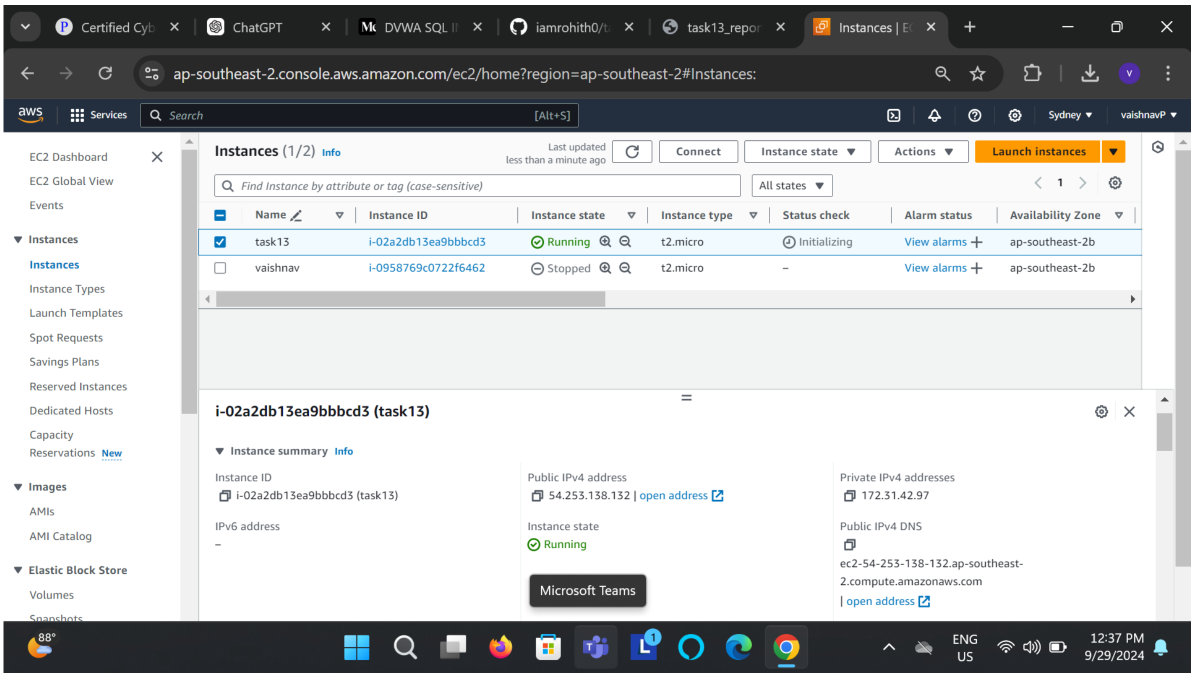
2.6 Setting Up Security Group Rules

The **Security Group** is a virtual firewall that controls the traffic to and from the instance:

1. For this instance, **SSH (port 22)** was pre-configured to allow remote access for Linux.
2. I later made modifications to the **inbound rules** to enable specific traffic, such as allowing ICMP for ping tests. (*refer screenshot 8)*

2.7 Final Review and Launch

1. I reviewed the configuration settings and confirmed they were correct.
2. After verifying the setup, I clicked **Launch Instance**, which successfully created and deployed the EC2 instance.


ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#Instances:

Instances (1/2)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☑	task13	i-02a2db13ea9bbbcd3	Running	t2.micro	Initializing	View alarms	ap-southeast-2b
☐	vaishnav	i-0958769c0722f6462	Stopped	t2.micro	–	View alarms	ap-southeast-2b

i-02a2db13ea9bbbcd3 (task13)

Instance summary

Instance ID: i-02a2db13ea9bbbcd3 (task13)
Public IPv4 address: 54.253.138.132 | open address
Private IPv4 addresses: 172.31.42.97
IPv6 address: –
Instance state: Running
Public IPv4 DNS: ec2-54-253-138-132.ap-southeast-2.compute.amazonaws.com | open address

Screenshot 6

3. Connectivity Testing and Instance Access

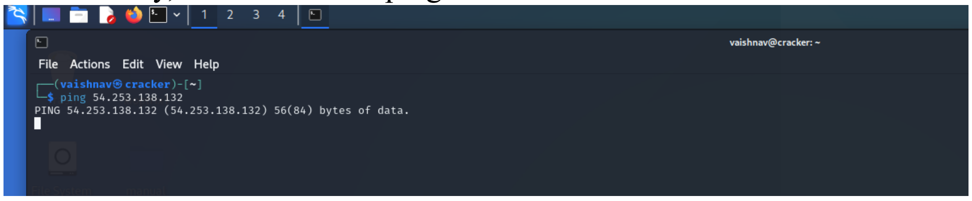
3.1 Verifying Public IP Connectivity via Ping

Once the instance was up and running, I tested the **public IP connectivity**:

1. The public IP assigned to my EC2 instance was:

Public IP: 54.253.138.132

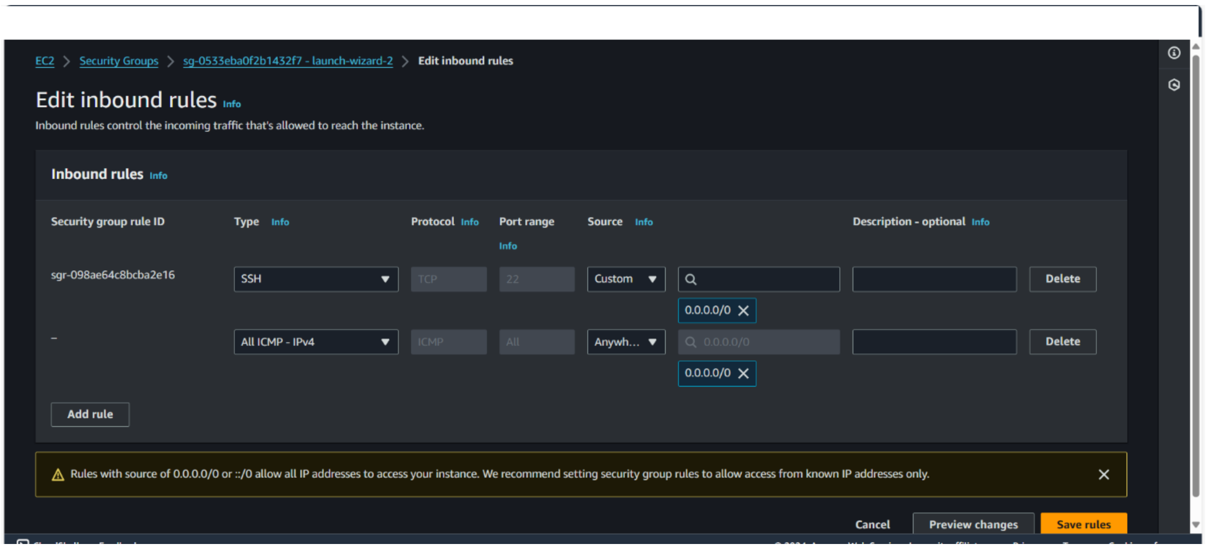
2. Initially, I was unable to ping the instance from outside.



```
vaishnav@cracker: ~
File  Actions  Edit  View  Help
┌──(vaishnav㉿cracker)-[~]
└─$ ping 54.253.138.132
PING 54.253.138.132 (54.253.138.132) 56(84) bytes of data.
```

Screenshot 7

3. However, after modifying the **inbound rules** in the security group to allow **ICMP traffic**.



EC2 > Security Groups > sg-0533eba0f2b1432f7 - launch-wizard-2 > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

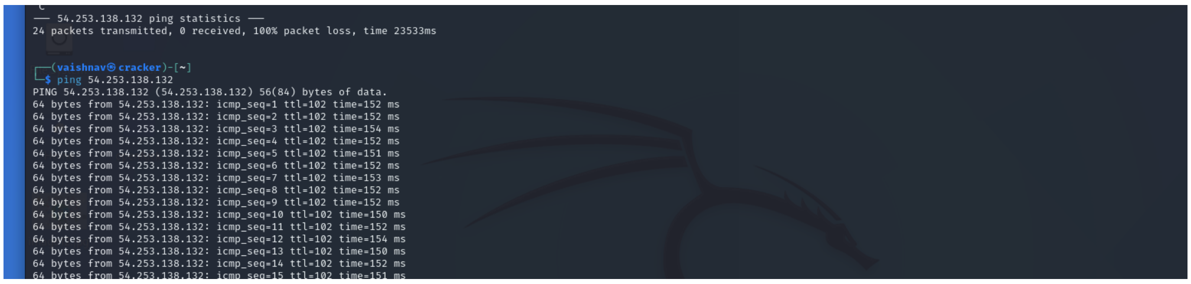
Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-098ae64c8bcba2e16	SSH	TCP	22	Custom	0.0.0.0/0	
–	All ICMP - IPv4	ICMP	All	Anywh...	0.0.0.0/0	

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Screenshot 8

I was able to successfully ping the public IP



```
── 54.253.138.132 ping statistics ──
24 packets transmitted, 0 received, 100% packet loss, time 23533ms

┌──(vaishnav㉿cracker)-[~]
└─$ ping 54.253.138.132
PING 54.253.138.132 (54.253.138.132) 56(84) bytes of data.
64 bytes from 54.253.138.132: icmp_seq=1 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=2 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=3 ttl=102 time=154 ms
64 bytes from 54.253.138.132: icmp_seq=4 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=5 ttl=102 time=151 ms
64 bytes from 54.253.138.132: icmp_seq=6 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=7 ttl=102 time=153 ms
64 bytes from 54.253.138.132: icmp_seq=8 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=9 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=10 ttl=102 time=150 ms
64 bytes from 54.253.138.132: icmp_seq=11 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=12 ttl=102 time=154 ms
64 bytes from 54.253.138.132: icmp_seq=13 ttl=102 time=150 ms
64 bytes from 54.253.138.132: icmp_seq=14 ttl=102 time=152 ms
64 bytes from 54.253.138.132: icmp_seq=15 ttl=102 time=151 ms
```

Screenshot 9

3.2 Accessing the EC2 Instance via SSH

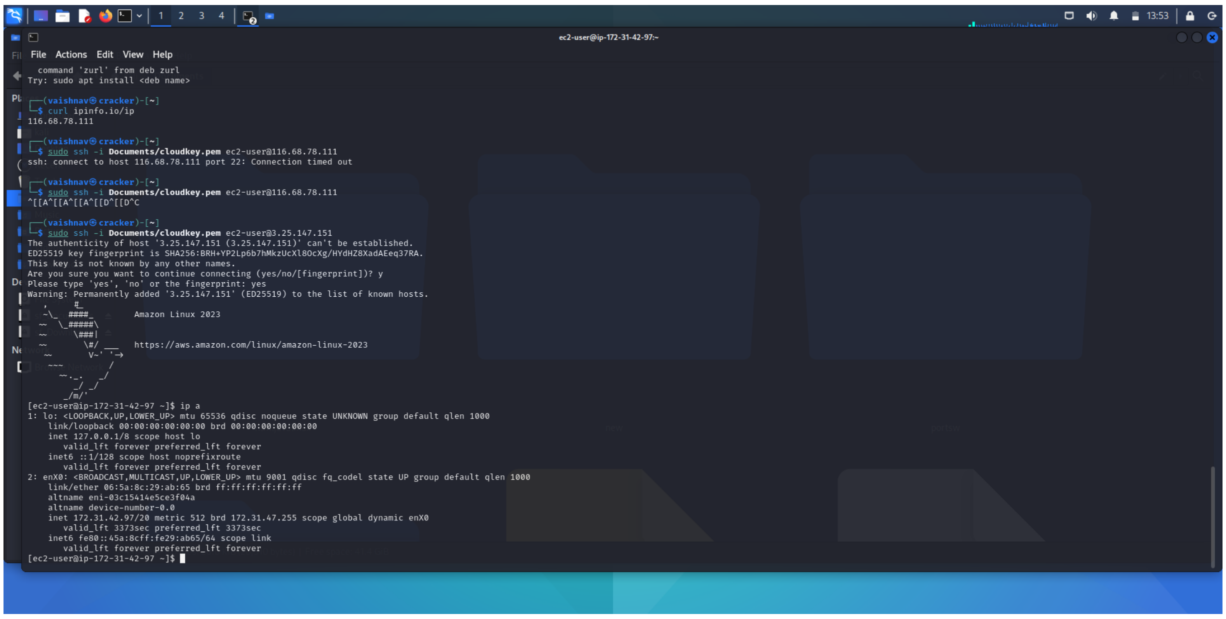
To access the EC2 instance remotely, I used **SSH** with the following steps:

1. I used the security key **[cloudkey]** (generated during the instance setup) to securely log in to the instance from my local machine.

2. On my **Kali Linux** system, I used the following command to establish the SSH connection:

```
sudo ssh -i Documents/cloudkey.pem ec2-user@3.25.147.151
```

3. I successfully logged into the instance and gained shell access to the EC2 virtual machine.



```
  command 'zurl' from deb zurl
Try: sudo apt install <deb name>

┌──(vaishnav㉿cracker)-[~]
└─$ curl ipinfo.io/ip
116.68.78.111

┌──(vaishnav㉿cracker)-[~]
└─$ sudo ssh -i Documents/cloudkey.pem ec2-user@116.68.78.111
ssh: connect to host 116.68.78.111 port 22: Connection timed out

┌──(vaishnav㉿cracker)-[~]
└─$ sudo ssh -i Documents/cloudkey.pem ec2-user@116.68.78.111
^[[A^[[A^[[A^[[D^[[D^C

┌──(vaishnav㉿cracker)-[~]
└─$ sudo ssh -i Documents/cloudkey.pem ec2-user@3.25.147.151
The authenticity of host '3.25.147.151 (3.25.147.151)' can't be established.
ED25519 key fingerprint is SHA256:BRH+YP2Lp6b7hMkzUcXl8OcXg/HYdHZ8XadAEeq37RA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '3.25.147.151' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-42-97 ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enX0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 06:5a:8c:29:ab:65 brd ff:ff:ff:ff:ff:ff
    altname eni-03c15414e5ce3f04a
    altname device-number-0.0
    inet 172.31.42.97/20 metric 512 brd 172.31.47.255 scope global dynamic enX0
       valid_lft 3373sec preferred_lft 3373sec
    inet6 fe80::45a:8cff:fe29:ab65/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@ip-172-31-42-97 ~]$ 
```

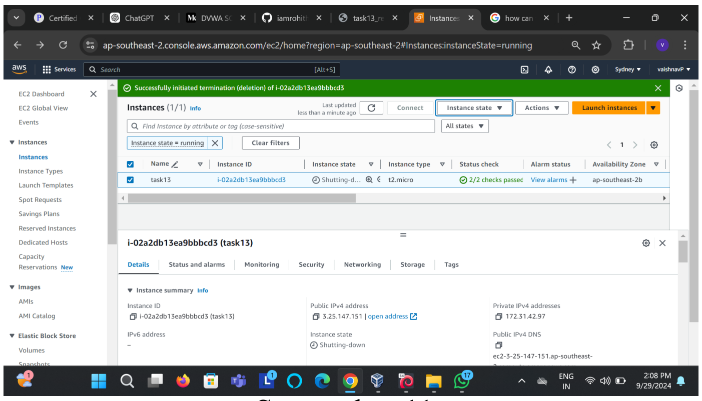
Screenshot 10 [Due to some issues, I had to stop and restart the instance, which resulted in the assignment of a new public IP.]

4. Testing Public IP Connectivity from the Cloud VM

4.1 Ping Test from the EC2 Instance

I attempted to verify the public IP connectivity by pinging my **public IP** from within the EC2 instance itself:

1. However, I was unable to ping the public IP due to restrictions imposed by the **ISP** blocking **ICMP (ping) packets**.
2. This is a common limitation where Internet Service Providers prevent ICMP traffic for security or network performance reasons.



ap-southeast-2.console.aws.amazon.com/ec2/home?region=ap-southeast-2#Instances:instanceState=running

Successfully initiated termination (deletion) of i-02a2db13ea9bbbcd3

Instances (1/1)

Instance state = running

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
task13	i-02a2db13ea9bbbcd3	Shutting-d...	t2.micro	2/2 checks passec	View alarms	ap-southeast-2b

i-02a2db13ea9bbbcd3 (task13)

Instance summary

Instance ID: i-02a2db13ea9bbbcd3 (task13)
Public IPv4 address: 3.25.147.151 | open address
Private IPv4 addresses: 172.31.42.97
IPv6 address: –
Instance state: Shutting-down
Public IPv4 DNS: ec2-3-25-147-151.ap-southeast-

Screenshot 11

5. Terminating the VM

Finally, I returned to the AWS EC2 dashboard and **terminated the VM** to ensure it no longer incurs any charges or resources.

SUMMARY OF RESULTS

1. I successfully started a free-tier VM on AWS and obtained the public IP.
2. After adjusting the security settings, I was able to ping the VM from my Kali Linux machine.
3. I logged into the VM via SSH, ran the ip a command, and captured the screenshot.
4. I attempted to ping the public IP from within the cloud VM, but the request was blocked by ISP restrictions.
5. After completing the tasks, I shut down and terminated the VM as required.