

CBE321 Lab 2 2022BCY0012 Shravasti Ohol

Activity 1 : EC2 Ubuntu Instance Communication

1. Creating two Ubuntu instances

- Name - server
- AMI - Ubuntu
- Key pair - key.pem
- Security group - existing

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 console. The first step, 'Name and tags', is completed with the name 'server'. The next step, 'Application and OS Images (Amazon Machine Image)', is currently selected. It displays a catalog of AMIs, including 'Ubuntu Server 24.04 LTS (HVM, SSD Volume Type)'. A tooltip for this AMI indicates it is 'Free tier eligible'. The summary section shows one instance being launched with the Canonical AMI and t2.micro instance type. The 'Launch instance' button is visible at the bottom.

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 console, now on the 'Instance type' step. It lists various instance types, including t2.micro, which is highlighted as 'Free tier eligible'. Other options like t2.small, t2.medium, and t2.large are also shown. The 'Summary' section remains the same, showing one instance launch with the Canonical AMI and t2.micro instance type. The 'Launch instance' button is visible at the bottom.

Network settings

Network: Info
vpc-007543e992d582d02

Subnet: Info
No preference (Default subnet in any availability zone)

Auto-assign public IP: Info
Enable
Additional charges apply when outside of free tier allowance

Firewall (security groups): Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group (radio button) or Select existing security group (radio button selected)

Common security groups: Info
Select security groups (dropdown menu)
securitygroup sg-0b7f8ec346725a5dd (selected)

Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

Configure storage

Advanced

1x 8 GiB gp3 Root volume 3000 IOPS (Not encrypted)

Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 22.04, amd64...read more
ami-053b12d5152c0cc71

Virtual server type (instance type): t2.micro

Firewall (security group): securitygroup

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 I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch instance Preview code

- e. Name - client
- f. AMI - Ubuntu
- g. Key pair - key.pem
- h. Security group - existing

Name and tags

Name: client

Add additional tags

Application and OS Images (Amazon Machine Image)

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)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
ami-053b12d5152c0cc71 (64-bit (x86)) / ami-05cfef798c2e4e9 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Summary

Number of instances: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.6.2...read more
ami-0964f769859f29ab4

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 I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch instance Preview code

| Name | Instance ID | Instance state | Instance type | Status check | Alarm status | Availability |
|--------|---------------------|----------------|---------------|-------------------|-------------------------------|--------------|
| server | i-04109f99b8c1d3136 | Terminated | t2.micro | - | View alarms + | ap-south-1b |
| server | i-0efb17a021b9e3f42 | Running | t2.micro | 2/2 checks passed | View alarms + | ap-south-1b |
| client | i-075123d8f8374862f | Running | t2.micro | 2/2 checks passed | View alarms + | ap-south-1b |

2. Checking connection between both instances using ping

a. Pinging client

```
shravastiohol@Shravastis-MacBook-Pro ~ % ping 3.110.194.59
PING 3.110.194.59 (3.110.194.59): 56 data bytes
64 bytes from 3.110.194.59: icmp_seq=0 ttl=53 time=100.723 ms
64 bytes from 3.110.194.59: icmp_seq=1 ttl=53 time=145.071 ms
64 bytes from 3.110.194.59: icmp_seq=2 ttl=53 time=64.864 ms
64 bytes from 3.110.194.59: icmp_seq=3 ttl=53 time=153.401 ms
^C
--- 3.110.194.59 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 64.864/116.015/153.401/35.680 ms
shravastiohol@Shravastis-MacBook-Pro ~ %
```

b. Pinging server

```
shravastiohol@Shravastis-MacBook-Pro ~ % ping 65.2.70.216
PING 65.2.70.216 (65.2.70.216): 56 data bytes
64 bytes from 65.2.70.216: icmp_seq=0 ttl=53 time=146.340 ms
64 bytes from 65.2.70.216: icmp_seq=1 ttl=53 time=177.984 ms
64 bytes from 65.2.70.216: icmp_seq=2 ttl=53 time=302.265 ms
64 bytes from 65.2.70.216: icmp_seq=3 ttl=53 time=487.233 ms
^C
--- 65.2.70.216 ping statistics ---
5 packets transmitted, 4 packets received, 20.0% packet loss
round-trip min/avg/max/stddev = 146.340/278.455/487.233/133.888 ms
shravastiohol@Shravastis-MacBook-Pro ~ %
```

3. Connecting to the server instance

```
shravastiohol@Shravastis-MacBook-Pro ~ % ssh -i ~/Downloads/key.pem ubuntu@ec2-65-2-70-216.ap-south-1.compute.amazonaws.com
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Wed Jan 15 09:52:03 UTC 2025

 System load: 0.0      Processes:          103
 Usage of /: 24.6% of 6.71GB  Users logged in: 0
 Memory usage: 20%           IPv4 address for enx0: 172.31.10.185
 Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-10-185:~$
```

4. Connecting to the client instance

```
... -- ssh -i ~/Downloads/key.pem ubuntu@ec2-65-2-70-216.ap-south-1.compute.amazonaws.com ... -- ssh -i ~/Downloads/key.pem ubuntu@ec2-3-110-194-59.ap-south-1.compute.amazonaws.com ...
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Wed Jan 15 09:53:46 UTC 2025

 System load: 0.0      Processes:          104
 Usage of /: 24.7% of 6.71GB  Users logged in: 0
 Memory usage: 20%           IPv4 address for enx0: 172.31.4.139
 Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-4-139:~$
```

5. Installing python package on server

```
ubuntu@ip-172-31-10-185:~$ sudo apt install python3 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.12.3-0ubuntu2).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

6. Installing python package on client

```
ubuntu@ip-172-31-4-139:~$ sudo apt install python3 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.12.3-0ubuntu2).
python3 set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-4-139:~$ █
```

7. Server side code for connection

```
ubuntu@ip-172-31-10-185:~$ nano server.py
ubuntu@ip-172-31-10-185:~$ cat server.py
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('0.0.0.0', 12345)) # Bind to all network interfaces on port 12345
server_socket.listen(1)

print("Server is listening...")
connection, address = server_socket.accept()
print(f"Connection established with {address}")

while True:
    message = connection.recv(1024).decode()
    if not message:
        break
    print(f"Client: {message}")
    connection.send(input("Server: ").encode())

connection.close()
server_socket.close()
ubuntu@ip-172-31-10-185:~$ █
```

8. Client side code for connection

```
ubuntu@ip-172-31-4-139:~$ nano client.py
ubuntu@ip-172-31-4-139:~$ cat client.py
import socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_ip = '<server_public_ip>'
server_port = 12345

client_socket.connect((server_ip, server_port))
print("Connected to the server.")

while True:
    message = input("Client: ")
    client_socket.send(message.encode())
    response = client_socket.recv(1024).decode()
    if not response:
        break
    print(f"Server: {response}")

client_socket.close()
ubuntu@ip-172-31-4-139:~$ █
```

9. Configuring inbound rules for security group to allow TCP connections from the specified port

The screenshot shows the AWS CloudFront 'Edit inbound rules' interface. It lists three existing rules:

| Security group rule ID | Type | Protocol | Port range | Source | Description - optional |
|------------------------|-----------------|----------|------------|---------|------------------------|
| sgr-Occbe50c619505b8d | Custom TCP | TCP | 12345 | Cu... ▾ | 0.0.0.0/0 X |
| sgr-0afa6da818939d36f | SSH | TCP | 22 | Cu... ▾ | 0.0.0.0/0 X |
| sgr-091761f13ce60e3cb | All ICMP - IPv4 | ICMP | All | Cu... ▾ | 0.0.0.0/0 X |

At the bottom, there is an 'Add rule' button and three action buttons: 'Cancel', 'Preview changes', and 'Save rules'.

10. Running script on client side

```
[ubuntu@ip-172-31-4-139:~$ python3 client.py
Connected to the server.
Client: hey, this is shravasti
Server: hey, this is the server
Client: ]
```

11. Running script on server side

```
[ubuntu@ip-172-31-10-185:~$ nano server.py
[ubuntu@ip-172-31-10-185:~$ python3 server.py
Server is listening...
Connection established with ('3.110.194.59', 51284)
Client: hey, this is shravasti
Server: hey, this is the server
[
```

CBE321 Lab 2 2022BCY0012 Shravasti Ohol

Activity 2 : Static website hosting EC2 Amazon Linux Instance

1. Creating an Amazon Linux Instance

- Name - webserver
- AMI - Amazon Linux
- Key - key.pem
- Security Group - existing

The screenshot shows the 'Launch an instance' wizard in the AWS Management Console. The top navigation bar includes the AWS logo, search bar, and user information (Asia Pacific (Mumbai) and Shravasti). The main page title is 'Launch an instance' with a 'Info' link. Below it, a brief description of what EC2 allows you to do. The first step, 'Name and tags', has a 'Name' field containing 'webserver' and a 'Add additional tags' button. The second step, 'Application and OS Images (Amazon Machine Image)', shows a 'Quick Start' tab selected, displaying recent AMIs like Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Del. A search bar at the top of this section allows searching for '1000s of application and OS images'. Below the tabs, a specific 'Amazon Linux 2023 AMI' is highlighted with its AMI ID, creation date, and compatibility details. To the right of the wizard, a summary panel shows 1 instance being launched, the selected AMI (Amazon Linux 2023 AMI), the instance type (t2.micro), and a note about the free tier. At the bottom right of the summary panel is a large orange 'Launch instance' button.

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
key [Create new key pair](#)

Network settings [Info](#)
Network [Info](#)
vpc-007543eb92d582d02
Subnet [Info](#)
No preference (Default subnet in any availability zone)
Auto-assign public IP [Info](#)
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
 Create security group Select existing security group
Common security groups [Info](#)
Select security groups
securitygroup sg-0b7f8ec346725a3dd [X](#)
VPC: vpc-007543eb92d582d02
Security groups that you add or remove here will be added to or removed from all your network interfaces.

Summary
Number of instances [Info](#)
1
Software Image (AMI)
Amazon Linux 2023 AMI 2023.6.2...[read more](#)
ami-096f476985fb29eb4
Virtual server type (instance type)
t2.micro
Firewall (security group)
securitygroup
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 I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

[Cancel](#) [Launch instance](#) [Preview code](#)

2. Changing key permissions and connecting to the instance

```
shravastiohol@Shravasti-MacBook-Pro ~ % chmod 400 ~/Downloads/key.pem
shravastiohol@Shravasti-MacBook-Pro ~ % ssh -i ~/Downloads/key.pem ec2-user@ec2-3-109-154-100.ap-south-1.compute.amazonaws.com
The authenticity of host 'ec2-3-109-154-100.ap-south-1.compute.amazonaws.com (3.109.154.100)' can't be established.
ED25519 key fingerprint is SHA256:uE04zmzUH3cC+Km143PPc90RoZyeUHbZwmQF12+wBUC.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-109-154-100.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
' _#_
~\_\_ #####_ Amazon Linux 2023
~~ \_\#\#\#\#
~~ \#\#\#
~~ \#/ ___ https://aws.amazon.com/linux/amazon-linux-2023
~~ \~' '-->
~~ / /
~~ . / /
~/ / /
~/m/
[ec2-user@ip-172-31-0-128 ~]$
```

3. Installing and updating packages

```
[ec2-user@ip-172-31-0-128 ~]$ sudo yum update -y
Last metadata expiration check: 0:06:50 ago on Wed Jan 15 10:22:43 2025.
Dependencies resolved.
Nothing to do.
Complete!
```

```
[ec2-user@ip-172-31-0-128 ~]$ sudo yum install httpd -y
Last metadata expiration check: 0:07:05 ago on Wed Jan 15 10:22:43 2025.
Dependencies resolved.
=====
Package           Architecture   Version      Repository    Size
=====
Installing:
httpd            x86_64        2.4.62-1.amzn2023  amazonlinux  48 k
Installing dependencies:
apr              x86_64        1.7.5-1.amzn2023.0.2  amazonlinux  130 k
apr-util         x86_64        1.6.3-1.amzn2023.0.1  amazonlinux  98 k
generic-logos-htpd noarch       18.0.0-12.amzn2023.0.3  amazonlinux  19 k
httpd-core       x86_64        2.4.62-1.amzn2023  amazonlinux  1.4 M
httpd-filesystem noarch       2.4.62-1.amzn2023  amazonlinux  14 k
httpd-tools      x86_64        2.4.62-1.amzn2023  amazonlinux  81 k
libbrotli        x86_64        1.0.9-4.amzn2023.0.2  amazonlinux  315 k
mailcap          noarch       2.1.49-3.amzn2023.0.3  amazonlinux  33 k
Installing weak dependencies:
apr-util-openssl x86_64        1.6.3-1.amzn2023.0.1  amazonlinux  17 k
mod_http2        x86_64        2.0.27-1.amzn2023.0.3  amazonlinux  166 k
mod_lua          x86_64        2.4.62-1.amzn2023  amazonlinux  61 k
```

```
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm          480 kB/s | 17 kB  00:00
(2/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm                  2.4 MB/s | 98 kB  00:00
(3/12): apr-1.7.5-1.amzn2023.0.2.x86_64.rpm                      2.3 MB/s | 130 kB 00:00
(4/12): generic-logos-htpd-18.0.0-12.amzn2023.0.3.noarch.rpm     825 kB/s | 19 kB  00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64                         2.0 MB/s | 48 kB  00:00
(6/12): httpd-filesystem-2.4.62-1.amzn2023.noarch.rpm            760 kB/s | 14 kB  00:00
(7/12): httpd-tools-2.4.62-1.amzn2023.x86_64.rpm                 3.2 MB/s | 81 kB  00:00
(8/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm                30 MB/s | 1.4 MB 00:00
(9/12): libbrotli-1.0.9-4.amzn2023.0.2.x86_64                   9.8 MB/s | 315 kB 00:00
```

```
Installed:
  apr-1.7.5-1.amzn2023.0.2.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
  httpd-2.4.62-1.amzn2023.x86_64
  httpd-filesystem-2.4.62-1.amzn2023.noarch
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  mod_http2-2.0.27-1.amzn2023.0.3.x86_64

  apr-util-1.6.3-1.amzn2023.0.1.x86_64
  generic-logos-htpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-core-2.4.62-1.amzn2023.x86_64
  httpd-tools-2.4.62-1.amzn2023.x86_64
  mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_lua-2.4.62-1.amzn2023.x86_64

Complete!
[ec2-user@ip-172-31-0-128 ~]$
```

4. Start and enable httpd

```
[ec2-user@ip-172-31-0-128 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-0-128 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-172-31-0-128 ~]$
```

5. Create index.html

```
[ec2-user@ip-172-31-0-128 ~]$ cd /var/www/html
[ec2-user@ip-172-31-0-128 html]$ nano index.html
[ec2-user@ip-172-31-0-128 html]$ sudo nano index.html
[ec2-user@ip-172-31-0-128 html]$ cat index.html
<html>
<head>
    <title>My Static Website</title>
</head>
<body>
    <h1>Welcome to My Website!</h1>
    <p>Hosted on an EC2 instance.</p>
</body>
</html>
[ec2-user@ip-172-31-0-128 html]$
```

6. Hosting the website and viewing using public IP address

Welcome to My Website!

Hosted on an EC2 instance.

Website!

⊕ http://3.109.154.100/

⊕ My Static Website