

EXPERIMENT-07

NAME: SHAIKH MUBASHIRA TUFEL AHMED
ROLL NO: 612055 **COURSE:** ADVANCE DEVOPS(ITL504)
BRANCH: T.E. INFORMATION TECHNOLOGY (SEM 5)

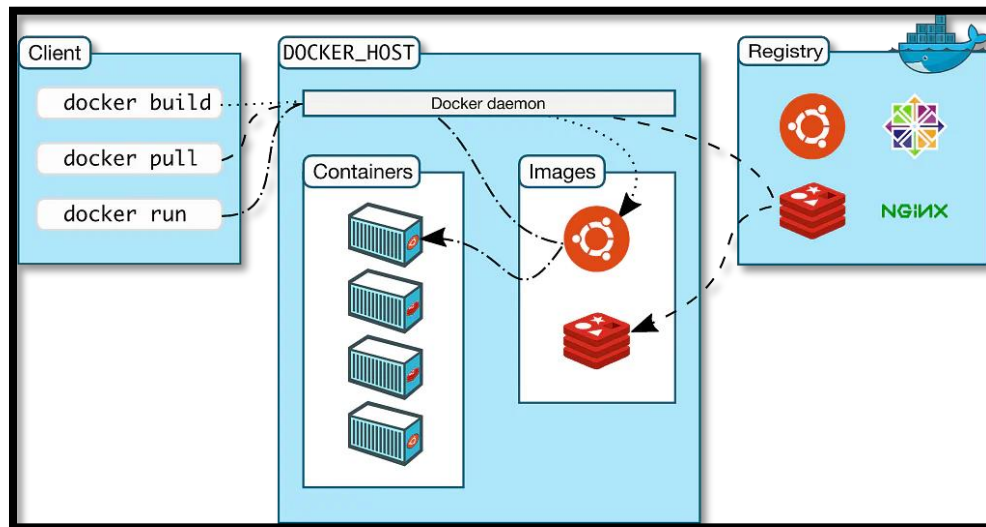
1. What is Containerization / Docker? Explain Docker Architecture with the help of diagram.

→ Docker is an open-source platform that enables developers to build, deploy, run, update and manage *containers*—standardized, executable components that combine application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.

Containers simplify development and delivery of distributed applications. They have become increasingly popular as organizations shift to cloud-native development and hybrid multicloud environments. It's possible for developers to create containers without Docker, by working directly with capabilities built into Linux and other operating systems. But Docker makes containerization faster, easier and safer.

Docker Architecture:

Docker follows Client-Server architecture, which includes the three main components that are **Docker Client**, **Docker Host**, and **Docker Registry**.



1. Docker Client

→ Docker client uses **commands** and **REST APIs** to communicate with the Docker Daemon (Server). When a client runs any docker command on the docker client terminal, the client terminal sends these docker commands to the Docker daemon. Docker daemon receives these commands from the docker client in the form of command and REST API's request.

→ Docker Client uses Command Line Interface (CLI) to run the following commands -

- `docker build`
- `docker pull`
- `docker run`

2. Docker Host

→ Docker Host is used to provide an environment to execute and run applications. It contains the docker daemon, images, containers, networks, and storage.

3. Docker Registry

Docker Registry manages and stores the Docker images.

There are two types of registries in the Docker -

- **Public Registry** - Public Registry is also called as **Docker hub**.
- **Private Registry** - It is used to share images within the enterprise.

Docker Objects

There are the following Docker Objects -

➔ Docker Images

Docker images are the **read-only binary templates** used to create Docker Containers. It uses a private container registry to share container images within the enterprise and also uses public container registry to share container images within the whole world. Metadata is also used by Docker images to describe the container's abilities.

➔ Docker Containers

Containers are the structural units of Docker, which is used to hold the entire package that is needed to run the application. The advantage of containers is that it requires very less resources.

In other words, we can say that the image is a template, and the container is a copy of that template.

➔ Docker Networking

Using Docker Networking, an isolated package can be communicated. Docker contains the following network drivers -

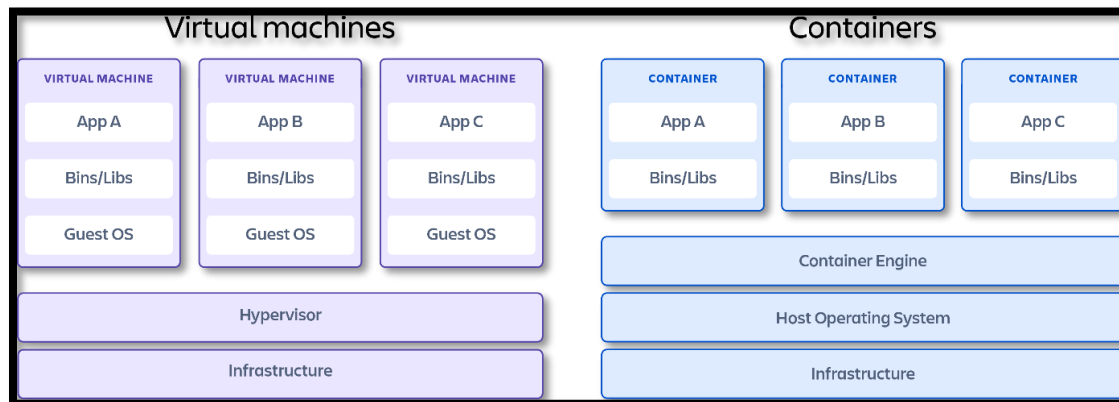
- ✚ **Bridge** - Bridge is a default network driver for the container. It is used when multiple Docker communicates with the same Docker host.
- ✚ **Host** - It is used when we don't need for network isolation between the container and the host.
- ✚ **None** - It disables all the networking.
- ✚ **Overlay** - Overlay offers Swarm services to communicate with each other. It enables containers to run on the different Docker host.
- ✚ **Macvlan** - Macvlan is used when we want to assign MAC addresses to the containers.

➔ Docker Storage

Docker Storage is used to store data on the container. Docker offers the following options for the Storage -

- ✚ **Data Volume** - Data Volume provides the ability to create persistence storage. It also allows us to name volumes, list volumes, and containers associates with the volumes.
- ✚ **Directory Mounts** - It is one of the best options for Docker storage. It mounts a host's directory into a container.
- ✚ **Storage Plugins** - It provides an ability to connect to external storage platforms.

2. Compare Containers vs VMs.



- A virtual machine (VM) is a virtual environment that functions as a virtual computer system with its own CPU, memory, network interface, and storage, created on a physical hardware system (located off- or on-premises).
- Containerization and virtualization are similar in that they both allow for full isolation of applications so that they can be operational in multiple environments. Where the main differences lie are in size and portability.
- The key differentiator between containers and virtual machines is that virtual machines virtualize an entire machine down to the hardware layers and containers only virtualize software layers above the operating system level.
- VMs are the larger of the two, typically measured by the gigabyte and containing their own OS, which allows them to perform multiple resource-intensive functions at once. The increased resources available to VMs allows them to abstract, split, duplicate, and emulate entire servers, operating systems, desktops, databases, and networks.

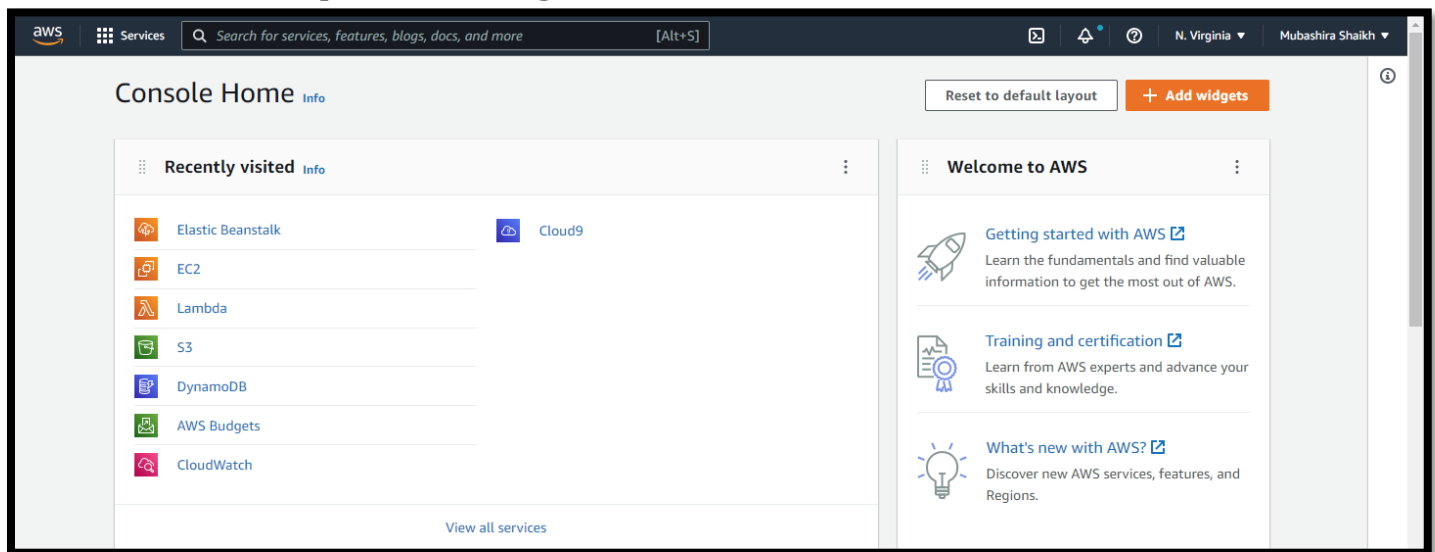
- Containers are much smaller, typically measured by the megabyte and not packaging anything bigger than an app and its running environment.
- Where VMs work well with traditional, monolithic IT architecture, containers were made to be compatible with newer and emerging technology like clouds, CI/CD, and DevOps.

3. Why are Containers lightweight?

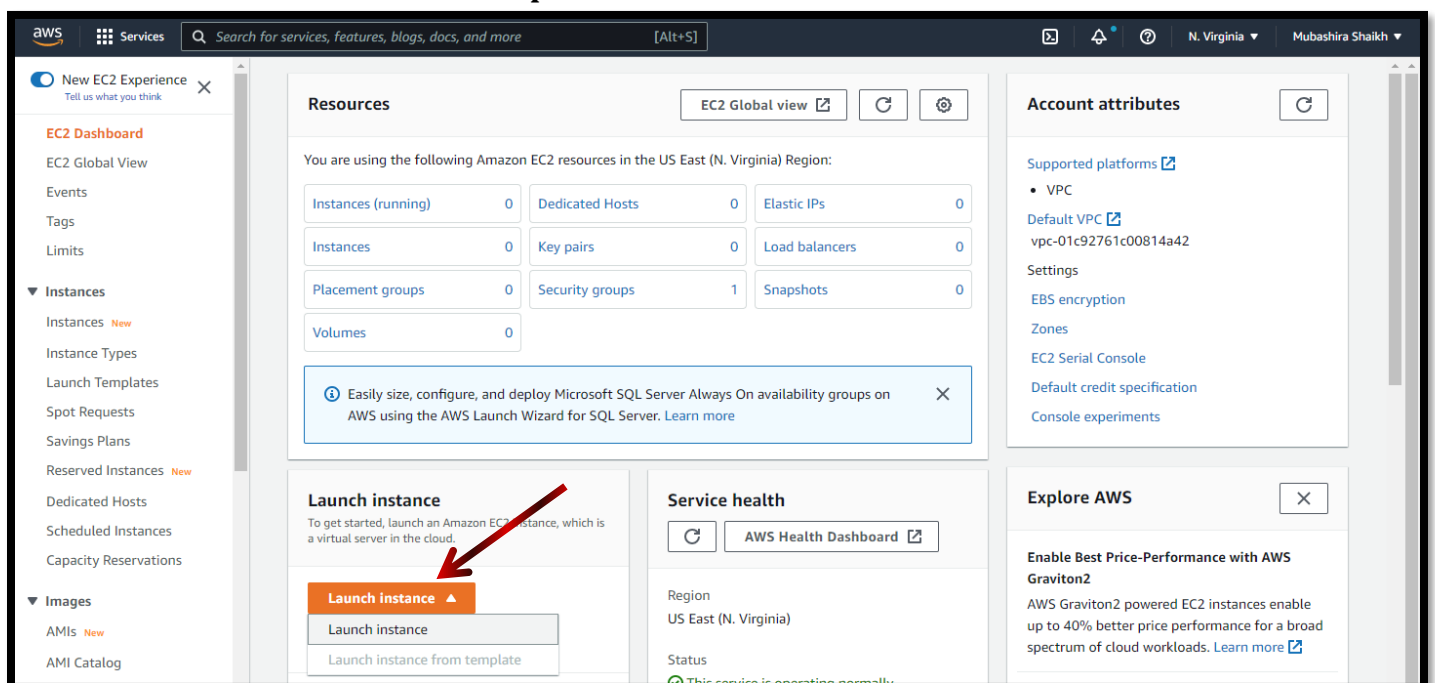
➔ Inside a container are all the necessary executables, binary code, libraries, and configuration files. Compared to server or machine virtualization approaches, however, containers share the machine's OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs. This makes them more lightweight and portable, with significantly less overhead.

4. Deploy a containerized web Application on AWS EC2 Linux. [Install Docker, pull nginx image and run it]. Pull python images and run the command to list all the locally stored docker images.

Step 1: AWS Management Console Dashboard → Click on 'EC2'



Step 2: Click on Launch instance



Step 3: Give a name to your instance and select Amazon Linux

aws Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia Mubashira Shaikh

You've been opted into the new launch experience. Find out more about this experience or send us feedback. You can still return to the previous version by opting-out. Opt out to the old experience

EC2 > Instances > Launch an instance

Launch an instance

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
Mubashira_Exp_7 Add additional tags

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-05fa00d4c63e32376

Virtual server type (instance type)
t2.micro

Firewall (security group)

aws Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia Mubashira Shaikh

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

Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat S Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)
Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
ami-05fa00d4c63e32376 (64-bit (x86)) / ami-05f3141013eebdc12 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs Free tier eligible

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-05fa00d4c63e32376

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

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

Step 4: Select instance type with free tier eligibility (t2.micro)

Instance type Info

Instance type
t2.micro Free tier eligible Compare instance types
Family: t2 1 vCPU 1 GiB Memory
On-Demand Linux pricing: 0.0116 USD per Hour
On-Demand Windows pricing: 0.0162 USD per Hour

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

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-05fa00d4c63e32376

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

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

Step 5: Create key pair. A .pem file will be downloaded which will be later used to connect to the instance

aws Services Search for services, features, blogs, docs, and more [Alt+S] N. Virginia Mubashira Shaikh

On-Demand Linux pricing: 0.0116 USD per Hour
On-Demand Windows pricing: 0.0162 USD per Hour

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
Select Create new key pair

Network settings Get guidance Edit

Network Info
vpc-01c92761c00814a42

Summary

Number of instances Info
1

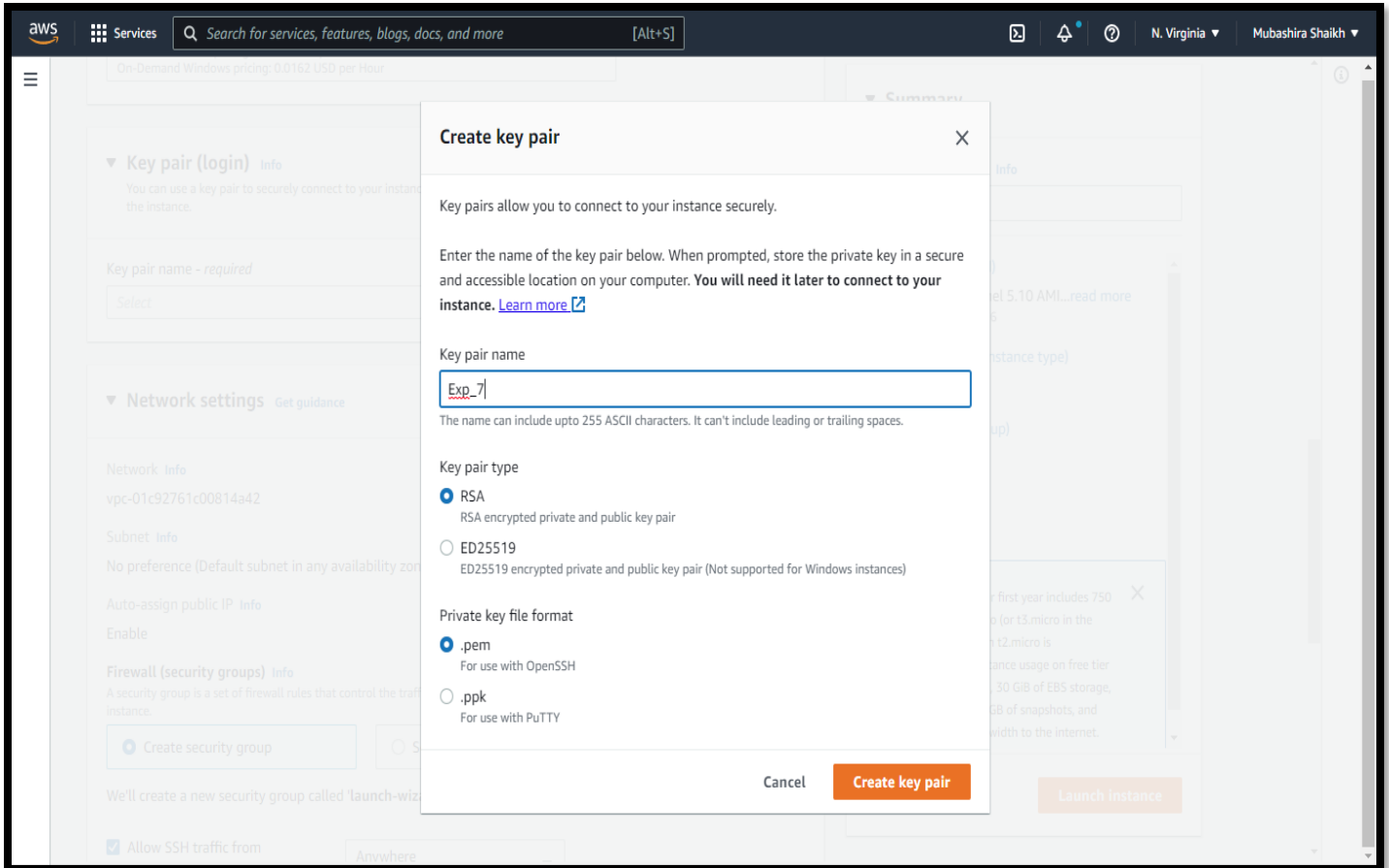
Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-05fa00d4c63e32376

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

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

Give name to key pair



Create key pair

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Key pair name

Exp-7

The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

☒ RSA
RSA encrypted private and public key pair

☐ ED25519
ED25519 encrypted private and public key pair (Not supported for Windows instances)

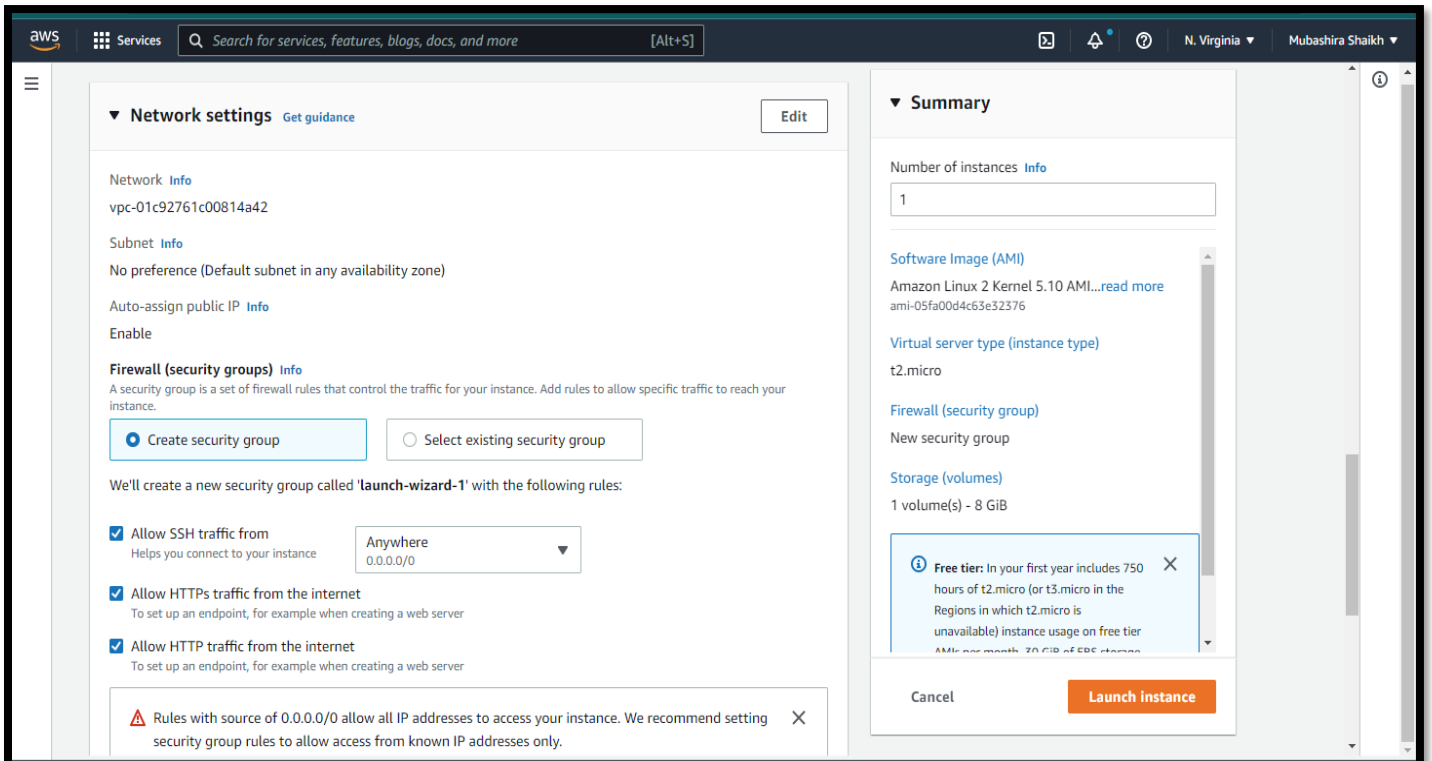
Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

Cancel Create key pair

Step 6: Network Settings → Select 'Allow SSH Traffic from' Anywhere, Allow HTTPs traffic from the internet and Allow HTTP traffic from the internet → Launch Instance



Network settings Get guidance Edit

Network Info
vpc-01c92761c00814a42

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

Auto-assign public IP Info
Enable

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

We'll create a new security group called 'launch-wizard-1' with the following rules:

☒ Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0

☒ Allow HTTPs traffic from the internet
To set up an endpoint, for example when creating a web server

☒ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

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

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-05fa00d4c63e32376

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. AMI per month: 20 GiB of EBS storage.

Cancel Launch Instance

Success message will be shown after successful creation of instance

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, 'Services' menu, a search bar, and user information. A blue banner at the top states: "You've been opted into the new launch experience. Find out more about this experience or send us feedback. You can still return to the previous version by opting-out." Below this, the breadcrumb trail is "EC2 > Instances > Launch an instance". A green success message box says: "Success Successfully initiated launch of instance (i-0ac3aac156ed312c7) Launch log". Under "Next Steps", it says "Get notified of estimated charges" and "How to connect to your instance". A "View all instances" button is at the bottom right.

Step 7: Launch Linux instance to get the remote host and username for SSH in MobaXterm

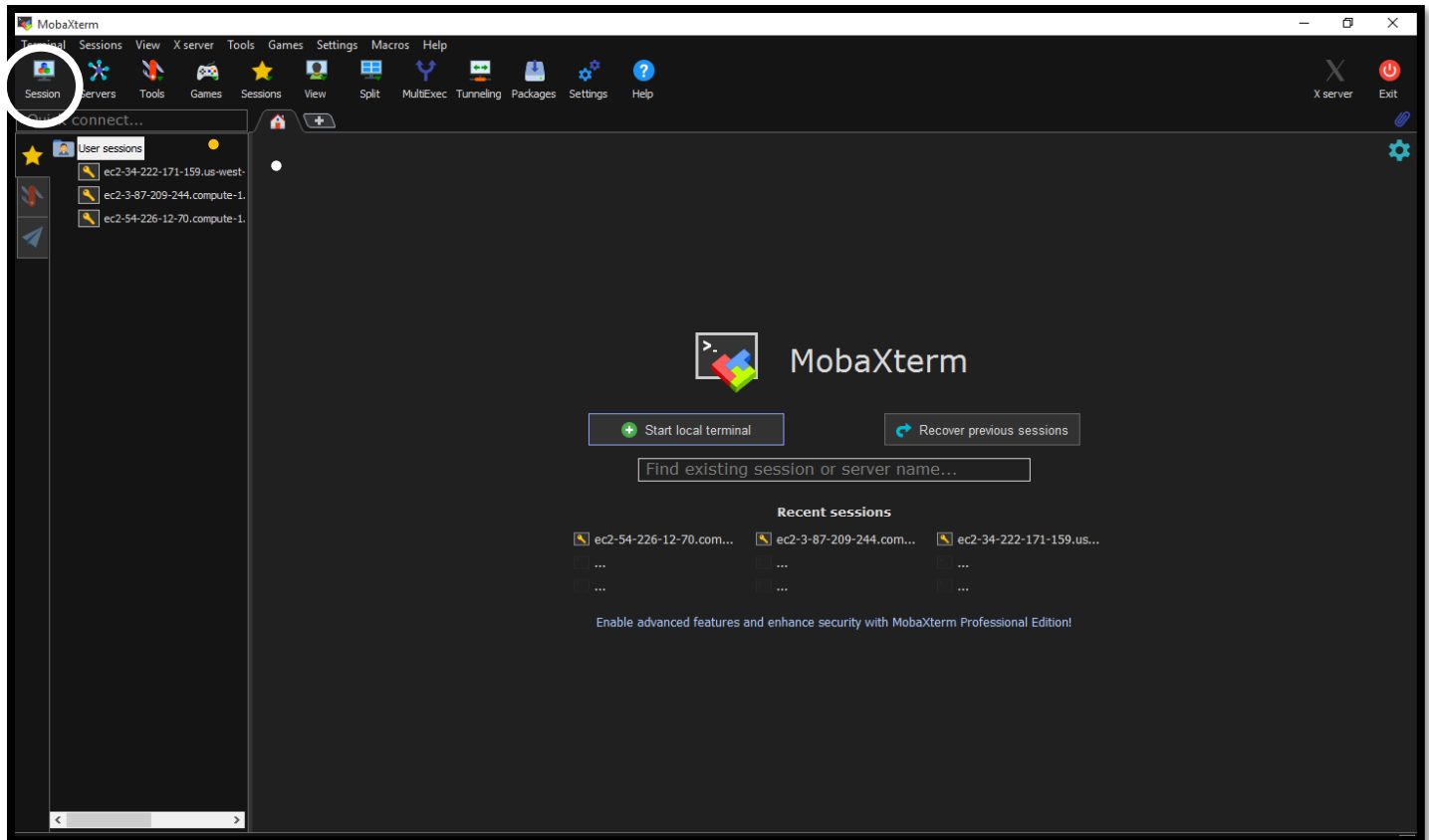
The screenshot shows the AWS Management Console 'Instances' page. The breadcrumb trail is "EC2 > Instances > i-0ac3aac156ed312c7 > Connect to instance". The 'Connect' button is circled in red. Below it, a table lists the instance details: Name (Mubashira_Exp_7), Instance ID (i-0ac3aac156ed312c7), Instance state (Running), Instance type (t2.micro), Status check (2/2 checks passed), Alarm status (No alarms), and Availability Zone (us-east-1c).

Go to SSH Client to get the remote host and username

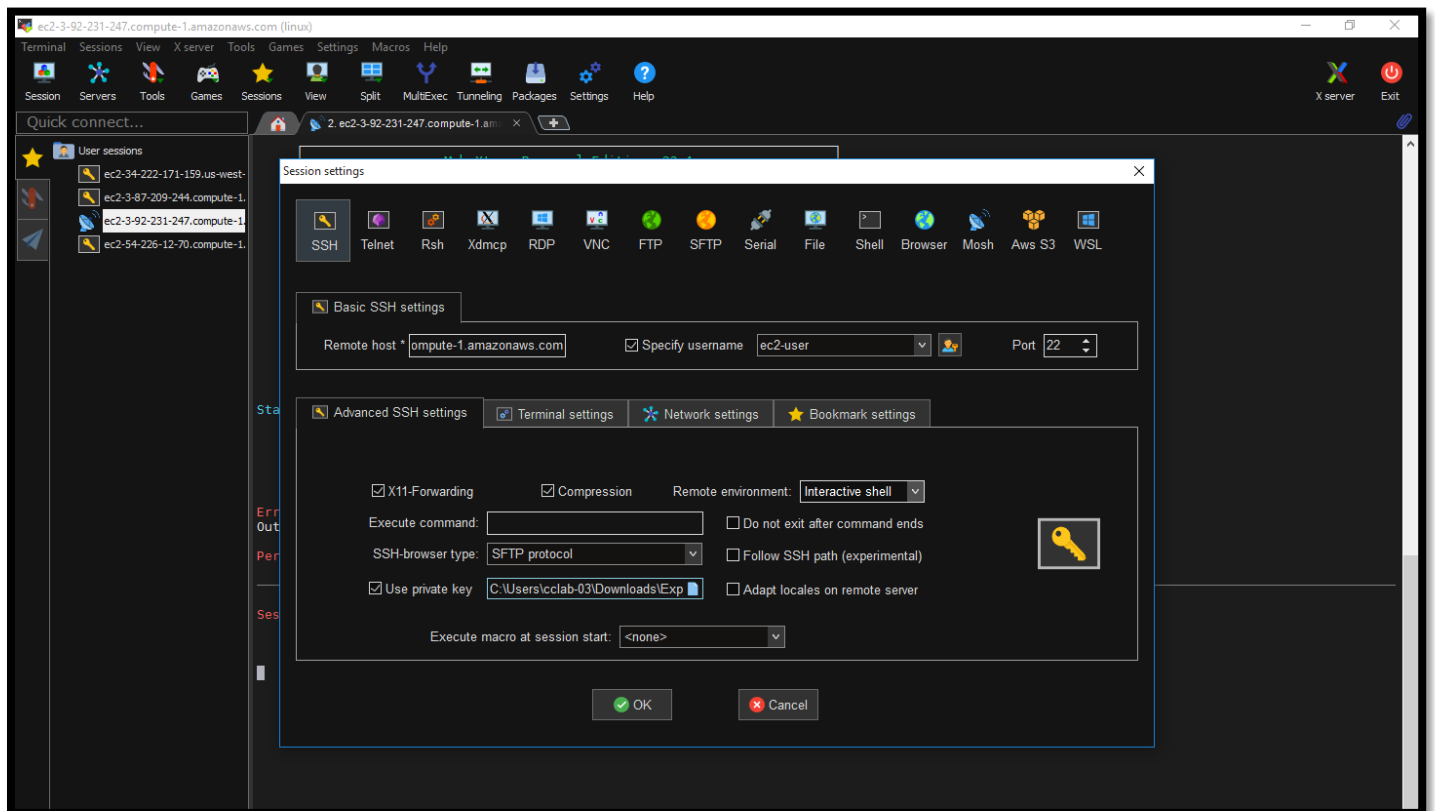
Copy the Public DNS which is your remote host and the username is the word before '@'

The screenshot shows the AWS Management Console 'Connect to instance' page. The breadcrumb trail is "EC2 > Instances > i-0ac3aac156ed312c7 > Connect to instance". The 'SSH client' tab is selected. It shows the Instance ID (i-0ac3aac156ed312c7 (Mubashira_Exp_7)) and a list of steps to connect. A green checkmark indicates "Public DNS copied". Below this, the Public DNS is shown: "ec2-3-92-231-247.compute-1.amazonaws.com". An example command is provided: "ssh -i 'Exp_7.pem' ec2-user@ec2-3-92-231-247.compute-1.amazonaws.com". A note at the bottom states: "Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name."

Step 8: Launch MobaXterm to connect the instance. Go to Sessions → New Session



Then select SSH. Fill the basic SSH settings and attach the .pem file downloaded earlier in advanced SSH settings 'Use private key' section. Then OK. Your Linux Instance will be running




```
ec2-3-92-231-247.compute-1.amazonaws.com (ec2-user)
Terminal Sessions View X server Tools Games Settings Macros Help
Quick connect...
/home/ec2-user/
Name
ssh
bash_logout
bash_profile
bashrc
Authenticating with public key "Imported-OpenSSH-Key"
MobaXterm Personal Edition v22.1
(SSH client, X server and network tools)
SSH session to ec2-user@ec2-3-92-231-247.compute-1.amazonaws.com
Direct SSH : ✓
SSH compression : ✓
SSH-browser : ✓
X11-forwarding : ✗ (disabled or not supported by server)
For more info, ctrl+click on help or visit our website.
Amazon Linux 2 AMI
https://aws.amazon.com/amazon-linux-2/
3 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-89-88 ~]$ sudo su
[root@ip-172-31-89-88 ec2-user]# yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 3.7 kB 00:00:00
Resolving Dependencies
--> Running transaction check
--> Package docker.x86_64 0:20.10.17-1.amzn2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.17-1.amzn2.x86_64
--> Processing Dependency: libcgrouper >= 0.40.rc1-5.15 for package: docker-20.10.17-1.amzn2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.17-1.amzn2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.17-1.amzn2.x86_64
--> Running transaction check
--> Package containerd.x86_64 0:1.6.6-1.amzn2 will be installed
--> Package libcgrouper.x86_64 0:0.41-21.amzn2 will be installed
--> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
--> Package runc.x86_64 0:1.1.3-1.amzn2 will be installed
--> Finished Dependency Resolution
Dependencies Resolved
Package Arch Version Repository Size
Installing:
```

Step 9: Run the command following commands:

- “docker –version” to check the version of docker
- “docker images” to see the list of Docker images on the system
- “docker service start” to start one or more stopped containers.
- “docker pull nginx” to download a Docker image
- “docker run -p80:80 nginx” to map TCP port 80 in the container to port 8080 on the Docker host

```
[root@ip-172-31-89-88 ec2-user]# docker --version
bash: docker--version: command not found
[root@ip-172-31-89-88 ec2-user]# docker --version
bash: docker--version: command not found
[root@ip-172-31-89-88 ec2-user]# docker --version
Docker version 20.10.17, build 100c701
[root@ip-172-31-89-88 ec2-user]# docker images
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
[root@ip-172-31-89-88 ec2-user]# service docker start
Redirecting to /bin/systemctl start docker.service
[root@ip-172-31-89-88 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
[root@ip-172-31-89-88 ec2-user]# docker pull nginx
bash: nginx: command not found
[root@ip-172-31-89-88 ec2-user]# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
7a6db449b51b: Pull complete
ca1981974b58: Pull complete
d4019c921e20: Pull complete
7cb804d746d4: Pull complete
e7a561826262: Pull complete
7247f6e5c182: Pull complete
Digest: sha256:b95a99febf7797479e0c5eb5ec0bdfa5d9f504bc94da550c2f58e839ea6914f
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
[root@ip-172-31-89-88 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
nginx latest 2b7d6430f78d 2 days ago 142MB
[root@ip-172-31-89-88 ec2-user]# docker run -p80:80 nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2022/08/25 06:53:54 [notice] 1#1: using the "epoll" event method
2022/08/25 06:53:54 [notice] 1#1: nginx/1.23.1
2022/08/25 06:53:54 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2022/08/25 06:53:54 [notice] 1#1: OS: Linux 5.10.130-118.517.amzn2.x86_64
2022/08/25 06:53:54 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 32768:65536
2022/08/25 06:53:54 [notice] 1#1: start worker processes
2022/08/25 06:53:54 [notice] 1#1: start worker process 31
```


Copy Public DNS(IPv4)

The screenshot shows the AWS Management Console interface. On the left, there is a navigation menu with options like 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Tags', 'Limits', 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Scheduled Instances', 'Capacity Reservations', 'Images', 'AMIs', 'AMI Catalog', and 'Elastic Block Store'. The main area displays the details of an EC2 instance named 'Mubashira_E...'. The instance ID is 'i-0ac3aac156ed312c7', the instance type is 't2.micro', and the availability zone is 'us-east-1c'. The instance state is 'running'. The Public DNS (IPv4) is 'ec2-3-92-231-247.compute-1.amazonaws.com'. The IPv4 Public IP is '3.92.231.247'. The Public DNS (IPv4) field is highlighted in blue and circled in red. An arrow points to the Public DNS (IPv4) field.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Pub
Mubashira_E...	i-0ac3aac156ed312c7	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-3-92-231-247.com...	3.92.231.247

Instance: i-0ac3aac156ed312c7 (Mubashira_Exp_7) Public DNS: ec2-3-92-231-247.compute-1.amazonaws.com

Description		Status Checks	Monitoring	Tags	
Instance ID	i-0ac3aac156ed312c7	Instance state	running	IPv4 Public IP	3.92.231.247
Instance type	t2.micro	Instance type	t2.micro	IPv6 IPs	-

Paste it on chrome

The screenshot shows a Chrome browser window with the address bar displaying 'ec2-3-92-231-247.compute-1.amazonaws.com'. The page content includes a welcome message and links to nginx.org and nginx.com.

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Run command “docker ps” to see the status of the process
and “docker ps -a” to show all the containers both stopped and running

```
[root@ip-172-31-89-88 ec2-user]# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
[root@ip-172-31-89-88 ec2-user]# docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
e81b96e3e4be   nginx    "/docker-entrypoint..."  2 minutes ago   Exited (0) 22 seconds ago   zealous_mahavira
abc1c5232856   nginx    "/docker-entrypoint..."  9 minutes ago   Exited (0) 2 minutes ago   beautiful_goldberg
[root@ip-172-31-89-88 ec2-user]#
```

Step 10: Select the instance and terminate it

