



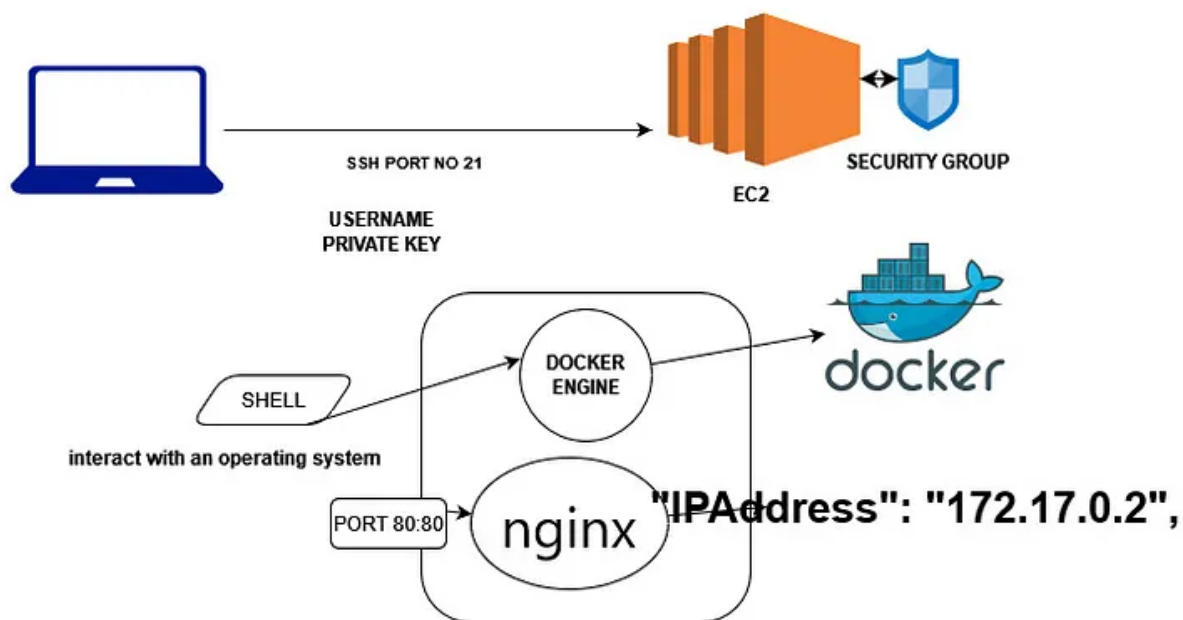
Manikanta Suru

Mar 14 · 4 min read · [Listen](#)

Save

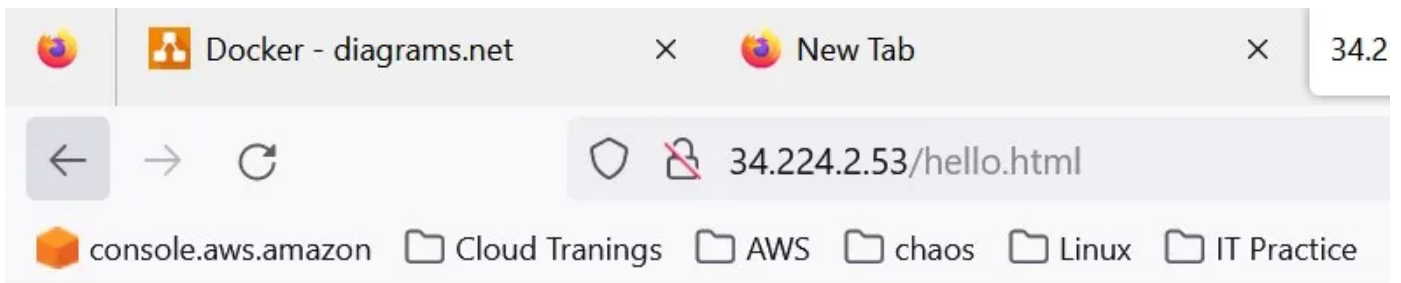


Let's Understand “Deploying a Static Web Site on Docker with Nginx and EC2 Set by step.



Let's understand Deploying a Web Site on Docker, with Nginx and EC2





Hello cloudncloud Community

1. *Create a Security Group (Firewall).*
2. *Launch EC2.*
3. *Connect through MobaXterm/Putty/super Putty*
4. *Install Docker in EC2*
5. *Pull the Image from the Docker hub and commands with a brief Explanation real-time use case.*

1. Create a Security Group:-

- Create a new security group
- Inbound rules: ALL TCP and 0.0.0.0/0
- Outbound rules: ALL TCP and 0.0.0.0/0

Create security group [info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [info](#)
 Docker-ALLOW-all
Name cannot be edited after creation.

Description [info](#)
 This Security Group access to all to ports any ip

VPC [info](#)
 vpc-0f02beb86dfa4b1c1

Inbound rules [info](#)

Type	Protocol	Port range	Source	Description - optional	
All TCP	TCP	0 - 65535	Anywhere-IPv4	0.0.0.0/0	Delete

Add rule

Outbound rules [info](#)

Type	Protocol	Port range	Destination	Description - optional	
All TCP	TCP	0 - 65535	Custom	0.0.0.0/0	Delete

Add rule

Select -> Create Security Group

Security group (sg-0477f9b536a540205 | Docker-ALLOW-all) was created successfully
[Details](#)

EC2 > Security Groups > sg-0477f9b536a540205 - Docker-ALLOW-all

sg-0477f9b536a540205 - Docker-ALLOW-all Actions

Details

Security group name Docker-ALLOW-all	Security group ID sg-0477f9b536a540205	Description This Security Group access to all to ports any ip	VPC ID vpc-0f02beb86dfa4b1c1
Owner 557822060553	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

2. Launch EC2.

select >Amazon >linux

Amazon Linux
aws

macOS
Mac

Ubuntu
ubuntu

Windows
Microsoft

Red Hat
Red Hat

S

Search

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-005f9685cb30f234b (64-bit (x86)) / ami-05a66dc4a507a82cc (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Select Key pair

▼ 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

Docker ▼

Create new key pair

Select the already existing Security Group

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

Security groups Info

Select security groups ▼

Docker-ALLOW-all sg-0477f9b536a540205 X
VPC: vpc-0f02beb86dfa4b1c1

Compare security group rules

EC2 > Instances > Launch an instance



Success

Successfully initiated launch of instance (i-03acdf8e2cb36d5b6)

► Launch log

3. Connect through MobaXterm/Putty/super Putty

The screenshot shows a MobaXterm terminal window with a dark background. The title bar indicates the connection is to '2. 34.224.2.53 (ec2-user)'. The terminal displays the MobaXterm version (v22.3) and its features. It shows the details of an SSH session to 'ec2-user@34.224.2.53', including direct SSH, compression, browser, and X11-forwarding status. Below this, there is a ASCII art logo for Amazon Linux 2 AMI and a URL to the Amazon Linux 2 website. The prompt shows the user is 'ec2-user' on the IP '172-31-61-150'.

```
• MobaXterm Personal Edition v22.3 •
(SSH client, X server and network tools)

► SSH session to ec2-user@34.224.2.53
  • 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/
[ec2-user@ip-172-31-61-150 ~]$
```

4. Install Docker in EC2

Update OS packages: `sudo yum update -y`

Install Docker

`sudo amazon-linux-extras install docker`

```
Running transaction
Installing : runc-1.1.4-1.amzn2.0.1.x86_64 1/
Installing : containerd-1.6.8-1.amzn2.0.1.x86_64 2/
Installing : libcgrouper-0.41-21.amzn2.x86_64 3/
Installing : pigz-2.3.4-1.amzn2.0.1.x86_64 4/
Installing : docker-20.10.17-1.amzn2.0.2.x86_64 5/
Verifying : containerd-1.6.8-1.amzn2.0.1.x86_64 1/
Verifying : pigz-2.3.4-1.amzn2.0.1.x86_64 2/
Verifying : libcgrouper-0.41-21.amzn2.x86_64 3/
Verifying : docker-20.10.17-1.amzn2.0.2.x86_64 4/
Verifying : runc-1.1.4-1.amzn2.0.1.x86_64 5/

Installed:
docker.x86_64 0:20.10.17-1.amzn2.0.2

Dependency Installed:
containerd.x86_64 0:1.6.8-1.amzn2.0.1      libcgrouper.x86_64 0:0.41-21.amzn2      pigz.x86_64 0:2.3.4-1.amzn2.0.1
runc.x86_64 0:1.1.4-1.amzn2.0.1
```

5. Pull the Image from the Docker hub and commands with a brief Explanation real-time use case.

`sudo service docker start`

```
[ec2-user@ip-172-31-61-150 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-61-150 ~]$
```

command is used to enable the Docker service to start automatically at boot time on systems that use systemd as the init system *Enable it*

`sudo systemctl enable docker`

```
[ec2-user@ip-172-31-61-150 ~]$ sudo systemctl enable docker
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
```

Add the user to the docker group

`sudo usermod -a -G docker ec2-user`

Map a container port to a host port.

`docker run -d -p 80:80 nginx`

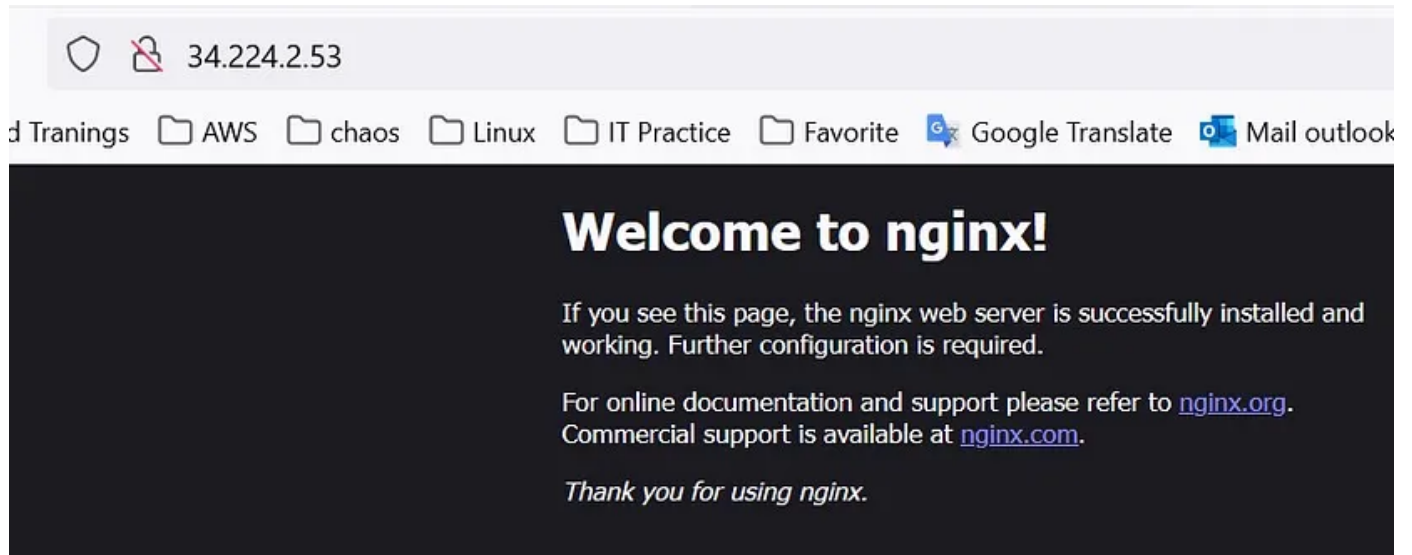
`docker run -d -p <any-host-port>:80 nginx`


```
[ec2-user@ip-172-31-61-150 ~]$ docker run -d -p 80:80 nginx
d9c7f79557cfda8042bab2262b3fba736607469c6307ad8820137f9b30f13d0e
[ec2-user@ip-172-31-61-150 ~]$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NA
d9c7f79557cf	nginx	"/docker-entrypoint..."	12 seconds ago	Up 12 seconds	0.0.0.0:80→80/tcp, :::80→80/tcp	de
ff4162b6e864	904b8cb13b93	"/docker-entrypoint..."	33 minutes ago	Up 17 minutes	80/tcp	in

```
[ec2-user@ip-172-31-61-150 ~]$
```

Go to any Browser and type your EC2 Instance ip on url tab



i want to login to the nginx container and see what is there inside change web page add new HTML page

`docker exec -it <ID> bash`

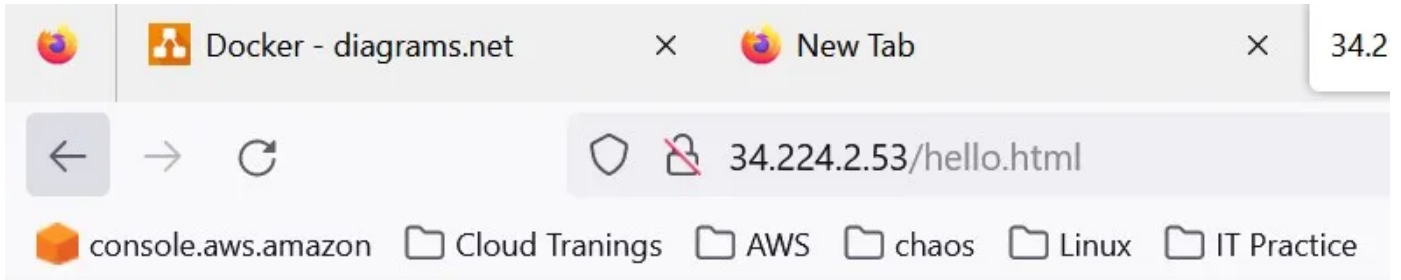
```
[ec2-user@ip-172-31-61-150 ~]$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS                               NAMES
d9c7f79557cf   nginx    "/docker-entrypoint..." 10 minutes ago Up 10 minutes 0.0.0.0:80→80/tcp, :::80→80/tcp determined_p
ff4162b6e864   904b8cb13b93 "/docker-entrypoint..." 44 minutes ago Up 28 minutes 80/tcp intelligent_
kilby
[ec2-user@ip-172-31-61-150 ~]$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS                               NAMES
d9c7f79557cf   nginx    "/docker-entrypoint..." 10 minutes ago Up 10 minutes 0.0.0.0:80→80/tcp, :::80→80/tcp determined_p
ff4162b6e864   904b8cb13b93 "/docker-entrypoint..." 44 minutes ago Up 28 minutes 80/tcp intelligent_
kilby
[ec2-user@ip-172-31-61-150 ~]$ docker exec -it d9c7f79557cf bash
root@d9c7f79557cf:/#
```

`cd /usr/share/nginx/html`

`root@d9c7f79557cf:/usr/share/nginx/html# echo "Hello cloudnloud Community" > hello.html`

```
root@d9c7f79557cf:/usr/share/nginx/html# rm -f echo Hello cloudncloud Community >hello.html
root@d9c7f79557cf:/usr/share/nginx/html# echo "<font size='20'>Hello cloudncloud Community</font>" > hello.html
```

Go to any Browser and type your EC2 Instance ip on url tab



Hello cloudncloud Community

Docker commands with a brief Explanation of real-time use case.

docker images → will show the images available in the server

```
[ec2-user@ip-172-31-61-150 ~]$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
```

docker pull <name>:<version> → will pull the image from docker hub

docker pull nginx

nginx - Official Image | Docker Hub

Official build of Nginx.

hub.docker.com


```
[ec2-user@ip-172-31-61-150 ~]$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
3f9582a2cbe7: Pull complete
9a8c6f286718: Pull complete
e81b85700bc2: Pull complete
73ae4d451120: Pull complete
6058e3569a68: Pull complete
3a1b8f201356: Pull complete
Digest: sha256:aa0afebbb3cfa473099a62c4b32e9b3fb73ed23f2a75a65ce1d4b4f55a5c2ef2
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
[ec2-user@ip-172-31-61-150 ~]$ docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
Digest: sha256:aa0afebbb3cfa473099a62c4b32e9b3fb73ed23f2a75a65ce1d4b4f55a5c2ef2
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest
```

`docker create <image-id>` — create container out of image

```
[ec2-user@ip-172-31-61-150 ~]$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
nginx         latest    904b8cb13b93   12 days ago    142MB
[ec2-user@ip-172-31-61-150 ~]$ docker create 904b8cb13b93
ff4162b6e86457ce679546100be3fe11d3cc895c3525ef26ea7313be8f3e1fdf
[ec2-user@ip-172-31-61-150 ~]$
```

`docker ps` — will show running containers it

```
[ec2-user@ip-172-31-61-150 ~]$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
ff4162b6e864   904b8cb13b93   "/docker-entrypoint..." 16 minutes ago Up About a minute 80/tcp       intelligent_kilby
```

`docker stop <container-id>` -> it will stop the container

```
[ec2-user@ip-172-31-61-150 ~]$ docker stop d9c7f79557cf
d9c7f79557cf
[ec2-user@ip-172-31-61-150 ~]$
```

`docker rm <container-id>` -> delete the container

`docker rmi <image-id>` — will remove images

`docker inspect <container-id>`

```
"SandboxKey": "/var/run/docker/netns/20be8d4c66bd",
"SecondaryIPAddresses": null,
"SecondaryIPv6Addresses": null,
"EndpointID": "305ff1cabd936a435f874c79d85dde709454758118034deeb6c22e2df6927640",
"Gateway": "172.17.0.1",
"GlobalIPv6Address": "",
"GlobalIPv6PrefixLen": 0,
"IPAddress": "172.17.0.2",
"IPPrefixLen": 16,
"IPv6Gateway": "",
"MacAddress": "02:42:ac:11:00:02",
"Networks": {
  "bridge": {
    "IPAMConfig": null,
    "Links": null,
    "Aliases": null,
    "NetworkID": "9713adcc17443331a90a145b309fc3885585532604a29b25c59cd77a40eda850",
    "EndpointID": "305ff1cabd936a435f874c79d85dde709454758118034deeb6c22e2df6927640",
    "Gateway": "172.17.0.1",
    "IPAddress": "172.17.0.2",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "MacAddress": "02:42:ac:11:00:02",
    "DriverOpts": null
  }
}
```

Happy Learning

👉 In case you would like to continue the discussion, you can always reach out to me on [Twitter](#) or on [LinkedIn](#) for professional networking, if you feel like following me on [GitHub](#) you can also do that.

👉 Follow [Cloudncloud Tech Community](#) for more insightful knowledge & resources & [CloudnLoud@youtube](#) YouTube channel

[Docker](#)

[AWS](#)

[Docker Image](#)

[Community](#)