

Aman Yadav

D15C

Roll No: 60

## Experiment 1a: Static Hosting

Successfully created bucket "amanbucket60"  
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

View Storage Lens dashboard

General purpose buckets    Directory buckets

General purpose buckets (2) [Info](#) All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
amanbucket60	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 16, 2024, 22:55:57 (UTC+05:30)

amanbucket60 [Info](#)

Objects    Properties    Permissions    Metrics    Management    Access Points

Objects (0) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Name	Type	Last modified	Size	Storage class
No objects You don't have any objects in this bucket.				

[Upload](#)

**Files and folders (1 Total, 251.0 B)**

All files and folders in this table will be uploaded.

<input type="checkbox"/>	Name	Folder	Type
<input type="checkbox"/>	s3.html	S3/	text/html

**Destination** [Info](#)

Destination  
<s3://amanbucket60>

► **Destination details**  
Bucket settings that impact new objects stored in the specified destination.

**Upload succeeded**  
View details below.

Destination	Succeeded	Failed
<a href="s3://amanbucket60">s3://amanbucket60</a>	1 file, 251.0 B (100.00%)	0 files, 0 B (0%)

[Files and folders](#) [Configuration](#)

**Files and folders (1 Total, 251.0 B)**

<input type="checkbox"/>	Name	Folder	Type	Size	Status	Error
<input type="checkbox"/>	<a href="#">s3.html</a>	S3/	text/html	251.0 B	Succeeded	-

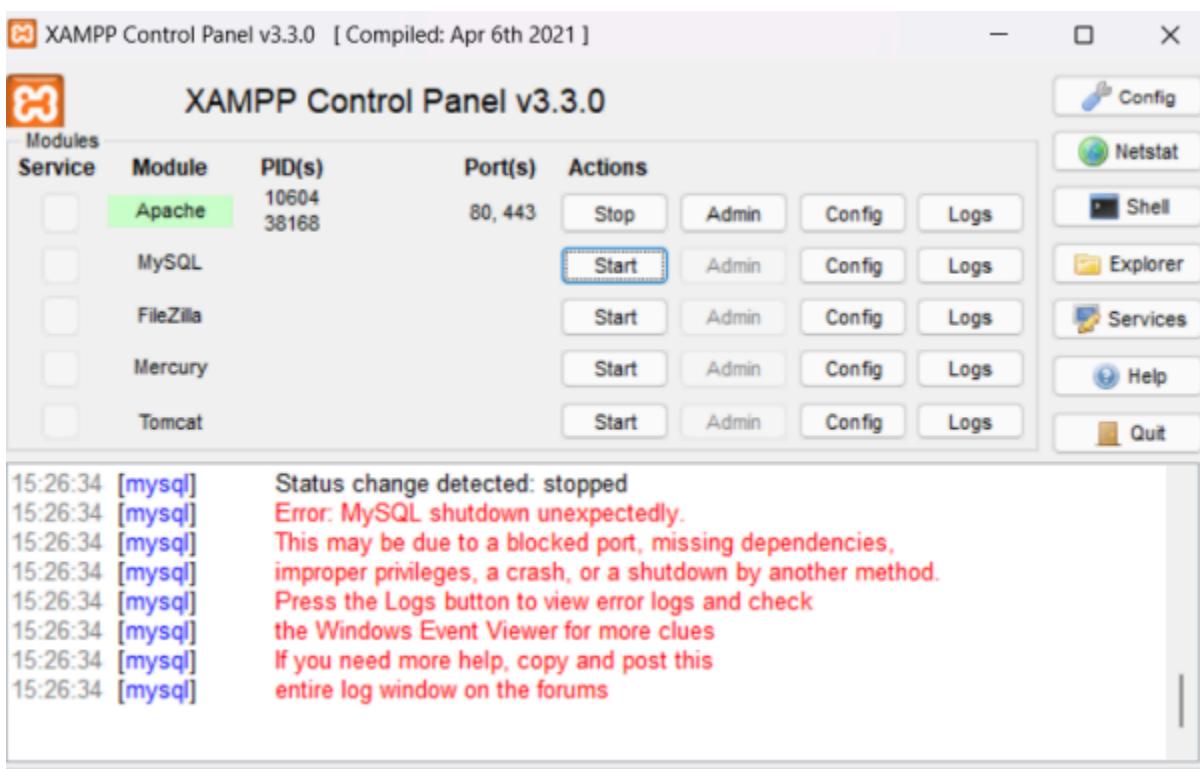
Hello, Welcome to S3 bucket

## Hosting a static website on Xampp

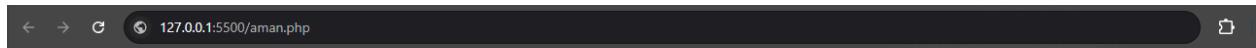
Create a .php file in some local repository

📁 dashboard	29-07-2024 14:20	File folder
📁 img	29-07-2024 14:20	File folder
📁 webalizer	29-07-2024 14:20	File folder
📁 xampp	29-07-2024 14:20	File folder
chrome applications	15-06-2022 21:37	Chrome HTML Do... 4 KB
# bitnami	15-06-2022 21:37	CSS Source File 1 KB
favicon	16-07-2015 21:02	ICO File 31 KB
index00	16-07-2015 21:02	PHP Source File 1 KB

Make sure you have installed Xampp on your local machine. After the installation, start Xampp control panel and start modules named Apache and MySql



Access the contents of the php file by typing localhost/your\_file.php on your browser. We have successfully hosted our php file on our local machine using Xampp



**Hello this is Aman Yadav**

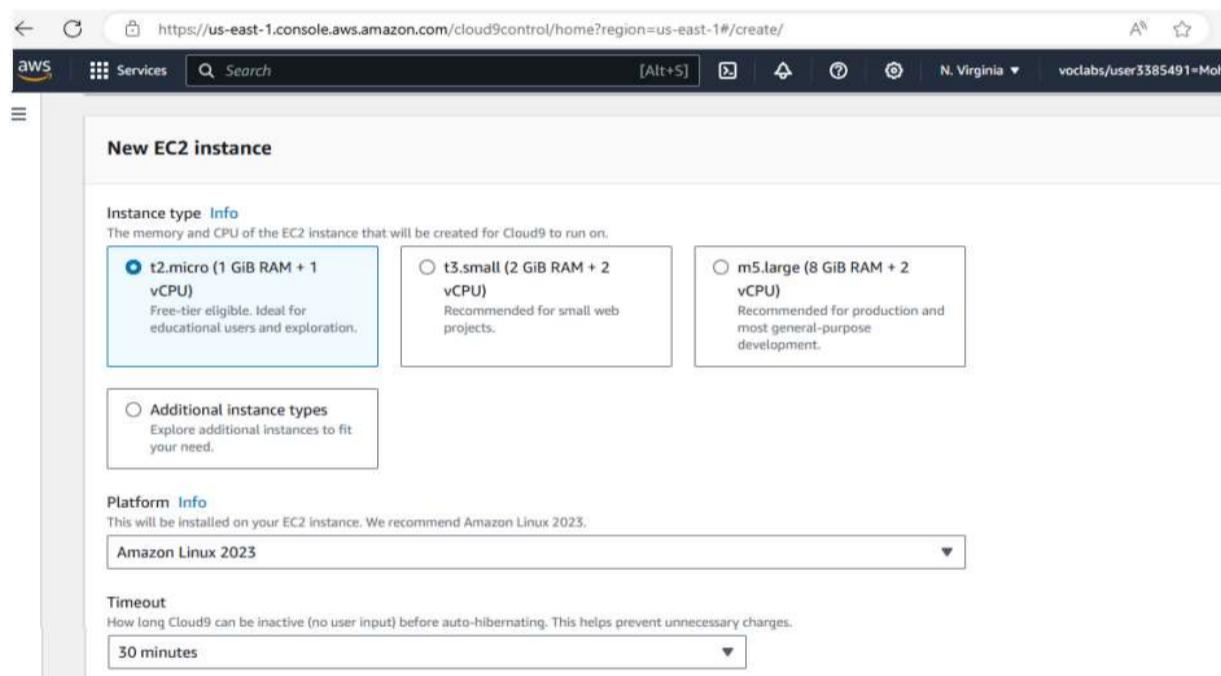
Name: Aman Yadav

Div: D15C

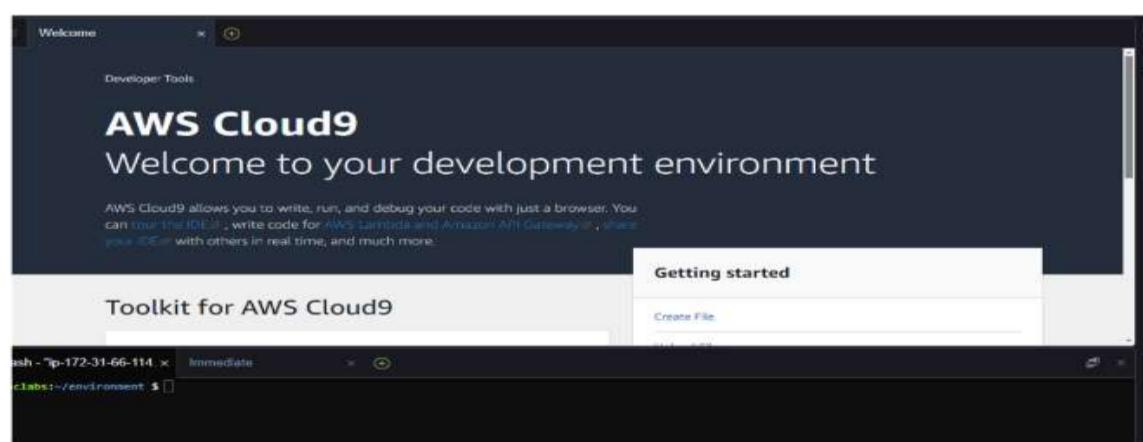
Roll No: 60

### Experiment 1b: Cloud9 Setup and Launch, Collaboration demonstration by creation of IAM groups and users.

Open your AWS account and search for Cloud9 service inside Developer tools. Create a new Cloud9 environment by filling in the required details. Make sure you use an EC2 instance to create your environment



We have successfully setup and launched our Cloud9 environment.



Now we are supposed to create a new user. Give a suitable name to the user and decide the password for the same.

The screenshot shows the 'Specify user details' step of the AWS IAM 'Create user' wizard. The URL is https://us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#users/create. The 'User name' field contains 'sahil motiramani'. A note below it says: 'The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ \_ (hyphen)'. There is an optional checkbox for 'Provide user access to the AWS Management Console' which is unchecked. A note next to it says: 'If you're providing console access to a person, it's a best practice [link] to manage their access in IAM Identity Center.' A callout box highlights the 'I want to create an IAM user' option, which is selected. It says: 'We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keypairs, or a backup credential for emergency account access.' Below this, there are options for 'Console password': 'Autogenerated password' (unchecked) and 'Custom password' (checked), with a password field containing '\*\*\*\*\*'. A note next to it says: 'Users must create a new password at next sign-in - Recommended' and 'Users automatically get the IAMUserChangePassword [link] policy to allow them to change their own password.' A callout box at the bottom right of the form area says: 'If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user. [Learn more link]'.

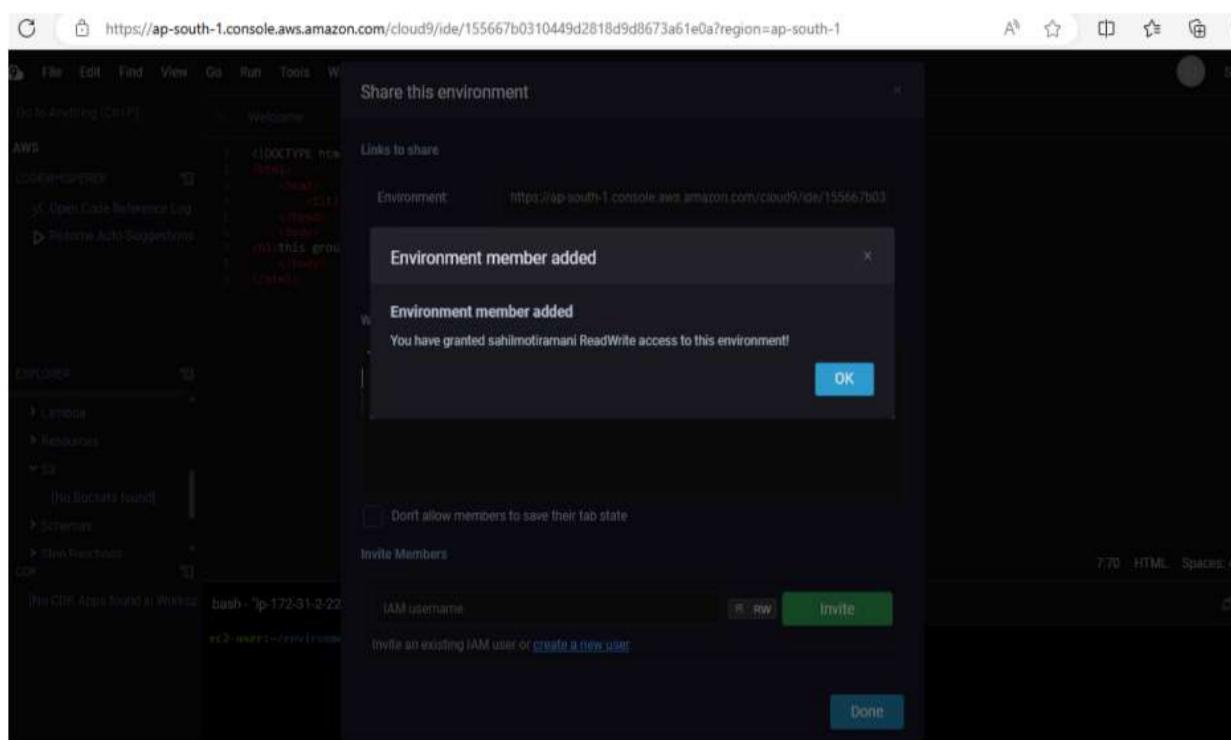
Similarly, create a new group and provide a suitable name for the same. Include the IAM users in this group together for our convenience i.e to provide similar kinds of permissions to the entire group rather than an individual user.

The screenshot shows the AWS IAM console. A green header bar indicates 'MSBCLoud9 user group created.' Below it, a 'Review and create' section shows 'Step 4' and 'Retrieve password'. Three options are listed: 'Add user to group' (selected), 'Copy permissions', and 'Attach policies directly'. The main area displays 'User groups (1/1)' with a table showing one group named 'MSBCLoud9' created on '2024-07-29 (Now)'. A note 'Set permissions boundary - optional' is present. Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom.

The user has successfully been created i.e There is a custom made username and a password for the IAM user.

The screenshot shows the AWS IAM console. A green header bar indicates 'MSBCLoud9 user group created.' Below it, a 'User details' section shows the user 'sahilmotiramani' with a console password type 'Custom password' and 'Require password reset' set to 'Yes'. The 'Permissions summary' section lists two policies: 'IAMUserChangePassword' (AWS managed, Permissions policy) and 'MSBCLoud9' (Group, Permissions group). The 'Tags - optional' section notes 'No tags associated with the resource.'

Go back to the cloud9 environment. Click on share this environment option so as to allow other collaborators to access your environment.



Further, we are supposed to log in from another browser using the credentials of the IAM user, so as to access the shared cloud9 environment with us. These steps could not be completed because Cloud9 services have been disrupted and there is no access to the IAM user from the remote login.

Name: Aman Yadav

Class: D15C

Roll No: 60

Experiment 2: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

Login into your AWS account and navigate to services. Search for Elastic Beanstalk service and click on create application. Give your application a suitable name.

Name, review, and create

Role details

Role name  
Enter a meaningful name to identify this role.  
Maximum 64 characters. Use alphanumeric and '+=\_,@-' characters.

Description  
Add a short explanation for this role.  
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: '\_+=,. @-/[\{\}]!#\$%^&\*();:<>'

Step 1: Select trusted entities Edit

Now, while creating the environment, we are asked to provide an IAM role with the necessary EC2 permissions. We are supposed to make sure that we have made an existing IAM role with the following set of permissions:

1. AWSElasticBeanStalkWebTier
2. AWSElasticBeanStalkWorkerTier
3. AWSElasticBeanStalkMulticontainerDocker

Step 2: Add permissions Edit

Permissions policy summary

Policy name	Type	Attached as
<a href="#">AWSElasticBeanstalkMulticontainerDocker</a>	AWS managed	Permissions policy
<a href="#">AWSElasticBeanstalkWebTier</a>	AWS managed	Permissions policy
<a href="#">AWSElasticBeanstalkWorkerTier</a>	AWS managed	Permissions policy

## Application information Info

### Application name

Maximum length of 100 characters.

### ► Application tags (optional)

## Environment information Info

Choose the name, subdomain and description for your environment. These cannot be changed later.

## Service access

IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. [Learn more](#)

### Service role

- Create and use new service role  
 Use an existing service role

### Existing service roles

Choose an existing IAM role for Elastic Beanstalk to assume as a service role. The existing IAM role must have the required IAM managed policies.



### EC2 key pair

Select an EC2 key pair to securely log in to your EC2 instances. [Learn more](#)



### EC2 instance profile

Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.



## Architecture

The processor architecture determines the instance types that are made available. You can't change this selection after you create the environment. [Learn more](#)

**x86\_64**

This architecture uses x86 processors and is compatible with most third-party tools and libraries.

**arm64 - new**

This architecture uses AWS Graviton2 processors. You might have to recompile some third-party tools and libraries.

## Instance types

Add instance types for your fleet. Change the order that the instances are in to set the preferred launch order. This only affects On-Demand instances. We recommend you include at least two instance types. [Learn more](#)

Choose x86 instance types ▾

t3.micro X t3.small X

## AMI ID

Elastic Beanstalk selects a default Amazon Machine Image (AMI) for your environment based on the Region, platform version, and processor architecture that you choose. [Learn more](#)

ami-00eb04e7d66370650

## Availability Zones

Number of Availability Zones (AZs) to use.

Any ▾

 Elastic Beanstalk is launching your environment. This will take a few minutes. X

[Elastic Beanstalk](#) > [Environments](#) > Amanapp-env

Amanapp-env [Info](#)



Actions ▾

Upload and deploy

### Environment overview

#### Health

 Pending

#### Environment ID

 e-ewxmkkvgzx

#### Domain

Amanapp-env.eba-89wnxzg2.eu-north-1.elasticbeanstalk.com [Copy](#)

#### Application name

aman\_app

### Platform

[Change version](#)

#### Platform

PHP 8.3 running on 64bit Amazon Linux 2023/4.3.2

#### Running version

-

#### Platform state

 Supported

[Events](#)

[Health](#)

[Logs](#)

[Monitoring](#)

[Alarms](#)

[Managed updates](#)

[Tags](#)

# Advanced DevOps Lab

## Experiment:3

**Aim:** To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

### **Theory:**

Container-based microservices architectures have profoundly changed the way development and operations teams test and deploy modern software. Containers help companies modernize by making it easier to scale and deploy applications, but containers have also introduced new challenges and more complexity by creating an entirely new infrastructure ecosystem.

Large and small software companies alike are now deploying thousands of container instances daily, and that's a complexity of scale they have to manage. So how do they do it?

Enter the age of Kubernetes.

Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. In fact, Kubernetes has established itself as the defacto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), backed by key players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes makes it easy to deploy and operate applications in a microservice architecture. It does so by creating an abstraction layer on top of a group of hosts so that development teams can deploy their applications and let Kubernetes manage the following activities:

- Controlling resource consumption by application or team
- Evenly spreading application load across a hosting infrastructure
- Automatically load balancing requests across the different instances of an application
- Monitoring resource consumption and resource limits to automatically stop applications from consuming too many resources and restarting the applications again
- Moving an application instance from one host to another if there is a shortage of resources in a host, or if the host dies
- Automatically leveraging additional resources made available when a new host is added to the cluster
- Easily performing canary deployments and rollbacks

**Steps:**

1. Create 3 EC2 Ubuntu Instances on AWS.

(Name 1 as Master, the other 2 as worker-1 and worker-2)

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances. There are three rows: node-1 (selected), node-2, and master. All three instances are in the 'Running' state. The instance type is 't3.micro' for all. The status check is '3/3 checks passed' for all. The availability zone is 'eu-north-1' for all. Below the table, a detailed view for 'node-1' is shown, including its instance ID, public and private IP addresses, and instance state.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability zone
node-1	i-0de4d1c4ed2c7b301	Running	t3.micro	3/3 checks passed	View alarms +	eu-north-1
node-2	i-0532cdd454d39c9ab	Running	t3.micro	3/3 checks passed	View alarms +	eu-north-1
master	i-02870ecd634f6f2f3	Running	t3.micro	3/3 checks passed	View alarms +	eu-north-1

2. Edit the Security Group Inbound Rules to allow SSH

The screenshot shows the AWS Security Groups page for the 'i-0839d4c5452845de2 (master)' security group. The 'Inbound rules' section is expanded, showing one rule: a TCP rule on port 22 from 0.0.0.0/0. The 'Source' column shows '0.0.0.0/0'.

Name	Security group rule ID	Port range	Protocol	Source
-	sgr-0b850473240f23536	22	TCP	0.0.0.0/0

3. SSH into all 3 machines

Now open the folder in the terminal 3 times for Master, Node1& Node 2 where our .pem key is stored and paste the Example command (starting with ssh -i ..... ) in the terminal.( ssh -i "Master\_Ec2\_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com)

Master:

4. From now on, until mentioned, perform these steps on all 3 machines.

Install Docker

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee
```

```
/etc/apt/trusted.gpg.d/docker.gpg > /dev/null
```

```
sudo add-apt-repository "deb [arch=amd64]
```

```
https://download.docker.com/linux/ubuntu$(lsb_release -cs) stable"
```

```
[ec2-user@ip-172-31-38-147 ~]$ sudo su
[root@ip-172-31-38-147 ec2-user]# yum install docker -y
Last metadata expiration check: 0:20:47 ago on Mon Aug 26 08:47:32 2024.
Dependencies resolved.
=====
|| Package           || Architecture || Version      || Repository || Size ||
| Installing:      ||             ||             ||           ||       |
| docker            || x86_64      || 25.0.6-1.amzn2023.0.1 || amazonlinux || 44 M  |
| Installing dependencies: ||             ||             ||           ||       |
| containerd        || x86_64      || 1.7.20-1.amzn2023.0.1 || amazonlinux || 35 M  |
| iptables-libs     || x86_64      || 1.8.8-3.amzn2023.0.2 || amazonlinux || 401 k |
| iptables-nft      || x86_64      || 1.8.8-3.amzn2023.0.2 || amazonlinux || 183 k |
| libcgroup         || x86_64      || 3.0-1.amzn2023.0.1  || amazonlinux || 75 k  |
| libnetfilter_conntrack || x86_64      || 1.0.8-2.amzn2023.0.2 || amazonlinux || 58 k  |
| libnfnetwork      || x86_64      || 1.0.1-19.amzn2023.0.2 || amazonlinux || 30 k  |
| libnftnl          || x86_64      || 1.2.2-2.amzn2023.0.2 || amazonlinux || 84 k  |
| pigz              || x86_64      || 2.5-1.amzn2023.0.3   || amazonlinux || 83 k  |
| runc              || x86_64      || 1.1.11-1.amzn2023.0.1 || amazonlinux || 3.0 M |
=====
i-0532cdd454d39c9ab (node-2)
Public IPs: 13.60.105.92 Private IPs: 172.31.38.147
```

```
sudo apt-get update
```

```
sudo apt-get install -y docker-ce
```

Then, configure cgroup in a daemon.json file.

```
sudo mkdir -p /etc/docker
```

```
cat <<EOF | sudo tee /etc/docker/daemon.json
```

```
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
```

```

"exec-opt": ["native.cgroupdriver=systemd"]
}

```

```

sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

```

Install Kubernetes on all 3 machines

```

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor
-o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

```

```
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
```

```
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ | sudo tee /etc/apt/sources.list.d/kubernetes.list
```

Package	Architecture	Version	Repository	Size
<hr/>				
Installing:				
kubeadm	x86_64	1.31.1-150500.1.1	kubernetes	11 M
kubectl	x86_64	1.31.1-150500.1.1	kubernetes	11 M
kubelet	x86_64	1.31.1-150500.1.1	kubernetes	15 M
<hr/>				
Installing dependencies:				
conctrack-tools	x86_64	1.4.6-2.amzn2023.0.2	amazonlinux	208 k
cri-tools	x86_64	1.31.1-150500.1.1	kubernetes	6.9 M
kubernetes-cni	x86_64	1.5.1-150500.1.1	kubernetes	7.1 M
libnetfilter_cthelper	x86_64	1.0.0-21.amzn2023.0.2	amazonlinux	24 k
libnetfilter_cttimeout	x86_64	1.0.0-19.amzn2023.0.2	amazonlinux	24 k
libnetfilter_queue	x86_64	1.0.5-2.amzn2023.0.2	amazonlinux	30 k
<hr/>				
Transaction Summary				
<hr/>				

```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

```

```

sudo systemctl enable --now kubelet
sudo apt-get install -y containerd

```

```
ubuntu@ip-172-31-27-176:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 133 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
```

```
sudo mkdir -p /etc/containerd
```

```
sudo containerd config default | sudo tee /etc/containerd/config.toml
```

```
ubuntu@ip-172-31-27-176:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
path = ""

[debug]
address = ""
format = ""
gid = 0
level = ""
uid = 0

[grpc]
address = "/run/containerd/containerd.sock"
gid = 0
```

```
sudo systemctl restart containerd
```

```
sudo systemctl enable containerd
```

```
sudo systemctl status containerd
```

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; p>
   Active: active (running) since Mon 2024-09-16 15:31:58 UTC; 210ms ago
     Docs: https://containerd.io
 Main PID: 4763 (containerd)
    Tasks: 7
   Memory: 13.9M (peak: 14.4M)
     CPU: 50ms
    CGroup: /system.slice/containerd.service
            └─4763 /usr/bin/containerd

Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 systemd[1]: Started containerd.service - c>
```

```
sudo apt-get install -y socat
```

```
ubuntu@ip-172-31-27-176:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 133 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat
  amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (11.2 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

## 5. Perform this **ONLY on the Master machine**

Initialize the Kubecluster

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
--ignore-preflight-errors=all
```

Copy the join command and keep it in a notepad, we'll need it later.

```
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:
kubeadm join 172.31.40.173:6443 --token a84ptk.jhdjs1nesolmhfuf \
```

Copy the mkdir and chown commands from the top and execute them

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf
```

Check the created pod using this command

Now, keep a watch on all nodes using the following command

```
watch kubectl get nodes
```

```
[root@ip-172-31-40-173 ec2-user]# kubectl get nodes
NAME                  STATUS    ROLES     AGE      VERSION
ip-172-31-40-173.eu-north-1.compute.internal  NotReady  control-plane  6m55s  v1.31.1
[root@ip-172-31-40-173 ec2-user]# 
```

## 6. Perform this **ONLY on the worker machines**

```
sudo kubeadm join 172.31.27.176:6443 --token ttay2x.n0squeukjai8sgfg3 \
--discovery-token-ca-cert-hash
sha256:d6fc5fb7e984c83e2807780047fec6c4f2acfe9da9184ecc028d77157608fbb6
```

```
[preflight] Running pre-flight checks
[WARNING FileExisting-tc]: tc not found in system path
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

[root@ip-172-31-89-46 ec2-user]# █
```

Now, notice the changes on the master terminal

```
[root@ip-172-31-85-89 ec2-user]# kubectl get nodes
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-89.ec2.internal  NotReady  control-plane  72s  v1.26.0
[root@ip-172-31-85-89 ec2-user]# kubectl get nodes
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-85-89.ec2.internal  NotReady  control-plane  105s  v1.26.0
ip-172-31-89-46.ec2.internal  NotReady  <none>     5s   v1.26.0
[root@ip-172-31-85-89 ec2-user]# kubectl get nodes
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-85-89.ec2.internal  NotReady  control-plane  119s  v1.26.0
ip-172-31-89-46.ec2.internal  NotReady  <none>     19s  v1.26.0
ip-172-31-94-70.ec2.internal  NotReady  <none>     12s  v1.26.0
[root@ip-172-31-85-89 ec2-user]# █
```

7. Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

kubectl apply -f <https://docs.projectcalico.org/manifests/calico.yaml>

```
ubuntu@ip-172-31-27-176:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

Now Run command kubectl get nodes -o wide we can see Status is ready.

NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
ip-172-31-18-135	Ready	<none>	6m19s	v1.31.1	172.31.18.135	<none>	Ubuntu 24.04 LTS	6.0.0-1012-aws	containerd://1.7.12
ip-172-31-27-176	Ready	control-plane	15m	v1.31.1	172.31.27.176	<none>	Ubuntu 24.04 LTS	6.0.0-1012-aws	containerd://1.7.12
ip-172-31-28-117	Ready	<none>	7m09s	v1.31.1	172.31.28.117	<none>	Ubuntu 24.04 LTS	6.0.0-1012-aws	containerd://1.7.12

That's it, we now have a Kubernetes cluster running across 3 AWS EC2 Instances. This cluster can be used to further deploy applications and their loads being distributed across these machines.

**Conclusion:** Thus we have understood the Kubernetes cluster architecture and have successfully created Kubernetes cluster on a linux machine with the help of AWS by creating three EC2 instances one master and two nodes, installed docker and kubernetes on all three machines and then initialized a kubernetes control-plane node on the master and then added the worker nodes to the cluster to distribute the workload.

# Advanced DevOps Lab

## Experiment 4

**Aim:** To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

### **Theory:**

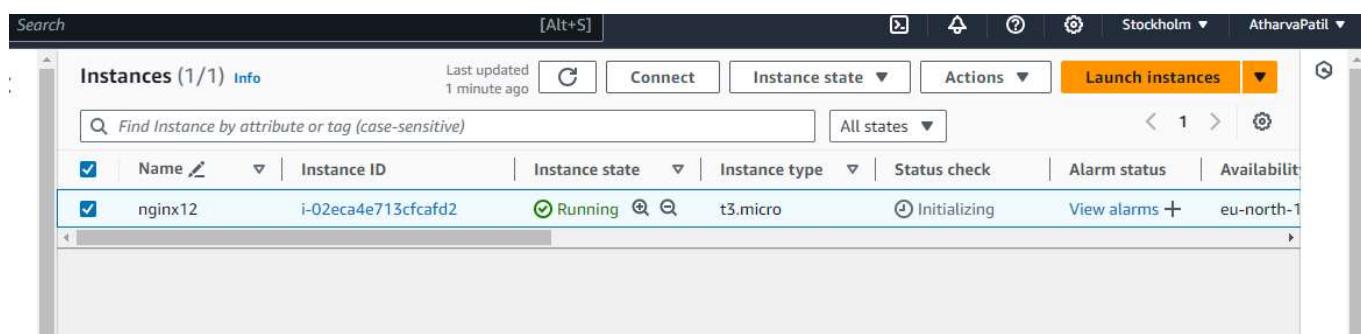
Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. In fact, Kubernetes has established itself as the defacto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), backed by key players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

### Kubernetes Deployment

A Kubernetes Deployment is used to tell Kubernetes how to create or modify instances of the pods that hold a containerized application. Deployments can scale the number of replica pods, enable the rollout of updated code in a controlled manner, or roll back to an earlier deployment version if necessary.

### **Steps:**

1. Create an EC2 Ubuntu Instance on AWS.



## 2. Edit the Security Group Inbound Rules to allow SSH

The screenshot shows the 'Edit inbound rules' section of the AWS CloudFormation console. It displays a table of security group rules. One rule is listed:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-089475d0793f4644f	SSH	TCP	22	Cust... ▾	0.0.0.0/0 X

Below the table is a button labeled 'Add rule'. At the bottom of the screen, there is a yellow warning message: '⚠️ 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.' with a close button 'X'.

## 3. SSH into the machine

```
ssh -i <keyname>.pem ubuntu@<public_ip_address>
```

## 4. Install Docker

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt-get update
sudo apt-get install -y docker-ce

Setting up libltdl7:amd64 (2.4.7-7build1) ...
Setting up docker-ce-cli (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-39-158:~$
```

Then, configure cgroup in a daemon.json file.

```
sudo mkdir -p /etc/docker
cd /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
```

```
ubuntu@ip-172-31-38-181:~$ cd /etc/docker
ubuntu@ip-172-31-38-181:/etc/docker$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-38-181:/etc/docker$ █
```

---

## 5. Install Kubernetes

```
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

```
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
```

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo
apt-key add -
cat << EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
ec2-user@ip-172-31-24-190 ~ $ kubectl version
Client Version: v1.31.1
Kustomize Version: v5.4.2
```

After installing Kubernetes, we need to configure internet options to allow bridging.

```
sudo swapoff -a
echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a
/etc/sysctl.conf
sudo sysctl -p
```

## 6. Initialize the Kubecluster

`sudo apt-get install -y containerd`

```
To see the stack trace of this error execute with --v=5 or higher    ubuntu@ip-172-31-20-171:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp8 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 130 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (74.5 MB/s)
(Reading database ... 68068 files and directories currently installed.)
Removing docker-ce (5:27.2.1-1~ubuntu.24.04-noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68048 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

`sudo mkdir -p /etc/containerd`

```
sudo containerd config default | sudo tee /etc/containerd/config.toml
sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
```

```
sudo apt-get install -y socat
```

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

```
Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.24.190:6443 --token xsbsql.6rollsaavnvtbsvu \
  --discovery-token-ca-cert-hash sha256:10d2b67f4f4749b51854065a554c74e6a956e4782d9ab4bb79b8591640b3edef
ec2-user@ip-172-31-24-190 ~ $ kubectl get nodes
```

Copy the mkdir and chown commands from the top and execute them

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Then, add a common networking plugin called flannel as mentioned in the code.

```
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/
kube-flannel.yml
```

7. Now that the cluster is up and running, we can deploy our nginx server on this cluster.

Apply this deployment file using this command to create a deployment

```
kubectl apply -f https://k8s.io/examples/application/deployment.yaml
```

```
ec2-user@ip-172-31-24-190 ~ $ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

Use ‘kubectl get pods’ to verify if the deployment was properly created and the pod is working correctly.

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-mwd8p	0/1	Pending	0	7s
nginx-deployment-d556bf558-zc25s	0/1	Pending	0	7s

Next up, create a name alias for this pod.

```
POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name")
```

8. Lastly, port forward the deployment to your localhost so that you can view it.

```
kubectl port-forward $POD_NAME 8080:80
```

Kubectl taint nodes

```
--allnode-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted
```

kubectl get nodes

kubectl get pods

```
POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath=".items[0].metadata.name") kubectl port-forward $POD_NAME 8080:80
```

9. Verify your deployment

Open up a new terminal and ssh to your EC2 instance.

Then, use this curl command to check if the Nginx server is running.

```
curl --head http://127.0.0.1:8080
ec2-user@ip-172-31-24-190 ~ $ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sat, 14 Sep 2024 06:54:21 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

If the response is 200 OK and you can see the Nginx server name, your deployment was successful.

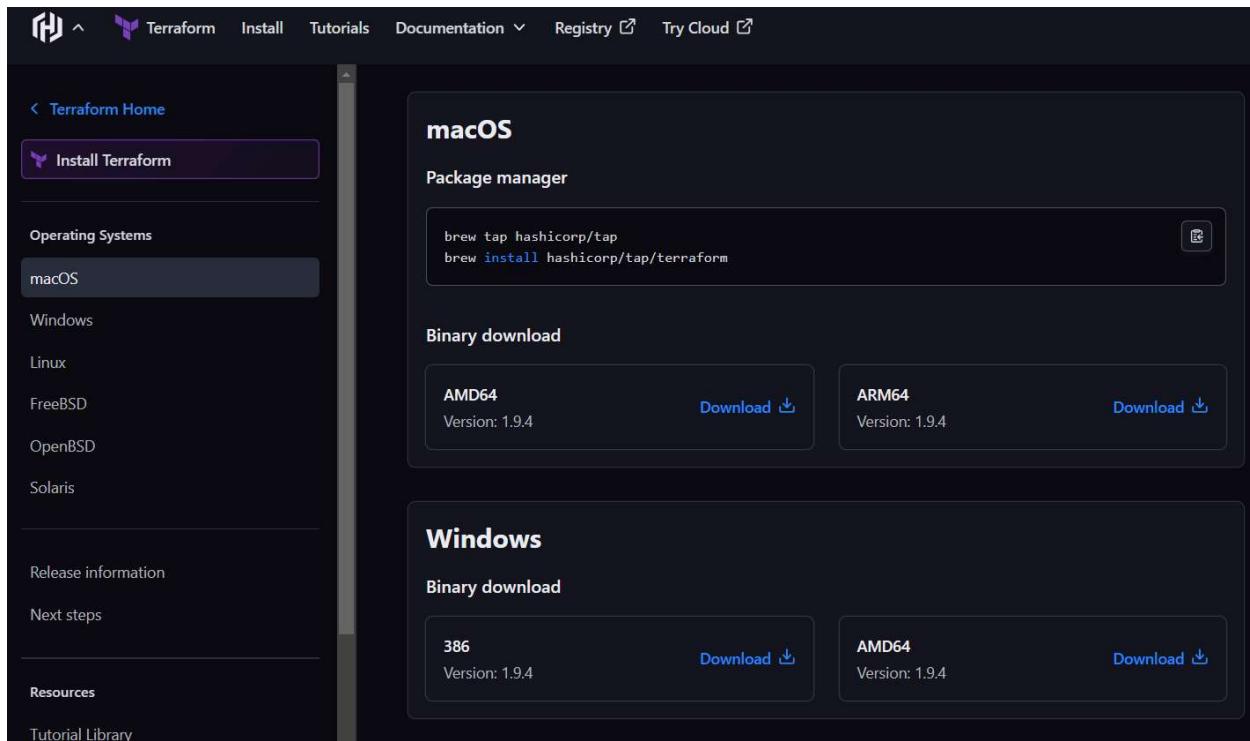
We have successfully deployed our Nginx server on our EC2 instance.

**Conclusion:** After installing docker and kubernetes on the EC2 Ubuntu instance we initialized the Kube cluster. Nginx is a web server software that is used for its low resource usage and high performance. We can use this as a server to host our server and deploy it on the cluster. After deploying we check if the pod is working correctly and lastly port forward the deployment to our local host. Then we check if the server is running, if the server responds with 200, ok then the deployment was successful.

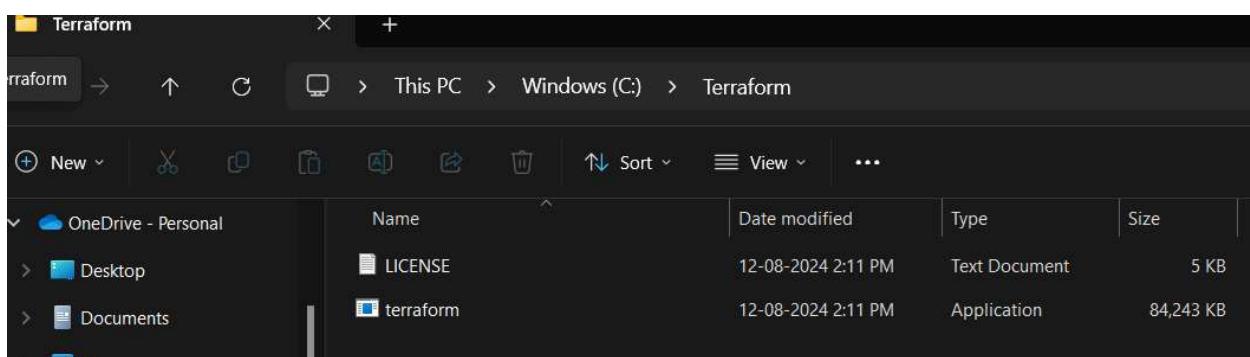
## Experiment 5

1) Downloading Terraform from official website

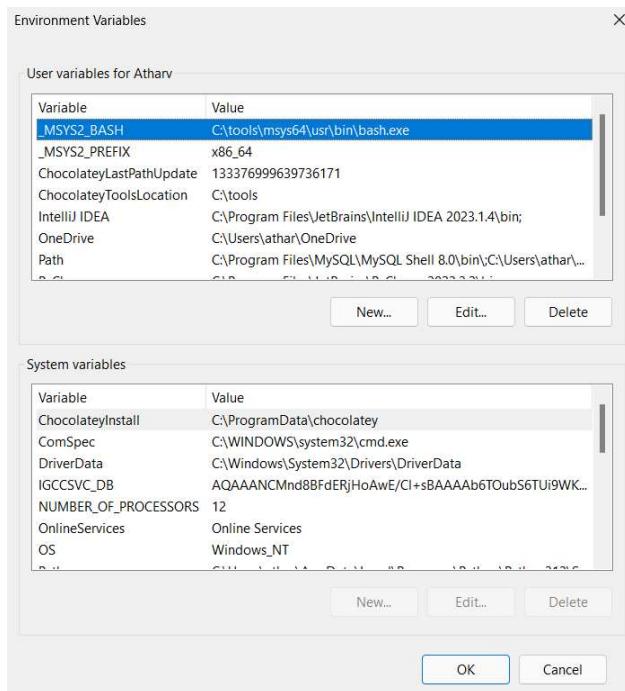
<https://developer.hashicorp.com/terraform/install>



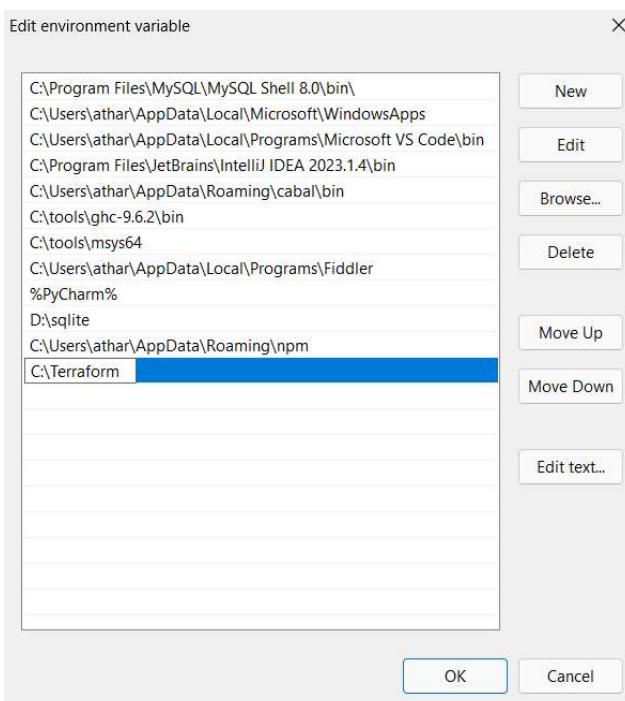
2) Save it into Windows C



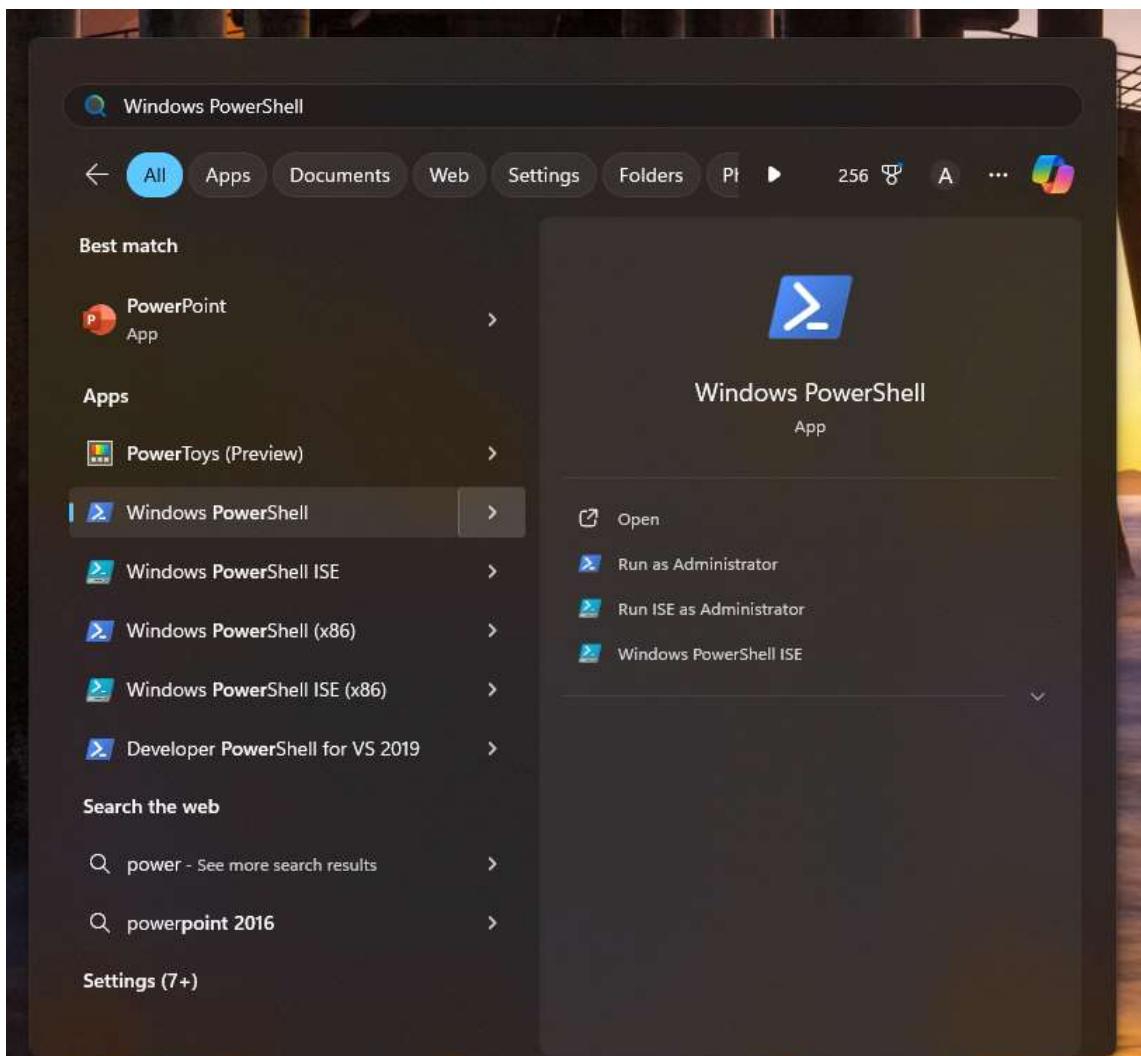
### 3)Change Environment Variables



4) Click on path and add new C:\Terraform



5)search for powershell and open it



5)To Check if terraform is installed open powershell and type Terraform —version

```
PS C:\Users\athar> Terraform --version
Terraform v1.9.4
on windows_386
PS C:\Users\athar>
```

6)To see all terraform commands type Terraform –help

```
PS C:\Users\athar> Terraform --help
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init      Prepare your working directory for other commands
  validate   Check whether the configuration is valid
  plan       Show changes required by the current configuration
  apply      Create or update infrastructure
  destroy    Destroy previously-created infrastructure

All other commands:
  console    Try Terraform expressions at an interactive command prompt
  fmt        Reformat your configuration in the standard style
  force-unlock Release a stuck lock on the current workspace
  get        Install or upgrade remote Terraform modules
  graph      Generate a Graphviz graph of the steps in an operation
  import     Associate existing infrastructure with a Terraform resource
  login      Obtain and save credentials for a remote host
  logout     Remove locally-stored credentials for a remote host
  metadata   Metadata related commands
  output     Show output values from your root module
  providers  Show the providers required for this configuration
  refresh    Update the state to match remote systems
  show       Show the current state or a saved plan
  state      Advanced state management
  taint      Mark a resource instance as not fully functional
  test       Execute integration tests for Terraform modules
  untaint   Remove the 'tainted' state from a resource instance
  version    Show the current Terraform version
  workspace  Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR  Switch to a different working directory before executing the
              given subcommand.
  -help       Show this help output, or the help for a specified subcommand.
  -version    An alias for the "version" subcommand.
PS C:\Users\athar> |
```

**Aim :** Exp 6 To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform.(S3 bucket or Docker)

### Creating the docker image using terraform

1: Check the docker version and functionality if its not downloaded you can download it from <https://www.docker.com/>

```
C:\Users\Avan\Desktop>docker --version
Docker version 27.0.3, build 7d4bcd8
```

```
C:\Users\Avan\Desktop>docker

Usage: docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run      Create and run a new container from an image
  exec    Execute a command in a running container
  ps       List containers
  build   Build an image from a Dockerfile
  pull    Download an image from a registry
  push    Upload an image to a registry
  images  List images
  login   Log in to a registry
  logout  Log out from a registry
  search  Search Docker Hub for images
  version Show the Docker version information
  info    Display system-wide information
```

(Now, create a folder named ‘Terraform Scripts’ in which we save our different types of scripts which will be further used in this experiment)

2: Firstly create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor (or you can use vscode) and write the following contents into it to create a Ubuntu Linux container.

```
terraform {
  required_providers {
    docker = {
```

```
source = "kreuzwerker/docker"
version = "2.21.0"
}

}
}

provider "docker" {
  host = "npipe://./pipe/docker_engine"
}

# Pull the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name = "foo"
  command = ["sleep", "3600"]
}
```

```
docker.tf  X
Docker > docker.tf
1  terraform {
2    required_providers {
3      docker = {
4        source  = "kreuzwerker/docker"
5        version = "2.21.0"
6      }
7    }
8  }
9
10 provider "docker" {
11   host = "npipe://./pipe/docker_engine"
12 }
13
14 # Pull the image
15 resource "docker_image" "ubuntu" {
16   name = "ubuntu:latest"
17 }
18
19 # Create a container
20 resource "docker_container" "foo" {
21   image = docker_image.ubuntu.image_id
22   name = "foo"
23   command = ["sleep", "3600"]
24 }
```

3: Execute **Terraform Init** command to initialize the resources (*Make sure you are in the Docker directory before executing the command*)

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.21.0"...
- Installing kreuzwerker/docker v2.21.0...
- Installed kreuzwerker/docker v2.21.0 (self-signed, key ID BD080C4571C6104C)
  Partner and community providers are signed by their developers.
  If you'd like to know more about provider signing, you can read about it here:
  https://www.terraform.io/docs/cli/plugins/signing.html
  Terraform has created a lock file .terraform.lock.hcl to record the provider
  selections it made above. Include this file in your version control repository
  so that Terraform can guarantee to make the same selections by default when
  you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

#### 4. Execute **Terraform plan** to see the available resources

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform plan
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# docker_container.foo will be created
+ resource "docker_container" "foo" {
    + attach          = false
    + bridge          = (known after apply)
    + command         = [
        + "sleep",
        + "3600",
    ]
    + container_logs = (known after apply)
    + entrypoint     = (known after apply)
    + env             = (known after apply)
    + exit_code       = (known after apply)
    + gateway         = (known after apply)
    + hostname        = (known after apply)
    + id              = (known after apply)
    + image           = (known after apply)
    + init             = (known after apply)
    + ip_address      = (known after apply)
    + ip_prefix_length = (known after apply)
    + ipc_mode        = (known after apply)
    + log_driver       = (known after apply)
    + logs            = false
    + must_run        = true
    + name            = "foo"
    + network_data    = (known after apply)
    + read_only        = false
    + remove_volumes  = true
    + restart          = "no"
    + rm               = false
    + runtime          = (known after apply)
    + security_opts   = (known after apply)
    + shm_size         = (known after apply)
    + start            = true
    + stdin_open       = false
    + stop_signal      = (known after apply)
    + stop_timeout     = (known after apply)
    + tty              = false
}
```

```
# docker_image.ubuntu will be created
+ resource "docker_image" "ubuntu" {
    + id          = (known after apply)
    + image_id    = (known after apply)
    + latest      = (known after apply)
    + name        = "ubuntu:latest"
    + output      = (known after apply)
    + repo_digest = (known after apply)
}

Plan: 2 to add, 0 to change, 0 to destroy.
```

5. Execute **Terraform apply** to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration. Using command : “**terraform apply**”

```
C:\Users\Avan\Desktop\TerraformScripts\Docke>terraform apply
docker_image.ubuntu: Refreshing state... [id=sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=23364bd2196c264de0ff6af7150449fea25e1a9f7553ac6955b95679547526e]

Note: Objects have changed outside of Terraform

Terraform detected the following changes made outside of Terraform since the last "terraform apply" which may have affected this plan:

# docker_image.ubuntu has been deleted
- resource "docker_image" "ubuntu" {
    - id          = "sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest"
    - image_id    = "sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
    - name        = "ubuntu:latest"
    # (2 unchanged attributes hidden)
}

Unless you have made equivalent changes to your configuration, or ignored the relevant attributes using ignore_changes, the following plan may include actions to undo or respond to these changes.

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# docker_container.foo will be created
+ resource "docker_container" "foo" {
    + attach          = false
    + bridge          = (known after apply)
    + command         = [
        + "sleep",
        + "3600",
    ]
    + container_logs  = (known after apply)
    + entrypoint      = (known after apply)
    + env             = (known after apply)
    + exit_code       = (known after apply)
    + gateway         = (known after apply)
```

```

+ shm_size      = (known after apply)
+ start        = true
+ stdin_open   = false
+ stop_signal   = (known after apply)
+ stop_timeout  = (known after apply)
+ tty           = false

+ healthcheck (known after apply)

+ labels (known after apply)
}

# docker_image.ubuntu will be created
resource "docker_image" "ubuntu" {
  + id          = (known after apply)
  + image_id    = (known after apply)
  + latest      = (known after apply)
  + name         = "ubuntu:latest"
  + output       = (known after apply)
  + repo_digest = (known after apply)
}

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Creation complete after 14s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

```

## 6. Docker images before executing this command

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
```

Docker images after the execution of command

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
ubuntu          latest    edbfe74c41f8  3 weeks ago  78.1MB
```

## 7. Executing the command **terraform providers** a provider is a plugin that allows terraform to interact with APIs of external services, enabling to manage the resources offered by those services.

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform providers

Providers required by configuration:
└── provider[registry.terraform.io/kreuzwerker/docker] 2.21.0

Providers required by state:

  provider[registry.terraform.io/kreuzwerker/docker]
```

## 8. Running the command **terraform validate** to check if the configuration is valid

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform validate
Success! The configuration is valid.
```

9. The **terraform state list** is used to display a list of resources managed by Terraform within the current state file

```
C:\Users\Student\Desktop\TerraformsScripts\Docker>terraform state list
docker_container.foo
docker_image.ubuntu
```

10. Execute **Terraform destroy** to delete the configuration, which will automatically delete the Ubuntu Container.

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
- destroy

Terraform will perform the following actions:

# docker_container.foo will be destroyed
resource "docker_container" "foo" {
  - attach           = false -> null
  - command          = [
    - "sleep",
    - "3600",
  ] -> null
  - cpu_shares       = 0 -> null
  - dns               = [] -> null
  - dns_opts          = [] -> null
  - dns_search         = [] -> null
  - entrypoint        = [] -> null
  - env               = [] -> null
  - gateway           = "172.17.0.1" -> null
  - group_add         = [] -> null
  - hostname          = "7e9ee45651ce" -> null
  - id                = "7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c" -> null
  - image              = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - init               = false -> null
  - ip_address         = "172.17.0.2" -> null
  - ip_prefix_length   = 16 -> null
  - ipc_mode           = "private" -> null
  - links              = [] -> null
  - log_driver          = "json-file" -> null
  - log_opts             = {} -> null
  - logs               = false -> null
  - max_retry_count     = 0 -> null
  - memory              = 0 -> null
  - memory_swap          = 0 -> null
  - must_run            = true -> null
  - name               = "foo" -> null}
```

```

- stop_timeout      = 0    -> null
- storage_opts     = {}   -> null
- sysctls          = {}   -> null
- tmpfs            = {}   -> null
- tty               = false -> null
  # (8 unchanged attributes hidden)
}

# docker_image.ubuntu will be destroyed
resource "docker_image" "ubuntu" {
  - id      = "sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
  - image_id = "sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - latest     = "sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - name       = "ubuntu:latest" -> null
  - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616888f04e34a9ab63ee" -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

docker_container.foo: Destroying... [id=7e9ee45651ce4057af54ab9bb95ef29084eb679a0f4114e88a5e0887c6480b8c]
docker_container.foo: Destruction complete after 0s
docker_image.ubuntu: Destroying... [id=sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.

```

## 11. Docker images after the destroy command execution

```
C:\Users\Avan\Desktop\TerraformScripts\Docker>docker images
REPOSITORY      TAG          IMAGE ID      CREATED      SIZE
```

Installation Links :Git : <https://git-scm.com/download/win>

Jenkins : <https://www.jenkins.io/download/thank-you-downloading-windows-installer-stable>

Docker : <https://www.docker.com/products/docker-desktop/>

**Aim:** To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

**Theory:** Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

### What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

### Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical

vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence. Thus, integrating static analysis into the SDLC can yield dramatic results in the overall quality of the code developed.

## **What are the key steps to run SAST effectively?**

There are six simple steps needed to perform SAST efficiently in organizations that have a very large number of applications built with different languages, frameworks, and platforms.

- 1. Finalize the tool.** Select a static analysis tool that can perform code reviews of applications written in the programming languages you use. The tool should also be able to comprehend the underlying framework used by your software.
- 2. Create the scanning infrastructure, and deploy the tool.** This step involves handling the licensing requirements, setting up access control and authorization, and procuring the resources required (e.g., servers and databases) to deploy the tool.
- 3. Customize the tool.** Fine-tune the tool to suit the needs of the organization. For example, you might configure it to reduce false positives or find additional security vulnerabilities by writing new rules or updating existing ones. Integrate the tool into the build environment, create dashboards for tracking scan results, and build custom reports.
- 4. Prioritize and onboard applications.** Once the tool is ready, onboard your applications. If you have a large number of applications, prioritize the high-risk applications to scan first. Eventually, all your applications should be onboarded and scanned regularly, with application scans synced with release cycles, daily or monthly builds, or code check-ins.
- 5. Analyze scan results.** This step involves triaging the results of the scan to remove false positives. Once the set of issues is finalized, they should be tracked and provided to the deployment teams for proper and timely remediation.
- 6. Provide governance and training.** Proper governance ensures that your development teams are employing the scanning tools properly. The software security touchpoints should be present within the SDLC. SAST should be incorporated as part of your application development and deployment process.

## **Integrating Jenkins with SonarQube:**

Windows installation

Step 1 Install JDK 1.8

Step 2 download and install jenkins

<https://www.blazemeter.com/blog/how-to-install-jenkins-on-windows>

### Ubuntu installation

<https://www.digitalocean.com/community/tutorials/how-to-install-java-with-a-pt-on-ubuntu-20-04#installing-the-default-jre-jdk>

Step 1 Install JDK 1.8

```
sudo apt-get install openjdk-8-jre
```

```
sudo apt install default-jre
```

<https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-20-04>

[Open SSH](#)

### Prerequisites:

- [Jenkins installed](#)
- [Docker Installed](#) (for SonarQube)
- SonarQube Docker Image

### Steps to integrate Jenkins with SonarQube

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you

The screenshot shows the Jenkins dashboard at [localhost:8060](http://localhost:8060). The main area displays a table of active builds:

S	W	Name	Last Success	Last Failure	Last Duration
		Jenkins_Pipeline	1 mo 0 days #4	N/A	3.5 sec
		maven_build	2 days 22 hr #1	N/A	2 min 9 sec
		Tomact	29 days #4	29 days #8	2.9 sec

On the left sidebar, there are links for 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', and 'My Views'. Below these are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (Built-In Node).

2. Run SonarQube in a Docker container using this command -  
 docker -v  
 docker pull sonarqube

TRY THE NEW CROSS-PLATFORM POWERSHELL <https://aka.ms/powershell>

```
PS C:\Users\sushmita> docker -v
Docker version 27.1.1, build 6312585
PS C:\Users\sushmita> docker pull sonarqube
Using default tag: latest
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest
```

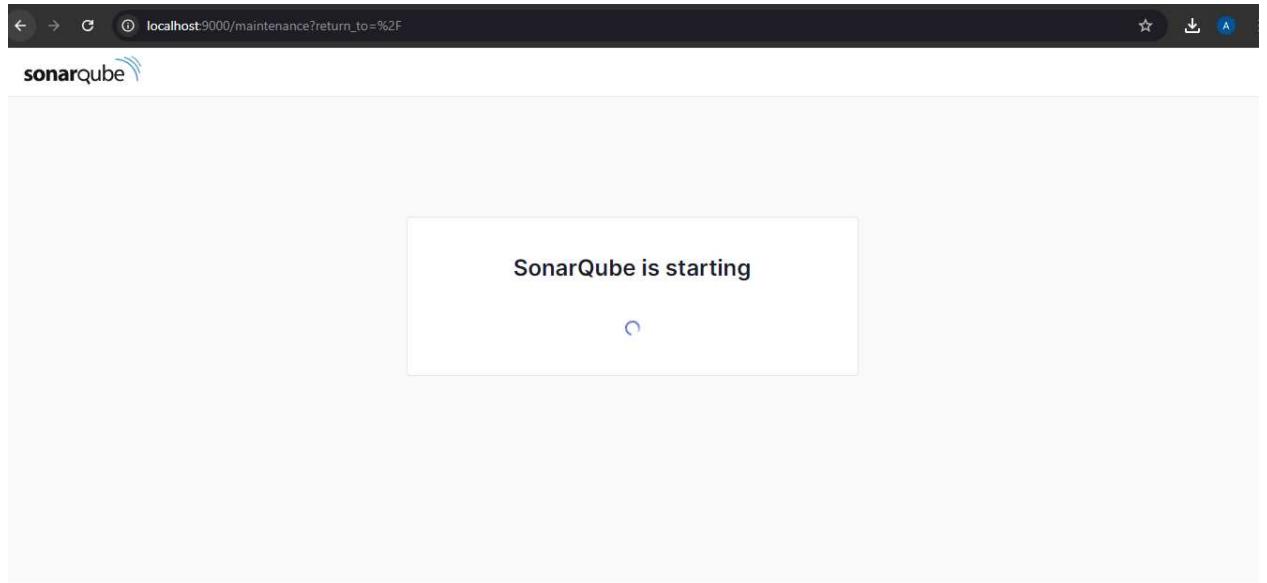
**Warning: run below command only once**

```
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
```

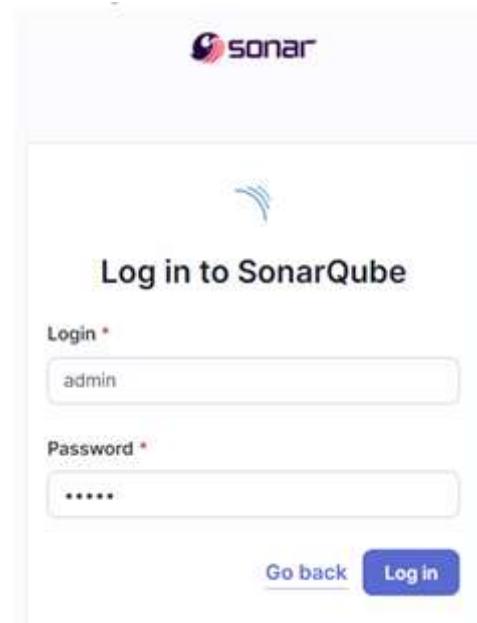
```
PS C:\Users\sushmita> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
ee4dad9f96364a57945728ce2499989ee0787873adacfcd36d9aaa39258c93f3
PS C:\Users\sushmita>
```

RAM 0.96 GB CPU 11.11% Disk -- GB avail. of -- GB BETA Terminal New version

- Once the container is up and running, you can check the status of SonarQube at localhost port 9000



- Login to SonarQube using username *admin* and password *admin*.



**5. Create a manual project in SonarQube with the name **sonarqube****

1 of 2

## Create a local project

Project display name \*

sonarqube



Project key \*

sonarqube



Main branch name \*

main

The name of your project's default branch [Learn More](#)[Cancel](#)[Next](#)

2 of 2

## Set up project for Clean as You Code

[X](#)

The new code definition sets which part of your code will be considered new code. This helps you focus attention on the most recent changes to your project, enabling you to follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

 Use the global setting[Previous version](#)

Any code that has changed since the previous version is considered new code.  
Recommended for projects following regular versions or releases.

 Define a specific setting for this project Previous version

Any code that has changed since the previous version is considered new code.  
Recommended for projects following regular versions or releases.

 Number of days

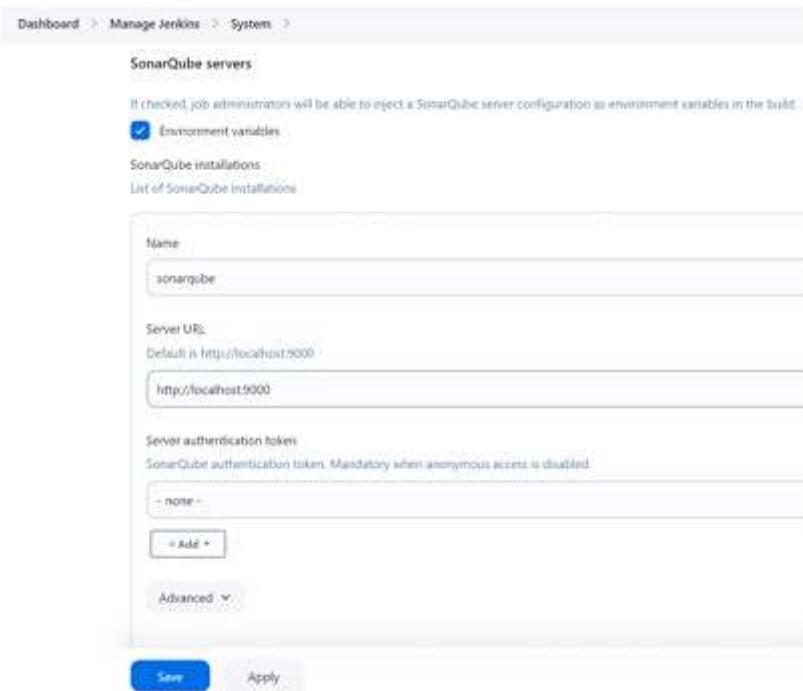
Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install



6. Under Manage Jenkins, Systems look for SonarQube Servers and enter the details.

Enter the Server Authentication token if needed.



7. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.



8. After the configuration, create a New Item in Jenkins, choose a freestyle project.

### New Item

Enter an item name

sonarqube

Select an item type



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

OK

9. Choose this GitHub repository in Source Code Management.

[https://github.com/shazforiot/MSBuild\\_firstproject.git](https://github.com/shazforiot/MSBuild_firstproject.git)

It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.

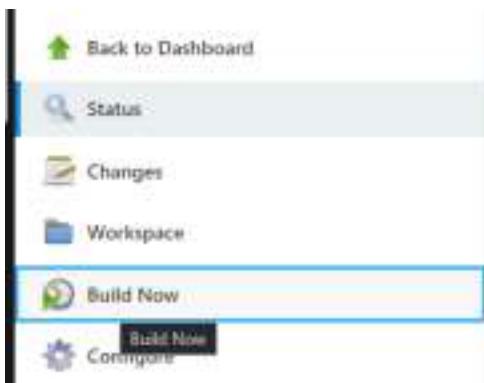


10. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL. 11. Go to [http://localhost:9000/<user\\_name>/permissions](http://localhost:9000/<user_name>/permissions) and allow Execute Permissions to the Admin user.

```
sonar.projectKey=sonarqube  
sonar.login=admin  
sonar.password= Star100%  
sonar.sources=.  
sonar.host.url=http://localhost:9000
```

The screenshot shows a configuration form for 'Execute SonarQube Scanner'. It includes fields for 'Path to project properties' (empty) and 'Analysis properties' (containing the provided configuration). Other sections include 'Additional arguments' (empty) and 'JVM Options' (empty).

## 12. Run The Build.



Check the console output.

**Console Output**   View as plain text

```

Started by user Atharva Prabhakar Patil
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/shazforiot/MSBuild_firstproject.git
> git init C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
> git --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
> git fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)

01:48:52.594 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
01:48:52.596 INFO Sensor C# File Caching Sensor [csharp] (done) | time=9ms
01:48:52.597 INFO Sensor Zero Coverage Sensor
01:48:52.641 INFO Sensor Zero Coverage Sensor (done) | time=44ms
01:48:52.653 INFO SCM Publisher SCM provider for this project is: git
01:48:52.656 INFO SCM Publisher 4 source files to be analyzed
01:48:55.556 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=2899ms
01:48:55.560 INFO CPD Executor Calculating CPD for 0 files
01:48:55.565 INFO CPD Executor CPD calculation finished (done) | time=0ms
01:48:55.607 INFO SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
01:48:56.427 INFO Analysis report generated in 606ms, dir size=201.0 kB
01:48:56.710 INFO Analysis report compressed in 281ms, zip size=22.2 kB
01:48:57.922 INFO Analysis report uploaded in 1208ms
01:48:57.925 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
01:48:57.927 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
01:48:57.929 INFO More about the report processing at http://localhost:9000/api/ce/task?id=8b35a5b7-a50e-424b-a77c-6aa39d925923
01:48:57.966 INFO Analysis total time: 1:09.987 s
01:48:57.995 INFO SonarScanner Engine completed successfully
01:48:58.203 INFO EXECUTION SUCCESS
01:48:58.205 INFO Total time: 3:30.482s
Finished: SUCCESS

```

13. Once the build is complete, check the project in SonarQube.

The screenshot shows the SonarQube web interface. At the top, there are tabs for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. Below the header, there's a 'Create Project' button and a 'My Favorites' dropdown. The main area displays the project 'sonarqube' (PUBLIC). It shows a 'Quality Gate' section with a green checkmark for 'Passed' (1 issue) and a red cross for 'Failed' (0 issues). Under 'Reliability', there are five categories (A, B, C, D, E) each with 0 issues. A message states 'The main branch of this project is empty.' At the bottom right, it says '1 of 1 shown'.

This screenshot shows the 'Overview' page for the 'main' branch of the 'sonarqube' project. The top navigation bar includes links for Overview, Issues, Security Hotspots, Measures, Code, Activity, Project Settings, and Project Information. The main content area features a large green 'Passed' badge with a checkmark. Below it, a warning message says 'The last analysis has warnings. See details'. The page is divided into three main sections: Security, Reliability, and Maintainability, each showing 0 open issues. At the bottom, there are sections for Accepted issues, Coverage, and Duplications.

In this way, we have integrated Jenkins with SonarQube for SAST.

## Conclusion

In this experiment, we have understood the importance of SAST and have successfully integrated Jenkins with SonarQube for Static Analysis and Code Testing.

### **Additional resources**

Sonarqube installation on aws Ubuntu

<https://www.coachdevops.com/2020/04/install-snarqube-on-ubuntu-how-to.html>

<https://awstip.com/installing-snarqube-on-aws-ec2-instance-and-integrating-it-with-aws-code-pipeline-abec99416ba4>

docker-compose up -d && docker ps

## Expt No. 08 Advanced DevOps Lab

**Aim:** Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

### Theory:

#### What is SAST?

Static application security testing (SAST), or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's applications susceptible to attack. SAST scans an application before the code is compiled. It's also known as white box testing.

#### What problems does SAST solve?

SAST takes place very early in the software development life cycle (SDLC) as it does not require a working application and can take place without code being executed. It helps developers identify vulnerabilities in the initial stages of development and quickly resolve issues without breaking builds or passing on vulnerabilities to the final release of the application.

SAST tools give developers real-time feedback as they code, helping them fix issues before they pass the code to the next phase of the SDLC. This prevents security-related issues from being considered an afterthought. SAST tools also provide graphical representations of the issues found, from source to sink. These help you navigate the code easier. Some tools point out the exact location of vulnerabilities and highlight the risky code. Tools can also provide in-depth guidance on how to fix issues and the best place in the code to fix them, without requiring deep security domain expertise.

It's important to note that SAST tools must be run on the application on a regular basis, such as during daily/monthly builds, every time code is checked in, or during a code release.

#### Why is SAST important?

Developers dramatically outnumber security staff. It can be challenging for an organization to find the resources to perform code reviews on even a fraction of its applications. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally, they are much faster than manual secure code reviews performed by humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities—such as buffer overflows, SQL injection, cross-site scripting, and others—with high confidence.

#### What is a CI/CD Pipeline?

CI/CD pipeline refers to the Continuous Integration/Continuous Delivery pipeline. Before we dive deep into this segment, let's first understand what is meant by the term 'pipeline'?

A pipeline is a concept that introduces a series of events or tasks that are connected in a sequence to make quick software releases. For example, there is a task, that task has got five different stages, and each stage has got some steps. All the steps in phase one have to be completed, to mark the latter stage to be complete.



Now, consider the CI/CD pipeline as the backbone of the DevOps approach. This Pipeline is responsible for building codes, running tests, and deploying new software versions. The Pipeline executes the job in a defined manner by first coding it and then structuring it inside several blocks that may include several steps or tasks.

## What is SonarQube?

SonarQube is an open-source platform developed by SonarSource for continuous inspection of

code quality. Sonar does static code analysis, which provides a detailed report of bugs, code smells, vulnerabilities, code duplications.

It supports 25+ major programming languages through built-in rulesets and can also be extended with various plugins.

## Benefits of SonarQube

- **Sustainability** - Reduces complexity, possible vulnerabilities, and code duplications, optimizing the life of applications.
- **Increase productivity** - Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code
- **Quality code** - Code quality control is an inseparable part of the process of software development.
- **Detect Errors** - Detects errors in the code and alerts developers to fix them automatically before submitting them for output.
- **Increase consistency** - Determines where the code criteria are breached and enhances the quality
- **Business scaling** - No restriction on the number of projects to be evaluated
- **Enhance developer skills** - Regular feedback on quality problems helps developers to improve their coding skills

## Integrating Jenkins with SonarQube:

### Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

### Steps to create a Jenkins CI/CD Pipeline and use SonarQube to perform SAST

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.

The screenshot shows the Jenkins dashboard with the following interface elements:

- Header:** Jenkins logo, search bar (Search (CTRL+K)), help icon, notifications, security shield, user profile (Atharva Prabhakar Patil), and log out button.
- Left Sidebar:**
  - New Item
  - Build History
  - Project Relationship
  - Check File Fingerprint
  - Manage Jenkins
  - My Views
  - Build Queue (No builds in the queue)
  - Build Executor Status
- Central Area:**
  - Filter buttons: All (+)
  - Table header: S, W, Name, Last Success, Last Failure, Last Duration
  - Table rows:
    - Jenkins\_Pipeline: Last Success 1 mo 1 day, Last Failure N/A, Last Duration 3.5 sec
    - maven\_build: Last Success 3 days 16 hr, Last Failure N/A, Last Duration 2 min 9 sec
    - sonarqube: Last Success 11 hr, Last Failure N/A, Last Duration 4 min 18 sec
    - Tomact: Last Success 29 days, Last Failure 29 days, Last Duration 2.9 sec

2. Run SonarQube in a Docker container using this command -

```
PS C:\Users\sushmita> docker rm -f sonarqube
sonarqube
PS C:\Users\sushmita> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
365e3fcbb8a59edb271a2d9d7489e503571b2b822c64ec235a36a0aabf175c59
PS C:\Users\sushmita>
```

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.

The screenshot shows the SonarQube interface with the following sections:

- Header:** sonarqube logo, navigation menu (Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, Q), and user profile.
- Breadcrumbs:** sonarqube-test / main
- Top Navigation:** Overview, Issues, Security Hotspots, Measures, Code, Activity, Project Settings, Project Information.
- Section: Analysis Method**

Use this page to manage and set-up the way your analyses are performed.

How do you want to analyze your repository?

  - With Jenkins
  - With GitHub Actions
  - With Bitbucket Pipelines
  - With GitLab CI
  - With Azure Pipelines
  - Other CI
 

SonarQube integrates with your workflow no matter which CI tool you're using.

4. Login to SonarQube using username *admin* and password *admin*.

5. Create a local project in SonarQube with the name **sonarqube-test**

1 of 2

### Create a local project

**Project display name \***

**Project key \***

**Main branch name \***

The name of your project's default branch [Learn More](#)

**Cancel** **Next**

Setup the project and come back to Jenkins Dashboard.

Download the sonar scanner and place it in a directory using

mkdir Downloads\sonarqube

cd Downloads/sonarqube

Download by pasting the link in browser:

<https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-4.2.0.1873-windows.zip>

Paste the following command in Docker terminal

unzip sonar-scanner-cli-4.2.0.1873-windows.zip

Also download the latest java version

6. Create a New Item in Jenkins, choose **Pipeline**.

### New Item

Enter an item name

» This field cannot be empty, please enter a valid name

Select an item type



#### Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



#### Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



#### Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Ok

7. Under Pipeline Script, enter the following -

```
node {
    stage('Cloning the GitHub Repo') {
        git 'https://github.com/shazforiot/GOL.git'
    }
    stage('SonarQube analysis') {
        withSonarQubeEnv('sonarqube-test') {
            bat """
                C:\\\\Users\\\\sushmita\\\\Downloads\\\\sonar-scanner-6.2.0.4584-windows-x6
                4\\\\bin\\\\sonar-scanner.bat ^
                    -D sonar.login=admin ^
                    -D sonar.password=atharva ^
                    -D sonar.projectKey=sonarqube-test ^
                    -D sonar.exclusions=vendor/*,resources/.java ^
                    -D sonar.host.url=http://localhost:9000/
            """
        }
    }
}
```

Script ?

```
1+ node {
2  stage('Cloning the GitHub Repo')
3+ {
4  git 'https://github.com/shazforiot/GOL.git'
5 }
6+ stage('SonarQube analysis') {
7+ withSonarQubeEnv('sonarqube-test') {
8 bat """
9 C:\\\\Users\\\\sushmita\\\\Downloads\\\\sonar-scanner-6.2.0.4584-windows-x64\\\\bin\\\\sonar-scanner.bat ^
10 -D sonar.login=admin ^
11 -D sonar.password=atharva ^
12 -D sonar.projectKey=sonarqube-test ^
13 -D sonar.exclusions=vendor/*,resources/.java ^
14 -D sonar.host.url=http://localhost:9000/
15 """
16 }
17 }
18 }
```

Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

## 8. Run The Build.

The screenshot shows the Jenkins dashboard with the following interface elements:

- Dashboard >** (top left)
- New Item**
- Build History**
- Project Relationship**
- Check File Fingerprint**
- Manage Jenkins**
- My Views**

**Build Queue:** No builds in the queue.

**Build Executor Status:** (dropdown menu)

**Stage View:** This section displays the execution stages for five recent builds (#7 to #11). The columns represent the stages:

	Cloning the GitHub Repo	SonarQube analysis
Average stage times:	23s	34s
#11 Sep 26 19:21 No Changes	9s	9s failed
#10 Sep 26 19:02 No Changes	6s	6s failed
#9 Sep 26 18:52 No Changes	53s	1min 26s failed
#8 Sep 26 18:15 No Changes		
#7 Sep 26 18:13 No Changes		

## 9. Check the console output once the build is complete.



## Console Output

[Download](#)[Copy](#)[View as plain text](#)

```
Started by user Atharva Prabhakar Patil
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-test
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning the GitHub Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\sonarqube-test\.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/shazforiot/GOL.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/GOL.git
> git --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
```

10. After that, check the project in SonarQube.

The screenshot shows the SonarQube web interface. At the top, there's a navigation bar with tabs for 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', and 'More'. A search bar and a 'Create Project' button are also at the top right. On the left, there are filters for 'My Favorites' (selected) and 'All', and sections for 'Quality Gate' (Passed: 2, Failed: 0) and 'Reliability' (A: 1, B: 0, C: 1, D: 0, E: 0). The main area displays two projects:

- sonarqube PUBLIC**: Last analysis: 1 hour ago. Status: Passed. Message: The main branch of this project is empty.
- sonarqube-test PUBLIC**: Last analysis: 16 minutes ago. Metrics: 68k Lines of Code, 164k Maintainability, 0.0% Hotspots Reviewed, 50.6% Coverage. Status: Passed.

At the bottom right, it says "2 of 2 shown".

Under different tabs, check all different issues with the code.

### 11. Code Problems -

#### Open Issues

The screenshot shows the SonarQube web interface. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search bar. Below the navigation is a breadcrumb trail: sonarqube-test / main. The main content area has tabs for Overview, Issues, Security Hotspots, Measures (which is currently selected), Code, and Activity. On the right side, there are buttons for Project Settings, View as (set to Tree), Select files, and Navigate. The left sidebar contains sections for Security Review, Duplications, Size, Complexity, and Issues. Under the Issues section, 'Overall Code' is expanded, showing 'Open Issues' with a count of 210,549, which is highlighted with a blue selection bar. Other categories listed are Confirmed Issues (0), Accepted Issues (0), and False Positive Issues (0). The main panel displays a hierarchical tree of open issues. At the top level are 'gameoflife-acceptance-tests', 'gameoflife-build', 'gameoflife-core', 'gameoflife-deploy', 'gameoflife-web', and 'pom.xml'. The 'pom.xml' node is expanded, showing a single issue under it.

#### Consistency

**sonarqube** Projects Issues Rules Quality Profiles Quality Gates Administration More

star sonarqube-test / main

Overview Issues Security Hotspots Measures Code Activity Project Settings Project I

My Issues All

Select issues Navigate to issue 196,662 issues 3075d effort

Filters

Issues in new code

Clean Code Attribute

- Consistency 197k
- Intentionality 14k
- Adaptability 0
- Responsibility 0

Add to selection **Ctrl + click**

> Software Quality

gameoflife-core/build/reports/tests/all-tests.html

Insert a <!DOCTYPE> declaration to before this <html> tag.

Reliability

Open Not assigned L1 - 5min effort - 4 years ago - ⚡ Bug - ⚡ Major

Remove this deprecated "width" attribute.

Maintainability

Open Not assigned L9 - 5min effort - 4 years ago - ⚡ Code Smell - ⚡ Major

Remove this deprecated "align" attribute.

Maintainability

Open Not assigned L11 - 5min effort - 4 years ago - ⚡ Code Smell - ⚡ Major

Embedded database should be used for evaluation purposes only

## Intentionality

**sonarqube** Projects Issues Rules Quality Profiles Quality Gates Administration More

star sonarqube-test / main

Overview Issues Security Hotspots Measures Code Activity Project SettI

My Issues All

Select issues Navigate to issue 13,88

Filters

Issues in new code

Clean Code Attribute

- Consistency 197k
- Intentionality 14k
- Adaptability 0
- Responsibility 0

Add to selection **Ctrl + click**

> Software Quality

gameoflife-acceptance-tests/Dockerfile

Use a specific version tag for the image.

Maintainability

Open Not assigned L1 - 5min effort - 4 years ago -

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.

Maintainability

Open Not assigned L12 - 5min effort - 4 years ago -

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.

Maintainability

Open Not assigned L12 - 5min effort - 4 years ago -

Embedded database should be used for evaluation purposes only

## Bugs and Code Smells

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. On the left, a sidebar displays filtering options for Severity (High, Medium, Low), Type (Bug, Vulnerability, Code Smell), Scope, Status, and Security Category. The 'Code Smell' category is currently selected. The main panel lists three specific issues under the 'gameoflife-web/tools/jmeter/printable\_docs/building.html' file:

- Add an "alt" attribute to this image.** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L29 - 5min effort - 4 years ago - ⚡ Code Smell - ⚡ Minor
- Add an "alt" attribute to this image.** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L31 - 5min effort - 4 years ago - ⚡ Code Smell - ⚡ Minor
- Add an "alt" attribute to this image.** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L31 - 5min effort - 4 years ago - ⚡ Code Smell - ⚡ Minor

A warning message at the bottom states: "Embedded database should be used for evaluation purposes only. This embedded database will not make it will not connect to remote instances of SonarQube, and there is no access for external users other than the different databases mentioned."

## Reliability

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The 'Issues' tab is selected. The sidebar filters by Clean Code Attribute (Consistency, Intentionality, Adaptability, Responsibility), Software Quality (Security, Reliability, Maintainability), and Severity (High, Medium). The 'Reliability' category is selected. The main panel lists three specific issues under the 'gameoflife-core/build/reports/tests/all-tests.html' file:

- Add "lang" and/or "xml:lang" attributes to this "<html>" element.** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L1 - 2min effort - 4 years ago - ⚡ Bug - ⚡ Major
- Add "<th>" headers to this "<table>".** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L9 - 2min effort - 4 years ago - ⚡ Bug - ⚡ Major
- Add "lang" and/or "xml:lang" attributes to this "<html>" element.** Reliability: **Intentionality:** accessibility wcag2-a. Status: Open, Not assigned. L9 - 2min effort - 4 years ago - ⚡ Bug - ⚡ Major

A warning message at the bottom states: "Embedded database should be used for evaluation purposes only."

## Duplicates

sonarqube-test / main

**Duplicated Lines (%)** 50.6% [See history](#)

File	Duplicated Lines (%)	Count
gameoflife-web/tools/jmeter/docs/api/org/apache/jm.../ReportCellRenderer.html	92.4%	1,28
gameoflife-web/tools/jmeter/docs/api/org/apache/jm.../RightAlignRenderer.html	92.4%	1,19
gameoflife-web/tools/jmeter/docs/api/org/apache/jm.../JMeterCellRenderer.html	92.1%	1,28
gameoflife-web/tools/jmeter/docs/api/org/apache/jm.../FunctionHelper.html	89.5%	1,07
gameoflife-web/tools/jmeter/docs/api/org/apache/jm.../AjpSamplerGui.html	89.0%	1,21

## Cyclomatic Complexities

sonarqube-test / main

**Cyclomatic Complexity** 1,112 [See history](#)

File	Complexity
pom.xml	1,094
gameoflife-web	18
gameoflife-deploy	—
gameoflife-core	—
gameoflife-build	—
gameoflife-acceptance-tests	—

In this way, we have created a CI/CD Pipeline with Jenkins and integrated it with SonarQube to find issues in the code like bugs, code smells, duplicates, cyclomatic complexities, etc.

**Conclusion:**

In this experiment, we performed a static analysis of the code to detect bugs, code smells, and security vulnerabilities on our sample Java application.

**Aim:** To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

### **Theory:**

#### **What is Nagios?**

Nagios is an open-source software for continuous monitoring of systems, networks, and infrastructures. It runs plugins stored on a server that is connected with a host or another server on your network or the Internet. In case of any failure, Nagios alerts about the issues so that the technical team can perform the recovery process immediately.

Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture.

#### **Why We Need Nagios tool?**

Here are the important reasons to use Nagios monitoring tool:

- Detects all types of network or server issues
- Helps you to find the root cause of the problem which allows you to get the permanent solution to the problem
- Active monitoring of your entire infrastructure and business processes
- Allows you to monitor and troubleshoot server performance issues
- Helps you to plan for infrastructure upgrades before outdated systems create failures
- You can maintain the security and availability of the service
- Automatically fix problems in a panic situation

#### **Features of Nagios**

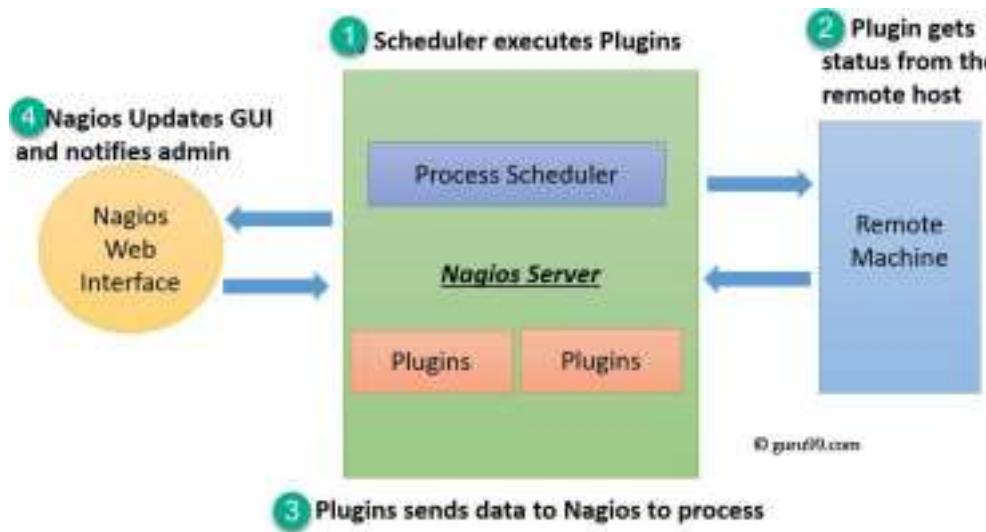
Following are the important features of Nagios monitoring tool:

- Relatively scalable, Manageable, and Secure
- Good log and database system
- Informative and attractive web interfaces
- Automatically send alerts if condition changes
- If the services are running fine, then there is no need to do check that host is an alive
- Helps you to detect network errors or server crashes
- You can troubleshoot the performance issues of the server.
- The issues, if any, can be fixed automatically as they are identified during the monitoring process
- You can monitor the entire business process and IT infrastructure with a single pass
- The product's architecture is easy to write new plugins in the language of your choice
- Nagios allows you to read its configuration from an entire directory which helps you to decide how to define individual files

- Utilizes topology to determine dependencies
- Monitor network services like HTTP, SMTP, HTTP, SNMP, FTP, SSH, POP, etc.
- Helps you to define network host hierarchy using parent hosts
- Ability to define event handlers that runs during service or host events for proactive problem resolution
- Support for implementing redundant monitoring hosts

## Nagios Architecture

Nagios is a client-server architecture. Usually, on a network, a Nagios server is running on a host, and plugins are running on all the remote hosts which should be monitored.



1. The scheduler is a component of the server part of Nagios. It sends a signal to execute the plugins at the remote host.

2. The plugin gets the status from the remote host
3. The plugin sends the data to the process scheduler
4. The process scheduler updates the GUI and notifications are sent to admins.

## Installation of Nagios

**Prerequisites:** AWS Free Tier

Steps:

1. Create an Amazon Linux EC2 Instance in AWS and name it - nagios-host

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with links like EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances, and Instances. The main area is titled 'Instances (1) Info' and shows one instance: 'nagios-host' (i-028182fbe9c070820). The instance is listed as 'Running' with a green checkmark, 't2.micro' as the instance type, and 'Initializing' as the status check. The availability zone is 'us-east-1d'. There are buttons for 'Launch instances' and 'Actions'.

- Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.  
You have to edit the inbound rules of the specified Security Group for this.

The screenshot shows the 'Inbound rules' section of a security group. It lists five rules:

- SSH (TCP port 22) - Source: 0.0.0.0/0
- HTTP (TCP port 80) - Source: 0.0.0.0/0
- HTTPS (TCP port 443) - Source: 0.0.0.0/0
- All ICMP - IPv4 - Source: 0.0.0.0/0
- (rule ID sgr-04093d1a208295e38) - Source: 0.0.0.0/0

A warning message at the bottom states: "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." There are buttons for 'Add rule', 'Cancel', 'Preview changes', and 'Save rules'.

- SSH into Your EC2 instance or simply use EC2 Instance Connect from the browser.



4. Update the package indices and install the following packages using yum

```
sudo yum update
sudo yum install httpd php
sudo yum install gcc glibc glibc-common
sudo yum install gd gd-devel
yum install: error: unrecognized arguments: -uy
[ec2-user@ip-172-31-38-150 ~]$ sudo yum install gd gd-devel -y
Last metadata expiration check: 0:05:12 ago on Sun Oct 6 11:15:04 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
<b>Installing:</b>				
gd	x86_64	2.3.3-5.amzn2023.0.3	amazonlinux	139
gd-devel	x86_64	2.3.3-5.amzn2023.0.3	amazonlinux	38
<b>Installing dependencies:</b>				
brotli	x86_64	1.0.9-4.amzn2023.0.2	amazonlinux	314
brotli-devel	x86_64	1.0.9-4.amzn2023.0.2	amazonlinux	31
bzip2-devel	x86_64	1.0.8-6.amzn2023.0.2	amazonlinux	214
cairo	x86_64	1.17.6-2.amzn2023.0.1	amazonlinux	684
cmake-filesystem	x86_64	3.22.2-1.amzn2023.0.4	amazonlinux	16
fontconfig	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	273
fontconfig-devel	x86_64	2.13.94-2.amzn2023.0.2	amazonlinux	128
fonts-filesystem	noarch	1:2.0.5-12.amzn2023.0.2	amazonlinux	9.5
freetype	x86_64	2.13.2-5.amzn2023.0.1	amazonlinux	423
freetype-devel	x86_64	2.13.2-5.amzn2023.0.1	amazonlinux	912
glib2-devel	x86_64	2.74.7-689.amzn2023.0.2	amazonlinux	486
google-noto-fonts-common	noarch	20201206-2.amzn2023.0.2	amazonlinux	15
google-noto-sans-vf-fonts	noarch	20201206-2.amzn2023.0.2	amazonlinux	492
graphite2	x86_64	1.3.14-7.amzn2023.0.2	amazonlinux	97
graphite2-devel	x86_64	1.3.14-7.amzn2023.0.2	amazonlinux	21
harfbuzz	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	868
harfbuzz-devel	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	404
harfbuzz-icu	x86_64	7.0.0-2.amzn2023.0.1	amazonlinux	18
jbigkit-libs	x86_64	2.1-21.amzn2023.0.2	amazonlinux	54
langpacks-core-font-en	noarch	3.0-21.amzn2023.0.4	amazonlinux	10
libICE	x86_64	1.0.10-6.amzn2023.0.2	amazonlinux	71
libSM	x86_64	1.2.3-8.amzn2023.0.2	amazonlinux	42

5. Create a new Nagios User with its password. You'll have to enter the password twice for confirmation.

```
sudo adduser -m nagios
sudo passwd nagios
```

```
Complete!
[ec2-user@ip-172-31-38-150 ~]$ sudo adduser -m nagios
[ec2-user@ip-172-31-38-150 ~]$ sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-38-150 ~]$
```

6. Create a new user group

```
sudo groupadd nagcmd
```

7. Use these commands so that you don't have to use sudo for Apache and Nagios

```
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
```

## 8. Create a new directory for Nagios downloads

```
mkdir ~/downloads
cd ~/downloads
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-38-150 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-38-150 ~]$ sudo usermod -a -G nagcmd nagios
[ec2-user@ip-172-31-38-150 ~]$ sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-38-150 ~]$ mkdir ~/downloads
[ec2-user@ip-172-31-38-150 ~]$ cd ~/downloads
[ec2-user@ip-172-31-38-150 downloads]$ wget https://go.nagios.org/24-09-17/6kqcx
```

## 9. Use wget to download the source zip files.

```
wget https://go.nagios.org/1/975333/2024-09-17/6kqcx
wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
```

```
[ec2-user@ip-172-31-38-150 ~]$ wget https://go.nagios.org/24-09-17/6kqcx
--2024-10-06 11:23:50-- https://go.nagios.org/1/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org) ... 3.92.127.219, ...
Connecting to go.nagios.org (go.nagios.org) |3.92.1|
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagios.tar.gz?utm_source=Nagios.org&utm_content=Download+5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc3
```

## 10. Use tar to unzip and change to that directory.

```
tar zxvf nagios-4.5.5.tar.gz
cd nagios-4.5.5
--2024-10-06 11:23:50-- https://go.nagios.org/1/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org) ... 3.92.120.28, 52.54.96.194, 3.215
72.219, ...
Connecting to go.nagios.org (go.nagios.org) |3.92.120.28|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e24439d4a81d8 [following]
--2024-10-06 11:23:50-- http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f011e969f2a75b0e2254439d4a81d8
Resolving assets.nagios.com (assets.nagios.com) ... 45.79.49.120, 2600:3c00:03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com) |45.79.49.120|:80... con
```

11. Run the configuration script with the same group name you previously created.

```
./configure --with-command-group=nagcmd
```

12. Compile the source code.

```
make all
```

```
Command Prompt
*** Configuration summary for nagios 4.5.5 2024-09-17 ***

General Options:
-----
    Nagios executable: nagios
    Nagios user/group: nagios,nagios
    Command user/group: nagios,nagcmd
        Event Broker: yes
    Install ${prefix}: /usr/local/nagios
    Install ${includedir}: /usr/local/nagios/include/nagios
        Lock file: /run/nagios.lock
    Check result directory: /usr/local/nagios/var/spool/checkresults
        Init directory: /lib/systemd/system
    Apache conf.d directory: /etc/httpd/conf.d
        Mail program: /bin/mail
        Host OS: linux-gnu
    IOBroker Method: epoll

Web Interface Options:
-----
    HTML URL: http://localhost/nagios/
    CGI URL: http://localhost/nagios/cgi-bin/
Traceroute (used by WAP): /usr/bin/traceroute

Review the options above for accuracy. If they look okay,
type 'make all' to compile the main program and CGIs.

[ec2-user@ip-172-31-80-195 nagios-4.5.5]$ make all
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o ../common/shared.o ../common/shared.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o workers.o workers.c
In function 'get_worker',
  inlined from 'get_worker' at workers.c:277:12:
workers.c:253:17: warning: '%s' directive argument is null [-Wformat-overflow=]
  253 |         log_debug_info(DEBUG_C_CHECKS, 1, "Found specialized worker(s) for '%s'", (slash && *slash != '/') ? slash : cmd_name);
  |
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o checks.o checks.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o config.o config.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o commands.o commands.c
gcc -Wall -I.. -I. -I..../include -I..../include -I.. -g -O2 -DHAVE_CONFIG_H -DSCORE -c -o events.o events.c
```

13. Install binaries, init script and sample config files. Lastly, set permissions on the external command directory.

```
sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
```

```
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagio
s.conf
if [ 0 -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagi
os.conf; \
fi

*** Nagios/Apache conf file installed ***

[ec2-user@ip-172-31-38-150 nagios-4.4.6]$
```

```
sudo nano /usr/local/nagios/etc/objects/contacts.cfg
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ sudo yum install openssl
evel
Last metadata expiration check: 0:13:31 ago on Sun Oct  6 11:15:04 2024.
Dependencies resolved.
=====
=====
Package          Architecture      Version
Repository      Size
=====
=====
Installing:
openssl-devel      x86_64        1:3.0.8-1.amzn2023.0.14
amazonlinux       3.0 M
Transaction Summary
```

#### 14. Configure the web interface.

```
sudo make install-webconf
```

```
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagio
s.conf
if [ 0 -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagi
os.conf; \
fi

*** Nagios/Apache conf file installed ***

[ec2-user@ip-172-31-38-150 nagios-4.4.6]$
```

#### 15. Create a nagiosadmin account for nagios login along with password. You'll have to specify the password twice.

```
sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

```
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
[sudo] password for ec2-user:
[sudo] password for ec2-user:
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$
```

#### 16. Restart Apache

```
sudo service httpd restart
```

```
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-38-150 nagios-4.4.6]$
```

#### 17. Go back to the downloads folder and unzip the plugins zip file.

```
cd ~/downloads
tar zxvf nagios-plugins-2.4.11.tar.gz
[ec2-user@ip-172-31-38-150 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
```

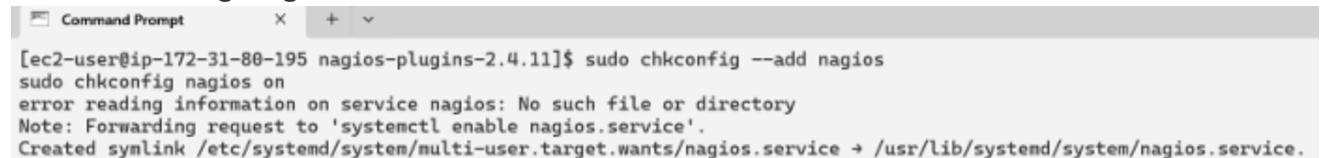
#### 18. Compile and install plugins

```
cd nagios-plugins-2.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
sudo make install
[ec2-user@ip-172-31-38-150 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios
--with-nagios-group=nagios
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios
--with-nagios-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
```

## 19. Start Nagios

Add Nagios to the list of system services

```
sudo chkconfig --add nagios  
sudo chkconfig nagios on
```



```
[ec2-user@ip-172-31-80-195 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios  
sudo chkconfig nagios on  
error reading information on service nagios: No such file or directory  
Note: Forwarding request to 'systemctl enable nagios.service'.  
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
```

Verify the sample configuration files

```
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg  
  
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v  
/usr/local/nagios/etc/nagios.cfg  
/usr/local/nagios/etc/nagios.cfg  
  
Nagios Core 4.4.6  
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors  
Copyright (c) 1999-2009 Ethan Galstad  
Last Modified: 2020-04-28  
License: GPL  
  
Website: https://www.nagios.org  
Reading configuration data...  
  Read main config file okay...  
  Read object config files okay...  
  
Running pre-flight check on configuration data...  
  
Checking objects...  
  Checked 8 services.  
  Checked 1 hosts.  
  Checked 1 host groups.  
  Checked 0 service groups.  
  Checked 1 contacts.  
  Checked 1 contact groups.  
  Checked 24 commands.  
  Checked 5 time periods.
```

If there are no errors, you can go ahead and start Nagios.

```
sudo service nagios start
```

```
Things look okay - No serious problems were detected during the pre-flight check  
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ sudo service nagios start  
Redirecting to /bin/systemctl start nagios.service  
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$
```

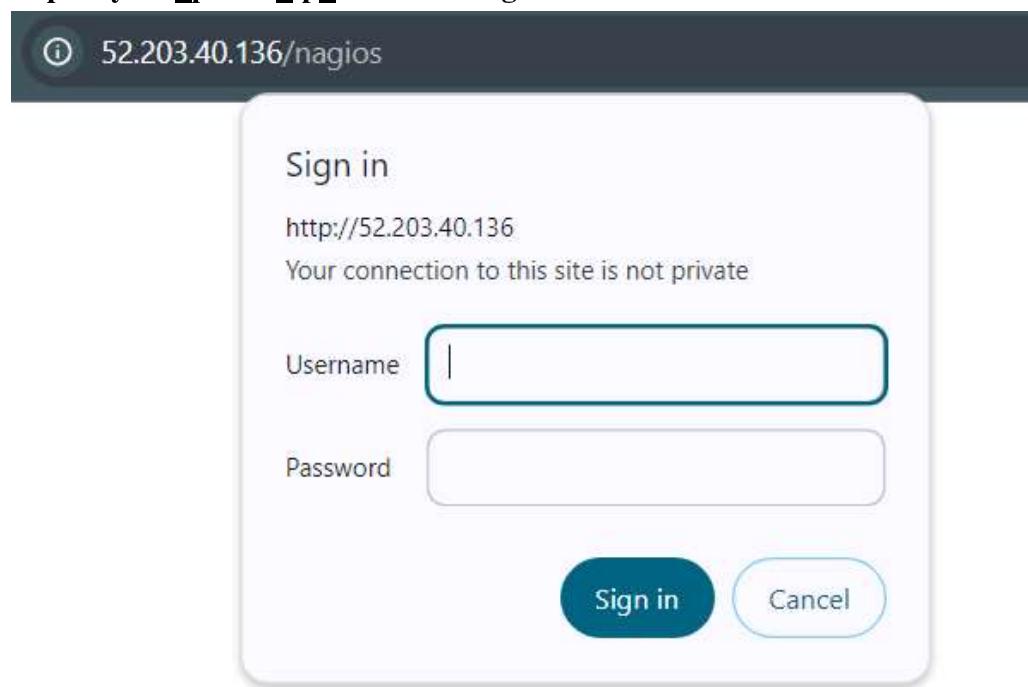
## 20. Check the status of Nagios

```
sudo systemctl status nagios
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-38-150 nagios-plugins-2.4.11]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.4.6
    Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
    Active: active (running) since Sun 2024-10-06 11:51:46 UTC; 1min 33s ago
      Docs: https://www.nagios.org/documentation
   Main PID: 89956 (nagios)
     Tasks: 6 (limit: 1112)
    Memory: 2.4M
       CPU: 36ms
      CGroup: /system.slice/nagios.service
              ├─89956 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
              ├─89957 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─89958 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─89959 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─89960 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              └─89961 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 06 11:51:46 ip-172-31-38-150.ec2.internal nagios[89956]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfull>
Oct 06 11:51:46 ip-172-31-38-150.ec2.internal nagios[89956]: qh: core query handler re
```

21. Go back to EC2 Console and copy the Public IP address of this instance

22. Open up your browser and look for  
**[http://<your\\_public\\_ip\\_address>/nagios](http://<your_public_ip_address>/nagios)**



Enter username as nagiosadmin and password which you set in Step 16.

23. After entering the correct credentials, you will see this page.

The screenshot shows the Nagios Core 4.4.6 dashboard. At the top right, it displays "Nagios® Core™ Version 4.4.6" and the date "April 28, 2020". Below that is a blue box with the message "A new version of Nagios Core is available! Visit [nagios.org](#) to download Nagios 4.5.5." On the left, there's a sidebar with links for General (Home, Documentation), Current Status (Tactical Overview, Map (Legacy), Hosts, Services, Host Groups, Service Groups, Problems), Reports (Availability, Trends (Legacy), Alerts, Notifications, Event Log), and a search bar. The main content area has a "Get Started" section with a bulleted list of steps: Start monitoring your infrastructure, Change the look and feel of Nagios, Extend Nagios with hundreds of addons, Get support, Get training, and Get certified. To the right is a "Quick Links" section with a bulleted list: Nagios Library (tutorials and docs), Nagios Labs (development blog), Nagios Exchange (plugins and addons), Nagios Support (tech support), Nagios.com (company), and Nagios.org (project). Below these are "Latest News" and "Don't Miss..." sections, both currently empty.

This means that Nagios was correctly installed and configured with its plugins so far.

**Conclusion:** We have successfully installed and configured Nagios Core, Nagios Plugins, and NRPE on a Linux machine. This enables us to effectively manage system performance and proactively address potential issues.

# Advanced DevOps

## Lab Experiment 10

**Aim:** To perform Port, Service monitoring, Windows/Linux server monitoring using

Nagios. **Steps:**

Prerequisites: AWS Free Tier, Nagios Server running on Amazon Linux Machine.

1. To Confirm that Nagios is running **on the server side**, run this  
*sudo systemctl status nagios* on the “NAGIOS HOST”.



```
ec2-user@ip-172-31-81-4: ~
Last login: Mon Sep 23 15:36:29 2024 from 152.58.4.81
[ec2-user@ip-172-31-81-4 ~]$ sudo systemctl status nagios
* nagios.service - Nagios Core 4.5.5
  Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
  Active: active (running) since Mon 2024-09-23 16:32:36 UTC; 18min ago
    Docs: https://www.nagios.org/documentation
  Process: 1969 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
  Process: 1971 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 1972 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 6.8M
     CPU: 320ms
    CGroup: /system.slice/nagios.service
            └─1972 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
              ├─1974 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─1975 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─1976 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              ├─1977 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
              └─1983 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

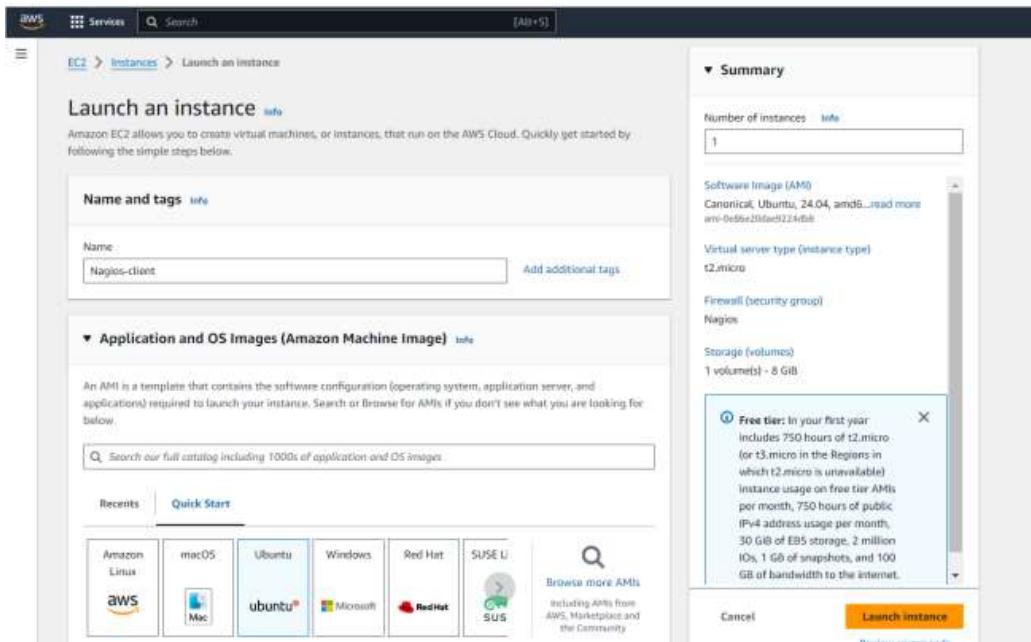
Sep 23 16:32:36 ip-172-31-81-4.ec2.internal systemd[1]: Started nagios.service - Nagios Core 4.5.5.
```

You can proceed if you get this message.

2. Before we begin,

To monitor a Linux machine, create an Ubuntu 20.04 server EC2 Instance in AWS.

Provide it with the same security group as the Nagios Host and name it ‘linux-client’ alongside the host.



**For now, leave this machine as is, and go back to your nagios HOST machine.**

3. On the server, run this command

```
ps -ef | grep nagios
Last login: Sat Oct  5 16:58:17 2024 from 42.111.112.18
[ec2-user@ip-172-31-43-65 ~]$ ps -ef | grep nagios
nagios  97412      1  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios
nagios  97413  97412  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios --worker /
.s.qh
nagios  97414  97412  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios --worker /
.s.qh
nagios  97415  97412  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios --worker /
.s.qh
nagios  97416  97412  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios --worker /
.s.qh
nagios  97417  97412  0 17:34 ?          00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios
ec2-user  98423  98399  0 17:51 pts/2      00:00:00 grep --color=auto nagios
```

4. Become a root user and create 2 folders

```
sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
```

```
[ec2-user@ip-172-31-43-65 ~]$ sudo su
mkdir /usr/local/nagios/etc/objects/monitorhosts
mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-43-65 ec2-user]# |
```

5. Copy the sample localhost.cfg file to linuxhost folder

```
cp /usr/local/nagios/etc/objects/localhost.cfg
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

```
[root@ip-172-31-81-4 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-81-4 ec2-user]# |
```

6. Open linuxserver.cfg using nano and make the following changes

```
nano
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

Change the hostname to linuxserver (EVERYWHERE ON THE FILE) Change address to the public IP address of your **LINUX CLIENT.**

```
GNU nano 5.8                               /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
#####
# LOCALHOST.CFG - SAMPLE OBJECT CONFIG FILE FOR MONITORING THIS MACHINE
#
#
# NOTE: This config file is intended to serve as an *extremely* simple
#       example of how you can create configuration entries to monitor
#       the local (Linux) machine.
#
#####

#####
# HOST DEFINITION
#
#####
# Define a host for the local machine
define host {
    use          linux-server           ; Name of host template to use
                           ; This host definition will inherit all variables that are defined
                           ; in (or inherited by) the linux-server host template definition.
    host_name    localhost
    alias        localhost
    address     127.0.0.1
}

#####
# HOST GROUP DEFINITION
#
```

Change hostgroup\_name under hostgroup to linux-servers1

Everywhere else on the file, change the hostname to linuxserver instead of localhost.

7. Open the Nagios Config file and add the following line

```
nano /usr/local/nagios/etc/nagios.cfg
```

```
##Add this line
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
root@ip-172-31-81-4:~# nano /usr/local/nagios/etc/nagios.cfg
GNU nano 5.8                               /usr/local/nagios/etc/nagios.cfg
#####
#
# NAGIOS.CFG - Sample Main Config File for Nagios 4.5.5
#
# Read the documentation for more information on this configuration
# file. I've provided some comments here, but things may not be so
# clear without further explanation.
#
#
#####
#
# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes. This should be the first option specified
# in the config file!!!
#
log_file=/usr/local/nagios/var/nagios.log

#
# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

# Definitions for monitoring a Windows machine

FG Help      ^O Write Out   ^W Where Is   ^X Cut          ^T Execute Justify   ^L Location Go To Line   ^U Undo   ^R Redo   ^A Set Mark   ^B To Bracket
^X Exit      ^R Read File   ^W Replace    ^P Paste         ^D Delete   ^G Find     ^C Copy      ^V Paste        ^I Insert   ^O Open     ^N New     ^F Find Next
^S Save      ^Q Quit       ^A All Select  ^K Kill        ^H Home     ^J End     ^E End Line  ^G Go To Line  ^U Undo   ^R Redo
^Z Undo      ^C Copy      ^V Paste        ^K Kill        ^H Home     ^J End     ^E End Line  ^G Go To Line  ^U Undo   ^R Redo
^X Exit      ^R Read File   ^W Replace    ^P Paste         ^D Delete   ^G Find     ^C Copy      ^V Paste        ^I Insert   ^O Open     ^N New     ^F Find Next
^S Save      ^Q Quit       ^A All Select  ^K Kill        ^H Home     ^J End     ^E End Line  ^G Go To Line  ^U Undo   ^R Redo
^Z Undo      ^C Copy      ^V Paste        ^K Kill        ^H Home     ^J End     ^E End Line  ^G Go To Line  ^U Undo   ^R Redo
```

8. Verify the configuration files

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

```
Running pre-flight check on configuration data...

Checking objects...
    Checked 8 services.
    Checked 2 hosts.
    Checked 2 host groups.
    Checked 0 service groups.
    Checked 1 contacts.
    Checked 1 contact groups.
    Checked 24 commands.
    Checked 5 time periods.
    Checked 0 host escalations.
    Checked 0 service escalations.

Checking for circular paths...
    Checked 2 hosts
    Checked 0 service dependencies
    Checked 0 host dependencies
    Checked 5 timeperiods

Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-43-65 ec2-user]# |
```

You are good to go if there are no errors.

9. Restart the nagios service

```
service nagios restart
[root@ip-172-31-81-4 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-81-4 ec2-user]# sudo systemctl status nagi
```

Now it is time to switch to the client machine.

10. SSH into the machine or simply use the EC2 Instance Connect feature.

11. Make a package index update and install gcc, nagios-nrpe-server and the plugins.

```
sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
ubuntu@ip-172-31-33-76:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [382 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [537 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [132 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [8860 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [384 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [159 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [45.0 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [14.9 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.4 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 kB]
```

12. Open nrpe.cfg file to make changes.

```
sudo nano /etc/nagios/nrpe.cfg
```

Under allowed\_hosts, add your nagios host IP address like so

```

ubuntu@ip-172-31-83-152:~$ nano /etc/nagios/nrpe.cfg
GNU nano 7.2
/etc/nagios/nrpe.cfg *
# This determines the effective user that the NRPE daemon should run as.
# You can either supply a username or a UID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_user=nagios

# NRPE GROUP
# This determines the effective group that the NRPE daemon should run as.
# You can either supply a group name or a GID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_group=nagios

# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
allowed_hosts=127.0.0.1,:1,3.86.12.126

# COMMAND ARGUMENT PROCESSING

```

Terminal menu bar: Help, Write Out, Where Is, Cut, Paste, Execute, Location, Undo, Set Mark, To Bracket, Exit, Read File, Replace, Justify, Go To Line, Redo, Copy, Where Was.

### 13. Restart the NRPE server

```

sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-83-152:~$ sudo nano /etc/nagios/nrpe.cfg

ubuntu@ip-172-31-83-152:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-83-152:~$ 
```

### 14. Now, check your nagios dashboard and you'll see a new host being added.

Click on Hosts.

Nagios® Core™ Version 4.5.5 September 17, 2024 Check for updates

Get Started

- Start monitoring your infrastructure
- Change the look and feel of Nagios
- Extend Nagios with hundreds of addons
- Get support
- Get training
- Get certified

Quick Links

- Nagios Library (tutorials and docs)
- Nagios Labs (development blog)
- Nagios Exchange (plugins and addons)
- Nagios Support (tech support)
- Nagios.com (company)
- Nagios.org (project)

Latest News

Don't Miss...

Copyright © 2010-2024 Nagios Core Development Team and Community Contributors. Copyright © 1999-2009 Ethan Galstad. See the THANKS file for more information on contributors.

Nagios Core is licensed under the GNU General Public License and is provided AS IS WITH NO WARRANTY OF ANY KIND, INCLUDING THE WARRANTY OF DESIGN, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Nagios, Nagios Core and the Nagios logo are trademarks, service marks, registered trademarks or registered service marks owned by Nagios Enterprises, LLC. Use of the Nagios marks is governed by the trademark use restrictions.

Click on linuxserver to see the host details

**Current Network Status**

Last Updated: Sun Oct 6 17:55:03 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

[View Service Status Detail For All Host Groups](#)  
[View Status Overview For All Host Groups](#)  
[View Status Summary For All Host Groups](#)  
[View Status Grid For All Host Groups](#)

**Host Status Totals**

Up	Down	Unreachable	Pending
2	0	0	0
All Problems	All Types		
0	2		

**Service Status Totals**

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0
All Problems	All Types			
2	8			

**Host Status Details For All Host Groups**

Host	Status	Last Check	Duration	Status Information
linuxserver	UP	10-06-2024 17:50:12	0d 0h 24m 51s	PING OK - Packet loss = 0%, RTA = 0.77 ms
localhost	UP	10-06-2024 17:53:57	1d 0h 21m 53s	PING OK - Packet loss = 0%, RTA = 0.03 ms

Results 1 - 2 of 2 Matching Hosts

You can click Services to see all services and ports being monitored.

**Host Information**

Last Updated: Sun Oct 6 17:43:35 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

[View Status Detail For This Host](#)  
[View Alert History For This Host](#)  
[View Trends For This Host](#)  
[View Alert Histogram For This Host](#)  
[View Availability Report For This Host](#)  
[View Notifications For This Host](#)

**Host**  
localhost  
(localhost)

**Member of**  
linux-servers, linux-servers1

**Address**  
127.0.0.1

**Host State Information**

<b>Host Status:</b> UP (for 1d 0h 10m 25s)
<b>Status Information:</b> PING OK - Packet loss = 0%, RTA = 0.03 ms
<b>Performance Data:</b> rta=0.030000ms;3000.000000;5000.000000;0.000000 p=0%:80;100,0
<b>Current Attempt:</b> 1/1 (HARD state)
<b>Last Check Time:</b> 10-06-2024 17:38:57
<b>Check Type:</b> ACTIVE
<b>Check Latency / Duration:</b> 0.000 / 4.140 seconds
<b>Next Scheduled Active Check:</b> 10-06-2024 17:43:57
<b>Last State Change:</b> 10-05-2024 17:33:10
<b>Last Notification:</b> N/A (notification 0)
<b>Is This Host Flapping?</b> NO (0.00% state change)
<b>In Scheduled Downtime?</b> NO
<b>Last Update:</b> 10-06-2024 17:43:34. (0d 0h 0m 1s ago)

**Active Checks:** ENABLED  
**Passive Checks:** ENABLED  
**Obsessing:** ENABLED  
**Notifications:** ENABLED  
**Event Handler:** ENABLED  
**Flap Detection:** ENABLED

**Host Commands**

- Locate host on map
- Disable active checks of this host
- Re-schedule the next check of this host
- Submit passive check result for this host
- Stop accepting passive checks for this host
- Stop obsessing over this host
- Disable notifications for this host
- Send custom host notification
- Schedule downtime for this host
- Schedule downtime for all services on this host
- Disable notifications for all services on this host
- Enable notifications for all services on this host
- Schedule a check of all services on this host
- Disable checks of all services on this host
- Enable checks of all services on this host
- Disable event handler for this host
- Disable flap detection for this host
- Clear flapping state for this host

**Host Comments**

Add a new comment

Entry Time Author Comment Comment ID Persistent Type Expires Actions

This host has no comments associated with it.

**Current Network Status**

Last Updated: Sun Oct 6 17:58:02 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

[View History For All hosts](#)  
[View Notifications For All Hosts](#)  
[View Host Status Detail For All Hosts](#)

**Host Status Totals**

Up	Down	Unreachable	Pending
2	0	0	0
All Problems	All Types		
0	2		

**Service Status Totals**

Ok	Warning	Unknown	Critical	Pending
6	1	0	1	0
All Problems	All Types			
2	8			

**Service Status Details For All Hosts**

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-06-2024 17:56:27	1d 0h 24m 52s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	10-06-2024 17:57:42	1d 0h 24m 14s	1/4	USERS OK - 6 users currently logged in
	HTTP	WARNING	10-06-2024 17:53:57	0d 0h 19m 5s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.001 second response time
	PING	OK	10-06-2024 17:55:12	1d 0h 22m 59s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms
	Root Partition	OK	10-06-2024 17:57:04	1d 0h 22m 22s	1/4	DISK OK - free space / 5567 MB (68.59% inode=98%)
	SSH	OK	10-06-2024 17:53:19	1d 0h 21m 44s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
	Swap Usage	CRITICAL	10-06-2024 17:54:34	1d 0h 31m 7s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
	Total Processes	OK	10-06-2024 17:56:22	1d 0h 20m 29s	1/4	PROCS OK. 39 processes with STATE = RSZDT

Results 1 - 8 of 8 Matching Services

As you can see, we have our linuxserver up and running. It is showing critical status on HTTP due to permission errors and swap because there is no partition created.

**In this case, we have monitored -**

**Servers: 1 linux server**

**Services: swap**

**Ports: 22, 80 (ssh, http)**

**Processes: User status, Current load, total processes, root partition, etc.**

## **Recommended Cleanup**

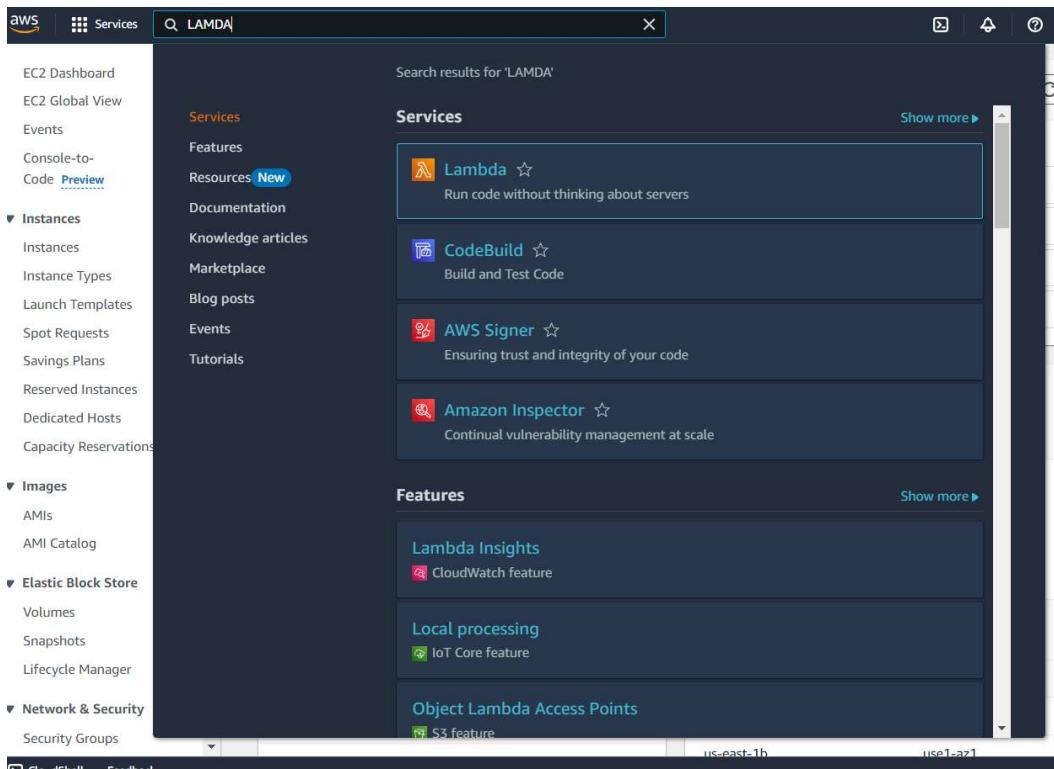
- Terminate both of your EC-2 instances to avoid charges.
- Delete the security group if you created a new one (it won't affect your bill, you may avoid it)

**Conclusion:** In conclusion, the experiment focused on monitoring ports, services, and a Linux server using Nagios. Through the step-by-step process, we successfully configured Nagios to monitor essential network services on the Linux server. By setting up both the Nagios host and client, we were able to track system performance, ensure service availability, and monitor key metrics like CPU and memory usage.

**Aim:** To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

## Step 1: Accessing AWS

Log in to your AWS Personal/Academy account. Navigate to the Lambda service by searching for "Lambda" in the AWS Management Console.



## Step 2: Creating a New Lambda Function

Click on the "Create function" button. Provide a name for your Lambda function and select the language you wish to use, such as Python 3.12. For architecture, choose x86, and for execution role, opt to create a new role with basic Lambda g permissions.

The screenshot shows the AWS Lambda landing page. At the top left, it says "Compute". In the center, there's a large heading "AWS Lambda" followed by the subtext "lets you run code without thinking about servers.". Below this, a paragraph explains: "You pay only for the compute time that you consume — there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service, all with zero administration." To the right, a white box contains the "Get started" section with the subtext "Author a Lambda function from scratch, or choose from one of many preconfigured examples." and a yellow "Create a function" button.

How it works

.NET | Java | **Node.js** | Python | Ruby | Custom runtime

```
1 * exports.handler = async (event) => {  
2     console.log(event);  
3     return 'Hello from Lambda!';  
4 };  
5
```

### Step 3: Configuring Basic Settings

To modify the basic settings, navigate to the "Configuration" tab and click on "Edit" under General Settings. Here, you can add a description and adjust the memory and timeout settings. For this experiment, I set the timeout to 1 second, which is sufficient for testing.

The screenshot shows the "Basic information" configuration page for a Lambda function. It includes fields for "Function name" (set to "lamda\_demo"), "Runtime" (set to "Python 3.12"), "Architecture" (set to "x86\_64"), and "Permissions" (with a note about default execution role creation). A "Change default execution role" link is also present.

Basic information

Function name

Enter a name that describes the purpose of your function.  
lamda\_demo

Runtime Info

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.12

Architecture Info

Choose the instruction set architecture you want for your function code.

x86\_64  
 arm64

Permissions Info

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▶ Change default execution role

**Permissions** [Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

Create a new role with basic Lambda permissions

Use an existing role

Create a new role from AWS policy templates

**Info** Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named ATHARV\_LAMDA-role-0u7c9ooi, with permission to upload logs to Amazon CloudWatch Logs.

► Additional Configurations

Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

[Cancel](#) [Create function](#)

**aws** Services Search [Alt+S]

Successfully created the function **lambda\_demo**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > lambda\_demo

**lambda\_demo**

▼ Function overview [Info](#)

Throttle [Copy ARN](#) Actions ▾

Diagram Template

**lambda\_demo**

+ Add trigger + Add destination

Description

Last modified 26 seconds ago

Function ARN arn:aws:lambda:eu-north-1:010928207735: function:lambda\_demo

Function URL [Info](#)

Code Test Monitor Configuration Aliases Versions

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

In this tutorial you will learn how to:

- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

[Learn more](#) [Start tutorial](#)

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

```
import json
def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

## Step 4: Testing the Function

Click on the "Test" tab and select "Create a new event." Name your event, set the event sharing to private, and choose the "hello-world" template.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

Create new event       Edit saved event

Event name

MyEventName

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Private  
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

Shareable  
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

Event JSON

```
1 "key1": "value1",
2 "key2": "value2",
3 "key3": "value3"
```

Format JSON

**Code source** [Info](#)

File Edit Find View Go Tools Window **Test** Deploy

Upload from ▾

Environment

Go to Anything (Ctrl-P)

lambda\_function

lambda\_demo /

lambda\_function.py

Configure test event Ctrl-Shift-C

Private saved events

● demo1

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps('Hello from Lambda!')
8     }
```

[Code](#) [Test](#) [Monitor](#) [Configuration](#) [Aliases](#) [Versions](#)

Code source [Info](#)

File Edit Find View Go Tools Window **Test** Deploy

Upload from ▾

Environment

Go to Anything (Ctrl-P)

lambda\_function

Environment Var

Execution result

Execution results

Test Event Name

demo1

Status: Succeeded | Max memory used: 32 MB | Time: 1.35 ms

Response

```
{ "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}
```

Function Logs

```
START RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa Version: $LATEST
END RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa
REPORT RequestId: 86829fa5-e154-46b3-8ff5-6f12ba6c3efa Duration: 1.35 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB
```

Request ID

```
86829fa5-e154-46b3-8ff5-6f12ba6c3efa
```

The screenshot shows the AWS Lambda function editor interface. The top navigation bar includes 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), 'Deploy', and a status message 'Changes not deployed'. On the left, there's a sidebar with 'Environment' and a search bar 'Go to Anything (Ctrl-P)'. The main area displays the code for 'lambda\_function.py':

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     my_string="Hello this is Exp 11"
6     return {
7         'statusCode': 200,
8         'body': json.dumps(my_string)
9     }
10
```

## Step 5: Running the Test

In the Code section, select the newly created event from the dropdown menu and click on "Test." You should see the output displayed below.

The screenshot shows the AWS Lambda function editor interface with the 'Execution result' tab selected. The top navigation bar remains the same. The main area displays the execution results for the 'demo1' test event:

**Test Event Name**: demo1

**Response**:

```
{
    "statusCode": 200,
    "body": "\"Hello this is Exp 11\""
}
```

**Function Log**

```
START RequestId: 298ba9d9-f73b-4abc-927b-6d5e5c8ce4ec Version: $LATEST
END RequestId: 298ba9d9-f73b-4abc-927b-6d5e5c8ce4ec
REPORT RequestId: 298ba9d9-f73b-4abc-927b-6d5e5c8ce4ec Duration: 1.62 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 0 ms
Request ID
298ba9d9-f73b-4abc-927b-6d5e5c8ce4ec
```

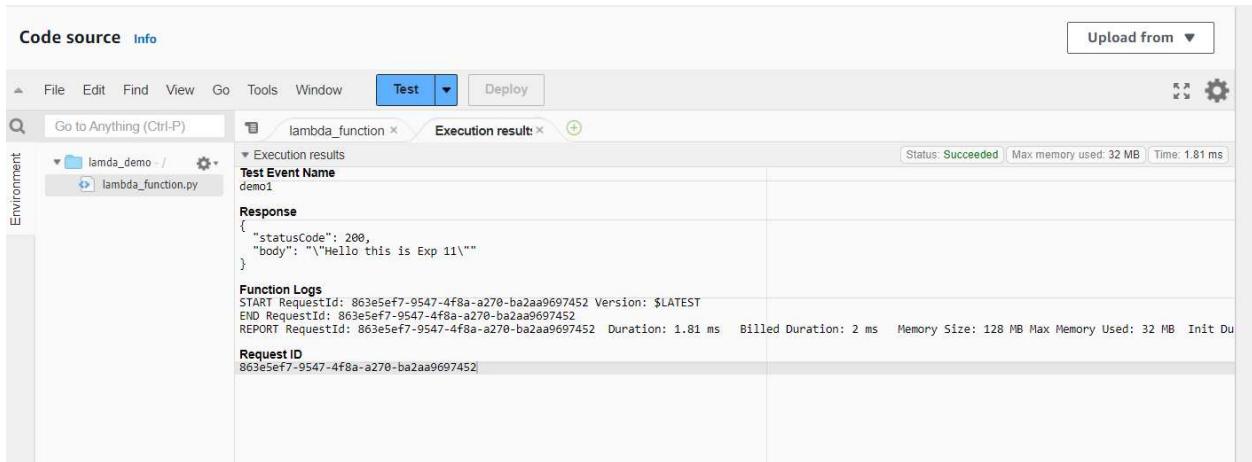
## Step 6: Editing and Deploying the Code

You can modify your Lambda function's code as needed. I updated the code to display a new string. After making changes, press 'Ctrl + S' to save and then click on "Deploy" to apply the updates.



## Step 7: Final Testing

Return to the "Test" tab and execute the test again to observe the output. You should see a status code of 200 along with your string output and function logs confirming a successful deployment.



The screenshot shows the AWS Lambda Test interface. At the top, there are tabs for 'Code source' and 'Info'. Below the tabs is a menu bar with File, Edit, Find, View, Go, Tools, Window, a 'Test' button, and a 'Deploy' button. To the right of the 'Test' button is an 'Upload from' dropdown. On the left, there's a sidebar labeled 'Environment' with a search bar 'Go to Anything (Ctrl-P)' and a folder icon 'lambda\_function'. The main area has a title 'Execution result:' with a close button. Under 'Execution results', it shows 'Test Event Name: demo1'. The 'Response' section contains the following JSON:

```
{ "statusCode": 200, "body": "Hello this is Exp 11\\\" }"
```

The 'Function Logs' section displays the execution details:

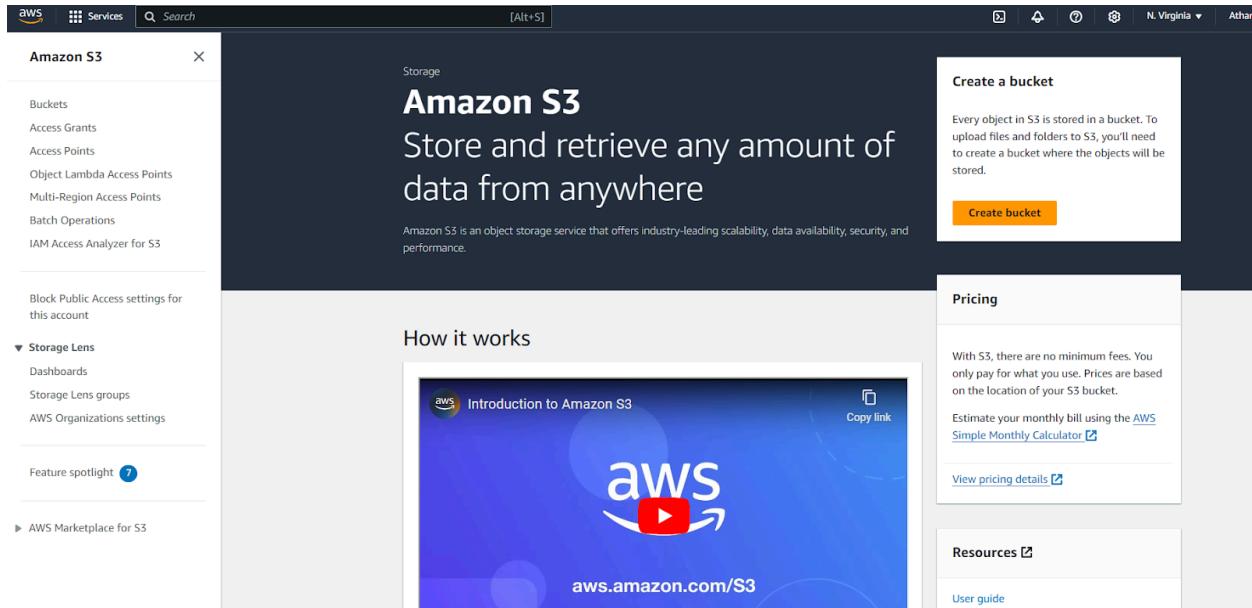
```
START RequestId: 863e5ef7-9547-4f8a-a270-ba2aa9697452 Version: $LATEST  
END RequestId: 863e5ef7-9547-4f8a-a270-ba2aa9697452  
REPORT RequestId: 863e5ef7-9547-4f8a-a270-ba2aa9697452 Duration: 1.81 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 32 MB Init Duration: 0 ms
```

The 'Request ID' is listed as 863e5ef7-9547-4f8a-a270-ba2aa9697452.

**Conclusion:** In this experiment, we successfully created an AWS Lambda function and followed the important steps involved. First, we set up the function using Python and adjusted the timeout setting to 1 second. Then, we created a test event to see how the function works and checked the output to ensure it was correct. We also modified the function's code and redeployed it to see the changes in real-time. So Lambda Function allows you to concentrate on writing code while AWS manages the infrastructure and automatically scales the service as needed. This makes it easier to develop and run applications without worrying about server management.

**Aim:** To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

**Step 1: Create a S3 bucket. 1) Search for S3 bucket in the services search. Then click on create bucket.**



**2) Keep the bucket as a general purpose bucket. Give a name to your bucket.**

### General configuration

AWS Region: US East (N. Virginia) us-east-1

Bucket type:  General purpose

Bucket name: nishantbucket24

Object Ownership:  ACLs disabled (recommended)

**Block Public Access settings for this bucket**

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

**Block all public access**  
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- Block public access to buckets and objects granted through new access control lists (ACLs)**  
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through any access control lists (ACLs)**  
S3 will ignore all ACLs that grant public access to buckets and objects.
- Block public access to buckets and objects granted through new public bucket or access point policies**  
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- Block public and cross-account access to buckets and objects through any public bucket or access point policies**  
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

**4) Keeping all other options the same, click on create. This would create your bucket. Now click on the name of the bucket.**

The screenshot shows the AWS S3 Buckets page. At the top, a green banner indicates that a bucket has been successfully created. Below the banner, the page displays an account snapshot and general purpose buckets. The 'General purpose buckets' tab is selected, showing two buckets: 'elasticbeanstalk-eu-north-1-01092807735' and 's3lamdaexp11'. The 's3lamdaexp11' bucket is highlighted with a blue border. The page includes a search bar, filter buttons for 'Info' and 'All AWS Regions', and buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket'.

Name	AWS Region	IAM Access Analyzer	Creation date
elasticbeanstalk-eu-north-1-01092807735	Europe (Stockholm) eu-north-1	<a href="#">View analyzer for eu-north-1</a>	August 14, 2024, 22:12:26 (UTC+05:30)
s3lamdaexp11	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	October 7, 2024, 09:40:50 (UTC+05:30)

**5) Here, click on upload, then add files. Select any image that you want to upload in the bucket and click on upload.**

Amazon S3 > Buckets > s3lamdaexp11

s3lamdaexp11 [Info](#)

Objects (0) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Actions [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) Actions [Create folder](#) [Upload](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
No objects				
You don't have any objects in this bucket.				
<a href="#">Upload</a>				

Amazon S3 > Buckets > s3lamdaexp11 > Upload

Upload [Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose Add files or Add folder.

Files and folders (1 Total, 990.9 KB)

All files and folders in this table will be uploaded.

Name	Folder
football.jpg	-

Remove [Add files](#) [Add folder](#)

Find by name  < 1 >

Destination [Info](#)

Destination

<s3://s3lamdaexp11>

▶ Destination details  
Bucket settings that impact new objects stored in the specified destination.

▶ Permissions  
Grant public access and access to other AWS accounts.

▶ Properties  
Specify storage class, encryption settings, tags, and more.

6) The image has been uploaded to the bucket.

upload succeeded  
View details below.

Upload: status Close

The information below will no longer be available after you navigate away from this page.

Summary																				
Destination s3://s3lamdaexp11	Succeeded <input checked="" type="radio"/> 1 file, 990.9 KB (100.00%)	Failed <input type="radio"/> 0 files, 0 B (0%)																		
<a href="#">Files and folders</a> <a href="#">Configuration</a>																				
<b>Files and folders</b> (1 Total, 990.9 KB)																				
<table border="1"> <thead> <tr> <th colspan="6">Find by name</th> </tr> <tr> <th>Name</th> <th>Folder</th> <th>Type</th> <th>Size</th> <th>Status</th> <th>Error</th> </tr> </thead> <tbody> <tr> <td>football.jpg</td> <td>-</td> <td>image/jpeg</td> <td>990.9 KB</td> <td><input checked="" type="radio"/> Succeeded</td> <td>-</td> </tr> </tbody> </table>			Find by name						Name	Folder	Type	Size	Status	Error	football.jpg	-	image/jpeg	990.9 KB	<input checked="" type="radio"/> Succeeded	-
Find by name																				
Name	Folder	Type	Size	Status	Error															
football.jpg	-	image/jpeg	990.9 KB	<input checked="" type="radio"/> Succeeded	-															

## Step 2: Configure Lambda function

1) Go to the lambda function you had created before. (Services → Lambda → Click on name of function). Here, click on add trigger

Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

<input checked="" type="radio"/> Author from scratch Start with a simple Hello World example.	<input type="radio"/> Use a blueprint Build a Lambda application from sample code and configuration presets for common use cases.	<input type="radio"/> Container image Select a container image to deploy for your function.
--	--	--

**Basic information**

Function name  
Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

Runtime Info  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.  
 ▼ ↻

Architecture Info  
Choose the instruction set architecture you want for your function code.  
 x86\_64  
 arm64

Permissions Info  
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

► Change default execution role

2) Under trigger configuration, search for S3 and select it.

Lambda > Functions > lamdaexp12

### lamdaexp12

**Function overview** [Info](#)

[Throttle](#) [Copy ARN](#) [Actions ▾](#)

[Export to Application Composer](#) [Download ▾](#)

**Diagram** [Template](#)

**lamdaexp12**

**Layers** (0)

+ Add trigger [+ Add destination](#)

Description

Last modified 22 seconds ago

Function ARN arn:aws:lambda:us-east-1:010928207735:function:lamdaexp12

Function URL [Info](#)

3) Here, select the S3 bucket you created for this experiment. Acknowledge the condition given by AWS. then click on Add. This will add the S3 bucket trigger to your function

Lambda > Add triggers

### Add trigger

**Trigger configuration** [Info](#)

S3 aws asynchronous storage

Lambda > Add triggers

### Add trigger

**Trigger configuration** [Info](#)

S3 aws asynchronous storage

**Bucket**  
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.

[X](#) [C](#)

Bucket region: us-east-1

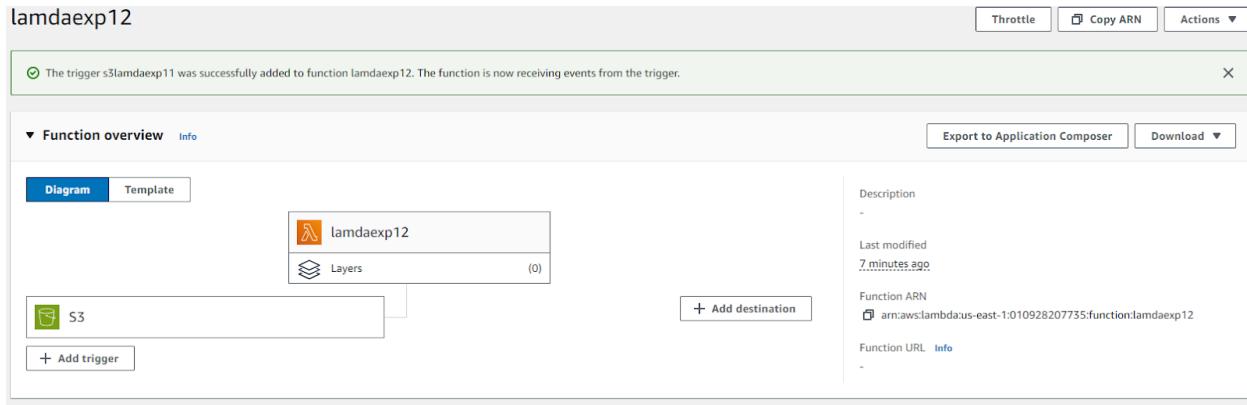
**Event types**  
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events [X](#)

**Prefix - optional**  
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters. Any **special characters** must be URL encoded.  
*e.g. images/*

**Suffix - optional**  
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters. Any **special characters** must be URL encoded.  
*e.g. jpg*

**Recursive invocation**  
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)



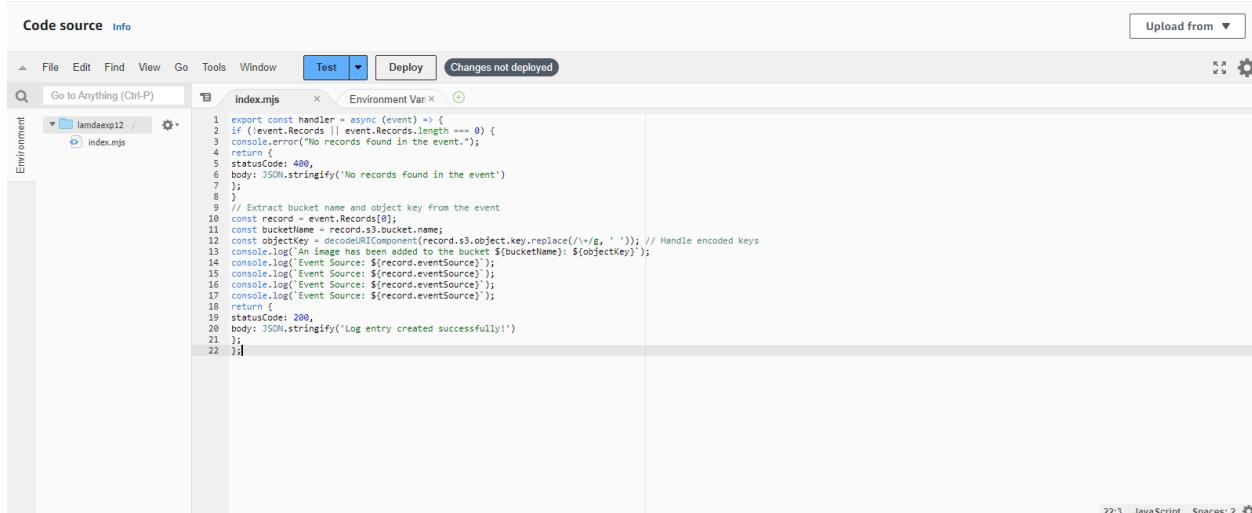
- 4) Scroll down to the code section of the function. Add the following javascript code to the code area by replacing the existing code

```

export const handler = async (event) => {
  if (!event.Records || event.Records.length === 0) {
    console.error("No records found in the event.");
    return {
      statusCode: 400,
      body: JSON.stringify('No records found in the event')
    };
  }
  // Extract bucket name and object key from the event
  const record = event.Records[0];
  const bucketName = record.s3.bucket.name;
  const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys
  console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
  console.log(`Event Source: ${record.eventSource}`);
  console.log(`Event Source: ${record.eventSource}`);
  console.log(`Event Source: ${record.eventSource}`);
  console.log(`Event Source: ${record.eventSource}`);
  return {
    statusCode: 200,
    body: JSON.stringify('Log entry created successfully!')
  };
}

```

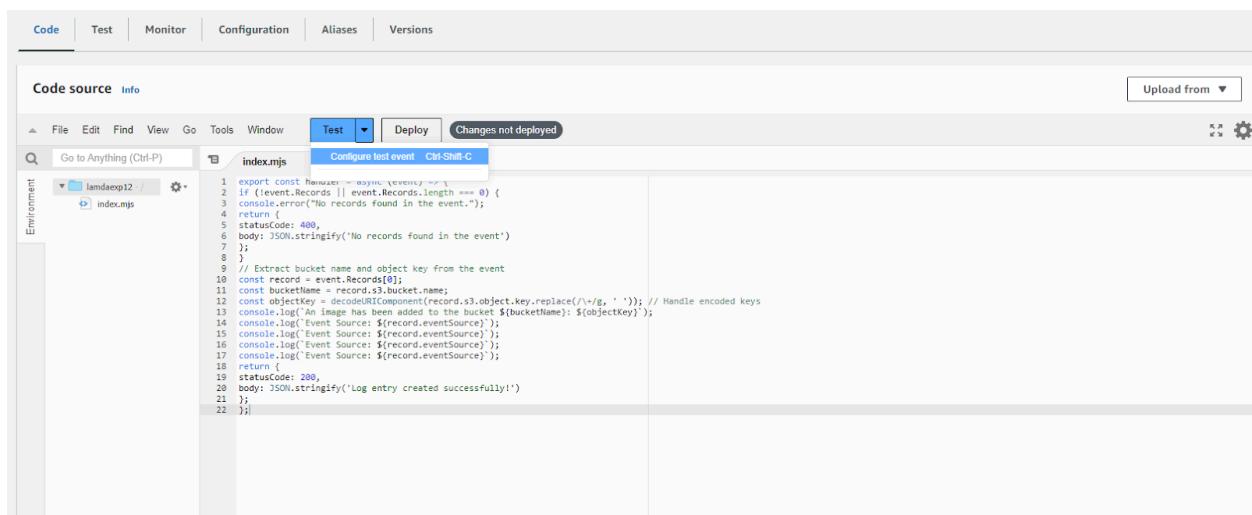
This JSON structure represents an S3 event notification triggered when an object is uploaded to an S3 bucket. It contains details about the event, including the bucket name (example-bucket), the object key (test/key), and metadata like the object's size, the event source (aws:s3), and the event time.



```

Code source Info
File Edit Find View Go Tools Window Test Deploy Changes not deployed
Go Anything (Ctrl-P) index.mjs Environment Var. 
index.mjs
1 exports.handler = async (event) => {
2   if (!event.Records || event.Records.length === 0) {
3     console.error('No records found in the event.');
4     return {
5       statusCode: 400,
6       body: JSON.stringify('No records found in the event')
7     };
8   }
9   // Extract bucket name and object key from the event
10  const record = event.Records[0];
11  const bucketName = record.s3.object.key.replace(/\//g, ' ');
12  const objectKey = decodeURIComponent(record.s3.object.key.replace(/\//g, ' '));
13  console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
14  console.log(`Event Source: ${record.eventSource}`);
15  console.log(`Event Source: ${record.eventSource}`);
16  console.log(`Event Source: ${record.eventSource}`);
17  console.log(`Event Source: ${record.eventSource}`);
18  return {
19    statusCode: 200,
20    body: JSON.stringify('Log entry created successfully!')
21  };
22}

```



```

Code source Info
File Edit Find View Go Tools Window Test Deploy Changes not deployed
Go Anything (Ctrl-P) index.mjs Environment Var. 
index.mjs
Configure test event Ctrl-Shift-C
index.mjs
1 exports.handler = async (event) => {
2   if (!event.Records || event.Records.length === 0) {
3     console.error('No records found in the event.');
4     return {
5       statusCode: 400,
6       body: JSON.stringify('No records found in the event')
7     };
8   }
9   // Extract bucket name and object key from the event
10  const record = event.Records[0];
11  const bucketName = record.s3.object.key;
12  const objectKey = decodeURIComponent(record.s3.object.key.replace(/\//g, ' '));
13  console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
14  console.log(`Event Source: ${record.eventSource}`);
15  console.log(`Event Source: ${record.eventSource}`);
16  console.log(`Event Source: ${record.eventSource}`);
17  console.log(`Event Source: ${record.eventSource}`);
18  return {
19    statusCode: 200,
20    body: JSON.stringify('Log entry created successfully!')
21  };
22}

```

Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

 Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional

hello-world

### Event JSON

[Format JSON](#)

```
1 [ ]  
2 "Records": [  
3 {  
4     "eventVersion": "2.0",  
5     "eventSource": "aws:s3",  
6     "awsRegion": "us-east-1",  
7     "eventTime": "1970-01-01T00:00:00.000Z",  
8     "eventName": "ObjectCreated:Put",  
9     "userIdentity": {  
10         "principalId": "EXAMPLE"  
11     },  
12     "requestParameters": {  
13         "sourceIPAddress": "127.0.0.1"  
14     },  
15     "responseElements": {  
16         "x-amz-request-id": "EXAMPLE123456789",  
17         "x-amz-id-2": "EXAMPLE123/5678abcdefghijklmabaisawesome/mnopqrstuvwxyzABCDEFGHI"  
18     },  
19     "s3": {  
20         "s3SchemaVersion": "1.0",  
21         "configurationId": "testConfigRule",  
22         "bucket": {  
23             "name": "example-bucket",  
24             "ownerIdentity": {  
25                 "principalId": "EXAMPLE"  
26             },  
27             "arn": "arn:aws:s3:::example-bucket"  
28         },  
29         "object": {  
30             "key": "test%2Fkey",  
31         }  
32     }  
33 }
```

1:1 JSON Spaces: 2

[Cancel](#)

[Invoke](#)

[Save](#)

The test event myevent1 was successfully saved.

Function URL [Info](#)

**Step 3: Check the logs**

1) To check the logs explicitly, search for CloudWatch on services and open it in a new tab

The screenshot shows the AWS search interface with the query 'cloud watch' entered in the search bar. The results are categorized under 'Services' and 'Features'.

**Services**

- CloudWatch (Monitor Resources and Applications)
- Athena (Serverless interactive analytics service)
- Amazon EventBridge (Serverless service for building event-driven applications)
- S3 (Scalable Storage in the Cloud)

**Features**

- CloudWatch dashboard (Systems Manager feature)
- Data sources (Athena feature)
- Create a SFTP server (AWS Transfer Family feature)
- Event buses

2) Here, Click on Logs → Log Groups. Select the log that has the lambda function name you just ran.

Name : Aman Yadav

Class : D15C

Roll No : 60

The screenshot shows the AWS CloudWatch Log groups interface. At the top, there is a search bar labeled "Filter log groups or try prefix search" and a checkbox for "Exact match". Below the search bar is a table with one row. The row contains a checkbox, the log group name "/aws/lambda/lamdaexp12", its log class "Standard", its retention policy "Never expire", and a "Configure" button. There are also buttons for "Actions", "View in Logs Insights", "Start tailing", and "Create log group".

### 3) Here, under Log streams, select the log stream you want to check.

The screenshot shows the AWS CloudWatch Log group details interface for the log group "/aws/lambda/lamdaexp12". It displays various metrics and settings for the log group. At the bottom, there is a navigation bar with tabs for "Log streams", "Tags", "Anomaly detection", "Metric filters", "Subscription filters", "Contributor Insights", and "Data protection". Below the navigation bar is a table showing one log stream. The log stream table includes columns for "Log stream", "Last event time", and a timestamp column showing "2024-10-07/[\$LATEST]0bfd52dd5b8a44ab1e15bf46be5f00" and "2024-10-07 04:34:00 (UTC)".

### 4) Here again, we can see that 'An image has been added to the bucket'.

The screenshot shows the AWS CloudWatch Log events interface for the log stream "2024-10-07/[\$LATEST]0bfd52dd5b8a44ab1e15bf46be5f00". The log events table lists several entries, including a message indicating "An image has been added to the bucket example-bucket: test/key". The log events table includes columns for "Timestamp" and "Message".

**Conclusion:**

In this experiment, In addition to demonstrating the integration of AWS Lambda with S3, this experiment showcases the scalability and flexibility of serverless architectures. By leveraging these services, we can build applications that respond in real-time to changes in data, such as the addition of files to S3 buckets, without the need for managing underlying server infrastructure. This not only enhances efficiency but also reduces operational costs, allowing developers to focus on building features rather than maintaining systems. Furthermore, the ability to log and monitor events through CloudWatch opens opportunities for further automation and analytics, paving the way for more complex workflows and data processing solutions.