

**NAME: HIMANSHU NAIK**  
**DIV:D15C**  
**ROLL NO :63**

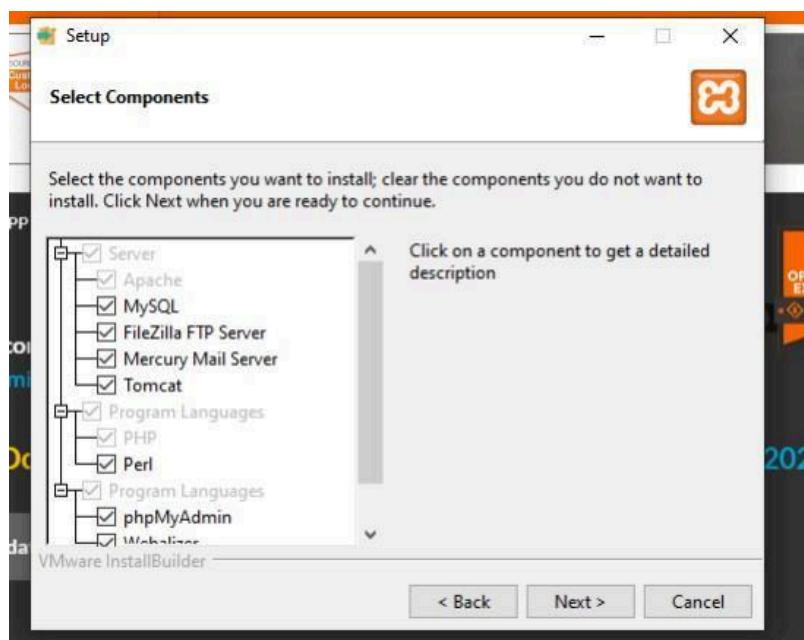
## **Exp 1A :Static Hosting:**

### **1) On local server (XAMPP) Step 1: Install XAMPP**

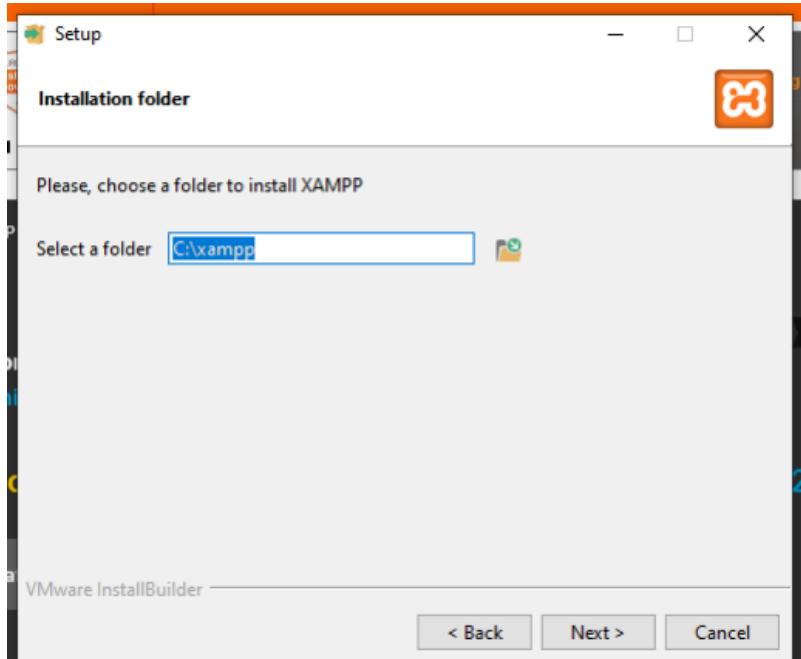
- 1) Select your OS. It will automatically start downloading.**



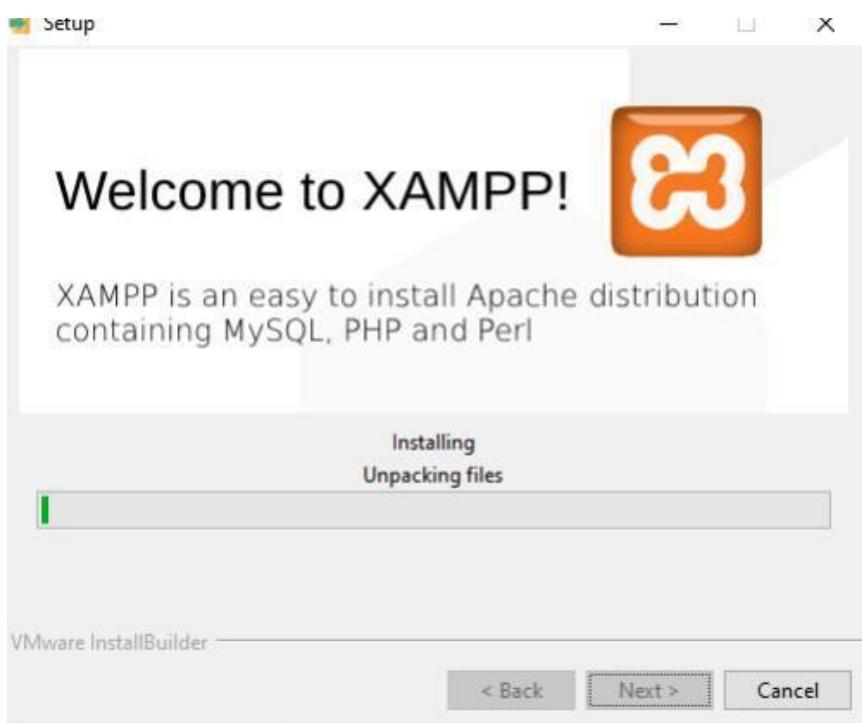
### **2) Open the setup file. Select all the required components and click next**



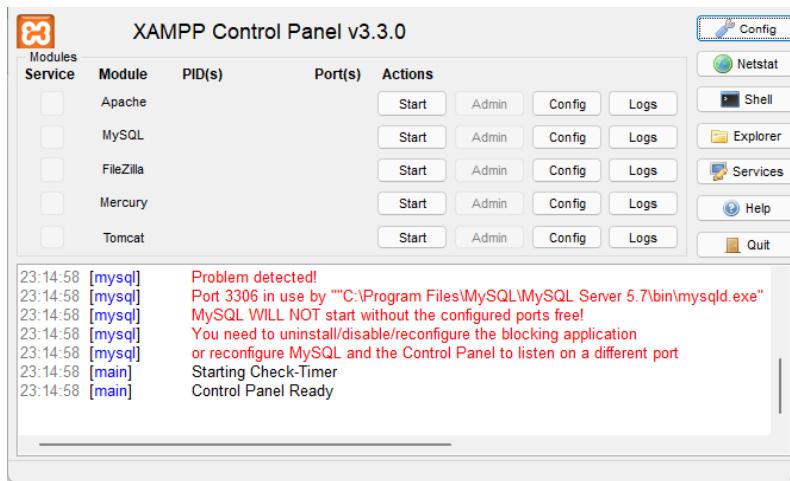
**3) Choose the folder to install XAMPP in. Make sure the folder is empty. Click next**



**4) Select the language, click next. XAMPP starts to install**



**Open XAMPP Control Panel, start the Apache service (Required) and mySQL service**



**Go in the browser and type localhost and hit enter you will get to see the file you created like in this case index1.php**

## Index of /php-practice

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<a href="#">Parent Directory</a>		-	
<a href="#">index1.php</a>	2024-09-20 00:14	0	

*Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12 Server at localhost Port 80*

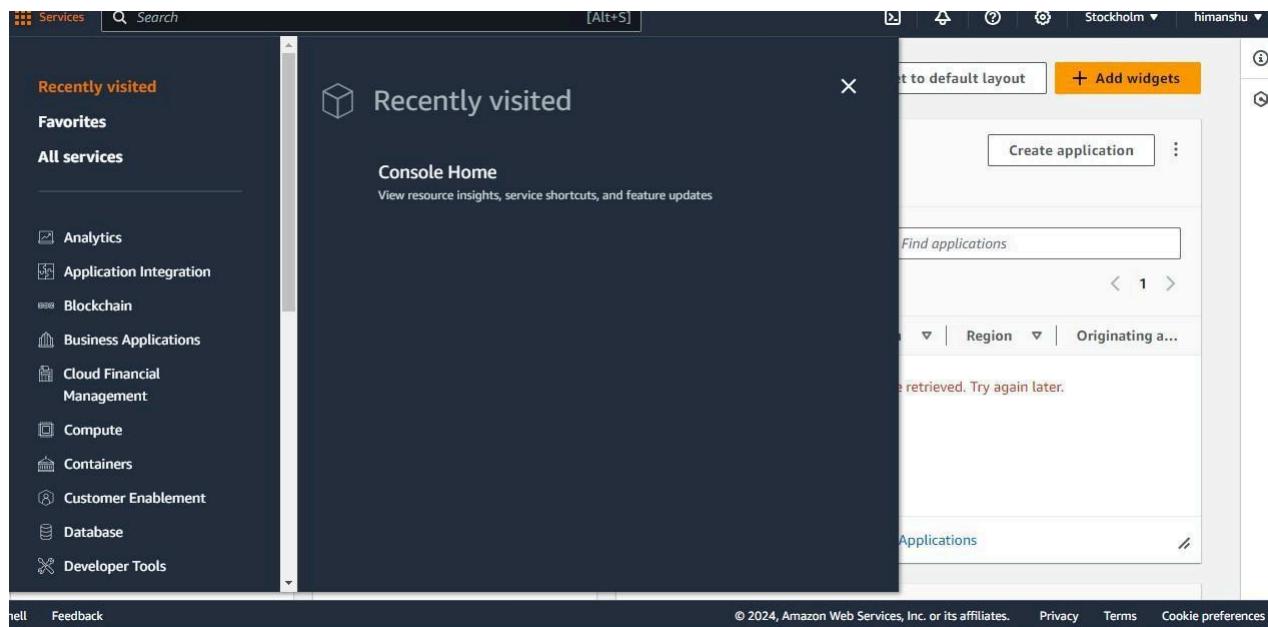
**Open your web browser. Type localhost/YOUR\_FILENAME.php. This will open your website on your browser**

H , Himanshu naik from d15c

## Hosting a static website on Amazon S3

After creating the bucket on AWS add the file in it that is to be hosted and configure the visibility as public and add /\* at the end of the resource key.

### 2) AWS S3 Step 1: Login to your AWS account. Go to services and open S3



### Step 2: Click on Create Bucket

A screenshot of the Amazon S3 landing page. The page has a dark header with the text 'Storage' and 'Amazon S3'. Below the header, there is a large callout box with the text 'Store and retrieve any amount of data from anywhere'. A smaller text below it says 'Amazon S3 is an object storage service that offers industry-leading scalability, data availability, security, and performance.' To the right of this, there is a 'Create a bucket' button. At the bottom, there are sections for 'How it works' (with a link to 'Introduction to Amazon S3') and 'Pricing' (with a note about no minimum fees and a link to the 'AWS Simple Monthly Calculator').

**Step 3: Give a name to your bucket, keeping other options default, scroll down and click on Create Bucket**

Create bucket [Info](#)

Buckets are containers for data stored in S3.

**General configuration**

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

Bucket type [Info](#)

General purpose  
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Directory - New  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

**Step4: Scroll down till you find Static website hosting, click on edit**

Store objects using a write-once-read-many (WORM) model to help you prevent objects from being deleted or overwritten for a fixed amount of time or indefinitely. Object Lock works only in versioned buckets. [Learn more](#)

Object Lock  
Disabled

**Requester pays** [Edit](#)  
When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. [Learn more](#)

Requester pays  
Disabled

**Static website hosting** [Edit](#)  
Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting  
Disabled

**Step 5: Enable static website hosting, in Index document, write the name of your document and in error document, give name as 404.html. Save your changes**

The screenshot shows the 'Edit static website hosting' interface for a bucket named 'ishansbucket'. At the top, there's a breadcrumb navigation: 'Amazon S3 > Buckets > ishansbucket > Edit static website hosting'. Below the navigation, the title 'Edit static website hosting' is displayed with a 'Info' link.

**Static website hosting**  
Use this bucket to host a website or redirect requests. [Learn more](#)

**Static website hosting**

Disable  
 Enable

**Hosting type**

Host a static website  
Use the bucket endpoint as the web address. [Learn more](#)  
 Redirect requests for an object  
Redirect requests to another bucket or domain. [Learn more](#)

**Index document**  
Specify the home or default page of the website.

A callout box provides a note: **For your customers to access content at the website endpoint, you must make all your content publicly readable.** To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#).

**Step 6: Go to Objects tab and click on upload file**

The screenshot shows the 'Objects' tab in the Amazon S3 console. At the top, there's a dashed box with the text: 'Drag and drop files and folders you want to upload here, or choose Add files or Add folder.'

**Files and folders (1 Total, 44.2 KB)**

All files and folders in this table will be uploaded.

< 1 >

<input type="checkbox"/>	Name	Folder	Type

**Step 7: This will take you to the Objects screen. Switch to Properties, scroll down to Static web hosting. There you would find the link (Bucket website endpoint) to your website.**

The screenshot shows the 'Static website hosting' section of the AWS S3 Bucket Properties page. It includes fields for 'Static website hosting' (Enabled), 'Hosting type' (Bucket hosting), and 'Bucket website endpoint'. A note states: 'When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket.' Below this is a 'Learn more' link.

**Step 8: Open the link. It will show a 403 forbidden error screen as the contents of the bucket are not available for the public users. To change this, go to Permissions tab, go to Block public access and click on edit**

## 403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: DM25X6E297MM1SXZ
- HostId: 4fGfAMLunY2ZQ8i9NcuIEDV8CDSLvqSbUj5+gtTqRIGOB5x2rj3/2w4/0mp3CvqJ+IXAPOvq+/4=

**Step 9: Uncheck the Block all public access checkbox and click on save**

### Edit Block public access (bucket settings) Info

#### Block public access (bucket settings)

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 all your S3 buckets and 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 your buckets 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.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid":  
        "PublicReadGetObject",  
      "Effect": "Allow",  
      "Principal": { "AWS": "*" }  
    },  
    {  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::YOUR-BUCKET-NAME-HERE/*"  
    }  
  ]  
}
```

Paste this code snippet in the policy textarea.

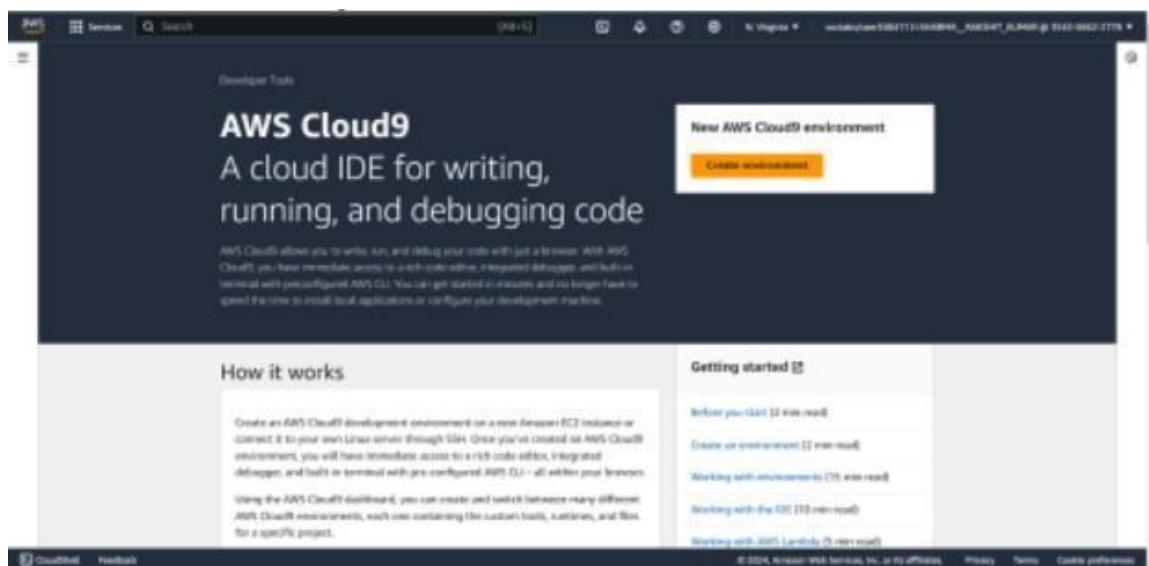
Replace YOUR-BUCKET-NAME-HERE with the name you have given to your bucket.

**Save the changes.**

```
▼ {  
  "Version": "2012-10-17",  
  ▼ "Statement": [  
    {  
      "Sid": "PublicReadGetObject",  
      "Effect": "Allow",  
      ▼ "Principal": {  
          "AWS": "*"  
        },  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::YOUR-BUCKET-NAME-HERE/*"  
    }  
  ]  
}
```

## Experiment 1B: IAM and cloud9

### 1. Open the AWS account and search for Cloud9.



### 2. Enter the name and other required configuration for creating an environment. In network settings, using the AWS system manager gives an error while creating the environment

A screenshot of the "New EC2 instance" creation form on the AWS Cloud9 control console. The top bar shows the URL "https://us-east-1.console.aws.amazon.com/cloud9control/home?region=us-east-1#/create/" and the AWS logo. The main form has sections for "Instance type Info", "Platform Info", and "Timeout".

**Instance type Info**  
The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

t2.micro (1 GiB RAM + 1 vCPU)  
Free-tier eligible. Ideal for educational users and exploration.

t3.small (2 GiB RAM + 2 vCPU)  
Recommended for small web projects.

m5.large (8 GiB RAM + 2 vCPU)  
Recommended for production and most general-purpose development.

Additional instance types  
Explore additional instances to fit your need.

**Platform Info**  
This will be installed on your EC2 instance. We recommend Amazon Linux 2023.

Amazon Linux 2023

**Timeout**  
How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

30 minutes

The screenshot shows the AWS CloudFormation console with the following details:

- Region:** us-east-1
- Stack Name:** vpc-Subnet-AutoScaling-20140411125409862-2176
- Tags - optional:** info
- Tags - optional (info):** A note states: "It's a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs."
- Information:** "The following IAM resources will be created in your account"

  - AWSServiceRoleForAWSCloud9:** AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the IAM console once you no longer have any AWS Cloud9 environments.
  - AWSCloud9SSMAccessRole and AWSCloud9SSMInstanceProfile:** A service role and an instance profile are automatically created if Cloud9 accesses its EC2 instance through AWS Systems Manager. If your environment no longer requires EC2 instances that block incoming traffic, you can delete these roles using the IAM console.

- Create:** A yellow button at the bottom right.
- Errors:** Three error messages are displayed in red boxes:
  - "There was an error creating the IAM resources needed for SSM connection."
  - "You don't have the permission required to perform this operation. Ask your administrator to give you permissions."
  - "User: arn:aws:sts::554258622778:assumed-role/vpc-role/us-east-1:20140411125409862-2176 is not authorized to perform: iam:CreateRole on resource: arn:aws:iam::554258622778:role/service-role/AWSSCDriverSSMRole because no identity-based policy allows the iam:CreateRole action."

### 3. Use the Secure Shell option in Network settings.

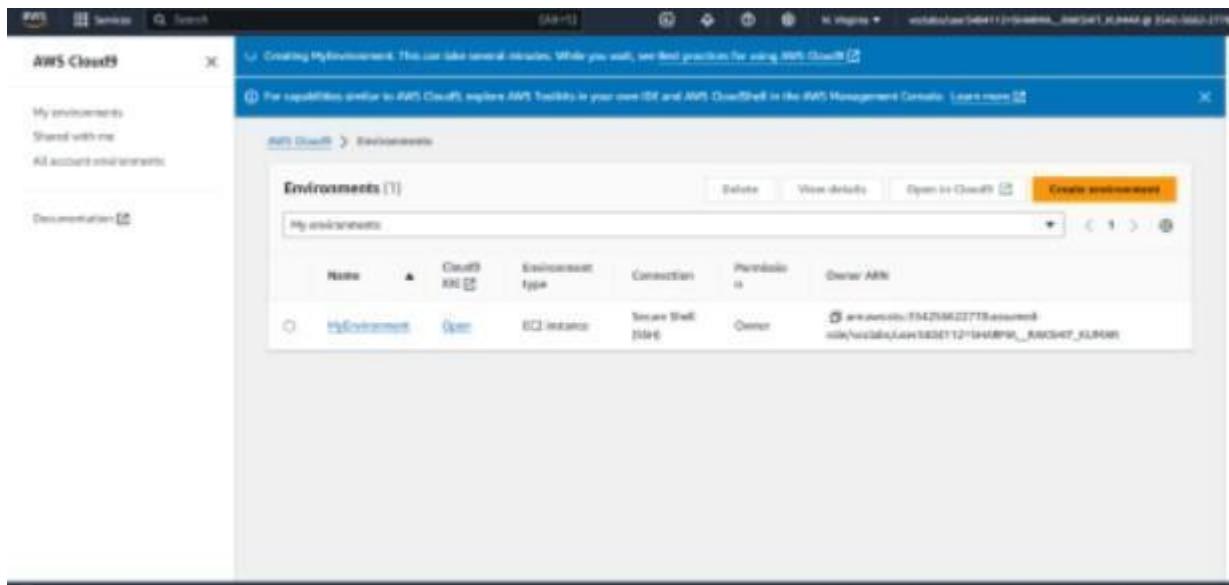
The screenshot shows the AWS CloudFormation console with the following details:

- Region:** us-east-1
- Stack Name:** vpc-Subnet-AutoScaling-20140411125409862-2176
- Network settings:** info
- Connection:** Shows the current connection type: "Secure Shell (SSH)" (selected) and "AWS Systems Manager (SSM)".
- Tags - optional:** info
- Tags - optional (info):** A note states: "It's a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs."
- Information:** "The following IAM resources will be created in your account"

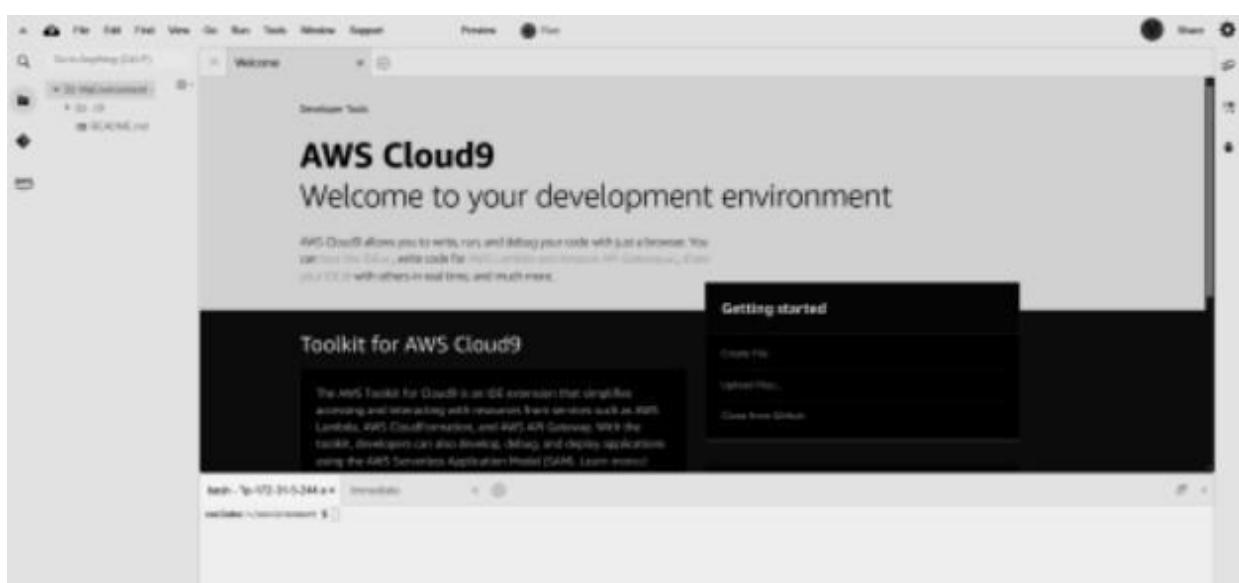
  - AWSServiceRoleForAWSCloud9:** AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the IAM console once you no longer have any AWS Cloud9 environments.

- Create:** A yellow button at the bottom right.

**4. Once the configuration is complete, click on create environment to create a Cloud9 environment.**



**5. Cloud9 Environment.is opened when u click on the environment name**



## IAM user creation steps

1. Open the aws account and search for IAM in service.

The screenshot shows the AWS IAM Dashboard. On the left, a sidebar lists navigation options under 'Access management' (User groups, Users, Roles, Policies, Identity providers, Account settings) and 'Access reports' (Access Analyzer, External access, Unused access). The main area displays 'Security recommendations' with two items: 'Add MFA for root user' (status: 'Root user has no active access keys') and 'Root user has no active access keys' (status: 'Using access keys attached to an IAM user instead of the root user improves security'). Below this is a summary of 'IAM resources' with counts: User groups (1), Users (1), Roles (6), Policies (1), and Identity providers (0).

2. Select the users option from the left panel and click on create user button.Give the user name

The screenshot shows the 'Specify user details' step of the IAM user creation wizard. The left sidebar shows steps: Step 1 (Specify user details), Step 2 (Set permissions), and Step 3 (Review and create). The main form is titled 'Specify user details' and contains a 'User details' section. It includes a 'User name' field with 'sample' entered, a note about character limits, and a checkbox for 'Provide user access to the AWS Management Console - optional'. A note at the bottom explains that if access is granted via IAM Identity Center, access keys won't be generated. At the bottom right are 'Cancel' and 'Next' buttons.

### 3. Click the add user option if you don't have an existing user group

Maximum 128 characters. Use alphanumeric and '+,-,@,\_' characters.

The screenshot shows the 'Permissions policies (951)' page in the AWS IAM console. A search bar and a 'Filter by Type' dropdown are at the top. Below is a table with columns: Policy name, Type, Use..., and Description. Three entries for 'AdministratorAccess' are listed under the 'AWS managed' type. At the bottom right are 'Cancel' and 'Create user group' buttons.

Policy name	Type	Use...	Description
AdministratorAccess	AWS managed	Permis...	Provides full access to AWS services
AdministratorAcce...	AWS managed	None	Grants account administrative perm
AdministratorAcce...	AWS managed	None	Grants account administrative perm

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

The screenshot shows the 'Review and create' step of the user group creation wizard. It includes options for 'Add user to group', 'Copy permissions', and 'Attach policies directly'. Below is a table for 'User groups (1/1)' showing one group named 'MSBCLLOUD9' created on '2024-07-29 (Now)'. At the bottom is a note about setting a 'permissions boundary - optional'.

Group name	Users	Attached policies	Created
MSBCLLOUD9	0	-	2024-07-29 (Now)

password

Permissions summary		
Name	Type	Used as
<a href="#">IAMUserChangePassword</a>	AWS managed	Permissions policy
<a href="#">MSBCLOUD9</a>	Group	Permissions group

**Tags - optional.**  
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

**Go back to the cloud9 environment. Click on the share this environment option so as to allow other collaborators to access your environment. Include your newly made IAM user in this environment and enable Read/Write permissions for it.**

Share this environment

Links to share

Environment: <https://ap-south-1.console.aws.amazon.com/cloud9/ide/155667b0310449d2818d9d8673a61e0a?region=ap-south-1>

Application: 65.0.138.120

To make your application accessible from the internet, please follow [our documentation](#).

Who has access

- ReadWrite
  - You (online)
  - sahilmotiramani (offline)

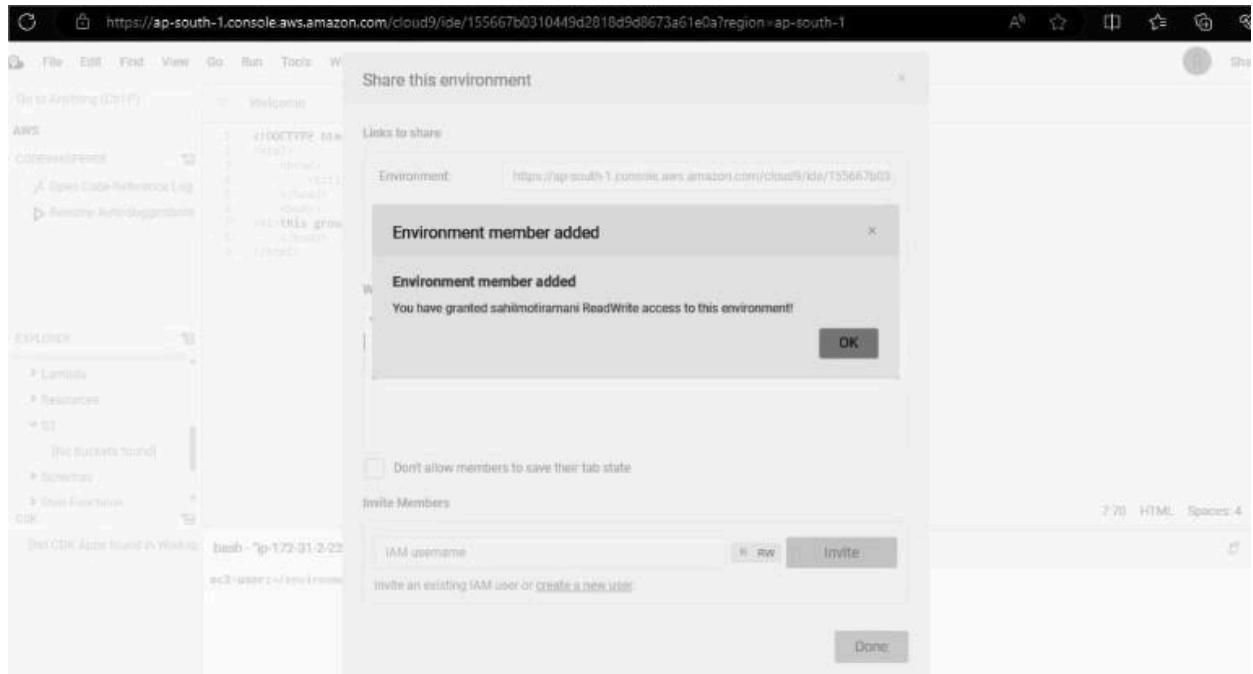
Don't allow members to save their tab state

Invite Members

bhushanmalpani  R  RW

Invite an existing IAM user or [create a new user](#)

Done



**Further, we are supposed to login from another browser using the credentials of the IAM user, 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.**

1) Go to services and choose elastic Beanstalk. following page will appear

The screenshot shows the Amazon Elastic Beanstalk landing page. It features a dark header with the service name and a sub-header 'End-to-end web application management.' Below this, a paragraph explains the service's purpose: 'Amazon Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker or familiar servers such as Apache, Nginx, Passenger, and IIS.' On the right side, there are two main call-to-action boxes: 'Get started' (with a 'Create application' button) and 'Pricing'. The 'Get started' section contains a brief description of how the service handles deployment, capacity provisioning, load balancing, and automatic scaling.

2) Configure the environment. Give the application name, check domain availability and choose PHP as platform.Then click next.

This screenshot shows the 'Configure environment' step of the AWS Elastic Beanstalk setup wizard. It includes three main sections: 'Environment tier', 'Application information', and 'Environment information'. In the 'Environment tier' section, 'Web server environment' is selected. In the 'Application information' section, the 'Application name' is set to 'sample'. In the 'Environment information' section, the environment name is chosen to be 'sample-env'. The 'Domain name' field is empty, and the 'Region' dropdown shows 'US East (N. Virginia)'.

Environment tier	Web server environment
Application name	sample
Environment information	sample-env

### 3) Configure the service access

Environment description  
created a new environment and checked domain availability.

**Platform info**

Platform type  
 Managed platform Platforms published and maintained by Amazon Elastic Beanstalk. Learn more [\[?\]](#)  
 Custom platform Platforms created and owned by you. This option is unavailable if you have no platforms.

Platform  
PHP

Platform branch  
PHP 8.3 running on 64bit Amazon Linux 2023

### 4) Choose one of the available VPC and instance subnet. Click next

Set up networking, database, and tags - *optional* [\[info\]](#)

**Virtual Private Cloud (VPC)**

VPC  
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console. [Learn more \[?\]](#)

vpc-0a4b2134962ed0c59 | (172.51.0.0/16)

Create custom VPC [\[?\]](#)

**Instance settings**  
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. [Learn more \[?\]](#)

Public IP address  
Assign a public IP address to the Amazon EC2 instances in your environment.  
 Activated

**Instance subnets**

Filter instance subnets

Availability Zone	Subnet	CIDR	Name
us-east-1d	subnet-04a4cfde8...	172.31.0.0/20	

## 5) Configure instance traffic and scaling. Keep all the options as default

Configure instance traffic and scaling - optional [Info](#)

**Instances** [Info](#)  
Configure the Amazon EC2 instances that run your application.

**Root volume (boot device)**

Root volume type:

Size: The number of gigabytes of the root volume attached to each instance.  
8 GB

IOPS: Input/output operations per second for a provisioned IOPS (SSD) volume.  
100 IOPS

Throughput: The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance.  
125 MiB/s

**Amazon CloudWatch monitoring**  
The time interval between when metrics are reported from the EC2 instances.

Monitoring interval:

**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  t3.small

**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-083f545ce1a73bf03

**Availability Zones**  
Number of Availability Zones (AZs) to use:  
Any

**Placement**  
Specify Availability Zones (AZs) to use:  
Choose Availability Zones (AZs)

**Scaling cooldown**  
360 seconds

[Cancel](#) [Skip to review](#) [Previous](#) [Next](#)

## 6) Configure updates, monitoring, and logging. Keep everything as default and click next.

Configure updates, monitoring, and logging - optional [Info](#)

**Monitoring** [Info](#)

**Health reporting**  
Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The EnvironmentHealth custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see [Amazon CloudWatch Pricing](#)

System:  Enhanced  Basic

CloudWatch Custom Metrics - Instance:

CloudWatch Custom Metrics - Environment:

**Health event streaming to CloudWatch Logs**  
Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

Log streaming:  Activated (standard CloudWatch charges apply.)

Retention:  1 year

**Instance log streaming to CloudWatch logs**

Configure the instances in your environment to stream logs to CloudWatch logs. You can set the retention to up to 10 years and configure Elastic Beanstalk to delete the logs when you terminate your environment. [Learn more](#)

**Log streaming**  
(Standard CloudWatch charges apply.)

Activated

**Retention**

7 days

**Lifecycle**

Keep logs after terminating environment

**Environment properties**

The following properties are passed in the application as environment properties. [Learn more](#)

No environment properties have been configured.

[Add environment property](#)

Cancel Previous Next

## 7) Click submit.

false	false							
<b>Platform software</b>								
Lifecycle	Log streaming	Allow URL fopen						
false	Deactivated	On						
Display errors	Document root	Max execution time						
Off	=	60						
Memory limit	Zlib output compression	Proxy server						
256M	Off	nginx						
Logs retention	Rotate logs	Update level						
7 days	Deactivated	minor						
X-Ray enabled								
Deactivated								
<b>Environment properties</b>								
<table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td colspan="2">No environment properties</td> </tr> <tr> <td colspan="2">There are no environment properties defined</td> </tr> </tbody> </table>			Key	Value	No environment properties		There are no environment properties defined	
Key	Value							
No environment properties								
There are no environment properties defined								

Cancel Previous Submit

## 8) Environment has been created successfully.

The screenshot shows the AWS Lambda console with the following details:

**Environment overview**

- Health: Warning
- Domain: <https://us-east-1.execute-api.amazonaws.com>
- Environment ID: e-0MKM9R
- Application name: sampel

**Platform**

- Platform: PHP 8.3 running on 64bit Amazon Linux 2023.4.3.1
- Running version: -
- Platform state: Supported

**Events [10] Info**

Filter events by text, property or value

Time	Type	Details
August 9, 2024 21:35:13 (UTC+5:30)	WARN	Environment health has transitioned from Pending to Warning. Initialization completed 27 seconds ago and took 2 minutes. There are no instances. Usable to assume role 'arn:aws:iam::996474913977:role/EMR_EC2_DefaultRole'. Verify that the role exists and is configured correctly.

## 9) Deploy something on the recently created environment

The screenshot shows the 'Upload and deploy' dialog and the environment details page.

**Upload and deploy**

To deploy a previous version, go to the [Application versions page](#)

**Upload application**

Choose file

File name: Screenshot 2023-11-10 185456.png  
File must be less than 500MB max file size.

**Version label**  
Unique name for this version of your application code.  
sampel-version-1

Current number of EC2 instances: 1

**Cancel** **Deploy**

**Environment overview**

Health: Warning (e-0MKM9R)

Domain: <https://us-east-1.execute-api.amazonaws.com>

Environment ID: e-0MKM9R

Application name: sampel

**Platform**

Platform: PHP 8.3 running on 64bit Amazon Linux 2023.4.3.1

Running version: -

Platform state: Supported

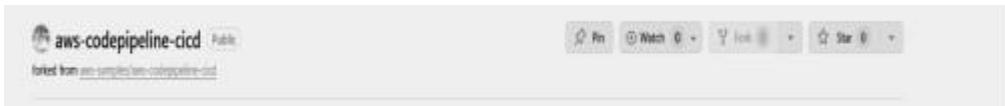
**Events [11] Info**

Filter events by text, property or value

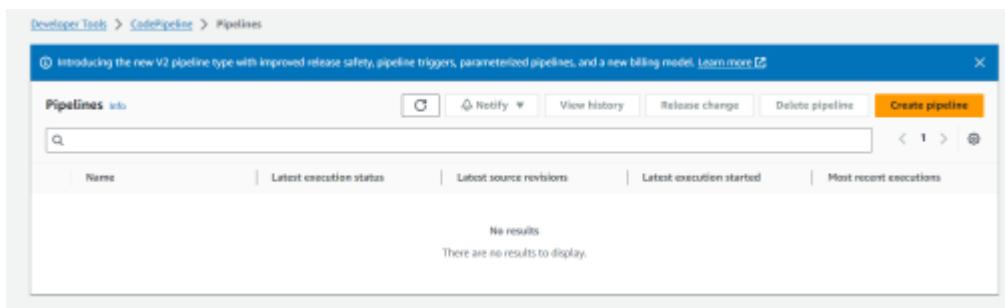
Time	Type	Details
August 9, 2024 21:25:22 (UTC+5:30)	WARN	Service role 'arn:aws:iam::996474913977:role/EMR_EC2_DefaultRole' is missing permissions required to check for

# Pipeline Creation:

- 1) Fork a git-hub repository. This forked repository will act as source for your code pipeline.



- 2) Go to developer tools and select CodePipeline and create a new pipeline



- 3) Create a pipeline:



## Choose pipeline settings Info

Step 1 of 5

### Pipeline settings

#### Pipeline name

Enter the pipeline name. You cannot edit the pipeline name after it is created.

firstpipeline

No more than 100 characters

#### Pipeline type

- i You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

#### Execution mode

Choose the execution mode for your pipeline. This determines how the pipeline is run.

Superseded

A more recent execution can overtake an older one. This is the default.

Queued (Pipeline type V2 required)

Executions are processed one by one in the order that they are queued.

Parallel (Pipeline type V2 required)

## Add source stage Info

Step 2 of 5

### Source

#### Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 1) ▼

Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.

[Connect to GitHub](#)

i **The GitHub (Version 1) action is not recommended**

The selected action uses OAuth apps to access your GitHub repository. This is no longer the recommended method. Instead, choose the GitHub (Version 2) action to access your repository by creating a connection. Connections use GitHub Apps to manage authentication and can be shared with other resources. [Learn more](#)

#### Change detection options

Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

GitHub webhooks (recommended)

Use webhooks in GitHub to automatically start my pipeline when a change occurs.

AWS CodePipeline

Use AWS CodePipeline to check periodically for changes.

## Add source stage

Step 2 of 5

### Source

#### Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 1)

Grant AWS CodePipeline access to your GitHub repository. This allows AWS CodePipeline to upload commits from GitHub to your pipeline.

Connected

You have successfully configured the action with the provider.

**The GitHub (Version 1) action is not recommended**

The selected action uses OAuth apps to access your GitHub repository. This is no longer the recommended method. Instead, choose the GitHub (Version 2) action to access your repository by creating a connection. Connections use GitHub Apps to manage authentication and can be shared with other resources. [Learn more](#)

#### Repository

pixelbypixels/aws-codepipeline-cicd

#### Branch

main

#### Change detection options

Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

GitHub webhooks (recommended)

Use webhooks in GitHub to automatically start my

AWS CodePipeline

Use AWS CodePipeline to check periodically for changes

## 5) Go to the deploy stage and ensure the following settings

### Add deploy stage

Step 4 of 5

**You cannot skip this stage**

Pipelines must have at least two stages. Your second stage must be either a build or deployment stage. Choose a provider for either the build stage or deployment stage.

### Deploy

#### Deploy provider

Create new AWS Lambda instances. Choose the provider, and then provide the configuration details for that provider.

AWS Elastic Beanstalk

#### Region

US East (N. Virginia)

#### Input artifacts

Choose an input artifact for this action. [Learn more](#)

No more than 100 characters.

#### Application name

Choose an application that you have already created in the AWS Elastic Beanstalk console. Or create an application in the AWS Elastic Beanstalk console and then return to this task.

test\_application

#### Environment name

Choose an environment that you have already created in the AWS Elastic Beanstalk console. Or create an environment in the AWS Elastic Beanstalk console and then return to this task.

test-application-env

Configure automatic rollback on stage failure

## 6) review the pipeline settings.

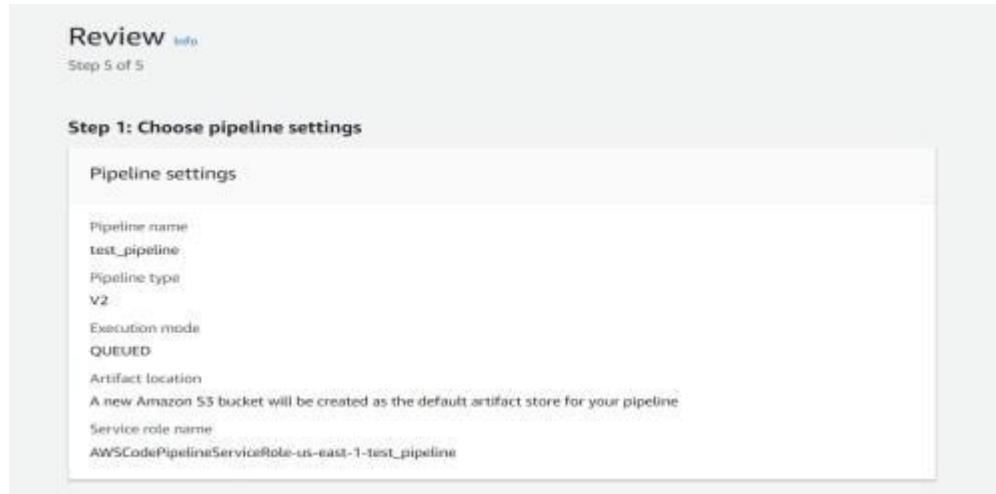
**Review** info

Step 5 of 5

**Step 1: Choose pipeline settings**

**Pipeline settings**

Pipeline name: test\_pipeline  
Pipeline type: V2  
Execution mode: QUEUED  
Artifact location: A new Amazon S3 bucket will be created as the default artifact store for your pipeline.  
Service role name: AWSCodePipelineServiceRole-us-east-1-test\_pipeline



**7) Then go ahead and check the URL provided in the EBS environment.**

**Success**  
Congratulations! The pipeline firstpipeline has been created.

Developer Tools > [Edit Pipeline](#) > [Pipeline](#) > [firstpipeline](#)

**firstpipeline**

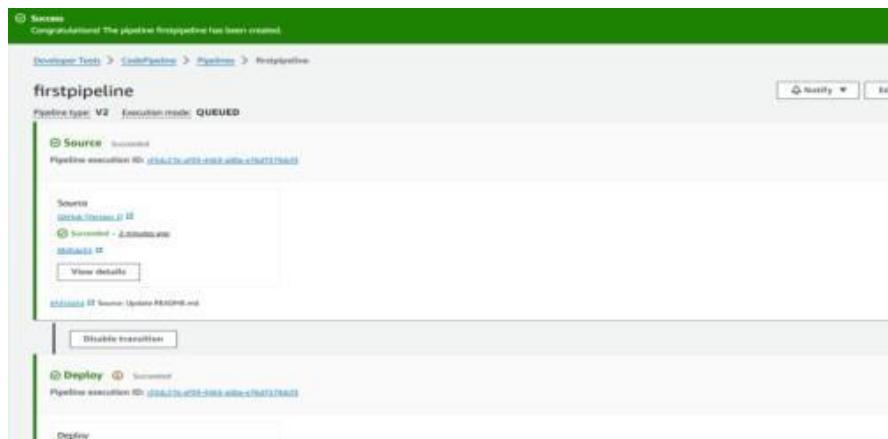
Pipeline type: V2 Execution mode: QUEUED

**Source** aws\_lambda  
Pipeline execution ID: 054d7b4f93-4003-400a-a5a0a19d725a03

Source  
[aws\\_lambda](#) 1  
[Succeeded](#) - 2.00000 sec  
[Details](#) 1  
[View details](#)

**Deploy** aws\_lambda  
Pipeline execution ID: 054d7b4f93-4003-400a-a5a0a19d725a03

[Deploy](#)



**8) Go to the repository and make the changes in the index.html file and commit them**

**Commit changes**

**Commit message**

**Extended description**  
Add an optional extended description.

Commit directly to the master branch  
 Create a new branch for this commit and start a pull request Learn more about pull requests

[Cancel](#) [Commit changes](#)



**9) To view the changes made, ensure they are committed and visible in the source panel in real time. After confirming that the deployment section indicates success, refresh the URL to see the updates reflected on your site or application.**

# Experiment:

## 3

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

### 1. Create 3 EC2 Ubuntu Instances on AWS. (Name 1 as Master, the other 2 as worker-1 and worker-2)

Instances (4) <small>Info</small>		Last updated 2 minutes ago		Connect	Instance state	Actions	Launch instances	
		<input type="text"/> Find Instance by attribute or tag (case-sensitive)		All states		< 1 >		
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability	
<input type="checkbox"/>	aws-cloud9-fir...	i-08f0cffbbb8e65871	Stopped		t2.micro	-		us-east-1d
<input type="checkbox"/>	master	i-04d752dc5a6094386	Running		t2.micro	2/2 checks passed		us-east-1d
<input type="checkbox"/>	worker-2	i-06d89cce0e15b9521	Running		t2.micro	2/2 checks passed		us-east-1d
<input type="checkbox"/>	worker-1	i-00e2c3d55f62382f0	Running		t2.micro	2/2 checks passed		us-east-1d

### 2. Edit the Security Group Inbound Rules to allow SSH and do it for all the three machines

```
$ ssh -i "server.pem" ec2-user@ec2-54-174-206-93.compute-1.amazonaws.com
The authenticity of host 'ec2-54-174-206-93.compute-1.amazonaws.com (54.174.206.93)' can't be established.
ED25519 key fingerprint is SHA256:T+tsGyII5gAvUVjeAZ7GjDIWXHOaI4EPF5g5oICrkoQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-174-206-93.compute-1.amazonaws.com' (ED25519)
to the list of known hosts.

#_
~\_ ####_
~~\_\#####\
~~ \###|
~~ \#/ .___. https://aws.amazon.com/linux/amazon-linux-2023
```

**3. From now on, until mentioned, perform these steps on all 3 machines. Install Docker for all the 3 machines**

```
[root@ip-172-31-90-172 ec2-user]# yum install docker -y
Last metadata expiration check: 0:21:16 ago on Fri Aug 30 04:01:12 2024.
Dependencies resolved.
```

Package	Architecture	Version
<b>Installing:</b>		
docker	x86_64	25.0.6-1.amzn2023.0.1
<b>Installing dependencies:</b>		
containerd	x86_64	1.7.20-1.amzn2023.0.1
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2
libcgroup	x86_64	3.0-1.amzn2023.0.1
libnetfilter_conntrack	x86_64	1.0.8-2.amzn2023.0.2
libnfnlink	x86_64	1.0.1-19.amzn2023.0.2
libnftnl	x86_64	1.2.2-2.amzn2023.0.2
pigz	x86_64	2.5-1.amzn2023.0.3

**Start the docker by running the command systemctl start docker in the terminal of all the ec2 instances.**

```
Complete!
[root@ip-172-31-82-133 ec2-user]# systemctl start docker
[root@ip-172-31-82-133 ec2-user]# █
```

## Install the kubernetes

**On all 3 machines by searching for kubeadm and clicking on install kubernetes.**

**Select the red hat based distribution. This process will automatically disable SELinux before configuring kubelet so no need to run it separately in the terminal.**

Debian-based distributions

Red Hat-based distributions

Without a package manager

#### 1. Set SELinux to permissive mode:

These instructions are for Kubernetes 1.31.

```
# Set SELinux in permissive mode (effectively disabling it)
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/
```

**Copy the below script, to install kubernetes we need a kubernetes repo so this script helps us in getting that and paste it in the terminal.**

```
# This overwrites any existing configuration in /etc/yum.repos.d/kubernetes.repo
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable/:v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable/:v1.31/rpm/repo/repodata/repomd.xml
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

Installed:
containerd-1.7.20-1.amzn2023.0.1.x86_64      docker-25.0.6-1.amzn2023.0.1.x86_64      iptables-libs-1
iptables-nft-1.8.8-3.amzn2023.0.2.x86_64    libcgroup-3.0-1.amzn2023.0.1.x86_64     libnetfilter_cc
libnfnetwork-1.0.1-19.amzn2023.0.2.x86_64   libnftnl-1.2.2-2.amzn2023.0.2.x86_64    pigz-2.5-1.amzn
runc-1.11.1-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-90-172 ec2-user]# systemctl start docker
[root@ip-172-31-90-172 ec2-user]# sudo su
[root@ip-172-31-90-172 ec2-user]# yum repolist
repo id                                repo name
amazonlinux                             Amazon Linux 2023 repository
```

**Run the command yum repolist to check whether the kubernetes repo has installed or not if successful installed then you can see a repo named as kubernetes**

```
[root@ip-172-31-90-172 ec2-user]# yum repolist
repo id                                repo name
amazonlinux                             Amazon Linux 2023 repository
kernel-livepatch                         Amazon Linux 2023 Kernel Livepatch repository
kubernetes                               Kubernetes
[root@ip-172-31-90-172 ec2-user]# ■
```

**Do the above steps for all the instances i.e for worker-1 and worker-2**

**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
```

```
[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

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 172.31.12.28:6443 --token 4bqwb8.lua2ud01lr02uu55 \
  --discovery-token-ca-cert-hash sha256:b4edc7948be9bca50767f623b58e0612feedc144a7364f95be8dbd8c4614a169
```

**Copy the join command and keep it in a notepad, we'll need it later. Copy the mkdir and chown commands from the top and execute them**

```
[ec2-user@ip-172.31.12.28 docker]$ 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 file as mentioned in the code. kubectl apply -f**

**<https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml>**

```
[ec2-user@ip-172.31.12.28 docker]$ kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

**Check the created pod using this command Now, keep a watch on all nodes using the following command - watch kubectl get nodes 6. Perform this only on the worker machines Run the following command sudo yum install iproute-tc -y sudo systemctl enable kubelet sudo systemctl restart kubelet Check the status of the pods using the following command This command will show the status of all the pods. kubectl get pods -n kube-system**

**Following command will show the status of the pod named daemonset.**

**kubectl get daemonset -n kube-system**

```
[ec2-user@ip-172.31.12.28 docker]$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS   AGE
coredns-55cb5b8774-fx12f          1/1    Running   0          100s
coredns-55cb5b8774-xn14v          1/1    Running   0          100s
etcd-ip-172.31.12.28.ec2.internal 1/1    Running   0          75s
kube-apiserver-ip-172.31.12.28.ec2.internal 1/1    Running   1          2m
kube-controller-manager-ip-172.31.12.28.ec2.internal 0/1 CrashLoopBackOff 1 70s
kube-proxy-4dv8m                  1/1    Running   2          100s
kube-scheduler-ip-172.31.12.28.ec2.internal 1/1    Running   1          76s
```

```
[ec2-user@ip-172.31.12.28 docker]$ kubectl get daemonset -n kube-system
NAME      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
kube-proxy 1         1         1         1           1           kubernetes.io/os=linux 3m
```

**Conclusion:.**

The process began with the creation of instances and configuration of settings to begin the communication. Docker was installed on all machines followed by the installation of Kubernetes components and the necessary repositories. The Master node was initialized with the `kubeadm init` command, and a plugin called Flannel was deployed to enable pod communication. Performing correct commands on the Worker nodes ensured they joined the cluster effectively. Also necessary commands confirmed the status of pods which indicated the proper working. Overall, the experiment provided information about working of the deployment and management of containerized applications in a distributed environment.

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

**Procedure: 1. Creation Of EC2 instance** • Create an EC2 AWS Linux instance on AWS .also edit the Security Group Inbound Rules to allow SSH. then select the t2.micro instance type

The screenshot shows the AWS EC2 Dashboard with the 'Instances' section selected. On the left sidebar, there are links for EC2 Global View, Events, Console-to-Code Preview, Instances, Images, AMIs, and AMI Catalog. The main area displays a summary of resources: Instances (running) 1, Auto Scaling Groups 0, Capacity Reservations 0, Dedicated Hosts 0, Elastic IPs 0, Instances 0, Key pairs 0, Leaf Load Balancers 0, Placement groups 0, Security groups 0, Snapshots 0, and Volumes 4. Below this, the 'Launch instance' section is visible, showing a note that instances will launch in the US East (N. Virginia) Region. A prominent orange 'Launch instance' button is at the top of this section. To the right, the 'Service health' status is shown as 'US East (N. Virginia)' with a green status indicator. The URL in the browser bar is EC2 > Instances > Launch an instance.

**Launch an instance**

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

**Name and tags**

Name:  Add additional tags

**Application and OS Images (Amazon Machine Image)**

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

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

Recent: Quick Start

Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux

Browse more AMIs

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI  
ami-0f373e6ff9c85 [64-bit (x86), uefi-preferred] / ami-08d427a5519fa06a [64-bit (x86), uefi]  
Virtualization: hvm EBS enabled: true Root device type: ebs

Free tier eligible

The screenshot shows the AWS EC2 instance creation process. On the left, under 'Network settings', it shows a VPC (vpc-0581e49e607677b63) and subnet (No preference [Default subnet in any availability zone]). It includes options for creating or selecting a security group, with a note about launching a new security group named 'launch-wizard-8'. On the right, the 'Summary' section shows 1 instance, AMI (Amazon Linux 2023.5.2...), instance type (t2.micro), and storage (1 volume(s) - 8 GiB). A 'Free Tier' message is displayed, indicating 750 hours of t2.micro usage included in the first year. At the bottom are 'Cancel' and 'Launch instance' buttons.

Instances (6) <a href="#">Info</a>		Last updated	<a href="#">Connect</a>	<a href="#">Instance state</a>	<a href="#">Actions</a>	<a href="#">Launch instances</a>
		less than a minute ago		All states		
<input type="checkbox"/>	Name ↗	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	worker-1_sb	i-07bed31a706f6f589	<span>Running</span> <a href="#">View details</a> <a href="#">Edit</a>	t2.micro	<span>2/2 checks passed</span> <a href="#">View alarms</a> <a href="#">+</a>	us-east-1a
<input type="checkbox"/>	worker-2_sb	i-05da8301287468d59	<span>Running</span> <a href="#">View details</a> <a href="#">Edit</a>	t2.micro	<span>2/2 checks passed</span> <a href="#">View alarms</a> <a href="#">+</a>	us-east-1f
<input type="checkbox"/>	kuber	i-097d2af538d0ca45a	<span>Pending</span> <a href="#">View details</a> <a href="#">Edit</a>	t2.micro	-	<a href="#">View alarms</a> <a href="#">+</a>

- Thus Kuber named -instance gets created. Then click on Id of that instance then click on connect button you will see this:

The screenshot shows the 'Connect to instance' dialog for instance i-097d2af538d0ca45a (kuber). It includes tabs for 'EC2 Instance Connect', 'Session Manager', 'SSH client', and 'EC2 serial console'. A warning message states: 'Port 22 (SSH) is open to all IPv4 addresses. Port 22 (SSH) is currently open to all IPv4 addresses, indicated by 0.0.0.0/0 in the inbound rule in your security group. For increased security, consider restricting access to only the EC2 Instance Connect service IP addresses for your Region: 18.206.107.24/29. Learn more.' Connection options include 'Connect using EC2 Instance Connect' (selected) and 'Connect using EC2 Instance Connect Endpoint'. The public IPv4 address is listed as 54.211.131.109. The username is set to 'ec2-user'. A note at the bottom says: 'Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.' At the bottom are 'Cancel' and 'Connect' buttons.

Then go into SSH client where you will get this command “keyname.pem”  
ubuntu@ copy it and then connect it and run the following command for establishing connection.(I have entered this command on git bash where i entered in downloads where server.pem is stored then as the key is not accessible hence we need to change its mode using “key name.pem”. Then use the given command for making connections).

```
ADMIN@DESKTOP-LPV2RP5 MINGW64 ~
$ cd Downloads/
ADMIN@DESKTOP-LPV2RP5 MINGW64 ~/Downloads
$ chmod 400 "server.pem"

ADMIN@DESKTOP-LPV2RP5 MINGW64 ~/Downloads
$ ssh -i "server.pem" ec2-user@ec2-3-237-37-35.compute-1.amazonaws.com
The authenticity of host 'ec2-3-237-37-35.compute-1.amazonaws.com (3.237.37.35)' can't be established.
ED25519 key fingerprint is SHA256:uhSLQC4WN+6i3np6iMFwW+vGF8aGkTrm89ryXQ4wT1g.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-237-37-35.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#_
#_ \###_
#_ \###\ Amazon Linux 2023
#_ \###_
#_ \#/ \_> https://aws.amazon.com/linux/amazon-linux-2023
#_ \_>
#_ /m/ \_>
Last login: Sat Sep 14 11:47:26 2024 from 18.206.107.28
[ec2-user@ip-172-31-65-181 ~]$
```

## Installation of Docker 1.

- . For installation of Docker into the machines run the following command: sudo yum install docker -y

```
[ec2-user@ip-172-31-65-181 ~]$ sudo yum install docker -y
Last metadata expiration check: 0:08:49 ago on Sat Sep 14 11:42:25 2024.
Dependencies resolved.
```

Package	Arch	Version	Repository	Size
<b>Installing:</b>				
<code>docker</code>	x86_64	25.0.6-1.amzn2023.0.2	amazonlinux	44 M
<b>Installing dependencies:</b>				
<code>containerd</code>	x86_64	1.7.20-1.amzn2023.0.1	amazonlinux	35 M
<code>iptables-libs</code>	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
<code>iptables-nft</code>	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
<code>libcgroup</code>	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k
<code>libnetfilter_conntrack</code>	x86_64	1.0.8-2.amzn2023.0.2	amazonlinux	58 k
<code>libnftnl</code>	x86_64	1.0.1-19.amzn2023.0.2	amazonlinux	30 k
<code>libnftnl</code>	x86_64	1.2.2-2.amzn2023.0.2	amazonlinux	84 k
<code>pigz</code>	x86_64	2.5-1.amzn2023.0.3	amazonlinux	83 k
<code>runc</code>	x86_64	1.1.13-1.amzn2023.0.1	amazonlinux	3.2 M

#### Transaction Summary

Install 10 Packages

```
Total download size: 84 M
Installed size: 317 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 4.5 MB/s | 401 kB   00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 4.5 MB/s | 183 kB   00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm 3.9 MB/s | 75 kB   00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.2.6 MB/s | 58 kB   00:00
(5/10): libnftnl-1.0.1-19.amzn2023.0.2.x86_64 1.2 MB/s | 30 kB   00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm 2.2 MB/s | 84 kB   00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm 2.8 MB/s | 83 kB   00:00
(8/10): runc-1.1.13-1.amzn2023.0.1.x86_64.rpm 30 MB/s | 3.2 MB   00:00
(9/10): containerd-1.7.20-1.amzn2023.0.1.x86_64 38 MB/s | 35 kB   00:00
(10/10): docker-25.0.6-1.amzn2023.0.2.x86_64.rpm 34 MB/s | 44 kB   00:01
```

Total	62 MB/s		84 MB	00:01
-------	---------	--	-------	-------

```
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
```

```
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
```

```
Verifying : containerd-1.7.20-1.amzn2023.0.1.x86_64
Verifying : docker-25.0.6-1.amzn2023.0.2.x86_64
Verifying : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
Verifying : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64
Verifying : libcgroup-3.0-1.amzn2023.0.1.x86_64
Verifying : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
Verifying : libnftnl-1.0.1-19.amzn2023.0.2.x86_64
Verifying : libnftnl-1.2.2-2.amzn2023.0.2.x86_64
Verifying : pigz-2.5-1.amzn2023.0.3.x86_64
Verifying : runc-1.1.13-1.amzn2023.0.1.x86_64
```

Installed:	containerd-1.7.20-1.amzn2023.0.1.x86_64	docker-25.0.6-1.amzn2023.0.2.x86_64	iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
	libcgroup-3.0-1.amzn2023.0.1.x86_64	libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64	libnftnl-1.0.1-19.amzn2023.0.2.x86_64
	pigz-2.5-1.amzn2023.0.3.x86_64	runc-1.1.13-1.amzn2023.0.1.x86_64	

Then, configure group in a daemon.json file by using following

commands cd /etc/docker

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

```
{
```

```
"exec-opts":
```

```
["native.cgroupdriver=systemd"],
```

```
"log-driver":
```

```
"json-file", "log-opt":
```

```
{
```

```
"max-size": "100m"
```

```
,
```

```
"storage-driver": "overlay2"
```

EOF

```
[ec2-user@ip-172-31-65-181 docker]$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": [
    "native.cgroupdriver=systemd"
  ],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF
{
  "exec-opts": [
    "native.cgroupdriver=systemd"
  ],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
```

Then after this run the following command to enable and start docker and also to load the daemon.json file. sudo systemctl enable docker sudo systemctl daemon-reload sudo systemctl restart docker

```
[ec2-user@ip-172-31-81-216 docker]$ sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[ec2-user@ip-172-31-81-216 docker]$ sudo systemctl daemon-reload
[ec2-user@ip-172-31-81-216 docker]$ sudo systemctl restart docker
```

docker -v

3. **Then Install Kubernetes with the following command.** • SELinux needs to be disable before configuring kubelet thus run the following command sudo setenforce 0 sudo sed -i 's/^SELINUX=enforcing\$/SELINUX=permissive/' /etc/selinux/config

```
[ec2-user@ip-172-31-65-181 docker]$ sudo setenforce 0
[ec2-user@ip-172-31-65-181 docker]$ sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

#### 4. Initialize the Kubecluster 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 172.31.26.174:6443 --token pv0yyi.xh11qhclfjr50pt8 \
--discovery-token-ca-cert-hash sha256:8293b2f6d29de466bd859007f5adbcdb3a
ecb0c446ba09033d32a5846b3d434f
```

- copy the token and save for future use .  
kubeadm join 172.31.26.174:6443 --token pv0yyi.xh11qhclfjr50pt8
\--discovery-token-ca-cert-hash  
sha256:8293b2f6d29de466bd859007f5adbcdb3aecb0c446ba09033d32a5846b
3d434f

#### 5. Now that the cluster is up and running, we can deploy our nginx server on this cluster. Apply deployment using this following command: kubectl apply -f

```
[ec2-user@ip-172-31-26-174 docker]$ kubectl apply -f https://k8s.io/examples/pods/s
imple-pod.yaml
pod/nginx created
```

Then use kubectl get nodes to check whether the pod gets created or not.

```
[ec2-user@ip-172-31-26-174 docker]$ kubectl get pods
NAME      READY  STATUS    RESTARTS   AGE
nginx     0/1    Pending   0          12s
```

To convert state from pending to running use the following command: kubectl describe pod nginx This command will help to describe the pods it gives reason for failure as it shows the taints which need to be untainted.

```
[ec2-user@ip-172-31-26-174 docker]$ kubectl describe pod nginx
Name:           nginx
Namespace:      default
Priority:       0
Service Account: default
Node:           <none>
Labels:          <none>
Annotations:    <none>
Status:          Pending
IP:             <none>
IPs:            <none>
Containers:
  nginx:
    Image:        nginx:1.14.2
    Port:         80/TCP
    Host Port:   0/TCP
    Environment: <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-k4lj6 (ro)

Conditions:
  Type      Status
  PodScheduled  False
Volumes:
  kube-api-access-k4lj6:
    Type:           Projected (a volume that contains injected data from m
    ultiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:      kube-root-ca.crt
    ConfigMapOptional:  <nil>
    DownwardAPI:       true
  QoS Class:      BestEffort
  Node-Selectors: <none>
  Tolerations:
    node.kubernetes.io/not-ready:NoExecute op=Exists for 3
    00s
    node.kubernetes.io/unreachable:NoExecute op=Exists for
    300s
Events:
  Type     Reason   Age   From           Message
  ----   -----   ---   ----
  Warning  FailedScheduling  7s   default-scheduler  0/1 nodes are available: 1 no
de(s) had untolerated taint {node-role.kubernetes.io/control-plane: }. preemption:
0/1 nodes are available: 1 Preemption is not helpful for scheduling.
[ec2-user@ip-172-31-26-174 ~]$ kubectl taint nodes --all node-role.kubernetes.io
/control-plane-
node/ip-172-31-26-174.ec2.internal untainted
```

- Lastly, mention the port you want to host. Here I have used localhost 8081 then checked it. kubectl port-forward nginx 8081:80

```
[ec2-user@ip-172-31-26-174 ~]$ kubectl port-forward nginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```

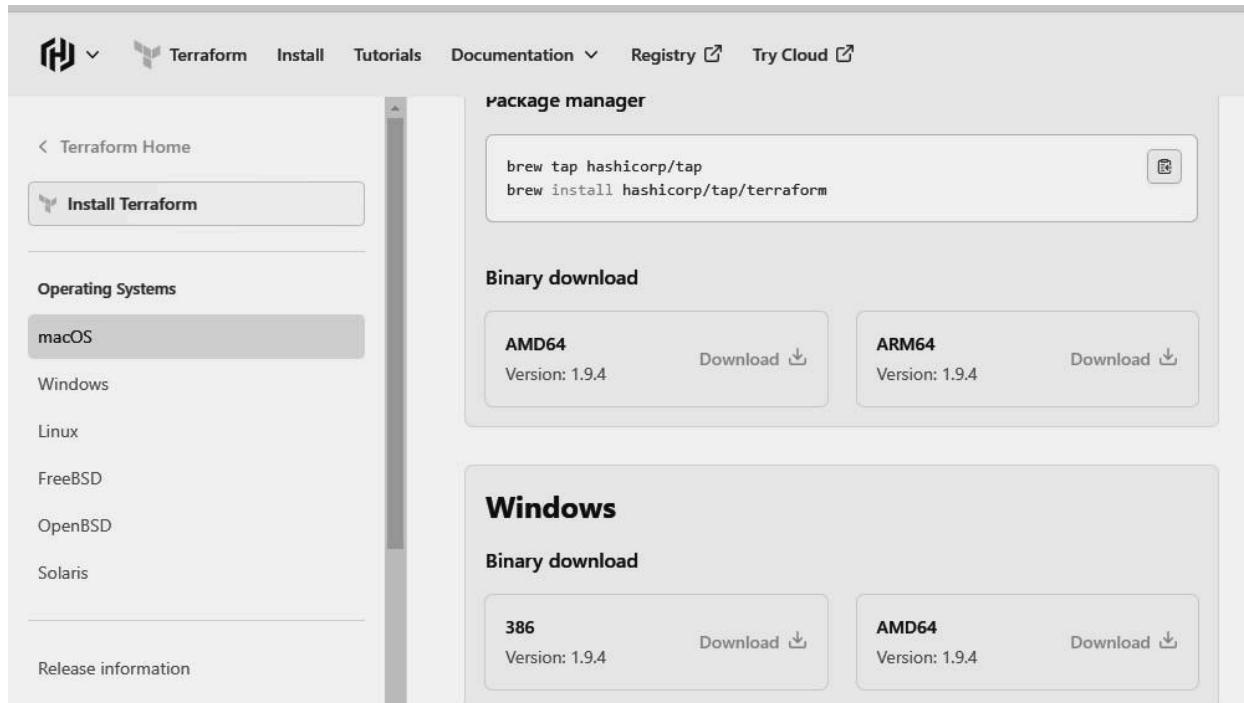
- Verify your deployment:** Open up a new terminal and ssh to your EC2 instance. Then, use this curl command to check if the Nginx server is running. curl --head http://127.0.0.1:8080 If the response is 200 OK and you can see the Nginx server name, your deployment was successful. We have successfully deployed our Nginx server on our EC2 instance.

**Conclusion:** First, I successfully launched an AWS EC2 instance running Amazon Linux. After that, I installed Docker and Kubernetes on the instance. Following the installation, I initialized the Kubernetes cluster, which provided me with a token, along with chown and mkdir commands. I then executed both the mkdir and chown commands successfully. Next, I installed the Flannel networking plugin without any issues. Initially, there was an error while deploying Nginx, but after correcting it, I successfully deployed Nginx using a simple-pod.yml file. I confirmed its deployment with the get pods command and hosted it locally , which worked as expected.

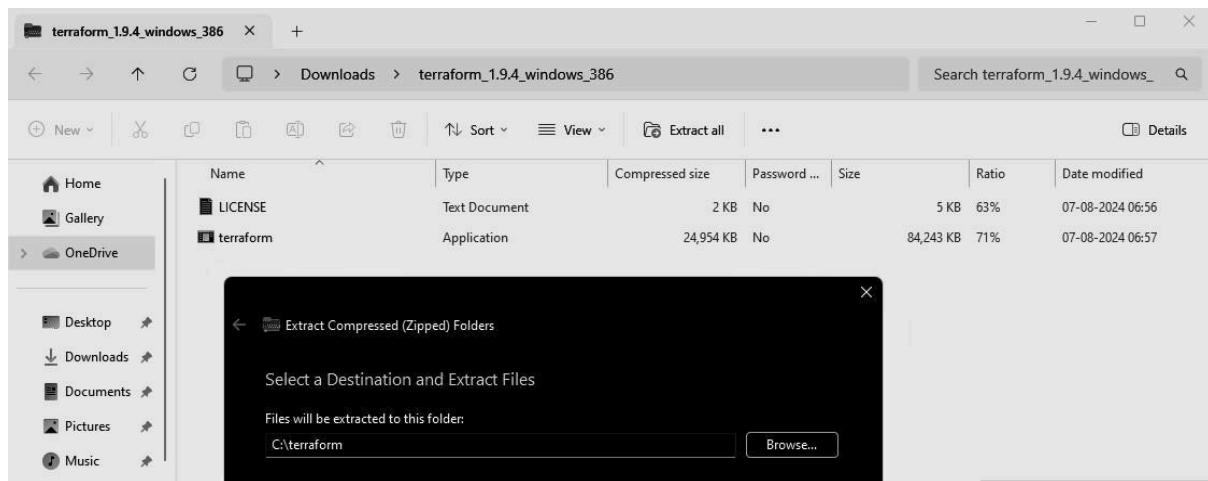
## 1: Download terraform

To install Terraform, First Download the Terraform Cli Utility for windows from terraforms official website <https://www.terraform.io/downloads.html>

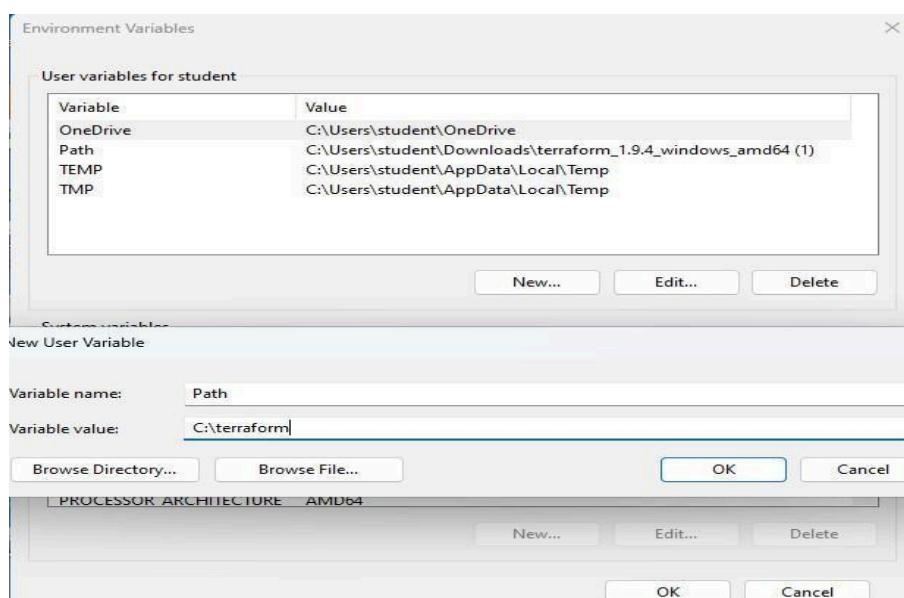
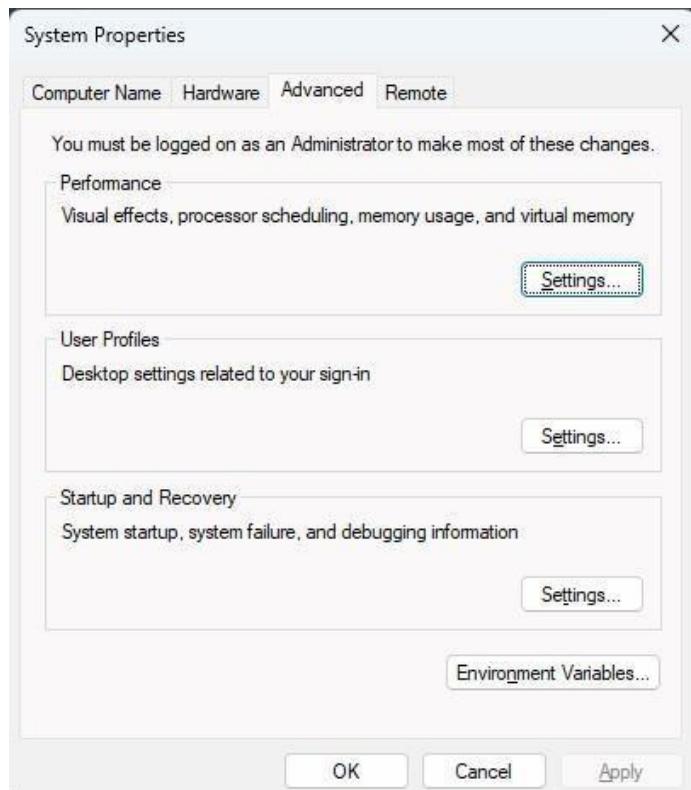
Select the Operating System Windows followed by either 32bit or 64 bit based on your OS type.



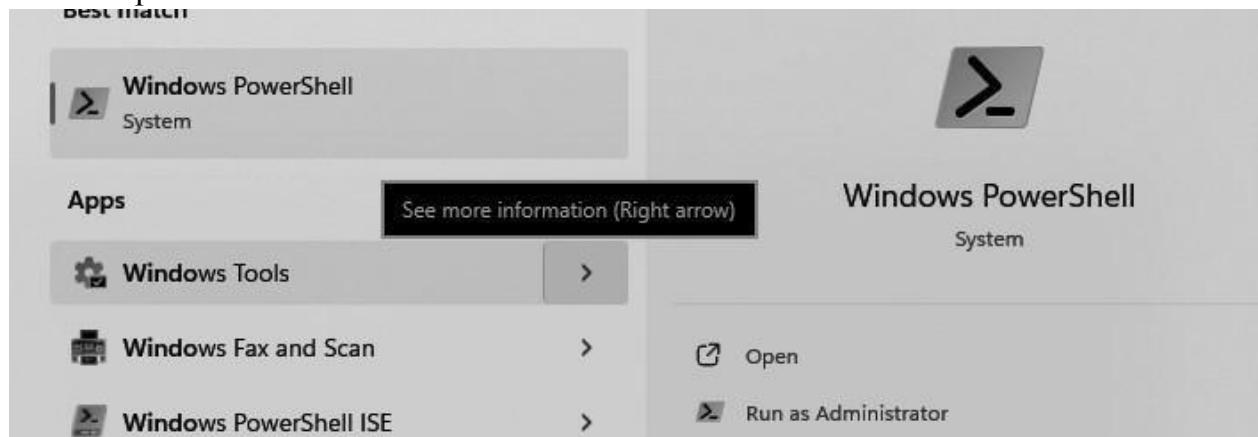
Step 2: Extract the downloaded setup file Terraform.exe in C:\Terraform directory



### 3: Set the System path for Terraform in Environment Variables



#### 4: Open PowerShell with Admin Access



Step 5 : Open Terraform in PowerShell and check its functionality

```
PS C:\Users\student> terraform
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
  .....           .....
```

# Experiment No:6

## Implementation:

### A. Creating docker image using terraform

Prerequisites:

1. Download and install Docker Desktop from

Website: <https://www.docker.com>.

```
C:\Users\excel>docker --version
Docker version 27.1.1, build 6312585
```

```
C:\Users\excel>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
```

Step 1:To Verify Docker Functionality

1. Create a folder named `Terraform Scripts` to store various scripts for this experiment.

Step 2:To Set Up Terraform Configuration

1. Inside the `Terraform Scripts` folder, create a new folder named `Docker`.
2. Within the `Docker` folder, create a file named `docker.tf` using Atom editor and insert the following content to configure an 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"]
}

```

```

EXPLORER      ...
DOCKER
docker.tf

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   }
 25

```

### Step 3:To Initialize Terraform

Run the command `terraform init` to initialize the Terraform configuration.

```
C:\Users\excel\Documents\college\Terraform scripts\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.
```

### Step 4:To Review Terraform Plan

Execute `terraform plan` to preview the resources that will be created.

```
C:\Users\excel\Documents\college\Terraform scripts\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",
    + "3000",
  ]
  + 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
}
```

```

+ 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.

```

## Step 5: To Apply Terraform Configuration

Run `terraform apply` to apply the configuration and create the Ubuntu Linux container.

```

C:\Users\excel\Documents\college\Terraform scripts\docker>terraform apply
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
}

```

```

+ 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: Still creating... [20s elapsed]
docker_image.ubuntu: Still creating... [30s elapsed]
docker_image.ubuntu: Still creating... [41s elapsed]
docker_image.ubuntu: Creation complete after 43s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 1s [id=71bffb28b5cee3d1699c27dbccb992b931000a847e6dfb219b3ca85ce5c6131]

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

```

Before executing `terraform apply` , list the Docker images.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
```

After executing `terraform apply` , list the Docker images again.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
ubuntu          latest   edbfe74c41f8  3 weeks ago  78.1MB
```

Step 6: Clean Up

To delete the created Ubuntu container, run `terraform destroy` .

```

C:\Users\excel\Documents\college\Terraform scripts\docker>terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131]

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          = "71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131" -> null
  id               = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  image             = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  init              = false -> null
  ip_address        = "172.17.0.2" -> null
  ip_prefix_length = 16 -> null
  ip_private        = "private" -> null
  links             = [] -> null
  log_driver        = "json-file" -> null
  log_opts          = {} -> null
  logs              = false -> null
  max_retry_count   = 0 -> null
  memory            = 512 -> null
  memory_swap       = 0 -> null
  must_run          = true -> null
  name              = "foo" -> null
}

- network_data      = [
  - {
    - gateway          = "172.17.0.1"
    - global_ipv6_prefix_length = 0
    - ip_address        = "172.17.0.2"
    - ip_prefix_length  = 16
    - network_name      = "bridge"
    # (2 unchanged attributes hidden)
  },
  ! -> null
- network_mode       = "bridge" -> null
- privileged          = false -> null
- publish_all_ports  = false -> null
- read_only           = false -> null
- remove_volumes     = true -> null
- restart             = false -> null
- rm                  = false -> null
- runtime             = "unc" -> null
- security_opts       = [] -> null
- shm_size            = 64 -> null
- start               = true -> null
- stdin_open          = false -> null
- stdt_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:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
  - image_id         = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - latest           = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
  - name              = "ubuntu:latest" -> null
  - repo_digest       = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> null
}

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

```

```
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=71bffb28b5cee3d1699c27dbcceb992b931000a847e6dfb219b3ca85ce5c6131]  
docker_container.foo: Destruction complete after 0s  
docker_image.ubuntu: Destroying... [id=sha256:edbf74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:  
docker_image.ubuntu: Destruction complete after 0s  
Destroy complete! Resources: 2 destroyed.
```

After executing `terraform destroy` , list the Docker images one more time.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>docker images  
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
```

Step 7:To check correctness of configured files.

Execute `terraform validate` to check the correctness of your Terraform configuration files.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform validate  
Success! The configuration is valid.
```

Step 8:To verify the details.

Run `terraform providers` to list the providers used in your configuration and verify their details.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform providers  
Providers required by configuration:  
└─ provider[registry.terraform.io/kreuzwerker/docker] 2.21.0
```

Step 9:To generate visual representation.

Generate a visual representation of the dependency graph of your Terraform resources.

```
C:\Users\excel\Documents\college\Terraform scripts\docker>terraform graph
digraph G {
    rankdir = "RL";
    node [shape = rect, fontname = "sans-serif"];
    "docker_container.foo" [label="docker_container.foo"];
    "docker_image.ubuntu" [label="docker_image.ubuntu"];
    "docker_container.foo" -> "docker_image.ubuntu";
}
```

## Experiment 7

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

### 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. 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:**

### **Prerequisites:**

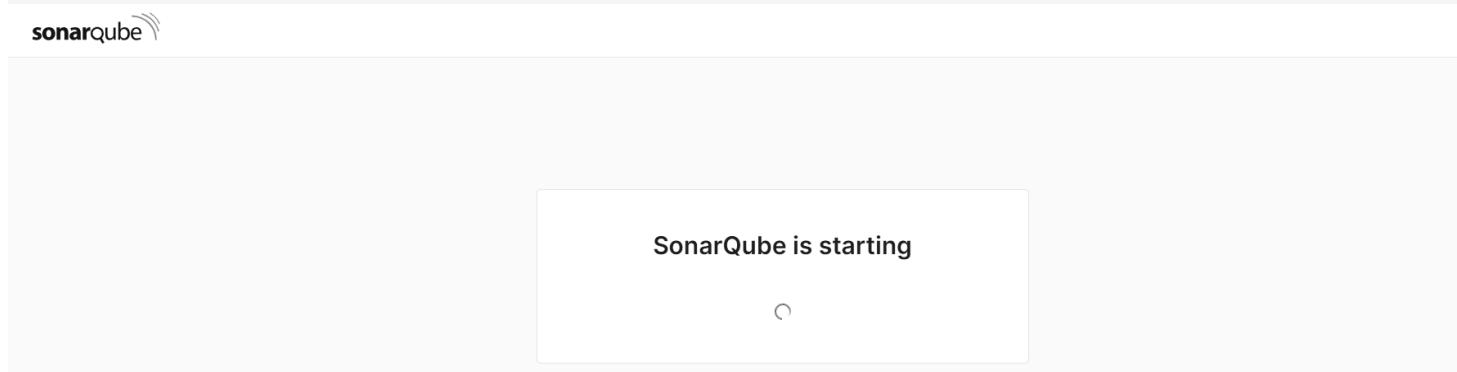
- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

## Steps to integrate Jenkins with SonarQube

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.
2. Run SonarQube in a Docker container using this command -

```
C:\Users\ADMIN>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
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
de76efbeef2054aeb442b86ba54c2916039b8757b388482d9780fffc69f5d8bbe
```

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



4. 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 \*

Project key \*

Main branch name \*

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

Setup the project and come back to Jenkins Dashboard.

6. Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

The screenshot shows the Jenkins Manage Plugins page. A search bar at the top contains the text "sonar". To the right of the search bar are two buttons: "Install" and a dropdown menu. Below the search bar, the results are listed. The first result is "SonarQube Scanner 2.17.2", which is categorized under "External Site/Tool Integrations" and "Build Reports". A brief description states: "This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality." To the right of the plugin details is the text "Released 7 mo 9 days ago".

7. Under Jenkins 'Configure System', look for SonarQube Servers and enter the details.

Enter the Server Authentication token if needed.

The screenshot shows the Jenkins System configuration page under "Manage Jenkins > System > SonarQube installations". The page title is "SonarQube installations" and the subtitle is "List of SonarQube installations". There is a form with fields for "Name" (containing "sonarqube"), "Server URL" (containing "http://localhost:9000"), and "Server authentication token" (with a dropdown menu showing "- none -" and a "+ Add" button). At the bottom of the form is a "Advanced" button.

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

The screenshot shows the Jenkins Global Tool Configuration page under "Manage Jenkins > Global Tool Configuration". The page title is "SonarQube Scanner installations". There is a "Add SonarQube Scanner" button. Below it, a section for "SonarQube Scanner" is shown. It has a "Name" field containing "sonarqube" and a checked checkbox for "Install automatically". A collapsed section titled "Install from Maven Central" is expanded, showing a "Version" field containing "SonarQube Scanner 6.1.0.4477". At the bottom of the page is a "Add Installer" button.

9. 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.

10. Choose this GitHub repository in Source Code Management.

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

Source Code Management

None

Git [?](#)

Repositories [?](#)

Repository URL [?](#)

Credentials [?](#)

- none -

[+ Add](#) [▼](#)

[Advanced](#) [▼](#)

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

11. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

**Build Steps**

**Execute SonarQube Scanner**

SonarQube Installation:

JDK:

Path to project properties:

Analysis properties:

```
sonar.projectKey=sonarqube
sonar.login=admin
sonar.password=admin123
sonar.sources=C:\\ProgramData\\Jenkins\\jenkins\\workspace\\SonarQube
sonar.host.url=http://127.0.0.1:9000
```

Additional arguments:

JVM Options:

**Save**   **Apply**

12. Go to [http://localhost:9000/<user\\_name>/permissions](http://localhost:9000/<user_name>/permissions) and allow Execute Permissions to the Admin user.

	Administrator System ?	Administer ?	Execute Analysis	Create ?
A Administrator admin	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	<input type="checkbox"/> Quality Gates	<input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects

13. Run The Build.

Status

</> Changes

Workspace

**Build Now**

Configure

Delete Project

SonarQube

Rename

## Check the console output.

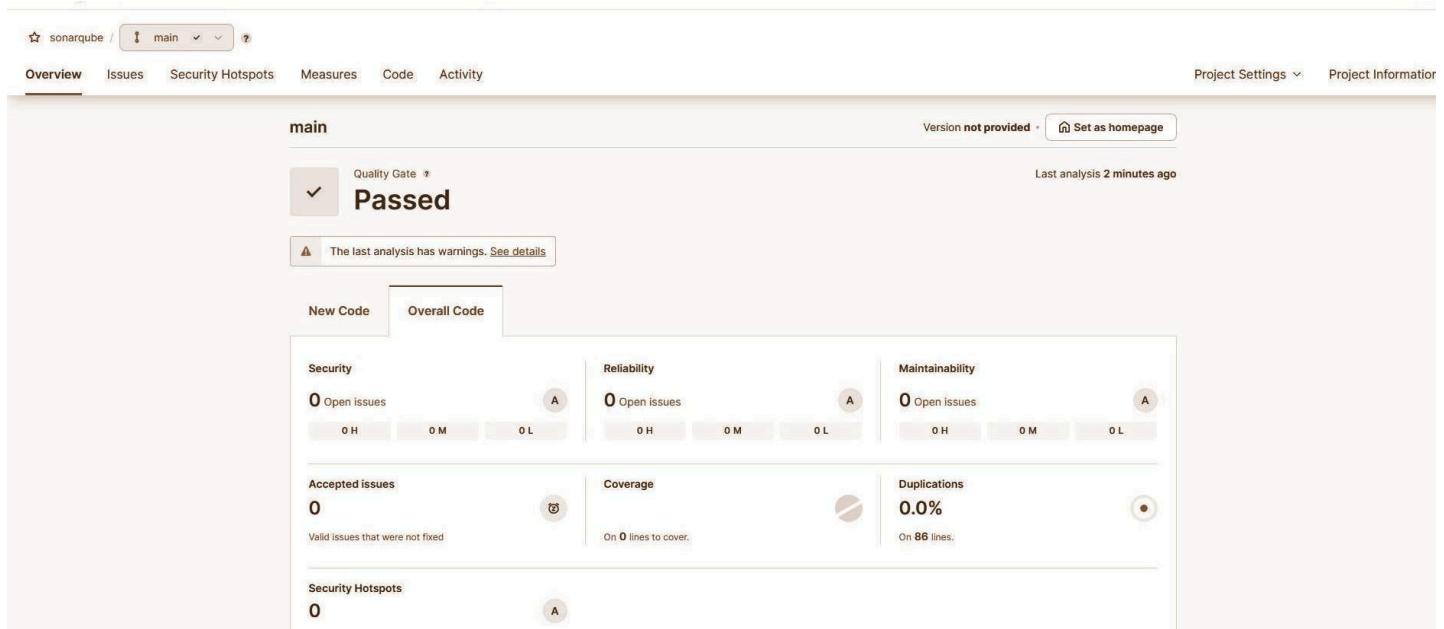
### ✓ Console Output

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

```
Started by user Shiven Bansal
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
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\SonarQube\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
> git.exe --version # timeout=10
> git --version # 'git' version 2.45.2.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcaee6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c6e72427c380bcaee6d6fee7b49adf # timeout=10
[SonarQube] $ C:\ProgramData\Jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.login=admin -Dsonar.host.url=http://127.0.0.1:9000 -Dsonar.sources=C:\ProgramData\Jenkins\jenkins\workspace\SonarQube -Dsonar.password=admin123 -
Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\SonarQube
16:16:39.198 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://127.0.0.1:9000'
16:16:39.206 INFO Scanner configuration file: C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin..\conf\sonar-scanner.properties
16:16:39.206 INFO Project root configuration file: NONE
16:16:39.230 INFO SonarScanner CLI 6.1.0.4477
16:16:39.230 INFO Java 21.0.4 Eclipse Adoptium (64-bit)
16:16:39.230 INFO Windows 11 10.0 amd64
16:16:39.230 INFO SONAR_SCANNER_OPTS=-Dsonar.ws.timeout=300
16:16:39.254 INFO User cache: C:\Windows\system32\config\systemprofile\.sonar\cache

16:16:58.734 INFO Using git CLI to retrieve untracked files
16:16:58.791 INFO Analyzing language associated files and files included via "sonar.text.inclusions" that are tracked by git
16:16:58.856 INFO 14 source files to be analyzed
16:16:59.154 INFO 14/14 source files have been analyzed
16:16:59.154 INFO Sensor TextAndSecretsSensor [text] (done) | time=1306ms
16:16:59.163 INFO -----
16:16:59.373 INFO Sensor C# [csharp]
16:16:59.373 WARN Your project contains C# files which cannot be analyzed with the scanner you are using. To analyze C# or VB.NET, you must use the SonarScanner for .NET 5.x or higher, see https://redirect.sonarsource.com/doc/install-configure-scanner-msbuild.html
16:16:59.373 INFO Sensor C# [csharp] (done) | time=0ms
16:16:59.373 INFO Sensor Analysis Warnings import [csharp]
16:16:59.379 INFO Sensor Analysis Warnings import [csharp] (done) | time=0ms
16:16:59.379 INFO Sensor C# File Caching Sensor [csharp]
16:16:59.379 WARN Incremental PR analysis: Could not determine common base path, cache will not be computed. Consider setting 'sonar.projectBaseDir' property.
16:16:59.379 INFO Sensor C# File Caching Sensor [csharp] (done) | time=6ms
16:16:59.379 INFO Sensor Zero Coverage Sensor
16:16:59.389 INFO Sensor Zero Coverage Sensor (done) | time=10ms
16:16:59.389 INFO SCM Publisher SCM provider for this project is: git
16:16:59.389 INFO SCM Publisher 4 source files to be analyzed
16:16:59.838 INFO SCM Publisher 4/4 source files have been analyzed (done) | time=449ms
16:16:59.846 INFO CPD Executor Calculating CPD for 0 files
16:16:59.846 INFO CPD Executor CPD calculation finished (done) | time=0ms
16:16:59.854 INFO SCM revision ID 'f2bc042c04c6e72427c380bcaee6d6fee7b49adf'
16:17:00.121 INFO Analysis report generated in 120ms, dir size=201.1 kB
16:17:00.195 INFO Analysis report compressed in 57ms, zip size=22.4 kB
16:17:00.393 INFO Analysis report uploaded in 195ms
16:17:00.394 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9000/dashboard?id=sonarqube
16:17:00.395 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
16:17:00.395 INFO More about the report processing at http://127.0.0.1:9000/api/ce/task?id=acd819f5-9e70-42ab-bff7-3cc893e2cae4
16:17:00.405 INFO Analysis total time: 18.743 s
16:17:00.408 INFO SonarScanner Engine completed successfully
16:17:00.494 INFO EXECUTION SUCCESS
16:17:00.494 INFO Total time: 21.288s
Finished: SUCCESS
```

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



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

## Conclusion:

In this experiment, I learned how to integrate Jenkins with SonarQube for performing Static Application Security Testing (SAST). I set up SonarQube in a Docker container and configured Jenkins to use the SonarQube scanner. By creating a manual project in SonarQube and configuring the necessary authentication and tools in Jenkins, I established a seamless connection between Jenkins and SonarQube for static code analysis.

I tested the integration with a sample GitHub repository, successfully running a build and analyzing the project's code quality through SonarQube. This hands-on experience enhanced my understanding of the SAST process, Jenkins automation, and SonarQube's capabilities for identifying potential code vulnerabilities.

## **Experiment - 8**

**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.

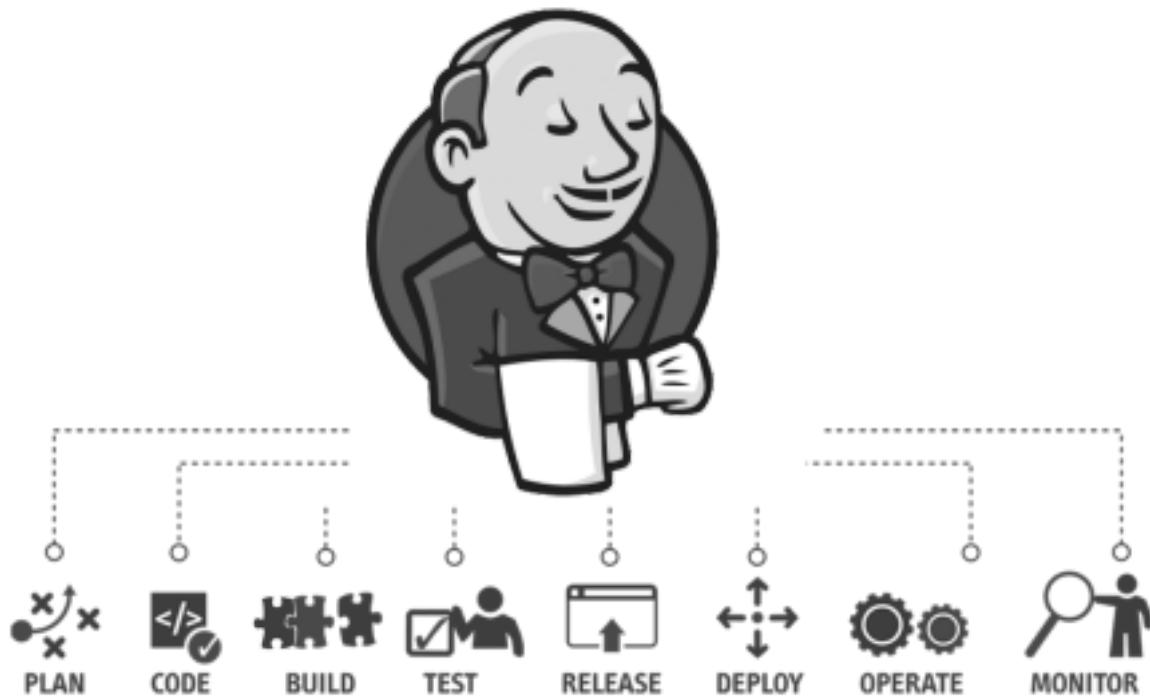
#### **Why is SAST important?**

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:

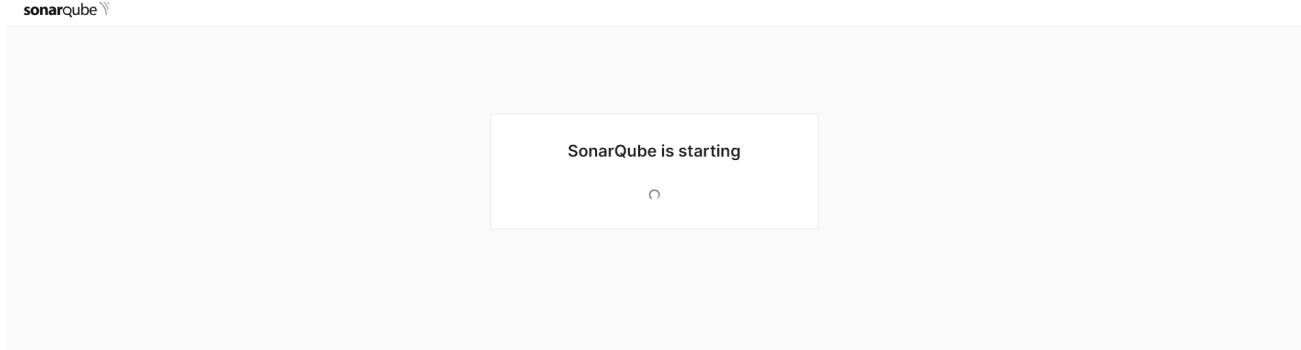
- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

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

```
C:\Users\ADMIN>docker run -d --name sonarqube2 -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9001:9000 sonarqube:latest  
fda86b00e3989f3eb5aca8396b29b2a0adc95bcfe0fc5d85cf1237491e7678b9
```

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.
4. Login to SonarQube using username *admin* and password *admin*.



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

1 of 2

### Create a local project

**Project display name \***  
sonarqube-test

**Project key \***  
sonarqube-test

**Main branch name \***  
main

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

**Cancel** **Next**

Setup the project and come back to Jenkins Dashboard.

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

Enter an item name  
SonarQube-8

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.
-  **Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
-  **Folder**

## Pipeline.

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') {  
            sh "<PATH_TO SONARQUBE FOLDER>/bin//sonar-scanner \  
-D sonar.login=<SonarQube_USERNAME> \  
-D sonar.password=<SonarQube_PASSWORD> \  
-D sonar.projectKey=<Project_KEY> \  
-D sonar.exclusions=vendor/**,resources/**,**/*.java \  
-D sonar.host.url=http://127.0.0.1:9000/"  
        }  
    }  
}
```

### Pipeline

#### Definition

##### Pipeline script

```
Script ?  
2+     stage('Cloning the GitHub Repo') {  
3+         git 'https://github.com/shazforiot/GOL.git'  
4+     }  
5+     stage('SonarQube analysis') {  
6+         withSonarQubeEnv('sonarqube') {  
7+             bat """  
8+             C:\ProgramData\Jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube\bin\sonar-scanner ^  
9+             -D sonar.login=admin ^  
10+            -D sonar.password=admin123 ^  
11+            -D sonar.projectKey=sonarqube-test ^  
12+            -D sonar.exclusions=vendor/**,resources/**,**/*.java ^  
13+            -D sonar.host.url=http://127.0.0.1:9001/  
14+            """  
15+        }  
16+    }  
17+ }
```

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

## 8. Run The Build.

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

The screenshot shows a CI pipeline interface with the following sections:

- Left Sidebar:** Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, Pipeline Syntax.
- Stage View:** Average stage times: (Average full run time: ~12min). It lists four stages:
  - Cloning the GitHub Repo: 5s
  - SonarQube analysis: 31min 25s
  - 12min 7s (Sep 26 18:04, No Changes)
  - 50min 43s (Sep 26 17:13, No Changes, aborted)
  - 12min 7s (Sep 26 17:11, No Changes)
  - 12min 7s (Sep 26 17:11, No Changes)
- Build History:** trend, Filter..., #4 (Sep 26, 2024, 6:04PM), #3 (Sep 26, 2024, 5:13PM).

The screenshot shows the SonarQube console output for build #4, which contains numerous WARN messages related to duplication references in JMeter documentation files. The messages are repeated across several log entries.

```
Skipping 4.248 KB. Full Log
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 788. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 810. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 823. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 844. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 509. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 1065. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 776. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 778. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 530. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 648. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 798. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 546. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 546. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 634. Keep only the first 100 references.
18:13:34.790 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/protocol/http/gui/HTTPArgumentsPanel.html for block at line 778. Keep only the first 100 references.
```

```

references.
18:13:39.657 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 75. Keep only the first 100 references.
18:13:39.657 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 41. Keep only the first 100 references.
18:13:39.657 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 17. Keep only the first 100 references.
18:13:39.657 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 286. Keep only the first 100 references.
18:13:39.657 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/gui/util/TextAreaCellRenderer.html for block at line 75. Keep only the first 100 references.
18:13:39.657 INFO CPD Executor CPD calculation finished (done) | time=158971ms
18:13:39.674 INFO SCM revision ID 'ba799ba7e1b576f04a4612322b0412c5e61e5e4'
18:15:49.696 INFO Analysis report generated in 5022ms, dir size=127.2 MB
18:16:08.759 INFO Analysis report compressed in 19048ms, zip size=29.6 MB
18:16:09.884 INFO Analysis report uploaded in 1125ms
18:16:09.887 INFO ANALYSIS SUCCESSFUL, you can find the results at: http://127.0.0.1:9001/dashboard?id=sonarqube-test
18:16:09.887 INFO Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
18:16:09.887 INFO More about the report processing at http://127.0.0.1:9001/api/ce/task?id=6f22c333-3777-4a21-b058-0ab4c049625
18:16:22.970 INFO Analysis total time: 12:02.242 s
18:16:22.975 INFO SonarScanner Engine completed successfully
18:16:23.699 INFO EXECUTION SUCCESS
18:16:23.706 INFO Total time: 12:05.758s
[Pipeline]
[Pipeline] // withSonarQubeEnv
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

## 10. After that, check the project in SonarQube.

The screenshot shows the SonarQube main dashboard for the 'main' project. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, More, and a search icon. Below the navigation is a breadcrumb trail showing the project name. The main content area features a large 'Passed' status indicator. It displays key metrics: 68k Lines of Code, Version not provided, and a note that the last analysis was 15 minutes ago. The dashboard is divided into several cards: Security (0 Open issues), Reliability (68k Open issues), Maintainability (164k Open issues), Accepted issues (0), Coverage (0%), and Duplications (50.6%). The 'Code' tab is selected, showing a breakdown of new and overall code. Other tabs available are Overview, Issues, Security Hotspots, Measures, and Activity. The bottom of the page includes footer links for Project Settings and Project Inform.

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

## 11. Code Problems -

### Bugs

The screenshot shows the SonarQube Issues page for the project "sonarqube-test". The left sidebar has the "Issues" tab selected. The main area displays a list of bugs, categorized by severity (Security, Reliability, Maintainability) and type (Bug, Vulnerability, Code Smell). A message at the bottom states: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine." The right side shows detailed views of specific bugs, including their descriptions, severity, and assigned tags.

### Code Smells

The screenshot shows the SonarQube Issues page for the project "sonarqube-test". The left sidebar has the "Issues" tab selected. The main area displays a list of code smells, categorized by severity (Security, Reliability, Maintainability) and type (Bug, Vulnerability, Code Smell). A message at the bottom states: "Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine." The right side shows detailed views of specific code smells, including their descriptions, severity, and assigned tags.

## Intentional Issues

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The top navigation bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', 'Administration', 'More', and a search bar. The 'Issues' tab is selected. On the left, a sidebar displays 'Filters' and 'Issues in new code'. Under 'Clean Code Attribute', there is a section for 'Intentionality' with 268 issues. Other sections include 'Adaptability' (0), 'Responsibility' (0), and 'Software Quality' with 'Security' (0), 'Reliability' (253), and 'Maintainability' (15). The main panel shows several code smells under 'gameoflife-acceptance-tests/Dockerfile', such as 'Use a specific version tag for the image.' and 'Surround this variable with double quotes; otherwise, it can lead to unexpected behavior.'. Each item has a checkbox, 'Intentionality' status (e.g., 'Maintainability'), and a timestamp like 'L1 5min effort - 4 years ago'. A note at the bottom states: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.'

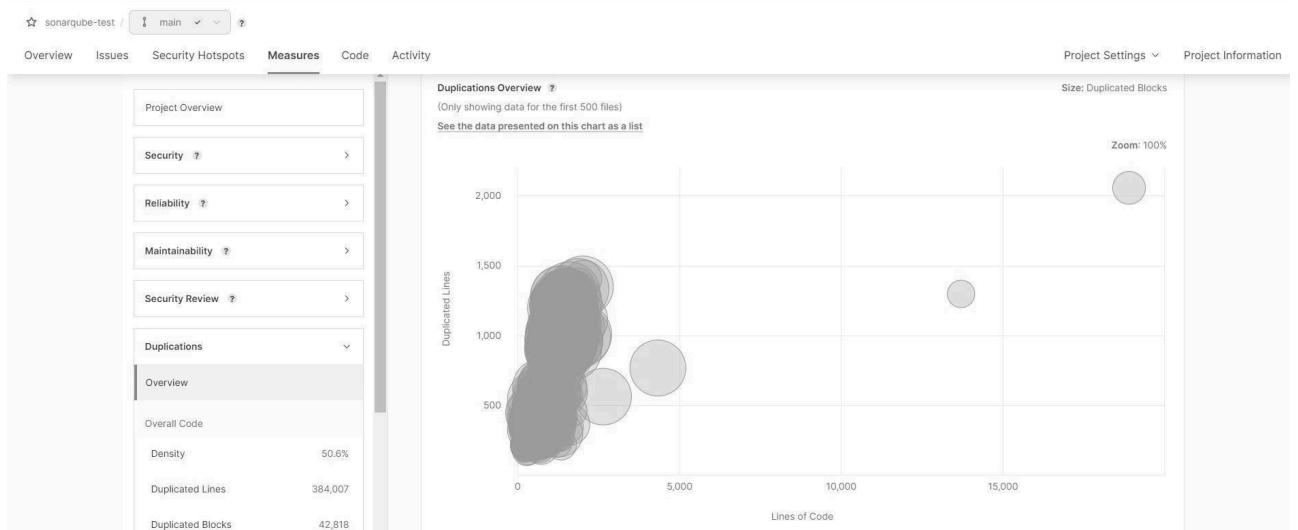
## Reliability Issue

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The top navigation bar and sidebar are identical to the previous screenshot. The main panel shows 'gameoflife-core/build/reports/tests/all-tests.html' with multiple instances of the issue 'Anchors must have content and the content must be accessible by a screen reader.' under 'Consistency' and 'Reliability'. Each instance has a checkbox, 'Consistency' status (e.g., 'Maintainability'), and a timestamp like 'L29 5min effort - 4 years ago'. A note at the bottom states: 'Embedded database should be used for evaluation purposes only.'

## Maintainability Issue

The screenshot shows the SonarQube interface for the project 'sonarqube-test'. The top navigation bar and sidebar are identical. The main panel shows 'gameoflife-core/build/reports/tests/all-tests.html' with multiple instances of the issue 'Remove this deprecated "width" attribute.' under 'Consistency' and 'html5 obsolete'. Each instance has a checkbox, 'Consistency' status (e.g., 'Maintainability'), and a timestamp like 'L9 5min effort - 4 years ago'. A note at the bottom states: 'Embedded database should be used for evaluation purposes only.'

## Duplicates



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, I successfully created a CI/CD pipeline using Jenkins integrated with SonarQube for static code analysis on a sample Java application. I set up SonarQube in a Docker container and configured Jenkins to clone the GitHub repository and perform the analysis. The pipeline detected various issues, including bugs, code smells, and security vulnerabilities, which I reviewed in SonarQube. This experience enhanced my skills in configuring CI/CD tools and highlighted the importance of maintaining code quality through automation. Overall, I gained valuable insights into integrating tools for effective software development practices.

# Adv DevOps Practical 9

**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.

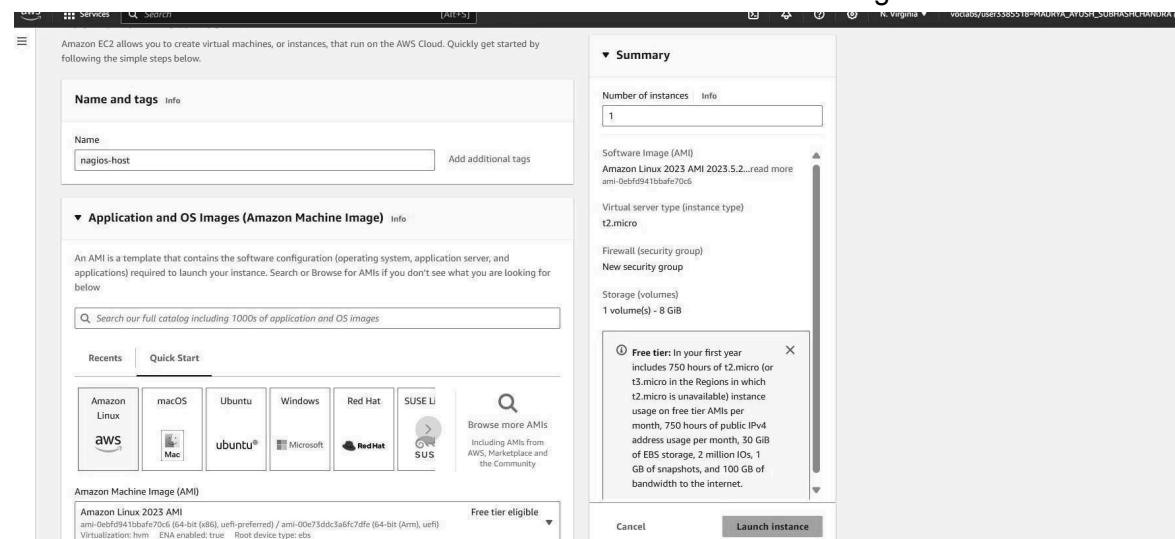
Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture

### Installation of Nagios Prerequisites: AWS

Free Tier

### Steps:

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



**instance type**

t2.micro

Family: t2 - 1 vCPU 1 GiB Memory Current generation: true  
On-Demand Windows base pricing: 0.0162 USD per Hour  
On-Demand SUSE base pricing: 0.0116 USD per Hour  
On-Demand RHEL base pricing: 0.026 USD per Hour  
On-Demand Linux base pricing: 0.0116 USD per Hour

Additional costs apply for AMIs with pre-installed software

**All generations**

**Compare instance types**

**Number of instances** Info

1

**Software Image (AMI)**  
Amazon Linux 2023 AMI 2023.5.2...read more  
ami-0ebfd941bbafe70c6

**Virtual server type (instance type)**  
t2.micro

**Firewall (security group)**  
New security group

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

**Free tier: In your first year includes 750 hours of t2.micro (or**

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

**Create security group** **Select existing security group**

We'll create a new security group called "launch-wizard-3" with the following rules:

Allow SSH traffic from Anywhere  
Helps you connect to your instance

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

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

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

**Firewall (security group)**  
New security group

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

**Free tier: In your first year includes 750 hours of t2.micro (or**

**EC2 Dashboard**

**Instances**

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

**Images**

AMIs

AMI Catalog

**Elastic Block Store**

Volumes

Snapshots

**Successfully initiated stopping of i-0c67658f4d6eea8fc, i-0414d4f92af63c03e, i-0d57570:061c25ae1, i-075644ff15b74f611**

**Instances (1/5) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv6
nagios-host	i-0011127bfbdb2f467	Running	t2.micro	Initializing	View alarms +	us-east-1d	ec2-44-204-11-28.compute-1.amazonaws.com	44.204.11.
Master	i-0c67658f4d6eea8fc	Stopped	t2.micro	2/2 checks passed	View alarms +	us-east-1d	-	-
node1	i-0414d4f92af63c03e	Stopping	t2.micro	2/2 checks passed	View alarms +	us-east-1d	ec2-54-159-206-1.compute-1.amazonaws.com	54.159.206
node2	i-0d57570c061c25ae1	Stopping	t2.micro	2/2 checks passed	View alarms +	us-east-1d	ec2-44-202-235-83.compute-1.amazonaws.com	44.202.235
exp_4	i-075644ff15b74f611	Stopped	t2.micro	2/2 checks passed	View alarms +	us-east-1d	-	-

**i-0011127bfbdb2f467 (nagios-host)**

**security details**

IAM Role -

Security groups

sg-09d51590eb1851b46 (launch-wizard-3)

Owner ID: 217253764927

Launch time: Sun Sep 29 2024 12:25:44 GMT+0530 (India Standard Time)

## 2. Under Security Group, make sure HTTP, HTTPS, SSH, ICMP are open from everywhere.

**Security Groups (6) Info**

**Create security group**

**Find resources by attribute or tag**

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-070583550d576c53e	launch-wizard-2	vpc-0dd4c0d8f48c2e4508	launch-wizard-2 created 2024-09-27T...	217253764927
-	sg-0300a1b62a1e9894	NodeGroup	vpc-0dd4c0d8f48c2e4508	Node	217253764927
-	sg-03f412e8ec9ec5946	launch-wizard-1	vpc-0dd4c0d8f48c2e4508	launch-wizard-1 created 2024-09-27T...	217253764927
-	sg-000c20590a5551206	default	vpc-0dd4c0d8f48c2e4508	default VPC security group	217253764927
-	sg-097fc30a345cta537	MasterGroup	vpc-0dd4c0d8f48c2e4508	Master	217253764927
-	sg-09d51590eb1851b46	launch-wizard-3	vpc-0dd4c0d8f48c2e4508	launch-wizard-3 created 2024-09-29T...	217253764927

EC2 > Security Groups > sg-09d51590eb1851b46

### sg-09d51590eb1851b46 - launch-wizard-3

[Actions ▾](#)

Details	
Security group name <a href="#">launch-wizard-3</a>	Security group ID <a href="#">sg-09d51590eb1851b46</a>
Owner <a href="#">217253764927</a>	Description <a href="#">launch-wizard-3 created 2024-09-29T06:49:51.498Z</a>
	VPC ID <a href="#">vpc-0d4c0d8f48c2e4508</a>
Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry

[Inbound rules](#) | [Outbound rules](#) | [Tags](#)

**Inbound rules (1)**

[Search](#) [C](#) [Manage tags](#) [Edit inbound rules](#)

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0ec19557ab93305...	IPv4	SSH	TCP	22	0.0.0.0/0	-

**Edit inbound rules** Info

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

**Inbound rules Info**

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
sgr-0ec19557ab9330565	SSH	TCP	22	Custom	<input type="text"/> <a href="#">Delete</a>
-	HTTP	TCP	80	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>
-	All ICMP - IPv6	IPv6 ICMP	All	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>
-	HTTPS	TCP	443	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>
-	All traffic	All	All	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>
-	Custom TCP	TCP	5666	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>
-	All ICMP - IPv4	ICMP	All	Anywhere-I...	<input type="text"/> <a href="#">Delete</a>

[Add rule](#)

Security group name	Security group ID	Description	VPC ID
<a href="#">launch-wizard-3</a>	<a href="#">sg-09d51590eb1851b46</a>	<a href="#">launch-wizard-3 created 2024-09-29T06:49:51.498Z</a>	<a href="#">vpc-0d4c0d8f48c2e4508</a>
Owner <a href="#">217253764927</a>	Inbound rules count 7 Permission entries	Outbound rules count 1 Permission entry	

[Inbound rules](#) | [Outbound rules](#) | [Tags](#)

**Inbound rules (7)**

[Search](#) [C](#) [Manage tags](#) [Edit inbound rules](#)

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sgr-034c50eff5e5fa00	IPv4	All ICMP - IPv6	IPv6 ICMP	All	0.0.0.0/0	-
-	sgr-058d0d3791dfcc60e	IPv4	HTTPS	TCP	443	0.0.0.0/0	-
-	sgr-0e8ad1dd008b14...	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0	-
-	sgr-0ec19557ab93305...	IPv4	SSH	TCP	22	0.0.0.0/0	-
-	sgr-00a0e56d560959f45	IPv4	HTTP	TCP	80	0.0.0.0/0	-
-	sgr-064c062d69916fa84	IPv4	Custom TCP	TCP	5666	0.0.0.0/0	-
-	sgr-0613b7b6aa9d30def	IPv4	All traffic	All	All	0.0.0.0/0	-

You have to edit the inbound rules of the specified Security Group for this.

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

**Connect to instance** Info

Connect to your instance i-0011127bbfdb2f467 (nagios-host) using any of these options

EC2 Instance Connect Session Manager SSH client EC2 serial console

Instance ID  
 i-0011127bbfdb2f467 (nagios-host)

1. Open an SSH client.  
2. Locate your private key file. The key used to launch this instance is exp\_09.pem  
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
 chmod 400 "exp\_09.pem"  
4. Connect to your instance using its Public DNS:  
 ec2-44-204-11-28.compute-1.amazonaws.com

Example:  
 ssh -i "exp\_09.pem" ec2-user@ec2-44-204-11-28.compute-1.amazonaws.com

**Note:** In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Or open command prompt and paste ssh command.

```
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Ayush Maurya>ssh -i "Downloads/exp_09.pem" ec2-user@ec2-44-204-11-28.compute-1.amazonaws.com
The authenticity of host 'ec2-44-204-11-28.compute-1.amazonaws.com (44.204.11.28)' can't be established.
ED25519 key fingerprint is SHA256:v2OKH/ezl9iu7/RT6m8LWkgWzEJnnQIqrG9gKZWc14.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-204-11-28.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

#_
~\_\_ #####_      Amazon Linux 2023
~~ \_\#\#\#\#\_
~~ \#\#\|_
~~ \#/ _-- https://aws.amazon.com/linux/amazon-linux-2023
~~ V~' '-->
~~ /_
~~ .-_- /_
~~ /_-/
~~ /m'/_
Last login: Sun Sep 29 07:11:40 2024 from 18.206.107.27
[ec2-user@ip-172-31-91-91 ~]$ |
```

#### sudo yum update

```
[ec2-user@ip-172-31-91-91 ~]$ 
sudo yum update
Last metadata expiration check: 0:19:03 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-91-91 ~]$ |
```

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:19:29 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
=====
  Package           Architecture   Version        Repository      Size
=====
[Installing:
  httpd            x86_64         2.4.62-1.amzn2023
  php8.3           x86_64         8.3.16-1.amzn2023.0.1
[Installing dependencies:
  apr              x86_64         1.7.2-2.amzn2023.0.2
  apr-util          x86_64         1.6.3-1.amzn2023.0.1
  generic-logos-httdp noarch       18.0.0-12.amzn2023.0.3
  httpd-core        x86_64         2.4.62-1.amzn2023
  httpd-filesystem noarch       2.4.62-1.amzn2023
  httpd-tools        x86_64         2.4.62-1.amzn2023
  libbrotli          x86_64         1.0.9-4.amzn2023.0.2
  libodium           x86_64         1.0.19-4.amzn2023
  libssl             x86_64         1.1.34-5.amzn2023.0.2
  mailcap            noarch       2.1.49-3.amzn2023.0.3
  nginx-filesystem  noarch       1:1.24.0-1.amzn2023.0.4
  php8.3-cli         x86_64         8.3.16-1.amzn2023.0.1
  php8.3-common      x86_64         8.3.16-1.amzn2023.0.1
  php8.3-process     x86_64         8.3.16-1.amzn2023.0.1
  php8.3-xml          x86_64         8.3.16-1.amzn2023.0.1
[Installing weak dependencies:
  apr-util-openssl x86_64         1.6.3-1.amzn2023.0.1
  mod_http2          x86_64         2.0.27-1.amzn2023.0.3
  mod_lua            x86_64         2.4.62-1.amzn2023
  php8.3-fpm          x86_64         8.3.16-1.amzn2023.0.1
  php8.3-mbstring    x86_64         8.3.16-1.amzn2023.0.1
  php8.3-opcache      x86_64         8.3.16-1.amzn2023.0.1
  php8.3-pdo           x86_64         8.3.16-1.amzn2023.0.1
  php8.3-sodium        x86_64         8.3.16-1.amzn2023.0.1
=====
Total                                         22 MB/s | 10 MB   00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing :
  Installing : php8.3-common-8.3.16-1.amzn2023.0.1.x86_64
  Installing : apr-1.7.2-2.amzn2023.0.2.x86_64
  Installing : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
  Installing : apr-util-1.6.3-1.amzn2023.0.1.x86_64
  Installing : mailcap-2.1.49-3.amzn2023.0.3.noarch
=====
S  Running scriptlet: httpd-filesystem-2.4.62-1.amzn2023.noarch
=====
1/1
1/25
2/25
3/25
4/25
5/25
6/25
```

```
[ec2-user@ip-172-31-91-91 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:21:30 ago on Sun Sep 29 06:56:15 2024.
Dependencies resolved.
=====
  Package           Architecture   Version        Repository      Size
=====
[Installing:
  gd               x86_64         2.3.3-5.amzn2023.0.3
  gd-devel          x86_64         2.3.3-5.amzn2023.0.3
[Installing dependencies:
  brotli           x86_64         1.0.9-4.amzn2023.0.2
  brotli-devel      x86_64         1.0.9-4.amzn2023.0.2
  bzip2-devel        x86_64         1.0.8-6.amzn2023.0.2
  cairo             x86_64         1.17.6-2.amzn2023.0.1
  cmake-filesystem  x86_64         3.22.2-1.amzn2023.0.4
  fontconfig         x86_64         2.13.94-2.amzn2023.0.2
=====
139 k
38 k
314 k
31 k
214 k
684 k
16 k
273 k
```

```

Installed:
brotli-devel-1.0.0-4_amzn2023.0.2.x86_64
cairo-1.17.6-2_amzn2023.0.1.x86_64
fontconfig-devel-2.13.91-2_amzn2023.0.2.x86_64
freetype-devel-2.13.2-5_amzn2023.0.1.x86_64
glib2-devel-2.74.7-68_amzn2023.0.2.x86_64
graphite2-1.3.14-7_amzn2023.0.2.x86_64
harfbuzz-devel-7.0.0-2_amzn2023.0.1.x86_64
langpacks-core-font-en-3.0-21_amzn2023.0.4.noarch
libX11-1.7.2-3_amzn2023.0.4.x86_64
libX11-xcb-1.7.2-3_amzn2023.0.4.x86_64
libXext-1.3.4-6_amzn2023.0.2.x86_64
libXrender-0.9.10-14_amzn2023.0.2.x86_64
libffi-devel-3.2-4-1_amzn2023.0.1.x86_64
libjpeg-turbo-2.1.4-2_amzn2023.0.5.x86_64
libpng-2.1.6.37-10_amzn2023.0.6.x86_64
libsep0-devel-1.2.4-1_amzn2023.0.3.x86_64
libsep1-1.2.4-1_amzn2023.0.3.x86_64
libxcb-devel-1.12.1.7-1_amzn2023.0.1.x86_64
pcre2-utf16-10.40-1_amzn2023.0.3.x86_64
psprof-capture-devel-3.40.1-2_amzn2023.0.2.x86_64
xz-devel-5.2.5-3_amzn2023.0.2.x86_64

bzip2-devel-1.0.6-6_amzn2023.0.2.x86_64
fontconfig-2.13.91-2_amzn2023.0.2.x86_64
freetype-2.13.2-5_amzn2023.0.1.x86_64
gd-2.3.3-5_amzn2023.0.2.noarch
google-noto-fonts-common-28261206-2_amzn2023.0.2.noarch
graphite2-devel-1.3.14-7_amzn2023.0.2.x86_64
harfbuzz-icu-7.0.0-2_amzn2023.0.1.x86_64
libICE-1.0.10-6_amzn2023.0.2.x86_64
libX11-common-1.7.2-3_amzn2023.0.4.noarch
libX11-devel-1.7.2-3_amzn2023.0.4.x86_64
libXau-devel-1.0.9-6_amzn2023.0.2.x86_64
libXpm-3.5.15-2_amzn2023.0.3.x86_64
libXt-1.2.0-4_amzn2023.0.2.x86_64
libicu-67.1-7_amzn2023.0.3.x86_64
libjpeg-turbo-devel-2.1.4-2_amzn2023.0.5.x86_64
libpng-devel-2.1.6.37-10_amzn2023.0.6.x86_64
libtiff-4.0.0-4_amzn2023.0.1.x86_64
libwebp-devel-1.3.4-1_amzn2023.0.6.x86_64
libxml2-2.10.4-1_amzn2023.0.5.x86_64
pcre2-utf32-10.40-1_amzn2023.0.3.x86_64
xml-common-0.6.3-56_amzn2023.0.2.noarch
zlib-devel-1.2.11-33_amzn2023.0.5.x86_64

bztp2-devel-1.0.6-6_amzn2023.0.2.x86_64
fontconfig-2.13.91-2_amzn2023.0.2.x86_64
freetype-2.13.2-5_amzn2023.0.1.x86_64
gd-devel-2.3.3-5_amzn2023.0.3.x86_64
google-noto-sans-vf=Fonte-20201206-2_amzn2023.0.2.noarch
harfbuzz-7.0.0-2_amzn2023.0.1.x86_64
jbigkit-libs-2.1-21_amzn2023.0.2.x86_64
libSM-1.2.3-8_amzn2023.0.2.x86_64
libICE-1.0.10-6_amzn2023.0.2.x86_64
libX11-devel-1.7.2-3_amzn2023.0.4.x86_64
libXau-devel-1.0.9-6_amzn2023.0.2.x86_64
libXpm-devel-3.5.15-2_amzn2023.0.3.x86_64
libblkid-devel-2.37.4-1_amzn2023.0.4.x86_64
libicu-devel-67.1-7_amzn2023.0.3.x86_64
libmount-devel-2.37.4-1_amzn2023.0.4.x86_64
libselinux-devel-3.4-5_amzn2023.0.2.x86_64
libtiff-devel-4.0.0-4_amzn2023.0.1.x86_64
libxcb-1.13.1-7_amzn2023.0.2.x86_64
pcre2-devel-10.40-1_amzn2023.0.3.x86_64
pixman-0.40.0-3_amzn2023.0.3.x86_64
xorg-x11proto-devel-2021.4-1_amzn2023.0.2.noarch

```

Complete!

- 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 (password :**

```

[ec2-user@ip-172-31-91-91 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
BAD PASSWORD: The password contains the user name in some form
Retype new password:
Sorry, passwords do not match.
New password:
Retype new password:
password: all authentication tokens updated successfully.
[ec2-user@ip-172-31-91-91 ~]|

```

**ayushmau)**

- Create a new user group

**sudo groupadd nagcmd**

```
[ec2-user@ip-172-31-91-91 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-91-91 ~]$ |
```

- 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**

```
[ec2-user@ip-172-31-91-91 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
[ec2-user@ip-172-31-91-91 ~]$ |
```

- Create a new directory for Nagios downloads

**mkdir**

**~/downloads cd**

```
[ec2-user@ip-172-31-91-91 ~]$ mkdir ~/downloads
cd ~/downloads
[ec2-user@ip-172-31-91-91 ~]$ |
```

**~/downloads**

- Use wget to download the source zip files.

**wget <https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz>**

```

[ec2-user@ip-172-31-91-91 downloads]$ cd ..
[ec2-user@ip-172-31-91-91 ~]$ cd ~/downloads
[ec2-user@ip-172-31-91-91 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-09-29 09:11:59-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fe7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: 'nagios-4.5.5.tar.gz'

nagios-4.5.5.tar.g 100%[=====] 1.97M 5.07MB/s in 0.4s

2024-09-29 09:11:59 (5.07 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]

[ec2-user@ip-172-31-91-91 downloads]$ |

```

wget <https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz>

```

[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ cd ..
[ec2-user@ip-172-31-91-91 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-29 09:14:28-- https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.4.11.tar.gz'

nagios-plugins-2.4 100%[=====] 2.62M 6.92MB/s in 0.4s

```

10. Use tar to unzip and change to that directory. tar zxvf nagios-4.5.5.tar.gz

```

[ec2-user@ip-172-31-91-91 downloads]$ tar zxvf nagios-4.0.8.tar.gz
nagios-4.0.8/
nagios-4.0.8/.gitignore
nagios-4.0.8/Changelog
nagios-4.0.8/INSTALLING
nagios-4.0.8/LEGAL
nagios-4.0.8/LICENSE
nagios-4.0.8/Makefile.in
nagios-4.0.8/README
nagios-4.0.8/README.asciidoc
nagios-4.0.8/THANKS
nagios-4.0.8/UPGRADING
nagios-4.0.8/base/
nagios-4.0.8/base/.gitignore

```

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

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

*Here we go an error*

```
[ec2-user@ip-172-31-91-91 downloads]$ ./configure --with-command-group=nagcmd  
-bash: ./configure: No such file or directory  
[ec2-user@ip-172-31-91-91 downloads]$ |
```

## Solution

Navigate to nagios folder in downloads

```
[ec2-user@ip-172-31-91-91 downloads]$ ls  
nagios-4.0.8  nagios-4.0.8.tar.gz  nagios-plugins-2.0.3.tar.gz  
[ec2-user@ip-172-31-91-91 downloads]$ cd nagios-4.0.8  
[ec2-user@ip-172-31-91-91 nagios-4.0.8]$ |
```

Error 2: Cannot find SSL headers.

Solution: Install openssl dev library

Steps:

sudo yum install openssl-devel

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo yum install openssl-devel  
Last metadata expiration check: 2:24:05 ago on Sun Sep 29 06:56:15 2024.  
Dependencies resolved.  
=====  
 Package           Arch      Version            Repository      Size  
=====  
 Installing:  
  openssl-devel    x86_64    1:3.0.8-1.amzn2023.0.14  amazonlinux   3.0 M  
  
 Transaction Summary  
=====  
 Install 1 Package  
  
 Total download size: 3.0 M  
 Installed size: 4.7 M  
 Is this ok [y/N]: y  
 Downloading Packages:
```

Now run

**./configure --with-command-group=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-172-31-91-91 nagios-4.5.5]$ |
```

12. Compile the source code.

**make all**

```
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o broker.o broker.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-172-31-91-91 nagios-4.5.5]$ make all

sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o nagios.o ./nagios.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o broker.o broker.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o nebmods.o nebmods.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o ../common/shared.o ../common/shared.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o query-handler.o query-handler.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o workers.o workers.c
In function 'get_wproc_list',
  inlined from 'get_worker' at workers.c:277:12:
workers.c:253:17: warning: '%s' directive argument is null [-Wformat-overflo
w=]
  253 |           log_debug_info(DEBUGL_CHECKS, 1, "Found specialized
worker(s) for '%s'", (slash && *slash != '/') ? slash : cmd_name);
|
|
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o checks.o checks.c
gcc -Wall -I.. -I.. -I../lib -I../include -I../include -I.. -g -O2 -DHAVE_
CONFIG_H -DNSCORE -c -o config.o config.c
gcc -Wall -T.. -T.. -T../lib -T../include -T../include -T.. -g -O2 -DHAVF
```

14. Edit the config file and change the email address.

**sudo nano /usr/local/nagios/etc/objects/contacts.cfg**

```

# CONTACTS
#
#####
# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {
    contact_name      nagiosadmin          ; Short name of user
    use               generic-contact        ; Inherit default values from generic-contact template (defined above)
    alias             Nagios Admin          ; Full name of user
    email             2022.ayush.maurya@ves.ac.in ; <>***** CHANGE THIS TO YOUR EMAIL ADDRESS *****

}

#####
# CONTACT GROUPS
#
#####

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup {
    contactgroup_name   admins
    alias              Nagios Administrators
    members            nagiosadmin
}

```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location ^U Undo ^A Set Mark ^J To Br^E Exit ^R Read File ^P Paste ^L Justify ^G Go To Line ^D Undo ^C Copy ^W Where Is

And change email with your email

## 15. Configure the web interface.

**sudo make install-webconf**

```
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
    ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

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

```

[ec2-user@ip-172-31-91-91 nagios-4.5.5]\$

## 16. 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-172-31-91-91 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ |
```

Password: Ayushmau

## 17. Restart Apache

**sudo service httpd restart**

```
Adding password for user nagiosadmin
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-91-91 nagios-4.5.5]$ |
```

18. 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-172-31-91-91 downloads]$ cd ~/downloads
[ec2-user@ip-172-31-91-91 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
nagios-plugins-2.4.11/build-aux/depcomp
nagios-plugins-2.4.11/build-aux/snippet/
nagios-plugins-2.4.11/build-aux/snippet/_Noreturn.h
nagios-plugins-2.4.11/build-aux/snippet/arg-nonnull.h
nagios-plugins-2.4.11/build-aux/snippet/c++defs.h
nagios-plugins-2.4.11/build-aux/snippet/warn-on-use.h
nagios-plugins-2.4.11/build-aux/test-driver
nagios-plugins-2.4.11/config_test/
```

19. Compile and install plugins

```
cd nagios-plugins-2.4.11
```

```
./configure --with-nagios-user=nagios --with-nagios-group=nagios
```

```
[ec2-user@ip-172-31-91-91 downloads]$ cd 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
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for Minix Amsterdam compiler... no
checking for ar... ar
checking for ranlib... ranlib
```

**make**

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ make
$ sudo make install
make all-recursive
make[1]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
Making all in gl
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11/
gl'
rm -f alloca.h-t alloca.h && \
{ echo '/* DO NOT EDIT! GENERATED AUTOMATICALLY! */'; \
cat ./alloca.in.h; \
} > alloca.h-t && \
mv -f alloca.h-t alloca.h
rm -f c++defs.h-t c++defs.h && \
sed -n -e '/_GL_CXXDEFS/, $p' \
< ../build-aux/snippet/c++defs.h \
> c++defs.h-t && \
mv c++defs.h-t c++defs.h
rm -f warn-on-use.h-t warn-on-use.h && \
sed -n -e '/^ifndef/, $p' \
< ../build-aux/snippet/warn-on-use.h \
> warn-on-use.h-t && \
mv warn-on-use.h-t warn-on-use.h
rm -f arg-nonnull.h-t arg-nonnull.h && \
sed -n -e '/GL_ARG_NONNULL/, $p' \
< ../build-aux/snippet/arg-nonnull.h \
> arg-nonnull.h-t && \
mv arg-nonnull.h-t arg-nonnull.h
/usr/bin/mkdir -p arpa
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Entering directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$
```

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.0.8
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 08-12-2014
License: GPL

Website: http://www.nagios.org
Reading configuration data...
Error in configuration file '/usr/local/nagios/etc/nagios.cfg' - Line 452 (Check result path '/usr/local/nagios/var/spool/checkresults' is not a valid directory)
  Error processing main config file!
```

Solution:

**# Create the missing directory:** If the directory is missing, create it with the necessary permissions:

```
sudo mkdir -p /usr/local/nagios/var/spool/checkresults
sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
sudo chmod 775 /usr/local/nagios/var/spool/checkresults
```

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo mkdir -p /usr/local/nagios/var/spool/checkresults
sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
sudo chmod 775 /usr/local/nagios/var/spool/checkresults
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$
```

Now run again

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

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
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.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 1 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
```

**sudo service nagios start**

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ sudo service nagios start
Starting nagios (via systemctl): [ OK ]
```

21. Check the status of Nagios

**sudo systemctl status nagios**

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo systemctl status nagios
● nagios.service - LSB: Starts and stops the Nagios monitoring server
  Loaded: Loaded (/etc/rc.d/init.d/nagios; generated)
  Active: active (running) since Sun 2024-09-29 08:04:30 UTC; 37s ago
    Docs: man:systemd-sysv-generator(8)
  Process: 68037 ExecStart=/etc/rc.d/init.d/nagios start (code=exited, status=0/SUCCESS)
  Tasks: 6 (limit: 1112)
  Memory: 2.0M
     CPU: 47ms
   CGroup: /system.slice/nagios.service
           └─68059 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─68061 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68062 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68063 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─68064 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─68065 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68063;pid=68063
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68062;pid=68062
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68064;pid=68064
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: wproc: Registry request: name=Core Worker 68061;pid=68061
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Warning: Could not open object cache file '/usr/local/nagios/var/objec>
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmxp2N>
Sep 29 08:04:30 ip-172-31-91-91.ec2.internal nagios[68059]: Successfully launched command file worker with pid 68065
Sep 29 08:04:39 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpTmg>
Sep 29 08:04:49 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpAfY>
Sep 29 08:04:59 ip-172-31-91-91.ec2.internal nagios[68059]: Error: Unable to create temp file '/usr/local/nagios/var/nagios.tmpCtQ>
[lines 1-26/26 (END)]
```

### Error:

The log messages suggest that Nagios is unable to create temporary files, particularly in the directory `/usr/local/nagios/var/`. This is typically caused by permission issues, or the directory might not exist.

### Solution:

Firstly check whether `/usr/local/nagios/var/` is there or not. If yes.....

**ls -ld /usr/local/nagios/var/**

Change ownership: Set the correct ownership for the Nagios user and group:

**sudo chown -R nagios:nagcmd /usr/local/nagios/var**

Set permissions: Ensure the directory has the right permissions:

**sudo chmod -R 775 /usr/local/nagios/var**

Restart Nagios: After adjusting the ownership and permissions, restart the Nagios service:

**sudo systemctl restart nagios**

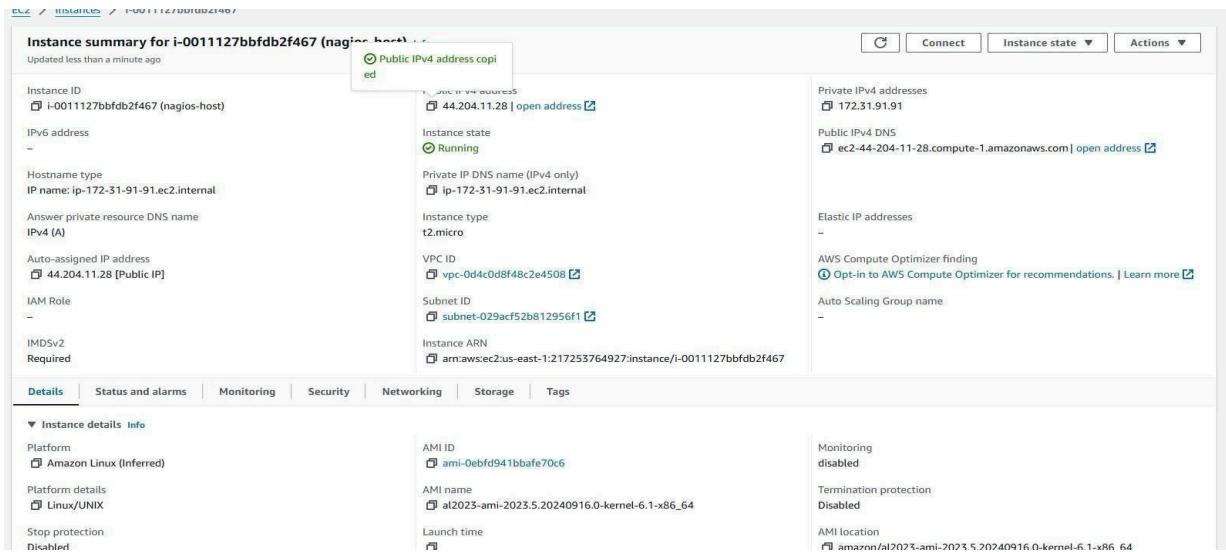
```
drwxr-xr-x. 4 root root 112 Sep 29 08:04 /usr/local/nagios/var/
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo chown -R nagios:nagcmd /usr/local/nagios/var
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo chmod -R 775 /usr/local/nagios/var
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ sudo systemctl restart nagios
[ec2-user@ip-172-31-91-91 nagios-plugins-2.0.3]$ |
```

Now run again

```
[ec2-user@ip-172-31-91-91 nagios-plugins-2.4.11]$ 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 Sun 2024-09-29 08:51:47 UTC; 42min ago
       Docs: https://www.nagios.org/documentation
      Tasks: 6 (limit: 1112)
     Memory: 2.9M
        CPU: 562ms
      CGroup: /system.slice/nagios.service
           ├─71188 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─71190 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─71191 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─71192 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─71193 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─71194 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: Registry request: name=Core Worker 71191;pid=71191
Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: Registry request: name=Core Worker 71190;pid=71190
Sep 29 08:51:47 ip-172-31-91-91.ec2.internal nagios[71188]: Successfully launched command file worker with pid 71194
Sep 29 08:59:22 ip-172-31-91-91.ec2.internal nagios[71188]: SERVICE ALERT: localhost;HTTP;WARNING;HARD;4;HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: SERVICE NOTIFICATION: nagiosadmin;localhost;Swap Usage;CRITICAL;notify-service-by-email;SWAP CRITICAL
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: NOTIFY job 10 from worker Core Worker 71194 is a non-check helper but exited with return code 1
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: host=localhost; service=Swap Usage; contact=nagiosadmin
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: early_timeout=0; exited_ok=1; wait_status=32512; error_code=0;
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: stderr line 01: /bin/sh: line 1: /bin/mail: No such file or directory
Sep 29 09:11:52 ip-172-31-91-91.ec2.internal nagios[71188]: wproc: stderr line 02: /usr/bin/printf: write error: Broken pipe
[lines 1-25/25 (END)]
```

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



## 23. 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.

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

The screenshot shows the Nagios Core web interface. At the top, there's a navigation bar with various links like 'Amazon.co.uk - Onli...', 'Express VPN', 'McAfee Security', 'LastPass password...', 'Automate the Boring...', 'Web Development...', 'Conceptzill...', 'Small Organization...', 'hotel-network/REA...', and 'Gurutech Networkin...'. Below the navigation bar is the Nagios Core logo with the text 'Nagios® Core™ Version 4.5.5' and the date 'September 17, 2024'. A message says 'Daemon running with PID 7118'. To the left is a sidebar with sections for 'General' (Home, Documentation), 'Current Status' (Tactical Overview, Map, Hosts, Services, Host Groups, Grid, Service Groups, Grid, Plugins, Services (Unhandled), Hosts (Unhandled), Network Outages, Quick Search), 'Reports' (Availability, Trends, Alerts, History, Summary, Histogram, Notifications, Event Log), and 'System' (Comments, Domains, Process Info, Performance Info, Scheduling Queue). The main content area has three columns: 'Get Started' (with a list of items like 'Start monitoring your infrastructure'), 'Latest News' (empty), and 'Don't Miss...' (empty). There's also a 'Quick Links' section with links to Nagios Library, Labs, Exchange, Support, and the official websites. At the bottom, there's a copyright notice and a license notice about the GNU General Public License. A 'Page Tour' button is located on the right side.

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

## Conclusion:

In this practical, we successfully installed and configured Nagios Core along with Nagios plugins and NRPE on an Amazon EC2 instance. We created a Nagios user, set up necessary permissions, and resolved common installation errors. Finally, we verified the setup by accessing the Nagios web interface, confirming that our monitoring system was fully operational.

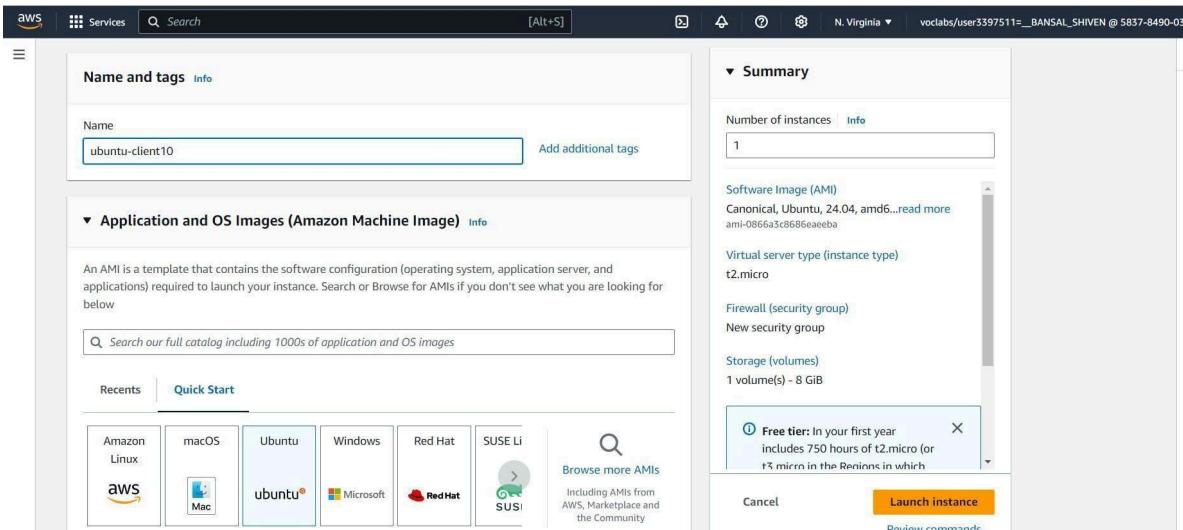
# Experiment 10

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

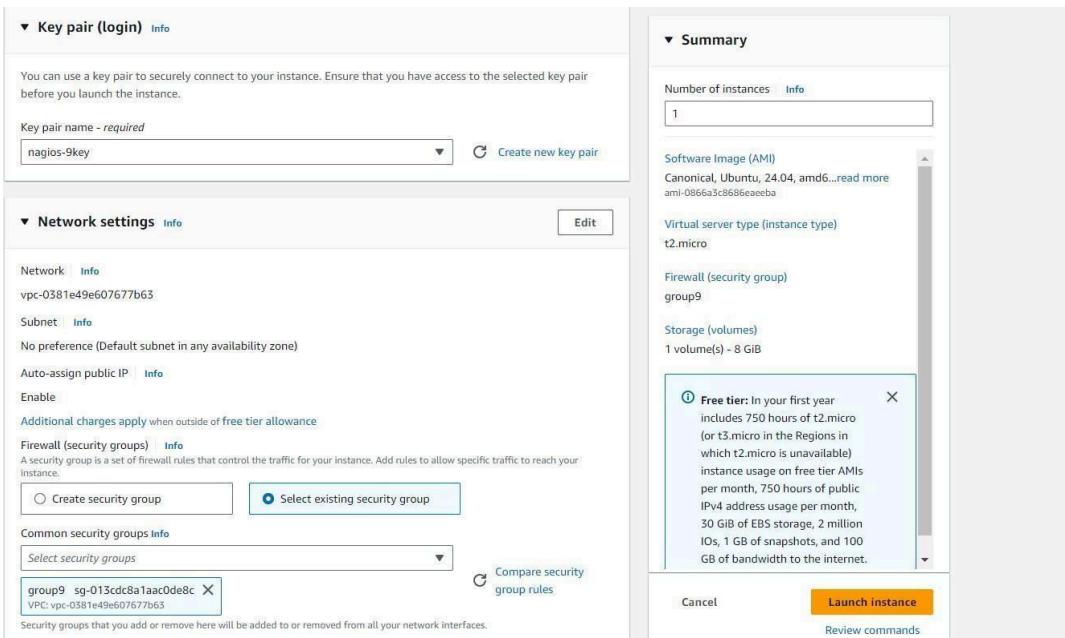
**Prerequisites:** An Amazon Linux instance with nagios (nagios-server) is already set up.

## Steps:

Step 1: Navigate to EC2 on the AWS console using the ‘Services’ section and click on ‘Create instance’. Give your instance a name and choose ‘Ubuntu’ as the instance type.



Ensure that you choose the same key pair and security group for the Ubuntu client instance as you did for the Nagios host instance. Then, click on ‘Create instance’.



Instances (3) <a href="#">Info</a>								
Last updated less than a minute ago								<a href="#">Connect</a>
<a href="#">Instance state = running</a>		<a href="#">Clear filters</a>		Actions				
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	
	i-040355adcc131cadd	<span>Running</span>	t2.micro	<span>2/2 checks passed</span>	<a href="#">View alarms</a>	us-east-1c	ec2-54-226-117-238.compute-1.amazonaws.com	<a href="#">Launch</a>
nagios-9	i-0fd2965bf41ae9cd0	<span>Running</span>	t2.micro	<span>2/2 checks passed</span>	<a href="#">View alarms</a>	us-east-1c	ec2-18-234-72-188.compute-1.amazonaws.com	
ubuntu-client10	i-016e919a97365096a	<span>Running</span>	t2.micro	<span>Initializing</span>	<a href="#">View alarms</a>	us-east-1c	ec2-54-80-53-159.compute-1.amazonaws.com	

Your Ubuntu client instance gets created along with the Nagios host instance.

Step 2: Click on the instance ID of your nagios-server instance and click on ‘Connect’. Then, click on ‘SSH client’ and copy the command under ‘Example’. Then, open the terminal in the folder where the .pem file for your instance’s key pair is located and paste the SSH command that you just copied. This connects your instance to your local terminal using SSH.

Step 3: ps -ef | grep nagios

Run the above command on the nagios-host instance. This verifies whether the nagios service is running or not.

```
[ec2-user@ip-172-31-35-113 ~]$ ps -ef | grep nagios
nagios  64399      1  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios  64401  64399  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios  64402  64399  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios  64403  64399  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios  64404  64399  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios  64447  64399  0 13:48 ?        00:00:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ec2-user  65271  65245  0 14:01 pts/0    00:00:00 grep --color=auto nagio
[ec2-user@ip-172-31-35-113 ~]$ |
```

Step 4: sudo su

mkdir -p /usr/local/nagios/etc/objects/monitorhosts

mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts

This makes you the root user and creates two folders with the above paths.

```
[ec2-user@ip-172-31-35-113 ~]$ sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
[root@ip-172-31-35-113 ec2-user]# |
```

Step 5: We need to create a config file in this folder. So, copy the contents of the existing localhost config to the new file ‘linuxserver.cfg’.

cp /usr/local/nagios/etc/objects/localhost.cfg

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

```
[root@ip-172-31-88-33 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhos
ts/linuxhosts/linuxserver.cfg |
```

Step 6: We need to make some changes in this config file. Open it using nano editor:-  
nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

1. Change hostname and alias from ‘hostname’ to ‘linuxserver’.
2. Change address to the public ip address of the ubuntu-client instance.

```
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       linuxserver
    alias           linuxserver
    address         54.80.53.159
}
```

Change hostgroup\_name to ‘linux-servers1’.

```
define hostgroup {

    hostgroup_name   linux-servers1      ; The name of the hostgroup
    alias            Linux Servers        ; Long name of the group
    members          linuxserver         ; Comma separated list of hosts that belong to this group
}
```

Change all the subsequent occurrences of hostname in the file from ‘localhost’ to ‘linuxserver’.

Step 7: Open the Nagios config file using the following command:

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

Then, add the following line to the config file:

cfg\_dir=/usr/local/nagios/etc/objects/monitorhosts/

```
GNU nano 5.8                               /usr/local/nagios/etc/nagios.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

# Definitions for monitoring a Windows machine
#cfg_file=/usr/local/nagios/etc/objects/windows.cfg

# Definitions for monitoring a router/switch
#cfg_file=/usr/local/nagios/etc/objects/switch.cfg

# Definitions for monitoring a network printer
#cfg_file=/usr/local/nagios/etc/objects/printer.cfg

# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

#cfg_dir=/usr/local/nagios/etc/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers
cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
```

Step 8: Now we verify the configuration files and check that they contain no errors using the following command:

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

```
[root@ip-172-31-35-113 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

Website: https://www.nagios.org
Reading configuration data...
  Read main config file okay...

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
```

Step 9: Once the files are verified and it is confirmed that there are no errors, we must restart the server.

```
service nagios restart
```

```
[root@ip-172-31-88-33 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
```

## Step 10: systemctl status nagios

Using the above command, we check the status of the nagios server and ensure that it is active (running).

```
[root@ip-172-31-88-33 ec2-user]# 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 Sun 2024-09-29 12:11:40 UTC; 1min 12s ago
     Docs: https://www.nagios.org/documentation
 Process: 70244 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0>
Process: 70245 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SU>
Main PID: 70246 (nagios)
   Tasks: 6 (limit: 1112)
  Memory: 4.0M
    CPU: 38ms
   CGroup: /system.slice/nagios.service
           ├─70246 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
           ├─70247 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70248 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70249 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           ├─70250 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
           └─70251 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully bound to port 5667
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: core query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: echo service query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: qh: help for the query handler registered
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Successfully registered manager as @wproc with query
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70250;pid=70250
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70249;pid=70249
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70248;pid=70248
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: wproc: Registry request: name=Core Worker 70247;pid=70247
Sep 29 12:11:40 ip-172-31-88-33.ec2.internal nagios[70246]: Successfully launched command file worker with pid 70251
```

Step 11: Connect your ubuntu-client instance to your local terminal using SSH in the same way as you connected the nagios-host instance to your local terminal using SSH (follow Step 2)

```
PS C:\Users\ADMIN> cd .\Downloads\
PS C:\Users\ADMIN\Downloads> ssh -i "nagios-9key.pem" ubuntu@ec2-54-80-53-159.compute-1.amazonaws.com
The authenticity of host 'ec2-54-80-53-159.compute-1.amazonaws.com (54.80.53.159)' can't be established.
ED25519 key fingerprint is SHA256:ictbLzP1p2YFXqDXkAH4CzoAWhCOQfhqnoXYFJGZ5q8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-80-53-159.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1016-aws x86_64)

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

System information as of Mon Oct  7 14:15:43 UTC 2024

  System load:  0.0          Processes:          104
  Usage of /:   22.8% of 6.71GB   Users logged in:      0
  Memory usage: 20%            IPv4 address for enX0: 172.31.38.70
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

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

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

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

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

ubuntu@ip-172-31-38-70:~$ |
```

Step 12: On your ubuntu-client instance, run the following commands:-

```
sudo apt update -y  
sudo apt install gcc -y  
sudo apt install -y nagios-nrpe-server nagios-plugins
```

The above commands check for any new updates and then install gcc, Nagios NRPE server and Nagios plugins.

```
ubuntu@ip-172-31-38-70:~$ 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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]  
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [382 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]  
  
Setting up samba-dsdb-modules:amd64 (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up libsmbclient0:amd64 (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up libcups2t64:amd64 (2.4.7-1.2ubuntu7.3) ...  
Setting up python3-samba (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up smbclient (2:4.19.5+dfsg-4ubuntu9) ...  
Setting up samba-common-bin (2:4.19.5+dfsg-4ubuntu9) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...  
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-172-31-38-70:~$ |
```

Step 13: Run the following command:

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

The above command opens the NRPE config file. Here, we need to add the public IP address of our host nagios-host instance to the NRPE configuration file.

Under allowed\_hosts, add the nagios-host public IPv4 address.

```

GNU nano 7.2                                         /etc/nagios/nrpe.cfg *
# 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=127.0.0.1,18.234.72.188|


# COMMAND ARGUMENT PROCESSING
# This option determines whether or not the NRPE daemon will allow clients
# to specify arguments to commands that are executed. This option only works
# if the daemon was configured with the --enable-command-args configure script
# option.

```

Step 14: Navigate to the Nagios dashboard. Click on 'hosts'. We see that linuxserver has been added as a host.

The screenshot shows the Nagios web interface. The top navigation bar includes links for Home, Documentation, and various status reports. The main content area is titled "Host Status Details For All Host Groups". It lists two hosts: "linuxserver" and "localhost", both of which are marked as "UP" with green icons. The "Status Information" column for "linuxserver" indicates "PING OK - Packet loss = 0%, RTA = 0.94 ms". The sidebar on the left contains links for General, Current Status, Problems, Reports, and System.

Host	Status	Last Check	Duration	Status Information
linuxserver	UP	10-07-2024 14:19:04	0d 0h 8m 31s	PING OK - Packet loss = 0%, RTA = 0.94 ms
localhost	UP	10-07-2024 14:17:40	0d 0h 34m 17s	PING OK - Packet loss = 0%, RTA = 0.04 ms

Click on ‘linuxserver’. Here, we can access all information about the ‘linuxserver’ host.

**Host Information**

Last Updated: Mon Oct 7 14:23:09 UTC 2024  
Last Check: Mon Oct 7 14:23:09 UTC 2024  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

**Host**  
**linuxserver**  
(linuxserver)

**Member of**  
linux-servers1

**54.80.53.159**

**Host State Information**

**Host Status:** UP (for 0d 0h 0m 5s)  
**Status Information:** PING OK - Packet loss = 0%, RTA = 0.9 ms  
**Performance Data:** rta=0.94200ms;3000.000000;5000.000000;0.000000 pi=0%;80;100.0  
**Current Attempt:** 1/1 (HARD state)  
**Last Check Time:** 10-07-2024 14:19:04  
**Check Interval:** ACTIVE  
**Check Latency / Duration:** 0.000 / 4.079 seconds  
**Next Scheduled Active Check:** 10-07-2024 14:24:04  
**Last State Change:** 10-07-2024 14:14:04  
**Last Notification:** N/A (notification 0)  
**In Host Flapping?** NO (0.00% state change)  
**In Scheduled Downtime?** NO  
**Last Update:** 10-07-2024 14:23:03 (0d 0h 0m 6s ago)

**Host Commands**

- Locate host on map
- Disable active checks of this host
- Ré-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 Delete all comments

Entry Time Author Comment Comment ID Persistent type Expires Actions

This host has no comments associated with it

Click on ‘Services’. Here, we can see all the services that are being monitored by ‘linuxserver’.

**Current Network Status**

Last Updated: Mon Oct 7 14:23:57 UTC 2024  
Updated every 90 seconds  
Nagios® Core™ 4.5.5 - www.nagios.org  
Logged in as nagiosadmin

**Host Status Totals**

Up	2
Down	0
Unreachable	0
Pending	0

All Problems All Types

**Service Status Totals**

Ok	6
Warning	1
Unknown	0
Critical	1
Pending	0

All Problems All Types

**Service Status Details For All Hosts**

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-07-2024 14:18:55	0d 0h 35m 2s	1/1	OK - load average: 0.00, 0.00, 0.00
localhost	Current Users	OK	10-07-2024 14:19:33	0d 0h 34m 24s	1/1	USERS OK - 0 users currently logged in
localhost	HTTP	WARNING	10-07-2024 14:23:09	0d 0h 30m 47s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time
localhost	PING	OK	10-07-2024 14:20:48	0d 0h 33m 9s	1/1	PING OK - Packet loss = 0%, RTA = 0.3 ms
localhost	Root Partition	OK	10-07-2024 14:21:25	0d 0h 32m 32s	1/1	DISK OK - free space / 6116 MB (75.36% inode=98%)
localhost	SSH	OK	10-07-2024 14:22:03	0d 0h 31m 54s	1/1	SSH OK - OpenSSH_8.7 (protocol 2.0)
localhost	Swap Usage	CRITICAL	10-07-2024 14:20:40	0d 0h 28m 17s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size
localhost	Total Processes	OK	10-07-2024 14:23:18	0d 0h 30m 39s	1/1	PROCS OK: 34 processes with STATE = R/Z/D/T

Results 1 - 8 of 8 Matching Services

**Conclusion :** In this experiment, we explored how to monitor ports, services, and both Windows and Linux servers using Nagios. To achieve this, we launched a Nagios-hosted EC2 Linux instance, which served as the platform for running the Nagios server and dashboard. Additionally, we deployed an Ubuntu client instance that connected to the Nagios host.

We configured the necessary settings on the Linux instance, including adding the Ubuntu client’s public IP address. Similarly, we made configuration changes on the Ubuntu client, where we added the IP address of the Nagios-hosted Linux instance. We also ensured that the Linux server instance was permitted as an authorized host on the Ubuntu client. After restarting the NRPE service, we verified that the ‘linuxserver’ host was successfully added for monitoring.

**AIM:**To understand AWS Lambda, its workflow, various functions and create your first Lambda

functions using Python / Java / Nodejs.

STEP1: Go on your AWS console account and search for lambda and then go on create function Select the author from scratch, add function name and then, choose a runtime env for your function, under the dropdown, you can see all the options AWS supports, Python, Nodejs, .NET and Java being the most popular ones.

The screenshot shows the AWS Lambda 'Create function' wizard. At the top, there's a navigation bar with the AWS logo, 'Services' (selected), a search bar, and a 'Create function' button. Below the navigation, the path 'Lambda > Functions > Create function' is shown. The main section is titled 'Create function' with an 'Info' link. A sub-instruction says 'Choose one of the following options to create your function.' There are four options:

- Author from scratch: Start with a simple Hello World example.
- Use a blueprint: Build a Lambda application from sample code and configuration presets for common use cases.
- Container image: Select a container image to deploy for your function.
- Browse serverless app repository: Deploy a sample Lambda application from the AWS Serverless Application Repository.

Below these options is a 'Basic information' tab, which is currently selected. The 'Basic information' section contains the following fields:

- Function name:** A text input field containing 'lamdaexp11'. A placeholder text says 'Enter a name that describes the purpose of your function.'
- Runtime:** A dropdown menu set to 'Python 3.12' with a refresh icon next to it. A note below says 'Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.'
- Architecture:** A dropdown menu set to 'x86\_64' with a refresh icon next to it. A note below says 'Choose the instruction set architecture you want for your function code.'
- Permissions:** A note stating '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.'

At the bottom of the 'Basic information' section, there's a link '▶ Change default execution role'.

STEP 2: After the function is created successfully go on code write the default code and then configure them.

The screenshot shows the AWS Lambda console. At the top, a success message says: "Successfully created the function lambdaexp11. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." Below this, the function name "lambdaexp11" is displayed. On the right, there are buttons for "Throttle", "Copy ARN", and "Actions". Under "Function overview", there are tabs for "Diagram" (selected) and "Template". The diagram shows a single function box labeled "lambdaexp11" with a "Layers" section below it showing "(0)". There are buttons "+ Add trigger" and "+ Add destination". On the right side, there are fields for "Description", "Last modified" (16 seconds ago), and "Function ARN" (arn:aws:lambda:eu-north-1:026090558619:function:lambdaexp11). Buttons "Export to Application Composer" and "Download" are also present.

The screenshot shows the AWS Lambda Code Source editor. At the top, tabs include "Code" (selected), "Test", "Monitor", "Configuration", "Aliases", and "Versions". Below this, a toolbar has "Upload from" and other options. The main area shows a code editor with a file named "lambda\_function.py" containing the following Python code:

```
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     }
9
```

STEP 3: Then go on edit basic settings and add the description and then save it .

The screenshot shows the AWS Lambda Configuration page. At the top, tabs include "Code", "Test", "Monitor", "Configuration" (selected), "Aliases", and "Versions". A sidebar on the left lists "General configuration", "Triggers", "Permissions", "Destinations", "Function URL", "Environment variables", "Tags", and "VPC". The main area displays "General configuration" settings:

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout	SnapStart	Info
0 min 3 sec	None	

An "Edit" button is located in the top right corner of the configuration table.

## Edit basic settings

**Basic settings** Info

Description - *optional*  
D15C

**Memory** Info  
Your function is allocated CPU proportional to the memory configured.  
128 MB  
Set memory to between 128 MB and 10240 MB

**Ephemeral storage** Info  
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing

512 MB  
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

**SnapStart** Info  
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations .

STEP 4: Click on “use an existing role” option and then ahead add the role and save it.

**SnapStart** Info  
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations .

None

Supported runtimes: Java 11, Java 17, Java 21.

**Timeout**  
0 min 1 sec

**Execution role**  
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console .

Use an existing role  
 Create a new role from AWS policy templates

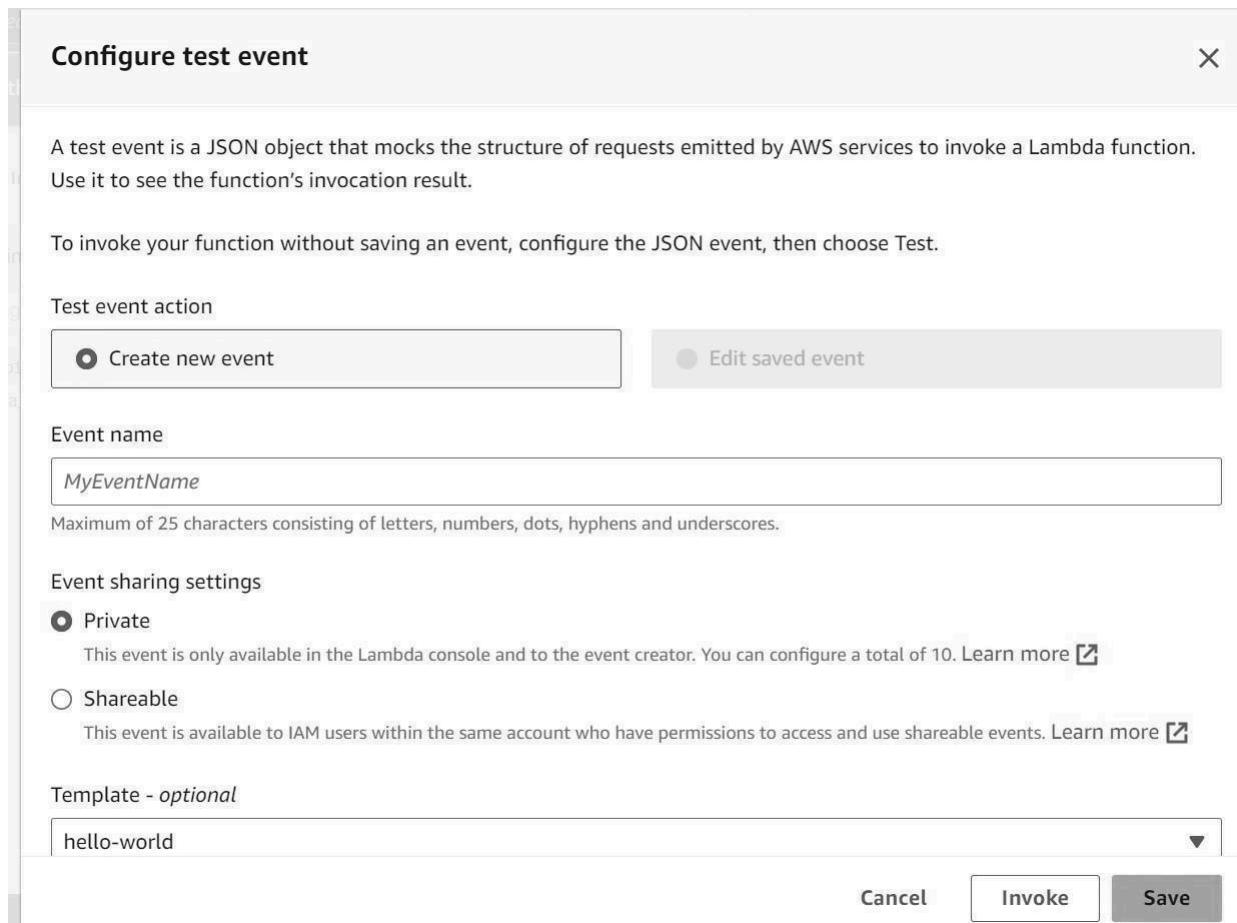
**Existing role**  
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

service-role/lamdaexp11-role-vj5j9g95

View the lamdaexp11-role-vj5j9g95 role on the IAM console.

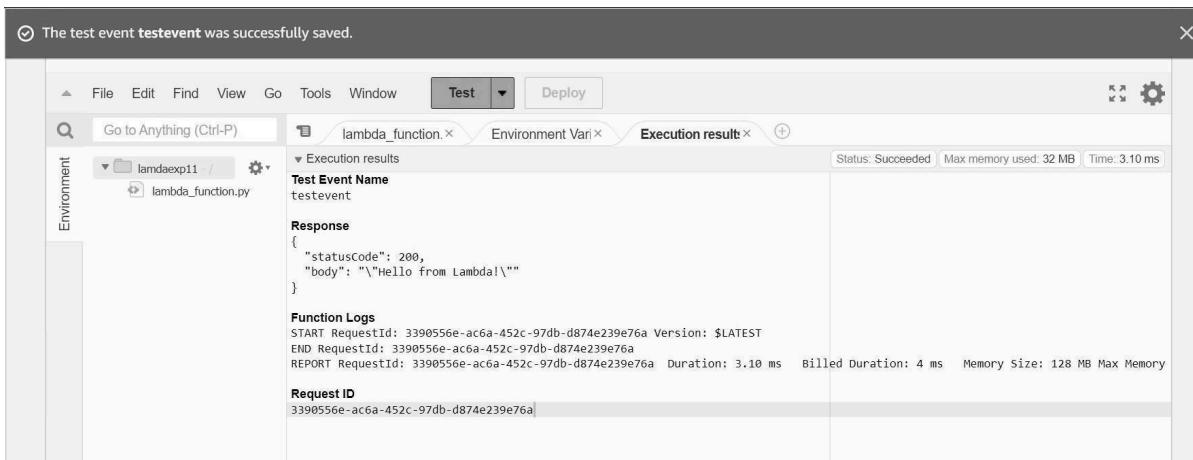
**Cancel** **Save**

STEP 5: Go on configure test event click on “create new event” edit the event sharing accordingly and select hello world template for template option and then save it.



STEP 6: Click on the test and test the code.

STEP 7: The function is successfully added .



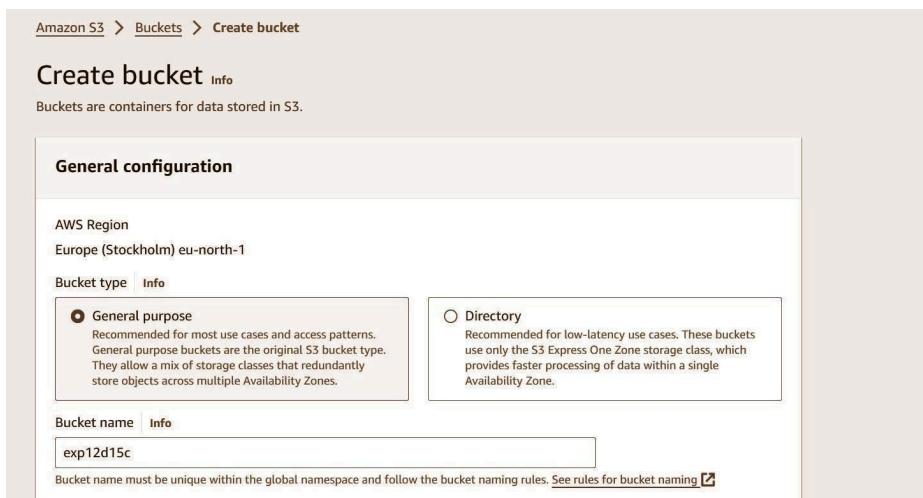
The screenshot shows the AWS Lambda function editor interface. At the top, a message says "Successfully updated the function lambdaexp11." Below the header is a toolbar with File, Edit, Find, View, Go, Tools, Window, Test, Deploy, and settings icons. The main area has tabs for Go to Anything (Ctrl-P), lambda\_function (selected), Environment Variables, and Execution results. On the left, there's a sidebar labeled "Environment" with a dropdown menu. The central code editor shows the following Python code:

```
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     }
9
```

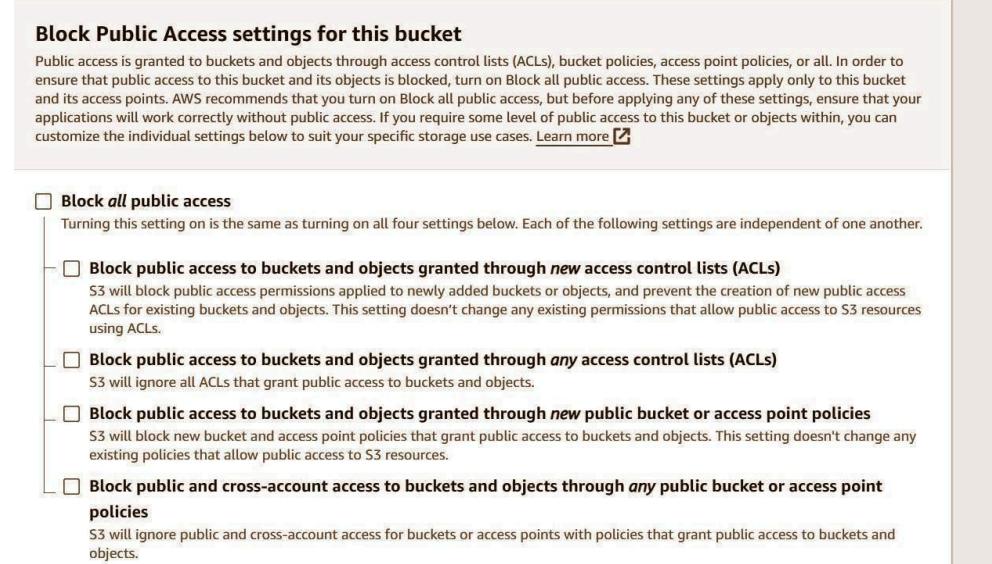
Conclusion: In conclusion, the experiment successfully involved the creation, coding, and deployment of AWS Lambda function. By writing and refining the source code, we demonstrated the ability to implement specific functionality within the Lambda environment. The successful testing of the function confirmed its operational integrity and effectiveness in executing the desired tasks.

**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 STEPS:

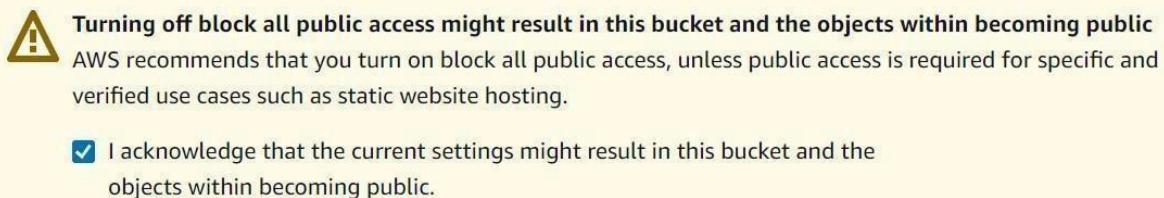
1. Create a S3 bucket and give it a bucket name



2. Allow public access to the bucket as we are going to add this bucket as a trigger for our lambda function



3. Give confirmation that you want to allow full public access and create the bucket



#### 4. You will see the confirmation that the bucket is created successfully

Successfully created bucket "exp12d15c"  
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

**Account snapshot - updated every 24 hours** [All AWS Regions](#)

Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

[View Storage Lens dashboard](#)

General purpose buckets | Directory buckets

#### 5. Now we need to upload something in the bucket so click on the upload button and add a file

Amazon S3 > Buckets > exp12d15c

**exp12d15c** [Info](#)

Objects Properties Permissions Metrics Management Access Points

**Objects (0) Info**

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

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](#)

[Find objects by prefix](#)

< 1 > [...](#)

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

[Upload](#)

#### 6. I have added a .png extension file; You can upload a .txt file as well

## 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, 293.3 KB)			
<span style="float: right;">Remove</span> <span style="float: right;">Add files</span> <span style="float: right;">Add folder</span>			
All files and folders in this table will be uploaded.			
<input type="text"/> Find by name <span style="float: right;">&lt; 1 &gt;</span>			
<input type="checkbox"/>	Name	Folder	Type
<input type="checkbox"/>	AppBar( title Text('Guidelines'), ),....	-	image/png

## 7. Here you can see the confirmation that the upload was a success

⌚ Upload succeeded  
View details below.

Summary		
Destination	Succeeded	Failed
s3://exp12d15c	⌚ 1 file, 293.3 KB (100.00%)	⌚ 0 files, 0 B (0%)

Files and folders   Configuration

Files and folders (1 Total, 293.3 KB)						
<input type="text"/> Find by name <span style="float: right;">&lt; 1 &gt;</span>						
Name	Folder	Type	Size	Status	Error	
AppBar( title...	-	image/png	293.3 KB	⌚ Succeeded	-	

- Now go back to the aws dashboard and search for lambda function service, Open the function we created in experiment 10. We are going to add this bucket as a trigger to this function
- On the function overview section of the dashboard you can see the “Add trigger” button. Click on that

Lambda > Functions > lamdaexp11

## lamdaexp11

Throttle Copy ARN Actions ▾

▼ Function overview Info

Diagram Template

 lamdaexp11

 Layers (0)

+ Add trigger + Add destination

Description  
D15C

Last modified  
16 minutes ago

Function ARN  
 arn:aws:lambda:eu-north-1:026090558619:function:lamdaexp11

Function URL Info

-

Export to Application Composer Download ▾



10. It will lead you to the trigger configuration tab; Where you have to select the service and the bucket you created. Add the required configuration information and then

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: eu-north-1

Event types

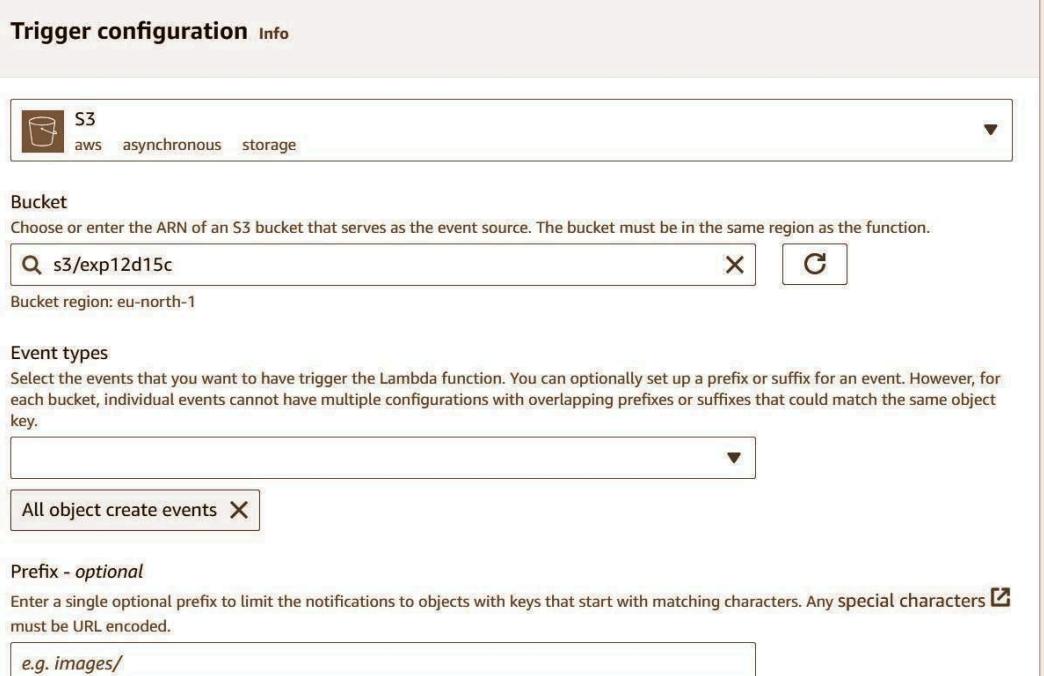
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

▾

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.



save.

11. Here you can see we have the confirmation message as well the the s3 bucket added to our triggers

The trigger exp12d15c was successfully added to function lamdaexp11. The function is now receiving events from the trigger.

**Function overview**

Description: D15C

Last modified: 18 minutes ago

Function ARN: arn:aws:lambda:eu-north-1:026090558619:function:lamdaexp11

Function URL: -

12. Test the code by clicking on the Test tab ; Here as you can see our code ran successfully

Execution result:

Status: Succeeded | Max memory used: 32 MB | Time: 1.97 ms

Test Event Name: testevent

Response:

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

Function Logs:

```
START RequestId: 6e57026c-6bfe-41fe-898d-2cdbe72fb1a3 Version: $LATEST
END RequestId: 6e57026c-6bfe-41fe-898d-2cdbe72fb1a3
REPORT RequestId: 6e57026c-6bfe-41fe-898d-2cdbe72fb1a3 Duration: 1.97 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory
```

Request ID: 6e57026c-6bfe-41fe-898d-2cdbe72fb1a3

Conclusion: In conclusion, the experiment successfully demonstrated the integration of an S3 bucket with an AWS Lambda function as a trigger. By creating the S3 bucket and configuring it to invoke the Lambda function upon object uploads, we established a seamless workflow for automated processing.