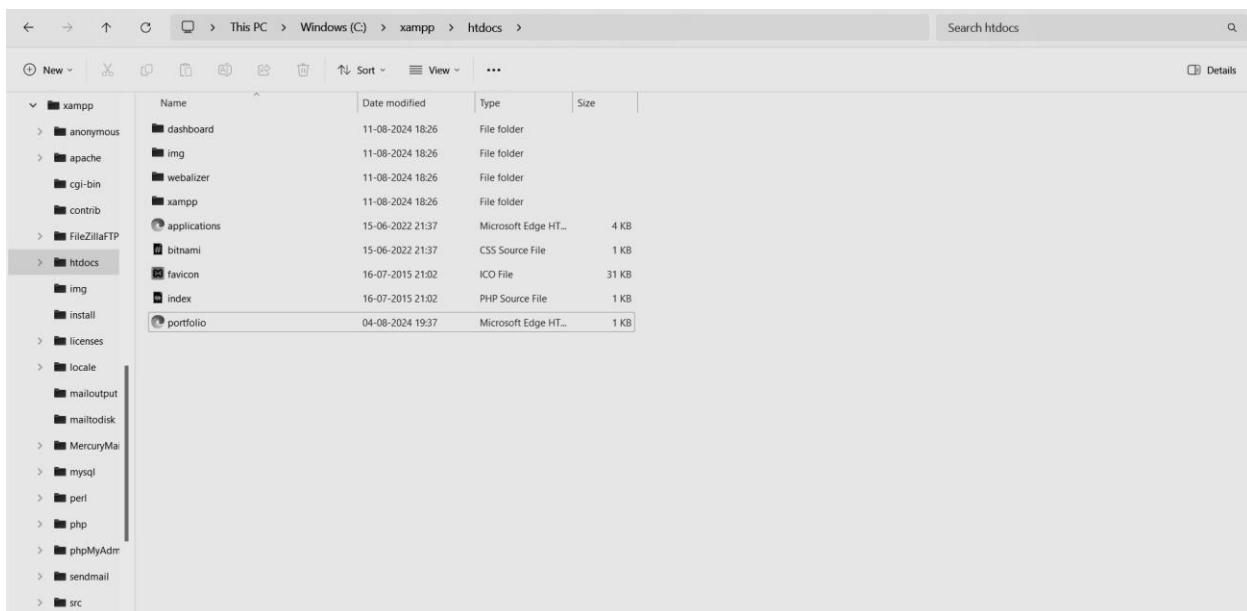


a)Hosting of html file on a local virtual machine using Xampp

- Create an portfolio file with name portfolio.html.

```
<> portfolio.html > html > body
1   <!DOCTYPE html>
2   <html lang="en">
3     <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <title>Portfolio</title>
7     </head>
8     <body>
9       <h1>Bhushan Malpani</h1>
10      <br>
11      |
12      <h2>Education</h2>
13      <ul>
14        <li><b>10th class (CBSE)</b>: 94.8%</li>
15        <li><b>12th class (HSC)</b>: 90%</li>
16        <li><b>Bachelor in Engineering(IT)</b></li>
17      </ul>
18      <h2>Skills</h2>
19      <ul>
20        <li>HTML,CSS,JS</li>
21        <li>Java,C++</li>
22      </ul>
23      <h2>Hobbies</h2>
24      <ul>
25        <li>Cricket</li>
26        <li>Music</li>
27      </ul>
28      <h2>Contact Me</h2>
29      <ul>
30        <li><b>Email</b>:bhushanmalpani45@gmail.com</li>
31        <li><b>Contact No.</b>:9423244006/</li>
32      </ul>
33    </body>
34  </html>
```

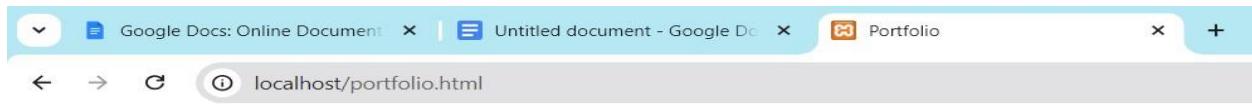
- Save this file in htdocs file in Xampp folder in C drive:



3. Make sure Xampp is installed and then start Xampp control panel and start the module Apache and Mysql.



4. Access the content of the portfolio page using **localhost/portfolio.html** in any web browser. Your portfolio should be hosted successfully on local machine using Xampp.



Bhushan Malpani

Education

- 10th class (CBSE): 94.8%
- 12th class (HSC): 90%
- Bachelor in Engineering(IT)

Skills

- HTML,CSS,JS
- Java,C++

Hobbies

- Cricket
- Music

Contact Me

- Email:bhushanmalpani45@gmail.com
- Contact No.:9423244006/;

Static Hosting Using AWS

Navigate to S3 Service:

In the AWS Management Console, select Services and then S3 under Storage.

The screenshot shows the 'Create bucket' page in the AWS Management Console. The navigation bar at the top shows 'Amazon S3 > Buckets > Create bucket'. The main title is 'Create bucket' with an 'Info' link. A sub-instruction says 'Buckets are containers for data stored in S3.' Below this is a section titled 'General configuration'.

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

Bucket type: General purpose Directory - New

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.

Bucket name: [Info](#)

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
[Choose bucket](#)

Format: s3://bucket/prefix

Create a Bucket:

- Click on **Create bucket**.
- **Bucket name:** Enter a unique name that is globally unique ● **Region:** Select a region (usually closest to your users).
- **Block Public Access Settings:** Uncheck **Block all public access**. Confirm the warning that appears about the public access settings. ● Click **Create bucket**.

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with navigation links like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and a section for Block Public Access settings. Below that is a Storage Lens section. At the bottom of the sidebar are Feature spotlight and AWS Marketplace for S3 links.

The main content area is titled "Amazon S3" and "Amazon S3". It features an "Account snapshot - updated every 24 hours" card with a "View Storage Lens dashboard" button. Below this is a tab bar with "General purpose buckets" (selected) and "Directory buckets".

The "General purpose buckets" table lists one item:

Name	AWS Region	IAM Access Analyzer	Creation date
www.bhushan.com	US East (N. Virginia) us-east-1	View analyzer for us-east-1	August 4, 2024, 19:26:54 (UTC+05:30)

At the top right of the table are buttons for "Create bucket", "Copy ARN", "Empty", and "Delete". There are also navigation arrows and a refresh icon above the table.

Upload Files:

- Click on **Upload**.
- Drag and drop your static website files or use the **Add files** and **Add folder** buttons to upload your HTML, CSS, JavaScript, and image files. • Click **Upload** to start the upload process.

The screenshot shows the AWS S3 'Upload' interface. At the top, the navigation bar includes the AWS logo, Services, a search bar, and a [Alt+S] keyboard shortcut. Below the navigation, the path is shown as Amazon S3 > Buckets > www.bhushan.com > Upload. The main title is 'Upload' with an 'Info' link. A note below says: '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](#)'.

In the center, 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.' Below this is a table titled 'Files and folders (1 Total, 853.0 B)'. It contains one item: 'portfolio.html' (text/html). There are 'Remove', 'Add files', and 'Add folder' buttons above the table. A search bar labeled 'Find by name' is also present.

The 'Destination' section shows 's3://www.bhushan.com'. A green banner at the bottom left indicates 'Upload succeeded' with a link to 'View details below.' A note below the banner says: 'The information below will no longer be available after you navigate away from this page.'

The 'Summary' section provides an overview of the upload results:

Destination	Succeeded	Failed
s3://www.bhushan.com	1 file, 853.0 B (100.00%)	0 files, 0 B (0%)

Below the summary, there are tabs for 'Files and folders' and 'Configuration'. The 'Files and folders' tab is selected, showing the same table as the main area. The table has columns: Name, Folder, Type, Size, Status, and Error. The single entry is 'portfolio.html' (text/html, 853.0 B, Succeeded).

4. Configure Bucket for Static Website Hosting

1. Access Bucket Properties:

- Go to the **Properties** tab of your bucket.

Enable Static Website Hosting:

- Scroll down to **Static website hosting** and click **Edit**.
- Select **Enable**.
- **Index document:** Enter `index.html` (or the name of your main HTML file).
- **Error document:** Optionally, enter `error.html` (for handling errors).
- Click **Save changes**.

The screenshot shows the 'Edit static website hosting' configuration page in the Amazon S3 console. At the top, the breadcrumb navigation shows: Amazon S3 > Buckets > www.bhushan.com > Edit static website hosting. The main title is 'Edit static website hosting' with an 'Info' link. Below it, the section 'Static website hosting' is titled 'Static website hosting' with the sub-instruction 'Use this bucket to host a website or redirect requests.' followed by a 'Learn more' link. There are two radio button options: 'Disable' (unchecked) and 'Enable' (checked). The 'Enable' option is selected. Under 'Hosting type', there are two options: 'Host a static website' (selected, checked) and 'Redirect requests for an object' (unchecked). The 'Host a static website' option has a sub-instruction 'Use the bucket endpoint as the web address.' followed by a 'Learn more' link. A callout box contains the 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.' Below this, the 'Index document' section is titled 'Index document' with the sub-instruction 'Specify the home or default page of the website.' It contains a text input field containing 'index.html'. The 'Error document - optional' section is shown below it.

The screenshot shows the AWS Lambda function configuration interface. The top navigation bar includes the AWS logo, 'Services' (selected), a search bar, and a keyboard shortcut '[Alt+S]'. Below the navigation is a section titled 'Specify the home or default page of the website.' with a text input field containing 'portfolio.html'. Underneath is an 'Error document - optional' section with a text input field containing 'error.html'. A 'Redirection rules - optional' section follows, with a note about redirection rules being written in JSON and a link to 'Learn more'. A large JSON editor area contains a single character '1'. At the bottom of the editor are status indicators: 'JSON', 'Ln 1, Col 1', 'Errors: 0', 'Warnings: 0', and a refresh icon. Below the editor are 'Cancel' and 'Save changes' buttons.

Access Permissions Tab: • Go to the

Permissions tab of your bucket.

The screenshot shows the 'Permissions' tab of an S3 bucket named 'www.bhushan.com'. The top navigation bar shows 'Amazon S3 > Buckets > www.bhushan.com'. The 'Permissions' tab is selected. Below the tabs is a 'Permissions overview' section with an 'Access finding' summary. It mentions IAM external access analyzers and provides a link to 'How IAM analyzer findings work'. There is also a link to 'View analyzer for us-east-1'. The main content area is titled 'Block public access (bucket settings)'. It explains that public access is granted through various controls and recommends turning on 'Block all public access'. It includes a note about individual settings for objects. An 'Edit' button is located in the top right corner of this section. Below this is a 'Block all public access' section with a 'On' toggle switch and a link to 'Individual Block Public Access settings for this bucket'.

Click **Save** to apply the policy.

The screenshot shows the 'Edit bucket policy' page for the bucket 'www.bhushan.com'. The top navigation bar includes 'Amazon S3 > Buckets > www.bhushan.com > Edit bucket policy'. Below the navigation is a section titled 'Bucket policy' with a 'Info' link. A note states: 'The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts.' A 'Bucket ARN' field contains 'arn:aws:s3:::www.bhushan.com'. The 'Policy' section displays a JSON policy document:

```
1▼ {
2  "Version": "2012-10-17",
3▼ "Statement": [
4▼   {
5      "Sid": "PublicReadGetObject",
6      "Effect": "Allow",
7      "Principal": "*",
8      "Action": "s3:GetObject",
9      "Resource": "arn:aws:s3:::www.bhushan.com/*"
10    }
11  ]
12 }
13 |
```

Test Your Website:

- Use the **Endpoint URL** from the static website hosting configuration to access and test your website.

Check Functionality:

- Ensure that all website pages and resources load correctly.

The screenshot shows the 'Bucket Properties' page for the bucket 'www.bhushan.com'. The left sidebar includes 'Amazon S3', 'Buckets', 'Block Public Access settings for this account', 'Storage Lens', and 'Feature spotlight'. The main content area shows the following configuration sections:

- Object Lock**: Status is 'Disabled'.
- Requester pays**: Status is 'Disabled'. A note says: 'When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled.' An 'Edit' button is available.
- Static website hosting**: Status is 'Enabled'. A note says: 'Use this bucket to host a website or redirect requests.' An 'Edit' button is available. Sub-options include 'Hosting type: Bucket hosting' and 'Bucket website endpoint: http://www.bhushan.com.s3-website-us-east-1.amazonaws.com'.

Bhushan Malpani

Education

- 10th class (CBSE): 94.8%
- 12th class (HSC): 90%
- Bachelor in Engineering(IT)

Skills

- HTML,CSS,JS
- Java,C++

Hobbies

- Cricket
- Music

Contact Me

- Email:bhushanmalpani45@gmail.com
- Contact No.:9423244006/;

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

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

The screenshot shows the 'Create environment' wizard in the AWS Management Console. The first step, 'Details', is completed with a name of 'Bhushan'. The second step, 'New EC2 instance', is currently selected. It lists three instance types: t2.micro (selected), t3.small, and m5.large. There is also an option for 'Additional instance types'. The platform is set to 'Amazon Linux 2023' and the timeout is set to '30 minutes'.

Details

Name: Bhushan
Limit of 60 characters, alphanumeric, and unique per user.

Description - optional
Limit 200 characters.

Environment type [Info](#)
Determines what the Cloud9 IDE will run on.

New EC2 instance
Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

Existing compute
You have an existing instance or server that you'd like to use.

New EC2 instance

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

Network settings [Info](#)

Connection
How your environment is accessed.

AWS Systems Manager (SSM)
Accesses environment via SSM without opening inbound ports (no ingress).

Secure Shell (SSH)
Accesses environment directly via SSH, opens inbound ports.

▶ **VPC settings [Info](#)**

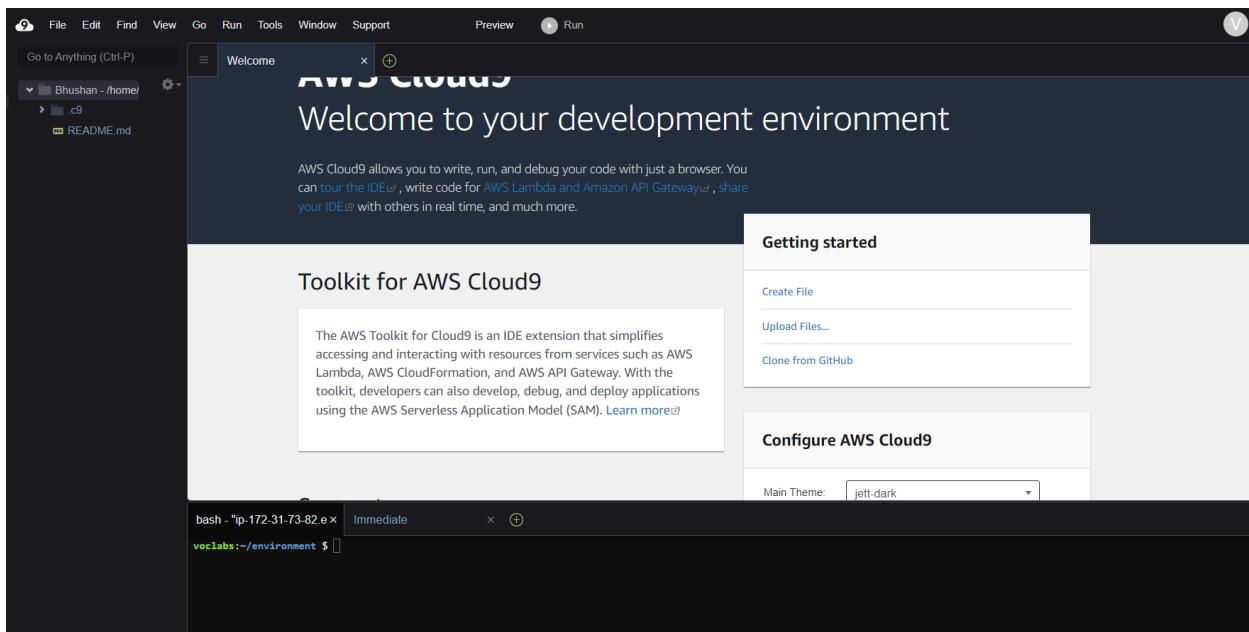
▶ **Tags - optional [Info](#)**
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

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 AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)
- 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 environments no longer require EC2 instances that block incoming traffic, you can delete these roles using the AWS IAM console. [Learn more](#)

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

2. We have successfully setup and launched our Cloud9 environment. Over here, we can build and develop programs as per our desire. We are also allowed to collaborate with multiple other users and access shared resources.



3. Moving on, we are supposed to create a new user. Give a suitable name to the user and decide the password for the same.

User details

User name
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

Provide user access to the AWS Management Console - *optional*
If you're providing console access to a person, it's a best practice [\[?\]](#) to manage their access in IAM Identity Center.

Console password
 Autogenerated password
You can view the password after you create the user.
 Custom password
Enter a custom password for the user.

 Show password

Users must create a new password at next sign-in - Recommended
Users automatically get the [IAMUserChangePassword](#) [\[?\]](#) policy to allow them to change their own password.

Info If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypaces, you can generate them after you create this IAM user. [Learn more](#) [\[?\]](#)

Cancel **Next**

Center, you can centrally manage user access to their AWS accounts and cloud applications.

I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keypaces, or a backup credential for emergency account access.

Console password
 Autogenerated password
You can view the password after you create the user.
 Custom password
Enter a custom password for the user.

 Show password

Users must create a new password at next sign-in - Recommended
Users automatically get the [IAMUserChangePassword](#) [\[?\]](#) policy to allow them to change their own password.

Info If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypaces, you can generate them after you create this IAM user. [Learn more](#) [\[?\]](#)

Cancel **Next**

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

NAME: Bhushan Malpani

DIV : D15C

Roll no: 33

Review and create

Step 4
Retrieve password

Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.

Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

User groups (1/1)			
Group name	Users	Attached policies	Created
MSBCLOUD9	0	-	2024-07-29 (Now)

Set permissions boundary - *optional*

Cancel Previous Next

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

User details

User name sahilmotiramani	Console password type Custom password	Require password reset Yes
------------------------------	--	-------------------------------

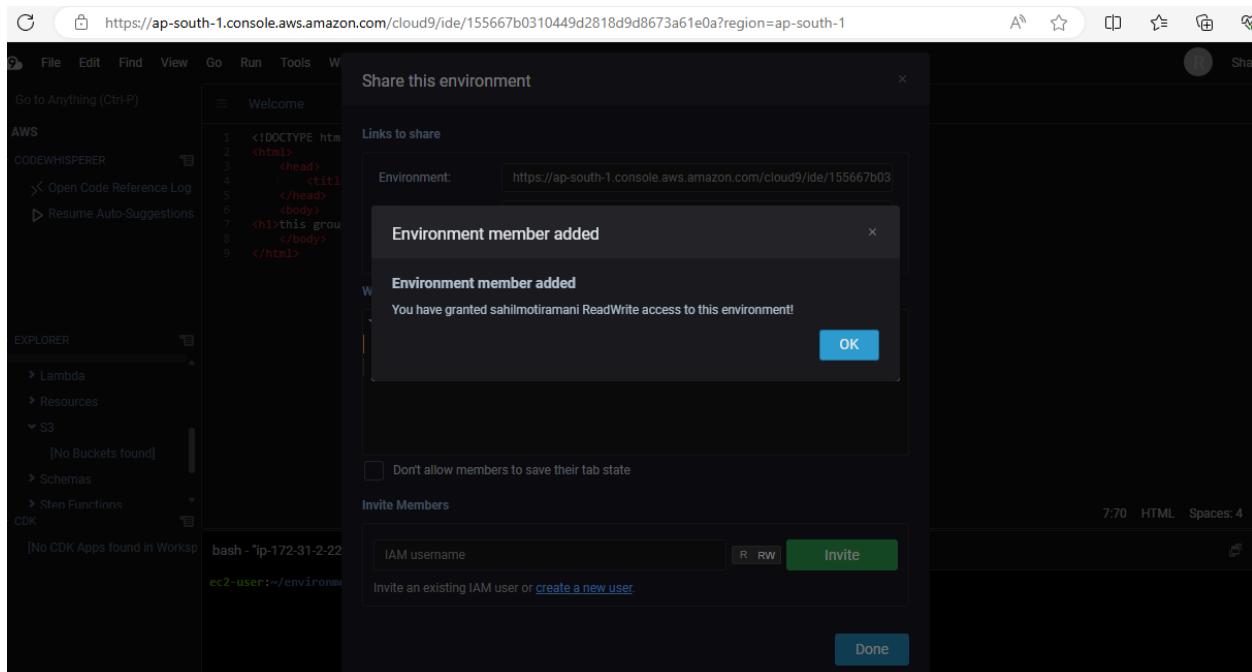
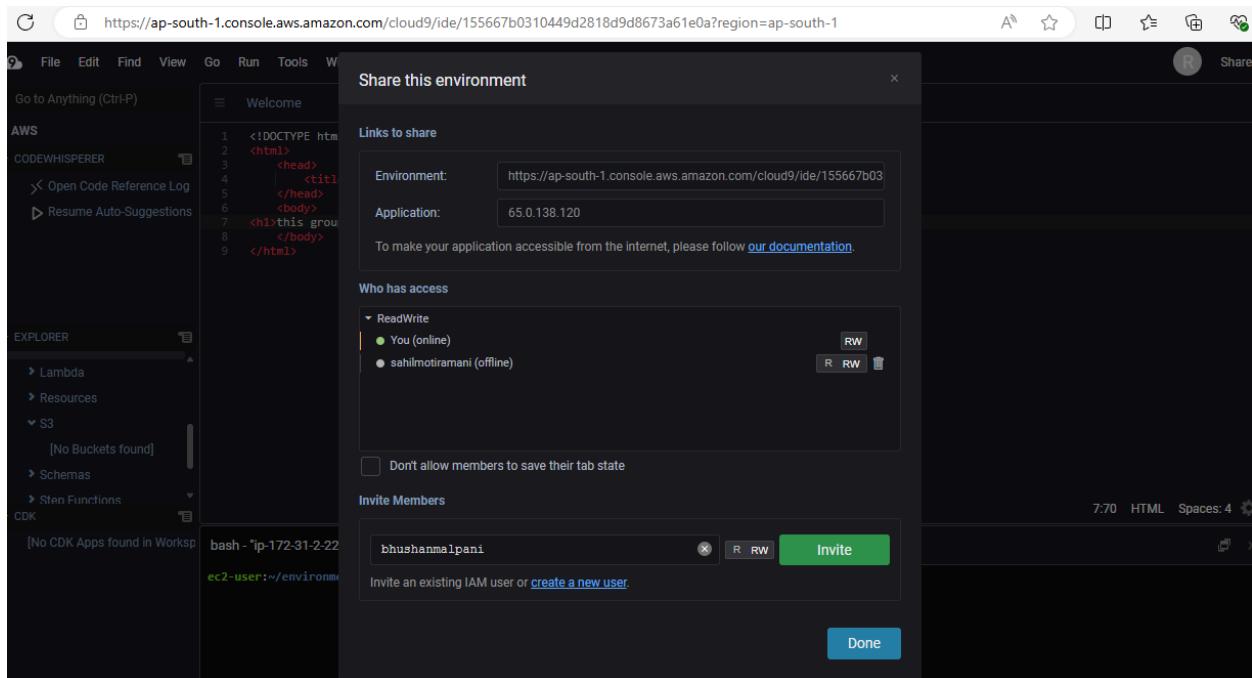
Permissions summary

Name	Type	Used as
IAMUserChangePassword	AWS managed	Permissions policy
MSBCLOUD9	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.

6. Go back to the cloud9 environment. Click on 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



7. Further, we are supposed to login from another browser using the credentials of the IAM user, so as to access the shared cloud9 environment with us.

These steps could not be completed because Cloud9 services have been disrupted and there is no access to the IAM user from the remote login

EXP 2

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

1. Create an Elastic Beanstalk environment. Give a suitable name to your environment.

Configure environment [Info](#)

Environment tier [Info](#)
Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

Web server environment
Run a website, web application, or web API that serves HTTP requests. [Learn more](#)

Worker environment
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. [Learn more](#)

Application information [Info](#)

Application name

Maximum length of 100 characters.

► Application tags (optional)

Environment information [Info](#)
Choose the name, subdomain and description for your environment. These cannot be changed later.

2. Select a suitable platform for your Elastic beanstalk environment. Here, we will select PHP file.

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

Platform version

4.3.1 (Recommended)

3. Next, we will have to give access to our Elastic beanstalk environment to carry out its tasks. For this, we have to grant certain permissions to the role, component or the entity that we will create.

Following our permissions that we are supposed to grant to our role.

- AWSElasticBeanStalkWebTier
- AWSElasticBeanStalkWorkerTier
- AWSElasticBeanStalkMulticontainerDocker

The screenshot shows the 'Configure service access' step of the AWS Elastic Beanstalk setup wizard. On the left, a sidebar lists optional steps: Step 2 (Configure service access), Step 3 (Set up networking, database, and tags), Step 4 (Configure instance traffic and scaling), Step 5 (Configure updates, monitoring, and logging), and Step 6 (Review). The main panel is titled 'Service access' and contains the following configuration:

- Service role:** A radio button is selected for "Use an existing service role". A dropdown menu shows "role1" and a clear icon.
- Existing service roles:** A dropdown menu shows "role1" and a clear icon.
- EC2 key pair:** A dropdown menu shows "Choose a key pair" and a clear icon.
- EC2 instance profile:** A dropdown menu shows "role1" and a clear icon.
- View permission details:** A button to view detailed IAM policy information.

At the bottom right, there are buttons for "Cancel", "Skip to review", "Previous", and a prominent orange "Next" button.

4. Click submit to create our Elastic beanstalk environment.

The screenshot shows the 'Platform software' configuration step of the AWS Elastic Beanstalk setup wizard. The configuration table includes the following rows:

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	Deactivated	minor
X-Ray enabled		
Deactivated		

Environment properties: A table showing environment properties with no defined values.

Key	Value
No environment properties	
There are no environment properties defined	

At the bottom right, there are buttons for "Cancel", "Previous", and a prominent orange "Submit" button.

Name:Bhushan Malpani

Roll No. 33

Div:D15C

5. When the creation is successful, we will see a message like this.

The screenshot shows the AWS Elastic Beanstalk console. On the left, a sidebar for the 'Elastic Beanstalk' service is visible with options like Applications, Environments, Change history, Application versions, Saved configurations, Environment: Advdevops33-env, Go to environment, Configuration, Events, Health, Logs, Monitoring, Alarms, Managed updates, and Tags. The main content area displays the 'Advdevops33-env' environment. A green banner at the top says 'Environment successfully launched.' Below it, the 'Environment overview' section shows Health (Warning), Environment ID (e-qupt3p8uap), Domain (Advdevops33-env.eba-icmyh99u.ap-south-1.elasticbeanstalk.com), and Application name (advdevops33). To the right, the 'Platform' section shows PHP 8.3 running on 64bit Amazon Linux 2023/4.3.1, Running version (empty), and Platform state (Supported). At the bottom, there are tabs for Events, Health, Logs, Monitoring, Alarms, Managed updates, and Tags, with the Events tab selected. An 'Events (10)' section is shown with a search bar.

Now, We need to create a pipeline

6. To create a pipeline go to services → pipeline→create pipeline

The screenshot shows the AWS CodePipeline service. On the left, a sidebar for 'Developer Tools' and 'CodePipeline' is visible with options like Source (CodeCommit), Artifacts (CodeArtifact), Build (CodeBuild), Deploy (CodeDeploy), Pipeline (CodePipeline), Getting started, Pipelines, and Settings. The main content area shows the 'Pipelines' page under 'CodePipeline'. It features a 'Pipelines info' section with a search bar and buttons for Notify, View history, Release change, Delete pipeline, and Create pipeline. Below is a table with columns: Name, Latest execution status, Latest source revisions, Latest execution started, and Most recent executions. The table currently has no data.

7. Give name to the pipeline

The screenshot shows the 'Choose pipeline settings' step of creating a new pipeline. On the left, a sidebar lists steps: Step 1 (Choose pipeline settings), Step 2 (Add source stage), Step 3 (Add build stage), Step 4 (Add deploy stage), and Step 5 (Review). The main area is titled 'Pipeline settings'. It includes fields for 'Pipeline name' (containing 'p1'), 'Pipeline type' (set to 'Queued (Pipeline type V2 required)'), and 'Execution mode' (set to 'Superseded'). A note at the bottom states: '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.'

8. Connect it to git hub

The screenshot shows the 'Create connection' step for GitHub. A blue banner at the top states: 'Beginning July 1, 2024, the console will create connections with codeconnections in the resource ARN. Resources with both service prefixes will continue to display in the console.' Below this, the title 'Connect to GitHub' is displayed. The 'GitHub connection settings' section contains a 'Connection name' field with 'Bhushan210104' entered. Under 'GitHub Apps', there is a search bar with '53711615', an 'X' button, and an 'Install a new app' button.

Name:Bhushan Malpani

Roll No. 33

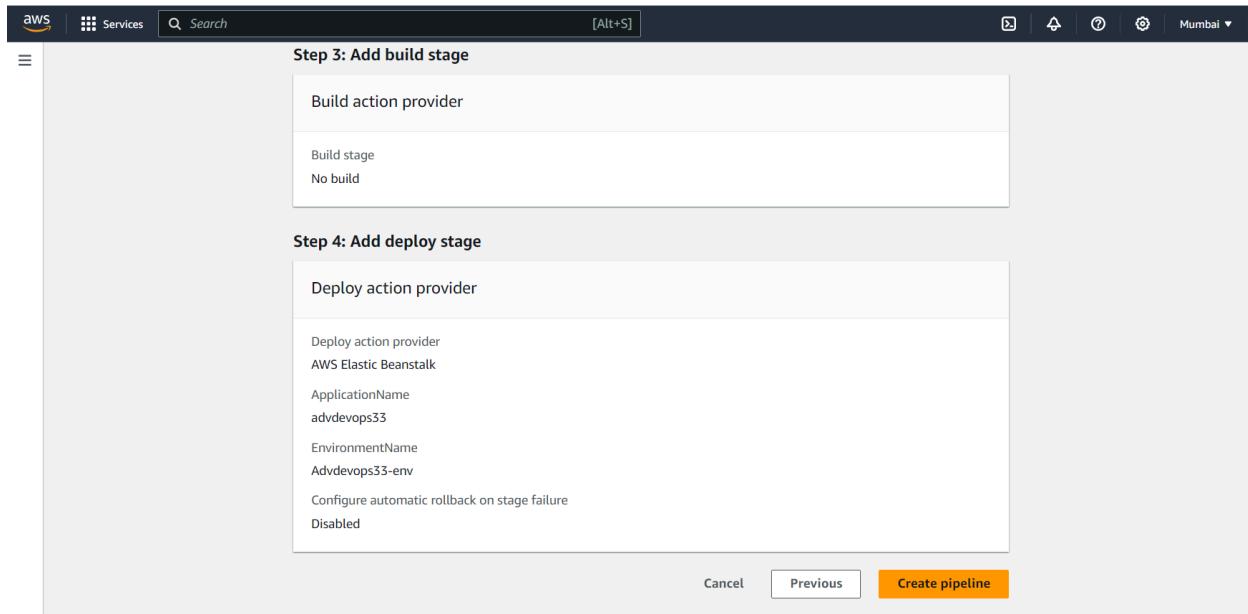
Div:D15C

9. After connecting it to github select the repository and configure other settings

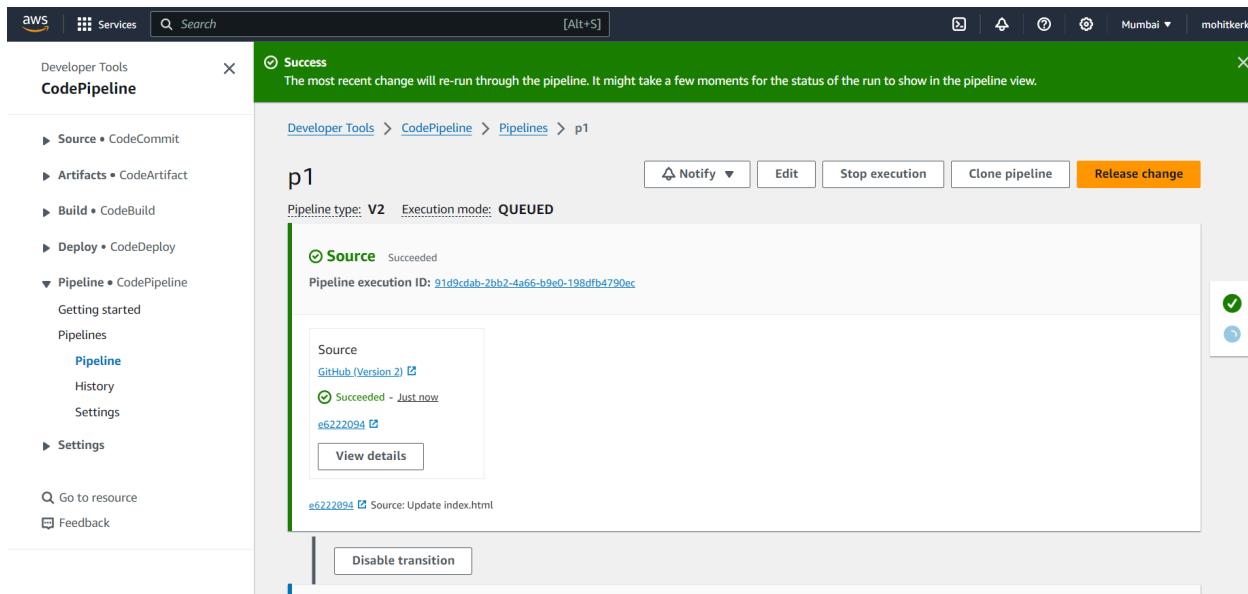
The screenshot shows the 'Connection' configuration page in the AWS CodePipeline console. At the top, there's a search bar and a 'Connect to GitHub' button. Below it, a green box indicates a successful connection with the message 'Ready to connect' and 'Your GitHub connection is ready for use.' The 'Repository name' field contains 'Bhushan210104/aws-codepipeline-s3-codedeploy-linux-2.0'. The 'Default branch' field contains 'master'. Under 'Output artifact format', the 'CodePipeline default' option is selected, with a note explaining it uses the default zip format for artifacts in the pipeline. There's also a 'Full clone' option which is not selected.

The screenshot shows the 'Step 5 Review' configuration page in the AWS CodePipeline console. It's a summary step where users can review their deployment configuration. The 'Deploy provider' is set to 'AWS Elastic Beanstalk', and the 'Region' is 'Asia Pacific (Mumbai)'. Under 'Input artifacts', there's a dropdown menu with a note about character limits. The 'Application name' is 'advdevops33', and the 'Environment name' is 'Advdevops33-env'. At the bottom, there are 'Cancel', 'Previous', and 'Next' buttons.

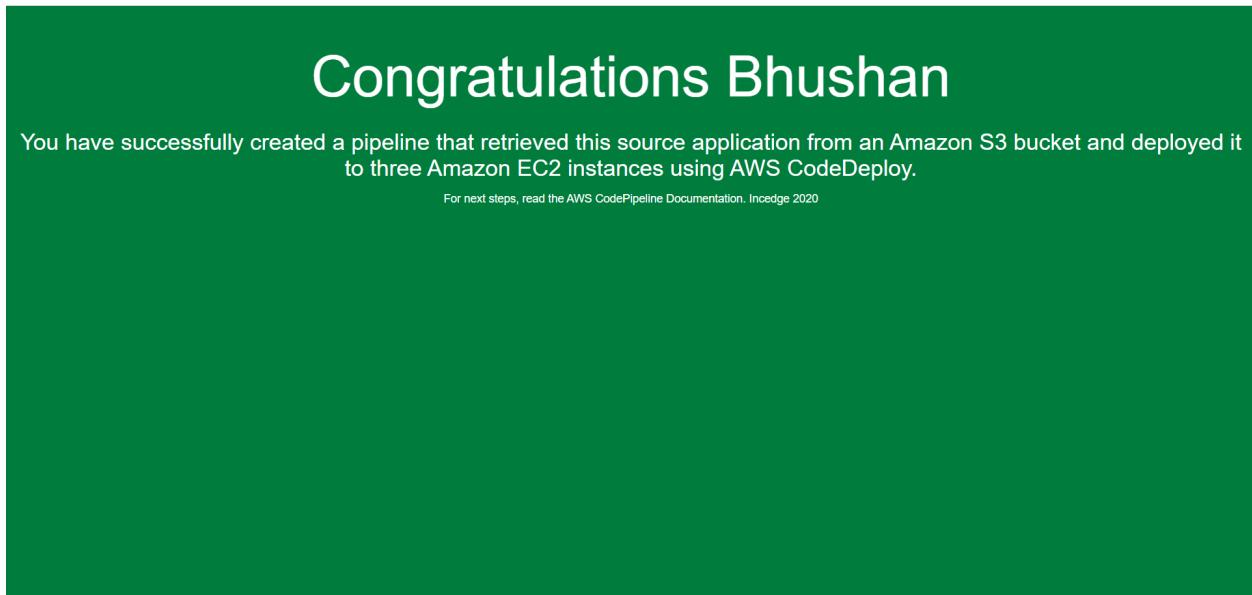
10.Click on create pipeline



11.The pipeline will be created in few minutes



12.The output will show after successful deployment.



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

Prerequisites:

Create 2 security groups by navigating to Search → Security Groups.

The screenshot shows the AWS Services search interface. The search bar at the top contains the text 'secur'. Below the search bar, the left sidebar lists various AWS services and features under categories like Instances, Images, and Elastic Block Store. The main pane displays search results for 'Security groups' under the 'Services' and 'Features' sections. Under 'Services', there is a card for 'Security Lake' with a star icon. Under 'Features', there are cards for 'Security Hub', 'Detective', and 'Amazon Inspector', each with a star icon. Both cards mention that they are 'AWS's security and compliance center' and 'Continual vulnerability management at scale' respectively.

Group 1: Master

The screenshot shows the 'Inbound rules' section of a security group configuration. The table lists ten rules:

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
All traffic	All	All	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
Custom TCP	TCP	6443	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
Custom TCP	TCP	10251	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
Custom TCP	TCP	10250	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
All TCP	TCP	0-65535	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
Custom TCP	TCP	10252	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete
SSH	TCP	22	Anywhere-... ▾	<input type="text"/> 0.0.0.0/0 Delete

At the bottom left, there is a button labeled 'Add rule'.

Group 2: Nodes

The screenshot shows the 'Inbound rules' section of the AWS CloudWatch Metrics interface. It lists seven rules:

- All traffic (Protocol: All, Port range: All, Source: Anywhere-..., Description: optional)
- SSH (Protocol: TCP, Port range: 22, Source: Custom, Description: optional)
- Custom TCP (Protocol: TCP, Port range: 10250, Source: Anywhere-..., Description: optional)
- All TCP (Protocol: TCP, Port range: 0 - 65535, Source: Anywhere-..., Description: optional)
- Custom TCP (Protocol: TCP, Port range: 30000 - 32767, Source: Anywhere-..., Description: optional)
- HTTP (Protocol: TCP, Port range: 80, Source: Anywhere-..., Description: optional)

At the bottom left is a 'Add rule' button.

Step 1: Set Up EC2 Instances.

- 1) Set up 3 EC2 instances called master, node1, node2 Select Amazon Linux as the OS image.

The screenshot shows the AWS Lambda console with the following details for a new function:

- Name:** HelloWorld
- Description:** A simple Lambda function that prints "Hello World" to the CloudWatch logs.
- Runtime:** Python 3.9
- Memory size:** 128 MB
- Timeout:** 3 seconds
- Handler:** app.lambda_handler
- Role:** Lambda execution role - *lambda-role*
- Code:** Lambda@Edge
- Deployment package:** zip file (1.0 KB)
- Test:** Hello World
- Logs:** CloudWatch Logs
- Environment variables:** None
- Tracing:** Off
- Logs:** CloudWatch Logs

IMPORTANT: The default instance type and free one provided by AWS is t2.micro, which provides only 1CPU and 1 GiB of memory. For running Kubernetes, a minimum of 2 CPUs and 2GiB of RAM is required, hence change **t2.micro** to **t2.medium**.

The screenshot shows the AWS Lambda console interface. A modal window is open for creating a new function. The 'Function name' is set to 'lambda-1', 'Runtime' is 'Node.js 18.x', and 'Handler' is 'index.handler'. The 'Code' section shows a successful upload of a ZIP file. The 'Environment' tab is selected, displaying environment variables such as 'AWS_LAMBDA_FUNCTION_NAME' and 'AWS_LAMBDA_FUNCTION_MEMORY_SIZE'. The 'Logs' tab shows a log entry for the function's first execution.

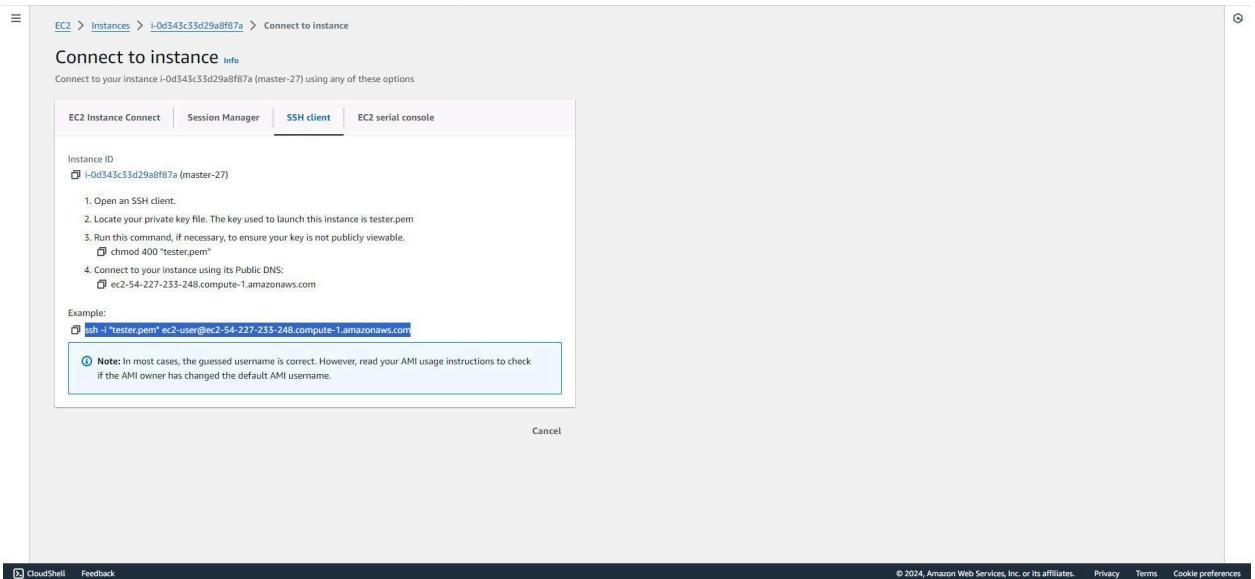
- 2) Create a key pair as you need the .pem file (private key file) on your system.
- 3) Click on 'Select existing security group' and select the master group for master instance, node group for both node instances.

The screenshot shows the AWS Lambda console interface. A modal window is open for creating a new function. The 'Function name' is set to 'lambda-1', 'Runtime' is 'Node.js 18.x', and 'Handler' is 'index.handler'. The 'Code' section shows a successful upload of a ZIP file. The 'Environment' tab is selected, displaying environment variables such as 'AWS_LAMBDA_FUNCTION_NAME' and 'AWS_LAMBDA_FUNCTION_MEMORY_SIZE'. The 'Logs' tab shows a log entry for the function's first execution.

- 4) These are the instances that are created. Click on the Instance ID of master.

Instances (1/6) Info											
Find Instance by attribute or tag (case-sensitive) Last updated less than a minute ago											
All states											
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4	Elastic IP	Actions	Launch instances
node-2-27	i-09b6410d609078e52	Terminated	t2.medium	...	View alarms +	us-east-1b	-	-	...	Actions	Launch instances
master27	i-03c6ac2c755886c08	Terminated	t2.medium	...	View alarms +	us-east-1b	-	-	...	Actions	Launch instances
node-1-27	i-00abf5ab72137c66	Terminated	t2.medium	...	View alarms +	us-east-1b	-	-	...	Actions	Launch instances
node_2_27	i-0303ad251d191e2fe	Running	t2.medium	...	View alarms +	us-east-1b	ec2-44-203-69-112.co...	44.203.69.112	...	Actions	Launch instances
master-27	i-0d343c33d29abf87a	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1b	ec2-54-227-233-248.co...	54.227.233.248	...	Actions	Launch instances
node_1_27	i-019b032c03a020826	Running	t2.medium	...	View alarms +	us-east-1b	ec2-3-89-90-121.com...	3.89.90.121	...	Actions	Launch instances

- 5) Click on Connect. This directs you to a connect dashboard. Click on SSH client, you get a SSH command. Use this command to access the instance terminal on your local system.



- 6) Go to the folder where your private key file (.pem file) is installed. Right click → Open in terminal.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4>
```

Paste the SSH command here and run it.

You might get the error of UNPROTECTED KEY FILE. This is because the .pem access is with all users of the system. Run the following commands to change the access to only the current user.

- icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /inheritance:r
- icacls "C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4\tester.pem" /grant:r "%USERNAME%:F"

Now rerun the SSH command.

7) After doing the above steps, the terminal for all 3 nodes can be seen.

```

ec2-user@ip-172-31-80-245:~ % + ~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\saira\OneDrive\Desktop\AdvDevOps\lab 3 4> ssh -i "tester.pem" ec2-user@ec2-54-227-233-248.compute-1.amazonaws.com
The authenticity of host 'ec2-54-227-233-248.compute-1.amazonaws.com (54.227.233.248)' can't be established.
ED25519 key fingerprint is SHA256:KEZfN8TkgsivFhCjxb34Xb1bs+AOHbm7YKH5AhjCC95.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-227-233-248.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

'`#_
`~ \_ #####_ Amazon Linux 2023
`~ \_ #####`#
`~ \###`#
`~ \#/ __> https://aws.amazon.com/linux/amazon-linux-2023
`~ V__>
`~ /`/
`~ /`/_`/
`~ /m/_`/
`~ /m/_`/
[ec2-user@ip-172-31-80-245 ~]$ |

ec2-user@ip-172-31-86-11:~ % + ~
2-user@ec2-3-89-90-121.compute-1.amazonaws.com
The authenticity of host 'ec2-3-89-90-121.compute-1.amazonaws.com (3.89.90.121)' can't be established.
ED25519 key fingerprint is SHA256:THchr7pke1QtAjoA0Gl91aSd+fUCu8b2D/rW3jizYOo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-89-90-121.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

'`#_
`~ \_ #####_ Amazon Linux 2023
`~ \_ #####`#
`~ \###`#
`~ \#/ __> https://aws.amazon.com/linux/amazon-linux-2023
`~ V__>
`~ /`/
`~ /`/_`/
`~ /m/_`/
[ec2-user@ip-172-31-86-11 ~]$ |

ec2-user@ip-172-31-95-90:~ % + ~
2-user@ec2-44-203-69-112.compute-1.amazonaws.com
The authenticity of host 'ec2-44-203-69-112.compute-1.amazonaws.com (44.203.69.112)' can't be established.
ED25519 key fingerprint is SHA256:+951lZ70193UkcmhX/ErjNsNLCxSubctMSyrCSe6Ek.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-44-203-69-112.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

'`#_
`~ \_ #####_ Amazon Linux 2023
`~ \_ #####`#
`~ \###`#
`~ \#/ __> https://aws.amazon.com/linux/amazon-linux-2023
`~ V__>
`~ /`/
`~ /`/_`/
`~ /m/_`/
[ec2-user@ip-172-31-95-90 ~]$ |

```

PERFORM THE FOLLOWING STEPS ON ALL 3 MACHINES

Step 2: Installation of Docker

1) Use command

'sudo su'

This allows you to act as the root user of the terminal

```

`~/m/`'
[ec2-user@ip-172-31-80-245 ~]$ sudo su
[root@ip-172-31-80-245 ec2-user]#

```

2) We can install docker using YUM(Yellowdog Updater, Modified). Use the command 'yum install docker -y'

```
[root@ip-172-31-80-245 ec2-user]# yum install docker -y
Last metadata expiration check: 0:05:22 ago on Thu Sep 26 03:53:56 2024.
Dependencies resolved.
=====
Package           Arch    Version        Repository      Size
=====
Installing:
  docker          x86_64  25.0.6-1.amzn2023.0.2  amazonlinux   44 M
Installing dependencies:
  containerd      x86_64  1.7.20-1.amzn2023.0.1  amazonlinux   35 M
  iptables-libs   x86_64  1.8.8-3.amzn2023.0.2  amazonlinux  401 k
  iptables-nft    x86_64  1.8.8-3.amzn2023.0.2  amazonlinux  183 k
  libcgroup       x86_64  3.0-1.amzn2023.0.1   amazonlinux   75 k
  libnetfilter_conntrack x86_64  1.0.8-2.amzn2023.0.2  amazonlinux   58 k
  libnfnetworklink x86_64  1.0.1-19.amzn2023.0.2  amazonlinux   30 k
  libnftnl        x86_64  1.2.2-2.amzn2023.0.2  amazonlinux  84 k
  pigz            x86_64  2.5-1.amzn2023.0.3   amazonlinux   83 k
  runc            x86_64  1.1.13-1.amzn2023.0.1  amazonlinux  3.2 M

Transaction Summary
=====
Install 10 Packages
```

```
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
  iptables-nft-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
  libnfnetworklink-1.0.1-19.amzn2023.0.2.x86_64
  libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-80-245 ec2-user]# |
```

3) Now, configure a daemon.json file by using the following chain of 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
- sudo systemctl enable docker

- sudo systemctl daemon-reload ● sudo systemctl restart docker

```
[root@ip-172-31-80-245 ec2-user]# 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
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opt": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service →
/usr/lib/systemd/system/docker.service.
[root@ip-172-31-80-245 docker]#
```

Step 3: Installing Kubernetes

- 1) For installing kubernetes, we will be using kubeadm, a framework used for creating kubernetes clusters using command line.

<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/> The following will be visible when you visit the website.

The screenshot shows the Kubernetes documentation website with the URL <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>. The page title is 'Install Kubernetes'. The top navigation bar includes links for Documentation, Kubernetes Blog, Training, Partners, Community, Case Studies, Versions, English, and a search bar. The main content area is titled 'The installation guide for your desired mirror version.' It features two tabs: 'Debian-based distributions' (selected) and 'Red Hat-based distributions'. Below these tabs, it says 'Without a package manager'. The first section, '1. Set SELinux to permissive mode:', contains instructions for Kubernetes 1.31, including a code block for setting SELinux to permissive mode:

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

A 'Caution:' box provides additional information about SELinux settings. The second section, '2. Add the Kubernetes yum repository.', is partially visible at the bottom. On the right side of the page, there are links for editing the page, creating a child page, creating a documentation issue, and printing the entire section. A sidebar on the left lists various Kubernetes documentation categories.

- 2) Select red hat-based distributions as amazon linux is based on red hat.

sudo setenforce 0

→ sets SELinux to permissive mode

`sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config` → edits the SELinux configuration file (/etc/selinux/config) to make the change persistent across reboots. If not used, SELinux reverts to enforcing mode after reboot.

Setting SELinux to permissive mode during Kubernetes installation prevents permission-related issues with container runtimes and components that may not function correctly under SELinux's enforcing policies.

Run the following commands:

- `sudo setenforce 0`
- `sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config` ● `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.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

This command is a repository script to create a kubernetes repository

```
/usr/lib/systemd/system/docker.service.
[root@ip-172-31-80-245 docker]# sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
[root@ip-172-31-80-245 docker]# 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.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[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.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-80-245 docker]#
```

- `yum repolist`

This command shows the repositories created on the machine.

```
[root@ip-172-31-80-245 docker]# yum repolist
repo id          repo name
amazonlinux      Amazon Linux 2023 repository
kernel-livepatch Amazon Linux 2023 Kernel Livepatch repository
kubernetes       Kubernetes
[root@ip-172-31-80-245 docker]#
```

Next step is to install kubelet, kubeadm, kubectl

- sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

```
[root@ip-172-31-80-245 docker]# sudo yum install -y kubelet kubeadm kubectl
--disableexcludes=kubernetes
Kubernetes                               65 kB/s | 9.4 kB   00:00
Dependencies resolved.
=====
Package           Arch    Version        Repository  Size
=====
Installing:
  kubeadm        x86_64  1.31.1-150500.1.1  kubernetes  11 M
  kubectl        x86_64  1.31.1-150500.1.1  kubernetes  11 M
  kubelet         x86_64  1.31.1-150500.1.1  kubernetes  15 M
Installing dependencies:
  conntrack-tools x86_64  1.4.6-2.amzn2023.0.2  amazonlinux 208 k
  cri-tools       x86_64  1.31.1-150500.1.1  kubernetes  6.9 M
  kubernetes-cni  x86_64  1.5.1-150500.1.1  kubernetes  7.1 M
  libnetfilter_cthelper x86_64  1.0.0-21.amzn2023.0.2  amazonlinux 24 k
  libnetfilter_cttimeout x86_64  1.0.0-19.amzn2023.0.2  amazonlinux 24 k
  libnetfilter_queue x86_64  1.0.5-2.amzn2023.0.2  amazonlinux 30 k
Transaction Summary
=====
```

```
Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64
  cri-tools-1.31.1-150500.1.1.x86_64
  kubeadm-1.31.1-150500.1.1.x86_64
  kubectl-1.31.1-150500.1.1.x86_64
  kubelet-1.31.1-150500.1.1.x86_64
  kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64
  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64
  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-80-245 docker]# |
```

Now, we need to enable the kubelet service. Run the command

- sudo systemctl enable --now kubelet

```
[root@ip-172-31-80-245 docker]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service
→ /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-80-245 docker]#
```

PERFORM THE FOLLOWING ON ONLY THE MASTER MACHINE

- 1) Firstly, we need to initialize kubernetes. For this, run the command: • `kubeadm init`

```
[root@ip-172-31-21-124 ec2-user]# kubeadm init
[init] Writing configuration to /etc/kubernetes/config.yaml...[1,1]
[init] Missing pre-flight checks
  [WARNING FileExisting-kubelet]: kubelet not found in system path
  [WARNING FileExisting-kubelet]: kubelet not found in system path
[init] Pulling images required for setting up a Kubernetes cluster
[init] This might take a few minutes...
[init] You can track the progress of this action by running 'kubectl config view --minify | grep "image pull progress%"'
[init] This will also perform this action for any other pod using these images/pods
[init] 17:39:05.142480 26634 checks.go:848| detected that the runtime image "registry.k8s.io/pause:3.0" of the container runtime is inconsistent with that used by kubelets. It is recommended to use "registry.k8s.io/pause:3.0" as the CNI sandbox image.
[init] Using certificate folder "/etc/kubernetes/pki"
[init] Generating "ca.crt" certificate and key
[init] Generating "apiserver" certificate and key
[init] apiserver serving cert is signed for DNS names [ip-172-31-21-124.ec2.internal kubernetes.default.svc kubernetes.default.svc.cluster.local] and IP address [10.40.0.1 172.31.124]
[init] Generating "apiserver-kubelet-client" certificate and key
[init] Generating "apiserver-csr" certificate and key
[init] Generating "apiserver-ingress-client" certificate and key
[init] Generating "etcd-peer" certificate and key
[init] Generating "etcdserver" certificate and key
[init] etcdserver serving cert is signed for DNS names [ip-172-31-21-124.ec2.internal localhost] and IP address [172.31.21.124 127.0.0.1 ::1]
[init] Generating "etcdhealthcheck-client" certificate and key
[init] Generating "apiserver-etcd-client" certificate and key
[init] Generating "ca.key" key and public key
[init] Generating "apiserver" certificate and key for Kubernetes
[init] Generating "admin.conf" bootstrap config file
[init] Creating a ConfigMap "kubelet-config" in namespace kube-system with the configuration for the kubelets in the cluster
[upload-certs] Skipping phase. Please run --upload-certs
[mark-control-plane] Marking the node ip-172-31-21-124.ec2.internal as control-plane by adding the labels: node-role.kubernetes.io/control-plane node.kubernetes.io/exclude-from-autoscaling=true
[mark-control-plane] Marking the node ip-172-31-21-124.ec2.internal as control-plane by adding the taints: (node-role.kubernetes.io/control-plane:NoSchedule)
[bootstrap-token] Using token: 761glg-qpkqswt9sgpt
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC roles to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC roles to allow Node Bootstrap tokens to post CMs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC roles to allow Node Bootstrap tokens to automatically approve CMs from a Node Bootstrap token
[bootstrap-token] Configured RBAC roles to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[init] Finalize) Updating "/etc/kubernetes/admin.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

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

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

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

  export KUBECONFIG=/etc/kubernetes/admin.conf

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

Then you can join any number of worker nodes by running the following on each as root:
[root@ip-172-31-21-124 ec2-user]# kubeadm join 172.31.80.245:6443 --token g7dyjj.r62w19oc05n7k4kc \
--discovery-token-ca-cert-hash sha256:a012677cdc93d202cf09e2370eb7231d212211bac6f916cd9cb547b7f00a4f2
[root@ip-172-31-21-124 ec2-user]#
```

From the output, we will receive a command that is used to link the nodes to the master. Copy it and save it somewhere local.

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

kubeadm join 172.31.80.245:6443 --token g7dyjj.r62w19oc05n7k4kc \
--discovery-token-ca-cert-hash sha256:a012677cdc93d202cf09e2370eb7231d212211bac6f916cd9cb547b7f00a4f2
```

- 2) From the output, we receive the following commands:

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

```
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
```

- 3) To check whether nodes are connected, run the command

- kubectl get nodes

This output shows only master is connected right now.

```
sudo chown $(id -u):$(id -g) $HOME/.kube/config
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS    ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   NotReady  control-plane  112s   v1.31.1
[root@ip-172-31-80-245 docker]#
```

PERFORM THE FOLLOWING ONLY ON THE NODE MACHINES

Use the command that you had copied before and use them on the node machines.

The image shows two terminal windows side-by-side. Both windows are running on the root user of their respective node machines. The top window is for node IP-172-31-86-11 and the bottom window is for node IP-172-31-95-90. Both windows display the same sequence of commands related to kubelet configuration and joining a cluster. The logs show the kubelet writing its configuration file, setting environment flags, starting, and then checking for health. It also indicates that the node has joined the cluster by sending a certificate signing request to the apiserver and becoming healthy after 501.57413ms. Finally, it prompts to run 'kubectl get nodes' to see the node join the cluster.

```
root@ip-172-31-86-11:/etc/docker/    +  ×
em get cm kubeadm-config -o yaml'
[Kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[Kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[Kubelet-start] Starting the kubelet
[Kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[Kubelet-check] The kubelet is healthy after 501.57413ms
[Kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

[root@ip-172-31-86-11 docker]#
```



```
root@ip-172-31-95-90:/etc/docker/    +  ×
em get cm kubeadm-config -o yaml'
[Kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[Kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[Kubelet-start] Starting the kubelet
[Kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[Kubelet-check] The kubelet is healthy after 1.001297846s
[Kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

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

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

[root@ip-172-31-95-90 docker]#
```

NOW GO BACK TO THE MASTER MACHINE AND RERUN ‘kubectl get nodes’

```
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   NotReady   control-plane   2m18s   v1.31.1
ip-172-31-86-11.ec2.internal   NotReady   <none>        9s     v1.31.1
ip-172-31-95-90.ec2.internal  NotReady   <none>        2s     v1.31.1
[root@ip-172-31-80-245 docker]# |
```

As we see, the status of the nodes are <NOT READY>. To change it, we need to install a network CNI plugin.

Use the following command only on the master machine::

kubectl apply -f

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

Now run ‘kubectl get nodes’ again.

```
[root@ip-172-31-80-245 docker]# kubectl get nodes
NAME           STATUS   ROLES      AGE   VERSION
ip-172-31-80-245.ec2.internal   Ready    control-plane   9m33s   v1.31.1
ip-172-31-86-11.ec2.internal   Ready    <none>        7m24s   v1.31.1
ip-172-31-95-90.ec2.internal  Ready    <none>        7m17s   v1.31.1
[root@ip-172-31-80-245 docker]# |
```

Conclusion:

In this experiment, we have learned how to create kubernetes clusters on a linux terminal, how the ssh command works on a local terminal and what requirements are necessary to create kubernetes clusters. We have used many command inline tools of Kubernetes like kubecmd, kubectl to set up the clusters and work with docker container to perform the connection of the nodes with master machine.

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

Theory

kubectl is the command-line tool used to interact with Kubernetes clusters. It serves as the primary interface for managing and orchestrating containers in a Kubernetes environment. By sending commands to the Kubernetes API server, kubectl allows you to control clusters, manage workloads, and inspect resource states.

To begin using Kubernetes, installing kubectl is essential. The installation process varies based on the operating system (Linux, Windows, or macOS). After installing, kubectl connects to the Kubernetes cluster using the kubeconfig file, which stores details like cluster name, server address, and access credentials. With this connection established, you can use kubectl to perform a variety of operations, such as creating, updating, scaling, and deleting applications.

Step 1: Create an EC2 instance use ubuntu application and select t2.medium category in instance type create a new key rsa type save it in local machine in an folder:

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

exp4

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

Key pair type

RSA
RSA encrypted private and public key pair

ED25519
ED25519 encrypted private and public key pair

Private key file format

.pem
For use with OpenSSH

.ppk
For use with PuTTY

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

Cancel **Create key pair**

Step 2: Click create to create the instance:

Instances (4) Info		Last updated		Connect	Instance state ▾	Actions ▾	Launch instances ▾	
		Find Instance by attribute or tag (case-sensitive)	All states ▾	< 1 >				
□	Name ↴	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	Availability Zone ▾	Public IP
□	Exp4	i-09a06120376188a8d	Running	t2.medium	Initializing	View alarms	us-east-1a	ec2-52-;

Step 3:

Navigate to ssh client copy the key:

The screenshot shows the AWS EC2 Instances Launch an instance wizard. The first step, 'Name and tags', has a 'Name' field set to 'bhushan'. The second step, 'Application and OS Images (Amazon Machine Image)', shows a search bar and a 'Quick Start' tab selected. Under 'Quick Start', there are icons for Amazon Linux, macOS, Ubuntu, Windows, Red Hat, and SUSE Linux. A 'Browse more AMIs' link is also present. The third step, 'Summary', shows the configuration: 1 instance, Canonical Ubuntu 24.04 AMI, t2.medium instance type, New security group, and 1 volume(s) - 8 GiB storage. It also displays a 'Free tier' message: 'In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance'. The 'Launch instance' button is at the bottom.

EC2 > Instances > Launch an instance

Launch an instance Info

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

Name and tags Info

Name
bhushan Add additional tags

Application and OS Images (Amazon Machine Image) Info

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

Recents Quick Start

Summary

Number of instances Info
1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type)
t2.medium

Firewall (security group)
New security group

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance

Cancel **Launch instance** Review commands

Summary

Number of instances Info
1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0e86e20dae9224db8

Virtual server type (instance type)
t2.medium

Firewall (security group)
New security group

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance

Cancel **Launch instance** Review commands

Screenshot of the AWS CloudFormation console showing the creation of a new instance.

Summary

- Number of instances: 1
- Software Image (AMI): Canonical, Ubuntu, 24.04, amd64... (ami-0e86e20dae9224db8)
- Virtual server type (instance type): t2.medium
- Firewall (security group): New security group
- Storage (volumes): 1 volume(s) - 8 GiB

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required: exp4

Network settings

Network: vpc-0ce401b5e9a4207c6
Subnet: Info
No preference (Default subnet in any availability zone)
Auto-assign public IP: Info
Enable

Launch instance

Additional charges apply when outside of free tier allowance.

Screenshot of the AWS CloudFormation console showing the connection options for the created instance.

Connect to instance

Connect to your instance i-0e8ff754d88c114cb (bhushan) using any of these options

EC2 Instance Connect | Session Manager | **SSH client** | EC2 serial console

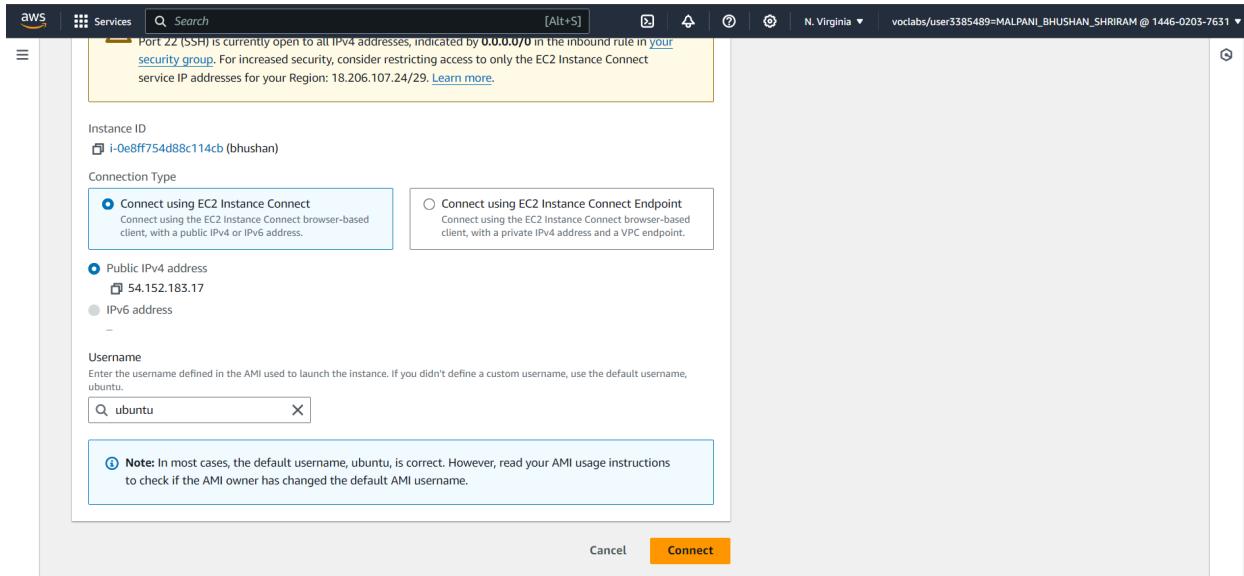
Instance ID: i-0e8ff754d88c114cb (bhushan)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is exp4.pem
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "exp4.pem"
- Connect to your instance using its Public DNS:
ec2-54-152-183-17.compute-1.amazonaws.com

Example:

ssh -i "exp4.pem" ubuntu@ec2-54-152-183-17.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.



Step 4:navigate to the folder open the terminal and paste the ssh command:

```
ssh -i "sahilexp4key.pem" ubuntu@ec2-52-201-236-39.compute-1.amazonaws.com
```

```
PS C:\Users\HP\Desktop\Exp4> ssh -i "exp4.pem" ubuntu@ec2-54-152-183-17.compute-1.amazonaws.com
The authenticity of host 'ec2-54-152-183-17.compute-1.amazonaws.com (64:ff9b::3698:b711)' can't be established.
ED25519 key fingerprint is SHA256:sk0m7P+Ism4YHKKxY7e6EX0S6KsyVWlghbvcGiH1v+s.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-152-183-17.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

System information as of Thu Sep 26 16:31:30 UTC 2024

 System load:  0.03      Processes:           122
 Usage of /:   22.9% of 6.71GB   Users logged in:     1
 Memory usage: 6%          IPv4 address for enX0: 172.31.85.133
 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
```

Step 5:Install docker

Use the commands given below to install docker

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

```
ubuntu@ip-172-31-85-133:~$ sudo mkdir -p /etc/apt/trusted.gpg.d/
ubuntu@ip-172-31-85-133:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/doc
ker.asc
```

```
ubuntu@ip-172-31-85-133:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
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 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [15.3 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [378 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [82.0 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [4548 B]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [271 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [115 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:15 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [10.2 kB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [353 kB]
```

Use:

sudo apt-get update

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
```

Use:

sudo apt-get install -y docker-ce

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get install -y docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0
  pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7
  libslirp0 pigz slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 142 not upgraded.
Need to get 123 MB of archives.
After this operation, 442 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.17.1-1~ubuntu.24.04~noble
[30.3 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:27.3.1-1~ubuntu.24.04~noble [15.
0 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:27.3.1-1~ubuntu.24.04~noble [25.6 MB]
```

Now the docker is installed;

Now lets enable the docker:

```
sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json {
```

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

```
ubuntu@ip-172-31-85-133:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
```

sudo systemctl enable docker

```
ubuntu@ip-172-31-85-133:~$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

sudo systemctl daemon-reload

```
ubuntu@ip-172-31-85-133:~$ sudo systemctl daemon-reload
sudo systemctl restart docker
```

sudo systemctl restart docker

```
ubuntu@ip-172-31-85-133:~$ sudo mkdir -p /etc/apt/keyrings
ubuntu@ip-172-31-85-133:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

Step 6: Now lets install kubernetes;

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

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

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

```
ubuntu@ip-172-31-85-133:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /'
```

sudo apt-get update

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:3 https://download.docker.com/linux/ubuntu noble InRelease
Hit:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb Packages [4865 B]
Fetched 6051 B in 0s (12.3 kB/s)
Reading package lists... Done
```

sudo apt-get install -y kubelet kubeadm kubectl

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 142 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubeadm 1.31.1-1.1 [11.4 M]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb kubectl 1.31.1-1.1 [11.2 M]
```

```
sudo apt-mark hold kubelet kubeadm kubectl
```

```
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.  
kubelet set on hold.  
kubeadm set on hold.  
kubectl set on hold.
```

```
sudo systemctl enable --now kubelet
```

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

```
ubuntu@ip-172-31-85-133:~$ sudo systemctl enable --now kubelet  
sudo kubeadm init --pod-network-cidr=10.244.0.0/16  
[init] Using Kubernetes version: v1.31.0  
[preflight] Running pre-flight checks  
W0926 16:54:00.720484 4952 checks.go:1080] [preflight] WARNING: Couldn't create the interface used for talking to the  
container runtime: failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for  
endpoint "unix:///var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime  
.v1.RuntimeService  
[WARNING FileExisting-socat]: socat not found in system path  
[preflight] Pulling images required for setting up a Kubernetes cluster  
[preflight] This might take a minute or two, depending on the speed of your internet connection  
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'  
error execution phase preflight: [preflight] Some fatal errors occurred:  
failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix:///  
var/run/containerd/containerd.sock": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService[preflight]  
If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=...'  
To see the stack trace of this error execute with --v=5 or higher  
ubuntu@ip-172-31-85-133:~$ |
```

```
sudo apt-get install -y containerd
```

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get install -y containerd  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following packages were automatically installed and are no longer required:  
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz  
  slirp4netns  
Use 'sudo apt autoremove' to remove them.  
The following additional packages will be installed:  
  runc  
The following packages will be REMOVED:  
  containerd.io docker-ce  
The following NEW packages will be installed:  
  containerd runc  
0 upgraded, 2 newly installed, 2 to remove and 142 not upgraded.  
Need to get 47.2 MB of archives.  
After this operation, 53.1 MB disk space will be freed.  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]  
Fetched 47.2 MB in 1s (72.0 MB/s)  
(Reading database ... 68064 files and directories currently installed.)  
Removing docker-ce (5:27.3.1-1~ubuntu.24.04~noble) ...  
Removing containerd.io (1.7.22-1) ...  
Selecting previously unselected package runc.  
(Reading database ... 68044 files and directories currently installed.)  
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...  
Unpacking runc (1.1.12-0ubuntu3.1) ...  
Selecting previously unselected package containerd.  
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...  
Unpacking containerd (1.7.12-0ubuntu4.1) ...  
Setting up runc (1.1.12-0ubuntu3.1) ...  
Setting up containerd (1.7.12-0ubuntu4.1) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
No services need to be restarted.
```

```

[plugins."io.containerd.transfer.v1.local"]
config_path = ""
max_concurrent_downloads = 3
max_concurrent_uploaded_layers = 3

[[plugins."io.containerd.transfer.v1.local".unpack_config]]
differ = ""
platform = "linux/amd64"
snapshooter = "overlayfs"

[proxy_plugins]

[stream_processors]

[stream_processors."io.containerd.ocicrypt.decoder.v1.tar"]
accepts = ["application/vnd.oci.image.layer.v1.tar+encrypted"]
args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
path = "ctd-decoder"
returns = "application/vnd.oci.image.layer.v1.tar"

[stream_processors."io.containerd.ocicrypt.decoder.v1.tar.gzip"]
accepts = ["application/vnd.oci.image.layer.v1.tar+gzip+encrypted"]
args = ["--decryption-keys-path", "/etc/containerd/ocicrypt/keys"]
env = ["OCICRYPT_KEYPROVIDER_CONFIG=/etc/containerd/ocicrypt/ocicrypt_keyprovider.conf"]
path = "ctd-decoder"
returns = "application/vnd.oci.image.layer.v1.tar+gzip"

[timeouts]
"io.containerd.timeout.bolt.open" = "0s"
"io.containerd.timeout.metrics.shimstats" = "2s"
"io.containerd.timeout.shim.cleanup" = "5s"
"io.containerd.timeout.shim.load" = "5s"
"io.containerd.timeout.shim.shutdown" = "3s"
"io.containerd.timeout.task.state" = "2s"

[ttrpc]
address = ""
gid = 0
uid = 0
ubuntu@ip-172-31-85-133:~$ |

```

sudo mkdir -p /etc/containerd

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

```

ubuntu@ip-172-31-85-133:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
path = ""

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

[grpc]
address = "/run/containerd/containerd.sock"
gid = 0
max_recv_message_size = 16777216
max_send_message_size = 16777216
tcp_address = ""
tcp_tls_ca = ""
tcp_tls_cert = "

```

```
sudo systemctl restart containerd  
sudo systemctl enable containerd  
sudo systemctl status containerd
```

```
ubuntu@ip-172-31-85-133:~$ sudo systemctl restart containerd  
sudo systemctl enable containerd  
sudo systemctl status containerd  
● containerd.service - containerd container runtime  
   Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; preset: enabled)  
     Active: active (running) since Thu 2024-09-26 16:55:51 UTC; 216ms ago  
       Docs: https://containerd.io  
     Main PID: 5364 (containerd)  
        Tasks: 7  
         Memory: 14.0M (peak: 14.3M)  
        CPU: 57ms  
      CGroup: /system.slice/containerd.service  
              └─5364 /usr/bin/containerd  
  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784241167Z" level=info msg="serving..." address=/run/containerd/containerd.sock  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784280374Z" level=info msg="serving..." address=/run/containerd/containerd.sock  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784319870Z" level=info msg="Start subscribing containerd event"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784456858Z" level=info msg="Start recovering state"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784507875Z" level=info msg="Start events monitor"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784516597Z" level=info msg="Start snapshots syncer"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784528411Z" level=info msg="Start cnf syncer for default"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784534976Z" level=info msg="Start streaming server"  
Sep 26 16:55:51 ip-172-31-85-133 containerd[5364]: time="2024-09-26T16:55:51.784579116Z" level=info msg="containerd successfully booted in 0.025101s"  
Sep 26 16:55:51 ip-172-31-85-133 systemd[1]: Started containerd.service - containerd container runtime.
```

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

```
ubuntu@ip-172-31-85-133:~$ sudo apt-get install -y socat  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following packages were automatically installed and are no longer required:  
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns  
Use 'sudo apt autoremove' to remove them.  
The following NEW packages will be installed:  
  socat  
0 upgraded, 1 newly installed, 0 to remove and 142 not upgraded.  
Need to get 374 kB of archives.  
After this operation, 1649 kB of additional disk space will be used.  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]  
Fetched 374 kB in 0s (11.0 MB/s)  
Selecting previously unselected package socat.  
(Reading database ... 68108 files and directories currently installed.)  
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...  
Unpacking socat (1.8.0.0-4build3) ...  
Setting up socat (1.8.0.0-4build3) ...  
Processing triggers for man-db (2.12.0-4build2) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Step 7: Initialize the kubernetes:

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

```

ubuntu@ip-172-31-85-133:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[kinit] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0926 16:57:08.588594    5580 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that
used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-85-133 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster
.local] and IPs [10.96.0.1 172.31.85.133]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-85-133 localhost] and IPs [172.31.85.133 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-85-133 localhost] and IPs [172.31.85.133 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using KubeConfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests"

```

```

ubuntu@ip-172-31-85-133:~$ mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-85-133:~$ |

```

kubectl apply -f

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

```

ubuntu@ip-172-31-85-133:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created

```

Step 8: Now we can deploy our nginx server on this cluster using following steps:

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

```

ubuntu@ip-172-31-85-133:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created

```

kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
nginx-deployment-d556bf558-bq75m	0/1	Pending	0	16s
nginx-deployment-d556bf558-t7mgm	0/1	Pending	0	16s

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

kubectl port-forward \$POD_NAME 8080:80

```

ubuntu@ip-172-31-85-133:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
ubuntu@ip-172-31-85-133:~$ 

```

kubectl taint nodes --all node-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted

```
kubectl get nodes
```

```
ubuntu@ip-172-31-85-133:~$ kubectl taint nodes ip-172-31-85-133 node-role.kubernetes.io/control-plane:NoSchedule-
node/ip-172-31-85-133 untainted
ubuntu@ip-172-31-85-133:~$ kubectl get nodes
kubectl get pods
NAME           STATUS   ROLES      AGE     VERSION
ip-172-31-85-133   Ready    control-plane   6m53s   v1.31.1
NAME              READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-bq75m   1/1    Running   0          4m26s
nginx-deployment-d556bf558-t7mgm   1/1    Running   0          4m26s
```

```
kubectl get pods
```

```
ubuntu@ip-172-31-85-133:~$ kubectl port-forward $POD_NAME 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
Handling connection for 8081
```

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

```
^Cubuntu@ip-172-31-85-133:~$ kubectl port-forward pod/nginx-deployment-d556bf558-t7mgm 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
Handling connection for 8081
```

Step 9 :check deployment:

Open new terminal in folder,

Paste the ssh key,

Type

```
curl --head http://127.0.0.8081
```

```
ubuntu@ip-172-31-85-133:~$ curl --head http://127.0.0.1:8081
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Thu, 26 Sep 2024 17:24:21 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes
```

Status Code 200 tells us that we have successfully deployed our nginx server on ec2 instance

Now we have successfully deployed our nginx server on our ec2 instance.

Conclusion

In this experiment, we installed kubectl, the command-line tool for managing Kubernetes clusters, using the appropriate package manager for our system. After setting up the kubeconfig file to connect with the Kubernetes cluster, we verified the connection and the cluster status using kubectl get nodes. We then deployed our first Kubernetes application by writing the necessary configuration in files, ensuring the application was defined correctly. Following the deployment, we inspected the resources and application status with kubectl get pods to confirm the application was running successfully in the cluster. This experiment demonstrated the basic interaction with Kubernetes using kubectl for managing resources and monitoring deployments.

EXPERIMENT NO. 05

Aim: To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine.

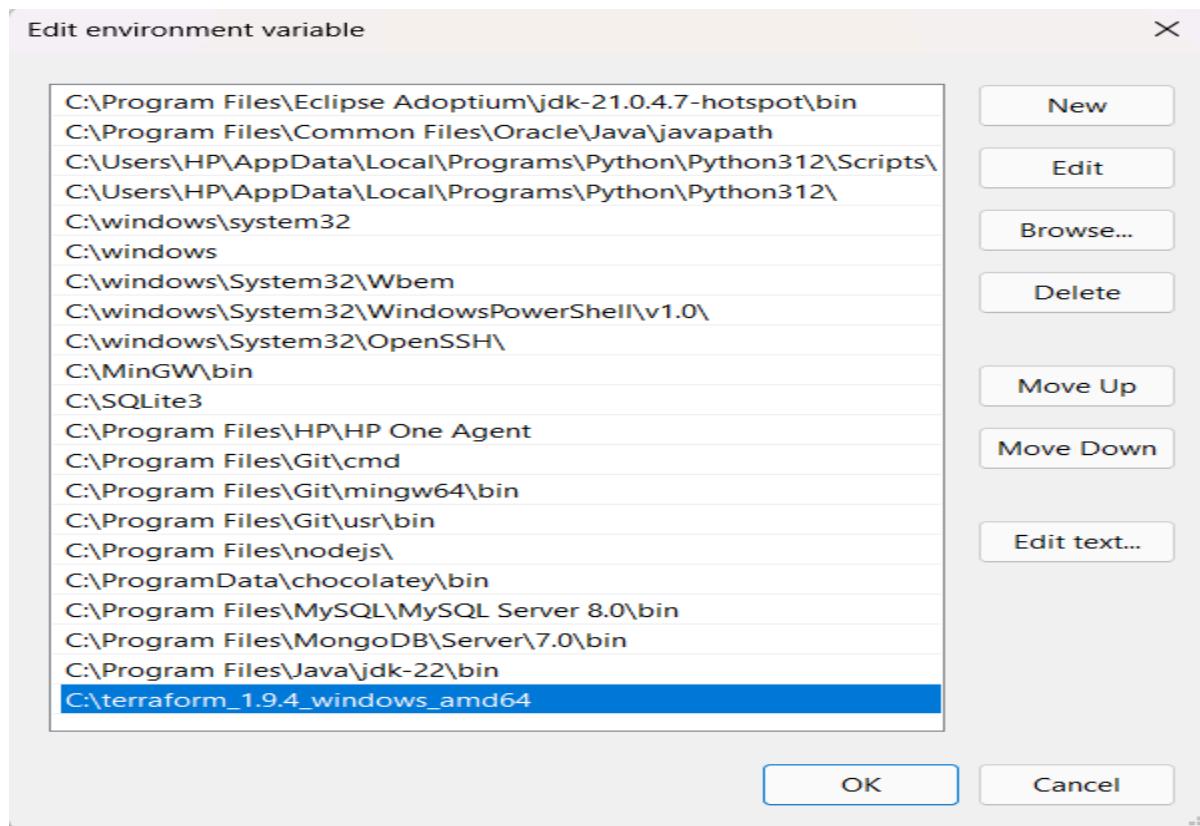
1) Install terraform in the system. You can choose the Operating System according to your requirement.

The screenshot shows the Terraform website's "Install Terraform" page. On the left sidebar, under "Operating Systems", "Windows" is selected. The main content area shows "Binary download" options for Windows and Linux. For Windows, there are links for "386" (Version: 1.9.4) and "AMD64" (Version: 1.9.4). For Linux, there are links for "Ubuntu/Debian", "CentOS/RHEL", "Fedora", "Amazon Linux", and "Homebrew".

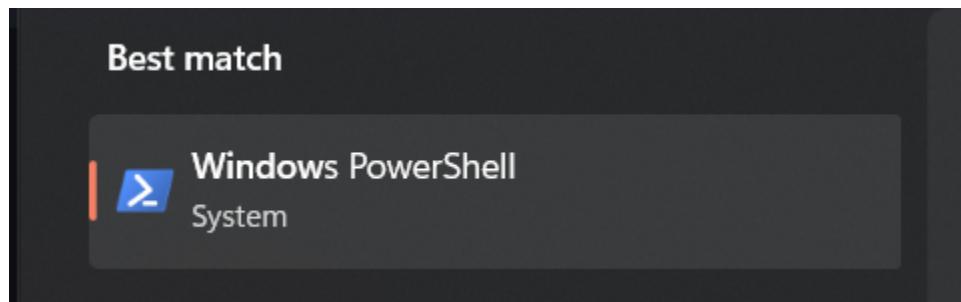
2) See the downloads file the zipped file will be there in the folder.

Today			
	terraform_1.9.4_windows_386	12-08-2024 14:22	zip
	terraform_1.9.4_windows_amd64	12-08-2024 14:21	zip

3) Extract this file in the C drive, copy the path of the file in the environment variable from settings



- 4) Open the windows PowerShell and run the command terraform



5)It will show all the properties of terraform:

```
PS C:\Users\HP> 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
  state      Advanced state management
  taint      Mark a resource instance as not fully functional
  test       Execute integration tests for Terraform modules
  untaint   Remove the 'tainted' state from a resource instance
  version   Show the current Terraform version
  workspace Workspace management

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

A. Creating docker image using terraform

Step 1: Check docker is working well using command docker.

```
PS C:\Users\HP> 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

Management Commands:
  builder  Manage builds
  buildx*  Docker Buildx
  checkpoint  Manage checkpoints
  compose*  Docker Compose
  container  Manage containers
  context   Manage contexts
  debug*   Get a shell into any image or container
  desktop*  Docker Desktop commands (Alpha)
  dev*     Docker Dev Environments
  extension* Manages Docker extensions
  feedback* Provide feedback, right in your terminal!
  image    Manage images
  init*   Creates Docker-related starter files for your project
  manifest  Manage Docker image manifests and manifest lists
  network  Manage networks
```

```
PS C:\Users\HP> docker --version
Docker version 27.0.3, build 7d4bcd8
```

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

Script:
terraform

```
1  terraform{  
2      required_providers{  
3          docker = {  
4              source = "kreuzwerker/docker"  
5              version = "2.21.0"  
6          }  
7      }  
8  }  
9  provider "docker" {  
10     host = "npipe:///./pipe//docker_engine"  
11 }  
12 # Pulls the image  
13 resource "docker_image" "ubuntu"{  
14     name = "ubuntu:latest"  
15 }  
16 # Create a container  
17 resource "docker_container" "foo"{  
18     image = docker_image.ubuntu.image_id  
19     name = "foo"  
20 }
```

Step 3: Execute Terraform Init command to initialize the resources

```
PS C:\Users\HP\Desktop\Terraform\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) Execute Terraform plan to see the available resources

```
PS C:\Users\HP\Desktop\Terraform\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          = (known after apply)
  + 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
  + stdio_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.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
PS C:\Users\HP\Desktop\Terraform\Docker> |
```

This is docker images before apply:

PS C:\Users\HP\Desktop\Terraform\Docker> docker images				
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE

Step5) Execute Terraform apply to apply the configuration, which will automatically create and run the Ubuntu Linux container based on our configuration. Using command : “**terraform apply**”

```
PS C:\Users\HP\Desktop\Terraform\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          = (known after apply)
    + container_logs   = (known after apply)
    + entrypoint       = (known after apply)
    + env              = (known after apply)
    + exit_code        = (known after apply)
    + gateway          = (known after apply)
    + hostname         = (known after apply)
    + id               = (known after apply)
    + image             = (known after apply)
    + init              = (known after apply)
    + ip_address        = (known after apply)
    + ip_prefix_length = (known after apply)
    + ipc_mode          = (known after apply)
    + log_driver        = (known after apply)
    + logs              = false
    + must_run          = true
    + name              = "foo"
    + network_data      = (known after apply)
    + read_only          = false
    + remove_volumes    = true
    + restart            = "no"
    + rm                = false
    + runtime            = (known after apply)
    + security_opts     = (known after apply)
    + shm_size           = (known after apply)
    + start              = true
    + stdin_open         = false
    + stop_signal        = (known after apply)
    + stop_timeout       = (known after apply)
    + tty                = false

    + healthcheck (known after apply)
    + labels (known after apply)
}

Plan: 1 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_container.foo: Creating...
docker_container.foo: Creation complete after 0s [id=db3fd3cf50f3c1565b4b8d26c7d876a6cdfc8228abf9af04039673e59cb0b0f7]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Docker images, After Executing Apply step:

```
PS C:\Users\HP\Desktop\Terraform\Docker> docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
ubuntu          latest   edbfe74c41f8  2 weeks ago  78.1MB
PS C:\Users\HP\Desktop\Terraform\Docker>
```

Step 6: Execute Terraform destroy to delete the configuration, which will automatically delete the Ubuntu Container.

```
PS C:\Users\HP\Desktop\Terraform\Docker> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=db3fd3cf50f3c1565b4b8d26c7d876a6cdfc8228abf9af04039673e59cb0b0f7]

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",
        - "infinity",
    ] -> 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         = "db3fd3cf50f3c1565b4b8d26c7d876a6cdfc8228abf9af04039673e59cb0b0f7" -> null
    - id               = "db3fd3cf50f3c1565b4b8d26c7d876a6cdfc8228abf9af04039673e59cb0b0f7" -> null
    - image             = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
    - init              = false -> null
    - ip_address        = "172.17.0.2" -> null
    - ip_prefix_length = 16 -> null
    - ipc_mode          = "private" -> null
    - links             = [] -> null
    - log_driver         = "json-file" -> null
    - log_opts           = {} -> null
    - logs              = false -> null
    - max_retry_count   = 0 -> null
    - memory             = 0 -> null
    - memory_swap        = 0 -> null
    - must_run           = true -> null
    - name               = "foo" -> null
    - network_data       = []
}
```

```
- read_only      = false -> null
- remove_volumes = true -> null
- restart        = "no" -> null
- rm              = false -> null
- runtime         = "runc" -> null
- security_opts  = [] -> null
- shm_size        = 64 -> null
- start           = true -> null
- stdio_open       = false -> null
- stop_timeout     = 0 -> null
- storage_opts    = {} -> null
- sysctls          = {} -> null
- tmpfs            = {} -> null
- tty              = false -> null
# (8 unchanged attributes hidden)
}

# docker_image.ubuntu will be destroyed
- resource "docker_image" "ubuntu" {
    - id      = "sha256: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.

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=db3fd3cf50f3c1565b4b8d26c7d876a6cdfc8228abf9af04039673e59cb0b0f7]
docker_container.foo: Destruction complete after 0s
docker_image ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image ubuntu: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
PS C:\Users\HP\Desktop\Terraform\Docker>
```

This is docker images after destroying:

```
PS C:\Users\HP\Desktop\Terraform\Docker> docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
PS C:\Users\HP\Desktop\Terraform\Docker> |
```

Adv DevOps Practical 7

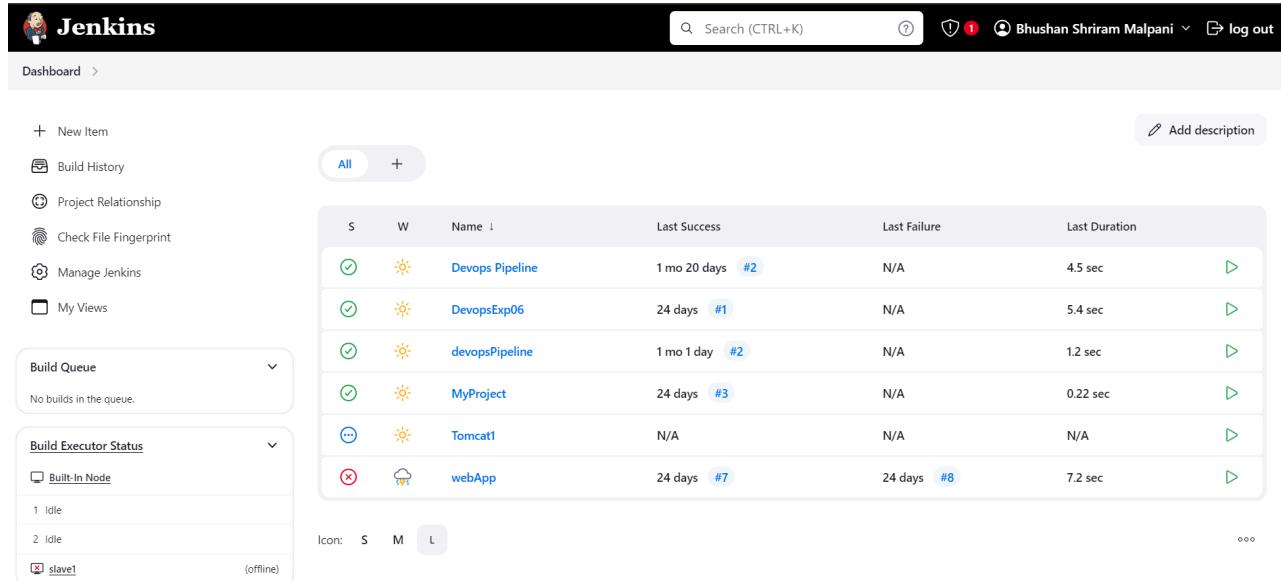
Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

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

Steps to integrate Jenkins with SonarQube

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



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

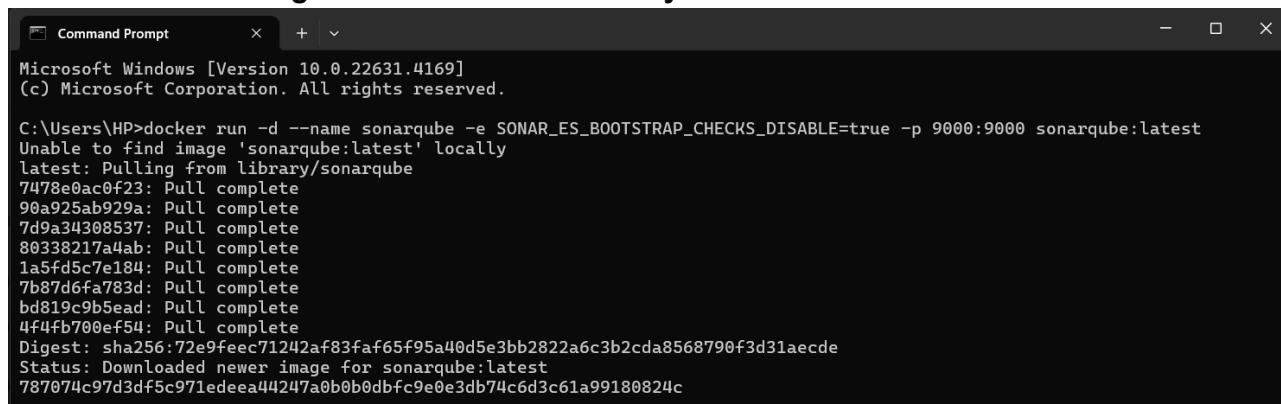
- Header:** Jenkins logo, search bar, user info (Bhushan Shriram Malpani), and log out button.
- Left Sidebar:**
 - Dashboard >
 - +
 - Build History
 - Project Relationship
 - Check File Fingerprint
 - Manage Jenkins
 - My Views
 - Build Queue (No builds in the queue)
 - Build Executor Status (1 Idle, 2 Idle, 1 slave1 offline)
- Central Content:**
 - Build History section with "All" selected.
 - Table of recent builds:

S	W	Name	Last Success	Last Failure	Last Duration
Green	Sun	Devops Pipeline	1 mo 20 days #2	N/A	4.5 sec
Green	Sun	DevopsExp06	24 days #1	N/A	5.4 sec
Green	Sun	devopsPipeline	1 mo 1 day #2	N/A	1.2 sec
Green	Sun	MyProject	24 days #3	N/A	0.22 sec
Idle	Sun	Tomcat1	N/A	N/A	N/A
Red	Cloud	webApp	24 days #7	24 days #8	7.2 sec
 - Icon selection buttons: S, M, L.

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

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

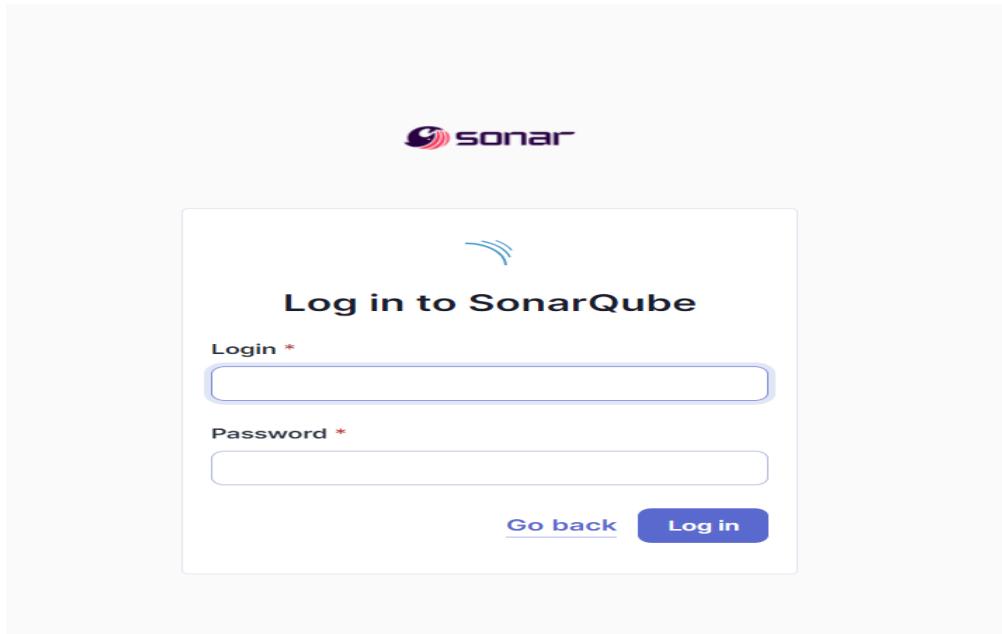
- **Warning: run below command only once**



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

C:\Users\HP>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
787074c97d3df5c971edeeaa44247a0b0b0dbfc9e0e3db74c6d3c61a99180824c
```

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

Setup the project and come back to Jenkins Dashboard.
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. A search bar at the top contains the text 'sonar'. Below the search bar, there are tabs for 'Available plugins' (which is selected), 'Installed plugins', and 'Advanced settings'. The results list shows three items:

- SonarQube Scanner 2.17.2**: This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality. It is marked as 'Released' and was last updated 7 months and 6 days ago. An 'Install' button is visible.
- Sonar Quality Gates 3.15.v1f12b_e61a_3a_4**: Library plugins (for use by other plugins) analysis Other Post-Build Actions. Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed"). It was last updated 27 days ago.
- Quality Gates 2.5**: Fails the build whenever the Quality Gates criteria in the Sonar analysis aren't met (the project Quality Gates status is different than "Passed")

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page with the 'Download progress' tab selected. On the left, there is a sidebar with links for 'Updates', 'Available plugins' (selected), 'Installed plugins', and 'Advanced settings'. The main area shows the download progress for the 'SonarQube Scanner' plugin:

Preparation	Success
• Checking internet connectivity	
• Checking update center connectivity	
• Success	

Below this, it says 'SonarQube Scanner' and 'Loading plugin extensions' with a green checkmark next to 'Success'. At the bottom, there are two buttons: 'Go back to the top page' and 'Restart Jenkins when installation is complete and no jobs are running'.

6. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for **SonarQube Servers** and enter the details.

Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube> for me
adv_devops_7_sonarqube

In **Server URL** Default is **http://localhost:9000**

The screenshot shows the Jenkins 'Manage Jenkins > System' page. The 'SonarQube installations' section is highlighted. It includes fields for 'Name' (containing 'advdevops_7'), 'Server URL' (containing 'https://localhost:9000'), and 'Server authentication token' (containing a dropdown menu with '- none -'). There is also an 'Advanced' dropdown at the bottom.

7. Search for SonarQube Scanner under Global Tool Configuration.

Choose the latest configuration and choose Install automatically.

Dashboard > Manage Jenkins > Tools

The screenshot shows the Jenkins Global Tool Configuration page for SonarQube installations. It includes fields for Name (advdevops_7), Server URL (https://localhost:9000), and Server authentication token (none). There is also an 'Advanced' dropdown and a 'Save' button at the bottom.

Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.

The screenshot shows the Jenkins SonarQube Scanner installations configuration page. It adds a new scanner named 'exp07' and checks the 'Install automatically' option. It also specifies the version 'SonarQube Scanner 6.2.0.4584'. There are 'Save' and 'Apply' buttons at the bottom.

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

The screenshot shows the Jenkins New Item configuration page. It enters the item name 'advdevops07' and selects the 'Freestyle project' type. It also lists other project types: Maven project, Pipeline, and Multi-configuration project. A 'Save' button is visible at the bottom.

9. Choose this GitHub repository in Source Code Management.

https://github.com/shazforiot/MSBuild_firstproject.git

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

Source Code Management



10. Under **Select project → Configuration → Build steps → Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

The screenshot shows the 'Configure' section of a build pipeline. The 'Build Environment' tab is selected. A context menu is open over the 'Build Steps' section, listing various actions: Execute SonarQube Scanner, Execute Windows batch command, Execute shell, Invoke Ant, Invoke Gradle script, Invoke top-level Maven targets, Run with timeout, Set build status to "pending" on GitHub commit, SonarScanner for MSBuild - Begin Analysis, and SonarScanner for MSBuild - End Analysis. Below the menu, there is an 'Add build step' button. At the bottom, there is a 'Post-build Actions' section with an 'Add post-build action' dropdown and 'Save' and 'Apply' buttons.

JDK ?

JDK to be used for this SonarQube analysis

(Inherit From Job)

Path to project properties ?

Analysis properties ?

```
sonar.projectKey=advdevops7  
sonar.host.url=http://localhost:9000  
sonar.login=admin  
sonar.sources=.
```

Additional arguments ?

JVM Options ?

Add build step ▾

Save

Apply

Then save

Status **adv_devops_exp7** Add description Disable Project

Changes Workspace Build Now Configure Delete Project SonarQube Rename

Permalinks

- Last build (#2), 1 day 20 hr ago
- Last stable build (#2), 1 day 20 hr ago
- Last successful build (#2), 1 day 20 hr ago
- Last completed build (#2), 1 day 20 hr ago

11. Go to http://localhost:9000/<user_name>/permissions and allow Execute Permissions to the Admin user.

Administration

Configuration Security Projects System Marketplace

Global Permissions

Grant and revoke permissions to make changes at the global level. These permissions include editing Quality Profiles, executing analysis, and performing global system administration.

	Administer System ?	Administer ?	Execute Analysis ?	Create ?
sonar-administrators System administrators	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input checked="" type="checkbox"/> Projects
sonar-users Every authenticated user automatically belongs to this group	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
Anyone DEPRECATED Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects
Administrator admin	<input checked="" type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects

IF CONSOLE OUTPUT FAILED:

Step 1: Generate a New Authentication Token in SonarQube

1. Login to SonarQube:

- Open your browser and go to `http://localhost:9000`.
- Log in with your admin credentials (default username is `admin`, and the password is either `admin` or your custom password if it was changed).

2. Generate a New Token:

- Click on your **username** in the top-right corner of the SonarQube dashboard.
- Select **My Account** from the dropdown menu.
- Go to the **Security** tab.
- Under **Generate Tokens**, type a name for the token (e.g., "Jenkins-SonarQube").
- Click **Generate**.
- Copy the token and save it securely. You will need it in Jenkins.

Step 2: Update the Token in Jenkins

1. Go to Jenkins Dashboard:

- Open Jenkins and log in with your credentials.

2. Configure the Jenkins Job:

- Go to the job that is running the SonarQube scanner (`adv_devops_exp7`).
- Click **Configure**.

3. Update the SonarQube Token:

- In the SonarQube analysis configuration (either in the pipeline script or under "Build" section, depending on your job type), update the `sonar.login` parameter with the new token.

The screenshot shows the Jenkins job configuration page for the 'adv_devops_exp7' job. It includes sections for JDK selection, project properties, analysis properties (containing SonarQube configuration), additional arguments, and JVM options.

JDK ?
JDK to be used for this SonarQube analysis
(Inherit From Job)

Path to project properties ?
[Empty input field]

Analysis properties ?
sonar.projectKey=sonarqube
sonar.projectName=advdevops_7
sonar.projectVersion=1.0
sonar.sources=.
sonar.host.url=http://localhost:9000
sonar.login=sqa_6e1401709960bcb21f8f8a53a65633a2a97ca501
sonar.projectBaseDir=C:/ProgramData/Jenkins/jenkins/workspace/advdevops07

Additional arguments ?
[Empty input field]

JVM Options ?
[Empty input field]

12. Run the Jenkins build.

✓ advdevops07

SonarQube

Permalinks

- Last build (#4), 3 min 20 sec ago
- Last stable build (#4), 3 min 20 sec ago
- Last successful build (#4), 3 min 20 sec ago
- Last failed build (#3), 5 min 14 sec ago
- Last unsuccessful build (#3), 5 min 14 sec ago
- Last completed build (#4), 3 min 20 sec ago

Check the console Output

✓ Console Output

Download Copy View as

```
Started by user Bhushan Shriram Malpani
Running as SYSTEM
Building on the built-in node in workspace C:\ProgramData\Jenkins\.jenkins\workspace\advdevops07
[advdevops07] $ C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\exp07\bin\sonar-scanner.bat -
Dsonar.host.url=https://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.projectName=advdevops_7 -Dsonar.host.url=http://localhost:9000 -
Dsonar.login=sqa_6e1401709960bc21f8f8a53a65633a2a97ca501 -Dsonar.projectBaseDir=C:/ProgramData/Jenkins/.jenkins/workspace/advdevops07 -
Dsonar.projectVersion=1.0 -Dsonar.sources=.
14:42:11.487 WARN Property 'sonar.host.url' with value 'https://localhost:9000' is overridden with value 'http://localhost:9000'
14:42:11.495 INFO Scanner configuration file:
C:\ProgramData\Jenkins\.jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\exp07\bin..\conf\sonar-scanner.properties
14:42:11.496 INFO Project root configuration file: NONE
14:42:11.508 INFO SonarScanner CLI 6.2.0.4584
14:42:11.510 INFO Java 21.0.4 Eclipse Adoptium (64-bit)
14:42:11.513 INFO Windows 11 10.0 amd64
14:42:11.530 INFO User cache: C:\windows\system32\config\systemprofile\.sonar\cache
14:42:11.955 INFO JRE provisioning: os[windows], arch[amd64]
```

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

☆ advdevops_7 PUBLIC

Passed

Last analysis: 7 minutes ago

The main branch of this project is empty.

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

Conclusion:

In this project, we combined Jenkins with SonarQube to automate static application security testing (SAST). SonarQube was deployed via Docker, while Jenkins was configured with the required plugins and authentication. It was then connected to a GitHub repository. The SonarQube scanner was incorporated as a build step, facilitating ongoing code analysis for vulnerabilities, code quality issues, and other concerns, ensuring continuous improvement through automated reports.

08 Advanced DevOps Lab

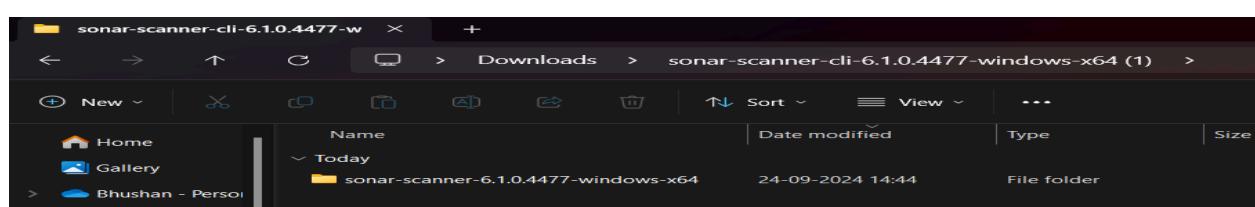
Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Step 1: Download sonar scanner

<https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/>
Visit this link and download the sonarqube scanner CLI.

The SonarScanner CLI is the scanner to use when there is no specific scanner for your build system. The SonarScanner does not yet officially support ARM architecture. Still, early adopters reported it is working fine. If you encounter problems, don't hesitate to share your experience with us on the [SonarQube](#) or [SonarCloud](#) Community Forum but keep in mind that there is no support at this time.

Extract the downloaded zip file in a folder.



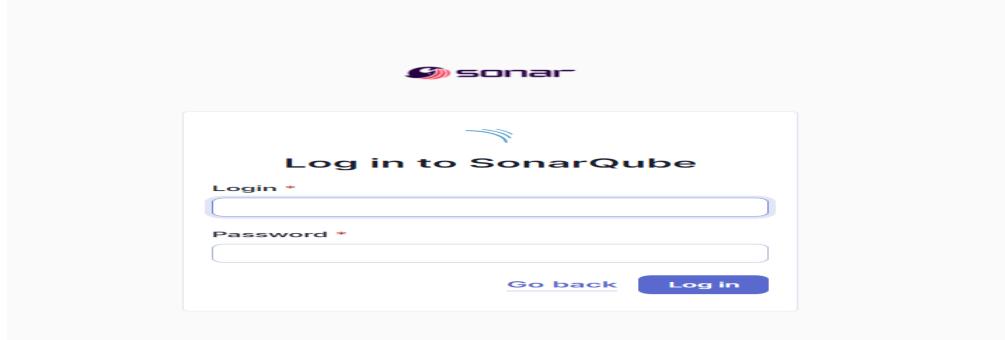
1. Install sonarqube image

Command: `docker pull sonarqube`

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

C:\Users\HP>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
7478a0ac0f23: 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
787074c97d3df5c971edeea44247a0b0b0dbfc9e0e3db74c6d3c61a99180824c
```

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



3. Login to SonarQube using username admin and password admin.

A screenshot of the SonarQube project creation wizard. At the top, there's a navigation bar with links for "Projects", "Issues", "Rules", "Quality Profiles", "Quality Gates", "Administration", "More", and a search bar. Below the navigation bar, the title "How do you want to create your project?" is displayed. A note below it says: "Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform." It also mentions: "First, you need to set up a DevOps platform configuration." There are five options for importing projects: "Import from Azure DevOps" (Setup), "Import from Bitbucket Cloud" (Setup), "Import from Bitbucket Server" (Setup), "Import from GitHub" (Setup), and "Import from GitLab" (Setup). Below these, a note says: "Are you just testing or have an advanced use-case? Create a local project." A "Create a local project" button is shown. At the very bottom of the page, there's a footer with links for "Documentation", "Community", "Blog", "API", "Feedback", and "Help".

4. Create a manual project in SonarQube with the name sonarqube

1 of 2

Create a local project

Project display name *

sonarqube

✓

Project key *

sonarqube

✓

Main branch name *

main

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

[Cancel](#)

[Next](#)

2 of 2

Set up project for Clean as You Code

The new code definition sets which part of your code will be considered new code. This helps follow the Clean as You Code methodology. Learn more: [Defining New Code](#)

Choose the baseline for new code for this project

Use the global setting

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

Define a specific setting for this project

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

The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with links like 'New Item', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins', 'My Views', 'Build Queue' (empty), 'Build Executor Status' (with 'Built-In Node' and two idle nodes), and a slave named 'slave1' (offline). The main area has a search bar and a user info bar for 'Bhushan Shriram Malpani'. Below is a table of build history:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		advdevops07	1 day 0 hr #4	1 day 0 hr #3	14 sec
		Devops Pipeline	1 mo 21 days #2	N/A	4.5 sec
		DevopsExp06	25 days #1	N/A	5.4 sec
		devopsPipeline	1 mo 2 days #2	N/A	1.2 sec
		MyProject	25 days #3	N/A	0.22 sec
		Tomcat1	N/A	N/A	N/A
		webApp	25 days #7	25 days #8	7.2 sec

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

The screenshot shows the 'Manage Jenkins > Plugins' page. The sidebar has 'Updates' (27), 'Available plugins' (selected), 'Installed plugins', and 'Advanced settings'. A search bar at the top is set to 'sonar'. The results table shows the 'SonarQube Scanner' plugin installed:

Install	Name ↓	Released
	SonarQube Scanner 2.17.2	7 mo 6 days ago
	Sonar Quality Gates 315.vf112b_e81a_3a_4	27 days ago
	Quality Gates 2.5	

Dashboard > Manage Jenkins > Plugins

The screenshot shows the 'Manage Jenkins > Plugins' page with 'Download progress' selected in the sidebar. It shows the SonarQube Scanner plugin is being downloaded:

- Preparation:**
 - Checking internet connectivity
 - Checking update center connectivity
 - Success
- SonarQube Scanner:** Success
- Loading plugin extensions:** Success

Buttons include 'Go back to the top page' and 'Restart Jenkins when installation is complete and no jobs are running'.

7. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for **SonarQube Servers** and enter the details.

Enter the Server Authentication token if needed.

In SonarQube installations: Under **Name** add <project name of sonarqube> for me
adv_devops_7_sonarqube

In **Server URL** Default is **http://localhost:9000**

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Environment variables

SonarQube installations

List of SonarQube installations

Name	X
sonarqube	
Server URL Default is http://localhost:9000 <input type="text" value="https://localhost:9000"/>	
Server authentication token <small>SonarQube authentication token. Mandatory when anonymous access is disabled.</small> <div style="border: 1px solid #ccc; padding: 2px; width: 100%;">- none -</div> <div style="margin-top: 5px; border: 1px solid #ccc; padding: 2px; width: 100%;">+ Add ▾</div>	

[Save](#)
Apply

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

Dashboard > Manage Jenkins > Tools

Gradle installations

[Add Gradle](#)

SonarScanner for MSBuild installations

[Add SonarScanner for MSBuild](#)

SonarQube Scanner installations

[SonarQube Scanner installations ▾](#) / Edited

Check the “Install automatically” option. → Under name any name as identifier → Check the “Install automatically” option.

SonarQube Scanner installations

The screenshot shows the Jenkins configuration interface for SonarQube Scanner installations. A new installation is being added with the name 'sonarqube_08'. The 'Install automatically' checkbox is checked, and the 'Install from Maven Central' option is selected, with the version set to 'SonarQube Scanner 6.2.0.4584'. An 'Add Installer' button is visible at the bottom.

9. After configuration, create a New Item → choose a pipeline project.

New Item

Enter an item name

advdevop_exp08

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.

OK

10. Under Pipeline script, enter the following:

```
node {  
stage('Cloning the GitHub Repo') {  
git 'https://github.com/shazforiot/GOL.git'
```

```

}

stage('SonarQube analysis') {
    withSonarQubeEnv('<Name_of_SonarQube_environment_on_Jenkins>') {
        sh """
            <PATH_TO SONARQUBE_SCANNER_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=<SonarQube_URL>(default: http://localhost:9000/)
        """
    }
}

```

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

Pipeline

Definition

Pipeline script

Script ?

```

1 node {
2     stage('Cloning the GitHub Repo') {
3         git 'https://github.com/shazforiot/GOL.git'
4     }
5
6     stage('SonarQube analysis') {
7         withSonarQubeEnv('sonarqube') {
8             bat """
9                 "C:\\\\Users\\\\HP\\\\Downloads\\\\sonar-scanner-cli-6.1.0.4477-windows-x64 (1)\\\\sonar-scanner-6.1.0.4477-windows-x64\\\\bin\\\\sona
10                -D sonar.login=admin ^
11                -D sonar.password=#bhushan45 ^
12                -D sonar.projectKey=sonarqube ^
13                -D sonar.exclusions=vendor/**,resources/**, **/*.java ^
14                -D sonar.host.url=http://localhost:9000
15            """
16        }
17    }
18

```

try sample Pipeline... ▾

Use Groovy Sandbox ?

Save **Apply**

11. Build project

The screenshot shows a CI pipeline interface with a sidebar on the left containing options like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Stages, Rename, and Pipeline Syntax. Below this is a Build History section with four entries: #4 (Sep 24, 2024, 3:22 PM), #3 (Sep 24, 2024, 3:20 PM), #2 (Sep 24, 2024, 3:18 PM), and #1 (Sep 24, 2024, 3:16 PM). The main area is titled "Stage View" and displays two stages: "Cloning the GitHub Repo" and "SonarQube analysis". The "Cloning the GitHub Repo" stage has an average time of 15s. The "SonarQube analysis" stage has an average time of 7min 12s. The builds are arranged in a grid:

Build	Time	Status
#4	2s	Success
#3	14s	Failed (232ms)
#2	25s	Failed
#1	21s	Failed

Permalinks

- Last build (#3), 2 min 0 sec ago

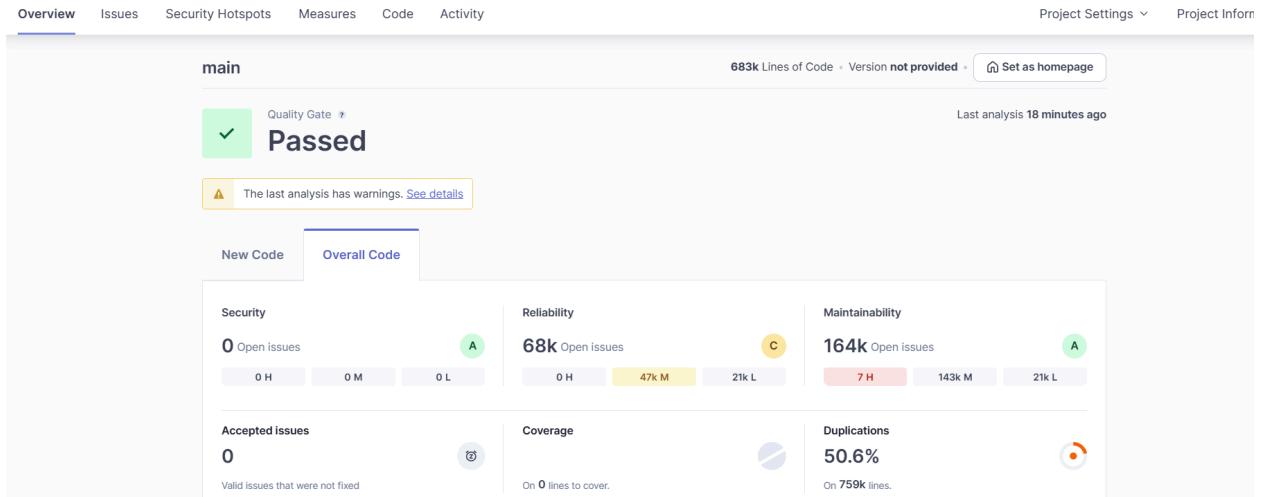
12. Check console

Console Output

Skipping 4,250 KB.. [Full Log](#)

```
15:36:08.390 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 789. Keep only the first 100 references.
15:36:08.390 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 512. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 248. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 886. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 249. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 662. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 615. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 664. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 913. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 810. Keep only the first 100 references.
15:36:08.391 WARN Too many duplication references on file gameoflife-web/tools/jmeter/docs/api/org/apache/jmeter/visualizers/PropertyControlGui.html
for block at line 668. Keep only the first 100 references.
```

13. Now, check the project in SonarQube



14. Code Problems

- Consistency

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

[Insert a <!DOCTYPE> declaration to before this <html> tag.](#) Consistency

Reliability ↕ user-experience +

Open ↕ Not assigned ↕ L1 • 5min effort • 4 years ago • Bug • Major

[Add "lang" and/or "xml:lang" attributes to this "<html>" element](#) Intentionality

Reliability ↕ accessibility wcag2-a +

Open ↕ Not assigned ↕ L1 • 2min effort • 4 years ago • Bug • Major

[Remove this deprecated "width" attribute.](#) Consistency

Maintainability ↕ html5 obsolete +

- Intentionality

Bulk Change Select issues ▲ ▼ Navigate to issue ⏪ ⏩ 210,549 issues 3135d effort

gameoflife-acceptance-tests/Dockerfile

Use a specific version tag for the image. Intentionality
Maintainability No tags +

Open ▾ Not assigned ▾ L1 • 5min effort • 4 years ago • ⚡ Code Smell • ⚡ Major

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality
Maintainability No tags +

Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚡ Code Smell • ⚡ Major

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality
Maintainability No tags +

● Bugs

Bulk Change Select issues ▲ ▼ Navigate to issue ⏪ ⏩ 67,624 issues 1646d effort

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

Add "lang" and/or "xml:lang" attributes to this "<html>" element Intentionality
Reliability accessibility wcag2-a +

Open ▾ Not assigned ▾ L1 • 2min effort • 4 years ago • ⚡ Bug • ⚡ Major

Insert a <!DOCTYPE> declaration to before this <html> tag. Consistency
Reliability user-experience +

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

Add "<th>" headers to this "<table>". Intentionality
Reliability accessibility wcag2-a +

Open ▾ Not assigned ▾ L9 • 2min effort • 4 years ago • ⚡ Bug • ⚡ Major

● Code Smells

Bulk Change Select issues ▲ ▼ Navigate to issue ⏪ ⏩ 163,781 issues 1705d effort

gameoflife-acceptance-tests/Dockerfile

Use a specific version tag for the image. Intentionality
Maintainability No tags +

Open ▾ Not assigned ▾ L1 • 5min effort • 4 years ago • ⚡ Code Smell • ⚡ Major

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality
Maintainability No tags +

Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚡ Code Smell • ⚡ Major

Surround this variable with double quotes; otherwise, it can lead to unexpected behavior. Intentionality
Maintainability No tags +

Open ▾ Not assigned ▾ L12 • 5min effort • 4 years ago • ⚡ Code Smell • ⚡ Major

Filters Clear All Filters

Issues in new code

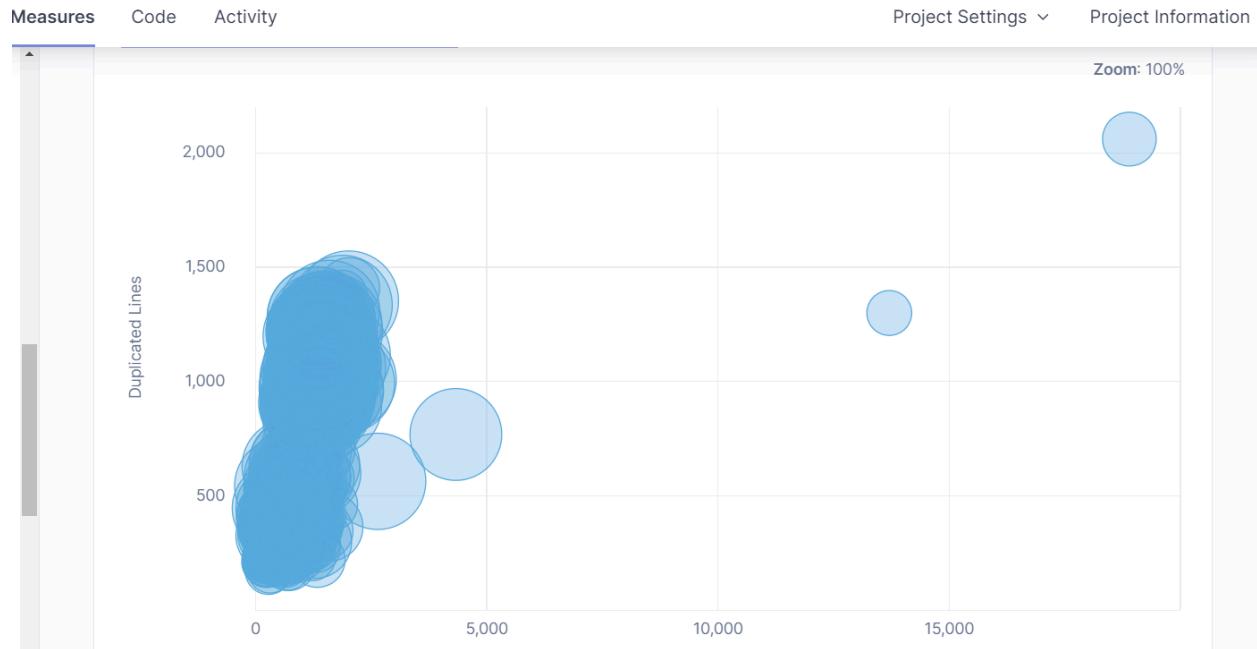
- ✓ Clean Code Attribute

Consistency	164k
Intentionality	15
Adaptability	0
Responsibility	0
- ✓ Software Quality

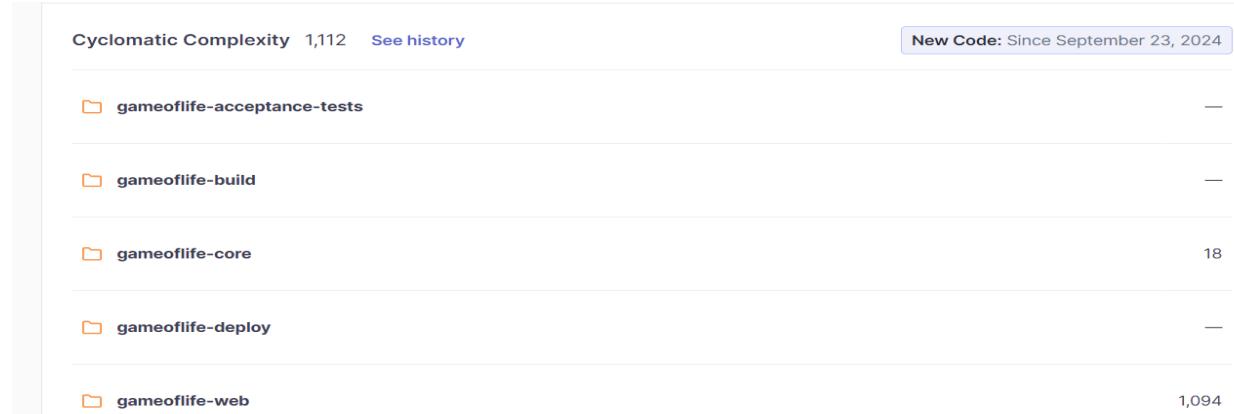
Security	0
Reliability	68k
Maintainability	164k

Add to selection Ctrl + click

- Duplications



- Cyclomatic Complexities



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

Conclusion:

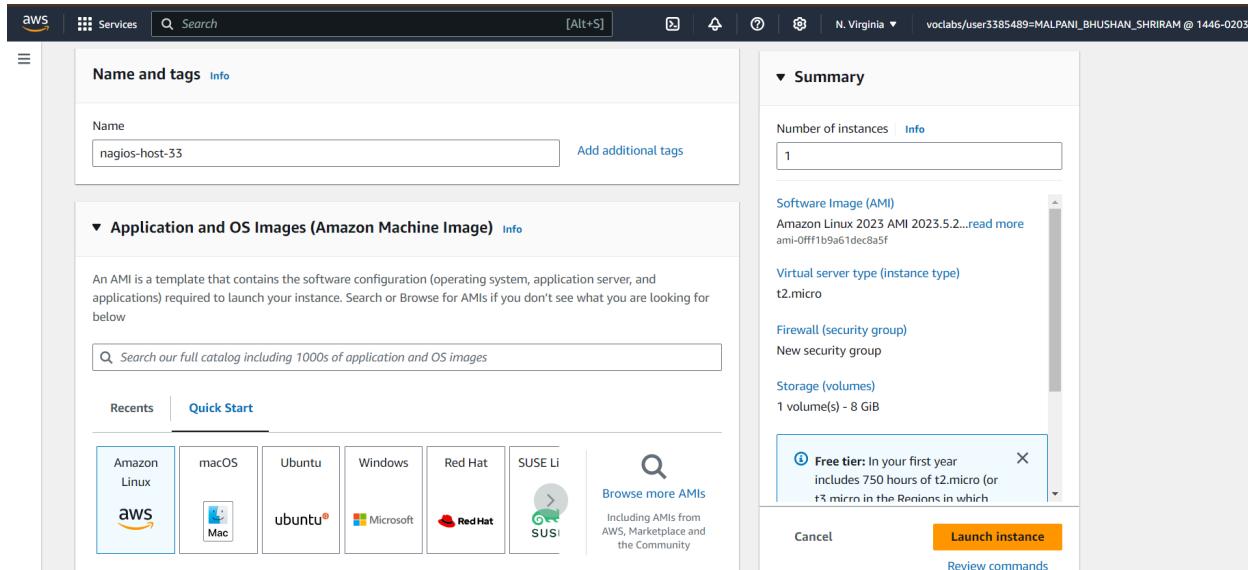
In this experiment, we integrated Jenkins with SonarQube to implement automated code quality checks within our CI/CD pipeline. The process began with deploying SonarQube using Docker, followed by setting up a dedicated project within SonarQube to perform detailed code quality analysis. We then configured Jenkins by installing the SonarQube Scanner plugin, which allowed Jenkins to communicate with SonarQube. This included adding the necessary SonarQube server details and configuring the scanner tool to analyze code efficiently.

A Jenkins pipeline was developed to automate key tasks, such as cloning the code from a GitHub repository and triggering the SonarQube analysis on the codebase. Through this

integration, the pipeline continuously monitors code quality, providing detailed reports on issues such as bugs, code smells, and security vulnerabilities. This setup not only ensures early detection of potential problems but also promotes consistent improvements in code quality throughout the development lifecycle.

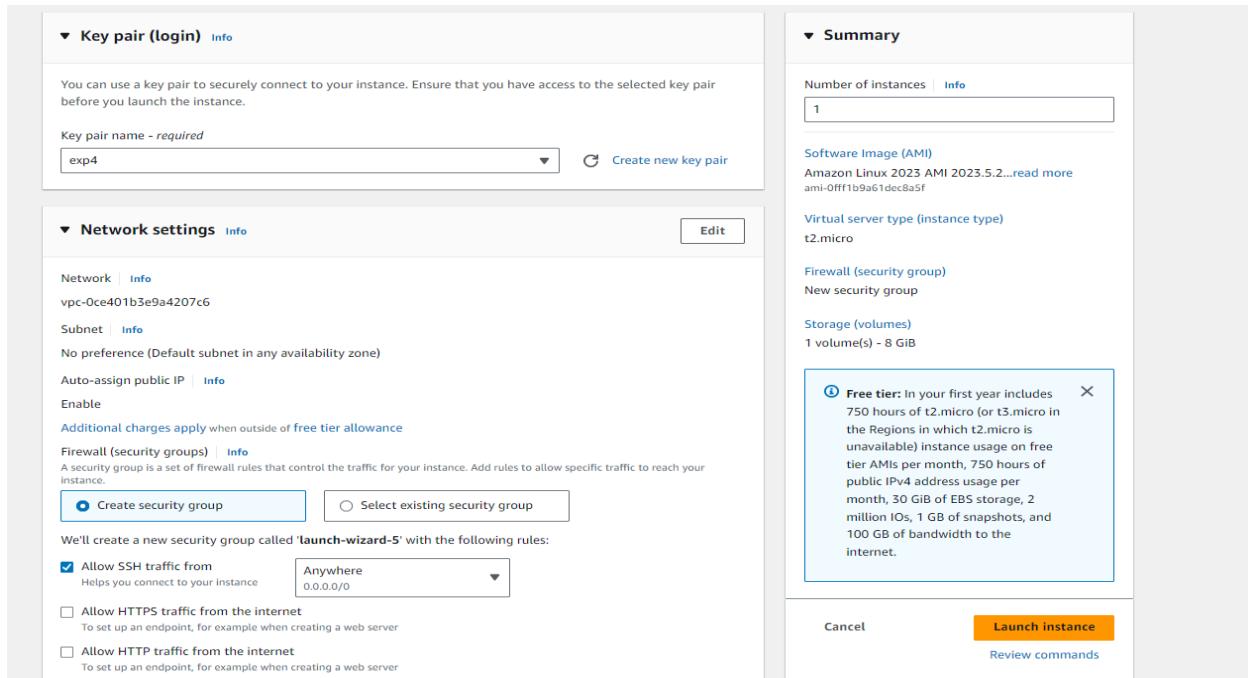
Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Step 1: Login to your AWS account. Search for EC2 on services. Open the interface and click on Create Instance.



Select The OS Image as Amazon Linux.

Step 2: If you do not have a private key created or a .pem file created, click on create a key pair. Else select the key pair that you had created before. (Make sure you know where the .pem file for that key is present on your system)



AWS will create a security group for this instance. Keep the name of that instance saved.
Step 3: After creating the instance, click on Security Groups from the left side pane. Find the security group that was created for your instance. Click on the instance ID for that group.

Security Groups (7) Info						
<input type="checkbox"/>	Name	▼	Security group ID	▼	Security group name	▼
<input type="checkbox"/>	-		sg-0a9f9cadd50e915a1		launch-wizard-5	
<input type="checkbox"/>	-		sg-06f8c60c04c8bd2f		launch-wizard-3	
<input type="checkbox"/>	-		sg-032ce5dafb9258e83		launch-wizard-1	
<input type="checkbox"/>	-		sg-0b1a0b0282b21848b		launch-wizard-4	

Here, click on Edit Inbound Rules.

EC2 > Security Groups > sg-0a9f9cadd50e915a1 - launch-wizard-5	Actions ▾																																
<h3>sg-0a9f9cadd50e915a1 - launch-wizard-5</h3>																																	
Details																																	
Security group name launch-wizard-5	Security group ID sg-0a9f9cadd50e915a1																																
Description launch-wizard-5 created 2024-10-06T08:45:27.524Z	VPC ID vpc-0ce401b3e9a4207c6																																
Owner 144602037631	Inbound rules count 1 Permission entry																																
Outbound rules count 1 Permission entry																																	
Inbound rules Outbound rules Tags																																	
<h4>Inbound rules (1)</h4>																																	
<table border="1"> <thead> <tr> <th><input type="checkbox"/></th> <th>Name</th> <th>▼</th> <th>Security group rule...</th> <th>▼</th> <th>IP version</th> <th>▼</th> <th>Type</th> <th>▼</th> <th>Protocol</th> <th>▼</th> <th>Port range</th> <th>▼</th> <th>Source</th> <th>▼</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td>sg-0a9f9cadd50e915a1</td> <td></td> <td>IPv4</td> <td></td> <td>tcp</td> <td></td> <td>all</td> <td></td> <td>TCP</td> <td></td> <td>22</td> <td></td> <td>0.0.0.0/0</td> <td></td> <td></td> </tr> </tbody> </table>		<input type="checkbox"/>	Name	▼	Security group rule...	▼	IP version	▼	Type	▼	Protocol	▼	Port range	▼	Source	▼	Description	<input type="checkbox"/>	sg-0a9f9cadd50e915a1		IPv4		tcp		all		TCP		22		0.0.0.0/0		
<input type="checkbox"/>	Name	▼	Security group rule...	▼	IP version	▼	Type	▼	Protocol	▼	Port range	▼	Source	▼	Description																		
<input type="checkbox"/>	sg-0a9f9cadd50e915a1		IPv4		tcp		all		TCP		22		0.0.0.0/0																				
<input type="checkbox"/> Manage tags Edit inbound rules																																	

Now, click on add rules, and add teh rules for the following protocols:
HTTP, All ICMP - IPv6, HTTPS, All traffic, Custom TCP (Port 5666), All ICMP - IPv4

Security group rule ID	Type	Info	Protocol	Info	Port range	Info	Source	Info	Description - optional	Info
sgr-0b9e8237df8a2ba26	SSH		TCP		22		Custom		0.0.0.0/0	X
-	HTTP		TCP		80		Anywhere-I...		0.0.0.0/0	X
-	All ICMP - IPv6		IPv6 ICMP		All		Anywhere-I...		0.0.0.0/0	X
-	HTTPS		TCP		443		Anywhere-I...		0.0.0.0/0	X
-	All traffic		All		All		Anywhere-I...		0.0.0.0/0	X
-	Custom TCP		TCP		5666		Anywhere-I...		0.0.0.0/0	X
-	All ICMP - IPv4		ICMP		All		Anywhere-I...		0.0.0.0/0	X

[Add rule](#)

Click on save. This will add all the inbound rules to the security group.

Inbound rules (7)										
	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description		
<input type="checkbox"/>	-	sgr-089969b97ae55c2c2	IPv4	HTTPS	TCP	443	0.0.0.0/0	-		
<input type="checkbox"/>	-	sgr-02eba0d00ef4936a3	IPv6	HTTP	TCP	80	::/0	-		
<input type="checkbox"/>	-	sgr-0a49f58bc41d26140	IPv4	All ICMP - IPv4	ICMP	All	0.0.0.0/0	-		
<input type="checkbox"/>	-	sgr-08b39a574c2b882...	IPv4	All traffic	All	All	0.0.0.0/0	-		
<input type="checkbox"/>	-	sgr-08f62562c25a7e75	IPv4	Custom TCP	TCP	5666	0.0.0.0/0	-		
<input type="checkbox"/>	-	sgr-0b9e8237df8a2ba26	IPv4	SSH	TCP	22	0.0.0.0/0	-		
<input type="checkbox"/>	-	sgr-04cc1e8ceda11c0d2	IPv6	All ICMP - IPv6	IPv6 ICMP	All	::/0	-		

Step 4: Now come back to the instances screen. Click on the instance ID of your instance. Then click on Connect.

The screenshot shows the AWS EC2 Instances page. The left sidebar includes links for EC2 Dashboard, EC2 Global View, Events, Console-to-Code, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, AMI Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups), CloudShell, and Feedback. The main content area displays 'Instances (1/1) Info' with a table showing one instance: 'nagios-host-33' (Instance ID: i-0c8e39056c4067e17, Status: Running, Instance type: t2.micro, Status check: Initializing). The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 IP, and Elastic IP. A 'Launch instances' button is at the top right.

Click on SSH client. Copy the example command.

The screenshot shows the 'Connect to instance' dialog for the instance 'i-0c8e39056c4067e17'. The navigation path is EC2 > Instances > i-0c8e39056c4067e17 > Connect to instance. The 'SSH client' tab is selected. The 'Instance ID' field contains 'i-0c8e39056c4067e17 (nagios-host-33)'. Below it is a numbered list of steps: 1. Open an SSH client., 2. Locate your private key file. The key used to launch this instance is exp4.pem, 3. Run this command, if necessary, to ensure your key is not publicly viewable. `chmod 400 "exp4.pem"`, 4. Connect to your instance using its Public DNS: `ec2-54-145-174-17.compute-1.amazonaws.com`. A callout bubble indicates 'Command copied' next to the public DNS. A note at the bottom states: '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.' A 'Cancel' button is at the bottom right.

Step 5: Now, we have to connect our local OS terminal to the instance using SSH. For this, Open terminal wher the private key file is located (.pem) Paste the copied SSH command and run it.

```
PS C:\Users\HP\Desktop\Exp4> ssh -i "exp4.pem" ec2-user@ec2-54-145-174-17.compute-1.amazonaws.com
The authenticity of host 'ec2-54-145-174-17.compute-1.amazonaws.com (64:ff9b::3691:ae11)' can't be established.
ED25519 key fingerprint is SHA256:20Wc5d/VkzKG8qqevKXZ4y2yXGmr+A6wL3E0ijQ4HZo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-145-174-17.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

  _#_
~\_\_ #####_      Amazon Linux 2023
~~ \_\#####\_
~~ \#\#
~~ \#/ _--> https://aws.amazon.com/linux/amazon-linux-2023
~~ \~' '--->
~~ \_/_/ / /
~~ \_/_/ / /
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 6: Now we start working on this terminal. First run the command
sudo yum update

This command will check for any updates for the YUM library.

```
[ec2-user@ip-172-31-39-94 ~]$ sudo yum update
Last metadata expiration check: 0:13:49 ago on Sun Oct  6 08:57:50 2024.
Dependencies resolved.
Nothing to do.
Complete!
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 7: We are going to install an Apache server and a PHP on this instance. For that, run this command.

```
sudo yum install httpd php
```

```
[ec2-user@ip-172-31-39-94 ~]$ sudo yum install httpd php
Last metadata expiration check: 0:14:13 ago on Sun Oct 6 08:57:50 2024.
Dependencies resolved.
=====
 Package          Architecture      Version       Repository    Size
=====
 Installing:
 httpd           x86_64          2.4.62-1.amzn2023
 php8.3          x86_64          8.3.10-1.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
 libsodium        x86_64          1.0.19-4.amzn2023
 libxml2          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          8.3.10-1.amzn2023.0.1
 php8.3-common   x86_64          8.3.10-1.amzn2023.0.1
 php8.3-process  x86_64          8.3.10-1.amzn2023.0.1
 php8.3-xm1      x86_64          8.3.10-1.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          8.3.10-1.amzn2023.0.1
=====
 Installed:
 apr-1.7.2-2.amzn2023.0.2.x86_64
 apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
 httpd-2.4.62-1.amzn2023.x86_64
 httpd-filesystem-2.4.62-1.amzn2023.noarch
 libbrotli-1.0.9-4.amzn2023.0.2.x86_64
 libxml2-1.1.34-5.amzn2023.0.2.x86_64
 mod_http2-2.0.27-1.amzn2023.0.3.x86_64
 nginx-filesystem-1:1.24.0-1.amzn2023.0.4.noarch
 php8.3-cli-8.3.10-1.amzn2023.0.1.x86_64
 php8.3-fpm-8.3.10-1.amzn2023.0.1.x86_64
 php8.3-opcache-8.3.10-1.amzn2023.0.1.x86_64
 php8.3-process-8.3.10-1.amzn2023.0.1.x86_64
 php8.3-xm1-8.3.10-1.amzn2023.0.1.x86_64
=====
 Complete!
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 8: Next we install C/C++ compiler (GCC) along with the necessary C libraries required for compiling and running C programs. Use the following command.

`sudo yum install gcc glibc glibc-common`

```
[ec2-user@ip-172-31-39-94 ~]$ sudo yum install gcc glibc glibc-common
Last metadata expiration check: 0:15:54 ago on Sun Oct 6 08:57:56 2024.
Package glibc-2.34-52.amzn2023.0.11.x86_64 is already installed.
Package glibc-common-2.34-52.amzn2023.0.11.x86_64 is already installed.
Dependencies resolved.
=====
 Package          Architecture      Version       Repository    Size
=====
 Installing:
 gcc             x86_64          11.4.1-2.amzn2023.0.2
=====
 Installing dependencies:
 annobin-docs    noarch        10.93-1.amzn2023.0.1
 annobin-plugin-gcc x86_64          10.93-1.amzn2023.0.1
 cpp             x86_64          11.4.1-2.amzn2023.0.2
 gc              x86_64          8.0.4-5.amzn2023.0.2
 glibc-devel     x86_64          2.34-52.amzn2023.0.11
 glibc-headers-x86 noarch        2.34-52.amzn2023.0.11
 guile22         x86_64          2.2.7-2.amzn2023.0.3
 kernel-headers  x86_64          6.1.109-118.189.amzn2023
 libmpc          x86_64          1.2.1-2.amzn2023.0.2
 libtool-ltdl    x86_64          2.4.7-1.amzn2023.0.3
 libcrypt-devel  x86_64          4.4.33-7.amzn2023
 make            x86_64          1:4.3-5.amzn2023.0.2
=====
 Transaction Summary
=====
 Install 13 Packages
=====
 Installed:
 annobin-docs-10.93-1.amzn2023.0.1.noarch
 cpp-11.4.1-2.amzn2023.0.2.x86_64
 gcc-11.4.1-2.amzn2023.0.2.x86_64
 glibc-headers-x86-2.34-52.amzn2023.0.11.noarch
 kernel-headers-6.1.109-118.189.amzn2023.x86_64
 libtool-ltdl-2.4.7-1.amzn2023.0.3.x86_64
 make-1:4.3-5.amzn2023.0.2.x86_64
=====
```

Step 9: We would also need GD library and its development tools. For that, run this command

```
sudo yum install gd gd-devel
```

```
[ec2-user@ip-172-31-39-94 ~]$ sudo yum install gd gd-devel
Last metadata expiration check: 0:17:02 ago on Sun Oct  6 08:57:50 2024.
Dependencies resolved.
=====
Package           Architecture Version      Repository   Size
=====
Installing:
gd                  x86_64      2.3.3-5.amzn2023.0.3    amazonlinux 139 k
gd-devel            x86_64      2.3.3-5.amzn2023.0.3    amazonlinux 38 k
Installing dependencies:
brotli              x86_64      1.0.9-4.amzn2023.0.2    amazonlinux 314 k
brotli-devel        x86_64      1.0.9-4.amzn2023.0.2    amazonlinux 31 k
bzip2-devel         x86_64      1.0.8-6.amzn2023.0.2    amazonlinux 214 k
cairo               x86_64      1.17.6-2.amzn2023.0.1    amazonlinux 684 k
cmake-fs            x86_64      3.22.2-1.amzn2023.0.4    amazonlinux 16 k
fontconfig          x86_64      2.13.94-2.amzn2023.0.2    amazonlinux 273 k
fontconfig-devel    x86_64      2.13.94-2.amzn2023.0.2    amazonlinux 128 k
fonts-fs            noarch     1:2.0.5-12.amzn2023.0.2    amazonlinux 9.5 k
freetype             x86_64      2.13.2-5.amzn2023.0.1    amazonlinux 423 k
freetype-devel      x86_64      2.13.2-5.amzn2023.0.1    amazonlinux 912 k
glib2-devel         x86_64      2.74.7-689.amzn2023.0.2    amazonlinux 486 k
google-noto-fonts-common x86_64  20201206-2.amzn2023.0.2    amazonlinux 15 k
google-noto-sans-vf-fonts x86_64  20201206-2.amzn2023.0.2    amazonlinux 492 k
graphite2            x86_64      1.3.14-7.amzn2023.0.2    amazonlinux 97 k
graphite2-devel     x86_64      1.3.14-7.amzn2023.0.2    amazonlinux 21 k
harfbuzz             x86_64      7.0.0-2.amzn2023.0.1    amazonlinux 868 k
harfbuzz-devel      x86_64      7.0.0-2.amzn2023.0.1    amazonlinux 404 k
harfbuzz-icu         x86_64      7.0.0-2.amzn2023.0.1    amazonlinux 18 k
jbigkit-libs         x86_64      2.1-21.amzn2023.0.2    amazonlinux 54 k
langpacks-core-font-en noarch    3.0-21.amzn2023.0.4    amazonlinux 10 k
=====
Installed:
brotli-1.0.9-4.amzn2023.0.2.x86_64
bzip2-devel-1.0.8-6.amzn2023.0.2.x86_64
cmake-fs-3.22.2-1.amzn2023.0.4.x86_64
fontconfig-devel-2.13.94-2.amzn2023.0.2.x86_64
freetype-2.13.2-5.amzn2023.0.1.x86_64
gd-2.3.3-5.amzn2023.0.3.x86_64
glib2-devel-2.74.7-689.amzn2023.0.2.x86_64
google-noto-sans-vf-fonts-20201206-2.amzn2023.0.2.noarch
graphite2-devel-1.3.14-7.amzn2023.0.2.x86_64
harfbuzz-devel-7.0.0-2.amzn2023.0.1.x86_64
jbigkit-libs-2.1-21.amzn2023.0.2.x86_64
libICE-1.0.10-6.amzn2023.0.2.x86_64
libX11-1.7.2-3.amzn2023.0.4.x86_64
libX11-devel-1.7.2-3.amzn2023.0.4.x86_64
libXau-1.0.9-6.amzn2023.0.2.x86_64
libXext-1.3.4-6.amzn2023.0.2.x86_64
libXpm-devel-3.5.15-2.amzn2023.0.3.x86_64
libXt-1.2.0-4.amzn2023.0.2.x86_64
libffi-devel-3.4.4-1.amzn2023.0.1.x86_64
libicu-devel-67.1-7.amzn2023.0.3.x86_64
libjpeg-turbo-devel-2.1.4-2.amzn2023.0.5.x86_64
libpng-2:1.6.37-10.amzn2023.0.6.x86_64
libselinux-devel-3.4-5.amzn2023.0.2.x86_64
libtiff-4.4.0-4.amzn2023.0.18.x86_64
libwebp-1.2.4-1.amzn2023.0.6.x86_64
libxcb-1.13.1-7.amzn2023.0.2.x86_64
libxml2-devel-2.10.4-1.amzn2023.0.6.x86_64
pcre2-utf16-10.40-1.amzn2023.0.3.x86_64
pixman-0.40.0-3.amzn2023.0.3.x86_64
xml-common-0.6.3-56.amzn2023.0.2.noarch
xz-devel-5.2.5-9.amzn2023.0.2.x86_64
=====
brotli-devel-1.0.9-4.amzn2023.0.2.x86_64
cairo-1.17.6-2.amzn2023.0.1.x86_64
fontconfig-2.13.94-2.amzn2023.0.2.x86_64
fonts-fs-1:2.0.5-12.amzn2023.0.2.noarch
freetype-devel-2.13.2-5.amzn2023.0.1.x86_64
gd-devel-2.3.3-5.amzn2023.0.3.x86_64
google-noto-fonts-common-20201206-2.amzn2023.0.2.noarch
graphite2-1.3.14-7.amzn2023.0.2.x86_64
harfbuzz-7.0.0-2.amzn2023.0.1.x86_64
harfbuzz-icu-7.0.0-2.amzn2023.0.1.x86_64
langpacks-core-font-en-3.0-21.amzn2023.0.4.noarch
libSM-1.2.3-8.amzn2023.0.2.x86_64
libX11-common-1.7.2-3.amzn2023.0.4.noarch
libX11-xcb-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
libXrender-0.9.10-14.amzn2023.0.2.x86_64
libblkid-devel-2.37.4-1.amzn2023.0.4.x86_64
libicu-67.1-7.amzn2023.0.3.x86_64
libjpeg-turbo-2.1.4-2.amzn2023.0.5.x86_64
libmount-devel-2.37.4-1.amzn2023.0.4.x86_64
libpng-devel-2:1.6.37-10.amzn2023.0.6.x86_64
libsepol-devel-3.4-3.amzn2023.0.3.x86_64
libtiff-devel-4.4.0-4.amzn2023.0.18.x86_64
libwebp-devel-1.2.4-1.amzn2023.0.6.x86_64
libxcb-devel-1.13.1-7.amzn2023.0.2.x86_64
pcre2-devel-10.40-1.amzn2023.0.3.x86_64
pcre2-utf32-10.40-1.amzn2023.0.3.x86_64
sysprof-capture-devel-3.40.1-2.amzn2023.0.2.x86_64
xorg-x11proto-devel-2021.4-1.amzn2023.0.2.noarch
zlib-devel-1.2.11-33.amzn2023.0.5.x86_64
=====
Complete!
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 10: Now, we create a user called as ‘nagios’ and make sure that it has a home directory, and set up a password for it.

```
sudo adduser -m nagios
```

```
sudo passwd nagios
```

```
[ec2-user@ip-172-31-39-94 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 11: Create a user group called as ‘nagcmd’ to execute nagios commands.

```
sudo groupadd nagcmd
```

```
[ec2-user@ip-172-31-39-94 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-39-94 ~]$ |
```

Step 12: Add users apache and nagios to this user group.

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

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

```
[ec2-user@ip-172-31-39-94 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
```

Step 13: We create a directory downloads, to store the files of nagios server that are downloaded.

```
mkdir ~/downloads
```

```
cd ~/downloads
```

```
[ec2-user@ip-172-31-39-94 ~]$ mkdir ~/downloads
cd ~/downloads
```

Step 14:Now we need to install the latest versions of nogios-core and nagios-plugins. Go to the respective websites and check whether a better version is available.If newer versions are available, then right click on the download button → Copy link address.

Paste this link address in place of the current link in command.

If not run these commands.

```
wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
```

```
[ec2-user@ip-172-31-39-94 downloads]$ wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz
--2024-10-06 09:18:48-- 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.gz      100%[=====] 1.97M 5.66MB/s    in 0.3s

2024-10-06 09:18:49 (5.66 MB/s) - 'nagios-4.5.5.tar.gz' saved [2065473/2065473]
```

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

```
[ec2-user@ip-172-31-39-94 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-10-06 09:19:14-- 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-2.4.11.tar.gz      100%[=====] 2.62M 6.48MB/s    in 0.4s

2024-10-06 09:19:15 (6.48 MB/s) - 'nagios-plugins-2.4.11.tar.gz' saved [2753049/2753049]
```

Step 15: Now, we need to extract nagios-core file into the same directory. For that, we will use tar command.

tar zxvf nagios-4.5.5.tar.gz

```
[ec2-user@ip-172-31-39-94 downloads]$ tar zxvf nagios-4.5.5.tar.gz
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/THANKS
nagios-4.5.5/UPGRADING
nagios-4.5.5/aclocal.m4
nagios-4.5.5/autoconf-macros/
nagios-4.5.5/autoconf-macros/.gitignore
nagios-4.5.5/autoconf-macros/CHANGELOG.md
nagios-4.5.5/autoconf-macros/LICENSE
nagios-4.5.5/autoconf-macros/LICENSE.md
nagios-4.5.5/autoconf-macros/README.md
nagios-4.5.5/autoconf-macros/add_group_user
nagios-4.5.5/autoconf-macros/ax_nagios_get_distrib
nagios-4.5.5/autoconf-macros/ax_nagios_get_files
nagios-4.5.5/autoconf-macros/ax_nagios_get_inetd
nagios-4.5.5/autoconf-macros/ax_nagios_get_init
nagios-4.5.5/autoconf-macros/ax_nagios_get_os
nagios-4.5.5/autoconf-macros/ax_nagios_get_paths
nagios-4.5.5/autoconf-macros/ax_nagios_get_ssl
```

```
nagios-4.5.5/xdata/.gitignore
nagios-4.5.5/xdata/Makefile.in
nagios-4.5.5/xdata/xcddefault.c
nagios-4.5.5/xdata/xcddefault.h
nagios-4.5.5/xdata/xodtemplate.c
nagios-4.5.5/xdata/xodtemplate.h
nagios-4.5.5/xdata/xpddefault.c
nagios-4.5.5/xdata/xpddefault.h
nagios-4.5.5/xdata/xrddefault.c
nagios-4.5.5/xdata/xrddefault.h
nagios-4.5.5/xdata/xsddefault.c
nagios-4.5.5/xdata/xsddefault.h
[ec2-user@ip-172-31-39-94 downloads]$ |
```

Step 16: We need to ensure that Nagios uses a specific group (in this case, `nagcmd`) for executing external commands.

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

An **error** could be encountered here: `./configure: no such path or directory`

Solution: Navigate to the nagios-4.5.5 folder in downloads. (version could vary)

Steps: ls

```
[ec2-user@ip-172-31-39-94 downloads]$ ls
nagios-4.5.5  nagios-4.5.5.tar.gz  nagios-plugins-2.4.11.tar.gz
```

- `cd nagios-4.5.5` (use the version shown by your `ls` command)
- `./configure --with-command-group=nagcmd`

Another **error** could be `Cannot find SSL headers.`

To solve this, we need to install OpenSSL Dev Library

Steps:

sudo yum install openssl-devel

```
[ec2-user@ip-172-31-39-94 downloads]$ sudo yum install openssl-devel
Last metadata expiration check: 0:23:37 ago on Sun Oct  6 08:57:50 2024.
Dependencies resolved.
=====
 Package           Architecture      Version       Repository   Size
=====
 Installing:
openssl-devel      x86_64          1:3.0.8-1.amzn2023.0.14  amazonlinux  3.0 M
Transaction Summary
=====
Install 1 Package

Total download size: 3.0 M
Installed size: 4.7 M
```

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

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
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 the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for strings.h... yes
```

```
*** Configuration summary for nagios 4.5.5 2024-09-17 ***:

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

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

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

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ |
```

Step 17: We need to compile all components of this software according to the instruction in the Makefile. For that, use this command:

make all

Then,

sudo make install

sudo make install-init

sudo make install-config

sudo make install-commandmode

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install: cannot stat 'nagios': No such file or directory
make[1]: *** [Makefile:188: install] Error 1
make[1]: Leaving directory '/home/ec2-user/nagios-4.5.5/base'
make: *** [Makefile:287: install] Error 2
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nagios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cgi /usr/local/nagios/etc/cgi.cgi
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/resource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/local/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/local/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/local/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/local/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/local/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/local/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switchover.cfg /usr/local/nagios/etc/objects/switchover.cfg
```

*** Config files installed ***

Remember, these are *SAMPLE* config files. You'll need to read the documentation for more information on how to actually define services, hosts, etc. to fit your particular needs.

```
/usr/bin/install -c -m 775 -o nagios -g nagioscmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw
```

*** External command directory configured ***

Step 18: We need to update the email linked with this server to our email for it to send notifications (if any needed).

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

```
ec2-user@ip-172-31-39-94:~/ | + | ~
GNU nano 5.8                               /usr/local/nagios/etc/objects/contacts.cfg
#####
# CONTACTS.CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS
#
#
# NOTES: This config file provides you with some example contact and contact
#         group definitions that you can reference in host and service
#         definitions.
#
#         You don't need to keep these definitions in a separate file from your
#         other object definitions. This has been done just to make things
#         easier to understand.
#
#####
#
# 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.bhushan.malpani@ves.ac.in ; <***** CHANGE THIS TO YOUR EMAIL ADDRESS *****

}
```

Here, change the email under ‘define contact{}’ to your email address.
To save this use the following shortcut sequence CTRL+O→Enter→CTRL+X.
CTRL+O: Overwrite the existing file with edited file
CTRL+X: Exit nano editor.

Step 19: We need to install the necessary configuration files for the Nagios web interface. Run the following command.

sudo make install-webconf

```
[ec2-user@ip-172-31-39-94 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 ***
```

Step 20: Now we need to setup a user to access this nagios web interface. So we run this command to create a user called ‘nagiosadmin’.

Keep this username and password saved as it is needed to login to the web interface.

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

```
[ec2-user@ip-172-31-39-94 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-172-31-39-94 nagios-4.5.5]$ |
```

Step 21: Restart the apache server to apply all the recent configurations.

sudo service httpd restart

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ |
```

Step 22: Now we go back to the downloads folder and extract the files of nagios plugin.

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz (Version may vary)

```
[ec2-user@ip-172-31-39-94 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/ltdmain.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/
nagios-plugins-2.4.11/config_test/Makefile
nagios-plugins-2.4.11/config_test/run_tests
```

```
nagios-plugins-2.4.11/plugins-root/t/check_dhcp.t
nagios-plugins-2.4.11/plugins-root/t/check_icmp.t
nagios-plugins-2.4.11/po/
nagios-plugins-2.4.11/po/Makefile.in.in
nagios-plugins-2.4.11/po/remove-potcdate.sin
nagios-plugins-2.4.11/po/Makevars
nagios-plugins-2.4.11/po/POTFILES.in
nagios-plugins-2.4.11/po/fr.po
nagios-plugins-2.4.11/po/de.po
nagios-plugins-2.4.11/po/fr.gmo
nagios-plugins-2.4.11/po/de.gmo
nagios-plugins-2.4.11/po/nagios-plugins.pot
nagios-plugins-2.4.11/po/stamp-po
nagios-plugins-2.4.11/po/ChangeLog
nagios-plugins-2.4.11/po/LINGUAS
nagios-plugins-2.4.11/release
[ec2-user@ip-172-31-39-94 downloads]$ |
```

Step 23: Again, we need to install the configurations for these files.

```
cd nagios-plugins-2.4.11 (version may vary)
./configure --with-nagios-user=nagios --with-nagios-group=nagios
```

```
[ec2-user@ip-172-31-39-94 ~]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-39-94 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 that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating gl/Makefile
config.status: creating nagios-plugins.spec
config.status: creating tools/build_perl_modules
config.status: creating Makefile
config.status: creating tap/Makefile
config.status: creating lib/Makefile
config.status: creating plugins/Makefile
config.status: creating lib/tests/Makefile
config.status: creating plugins-root/Makefile
config.status: creating plugins-scripts/Makefile
config.status: creating plugins-scripts/utils.pm
config.status: creating plugins-scripts/utils.sh
config.status: creating perlmods/Makefile
config.status: creating test.pl
config.status: creating pkg/solaris/pkginfo
config.status: creating po/Makefile.in
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
config.status: executing libtool commands
config.status: executing po-directories commands
config.status: creating po/POTFILES
config.status: creating po/Makefile
[ec2-user@ip-172-31-39-94 nagios-plugins-2.4.11]$ |
```

Step 24: We need to compile all components of this software according to the instruction in the Makefile. For that, use the commands:

```
make
sudo make install
```

```
fi
make[1]: Leaving directory '/home/ec2-user/nagios-plugins-2.4.11/po'
make[1]: Entering directory '/home/ec2-user/nagios-plugins-2.4.11'
make[2]: Entering directory '/home/ec2-user/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/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/nagios-plugins-2.4.11'
[ec2-user@ip-172-31-39-94 nagios-plugins-2.4.11]$ |
```

Step 25: We need to register the Nagios service with the system, which would make it able to manage the server status. So run the following commands

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

Step 26: We need to verify the Nagios configuration for any syntax errors or issues before starting or restarting the Nagios service.

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

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ 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...
Error: Unexpected token or statement in file '/usr/local/nagios/etc/objects/contacts.cfg' on line 1.
  Error processing object config files!

***> One or more problems was encountered while processing the config files...

Check your configuration file(s) to ensure that they contain valid
directives and data definitions. If you are upgrading from a previous
version of Nagios, you should be aware that some variables/definitions
may have been removed or modified in this version. Make sure to read
the HTML documentation regarding the config files, as well as the
'Whats New' section to find out what has changed.
```

Error: 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)

It is an error in processing main config file!

Solution: Create the missing directory, set the permissions, verify it.

- sudo mkdir -p /usr/local/nagios/var/spool/checkresults (Create)
- sudo chown nagios:nagios /usr/local/nagios/var/spool/checkresults
- sudo chmod 775 /usr/local/nagios/var/spool/checkresults (permissions)

Now rerun the command

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

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ 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

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ |
```

Step 27:

`sudo service nagios start`

```
[ec2-user@ip-172-31-39-94 nagios-4.5.5]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
```

`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 2024-10-06 10:41:35 UTC; 16s ago
     Docs: https://www.nagios.org/documentation
 Process: 67265 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Process: 67272 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
 Main PID: 67273 (nagios)
   Tasks: 6 (limit: 1112)
    Memory: 5.7M
      CPU: 65ms
     CGroup: /system.slice/nagios.service
             ├─67273 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─67275 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─67276 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─67277 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─67278 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─67318 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: qh: core query handler registered
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: qh: echo service query handler registered
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: qh: help for the query handler registered
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: wproc: Successfully registered manager as @wproc with query handler
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: wproc: Registry request: name=Core Worker 67277;pid=67277
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: wproc: Registry request: name=Core Worker 67278;pid=67278
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: wproc: Registry request: name=Core Worker 67275;pid=67275
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: wproc: Registry request: name=Core Worker 67276;pid=67276
Oct 06 10:41:35 ip-172-31-39-94.ec2.internal nagios[67273]: Successfully launched command file worker with pid 67318
```

Step 28: Now, go to EC2 instance and click on instance id. Then, click on the copy icon just before the public ip address on public IP.

Instance summary for i-0c8e39056c4067e17 (nagios-host-33) Info	
Instance ID	i-0c8e39056c4067e17 (nagios-host-33)
IPv6 address	-
Hostname type	IP name: ip-172-31-39-94.ec2.internal
Answer private resource DNS name	IPv4 (A)
Auto-assigned IP address	54.145.174.17 [Public IP]
IAM Role	-
IMDSv2	Required
Details Status and alarms Monitoring Security Networking Storage Tags	
Instance details Info	
Platform	Amazon Linux (Inferred)
AMI ID	ami-0fff1b9a61dec8a5f
AMI name	al2023-ami-2023.5.20241001.1-kernel-6.1-x86_64
Launch time	2024-05-17T17:45:22Z
Monitoring	disabled
Termination protection	Disabled
AMI location	Amazon S3

Step 29: Open a new tab. In the address bar type <http://<publicipaddress>/nagios>. This would be in the output

Conclusion:

In this experiment, we have learned how to install and configure Nagios Core, Nagios Plugins and NRPE on a Linux machine. We are using an Amazon Linux OS instance configured with the need security rules. We need to make sure that the Nagios-core and Nagios-plugins links that are used are the ones which are up-to date (wget commands). It is needed to extract and configure these files so that no issues are detected while starting the server. Once all the setup

is complete, we can start the nagios server. Using the public IP address of the EC2 instance, we can access the nagios dashboard by navigating that IP to nagios.

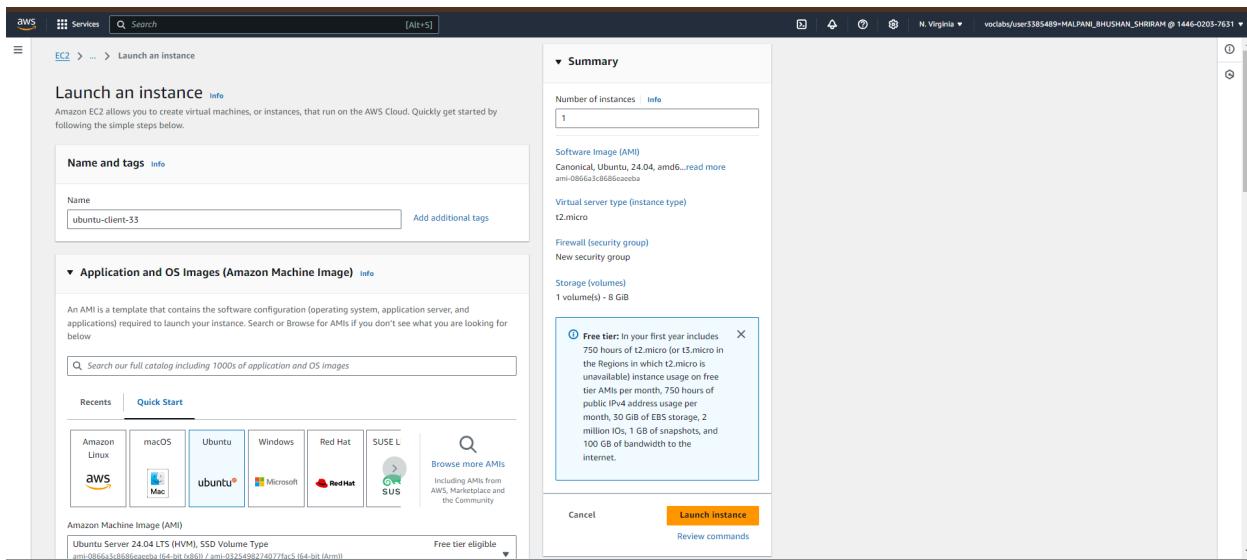
Aim: To perform Port, Service monitoring, Windows/Linux server monitoring using Nagios.

Prerequisites:

- 1) An Amazon Linux instance with nagios already set up.

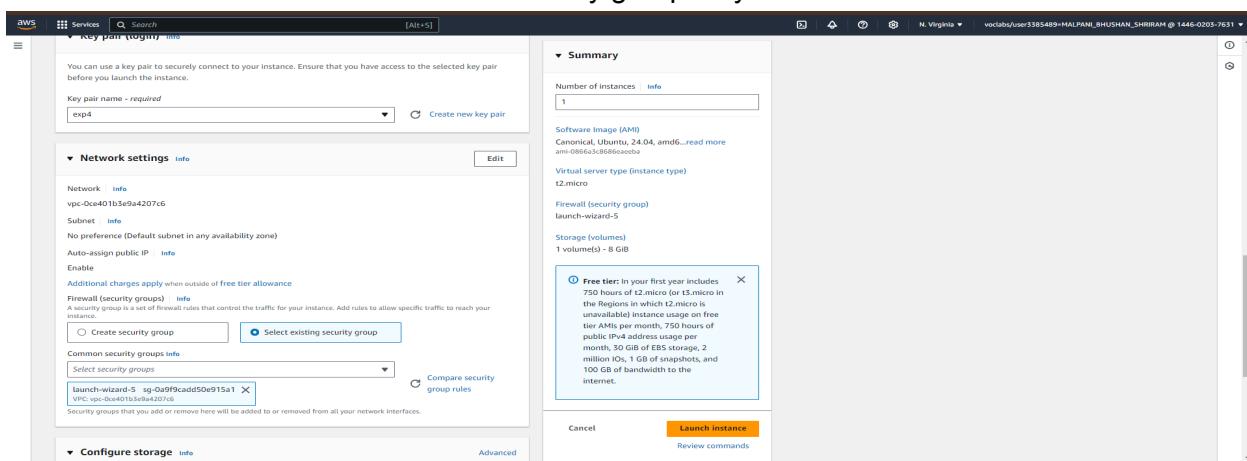
Step 1: Set up ubuntu instance

- 1) Login to your AWS account. Search for EC2 on services. Open the interface and click on Create Instance.



Select The OS Image as Ubuntu.

- 2) Make sure to select the same private key that you created for the Amazon Linux instance. Also select the same security group as you created for the Linux instance.



- 3) Now come back to the instances screen. Click on the instance ID of your instance. Then click on Connect. Click on SSH client. Copy the example command. Now, we have to connect our local OS terminal to the instance using SSH. For this, open terminal where the private key file is located (.pem). Paste the copied SSH command and run it.

Step 2: Execute the following on Nagios Host machine (Linux)

- 1) We need to verify whether the nagios service is running or not. For that, run this command.

```
ps -ef | grep nagios
```

```
ubuntu@ip-172-31-36-126:~$ ps -ef | grep nagios
ubuntu      1151      1139  0 11:12 pts/0    00:00:00 grep --color=auto nagios
```

- 2) Now, make yourself as the root user, and create a folder with the path '/usr/local/nagios/etc/objects/monitorhosts/linuxhosts'

```
sudo su
```

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

```
ubuntu@ip-172-31-36-126:~$ sudo su
mkdir -p /usr/local/nagios/etc/objects/monitorhosts/linuxhosts
root@ip-172-31-36-126:/home/ubuntu# |
```

- 3) 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
```

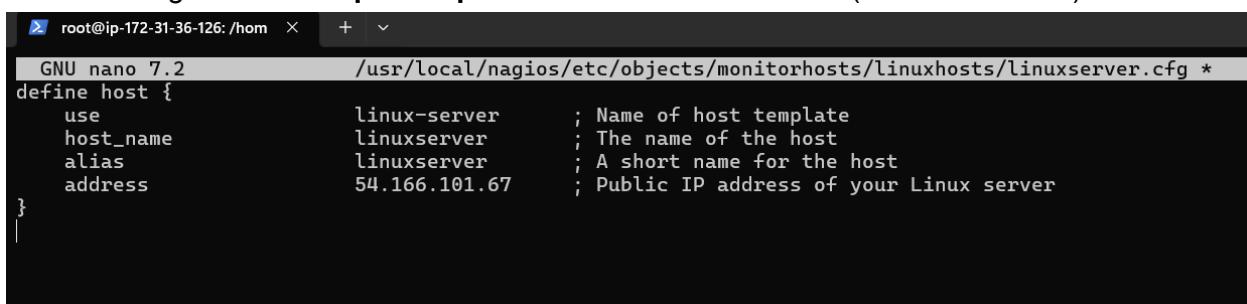
```
/usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
```

- 4) 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
```

Change **hostname** and **alias** to **linuxserver**

Change address to **public ip address of client instance** (Ubuntu instance)



```
GNU nano 7.2          /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg *
define host {
    use           linux-server      ; Name of host template
    host_name     linuxserver       ; The name of the host
    alias         linuxserver       ; A short name for the host
    address       54.166.101.67     ; Public IP address of your Linux server
}
```

Change hostgroup_name to **linux-servers1**

```
define hostgroup{
    hostgroup_name      linux-server1 ; The name of the hostgroup
    alias               linux Servers ; Long name of the group
    members             localhost     ; Comma separated list of hosts to>
}
```

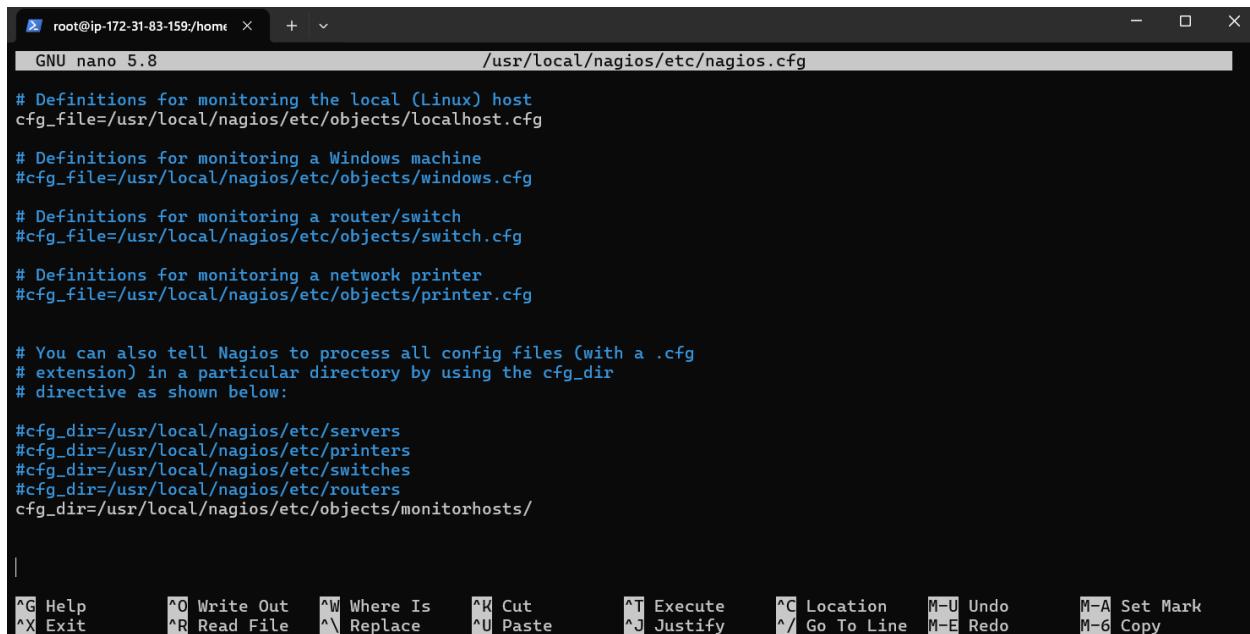
Change the **occurrences of hostname** further in the document from **localhost** to **linuxserver**

- 5) Now, we need to edit the nagios configuration file to add this directory.

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

Run this command and add the following line

cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/



```
# root@ip-172-31-83-159:/home | + - ×
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/
```

- 6) Now we verify the configuration files.

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

```
[root@ip-172-31-83-157 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...
  Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
  Checked 16 services.
  Checked 2 hosts.
  Checked 2 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 24 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 2 hosts
  Checked 0 service dependencies
  Checked 0 host dependencies
  Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors: 0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-83-157 ec2-user]# |
```

- 7) Once the files are verified, we need to restart the server.

service nagios restart

```
[root@ip-172-31-83-159 nagios-plugins-2.0.3]# service nagios restart
Restarting nagios (via systemctl): [ OK ]
[root@ip-172-31-83-159 nagios-plugins-2.0.3]# |
```

Step 3: Execute the following on Nagios Client machine (Ubuntu)

- 1) First, we check for any new updates, then we install gcc, nagios nrpe server and nagios plugins.

```
sudo apt update -y
```

```
sudo apt install gcc -y
```

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

```
ubuntu@ip-172-31-81-89:~$ 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://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
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 [380 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]
```

Running kernel seems to be up-to-date.

Restarting services...

Service restarts being deferred:

```
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart serial-getty@ttyS0.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
```

No containers need to be restarted.

User sessions running outdated binaries:

```
ubuntu @ session #4: sshd[1495,1569]
ubuntu @ user manager service: systemd[1500]
```

No VM guests are running outdated hypervisor (qemu) binaries on this host.

```
ubuntu@ip-172-31-81-89:~$ |
```

- 2) We need to add the public IP address of our host Nagios machine (Linux) to the nrpe configuration file.

sudo nano /etc/nagios/nrpe.cfg

Under allowed_hosts, add the nagios host ip address (public)

```
GNU nano 7.2
# You can either supply a username or a UID.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
nrpe_user=nagios

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

# ALLOWED HOST ADDRESSES
# This is an optional comma-delimited list of IP address or hostnames
# that are allowed to talk to the NRPE daemon. Network addresses with a bit mask
# (i.e. 192.168.1.0/24) are also supported. Hostname wildcards are not currently
# supported.
#
# Note: The daemon only does rudimentary checking of the client's IP
# address. I would highly recommend adding entries in your /etc/hosts.allow
# file to allow only the specified host to connect to the port
# you are running this daemon on.
#
# NOTE: This option is ignored if NRPE is running under either inetd or xinetd
allowed_hosts=127.0.0.1,54.210.81.106

# 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.

^G Help      ^O Write Out    ^W Where Is     ^K Cut          ^T Execute      ^C Location     M-U Undo
^X Exit      ^R Read File    ^A Replace      ^U Paste         ^J Justify      ^Y Go To Line   M-E Redo
                                         M-A Set Mark   M-G Copy       M-J To Bracket
                                         M-Q Where Was
```

Step 4: Check the Nagios Dashboard

- 1) Go to Nagios dashboard, click on hosts.

Here, we can see that the linuxserver is also added as a host.

Host	Status	Last Check	Duration	Status Information
linuxserver	UP	09-28-2024 04:42:16	0d 0h 2m 35s	PING OK - Packet loss = 0%, RTA = 1.15 ms
localhost	UP	09-28-2024 04:38:21	0d 0h 24m 0s	PING OK - Packet loss = 0%, RTA = 0.03 ms

- 2) Click on linuxserver. Here, we can check all the information about linuxserver host.

Host Information

Last Updated: Sat Sep 28 04:43:37 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as nagiosadmin

General

Home Documentation

Current Status

Tactical Overview Map Hosts Services Host Groups Summary Grid Service Groups Summary Grid Problems Services (Unhandled) Hosts (Unhandled) Network Outages Quick Search:

Reports

Availability Trends Alerts History Summary Histogram Notifications Event Log

System

Comments Downtime Process Info Performance Info Scheduling Queue Configuration

Host
linuxserver
(linuxserver)

Host State Information

Status: UP (for 0d 0h 3m 13s)
State Information: PING OK - Packet loss = 0%, RTA = 1.15 ms
Performance Data: rtt=1.151000ms;3000.000000;5000.000000;0.000000 p=0%;80;100.0
Current Attempt: 1/10 (HARD state)
Last Check Time: 09-28-2024 04:42:16
Check Type: ACTIVE
Check Latency / Duration: 0.000 / 0.033 seconds
Next Scheduled Active Check: 09-28-2024 04:47:16
Last State Change: 09-28-2024 04:40:24
Last Notification: N/A (notification 0)
Is This Host Flapping? NO (0.00% state change)
In Scheduled Downtime? NO
Last Update: 09-28-2024 04:43:33 (0d 0h 0m 4s ago)

Host Commands

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

Host Comments

Add a new comment Delete all comments

Entry Time Author Comment Comment ID Persistent Type Expires Actions

This host has no comments associated with it

- 3) Click on services. Here we can see all the services that are being monitored by linuxserver.

Current Network Status

Last Updated: Sat Sep 28 04:44:10 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.5.5 - www.nagios.org
Logged in as nagiosadmin

Host Status Totals

Up	Down	Unreachable	Pending
2	0	0	0

All Problems All Types

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
10	1	0	2	3

All Problems All Types

Service Status Details For All Hosts

Host	Service	Status	Last Check	Duration	Attempt	Status Information	
linuxserver	Current Load	OK	09-28-2024 04:41:01	0d 0h 3m 45s+	1/4	OK - load average: 0.00, 0.01, 0.00	
	Current Users	OK	09-28-2024 04:41:39	0d 0h 3m 85s+	1/4	USERS OK - 2 users currently logged in	
	HTTP	CRITICAL	09-28-2024 04:43:16	0d 0h 1m 54s+	2/4	connect to address 197.22.153.120 and port 80. Connection refused	
	PING	OK	09-26-2024 04:42:54	0d 0h 2m 46s+	1/4	PING OK - Packet loss = 0%, RTA = 1.11 ms	
	Root Partition	OK	09-26-2024 04:43:31	0d 0h 3m 46s+	1/4	DISK OK - free space: / 6116 MB (75.36% inode=98%)	
	SSH	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:44:09 UTC 2024	
	Swap Usage	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:44:46 UTC 2024	
	Total Processes	PENDING	N/A	0d 0h 3m 48s+	1/4	Service check scheduled for Sat Sep 28 04:45:24 UTC 2024	
	localhost	Current Load	OK	09-28-2024 04:43:36	0d 0h 2m 34s	1/4	OK - load average: 0.00, 0.02, 0.00
		Current Users	OK	09-28-2024 04:40:14	0d 0h 2m 56s	1/4	USERS OK - 2 users currently logged in
HTTP		WARNING	09-28-2024 04:43:51	0d 0h 2m 19s	4/4	HTTP WARNING: HTTP/1.1 403 Forbidden - 319 bytes in 0.000 second response time	
PING		OK	09-26-2024 04:41:29	0d 0h 22m 41s	1/4	PING OK - Packet loss = 0%, RTA = 0.03 ms	
Root Partition		OK	09-26-2024 04:42:06	0d 0h 22m 4s	1/4	DISK OK - free space: / 6116 MB (75.36% inode=50%)	
SSH		OK	09-26-2024 04:42:44	0d 0h 2m 26s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)	
Total Processes	OK	09-28-2024 04:43:59	0d 0h 26m 11s	1/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.		

Results 1 - 16 of 16 Matching Services

System

Comments Downtime Process Info Performance Info Scheduling Queue Configuration

In this case, we have monitored -

Servers: 1 linux server

Services: swap

NAME: Bhushan Malpani

DIV: D15C

ROLL NO: 33

Ports: 22, 80 (ssh, http)

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

Conclusion:

In this experiment, we learned to perform port service monitoring and server monitoring using Nagios. For this, we need the Linux instance used to host the Nagios dashboard and server. Also, we would need an Ubuntu instance which would be linked to a second host. We need to set up some configurations on the Linux instance and add the IP address of the Ubuntu instance. After that, we need to make the same initial setup on the ubuntu instance as the linux instance. Add the IP address of linux instance in allowed hosts. After restarting the NRPE server, we can see the 'linuxserver' host added.

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

Prerequisites:

- 1) AWS account (academy recommended)

Step 1: Set up AWS Lambda Function

- 1) Search for Lambda in the services tab. Click on it once found.

The screenshot shows the AWS Services search interface. A search bar at the top contains the text 'lambda'. Below it, a sidebar on the left lists categories: Services, Features, Documentation, Knowledge articles, Marketplace, Blog posts, Events, and Tutorials. The main area is titled 'Services' and displays several services: Lambda (highlighted with a blue border), CodeBuild, AWS Signer, Amazon Inspector, and Lambda Insights. Each service has a small icon and a brief description. On the right side of the main area, there's a 'Create application' button and a 'Regions' section showing 'N. Virginia'.

	Function name	Description	Package type	Runtime	Last modified
<input type="checkbox"/>	MainMonitoringFunction	-	Zip	Python 3.8	3 months ago
<input type="checkbox"/>	RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	3 months ago
<input type="checkbox"/>	RoleCreationFunction	Create SLR if absent	Zip	Python 3.8	3 months ago
<input type="checkbox"/>	RedshiftOverwatch	Deletes Redshift Cluster if the count is more than 2.	Zip	Python 3.8	3 months ago
<input type="checkbox"/>	ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	3 months ago

- 2) Click on create functions.
- 3) Give a name to your Lambda function. Select the runtime as Node.js 20.x (You can also use python). Select the architecture as x86_64. Set the default execution role as LabRole if you are doing this on your academy account. (Use an existing role → LabRole)

[Lambda](#) > [Functions](#) > [Create function](#)

Create function Info

Choose one of the following options to create your function.

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.

Basic information

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

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

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

Architecture Info
Choose the instruction set architecture you want for your function code.
 x86_64
 arm64

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

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).
 Create a new role with basic Lambda permissions
 Use an existing role
 Create a new role from AWS policy templates

Existing role

- 4) Once the function is created, click on the name of the function (myLambda27 in my case).

[Lambda](#) > [Functions](#)

Functions (6)					
<input type="text"/> Filter by tags and attributes or search by keyword Last fetched 3 minutes ago C Actions ▼ Create function C					
Function name	Description	Package type	Runtime	Last modified	
MainMonitoringFunction	-	Zip	Python 3.8	3 months ago	<input type="checkbox"/>
RedshiftEventSubscription	Create Redshift event subscription to SNS Topic.	Zip	Python 3.8	3 months ago	<input type="checkbox"/>
RoleCreationFunction	Create SLR if absent	Zip	Python 3.8	3 months ago	<input type="checkbox"/>
RedshiftOverwatch	Deletes Redshift Cluster if the count is more than 2.	Zip	Python 3.8	3 months ago	<input type="checkbox"/>
ModLabRole	updates LabRole to allow it to assume itself	Zip	Python 3.8	3 months ago	<input type="checkbox"/>
bhushan33	-	Zip	Node.js 20.x	32 seconds ago	<input type="checkbox"/>

5) This is the dashboard of our lambda function.

The screenshot shows the AWS Lambda Functions overview for a function named 'bhushan33'. At the top, there are tabs for 'Diagram' and 'Template', with 'Diagram' selected. Below the tabs is a box containing the function name 'bhushan33' and a 'Layers' section showing '(0)'. There are buttons for '+ Add trigger' and '+ Add destination'. On the right side, there are sections for 'Description' (empty), 'Last modified' (1 minute ago), 'Function ARN' (arn:aws:lambda:us-east-1:144602037631:function:bhushan33), and 'Function URL' (Info). Below the main box are tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions', with 'Code' selected. The 'Code source' tab is open, showing the code editor with the file 'index.mjs' containing the following code:

```
1 export const handler = async (event) => {
2     // TODO implement
3     const response = {
4         statusCode: 200,
5         body: JSON.stringify('Hello from Lambda!'),
6     };
7     return response;
8 };
```

6) This function has the following default code, which is used to print “Hello form Lambda!”.

The screenshot shows the AWS Lambda Code Source editor for the 'index.mjs' file. The interface includes a toolbar with 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (selected), and 'Deploy'. The code editor window displays the same code as the previous screenshot:1 export const handler = async (event) => {
2 // TODO implement
3 const response = {
4 statusCode: 200,
5 body: JSON.stringify('Hello from Lambda!'),
6 };
7 return response;
8 };At the bottom right of the editor, there are status indicators for '1:1 JavaScript' and 'Spaces: 2'.

Step 3: Set up configurations and test events

- 1) Just above the test code, you would find Configuration, click on it. Then click on Edit.

The screenshot shows the AWS Lambda Configuration page for a function named 'bhushan33'. The 'Configuration' tab is selected. On the left, there's a sidebar with 'General configuration', 'Triggers', 'Permissions', 'Destinations', and 'Function URL'. The main area displays 'General configuration' settings with an 'Edit' button. The settings include: Description (empty), Memory (128 MB), Ephemeral storage (512 MB), Timeout (0 min 3 sec), and SnapStart (None).

- 2) Here, change the Timeout to 1 sec. This is the time for which the function can be running before it is forcibly terminated.

The screenshot shows the 'Edit basic settings' page for the 'bhushan33' function. Under the 'Basic settings' tab, the 'Timeout' field is set to 1 second. Other settings visible include Memory (128 MB), Ephemeral storage (512 MB), and SnapStart (None). The 'Execution role' section shows 'Use an existing role' selected with 'LabRole' chosen from the dropdown.

- 3) We can see the executed changes.

The screenshot shows the AWS Lambda Configuration page for the 'bhushan33' function again. The 'Configuration' tab is selected. The 'General configuration' sidebar is visible. The main area shows the same 'General configuration' settings as before, but the 'Timeout' value has been updated to 1 second.

- 4) Switch back to the code tab. Click on the dropdown arrow near test. Then select configure test event.

The screenshot shows the AWS Lambda Code source interface. The top navigation bar includes File, Edit, Find, View, Go, Tools, Window, Test (which is highlighted in blue), and Deploy. Below the navigation bar is a search bar labeled "Go to Anything (Ctrl-P)". To the right of the search bar is a file list showing "index.mjs". The main area displays the code content:

```

1 export const handler = async (event) => {
2     // TODO implement
3     const response = {
4         statusCode: 200,
5         body: JSON.stringify('Hello from Lambda!'),
6     };
7     return response;
8 }
9

```

- 5) Here, create a new event, keep the other options default and save the event.

The screenshot shows the "Configure test event" dialog box. At the top, it says "A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result." Below that, it says "To invoke your function without saving an event, configure the JSON event, then choose Test." The "Test event action" section has two options: "Create new event" (which is selected) and "Edit saved event".

Event name: lambda33
 Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings:

- Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)
- Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional: hello-world

Event JSON:

```

1 * []
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 []

```

Buttons at the bottom include Cancel, Invoke (disabled), and Save.

- 6) Now, again click on the dropdown. This time, select the event you have created. Then, click on TEST.

The screenshot shows the AWS Lambda Code source interface. The 'Test' button in the top right is highlighted. A dropdown menu is open, showing the option 'lambda33'. The code editor window contains a file named 'index.mjs' with the following content:

```

1 export const ha
2 // TODO imple
3 const response = {
4   statusCode: 200,
5   body: JSON.stringify('Hello from Lambda!'),
6 };
7 return response;
8
9

```

- 7) We can see the expected output for the sample code.

The screenshot shows the AWS Lambda Code source interface with the 'Execution results' tab selected. It displays the following information for the test event 'lambda33':

- TestEvent Name: lambda33
- Response:


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


```
START RequestId: e71b9f42-0e1d-4570-af2d-c64591ee5ee2 Version: $LATEST
END RequestId: e71b9f42-0e1d-4570-af2d-c64591ee5ee2
REPORT RequestId: e71b9f42-0e1d-4570-af2d-c64591ee5ee2 Duration: 2.92 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 146.79 ms
Request ID
e71b9f42-0e1d-4570-af2d-c64591ee5ee2
```

- 8) For a test, declare a string and call it in line 6. After making the changes click on deploy.

The screenshot shows the AWS Lambda Code source interface with the 'Deploy' button highlighted in the top right. The code editor window contains a file named 'index.mjs' with the following content:

```

1 const test = "Bhushan"
2 export const handler = async (event) => {
3   // TODO implement
4   const response = {
5     statusCode: 200,
6     body: JSON.stringify(test),
7   };
8   return response;
9 }
10

```

9) Run the test. We can see that the string we declared has come in the output.

The screenshot shows the AWS Lambda Test console interface. At the top, a green header bar indicates "Successfully updated the function bhushan33." Below this, the main window has a toolbar with "File", "Edit", "Find", "View", "Go", "Tools", "Window", "Test" (which is selected), and "Deploy". A status message "Changes not deployed" is visible. The left sidebar shows the "Environment" tab and a file tree with "bhushan33 /" and "index.mjs". The main content area displays the "Execution results" tab, which includes:

- Test Event Name**: lambda33
- Response**:

```
[{"statusCode": 200, "body": "\\"Bhushan\\\""}]
```
- Function Logs**:

```
START RequestId: d03e066d-51c3-4398-9bff-053e5975be68 Version: $LATEST
END RequestId: d03e066d-51c3-4398-9bff-053e5975be68
REPORT RequestId: d03e066d-51c3-4398-9bff-053e5975be68 Duration: 23.27 ms Billed Duration: 24 ms Memory Size: 128 MB Max Memory Used: 62 MB Init Duration: 168.40 ms
```
- Request ID**: d03e066d-51c3-4398-9bff-053e5975be68

The status bar at the bottom right shows "Status: Succeeded" and "Max memory used: 62 MB".

Conclusion:

In this experiment, we effectively investigated the AWS Lambda service by developing and configuring Lambda functions using Python, Java, or Node.js. We discovered how to establish a Lambda function, tweak its settings (like modifying the timeout duration), and evaluate the function with personalized events. Throughout this experience, we noted how Lambda manages executions, including timeout handling and producing expected results according to modifications in the code.

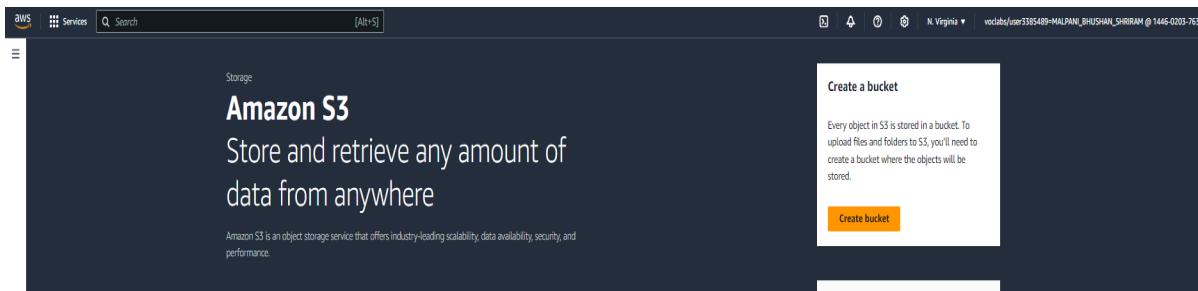
Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.

Prerequisites:

- 1) AWS account (academy preferable)
- 2) Lambda function (created in the previous experiment).

Step 1: Create a s3 bucket.

- 1) Search for S3 bucket in the services search. Then click on create bucket.



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

The screenshot shows the 'Create bucket' wizard in the AWS Management Console. The current step is 'General configuration'. The 'Bucket name' field is filled with 's3lambda33'. The 'AWS Region' dropdown is set to 'US East (N. Virginia) us-east-1'. The 'Bucket type' section contains two options: 'General purpose' (selected) and 'Directory'. The 'General purpose' option is described as recommended for most use cases and access patterns, noting that general purpose buckets are the original S3 bucket type and allow a mix of storage classes. The 'Directory' option is described as recommended for low-latency use cases, using the S3 Express One Zone storage class for faster processing of data within a single Availability Zone. Below the configuration, there is a note about unique bucket names and a link to 'rules for bucket naming'. There is also a section for 'Copy settings from existing bucket - optional' with a 'Choose bucket' button and a note about copied settings.

3) Uncheck block all public access.

Block Public Access settings for this bucket

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

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

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

⚠ Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

I acknowledge that the current settings might result in this bucket and the objects within becoming public.

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

Successfully created bucket "s3lambda33"

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

Amazon S3 > Buckets

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

General purpose buckets (1) [Info](#) All AWS Regions

Buckets are containers for data stored in S3.

Find buckets by name

Name: s3lambda33 | AWS Region: US East (N. Virginia) us-east-1 | IAM Access Analyzer: View analyzer for us-east-1 | Creation date: October 6, 2024, 22:53:44 (UTC+05:30)

[Copy ARN](#) | [Empty](#) | [Delete](#) | [Create bucket](#)

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

s3lambda33 [Info](#)

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (0) [Info](#)

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

[Upload](#)

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

- 6) The image has been uploaded to the bucket.

The screenshot shows the AWS S3 'Upload: status' page. At the top, a green bar indicates 'Upload succeeded'. Below it, a summary table shows the destination as 's3://s3lambda33', with one file successfully uploaded ('Succeeded: 1 file, 22.2 KB (100.00%)') and zero files failed ('Failed: 0 files, 0 B (0%)'). The 'Files and folders' tab is selected, displaying a table with one item: 'Ganpati.jpg' (image/jpeg, 22.2 KB, Succeeded). A 'Find by name' search bar is present above the table.

Step 2: Configure Lambda function

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

The screenshot shows the AWS Lambda 'Function overview' page for the function 'bhushan33'. It includes tabs for 'Diagram' (selected), 'Template', 'Throttle', 'Copy ARN', 'Actions', 'Export to Application Composer', and 'Download'. The 'Diagram' section shows a single function icon labeled 'bhushan33' and a 'Layers' section with '(0)'. A '+ Add trigger' button is visible. The right side of the page displays the function's description, last modified time ('23 minutes ago'), Function ARN ('arn:aws:lambda:us-east-1:144602037631:function:bhushan33'), and Function URL ('Info').

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

The screenshot shows the 'Add trigger' configuration page under 'Trigger configuration'. It features a search bar with the text 'S3' and the results 'aws asynchronous storage' listed below it. The 'Trigger configuration' section also includes a link to 'Info'.

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

The screenshot shows the 'Trigger configuration' section of the AWS Lambda console. It is set up for an S3 trigger. The 'Bucket' field contains 's3/s3lambda33' and the 'Bucket region' is 'us-east-1'. Under 'Event types', 'All object create events' is selected. There are optional fields for 'Prefix' (e.g., 'images/') and 'Suffix' (e.g., '.jpg'). A note about recursive invocation is present, with a checked checkbox acknowledging the risk. A note about Lambda permissions is also shown. At the bottom right are 'Cancel' and 'Add' buttons.

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

```
export const handler = async (event) => {
  if (!event.Records || event.Records.length === 0) {
    console.error("No records found in the event.");
    return { statusCode: 400, body: JSON.stringify('No
      records
      found in the event')
  };
}
```

```

    // Extract bucket name and object key from the event const record = event.Records[0];
const bucketName = record.s3.bucket.name; const objectKey =
decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle
encoded keys

    console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
    console.log(`Event Source: ${record.eventSource}`);
    console.log(`Event Source: ${record.eventSource}`); console.log(`Event Source: ${record.eventSource}`); return
}

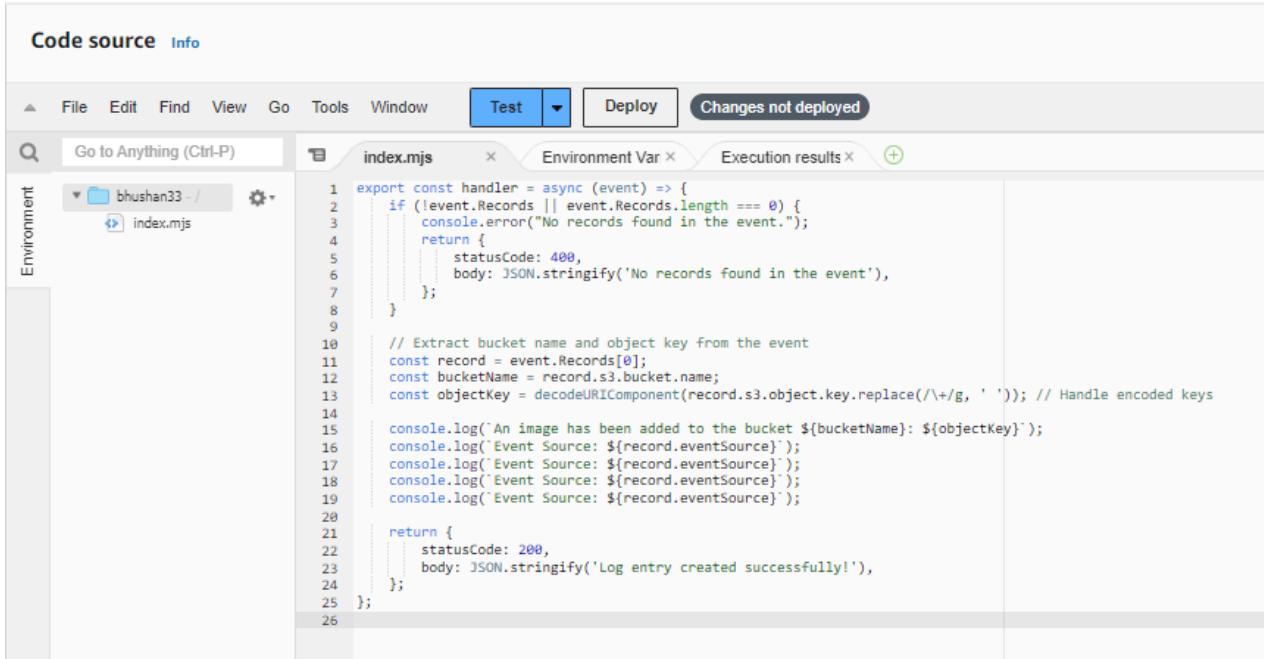
    statusCode: 200, body: JSON.stringify('Log entry
created successfully!')

};

}

```

This code checks for records in the event, extracts the bucket name and object key, logs the details, and returns a success message if an image is added to the bucket.



The screenshot shows the AWS Lambda function editor interface. The top navigation bar includes 'File', 'Edit', 'Find', 'View', 'Go', 'Tools', 'Window', 'Test' (which is currently selected), 'Deploy', and 'Changes not deployed'. The left sidebar shows the 'Environment' section with a dropdown menu and a file tree containing 'bhushan33 - /' and 'index.mjs'. The main workspace displays the following code:

```

1  export const handler = async (event) => {
2      if (!event.Records || event.Records.length === 0) {
3          console.error("No records found in the event.");
4          return {
5              statusCode: 400,
6              body: JSON.stringify('No records found in the event'),
7          }
8      }
9      // Extract bucket name and object key from the event
10     const record = event.Records[0];
11     const bucketName = record.s3.bucket.name;
12     const objectKey = decodeURIComponent(record.s3.object.key.replace(/\+/g, ' ')); // Handle encoded keys
13
14     console.log(`An image has been added to the bucket ${bucketName}: ${objectKey}`);
15     console.log(`Event Source: ${record.eventSource}`);
16     console.log(`Event Source: ${record.eventSource}`);
17     console.log(`Event Source: ${record.eventSource}`);
18     console.log(`Event Source: ${record.eventSource}`);
19
20     return {
21         statusCode: 200,
22         body: JSON.stringify('Log entry created successfully!'),
23     };
24 };
25
26

```

Now, click on the dropdown near test, then click on configure test event.

6) Here, select edit saved event. Select the event that you had created before. Under Event JSON, paste the following code.

```
{  
  "Records": [  
    {  
      "eventVersion": "2.0",  
      "eventSource": "aws:s3",  
      "awsRegion": "us-east-1",  
      "eventTime": "1970-01-01T00:00:00.000Z",  
      "eventName": "ObjectCreated:Put",  
      "userIdentity": {  
        "principalId": "EXAMPLE"  
      },  
      "requestParameters": {  
        "sourceIPAddress": "127.0.0.1"  
      },  
      "responseElements": {  
        "x-amz-request-id": "EXAMPLE123456789", "x-amz-id-  
        2":  
          "EXAMPLE123/5678abcdefghijklmabaisawesome/mnopqrstuvwxyzABCDEFGHI"  
      },  
      "s3": {  
        "s3SchemaVersion": "1.0",  
        "configurationId": "testConfigRule",  
  
      "bucket": {  
        "name": "example-bucket",  
        "ownerIdentity": {  
          "principalId": "EXAMPLE"  
        },  
        "arn": "arn:aws:s3:::example-bucket"  
      },  
      "object": {  
    }
```

```

    "key": "test%2Fkey",
    "size": 1024,
    "eTag": "0123456789abcdef0123456789abcdef",
    "sequencer": "0A1B2C3D4E5F678901"
}
}
]
}
}

```

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

Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda function. Use it to see the function's invocation result.

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save changes.

Test event action

Create new event Edit saved event

Event name

lambda33 ▼ C Delete

Event JSON

Format JSON

```

1  {
2    "Records": [
3      {
4        "eventVersion": "2.0",
5        "eventSource": "aws:s3",
6        "awsRegion": "us-east-1",
7        "eventTime": "1970-01-01T00:00:00.000Z",
8        "eventName": "ObjectCreated:Put",
9        "userIdentity": {
10          "principalId": "EXAMPLE"
11        },
12        "requestParameters": {
13          "sourceIPAddress": "127.0.0.1"
14        },
15        "responseElements": {
16          "x-amz-request-id": "EXAMPLE123456789",
17          "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
18        },
19        "s3": {
20          "s3SchemaVersion": "1.0",
21          "configurationId": "testConfigRule",
22          "bucket": {
23            "name": "example-bucket",
24            "ownerIdentity": {
25              "principalId": "EXAMPLE"
26            },
27            "arn": "arn:aws:s3:::example-bucket"
28          },
29          "object": {
30            "key": "test%2Fkey",
31          }
32        }
33      }
34    ]
35  }

```

Cancel Invoke Save

Save the changes. Then deploy the code changes by clicking on deploy.

- 7) After deploying, click on Test. The console output shows that 'an image has been added to the bucket'

The JSON response shows that the log entry was created successfully.

The screenshot shows the AWS Lambda Test interface. In the top navigation bar, 'Test' is selected. The left sidebar shows the environment configuration for 'bhushan33'. The main area displays the 'Execution results' tab, which includes the following details:

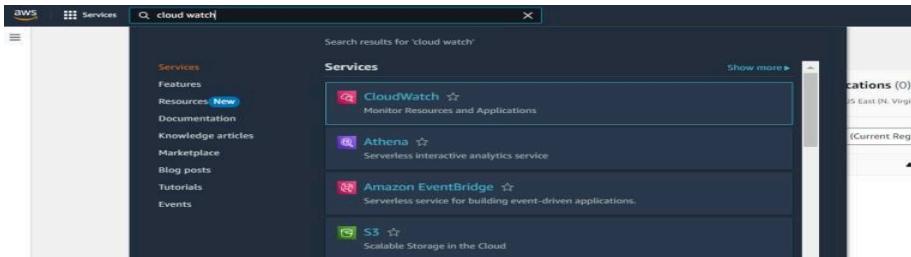
- Test Event Name:** Lambda33
- Response:**

```
{
  "statusCode": 200,
  "body": "\"Log entry created successfully!\""
}
```
- Function Logs:**

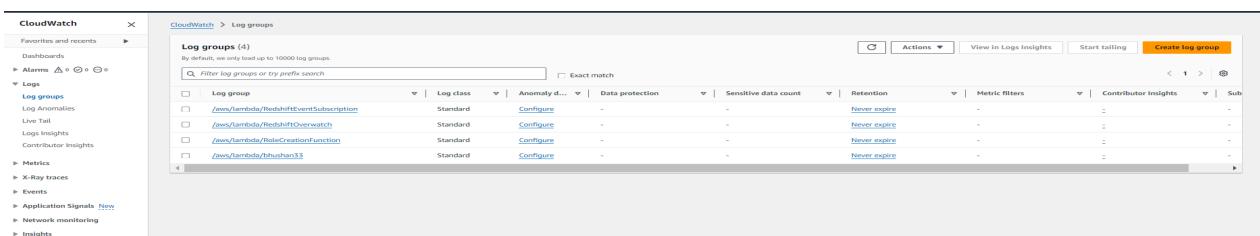
```
START RequestId: cab39e48-6b3c-4ab8-a6a3-60096951c968 Version: $LATEST
2024-10-06T17:36:45.409Z cab39e48-6b3c-4ab8-a6a3-60096951c968 INFO An image has been added to the bucket example-bucket: test/key
2024-10-06T17:36:45.430Z cab39e48-6b3c-4ab8-a6a3-60096951c968 INFO Event Source: aws:s3
2024-10-06T17:36:45.431Z cab39e48-6b3c-4ab8-a6a3-60096951c968 INFO Event Source: aws:s3
2024-10-06T17:36:45.448Z cab39e48-6b3c-4ab8-a6a3-60096951c968 INFO Event Source: aws:s3
2024-10-06T17:36:45.448Z cab39e48-6b3c-4ab8-a6a3-60096951c968 INFO Event Source: aws:s3
END RequestId: cab39e48-6b3c-4ab8-a6a3-60096951c968
REPORT RequestId: cab39e48-6b3c-4ab8-a6a3-60096951c968 Duration: 43.18 ms Billed Duration: 44 ms Memory Size: 128 MB Max Memory Used: 64 MB Init Duration: 141.44 ms
```
- Request ID:** cab39e48-6b3c-4ab8-a6a3-60096951c968

Step 3: Check the logs

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



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



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

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

Conclusion:

In this experiment, we developed and deployed a Lambda function designed to respond to file uploads in an S3 bucket. The function was triggered automatically whenever a new object was added to the bucket, illustrating how AWS services can efficiently automate workflows. The Lambda function extracted and logged key details from the event, such as the bucket's name and the object's key. We tested this by uploading a sample file, and upon reviewing the logs in CloudWatch, we confirmed that the function executed successfully, capturing the upload event. This experiment demonstrated the powerful synergy between AWS Lambda and S3, enabling seamless, event-driven automation.