

CASE STUDY: TOPIC NUMBER 6

Topic Name: Automated Deployment with Monitoring

- **Concepts Used:** Jenkins, EC2, Nagios.
- **Problem Statement:** "Set up a Jenkins CI/CD pipeline to deploy a simple web application on an EC2 instance. Configure Nagios to monitor the deployed application's availability."
- **Tasks:**
 - Create a Jenkins pipeline that builds and deploys a sample web app to an EC2 instance.
 - Install and configure Nagios to monitor the HTTP status of the deployed application.
 - Verify the pipeline by triggering a build and checking the monitoring status in Nagios.

1. Introduction

Case Study Overview:

This case study focuses on creating an automated deployment and monitoring system using Jenkins, Amazon EC2, and Nagios. The primary goal is to set up a Jenkins CI/CD pipeline that automates the deployment of a simple web application on an EC2 instance. Additionally, Nagios is used to monitor the deployed application's availability, ensuring continuous functionality. This setup provides a robust solution for teams aiming to automate deployments while maintaining real-time monitoring.

Key Feature and Application:

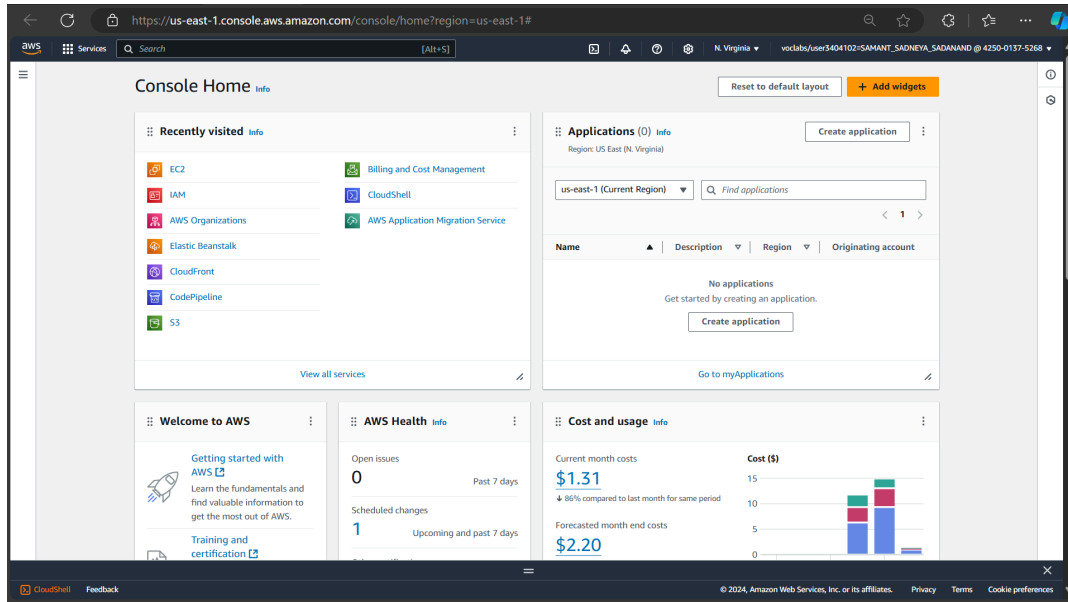
The unique aspect of this case study is the integration of Jenkins with a CI/CD pipeline, which allows for automated builds, testing, and deployment. This automation reduces manual intervention, leading to a more efficient deployment process. By combining it with Nagios for monitoring, the solution ensures the deployed application remains accessible and responsive. This setup is ideal for dynamic environments where frequent updates and deployments are necessary, such as agile development teams or small-scale applications.

2. Step-by-Step Explanation:

1. Installing Jenkins on AWS EC2

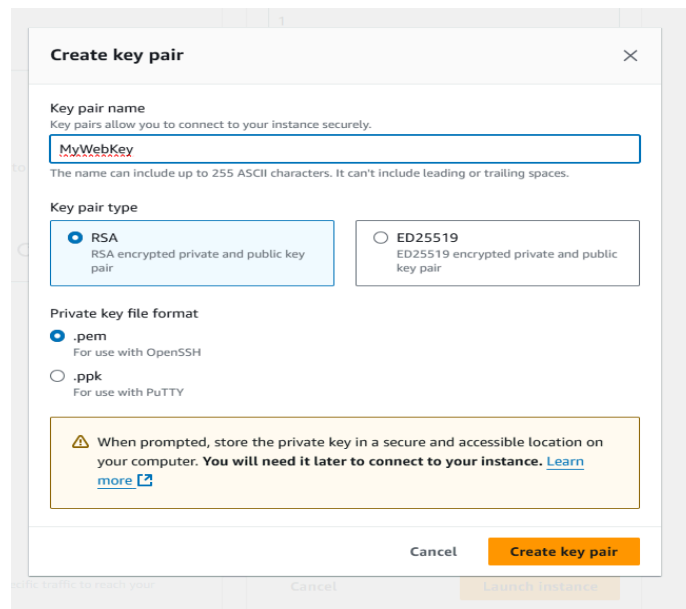
Prerequisites

1. **AWS Account:** Sign up for an AWS account if you don't have one.



Step 1: Create a Key Pair

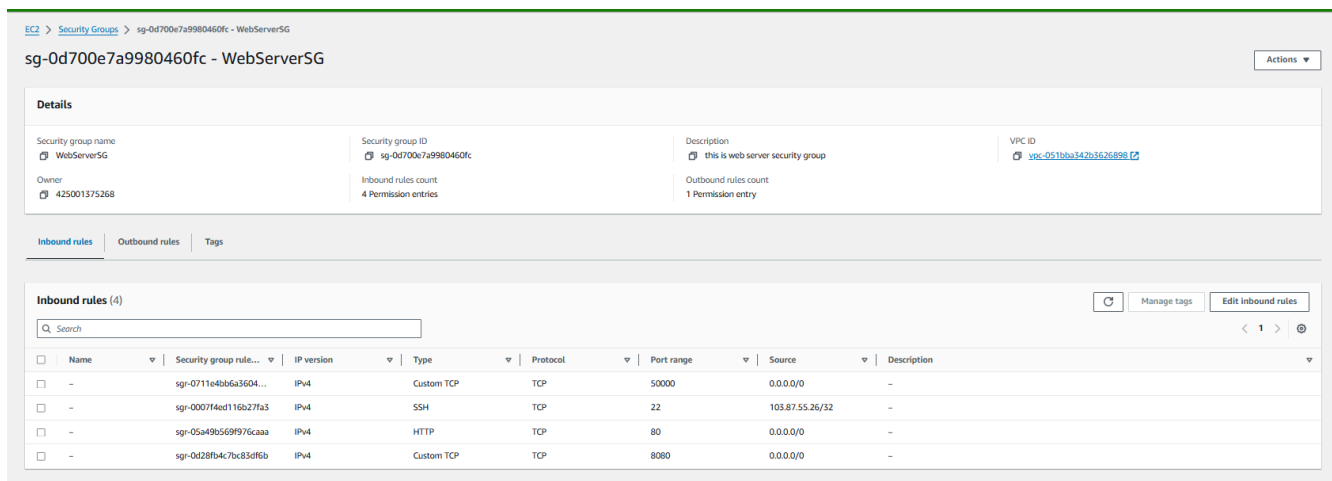
1. **Open the EC2 Console:** Go to the [Amazon EC2 console](#).
2. **Key Pairs:** In the navigation pane, under **NETWORK & SECURITY**, select **Key Pairs**.
3. **Create Key Pair:**
 - Click **Create key pair**. Here, I have created “MyWebKey.pem”.
 - Name the key pair and select **pem** for **File format**.



- Click **Create key pair** to download the private key file. **Store it securely.**
4. **Set Permissions:** `chmod 400 /path/to/your-key-pair.pem`

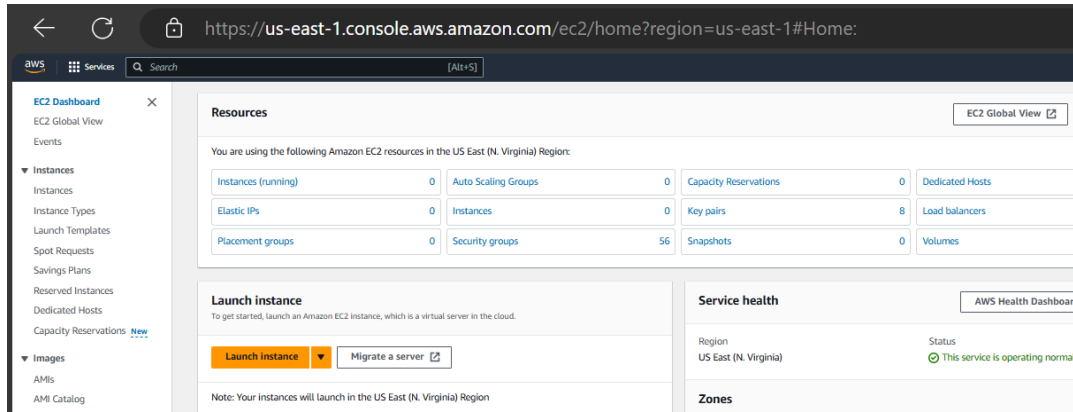
Step 2: Create a Security Group

1. **Security Groups:** In the EC2 console, select **Security Groups** and then **Create Security Group**.
2. **Configure Rules:**
 - **Name:** Enter a name. Here I have given name “WebserverSG”
 - **Description:** Provide a description.(optional)
 - **Inbound Rules:**
 - **SSH:** Allow inbound SSH from your IP.
 - Type: SSH
 - Source: Your public IP with /32 (e.g., 103.87.55.26/32).
 - **HTTP:** Allow inbound HTTP from anywhere.
 - Type: HTTP
 - Source: 0.0.0.0/0
 - **Custom TCP Rule:** Allow Jenkins (8080) access.
 - Type: Custom TCP Rule
 - Port Range: 8080
 - Source: 0.0.0.0/0 (or restrict to your IP).
 - **Custom TCP Rule:** Allow Jenkins TCP (50000) for agent access. you will see its use later .
 - Type: Custom TCP Rule
 - Port Range: 50000
 - Source: 0.0.0.0/0 (or restrict to your IP).
3. Click on Create the Security Group.



Step 3: Launch an EC2 Instance

1. **Launch Instance:** In the EC2 dashboard, click **Launch Instance**.

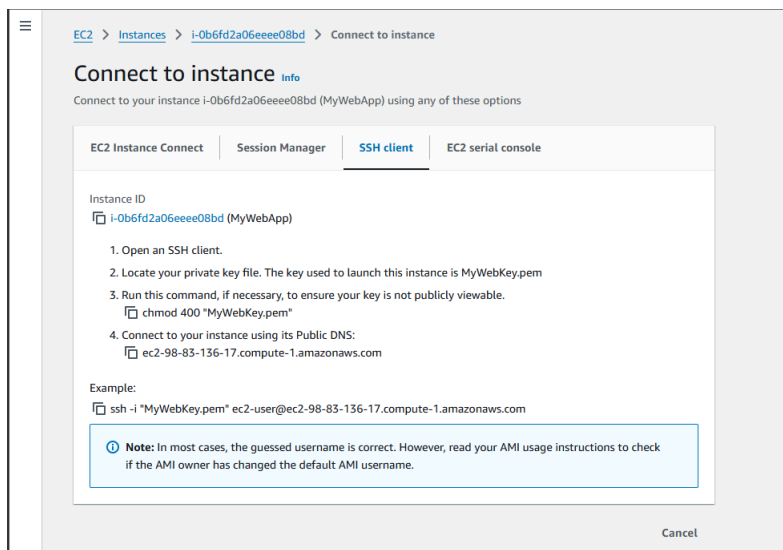


2. **Choose AMI:** Select the Amazon Linux 2023 AMI (Free tier eligible).
3. **Instance Type:** Choose t2.micro (Free tier eligible).
4. **Configure Instance:**
 - **Key Pair:** Select the key pair you created.(MyWebKey)
 - **Security Group:** Select the security group (JenkinsSG) you just created.
5. **Launch:** Review and click **Launch Instance**.
6. **Check Status:** Go to **Instances** in the navigation pane to monitor the instance status. Wait until it shows running. Thus Instance created successfully.

Instance ID = i-0b6fd2a06eeee08bd Clear filters										
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4	
<input type="checkbox"/>	MyWebApp	i-0b6fd2a06eeee08bd	Running	t2.micro	2/2 checks pass	View alarms	us-east-1c	ec2-98-83-136-17.com...	98.83.136.17	

Step 4: Connect to Your EC2 Instance

1. **Get Public DNS:** Select the instance and find the **Public DNS (IPv4)**. Now open your command prompt.



```
[ec2-user@ip-172-31-40-207 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2024-10-19 14:28:06-- https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.38.133, 2a04:4e42:79::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.38.133|:443
... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/jenkins.repo 100%[=====>]      85  --.-KB/s    in 0s

2024-10-19 14:28:06 (9.62 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]
```

3. Import Jenkins Key:

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

```
[ec2-user@ip-172-31-40-207 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

Also check any upgradation and upgrade.

```
[ec2-user@ip-172-31-40-207 ~]$ sudo yum upgrade
Jenkins-stable          331 kB/s | 29 kB      00:00
Dependencies resolved.
Nothing to do.
Complete!
```

4. Install Java (Jenkins requires Java): sudo dnf install java-17-amazon-corretto -y

```
[ec2-user@ip-172-31-40-207 ~]$ sudo dnf install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:17 ago on Sat Oct 19 14:28:31 2024.
Dependencies resolved.
=====
Package           Arch   Version                               Repository      Size
=====
Installing:
java-17-amazon-corretto
x86_64 1:17.0.12+7-1.amzn2023.1 amazonlinux 187 k
Installing dependencies:
alsa-lib           x86_64 1.2.7.2-1.amzn2023.0.2 amazonlinux 504 k
dejavu-sans-fonts  noarch 2.37-16.amzn2023.0.2 amazonlinux 1.3 M
dejavu-sans-mono-fonts
noarch 2.37-16.amzn2023.0.2 amazonlinux 467 k
dejavu-serif-fonts
noarch 2.37-16.amzn2023.0.2 amazonlinux 1.0 M
giflib             x86_64 5.2.1-9.amzn2023.0.1 amazonlinux 49 k
java-17-amazon-corretto-headless
x86_64 1:17.0.12+7-1.amzn2023.1 amazonlinux 91 M
javapackages-filesystem
noarch 6.0.0-7.amzn2023.0.6 amazonlinux 12 k
libXi              x86_64 1.8.2-1.amzn2023.0.1 amazonlinux 42 k
libXinerama        x86_64 1.1.5-6.amzn2023.0.1 amazonlinux 16 k
libXrandr          x86_64 1.5.4-3.amzn2023.0.1 amazonlinux 29 k
libXtst            x86_64 1.2.5-1.amzn2023.0.1 amazonlinux 22 k
=====
```

```

Installed:
  alsa-lib-1.2.7.2-1.amzn2023.0.2.x86_64
  dejavu-sans-fonts-2.37-16.amzn2023.0.2.noarch
  dejavu-sans-mono-fonts-2.37-16.amzn2023.0.2.noarch
  dejavu-serif-fonts-2.37-16.amzn2023.0.2.noarch
  giflib-5.2.1-9.amzn2023.0.1.x86_64
  java-17-amazon-corretto-1:17.0.12+7-1.amzn2023.1.x86_64
  java-17-amazon-corretto-headless-1:17.0.12+7-1.amzn2023.1.x86_64
  javapackages-filesystem-6.0.0-7.amzn2023.0.6.noarch
  libXi-1.8.2-1.amzn2023.0.1.x86_64
  libXinerama-1.1.5-6.amzn2023.0.1.x86_64
  libXrandr-1.5.4-3.amzn2023.0.1.x86_64
  libXtst-1.2.5-1.amzn2023.0.1.x86_64

Complete!

```

5. Install Jenkins:

```
sudo yum install jenkins -y
```

```

Complete!
[ec2-user@ip-172-31-40-207 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:00:39 ago on Sat Oct 19 14:28:31 2024.
Dependencies resolved.
=====
Package      Arch      Version      Repository      Size
=====
Installing:
jenkins      noarch    2.462.3-1.1  jenkins         89 M
=====
Transaction Summary
=====
Install 1 Package

Total download size: 89 M
Installed size: 89 M
Downloading Packages:
jenkins-2.462.3-1.1.noarch.rpm 27 MB/s | 89 MB  00:03
-----
Total                27 MB/s | 89 MB  00:03
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :
  Running scriptlet: jenkins-2.462.3-1.1.noarch 1/1
  Installing     : jenkins-2.462.3-1.1.noarch 1/1
  Running scriptlet: jenkins-2.462.3-1.1.noarch 1/1
  Verifying      : jenkins-2.462.3-1.1.noarch 1/1

Installed:
jenkins-2.462.3-1.1.noarch

Complete!

```

6. Enable and Start Jenkins:

```
sudo systemctl enable jenkins
```

```

[ec2-user@ip-172-31-40-207 ~]$ sudo systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.

```

```
sudo systemctl start jenkins
```

7. Check Jenkins Status:

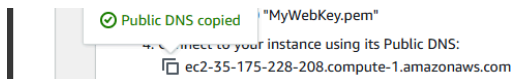
```
sudo systemctl status jenkins
```

```
[ec2-user@ip-172-31-40-207 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-40-207 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled)
   Active: active (running) since Sat 2024-10-19 14:30:01 UTC; 1min 1s ago
     Main PID: 74158 (java)
       Tasks: 46 (Limit: 1112)
      Memory: 337.8M
         CPU: 15.479s
    CGroup: /system.slice/jenkins.service
            └─74158 /usr/bin/java -Djava.awt.headless=true -ja

Oct 19 14:29:54 ip-172-31-40-207.ec2.internal jenkins[74158]: T>
Oct 19 14:29:54 ip-172-31-40-207.ec2.internal jenkins[74158]: *>
Oct 19 14:29:54 ip-172-31-40-207.ec2.internal jenkins[74158]: *>
Oct 19 14:29:54 ip-172-31-40-207.ec2.internal jenkins[74158]: *>
Oct 19 14:30:01 ip-172-31-40-207.ec2.internal jenkins[74158]: 2>
Oct 19 14:30:01 ip-172-31-40-207.ec2.internal jenkins[74158]: 2>
Oct 19 14:30:01 ip-172-31-40-207.ec2.internal systemd[1]: Start>
Oct 19 14:30:01 ip-172-31-40-207.ec2.internal jenkins[74158]: 2>
Oct 19 14:30:01 ip-172-31-40-207.ec2.internal jenkins[74158]: 2>
Oct 19 14:30:06 ip-172-31-40-207.ec2.internal jenkins[74158]: 2>
lines 1-20/20 (END)
```

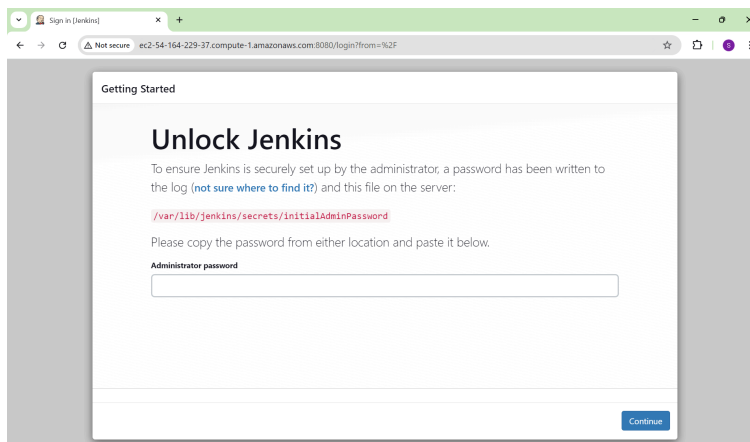
Step 6: Configure Jenkins

1. **Access Jenkins:** Open your web browser and go to browser and type `http://<your_instance_public_dns>:8080`



← ↻ 🌐 <http://ec2-54-164-229-37.compute-1.amazonaws.com:8080>

2. **Unlock Jenkins:**

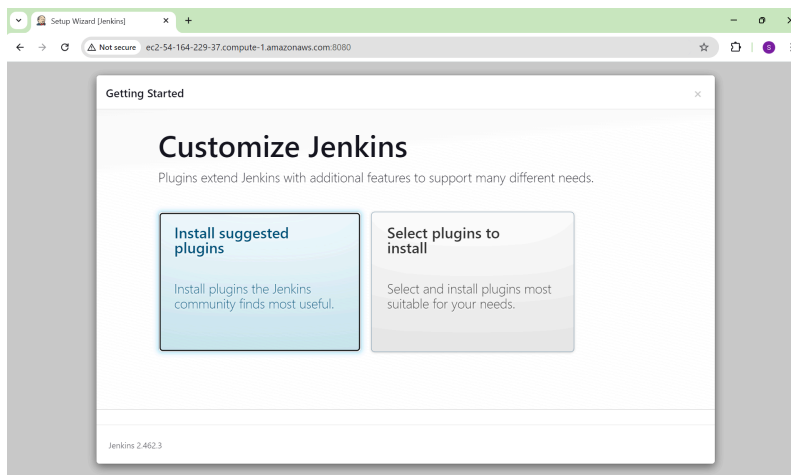


Find the initial admin password:

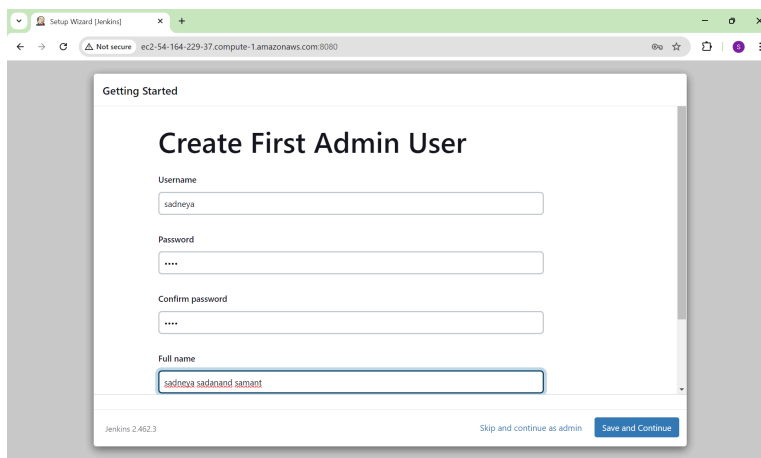
```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

```
[ec2-user@ip-172-31-45-86 ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
da0e63b6884a46359abe8e3a364f7c22
```

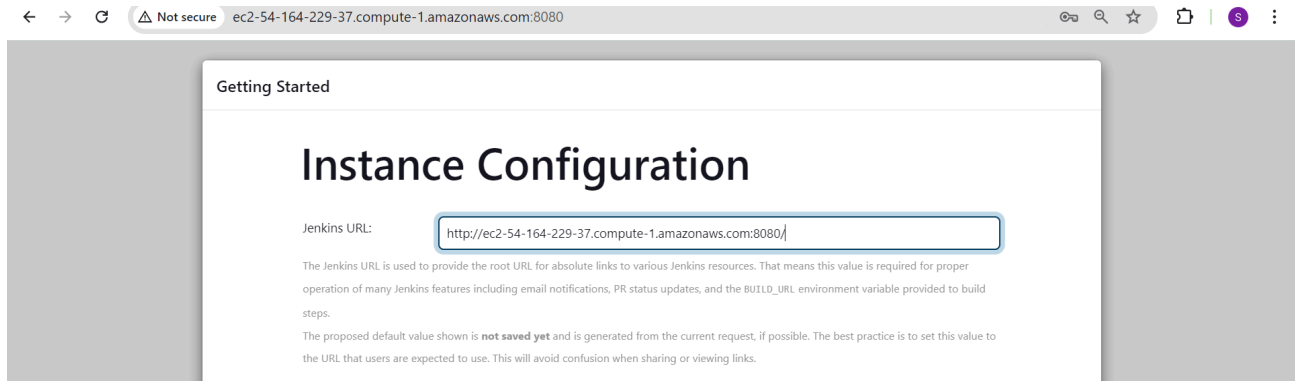
- Copy the password and paste it into the web interface to unlock Jenkins.
3. **Customize Jenkins:** Follow the setup wizard to install suggested plugins and create an admin user.



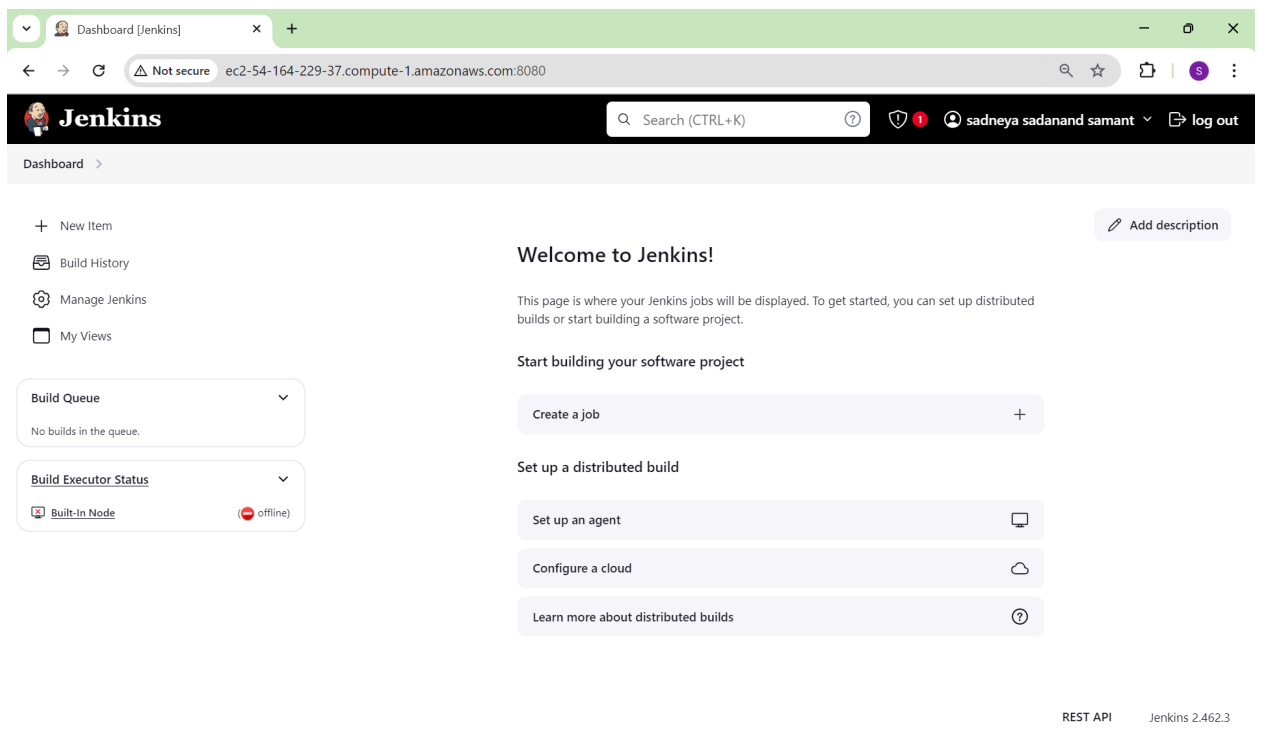
4. set username,password and full name.



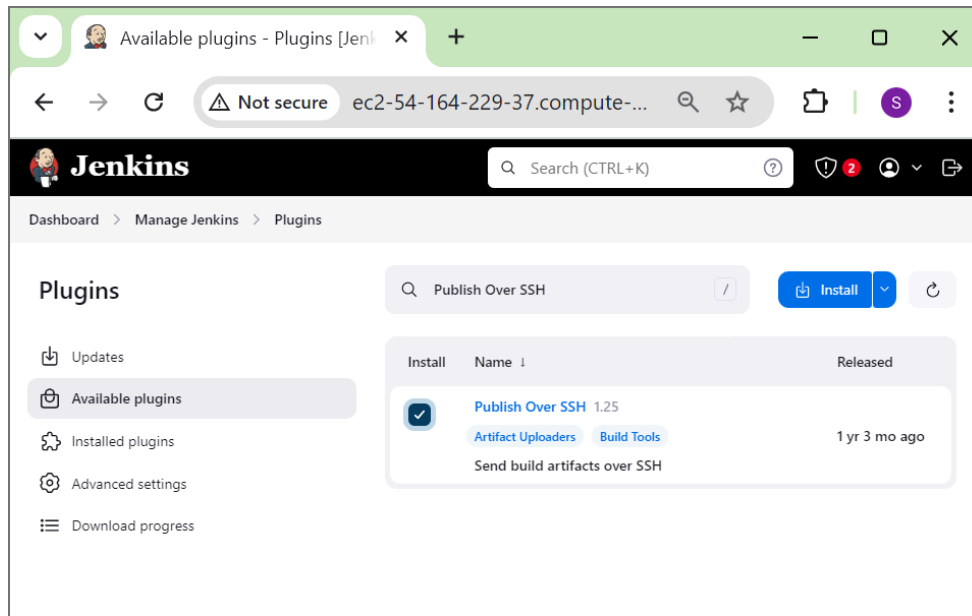
5. **Finish Setup:** Once configured, Jenkins will be accessible at `http://<your_instance_public_dns>:8080`.



Thus click on finish jenkins will open.



6. Install publish over ssh



7. Go to Manage Jenkins->credentials create a global credential for ssh.

1.Add kind: SSH**2. Scope:Global****3.ID:** assign Id here i have given myEC2SSHKey**4.Username:**ec2-user for amazon linux**5.private key:** give the key which we created on creating an instance.

New credentials

Kind
SSH Username with private key

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

ID ?
myEC2SSHKey

Description ?
EC2 SSH key

Username
ec2-user

☐ Treat username as secret ?

Private Key
☒ Enter directly

Key
Enter New Secret Below
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAvg5j68xygUR/RpS2Na0MOOZ3GCS/1qUP*1yLxd+Oup+YDxD
/CuJy1XG211Kd8X/Yz51PKWkw3MKS11y8BxaCG+L+nB7G0B31ISnS9V735Vg

Passphrase

Create

8. 1. Go to manage jenkins then scroll down there you will get publish over ssh section there copy paster your key and

Publish over SSH

Jenkins SSH Key ?

Passphrase ?

Path to key ?

Key ?

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA44rUO3WKOQH/7Z0ye3mtuZ0ESA5sleCmQwaEP7DPWo5lCwd9pdOM73rEVWRKD
ZSCBNQK8gA0j4I9d6j/70HQZYGam3j9S09fyG110091LK44a2dgZSHRk3tV
aPhrS5wijnCp2apMx20MTMaR0KujLrB0ZEt1hWb3k+YQgj+SQMkPj45H9Qf
kHcllqdgNaYikb8CreFAd09m09gUgUXSMSh699skXNNsDdlnhp
-----END RSA PRIVATE KEY-----
```

☐ Disable exec ?

2. then click on ADD you will get SSH Servers give name then give your public DNS as hostname and then give username and remote directory where you want to store the project.
3. Then click on Test configuration.then it will give either success or failure.

SSH Servers

SSH Server

Name ?

MyWebApp

Hostname ?

ec2-54-164-229-37.compute-1.amazonaws.com

Username ?

ec2-user

Remote Directory ?

/home/ec2-user/myapp/

☐ Avoid sending files that have not changed ?

Advanced

Success

Test Configuration

9.

1. As my built-in-node have limited space thus i need to use master-slave architecture. here I have created a node named "laptop_node"

New node

Node name

laptop_node

Type

☒ Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

Create

Name ⓘ
laptop_node

Description ⓘ
this is second node.

Plain text [Preview](#)

Number of executors ⓘ
2

Remote root directory ⓘ
C:\Users\Sadneya\OneDrive\Desktop\jenkins

Labels ⓘ
1

Usage ⓘ
Use this node as much as possible

Launch method ⓘ
Launch agent by connecting it to the controller

Availability ⓘ
Keep this agent online as much as possible

Node Properties

☐ Disable deferred wipeout on this node ⓘ
☐ Disk Space Monitoring Thresholds
☐ Environment variables
☐ Tool Locations

[Save](#)

REST API

2. Go to manage jenkins->security and make agent TCP port fixed at 50000.that we before added in security group.

Agents

TCP port for inbound agents ⓘ

☒ Fixed

50000

☐ Random

☐ Disable

3. Now click on node you created. Here you will get the commands then copy paste this commands on your command prompt.

Run from agent command line: (Windows) [Copy](#)

```
curl.exe -s0 http://ec2-54-164-229-37.compute-1.amazonaws.com:8080/jnlpJars/agent.jar  
java -jar agent.jar -url http://ec2-54-164-229-37.compute-1.amazonaws.com:8080/ -secret 900095ef2c123fd89e791bf7511528c2b1701dec87739d6a59cd3bd7b6319cad -name "laptop_node" -workDir "C:\Users\Sadneya\OneDrive\Desktop\jenkins"
```

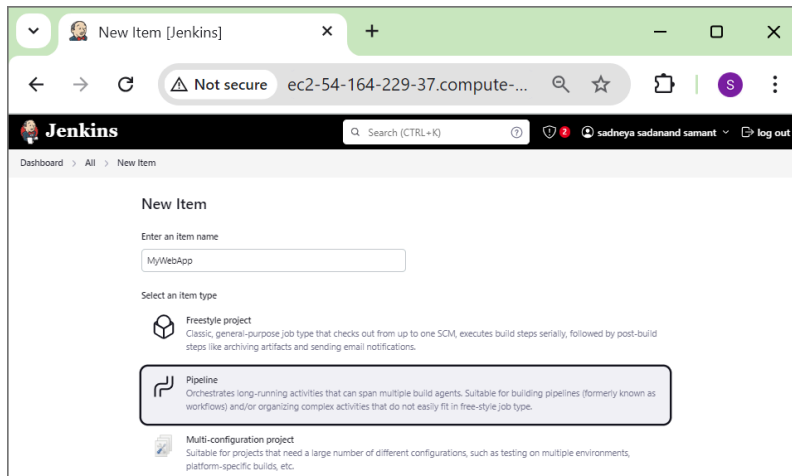
```
C:\Users\Sadneya\OneDrive\Desktop\jenkins>curl.exe -s0 http://ec2-54-164-229-37.compute-1.amazonaws.com:8080/jnlpJars/agent.jar

C:\Users\Sadneya\OneDrive\Desktop\jenkins>java -jar agent.jar -url http://ec2-54-164-229-37.compute-1.amazonaws.com:8080/ -secret 900095ef2c123fd89e791bf751528c2b1701dec87739d6a59cd3bd7b6319cad -name "Laptop_node" -workDir "C:\Users\Sadneya\OneDrive\Desktop\jenkins"
Oct 19, 2024 10:42:51 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using C:\Users\Sadneya\OneDrive\Desktop\jenkins\remoting as a remoting work directory
Oct 19, 2024 10:42:51 AM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to C:\Users\Sadneya\OneDrive\Desktop\jenkins\remoting
Oct 19, 2024 10:42:51 AM hudson.remoting.Launcher createEngine
INFO: Setting up agent: Laptop_node
Oct 19, 2024 10:42:51 AM hudson.remoting.Engine startEngine
INFO: Using Remoting version: 3248.3250.v3277a_8e88c9b_
Oct 19, 2024 10:42:51 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using C:\Users\Sadneya\OneDrive\Desktop\jenkins\remoting as a remoting work directory
Oct 19, 2024 10:42:51 AM hudson.remoting.Launcher$CuiListener status
INFO: Locating server among [http://ec2-54-164-229-37.compute-1.amazonaws.com:8080/]
Oct 19, 2024 10:42:52 AM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver resolve
INFO: Remoting server accepts the following protocols: [JNLP4-connect, Ping]
Oct 19, 2024 10:42:52 AM hudson.remoting.Launcher$CuiListener status
INFO: Agent discovery successful
    Agent address: ec2-54-164-229-37.compute-1.amazonaws.com
    Agent port: 50000
    Identity: 89:0f:ad:cc:3a:7d:24:57:ef:54:87:39:de:74:38:37
Oct 19, 2024 10:42:52 AM hudson.remoting.Launcher$CuiListener status
INFO: Handshaking
Oct 19, 2024 10:42:52 AM hudson.remoting.Launcher$CuiListener status
INFO: Connecting to ec2-54-164-229-37.compute-1.amazonaws.com:50000
Oct 19, 2024 10:42:52 AM hudson.remoting.Launcher$CuiListener status
INFO: Server reports protocol JNLP4-connect-proxy not supported, skipping
Oct 19, 2024 10:42:52 AM hudson.remoting.Launcher$CuiListener status
INFO: Trying protocol: JNLP4-connect
Oct 19, 2024 10:42:53 AM org.jenkinsci.remoting.protocol.impl.BIONetworkLayer$Reader run
INFO: Waiting for ProtocolStack to start.
Oct 19, 2024 10:42:53 AM hudson.remoting.Launcher$CuiListener status
INFO: Remote identity confirmed: 89:0f:ad:cc:3a:7d:24:57:ef:54:87:39:de:74:38:37
Oct 19, 2024 10:42:54 AM hudson.remoting.Launcher$CuiListener status
INFO: Connected
|
```

Thus at final it gives output connected.

10.

1. Create a pipeline here i have created pipeline name MyWebApp



1. Deploy your code in the pipeline you created before

```
pipeline {
  agent any

  environment {
    EC2_USER = 'ec2-user' // Your EC2 user
    EC2_IP = '54.162.136.27' // Your EC2 public IP address without trailing slash
    SSH_KEY = credentials('myEC2SSHKey') // Your Jenkins credential ID for SSH
  }

  stages {
    stage('Checkout') {
      steps {
        script {
          git branch: 'main', url: 'https://github.com/sadneya145/Leafing-copy.git'
        }
      }
    }

    stage('Clean Previous Installations') {
      steps {
        script {
          dir('frontend') {
            if (fileExists('node_modules')) {
              bat 'rmdir /s /q node_modules'
            }
            if (fileExists('package-lock.json')) {
              bat 'del package-lock.json'
            }
          }
        }
      }
    }

    stage('Install Frontend Dependencies') {
      steps {
        script {
          try {
            dir('frontend') {
              bat 'npm install'
            }
          } catch (Exception e) {
            error "Dependency installation failed: ${e.message}"
          }
        }
      }
    }
  }
}
```

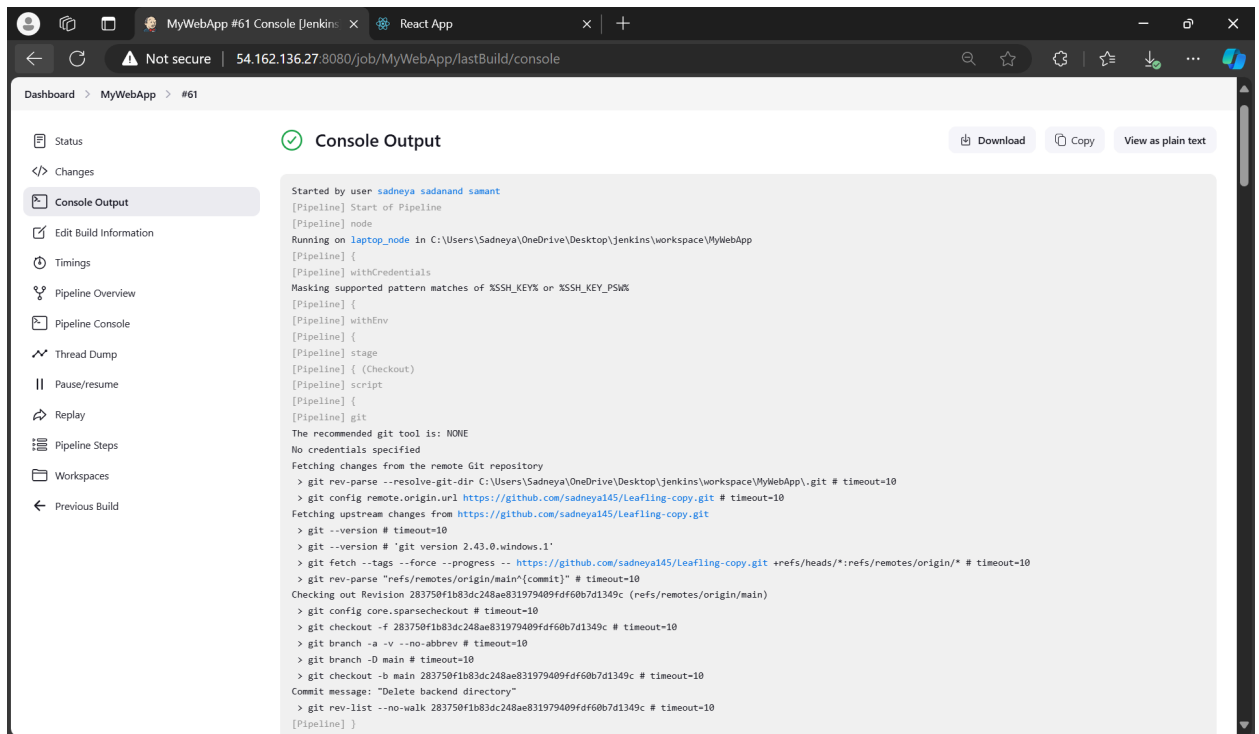
```
    }
  }
}

stage('Build Frontend') {
  steps {
    script {
      dir('frontend') {
        try {
          bat 'npm run build'
        } catch (Exception e) {
          error "Frontend build failed: ${e.message}"
        }
      }
    }
  }
}

stage('Deploy to EC2') {
  steps {
    script {
      // Create the target directory on EC2 and copy frontend build files
      bat """
        ssh -i C:\\Users\\Sadneya\\Downloads\\MyWebKey.pem -o
StrictHostKeyChecking=no ${EC2_USER}@${EC2_IP} "mkdir -p /home/ec2-user/myapp"
        scp -i C:\\Users\\Sadneya\\Downloads\\MyWebKey.pem -o
StrictHostKeyChecking=no -r frontend\\build\\*
${EC2_USER}@${EC2_IP}:/home/ec2-user/myapp/
        """
    }
  }
}

post {
  success {
    echo 'Frontend deployment successful!'
  }
  failure {
    echo 'Frontend deployment failed. Check logs for more details.'
  }
}
}
```


Then after build the output will be:



The screenshot shows the Jenkins web interface for a build named 'MyWebApp #61'. The 'Console Output' tab is selected, displaying the following log:

```
Started by user sadneya sadanand samant
[Pipeline] Start of Pipeline
[Pipeline] node
Running on laptop_node in C:\Users\Sadneya\OneDrive\Desktop\jenkins\workspace\MyWebApp
[Pipeline] {
[Pipeline] withCredentials
Masking supported pattern matches of %SSH_KEY% or %SSH_KEY_PSWD%
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout)
[Pipeline] script
[Pipeline] {
[Pipeline] git
[Pipeline] git

The recommended git tool is: NONE
No credentials specified
Fetching changes from the remote Git repository
> git rev-parse --resolve-git-dir C:\Users\Sadneya\OneDrive\Desktop\jenkins\workspace\MyWebApp\.git # timeout=10
> git config remote.origin.url https://github.com/sadneya145/Leafing-copy.git # timeout=10
Fetching upstream changes from https://github.com/sadneya145/Leafing-copy.git
> git --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
> git fetch --tags --force --progress -- https://github.com/sadneya145/Leafing-copy.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse "refs/remotes/origin/main" (commit) # timeout=10
Checking out Revision 283750f1b83dc248ae831979409fdf60b7d1349c (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 283750f1b83dc248ae831979409fdf60b7d1349c # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git branch -D main # timeout=10
> git checkout -b main 283750f1b83dc248ae831979409fdf60b7d1349c # timeout=10
Commit message: "Delete backend directory"
> git rev-list --no-walk 283750f1b83dc248ae831979409fdf60b7d1349c # timeout=10
[Pipeline] }
```

```
C:\Users\Sadneya\OneDrive\Desktop\jenkins\workspace\MyWebApp@2>ssh -i C:\Users\Sadneya\Downloads\MyWebKey.pem -o StrictHostKeyChecking=no ec2-user@54.162.136.27 "mkdir -p /home/ec2-user/myapp"
Warning: Permanently added '54.162.136.27' (ED25519) to the list of known hosts.
```

```
C:\Users\Sadneya\OneDrive\Desktop\jenkins\workspace\MyWebApp@2>scp -i C:\Users\Sadneya\Downloads\MyWebKey.pem -o StrictHostKeyChecking=no -r frontend\build\* ec2-user@54.162.136.27:/home/ec2-user/myapp/
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Frontend deployment successful!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

```
sudo passwd nagios
```

```
[ec2-user@ip-172-31-40-207 ~]$ sudo adduser -m nagios
sudo passwd nagios
Changing password for user nagios.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: all authentication tokens updated successfully.
```

4. Create a New User Group:

```
sudo groupadd nagcmd
```

5. Modify User Groups:

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

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

```
[ec2-user@ip-172-31-40-207 ~]$ sudo groupadd nagcmd
[ec2-user@ip-172-31-40-207 ~]$ sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache
```

6. Setting Up Nagios

Create a Directory for Nagios Downloads:

```
mkdir ~/downloads
```

```
cd ~/downloads
```

```
[ec2-user@ip-172-31-40-207 ~]$ mkdir ~/downloads
cd ~/downloads
```

7. Download Nagios Source Files:

```
wget https://go.nagios.org/l/975333/2024-09-17/6kqcx
```

```
[ec2-user@ip-172-31-40-207 downloads]$ wget https://go.nagios.org/l/975333/2024-09-17/6kqcx
--2024-10-19 13:42:27-- https://go.nagios.org/l/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org)... 34.237.219.119, 18.208.125.13, 3.92.120.28, ...
Connecting to go.nagios.org (go.nagios.org)|34.237.219.119|:443... connecte
d.
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-19 13:42:27-- http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00:f03c:92ff:fe7f:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-10-19 13:42:27-- https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
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: '6kqcx'

6kqcx          100%[=====] 1.97M  6.72MB/s   in 0.3s

2024-10-19 13:42:28 (6.72 MB/s) - '6kqcx' saved [2065473/2065473]
```

8. Download Nagios Plugins:

wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz

```
[ec2-user@ip-172-31-40-207 downloads]$ wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
--2024-10-19 13:43:09-- http://nagios-plugins.org/download/nagios-plugins-2.0.3.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|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2659772 (2.5M) [application/x-gzip]
Saving to: 'nagios-plugins-2.0.3.tar.gz'

nagios-plugins-2.0 100%[=====] 2.54M 7.78MB/s in 0.3s

2024-10-19 13:43:10 (7.78 MB/s) - 'nagios-plugins-2.0.3.tar.gz' saved [2659772/2659772]
```

9. Unzip the Nagios Source Files:

tar zxvf 6kqcx

```
[ec2-user@ip-172-31-40-207 downloads]$ tar zxvf 6kqcx
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
```

cd nagios-4.5.5

```
[ec2-user@ip-172-31-40-207 downloads]$ cd nagios-4.5.5
```

10. Run Configuration Script:

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

```
[ec2-user@ip-172-31-40-207 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
checking for sys/stat.h... yes
```

```
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
```

****Error Handling:** If you encounter an error about missing SSL headers, install the following:

11. Install SSL Development Package:

```
sudo yum install openssl-devel -y
```

```
[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:11:20 ago on Sat Oct 19 13:33:03 2024.
Dependencies resolved.
=====
Package                Arch      Version                               Repository      Size
=====
Installing:
openssl-devel           x86_64    1:3.0.8-1.amzn2023.0.16              amazonlinux     3.0 M
Transaction Summary
=====
Install 1 Package

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.16.x86_64.rpm    30 MB/s | 3.0 MB  00:00
-----
Total                                           21 MB/s | 3.0 MB  00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing    : openssl-devel-1:3.0.8-1.amzn2023.0.16.x86_64 1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.16.x86_64 1/1
  Verifying     : openssl-devel-1:3.0.8-1.amzn2023.0.16.x86_64 1/1

Installed:
openssl-devel-1:3.0.8-1.amzn2023.0.16.x86_64

Complete!
```

12. Rerun Configuration Script: You will get final output like this

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

```
config.status: creating t/Makefile
config.status: creating t-tap/Makefile
config.status: creating include/ignored_config.h
config.status: creating include/config.h
config.status: creating lib/snprintf.h
config.status: creating lib/iobroker.h

Creating sample config files in sample-config/ ...

*** 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

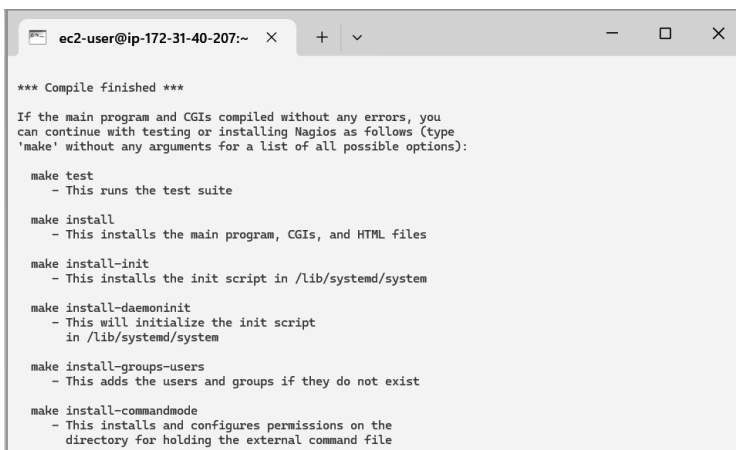
Web Interface Options:
-----
HTML URL: http://localhost/nagios/
CGI URL: http://localhost/nagios/cgi-bin/
Traceroute (used by WAP): /usr/bin/traceroute

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

13. Install Nagios:

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

```
[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ sudo make install
sudo make install-init
sudo make install-config
sudo make install-commandmode
```



ec2-user@ip-172-31-40-207:~ X + v - □ X

```
*** Compile finished ***

If the main program and CGIs compiled without any errors, you
can continue with testing or installing Nagios as follows (type
'make' without any arguments for a list of all possible options):

make test
- This runs the test suite

make install
- This installs the main program, CGIs, and HTML files

make install-init
- This installs the init script in /lib/systemd/system

make install-daemoninit
- This will initialize the init script
  in /lib/systemd/system

make install-groups-users
- This adds the users and groups if they do not exist

make install-commandmode
- This installs and configures permissions on the
  directory for holding the external command file
```

```

*** Main program, CGIs and HTML files installed ***

You can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

make install-init
- This installs the init script in /lib/systemd/system

make install-commandmode
- This installs and configures permissions on the
  directory for holding the external command file

make install-config
- This installs sample config files in /usr/local/nagios/etc

```

14. Configure Nagios Web Interface:

sudo make install-webconf

```

[ec2-user@ip-172-31-40-207 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 ***

```

15. Create Nagios Admin Account:

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

```

[ec2-user@ip-172-31-40-207 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

```

16. Restart Apache:

sudo service httpd restart

```

[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ sudo chkconfig --add nagios
error reading information on service nagios: No such file or directory

```

17. Unzip Nagios Plugins:

cd ~/downloads

tar zxvf nagios-plugins-2.0.3.tar.gz

cd nagios-plugins-2.0.3

```

[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-40-207 downloads]$ tar zxvf nagios-plugins-2.0.3.tar.gz
nagios-plugins-2.0.3/
nagios-plugins-2.0.3/perlmods/
nagios-plugins-2.0.3/perlmods/Config-Tiny-2.14.tar.gz
nagios-plugins-2.0.3/perlmods/parent-0.226.tar.gz
nagios-plugins-2.0.3/perlmods/Test-Simple-0.98.tar.gz
nagios-plugins-2.0.3/perlmods/Makefile.in
nagios-plugins-2.0.3/perlmods/version-0.9903.tar.gz
nagios-plugins-2.0.3/perlmods/Makefile.am
nagios-plugins-2.0.3/perlmods/Module-Runtime-0.013.tar.gz
nagios-plugins-2.0.3/perlmods/Module-Metadata-1.000014.tar.gz
nagios-plugins-2.0.3/perlmods/Params-Validate-1.08.tar.gz
nagios-plugins-2.0.3/perlmods/Class-Accessor-0.34.tar.gz
nagios-plugins-2.0.3/perlmods/Try-Tiny-0.18.tar.gz
nagios-plugins-2.0.3/perlmods/Module-Implementation-0.07.tar.gz
nagios-plugins-2.0.3/perlmods/Makefile

```

```
nagios-plugins-2.0.3/pkg/solaris/pkginfo.in
nagios-plugins-2.0.3/pkg/solaris/pkginfo
nagios-plugins-2.0.3/pkg/redhat/
nagios-plugins-2.0.3/pkg/redhat/requires
[ec2-user@ip-172-31-40-207 downloads]$ |
```

18. Verify Nagios Configuration:

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

```
[ec2-user@ip-172-31-40-207 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
```

19. **Start Nagios Service:** sudo service nagios start

20. **Check Nagios Status:** sudo systemctl status nagios


```
[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
[ec2-user@ip-172-31-40-207 nagios-4.5.5]$ 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 Sat 2024-10-19 13:56:27 UTC; 5s ago
     Docs: https://www.nagios.org/documentation
   Process: 72024 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Process: 72025 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 72026 (nagios)
    Tasks: 6 (limit: 1112)
   Memory: 5.5M
      CPU: 79ms
   CGroup: /system.slice/nagios.service
           └─72026 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             └─72027 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
               └─72028 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                 └─72029 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                   └─72030 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                     └─72031 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' created
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: qh: core query handler registered
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: qh: echo service query handler registered
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: qh: help for the query handler registered
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: wproc: Successfully registered manager
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: wproc: Registry request: name=Core Worker
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: wproc: Registry request: name=Core Worker
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: wproc: Registry request: name=Core Worker
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: wproc: Registry request: name=Core Worker
Oct 19 13:56:27 ip-172-31-40-207.ec2.internal nagios[72026]: Successfully launched command file worker
lines 1-28/28 (END)
```

21. **Get Public IP Address:**Go back to the EC2 Console and copy the public IP address of your instance.
22. **Access Nagios Web Interface:** Open your web browser and navigate to:
`http://<your_public_ip_address>/nagios`

Enter the username (nagiosadmin) and the password you set in Step 15.

Making Changes for application in Nagios

1. Goto configurations file

```
[ec2-user@ip-172-31-45-86 ~]$ sudo nano /usr/local/nagios/etc/nagios.cfg
```

Add the file which we are newly creating for monitoring of our application

cfg_file=/usr/local/nagios/etc/objects/myweb.cfg add this

```
log_file=/usr/local/nagios/var/nagios.log

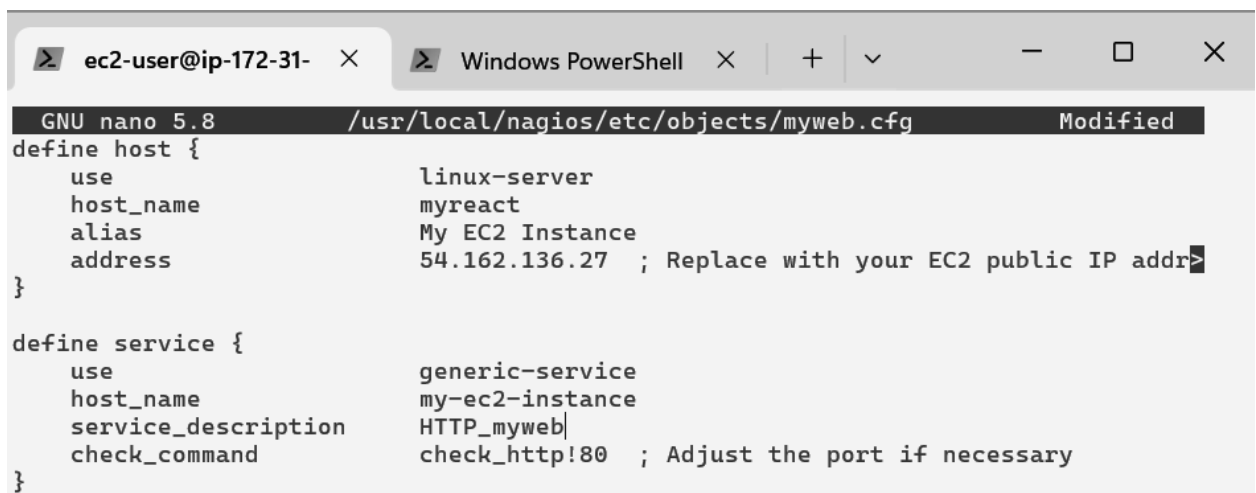
# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg
cfg_file=/usr/local/nagios/etc/objects/myec2.cfg
# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/myweb.cfg
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg
```

2. .write inside that file the required information

sudo nano /usr/local/nagios/etc/objects/myweb.cfg

```
[ec2-user@ip-172-31-45-86 backend]$ sudo nano /usr/local/nagios/etc/objects/myweb.cfg
```



```
GNU nano 5.8 /usr/local/nagios/etc/objects/myweb.cfg Modified
define host {
    use                linux-server
    host_name          myreact
    alias              My EC2 Instance
    address            54.162.136.27 ; Replace with your EC2 public IP address
}

define service {
    use                generic-service
    host_name          my-ec2-instance
    service_description HTTP_myweb
    check_command      check_http!80 ; Adjust the port if necessary
}
```

3. Again Verify the changes by : `sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg`

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

```
[ec2-user@ip-172-31-45-86 backend]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
License: GPL

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

Running pre-flight check on configuration data...

Checking objects...
  Checked 12 services.
  Checked 3 hosts.
  Checked 1 host groups.
  Checked 0 service groups.
  Checked 1 contacts.
  Checked 1 contact groups.
  Checked 25 commands.
  Checked 5 time periods.
  Checked 0 host escalations.
  Checked 0 service escalations.
Checking for circular paths...
  Checked 3 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
```

4. again start nagios `sudo systemctl start nagios` and check status by `sudo systemctl status nagios`

```
[ec2-user@ip-172-31-45-86 ~]$ sudo systemctl start nagios
[ec2-user@ip-172-31-45-86 ~]$ sudo systemctl enable nagios
[ec2-user@ip-172-31-45-86 ~]$ 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 Sat 2024-10-19 15:39:56 UTC; 1min 34s ago
     Docs: https://www.nagios.org/documentation
    Main PID: 11217 (nagios)
      Tasks: 8 (limit: 1112)
     Memory: 4.1M
        CPU: 40ms
    CGroup: /system.slice/nagios.service
            └─11217 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
              └─11218 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                └─11219 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                  └─11220 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                    └─11221 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
                      └─11222 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
                        └─11355 /usr/local/nagios/libexec/check_ping -H 98.82.169.214 -w 3000.0,80% -c 5000.0,10%
                          └─11356 /usr/bin/ping -n -U -w 30 -c 5 98.82.169.214

Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: qh: echo service query handler registered
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: qh: help for the query handler registered
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: wproc: Successfully registered manager
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: wproc: Registry request: name=Core Worker
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: wproc: Registry request: name=Core Worker
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: wproc: Registry request: name=Core Worker
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: wproc: Registry request: name=Core Worker
Oct 19 15:39:56 ip-172-31-45-86.ec2.internal nagios[11217]: Successfully launched command file worker
Oct 19 15:40:26 ip-172-31-45-86.ec2.internal nagios[11217]: HOST ALERT: my-ec2-instance;DOWN;SOFT;1;
Oct 19 15:40:58 ip-172-31-45-86.ec2.internal nagios[11217]: SERVICE ALERT: my-ec2-instance;HTTP;WARNING;1;
lines 1-28/28 (END)
```

5. Go to commands file there make changes in check_http section

Add -> command line /usr/local/nagios/libexec/check_http -H \$HOSTADDRESS\$ -p \$ARG1\$ \$ARG2\$

```
define command {
    command_name    check_snmp
    command_line    $USER1$/check_snmp -H $HOSTADDRESS$ $ARG1$
}

define command {
    command_name    check_http
    # command_line    $USER1$/check_http -I $HOSTADDRESS$ $ARG1$
    command_line    /usr/local/nagios/libexec/check_http -H $HOSTADDRESS$ -p $ARG1$ $ARG2$
}

define command {
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo
 ^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line M-E Redo

6. **Nagios page:** Go back to your nagios page you will see output. here you will see my-ec2-instance.

The screenshot shows the Nagios web interface at 54.162.136.27/nagios/. The interface includes a sidebar with navigation links, a top status bar, and a main content area displaying host status details.

Current Network Status
 Last Updated: Sun Oct 20 15:40:10 UTC 2024
 Updated every 90 seconds
 Nagios® Core™ 4.4.6 - www.nagios.org
 Logged in as nagiosadmin

Host Status Totals
 Up: 3, Down: 0, Unreachable: 0, Pending: 0
 All Problems: 0, All Types: 3

Service Status Totals
 OK: 7, Warning: 0, Unknown: 1, Critical: 3, Pending: 0
 All Problems: 4, All Types: 11

Host Status Details For All Host Groups

Host	Status	Last Check	Duration	Status Information
localhost	UP	10-20-2024 15:38:18	1d 6h 12m 25s	PING OK - Packet loss = 0%, RTA = 0.04 ms
my-ec2-instance	UP	10-20-2024 15:35:47	0d 0h 9m 23s	PING OK - Packet loss = 0%, RTA = 0.59 ms
myread	UP	10-20-2024 15:38:35	0d 0h 11m 35s	PING OK - Packet loss = 0%, RTA = 0.48 ms

Results 1 - 3 of 3 Matching Hosts

Go to host section present on left sidebar and click on “my-ec2-intsance” it will give host information.

The screenshot displays the Nagios web interface for a host named 'my-react' (IP: 54.162.136.27). The interface is divided into several sections:

- Host Information:** Shows the host is 'UP' (for 0d 0h 18m 45s). It includes status information, performance data (PING OK, Packet loss = 0%, RTA = 0.42 ms), current attempt (1/10), last check time (10-20-2024 15:43:35), check type (ACTIVE), check latency (0.000 / 4.162 seconds), next scheduled active check (10-20-2024 15:48:35), last state change (10-20-2024 15:28:35), last notification (N/A), and is not flapping.
- Host State Information:** Lists various checks and their status: Active Checks (ENABLED), Passive Checks (ENABLED), Obssessing (ENABLED), Notifications (ENABLED), Event Handler (ENABLED), and Flap Detection (ENABLED).
- Host Commands:** A list of commands that can be executed on the host, such as 'Locate host on map', 'Disable active checks of this host', 'Re-schedule the next check of this host', etc.
- Host Comments:** A section for adding or deleting comments about the host.

The left sidebar contains navigation links for General, Home, Documentation, Current Status, Tactical Overview, Map (Legacy), Hosts, Services, Host Groups, Summary, Grid, Service Groups, Summary, Grid, Problems, Services (Unhandeled), Hosts (Unhandeled), Network Outages, Quick Search, Reports, Availability, Trends (Legacy), Alerts, History, Summary, Histogram (Legacy), Notifications, Event Log, System, Comments, Downtime, Process Info, Performance Info, Scheduling Queue, and Configuration.

Now click on services on left sidebar you will get detailed information about network status.

The screenshot displays the Nagios web interface for service status details. The top section shows 'Current Network Status' and 'Host Status Totals'. Below this, the 'Service Status Details For All Hosts' table is shown, listing various services and their status across different hosts.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
localhost	Current Load	OK	10-20-2024 15:44:08	0d 0h 20m 40s	1/4	OK - load average: 0.00, 0.00, 0.00
localhost	Current Users	OK	10-20-2024 15:44:20	1d 6h 20m 3s	1/4	USERS OK - 2 users currently logged in
localhost	HTTP	UNKNOWN	10-20-2024 15:45:23	0d 23h 54m 8s	4/4	Usage:
localhost	Jenkins Monitoring	CRITICAL	10-20-2024 15:44:14	1d 6h 42m 32s	3/3	HTTP CRITICAL - Invalid HTTP response received from host on port 8080: HTTP/1.1 403 Forbidden
localhost	Jenkins Status	CRITICAL	10-20-2024 15:40:41	1d 7h 4m 18s	3/3	HTTP CRITICAL - Invalid HTTP response received from host on port 8080: HTTP/1.1 403 Forbidden
localhost	PING	OK	10-20-2024 15:46:25	1d 6h 19m 30s	1/4	PING OK - Packet loss = 0%, RTA = 0.04 ms
localhost	Root Partition	OK	10-20-2024 15:47:28	1d 6h 17m 32s	1/4	DISK OK - free space / 4565 MB (56% inode=97%):
localhost	SSH	OK	10-20-2024 15:43:18	1d 6h 17m 32s	1/4	SSH OK - OpenSSH_8.7 (protocol 2.0)
localhost	Swap Usage	CRITICAL	10-20-2024 15:43:42	1d 7h 44m 11s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
localhost	Total Processes	OK	10-20-2024 15:44:59	1d 6h 17m 32s	1/4	PROCS OK - 36 processes with STATE = RSZDT
my-ec2-instance	HTTP	OK	10-20-2024 15:40:47	0d 0h 17m 1s	1/3	HTTP OK: HTTP/1.1 200 OK - 1347 bytes in 0.001 second response time
my-ec2-instance	HTTP_myweb	OK	10-20-2024 15:46:44	0d 0h 2m 37s+	1/3	HTTP OK: HTTP/1.1 200 OK - 1347 bytes in 0.002 second response time

Results 1 - 12 of 12 Matching Services

Conclusion

This case study involved setting up an automated CI/CD pipeline with Jenkins to deploy a web app on AWS EC2, and using Nagios for monitoring. We faced challenges like SSH configuration, limited Jenkins disk space, and SSL issues with Nagios, which were resolved through security adjustments and required package installations. Key takeaways included the importance of secure automation and effective monitoring for maintaining a reliable deployment process.