

Project 4 → Deploy Netflix Clone on Kubernetes

Completion steps →



Aakib · [Follow](#)

18 min read · Nov 3, 2023

198

6

+

▶

↑

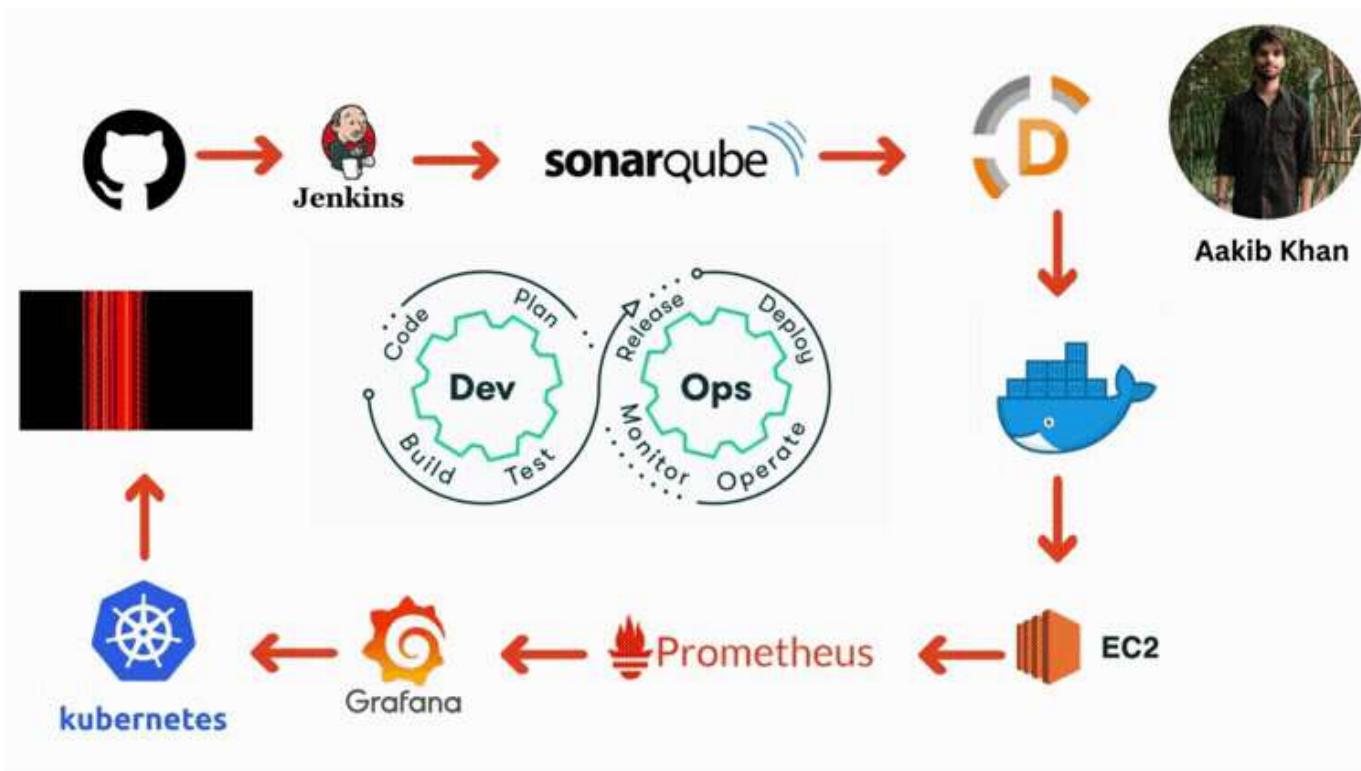
Phase 1 → deploy netflix locally on ec2 (t2.large)

Phase 2 → Implementation of security with sonarqube and trivy

Phase 3 → Now we automate the whole deployment using by jenkins pipeline

Phase 4 → Monitering via Prometheus and grafana

Phase 5 → Kubernetes →



Phase 1 → deploy netflix locally on ec2 (t2.large)

Step 1 → setup ec2

1. go to aws console and launch ubuntu 22.04 with t2.large and 25 gb of storage allocated with it don't forget to enable public ip in vpc settings

Activities Google Chrome Oct 27 20:15

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:

Gmail YouTube Maps Magma India... docker comm... New Tab Find Geop... 30+ Java Exec... kickresunme (1) ndtv live hi... Accijob: Full... AMCOP-Mag... Magma India All Bookmarks N. Virginia aakib khan

aWS Services Search [Alt+S]

Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Li

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
ami-0fc5d935ebfbc3bc (64-bit (x86)) / ami-016405166ec7fa705 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description
Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-09-19

Architecture 64-bit (x86) **AMI ID** ami-0fc5d935ebfbc3bc **Verified provider**

Instance type **t2.large**

Family: t2 2 vCPU 8 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.1200 USD per Hour
On-Demand RHEL base pricing: 0.1528 USD per Hour
On-Demand SUSE base pricing: 0.1920 USD per Hour
On-Demand Linux base pricing: 0.0928 USD per Hour

Additional costs apply for AMIs with pre-installed software

Summary

Number of instances **1**

Software image (AMI)
Canonical, Ubuntu, 22.04 LTS, ...read more
ami-0fc5d935ebfbc3bc

Virtual server type (instance type)
t2.large

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 usage on free tier AMIs per month, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel Launch instance Review commands

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

2. let's connect to your ec2 via ssh using command "ssh -i "mykey.pem"
ubuntu@ec2-54-197-62-157.compute-1.amazonaws.com"

Activities Google Chrome Oct 27 20:35

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ConnectToInstance:instanceId=i-0add587b9cccd10af9

Gmail YouTube Maps Magma India... docker comm... New Tab Find Geop... 30+ Java Exec... kickresunme (1) ndtv live hi... Accijob: Full... AMCOP-Mag... Magma India All Bookmarks N. Virginia aakib khan

aWS Services Search [Alt+S]

EC2 > Instances > i-0add587b9cccd10af9 > Connect to instance

Connect to instance

Connect to your instance i-0add587b9cccd10af9 (netflix app) using any of these options

SSH client

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

Instance ID
i-0add587b9cccd10af9 (netflix app)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is mykey.pem
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 mykey.pem
- Connect to your instance using its Public DNS:
ec2-54-197-62-157.compute-1.amazonaws.com

Example:
ssh -i "mykey.pem" ubuntu@ec2-54-197-62-157.compute-1.amazonaws.com

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

Cancel

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

3. run the following commands

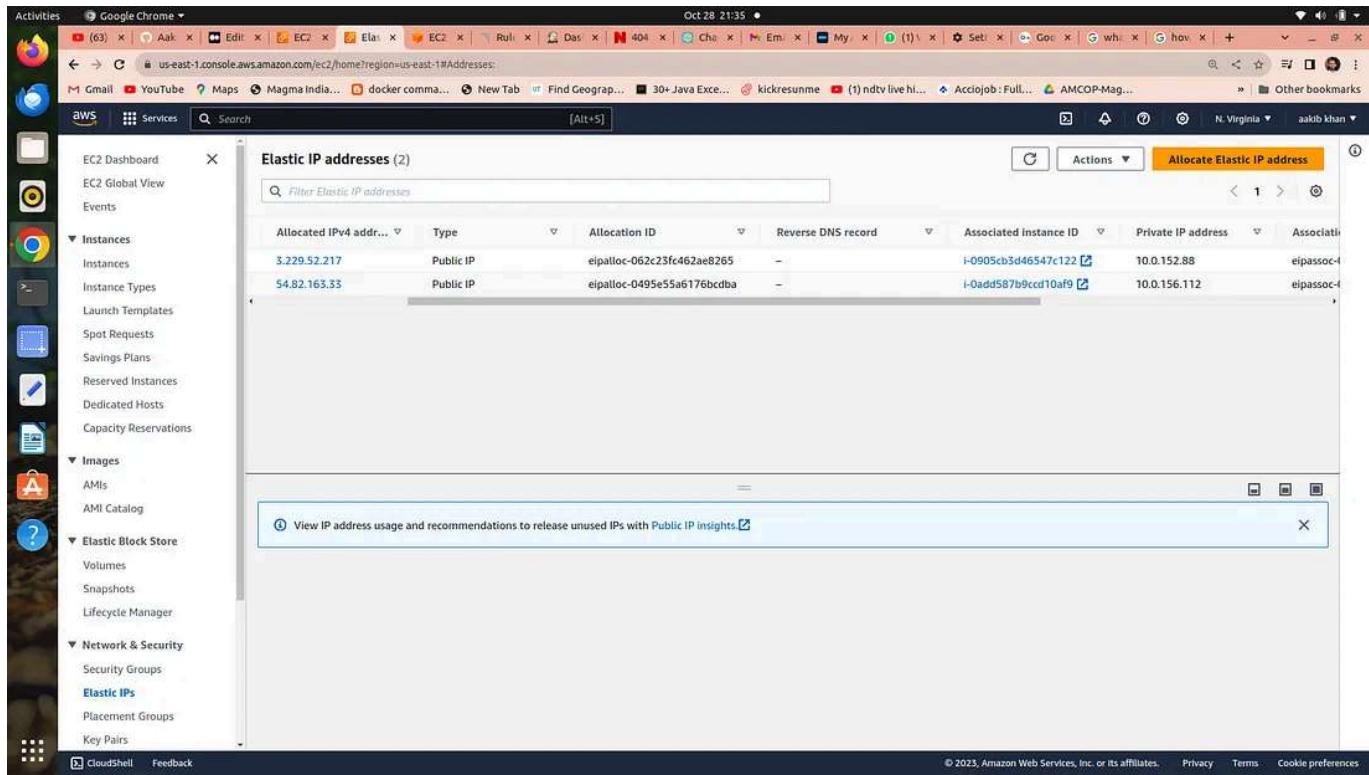
a. sudo su

b. apt update

clone the github repo by

c. git clone <https://github.com/Aakibgithuber/Deploy-Network-Clone-on-Kubernetes>

Make sure to create an elastic ip address to associate with your instance



The screenshot shows the AWS Cloud9 interface with the 'Elastic IP addresses' section open. The left sidebar shows various AWS services like EC2, S3, and Lambda. The main pane displays a table of allocated IPv4 addresses:

Allocated IPv4 address	Type	Allocation ID	Reverse DNS record	Associated Instance ID	Private IP address	Associate
3.229.52.217	Public IP	eipalloc-062c23fc462ae8265	-	i-0905cb3d46547c122	10.0.152.88	eipassoc-0905cb3d46547c122
54.82.163.33	Public IP	eipalloc-0495e55a6176bcd8a	-	i-0add587b5cc10af9	10.0.156.112	eipassoc-0add587b5cc10af9

Step 2 → setup docker and build images

run the following commands to install docker→

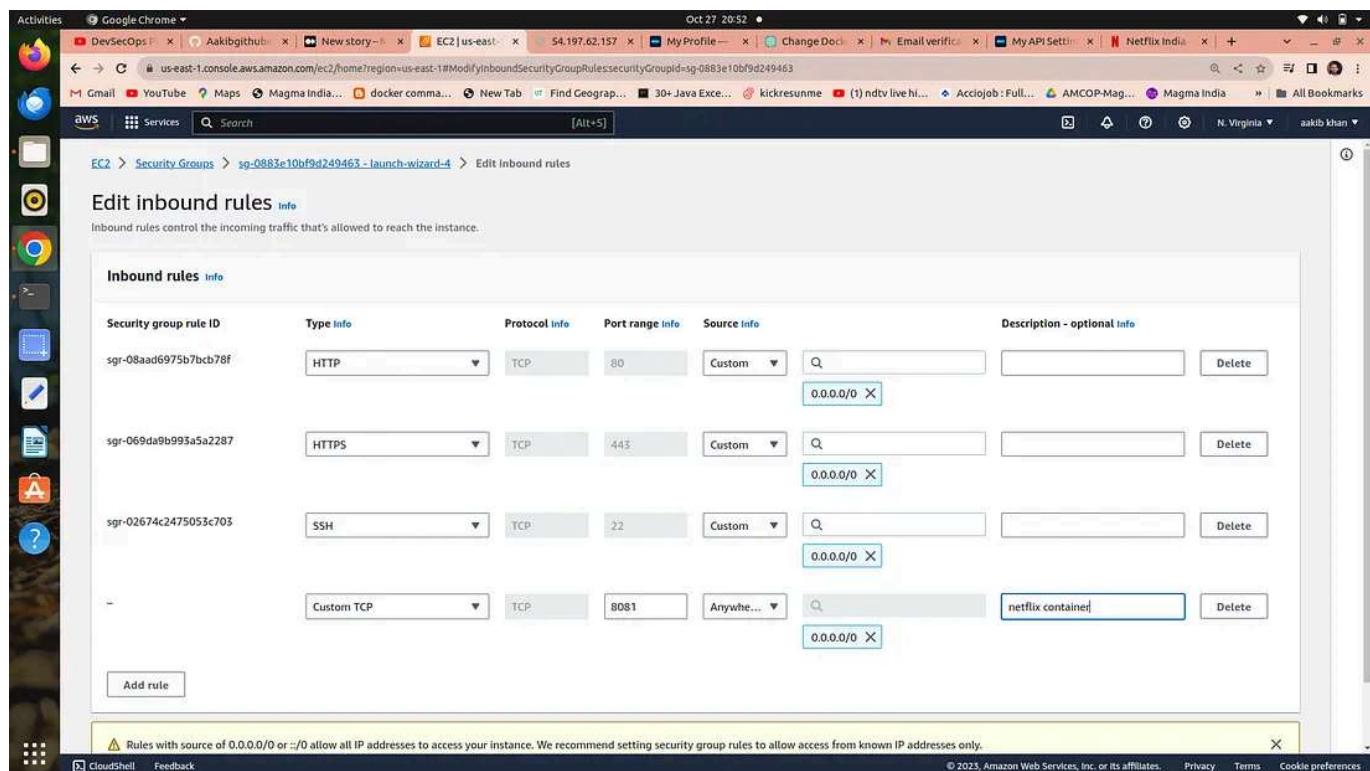
- a. `apt-get install docker.io`
- b. `usermod -aG docker $USER` # Replace with your username e.g 'ubuntu'
- c. `newgrp docker`
- d. `sudo chmod 777 /var/run/docker.sock` → #is used to grant full read, write, and execute permissions to all users for the Docker socket file, which allows unrestricted access to the Docker daemon and is a significant security risk.

run the following commands to build and run docker container →

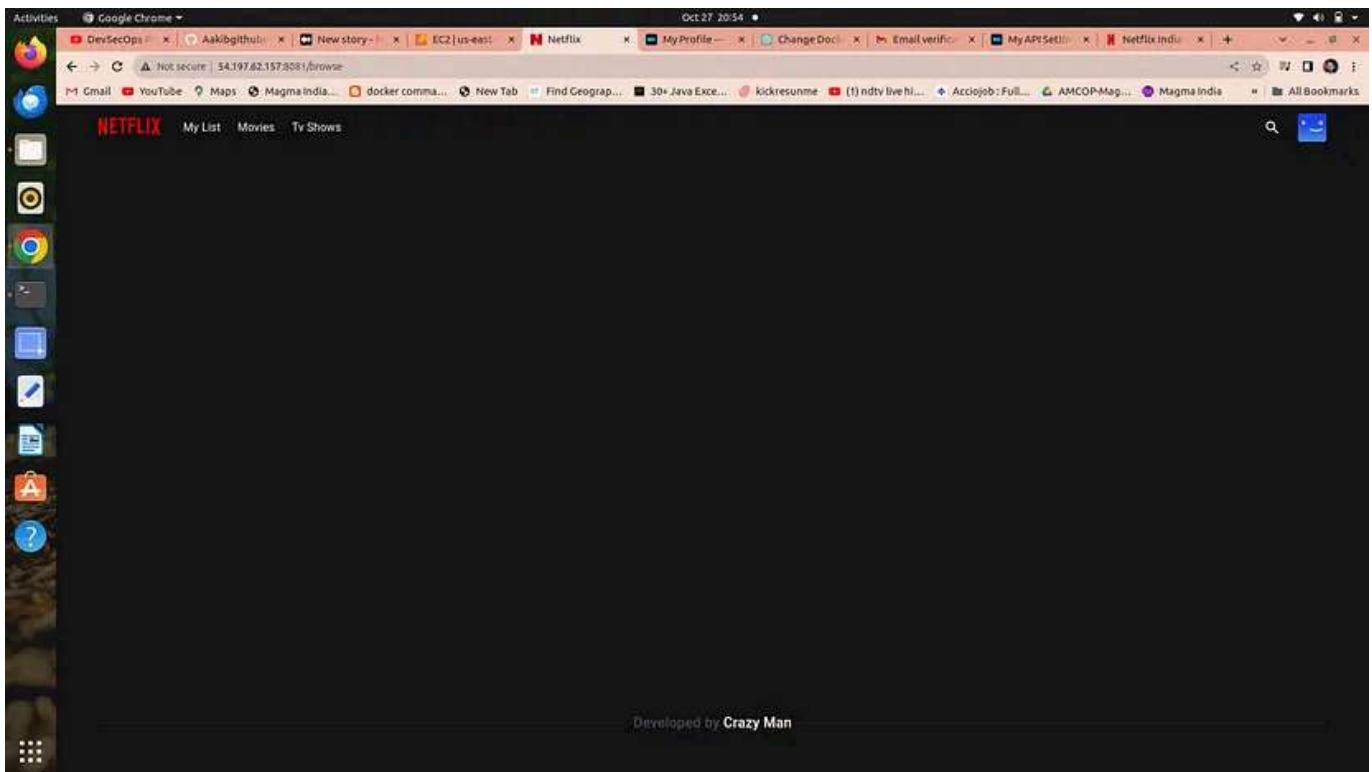
a. `docker build -t netflix .`

b. `docker run -d --name netflix -p 8081:80 netflix:latest` → this maps the container port to your ec2 port

c. go to your ec2 →security groups →open the port 8081 by adding rule
custom tcp = 8081



your application is running go to your ec2 copy public ip and browse http://your_public_ip:8081 and it's open like this



It shows a blank page of netflix because you don't have a api that's communicate with netflix database let's solve this

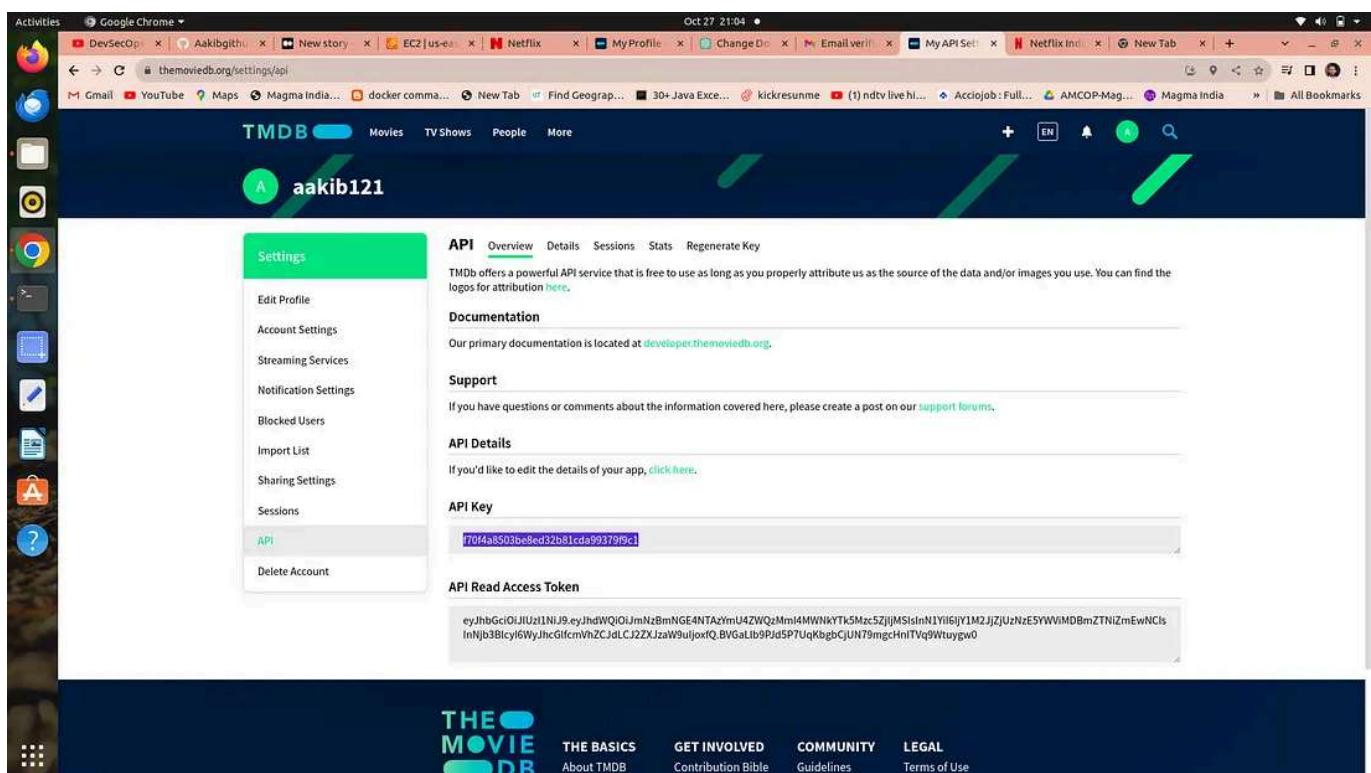
what is an API→

An API (Application Programming Interface) is like a menu for a restaurant that lets you order food (data or functions) from a software system or application, allowing different programs to talk to each other and share information in a structured way.

Step 3 → setup netflix API

- Open a web browser and navigate to TMDB (The Movie Database) website.
- Click on sign up

- Now enter your username and pass for sign in then go to your profile and select “Settings.”
- Click on “API” from the left-side panel.
- Create a new API key by clicking “generate new api key ” and accepting the terms and conditions.
- Provide the required basic details such as name and website url of netflix and click “Submit.”
- You will receive your TMDB API key.



copy the api key

Now delete the existing image by

- a. docker stop <containerid>
- b. docker rmi -f netflix

Run the new container by following command

1. docker build -t netflix:latest — build-arg

TMDB_V3_API_KEY=your_api_key .

2. docker run -d -p 8081:80 netflix

your container is created

now again browse the same url you will see the whole netflix database is connected to your netflix clone app



congrats your application is running

Phase 2 → Implementation of security with sonarqube and trivy

what is sonarqube →

SonarQube is like a “code quality detective” tool for software developers that scans your code to find and report issues, bugs, security vulnerabilities, and code smells to help you write better and more reliable software. It provides

insights and recommendations to improve the overall quality and maintainability of your codebase

what is trivy →

Trivy is like a “security guard” for your software that checks for vulnerabilities in the components (like libraries) your application uses, helping you find and fix potential security problems to keep your software

[Open in app ↗](#)

[Sign up](#)

[Sign in](#)

Medium



Search



Write



step 1 → setup sonarqube on your ec2

run the following commands to isntall and run the conatainer of sonarqube on port no. 9000

a. docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

```
root@ip-10-0-156-112:/home/ubuntu/Deploy-Nerlix-Clone-on-Kubernetes# docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
43f89b94cd7d: Pull complete
50431c77a77b: Pull complete
dfd8e860e672: Pull complete
637e2db99ae6: Pull complete
7de1c2853278: Pull complete
65f8a9882ff2: Pull complete
2da5e7c0f043: Pull complete
Digest: sha256:31604c5edd7ba6dc116fbe00a11744c6559a46524d8232852fe8aa8338149175
Status: Downloaded newer image for sonarqube:lts-community
28ce1ae9f2ad100647718c30c5283696425f3797273fdb3eafe76e101c5a6624
root@ip-10-0-156-112:/home/ubuntu/Deploy-Nerlix-Clone-on-Kubernetes#
```

go and open port 9000 forn sonarqube to your ec2 security group

Activities Google Chrome Oct 27 21:46

DevSecO Aakibgill Newsto EC2 us- EC2 Instan Netflix MyProf Change Email ver My APIs (1) What New Tab AWS Services Search [Alt+S]

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

Inbound rules

Security group rule ID	Type info	Protocol info	Port range info	Source info	Description - optional info
sgr-0901e0edb806d4691	Custom TCP	TCP	8081	Custom	netflix container
sgr-08aad6975b7bc78f	HTTP	TCP	80	Custom	
sgr-069da9b993a5a2287	HTTPS	TCP	443	Custom	
sgr-02674c2475053c703	SSH	TCP	22	Custom	
-	Custom TCP	TCP	9000	Anywhere	sonarqube

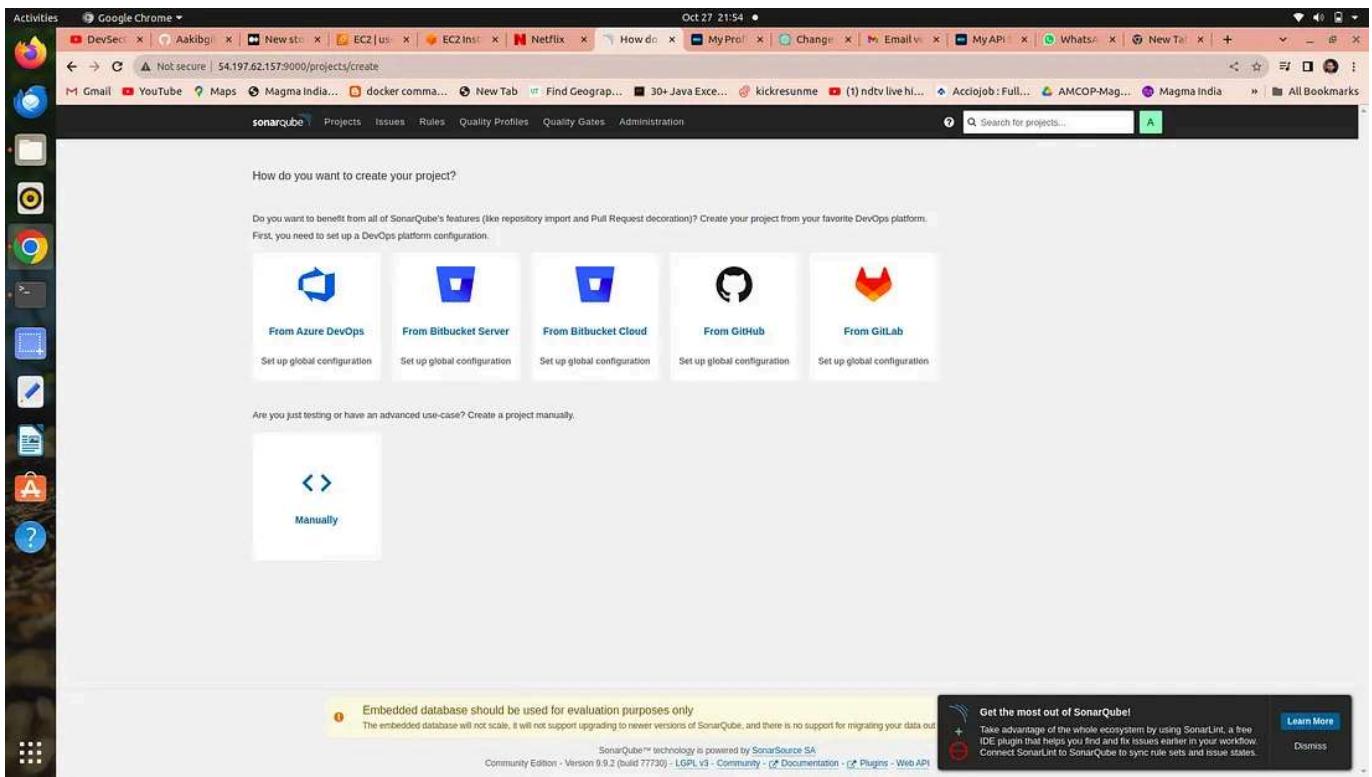
Add rule Closeshell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the AWS CloudWatch Metrics interface. At the top, there are tabs for 'Metrics' and 'Logs'. Below the tabs, there is a search bar and a dropdown menu for selecting a metric namespace. The main area displays a table of metrics with columns for 'Metric Name', 'Unit', 'Period', 'Value Type', and 'Last Value'. One row is highlighted in yellow, showing the metric name 'HelloWorld:InvokeCount', unit 'Count', period '1 minute', value type 'Sum', and last value '1'. There is also a 'Create New Metric' button at the bottom of the table.

Now browse http://your_public_ip:9000 you will see sonarqube is running

username =admin

password=admin



your sonarqube dashboard

Step 2 → setup Trivy

run the following commands

a. sudo apt-get install wget apt-transport-https gnupg lsb-release

b. wget -qO - <https://aquasecurity.github.io/trivy-repo/deb/public.key> | sudo apt-key add -

c. echo deb <https://aquasecurity.github.io/trivy-repo/deb> \$(lsb_release -sc) main | sudo tee -a /etc/apt/sources.list.d/trivy.list

d. sudo apt-get update

e. sudo apt-get install trivy

now your trivy is ready to check and scan your image for any vulnerabilities

to check run the following commands

a. trivy image <image id>

```
root@ip-10-0-156-112:/home/ubuntu/Deploy-Network-Clone-on-Kubernetes# trivy image 8a40eaaa671f
2023-10-27T16:30:26.899Z    INFO  Need to update DB
2023-10-27T16:30:26.899Z    INFO  DB Repository: gcr.io/aquasecurity/trivy-db
2023-10-27T16:30:26.899Z    INFO  Downloading DB...
40.71 MiB / 40.71 MiB [=====] 100.00% 22.48 MiB p/s 2.0s
2023-10-27T16:30:29.111Z    INFO  Vulnerability scanning is enabled
2023-10-27T16:30:29.111Z    INFO  Secret scanning is enabled
2023-10-27T16:30:29.111Z    INFO  If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2023-10-27T16:30:29.111Z    INFO  Please see also https://aquasecurity.github.io/trivy/v0.46/docs/scanner/secret/#recommendation for faster secret detection
2023-10-27T16:30:38.198Z    INFO  Detected OS: alpine
2023-10-27T16:30:38.196Z    INFO  Detecting Alpine vulnerabilities...
2023-10-27T16:30:38.117Z    INFO  Number of language-specific files: 0

8a40eaaa671f (alpine 3.17.5)
Total: 2 (UNKNOWN: 0, LOW: 0, MEDIUM: 2, HIGH: 0, CRITICAL: 0)



| Library    | Vulnerability | Severity | Status | Installed Version | Fixed Version | Title                                                                                                                                              |
|------------|---------------|----------|--------|-------------------|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| libcrypto3 | CVE-2023-5363 | MEDIUM   | fixed  | 3.0.11-r0         | 3.0.12-r0     | Incorrect cipher key and IV length processing<br><a href="https://avd.aquasec.com/nvd/cve-2023-5363">https://avd.aquasec.com/nvd/cve-2023-5363</a> |
| libssl3    |               |          |        |                   |               |                                                                                                                                                    |


root@ip-10-0-156-112:/home/ubuntu/Deploy-Network-Clone-on-Kubernetes#
```

here's the scan of your image of netflix

Phase 3 → Now we automate the whole deployment using by jenkins pipeline

Step 1 → install and configure jenkins via terminal ...jenkins also required java for run itself so we install java and then jenkins

commands to run on the terminal →

setup java

1. sudo apt update
2. sudo apt install fontconfig openjdk-17-jre
3. java -version

output → openjdk version “17.0.8” 2023-07-18

OpenJDK Runtime Environment (build 17.0.8+7-Debian-1deb12u1)

OpenJDK 64-Bit Server VM (build 17.0.8+7-Debian-1deb12u1, mixed mode, sharing)

Setup jenkins

1. sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
<https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key>
2. echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
<https://pkg.jenkins.io/debian-stable> binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
3. sudo apt-get update
4. sudo apt-get install jenkins
5. sudo systemctl start jenkins
6. sudo systemctl enable jenkins
7. Access Jenkins in a web browser using the public IP of your EC2 instance
<http://publicIp:8080>

Install some suggested plugins for pipeline to run without errors

1 Eclipse Temurin Installer (Install without restart)

2 SonarQube Scanner (Install without restart)

3 NodeJs Plugin (Install Without restart)

4 Email Extension Plugin

6. **owasp** →The OWASP Plugin in Jenkins is like a “security assistant” that helps you find and fix security issues in your software. It uses the knowledge and guidelines from the Open Web Application Security Project (OWASP) to scan your web applications and provide suggestions on how to make them more secure. It’s a tool to ensure that your web applications are protected against common security threats and vulnerabilities.

7. Prometheus metrics → to monitor jenkins on grafana dashboard

8. Download all the docker related plugins

The screenshot shows the Jenkins Plugin Manager interface. A search bar at the top contains the text "docker". Below the search bar, a list of plugins is displayed, each with a name, version, description, and a toggle switch for enabling or disabling it. The plugins listed are:

- Docker API Plugin 1.1.1-79-v200_53427e0d41
- Docker Commons Plugin 439.v9_30_09_0a_7b_29
- Docker Compose Build Step Plugin 1.0
- Docker Compose plugin for Jenkins
- Docker Pipeline 3.72.v930f50933b43
- Docker plugin 1.3
- Docker Slaves Plugin 1.0.7
- docker-build-step 2.10

Each plugin entry includes a "Report an issue with this plugin" link. The Jenkins version is shown as 2.414.2 at the bottom right.

The screenshot shows the Jenkins Manage Jenkins > Plugins page. The "Download progress" section is highlighted. It displays a list of plugins being downloaded, each with a status indicator (Pending) and a circular progress bar. The listed plugins are:

- Eclipse Temurin installer
- SonarQube Scanner
- Config File Provider
- NodeJS
- Email Extension Template
- Loading plugin extensions

Below the list, there are two links: "→ Go back to the top page" and "→ Restart Jenkins when installation is complete and no jobs are running". The Jenkins version is shown as 2.414.2 at the bottom right.

Step 2 → add credentials of Sonarqube and Docker

1st we generate a token for sonarqube to use in jenkins credentials as secret text

setup sonarqube credentials

1. go to <http://publicip:9000>
2. now enter your username and password
3. click on security → users → token → generate token

The screenshot shows the SonarQube Administration interface. The URL in the browser is <http://3.82.10.24:9000/admin/users>. The page title is "Administration". Under the "Security" tab, the "Users" section is displayed. It shows a table with one user entry:

SCM Accounts	Last connection	Groups	Tokens
sonar-administrators sonar-users	< 1 hour ago	0	0

Below the table, there is a "Create User" button. At the bottom of the page, there is a warning message about the embedded database and a "Get the most out of SonarQube!" promotional banner.

The screenshot shows a Linux desktop environment with a dock on the left containing various icons. A Google Chrome window is open, displaying the SonarQube 'Tokens of Administrator' page. The page has a 'Generate Tokens' section where a token named 'jenkins' is created with an expiration of 30 days. A success message states 'New token "jenkins" has been created. Make sure you copy it now, you won't be able to see it again!'. Below this, a table lists the token details: Name (jenkins), Type (User), Project (empty), Last use (Never), Created (October 28, 2023), and Expiration (November 27, 2023). A 'Revoke' button is visible next to the expiration date. At the bottom right of the page is a 'Done' button. The browser's address bar shows the URL '3.82.10.24:9000/admin/users'. The status bar at the bottom of the screen indicates 'Oct 28 15:16'.

4. copy the token and go to your jenkins → manage jenkins → credentials → global → add credentials

5. select secret text from dropdown

6. secret text ==your token , id =sonar-token → click on create

setup projects in sonarqube for jenkins

1. go to your sonarqube server

2. click on projects

3. in the name field type Netflix

4. click on set up

Create a project

All fields marked with * are required

Project display name *

Up to 255 characters. Some scanners might override the value you provide.

Project key *

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumerics, '-' (dash), '_' (underscore), '.' (period) and '-' (colon), with at least one non-digit.

Main branch name *

The name of your project's default branch [\[?\]](#) [I want More](#)

[Get Up](#)

Are you just testing or have an advanced use-case? Analyze your project locally.



Locally

click on this option

We initialized your project on SonarQube, now it's up to you to launch analyses!

1 Provide a token

Generate a project token

Token name	Expires in
Analyze "Netflix"	30 days
<input type="button" value="Generate"/>	

Please note that this token will only allow you to analyze the current project. If you want to use the same token to analyze multiple projects, you need to generate a global token in your user account. See the documentation for more information.

Use existing token

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point in time in your user account.

click on generate

Activities Google Chrome Oct 28 15:41 •

Not secure | 3.82.10.24:9000/dashboard?id=Netflix&selectedTutorial=local

Gmail YouTube Maps Magma India... docker comman... New Tab Find Geograph... 30+ Java Execu... kickresumne ndtv live h... Acciojob: Full... AMCP-Mag... Other bookmarks

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Netflix main

Overview Issues Security Hotspots Measures Code Analysis Project Settings Project Information

Provide a token Analyze "Netflix":sqn_704fe155979c26cc7beed23a066874174734b129

Run analysis on your project

What option best describes your build? Maven Gradle .NET Other (for JS, TS, Go, Python, PHP, ...)

What is your OS? Linux Windows macOS

Download and unzip the Scanner for Linux

Visit the official documentation of the Scanner to download the latest version, and add the bin directory to the PATH environment variable.

Execute the Scanner

Running a SonarQube analysis is straightforward. You just need to execute the following commands in your project's folder.

```
sonar-scanner \
-sonar.projectKey=Netflix \
-sonar.sources= \
-sonar.host.url=http://3.82.10.24:9000 \
-sonar.login=top_704fe155979c26cc7beed23a066874174734b129
```

Please visit the official documentation of the Scanner for more details.

Is my analysis done? If your analysis is successful, this page will automatically refresh in a few moments.

You can set up Pull Request Decoration under the project settings. To set up analysis with your favorite CI tool, see the tutorials.

Check these useful links while you wait: Branch Analysis, Pull Request Analysis.

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 9.9.2 (build 77730) - LGPL v3 - Community - Documentation - Plugins - Web API

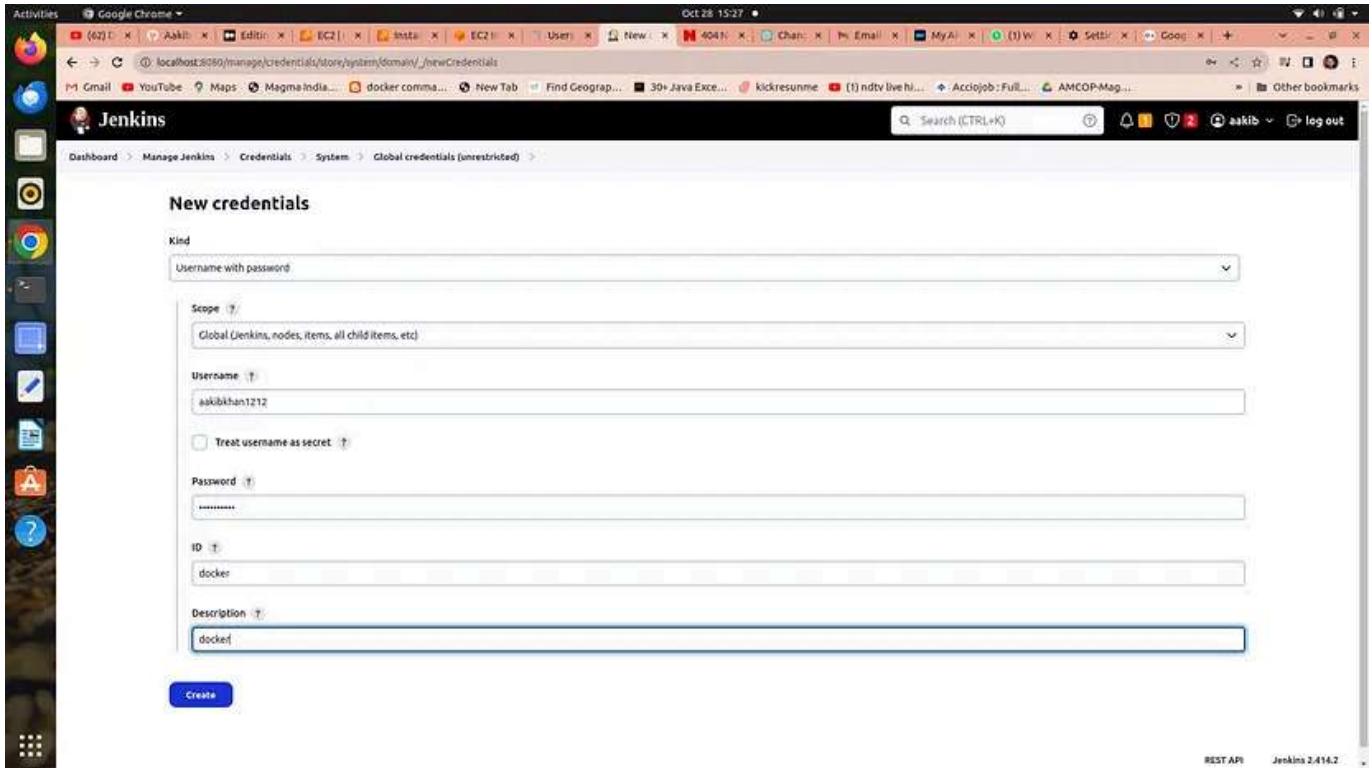
select the os and the following commands used in jenkins pipeline code that set sonarqube to watch over jenkins

Setup docker credentials

1. go to your jenkins → manage jenkins → credentials → global → add credentials

2. provide your username and password of your dockerhub

3. id==docker



Step 3→Now we are going to setup tools for jenkins

go to manage jenkins → tools

a. add jdk

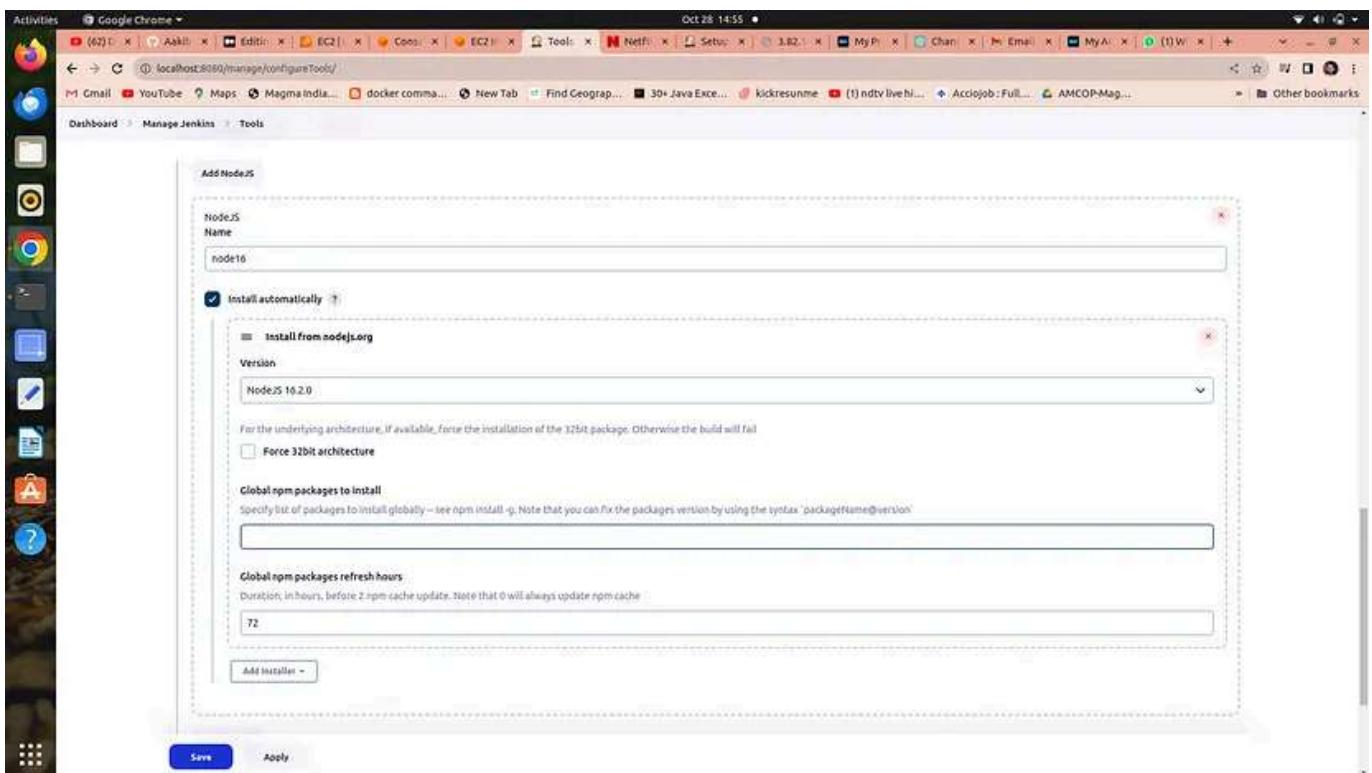
1. click on add jdk and select installer adoptium.net

2. choose jdk 17.0.8.1+1version and in name section enter jdk 17

The screenshot shows the Jenkins 'Manage Jenkins > Tools' configuration page. Under the 'JDK installations' section, there is a configuration window for 'JDK'. The 'Name' field is set to 'jdk17'. The 'Install automatically' checkbox is checked. Under 'Install from adoptium.net', the 'Version' dropdown is set to 'jdk-17.0.8+1-1'. There is also an 'Add Installer' button. Below this, there is a link to 'Add JDK'. Under the 'Git installations' section, there is a configuration window for 'Git'. The 'Name' field is empty. There are 'Save' and 'Apply' buttons at the bottom.

b. add node js

1. click on add nodejs
2. enter node16 in name section
3. choose version nodejs 16.2.0



c. add docker →

1. click on add docker
2. name==docker
3. add installer ==download from docker.com

The screenshot shows the Jenkins 'NodeJS installations' configuration page. Under the 'Docker installations' section, a new Docker instance is being added with the name 'docker'. The 'Install automatically' checkbox is checked. Under 'Download from docker.com', the 'Docker version' is set to 'latest'. There is also an 'Add Installer' button. At the bottom are 'Save' and 'Apply' buttons.

d. add sonarqube →

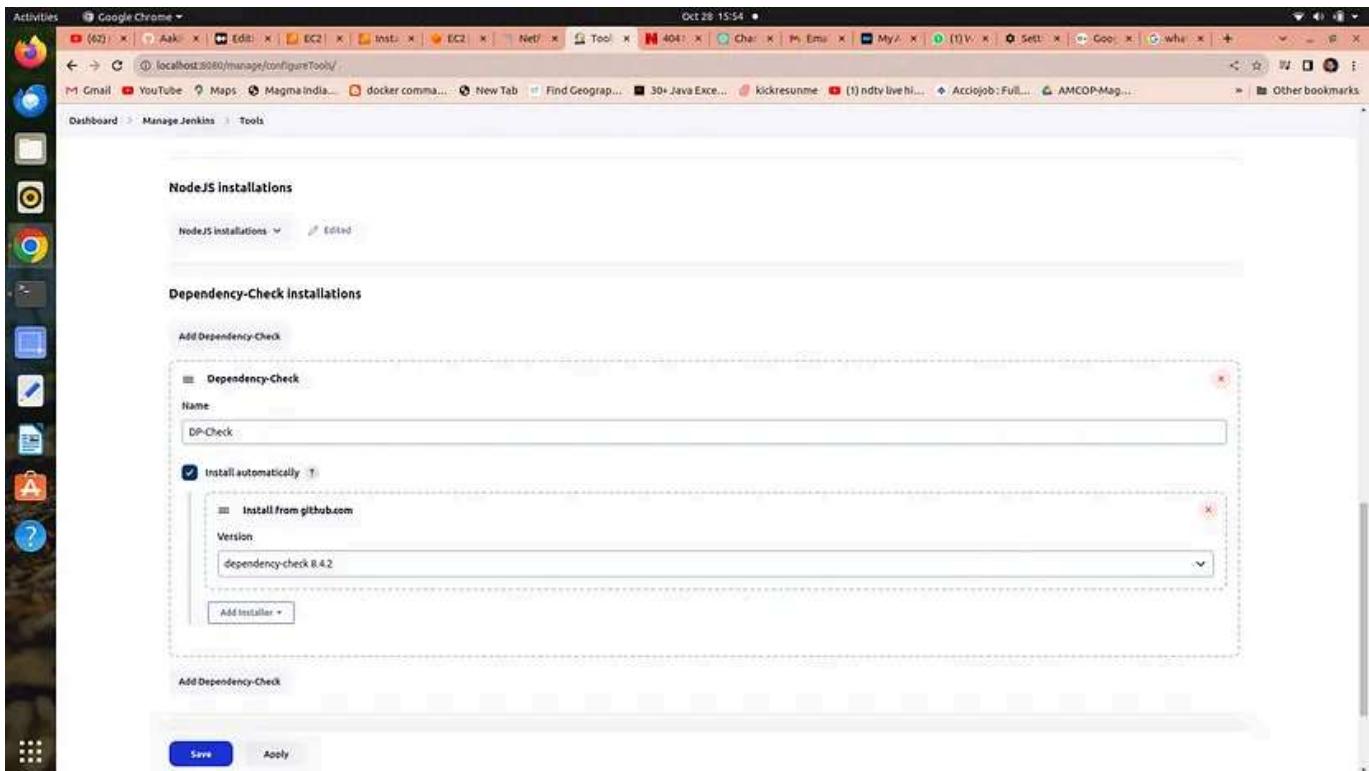
1. add sonar scanner
2. name ==sonar-scanner

The screenshot shows the Jenkins 'SonarQube Scanner installations' configuration page. Under the 'SonarQube Scanner' section, a new scanner is being added with the name 'sonarscanner'. The 'Install automatically' checkbox is checked. Under 'Install from Maven Central', the 'Version' is set to 'SonarQube Scanner 5.0.1.3006'. There is also an 'Add Installer' button. At the bottom are 'Save' and 'Apply' buttons.

e. add owasp dependency check →

Adding the Dependency-Check plugin in the “Tools” section of Jenkins allows you to perform automated security checks on the dependencies used by your application

1. add dependency check
2. name == DP-Check
3. from add installer select install from github.com



Step 4 →Configure global setting for sonarube

1. go to manage jenkins →Configure global setting →add sonarqube servers
2. name ==sonar-server
3. server_url==http://public_ip:9000
4. server authentication token == sonar-token

The screenshot shows a Jenkins interface with a sidebar containing various icons. The main window displays the 'SonarQube servers' configuration page under 'Manage Jenkins > System'. A modal dialog is open, prompting for a 'Name' (set to 'sonar-server'), 'Server URL' (set to 'http://localhost:9000'), and a 'Server authentication token' (set to 'sonar-token'). Below the dialog, there are 'Add SonarQube' and 'Docker Slaves' sections, each with 'Save' and 'Apply' buttons.

Step 5 → let's run the Pipeline →

1. go to new item → select pipeline → in the name section type netflix
2. scroll down to the pipeline script and copy paste the following code

```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
    environment {
        SCANNER_HOME=tool 'sonar-scanner'
    }
    stages {
        stage('clean workspace'){
            steps{
                cleanWs()
            }
        }
        stage('Checkout from Git'){
            steps{
                git branch: 'main', url:
https://github.com/Aakibgithuber/Deploy-Netflix-Clone-on-Kubernetes.git
            }
        }
    }
}
```

```

        }
    }
    stage("Sonarqube Analysis "){
        steps{
            withSonarQubeEnv('sonar-server') {
                sh ''' $SCANNER_HOME/bin/sonar-scanner -
Dsonar.projectName=Netflix \
-Dsonar.projectKey=Netflix '''
            }
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false,
credentialsId: 'Sonar-token'
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
    stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ -- disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check' dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
        }
    }
    stage('TRIVY FS SCAN') {
        steps {
            sh "trivy fs . > trivyfs.txt"
        }
    }
    stage("Docker Build & Push"){
        steps{
            script{
                withDockerRegistry(credentialsId: 'docker',
toolName: 'docker'){
                    sh "docker build --build-arg TMDB_V3_API_KEY=<yourapikey> -t netflix ."
                    sh "docker tag netflix aakibkhan1212/netflix:latest "
                    sh "docker push aakibkhan1212/netflix:latest "
                }
            }
        }
    }
}

```

```

stage("TRIVY"){
    steps{
        sh "trivy image nasi101/netflix:latest > trivyimage.txt"
    }
}
stage('Deploy to container'){
    steps{
        sh 'docker run -d --name netflix -p 8081:80 nasi101/netflix:latest'
    }
}
}

```

If you get docker login failed errorr

```

sudo su
sudo usermod -aG docker jenkins
sudo systemctl restart jenkins

```

3. Your Pipeline is started to build the following docker image and send it to dockerhub with all security checks

The screenshot shows the Jenkins Pipeline Netflix dashboard. On the left, there's a sidebar with various Jenkins management links like Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, Pipeline Syntax, Build History, and Atom feeds. The main area has tabs for Status and Pipeline Netflix. Under Pipeline Netflix, the Stage View is displayed, showing a timeline of stages: Tool Install (94ms), clean workspace (255ms), Checkout from Git (6s), Sonarqube Analysis (16min 45s), quality gate (60ms), Install Dependencies (61ms), OWASP FS SCAN (51ms), TRIVY FS SCAN (129ms), Docker Build & Push (91ms), TRIVY (39ms), and Deploy to container (28ms). The Sonarqube Analysis stage is highlighted in blue. Below the Stage View, the Build History shows two builds: #1 (Oct 28, 2023, 21:49) and #2 (Oct 28, 2023, 21:10). Both builds are shown as failed. At the bottom, Permalinks and a REST API link are provided.

Phase 4 → Monitoring via Prometheus and Grafana

- Prometheus is like a detective that constantly watches your software and gathers data about how it's performing. It's good at collecting metrics, like how fast your software is running or how many users are visiting your website.
- Grafana, on the other hand, is like a dashboard designer. It takes all the data collected by Prometheus and turns it into easy-to-read charts and graphs. This helps you see how well your software is doing at a glance and spot any problems quickly.

In other words, Prometheus collects the information, and Grafana makes it look pretty and understandable so you can make decisions about your software. They're often used together to monitor and manage applications and infrastructure.

Step 1 → Setup another server or EC2 for monitoring

1. go to ec2 console and launch an instance having a base image of Ubuntu and with t2.medium specs because **Minimum Requirements to Install Prometheus :**

- 2 CPU cores.
- 4 GB of memory.
- 20 GB of free disk space.

Step 2 → Installing Prometheus:

1. First, create a dedicated Linux user for Prometheus and download Prometheus:

- a. sudo useradd -s /bin/false -m prometheus

b. wget

<https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz>

2. Extract Prometheus files, move them, and create directories:

a. tar -xvf prometheus-2.47.1.linux-amd64.tar.gz

b. cd prometheus-2.47.1.linux-amd64/

c. sudo mkdir -p /data /etc/prometheus

d. sudo mv prometheus promtool /usr/local/bin/

e. sudo mv consoles/ console_libraries/ /etc/prometheus/

f. sudo mv prometheus.yml /etc/prometheus/prometheus.yml

3. Set ownership for directories:

a. sudo chown -R prometheus:prometheus /etc/prometheus/ /data/

4. Create a systemd unit configuration file for Prometheus:

a. sudo nano /etc/systemd/system/prometheus.service

Add the following code to the `prometheus.service` file:

```

[Unit]
Description=Prometheus
Wants=network-online.target
After=network-online.target

StartLimitIntervalSec=500
StartLimitBurst=5

[Service]
User=prometheus
Group=prometheus
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/prometheus \
--config.file=/etc/prometheus/prometheus.yml \
--storage.tsdb.path=/data \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries \
--web.listen-address=0.0.0.0:9090 \
--web.enable-lifecycle

[Install]
WantedBy=multi-user.target

```

b. press →**ctrl+o** #for save and then **ctrl+x** #for exit from the file

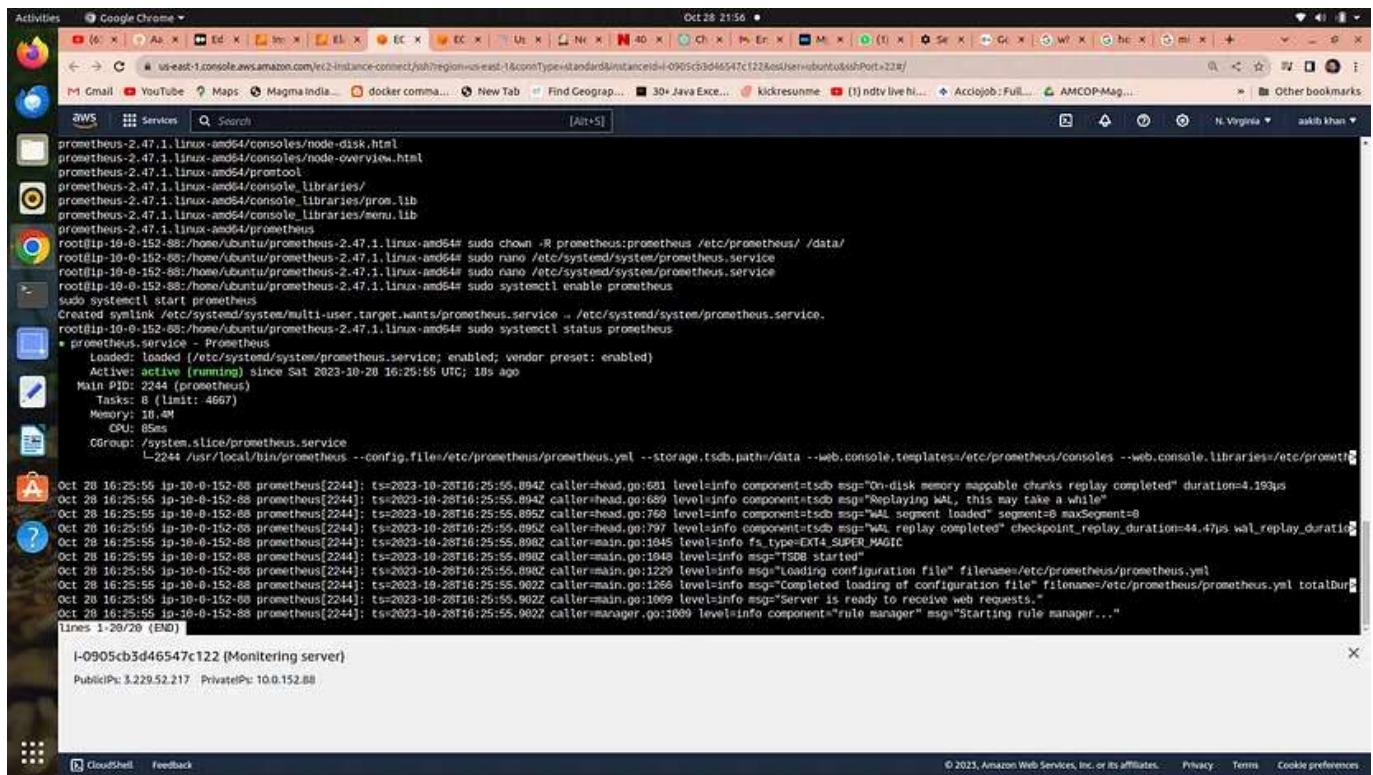
Here's a explanation of the key parts in this above file:

- `User` and `Group` specify the Linux user and group under which Prometheus will run.
- `ExecStart` is where you specify the Prometheus binary path, the location of the configuration file (`prometheus.yml`), the storage directory, and other settings.
- `web.listen-address` configures Prometheus to listen on all network interfaces on port 9090.

- `web.enable-lifecycle` allows for management of Prometheus through API calls.

5. Enable and start Prometheus:

- `sudo systemctl enable prometheus`
- `sudo systemctl start prometheus`
- `sudo systemctl status prometheus`



```

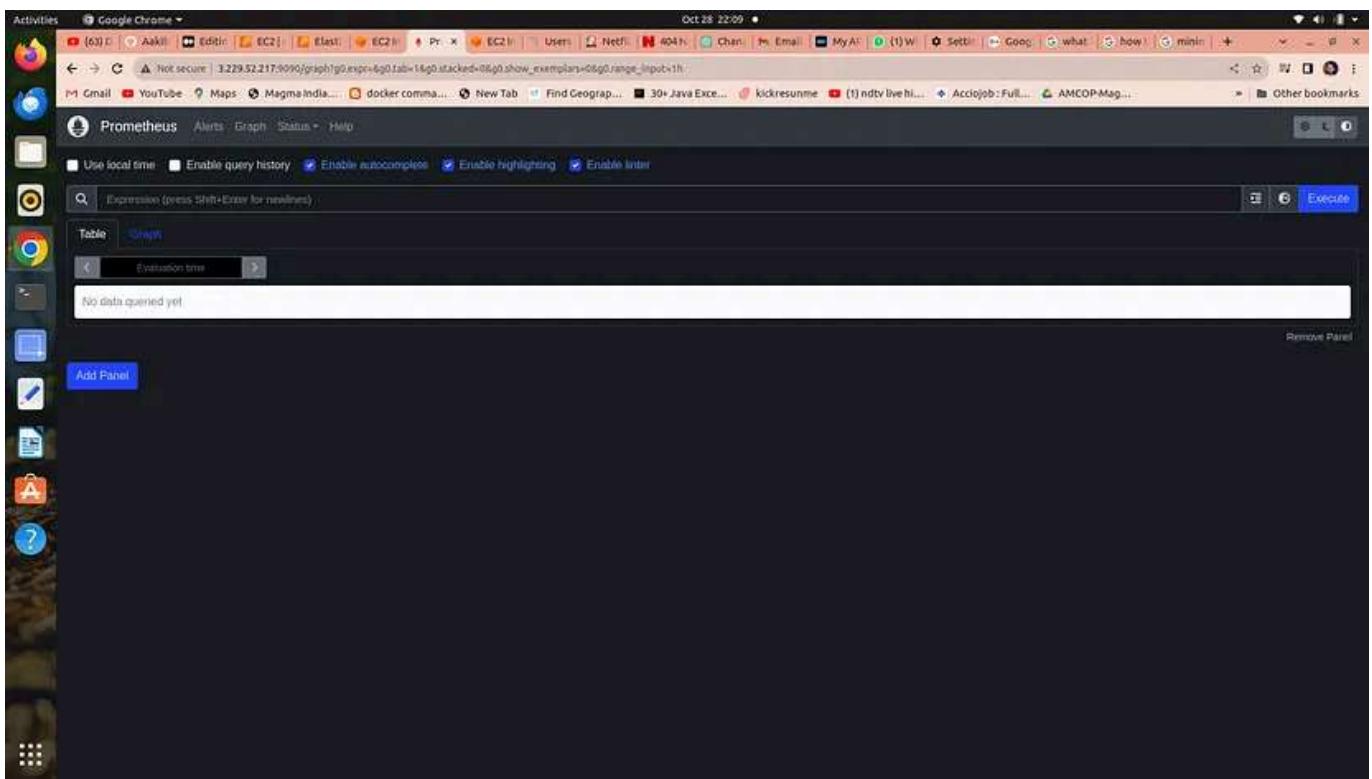
Activities Google Chrome Oct 28 21:56
[Alt+5]
us-east-1.console.aws.amazon.com/vc2-instance-connect/jsh/region=us-east-1&connType=standard&instanceId=i-0905cb3d46547c122&osUser=ubuntu&sshPort=228
Gmail YouTube Maps Magma India... docker comm... New Tab Find Geograph... 3D Java Execut... kickresumne [!] ndtv live Ni... Acciojob: Full... AMCOP-Mag...
AWS Services Search
prometheus-2.47.1.linux-amd64/consoles/node-disk.html
prometheus-2.47.1.linux-amd64/consoles/node-overview.html
prometheus-2.47.1.linux-amd64/promtool
prometheus-2.47.1.linux-amd64/console/libraries/
prometheus-2.47.1.linux-amd64/console/libraries/prom.lib
prometheus-2.47.1.linux-amd64/console/libraries/menu.lib
prometheus-2.47.1.linux-amd64/prometheus
root@ip-10-0-152-88:~# /home/ubuntu/prometheus-2.47.1.linux-amd64 sudo chown -R prometheus:prometheus /etc/prometheus/ /data/
root@ip-10-0-152-88:~# /home/ubuntu/prometheus-2.47.1.linux-amd64 sudo nano /etc/systemd/system/prometheus.service
root@ip-10-0-152-88:~# /home/ubuntu/prometheus-2.47.1.linux-amd64 sudo nano /etc/systemd/system/prometheus.service
root@ip-10-0-152-88:~# /home/ubuntu/prometheus-2.47.1.linux-amd64 sudo systemctl enable prometheus
root@ip-10-0-152-88:~# sudo systemctl start prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
root@ip-10-0-152-88:~# /home/ubuntu/prometheus-2.47.1.linux-amd64 sudo systemctl status prometheus
● prometheus.service - Prometheus
   Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2023-10-28 16:25:55 UTC; 10s ago
       Main PID: 2244 (prometheus)
         Tasks: 8 (limit: 4667)
        Memory: 8.1M
          CPU: 85ms
        CGroup: /system.slice/prometheus.service
           └─ 2244 /usr/local/bin/prometheus --config.file=/etc/prometheus/prometheus.yml --storage.tsdb.path=/data --web.console.templates=/etc/prometheus/consoles --web.console.libraries=/etc/prometheus/libraries

Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.894Z caller=head.go:601 level=info component=tls msg="On-disk memory mappable chunks replay completed" duration=4.193us
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.894Z caller=head.go:689 level=info component=tls msg="Replaying WAL, this may take a while"
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.895Z caller=head.go:766 level=info component=tls msg="WAL segment loaded" segment=0 maxSegment=0
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.896Z caller=head.go:797 level=info component=tls msg="WAL replay completed" checkpoint_replay_duration=44.47µs wal_replay_duration=44.47µs
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.896Z caller=main.go:1645 level=info fs_type=EXT4_SUPER_MAGIC
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.896Z caller=main.go:1229 level=info msg="TSB started"
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.896Z caller=main.go:1229 level=info msg="Loading configuration file" filename=/etc/prometheus/prometheus.yml
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.902Z caller=main.go:1236 level=info msg="Completed loading of configuration file" filename=/etc/prometheus/prometheus.yml totalDuration=1.000000µs
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.902Z caller=manager.go:1009 level=info component="rule manager" msg="Starting rule manager..."
Oct 28 16:25:55 ip-10-0-152-88 prometheus[2244]: ts=2023-10-28T16:25:55.902Z caller=manager.go:1009 level=info component="rule manager" msg="Starting rule manager..."
```

I-0905cb3d46547c122 (Monitoring server)
 PublicIP: 3.229.52.211 PrivateIP: 10.0.152.88

Now go to your security group of your ec2 to enable port 9090 in which prometheus will run

go to → http://public_ip:9090 to see the webpage of prometheus



Step 3 → Installing Node Exporter:

Node exporter is like a “reporter” tool for Prometheus, which helps collect and provide information about a computer (node) so Prometheus can monitor it. It gathers data about things like CPU usage, memory, disk space, and network activity on that computer.

A Node Port Exporter is a specific kind of Node Exporter that is used to collect information about network ports on a computer. It tells Prometheus which network ports are open and what kind of data is going in and out of those ports. This information is useful for monitoring network-related activities and can help you ensure that your applications and services are running smoothly and securely.

Run the following commands for installation

1. Create a system user for Node Exporter and download Node Exporter:

a. sudo useradd – system – no-create-home – shell /bin/false
node_exporter

b . wget

https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz

2. Extract Node Exporter files, move the binary, and clean up:

- a. tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
- b. sudo mv node_exporter-1.6.1.linux-amd64/node_exporter /usr/local/bin/
- c. rm -rf node_exporter*

3. Create a systemd unit configuration file for Node Exporter:

- a. sudo nano /etc/systemd/system/node_exporter.service

add the following code to the `node_exporter.service` file:

provide more detailed information about what might be going wrong. For example:

```
[Unit]
Description=Node Exporter
Wants=network-online.target
After=network-online.target
```

```
StartLimitIntervalSec=500
StartLimitBurst=5
```

```
[Service]
User=node_exporter
Group=node_exporter
```

```
Type=simple
Restart=on-failure
RestartSec=5s
ExecStart=/usr/local/bin/node_exporter --collector.logind

[Install]
WantedBy=multi-user.target
```

4. Enable and start Node Exporter:

- a. sudo systemctl enable node_exporter
- b. sudo systemctl start node_exporter
- c. sudo systemctl status node_exporter

node exporter service is now running

```
root@ip-10-0-152-88:/home/ubuntu/prometheus-2.47.1.linux-amd64$ tar -xvf node_exporter-1.6.1.linux-amd64.tar.gz
root@ip-10-0-152-88:/home/ubuntu/prometheus-2.47.1.linux-amd64$ sudo mv node_exporter-1.6.1.linux-amd64/node_exporter /usr/local/bin/
root@ip-10-0-152-88:~$ rm -rf node_exporter*
root@ip-10-0-152-88:~$ node_exporter-1.6.1.linux-amd64/NOTICE
root@ip-10-0-152-88:~$ node_exporter-1.6.1.linux-amd64/node_exporter
root@ip-10-0-152-88:~$ node_exporter-1.6.1.linux-amd64/LICENSE
root@ip-10-0-152-88:~$ node_exporter-1.6.1.linux-amd64$ sudo nano /etc/systemd/system/node_exporter.service
root@ip-10-0-152-88:~$ sudo nano /etc/systemd/system/node_exporter.service
root@ip-10-0-152-88:~$ sudo systemctl enable node_exporter
root@ip-10-0-152-88:~$ sudo systemctl start node_exporter
root@ip-10-0-152-88:~$ sudo ln -s /etc/systemd/system/multi-user.target.wants/node_exporter.service /etc/systemd/system/node_exporter.service
root@ip-10-0-152-88:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2023-10-28 16:52:42 UTC; 12s ago
     Main PID: 2314 (node_exporter)
       Tasks: 4 (limit: 4667)
      Memory: 2.1M
        CPU: 7ms
       CGroup: /system.slice/node_exporter.service
           └─ 2314 /usr/local/bin/node_exporter --collector.logind

Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.739Z caller=node_exporter.go:117 level=info collector=thermal_zone
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=time
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=timex
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=udp_queues
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=ups
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=volume
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=xfs
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=node_exporter.go:117 level=info collector=zfs
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=tls_config.go:274 level=info msg="Listening on" address=[::]:9100
Oct 28 16:52:42 ip-10-0-152-88 node_exporter[2314]: ts=2023-10-28T16:52:42.740Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9100
root@ip-10-0-152-88:/home/ubuntu/prometheus-2.47.1.linux-amd64$
```

I-0905cb3d46547c122 (Monitoring server)
PublicIP: 5.229.52.217 PrivateIP: 10.0.152.88

You can access Node Exporter metrics in Prometheus.

Step 4→Configure Prometheus Plugin Integration:

1. go to your EC2 and run →cd /etc/prometheus

2. you have to edit the prometheus.yml file to moniter anything

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'node_exporter'
    static_configs:
      - targets: ['localhost:9100']

  - job_name: 'jenkins'
    metrics_path: '/prometheus'
    static_configs:
      - targets: ['<your-jenkins-ip>:<your-jenkins-port>']
```

add the above code with proper indentation like this →

```
evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
# scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alertmanagers:
  - static_configs:
    - targets:
      # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label 'job=<job_name>' to any timeseries scraped from this config.
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ['localhost:9090']
        - job_name: 'node_exporter'
          static_configs:
            - targets: ['localhost:9100']
              - job_name: 'jenkins'
                metrics_path: '/prometheus'
                static_configs:
                  - targets: ['54.82.163.38:8880']
```

press esc+**wq** to save and exit

a. Check the validity of the configuration file →

```
promtool check config /etc/prometheus/prometheus.yml
```

o/p →success

b. Reload the Prometheus configuration without restarting →

```
curl -X POST http://localhost:9090/-/reload
```

go to your prometheus tab again and click on status and select targets you will there is three targets present as we enter in yaml file for monitoring

The screenshot shows the Prometheus Targets dashboard. It lists three targets:

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://54.17.183.33:8083/metrics	DOWN	instance="54.17.183.33:8083" job="jenkins"	1.475s ago	3.162ms	series returned HTTP status: 503 Forbidden
http://54.17.183.33:9100/metrics	UP	instance="54.17.183.33:9100" job="node exporter"	11.629s ago	15.535ms	
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	767.000ms ago	5.005ms	

prometheus targets dashboard

Step 5 → Setup Grafana →

Install Dependencies:

- a. sudo apt-get update
- b. sudo apt-get install -y apt-transport-https software-properties-common

Add the GPG Key for Grafana:

- a. wget -q -O <https://packages.grafana.com/gpg.key> | sudo apt-key add -

Add the repository for Grafana stable releases:

- a. echo "deb <https://packages.grafana.com/oss/deb> stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

Update the package list , install and start Grafana:

- a. sudo apt-get update
- b. sudo apt-get -y install grafana
- c. sudo systemctl enable grafana-server
- d. sudo systemctl start grafana-server
- e. sudo systemctl status grafana-server

Activities Google Chrome Oct 29 18:10 •

① us-east-1.console.aws.amazon.com/ec2/instance-connect/jsh?region=us-east-1&connType=standard&instanceId=i-0905cb3d46547c122&user=bonito&sshPort=22/

Gmail YouTube Maps Magma India docker comm... New Tab Find Geograph... 30+ Java Exec... kickresume (1) ndtv live hi... Accijob:Full... AMCOP-Mag... Magma India All Bookmarks

aWS Services Search [Alt+S]

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.

```
root@ip-10-0-152-88:~# /home/ubuntu sudo systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /lib/systemd/system/grafana-server.service.
root@ip-10-0-152-88:~# home/ubuntu sudo systemctl start grafana-server
root@ip-10-0-152-88:~# home/ubuntu sudo systemctl status grafana-server
● grafana-server.service - Grafana instance
  Loaded: loaded (/lib/systemd/system/grafana-server.service; enabled; vendor preset: enabled)
  Active: active (running) since Sun 2023-10-29 12:40:21 UTC; 12s ago
    Docs: http://docs.grafana.org
  Main PID: 3474 (grafana)
     Tasks: 11 (limit: 4667)
       Memory: 95.0M
          CPU: 5.657s
         CGroup: /system.slice/grafana-server.service
                 └─ 3474 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana.pid --packaging=deb cfg=default.paths.logs=/var/log/grafana cfg=default
```

Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_migration t=2023-10-29T12:40:30.50634104Z level=info msg="Starting legacy migration"
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_migration orgID=1 t=2023-10-29T12:40:30.50835693Z level=info msg="Migrating alerts for organisation"
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_migration orgID=1 t=2023-10-29T12:40:38.595333261Z level=info msg="Alerts found to migrate" alerts=8
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_migration orgID=1 t=2023-10-29T12:40:38.59602563Z level=warn msg="No available receivers"
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_state_manager t=2023-10-29T12:40:38.511797237Z level=info msg="State cache has been initialized" states=8 duration=81.342ms
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_scheduler t=2023-10-29T12:40:39.512288544Z level=info msg="Starting scheduler" tickInterval=10s
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=ticker t=2023-10-29T12:40:39.512469207Z level=info msg="Starting first tick" 2023-10-29T12:40:40Z
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_multiorg.alertmanager t=2023-10-29T12:40:39.51260237Z level=info msg="Starting MultiOrg Alertmanager"
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=galert_migration t=2023-10-29T12:40:39.514136477Z level=info msg="Completed legacy migration"
Oct 29 12:40:30 ip-10-0-152-88 grafana[3474]: logger=plugins.update.checker t=2023-10-29T12:40:39.527837754Z level=info msg="Update check succeeded" duration=76.16678ms
Lines 1-21/21 (END)

I-0905cb3d46547c122 (Monitoring server)

PublicIP: 3.229.52.217 PrivateIP: 10.0.152.88

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Now go to your ec2 security group and open port no. 3000 in which grafana runs

Activities Google Chrome Oct 29 17:59 •

① us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1&ModifyingboundSecurityGroupRulessecurityGroupID=sgr-03e3d5c2984fa003

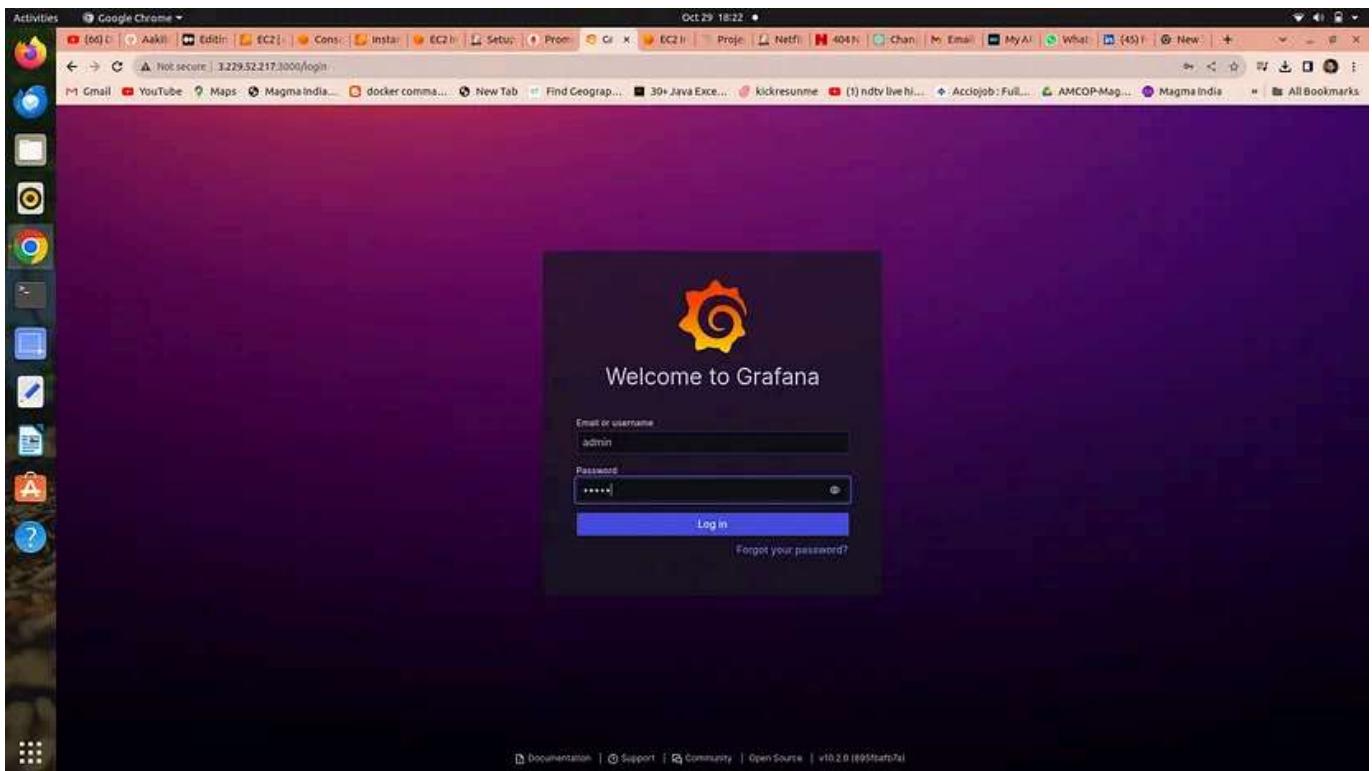
Gmail YouTube Maps Magma India docker comm... New Tab Find Geograph... 30+ Java Exec... kickresume (1) ndtv live hi... Accijob:Full... AMCOP-Mag... Magma India All Bookmarks

aWS Services Search [Alt+S]

Security group rule ID	Type info	Protocol info	Port range info	Source info	Description - optional info
sgr-062c12271fb924630	Custom TCP	TCP	9100	Custom	node exporter
sgr-0f68c2a94956d1a5d	Custom TCP	TCP	8080	Custom	jenkins
sgr-0fcbd89be2def3e63	HTTP	TCP	80	Custom	
sgr-02757b75084bab071	Custom TCP	TCP	9090	Custom	prometheus
sgr-0a6805a82bbcd916d	SSH	TCP	22	Custom	prometheus
sgr-0fb4f5f2fdc1fb52e	HTTPS	TCP	443	Custom	
-	Custom TCP	TCP	3000	Anywhere...	grafana

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Go and browse http://public_ip:3000 to access your grafana web interface



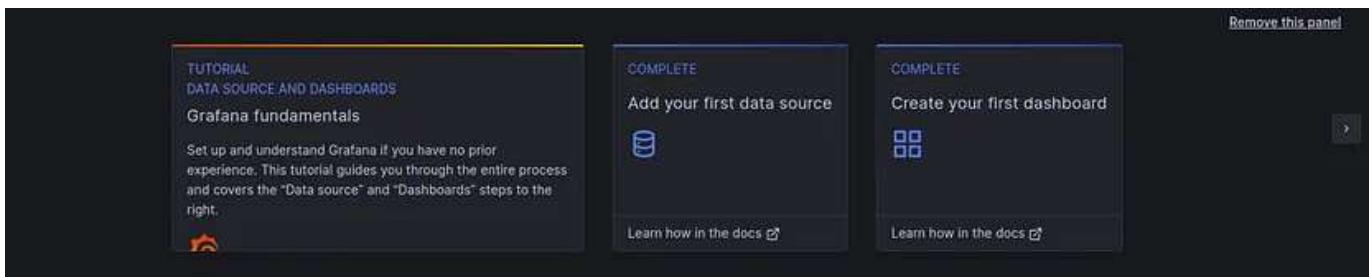
username = admin, password =admin

Step 6 → Add Prometheus Data Source:

To visualize metrics, you need to add a data source.

Follow these steps:

- Click on the gear icon (⚙️) in the left sidebar to open the “Configuration” menu.
- Select “Data Sources.”
- Click on the “Add data source” button.

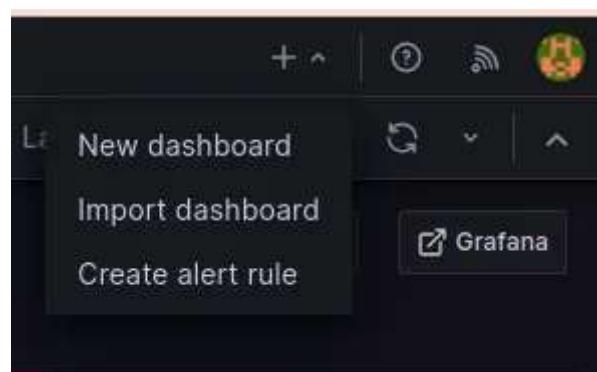


- Choose “Prometheus” as the data source type.
- In the “HTTP” section:
- Set the “URL” to `http://localhost:9090` (assuming Prometheus is running on the same server).
- Click the “Save & Test” button to ensure the data source is working.

Step 7 → Import a Dashboard

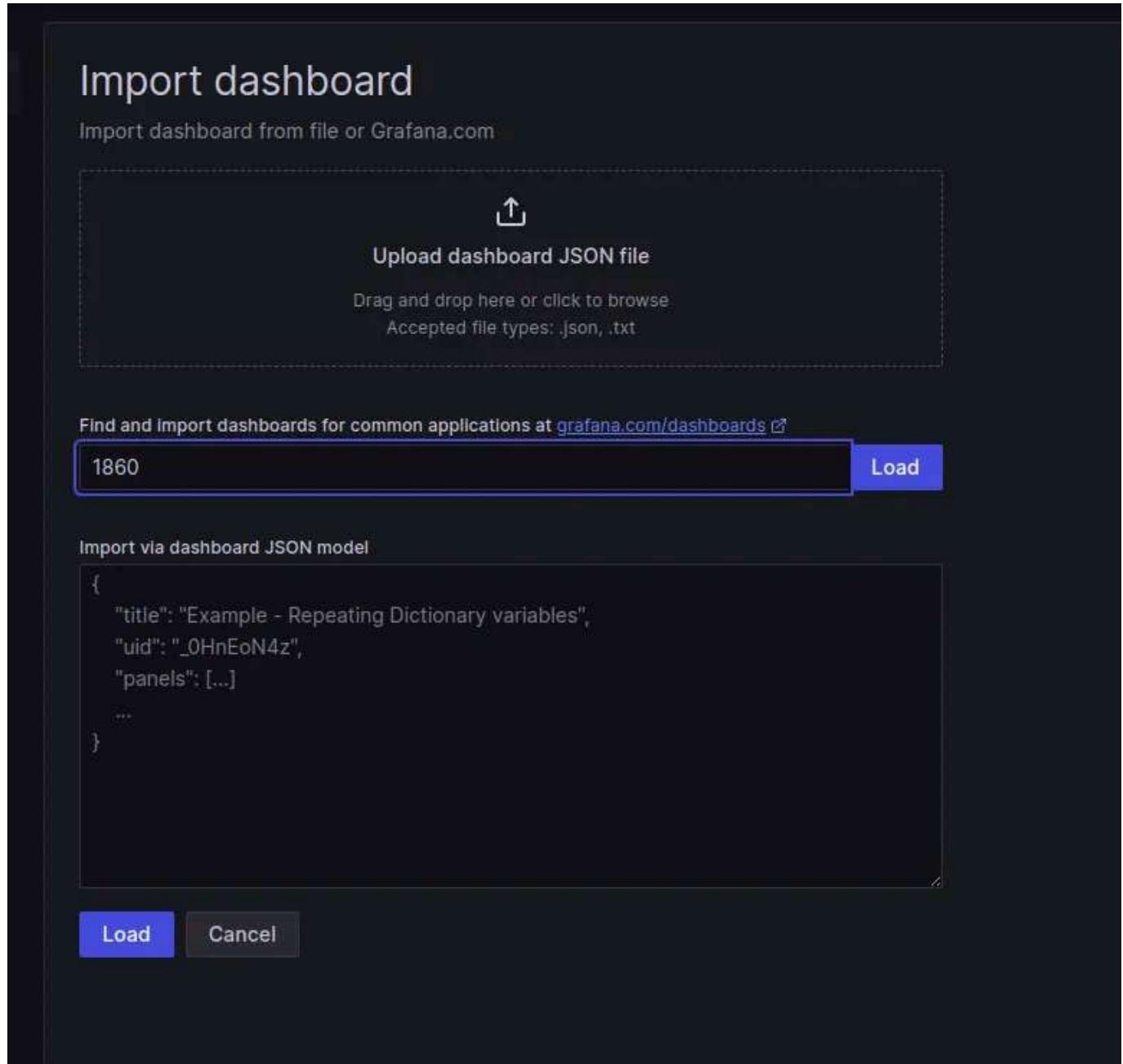
Importing a dashboard in Grafana is like using a ready-made template to quickly create a set of charts and graphs for monitoring your data, without having to build them from scratch.

- Click on the “+” (plus) icon in the left sidebar to open the “Create” menu.

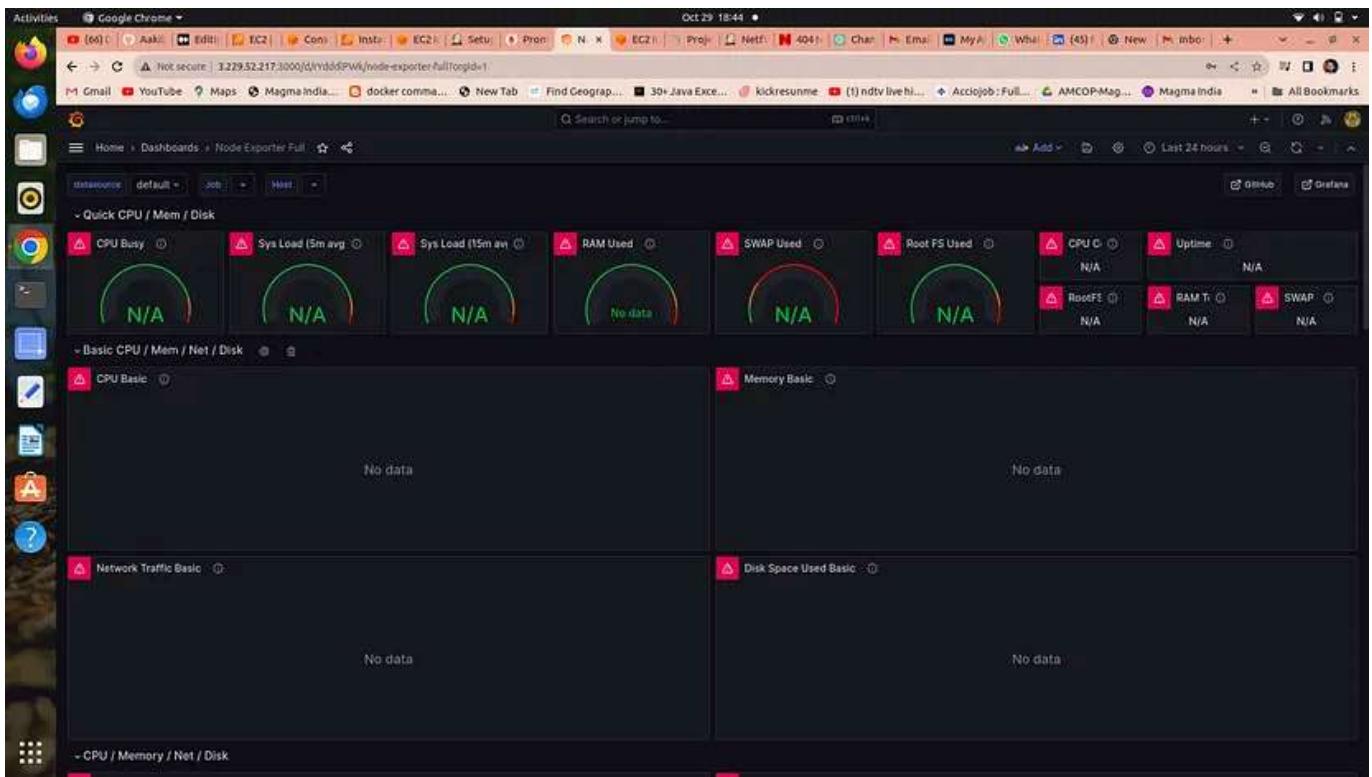


- Select “Dashboard.”
- Click on the “Import” dashboard option.

- Enter the dashboard code you want to import (e.g., code 1860).
- Click the “Load” button.



- Select the data source you added (Prometheus) from the dropdown.
- Click on the “Import” button.

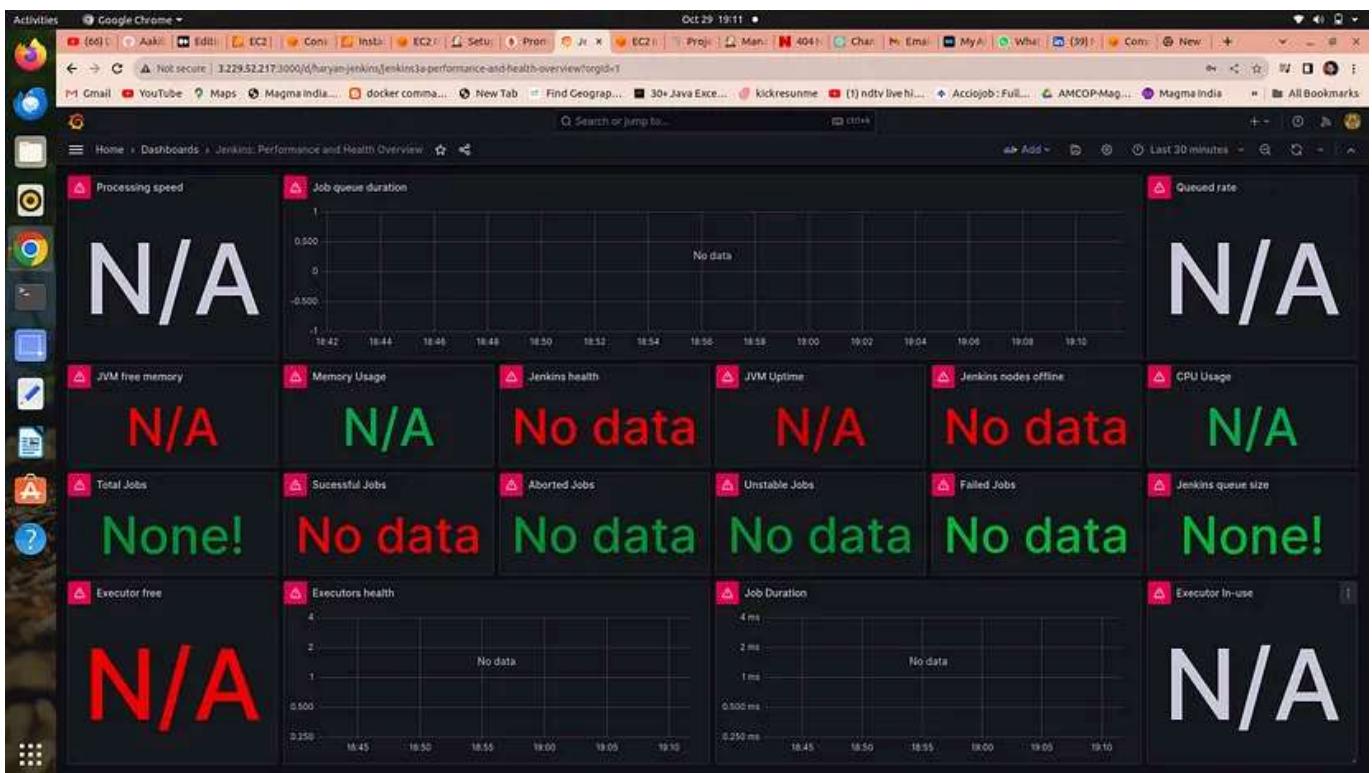


Step 7 → Configure global setting for prometheus

go to manage jenkins →system →search for prometheus — apply →save

Step 8→ import a dashboard for jenkins

- Click on the “+” (plus) icon in the left sidebar to open the “Create” menu.
- Select “Dashboard.”
- Click on the “Import” dashboard option.
- Enter the dashboard code you want to import (e.g., code 9964).
- Click the “Load” button.
- Select the data source you added (Prometheus) from the dropdown.



Phase 5: Kubernetes →

Step 1 → Setup EKS(Elastic Kubernetes Service) on aws

1. go to console and search for EKS

2. click on add a cluster



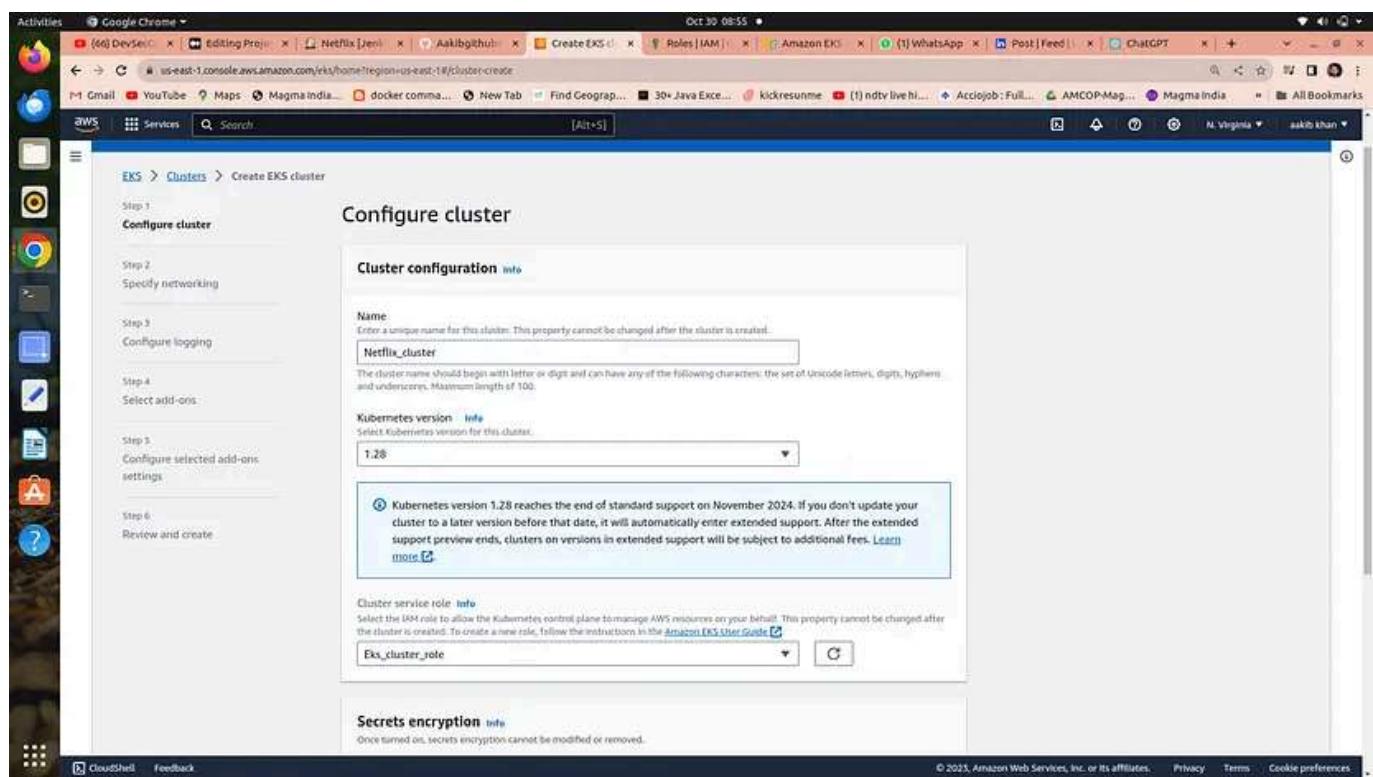
click on create

3. add a name to your cluster and choose a service role if you don't have then follow below documentation to create it

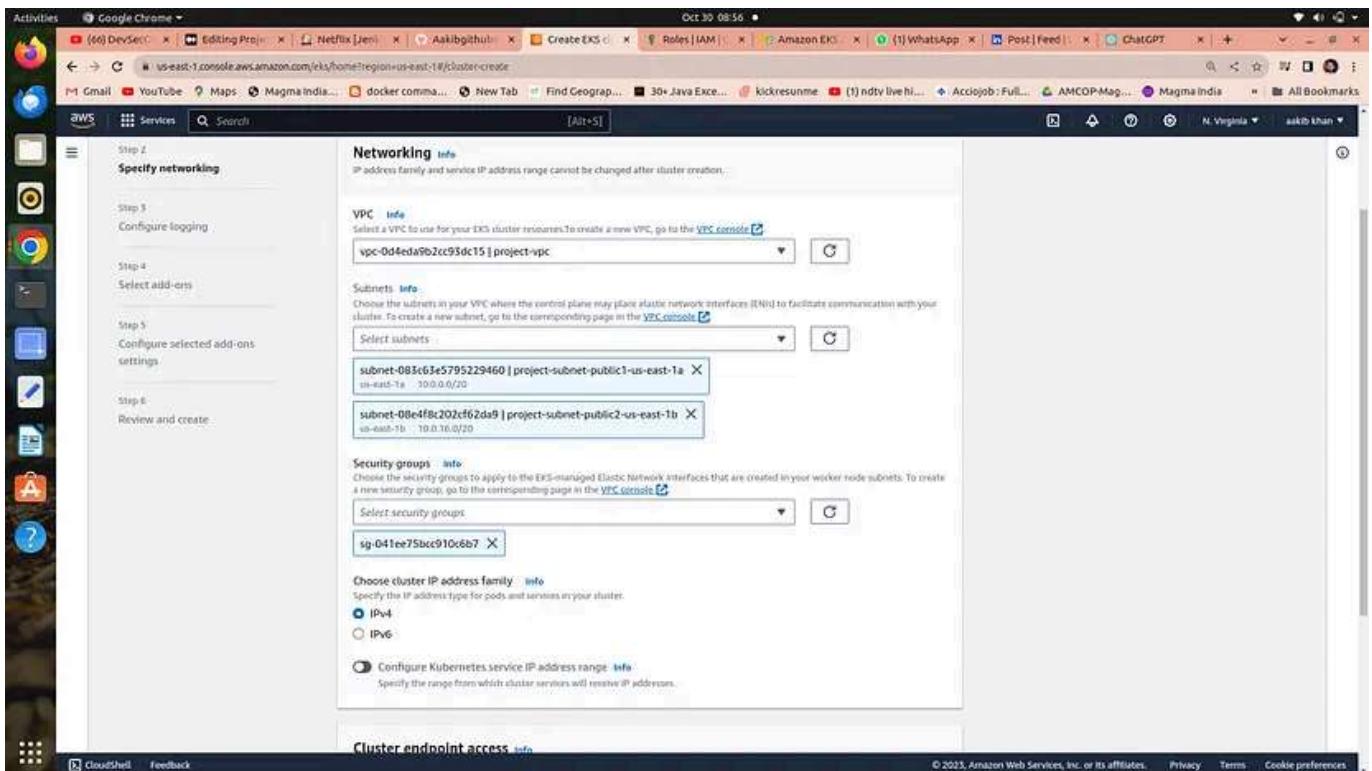
Amazon EKS cluster IAM role

The Amazon EKS cluster IAM role is required for each cluster. Kubernetes clusters managed by Amazon EKS use this role...

docs.aws.amazon.com



4. click on next and choose the default vpc and in subnet option make sure to remove all the private subnet and remain everything as it is



5. click on next →next →create

6. when you cluster is ready then go to compute option and add a node group

The screenshot shows the AWS EKS Cluster Compute tab. The cluster is named 'netflix_cluster' and is in an 'Active' state. The Kubernetes version is 1.28, and support ends in November 2024. The provider is EKS. The Compute tab is selected, showing 'Nodes (0)' and 'Node groups (0)'. Both sections indicate that the cluster does not have any nodes or node groups.

7. choose the below options

A node group is a group of EC2 instances that supply compute capacity to your Amazon EKS cluster. You can add multiple node groups to your cluster.

Node group configuration

These properties cannot be changed after the node group is created.

Name
Assign a unique name for this node group.

The node group name should begin with letter or digit and can have any of the following characters: the set of Unicode letters, digits, hyphens and underscores. Maximum length of 63.

Node IAM role [Info](#)
Select the IAM role that will be used by the nodes. To create a new role, go to the [IAM console](#).
 [Edit](#) [Delete](#)

Important The selected role must not be used by a self-managed node group as this could lead to a service interruption upon managed node group deletion.
[Learn more](#)

Set compute and scaling configuration

Node group compute configuration

These properties cannot be changed after the node group is created.

AMI type [Info](#)

Select the EKS-optimized Amazon Machine Image for nodes.

Amazon Linux 2 (AL2_x86_64)



Capacity type

Select the capacity purchase option for this node group.

On-Demand



Instance types [Info](#)

Select instance types you prefer for this node group.

Select



t3.medium

vCPU: 2 vCPUs Memory: 4 GiB Network: Up to 5 Gigabit Max ENI: 3 Max IPs: 18



Disk size

Select the size of the attached EBS volume for each node.

20

GiB

Remain everything as it is → click on create

The screenshot shows the AWS EKS Node Group configuration page. On the left, there's a sidebar with 'Clusters' and 'Related services' like Amazon ECR and AWS Batch. The main area has tabs for 'Details', 'Nodes', 'Health issues', 'Kubernetes labels', 'Update config', and 'Kubernetes taints'. The 'Details' tab is selected. It displays the following information:

AMI release version	Info	Instance types	Disk size
1.28.2-20231027		t3.medium	20 GiB

Below this, under the 'Details' section, are several rows of node group configuration:

Node group ARN	Autoscaling group name	Capacity type	Subnets
arn:aws:eks:us-east-1:731234214840:nodegroup/Netflix_clus/Netnode/3ec5c775-fc75-fcfc-8327-5680-fb72597456dd	eks-Netnode-3ec5c775-fc75-fcfc-8327-5680-fb72597456dd	On-Demand	subnet-083c65a5795229460 subnet-08e4fbfc202cf62da9
Created	Node IAM role ARN	Desired size	Configure remote access to nodes
2 hours ago	arn:aws:iam::731234214840:role/ec2_service_role	1 node	off
		Minimum size	
		1 node	
		Maximum size	
		1 node	

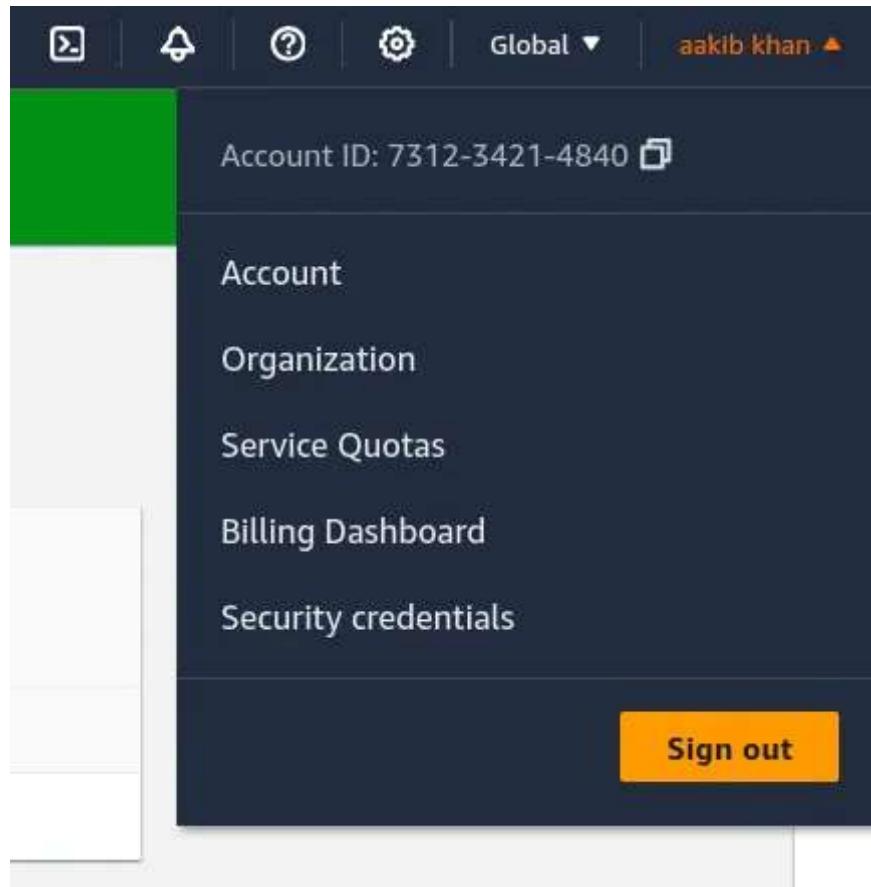
Step 2 → Installing aws -cli to control the cluster from our terminal

Now go to your terminal and run

1. pip install awscli
2. aws configure
3. enter a access key and then secret access key

How to create access key

1. go to your console click on your username and then select Security Credentials option



2. scroll down to access keys and click on create

The screenshot shows the AWS Identity and Access Management (IAM) console. On the left, there's a sidebar with various navigation options like Dashboard, Access management, Access reports, and Related consoles. The main area displays a user named 'aakib khan' with the following details:

- Account name: aakib khan
- Email address: dehiluniversityyy@gmail.com
- AWS account ID: 731234214840
- Canonical user ID: c8e1e815e1838237df49fac2ace12abeb4fb36d0352b67df586f253be508d978

Under 'Multi-factor authentication (MFA)', there is one entry:

Device type	Identifier	Certifications	Created on
Virtual	arn:aws:iam::731234214840:mfa/narzo60	Not Applicable	13 days ago

Under 'Access keys', there are no active keys listed:

Access key ID	Created on	Access key last used	Region last used	Service last used	Status
No access keys					

At the bottom, there are buttons for 'Create access key' and 'Create access key'.

copy and paste it to terminal

```
aakib@aakib-HP-Laptop-15s-fq2xxx:~$ sudo su
[sudo] password for aakib:
root@aakib-HP-Laptop-15s-fq2xxx:/home/aakib# mkdir eks
root@aakib-HP-Laptop-15s-fq2xxx:/home/aakib# cd eks
root@aakib-HP-Laptop-15s-fq2xxx:/home/aakib/eks# aws eks update-kubeconfig --name Netflix --region us-east-1

An error occurred (UnrecognizedClientException) when calling the DescribeCluster
operation: The security token included in the request is invalid.
root@aakib-HP-Laptop-15s-fq2xxx:/home/aakib/eks# aws configure
AWS Access Key ID [*****GIP5]: AKIA2UQHUU04M2DVOBV4
AWS Secret Access Key [*****0mbb]: /uif3J236sr/dDoS26G4/fhieiwflyx9LZ
ovPNgM
```

Run the following command

```
1. aws eks update-kubeconfig — name YourClusterName — region us-east-1
```

This command is used to update the Kubernetes configuration (kubeconfig) file for an Amazon Elastic Kubernetes Service (EKS) cluster named "Netflix_cluster" in the "us-east-" region, allowing you to interact with the cluster using `kubectl`.

Step 3→Install helm →

1. curl <https://baltocdn.com/helm/signing.asc> | gpg — dearmor | sudo tee /usr/share/keyrings/helm.gpg > /dev/null
2. sudo apt-get install apt-transport-https — yes
3. echo “deb [arch=\$(dpkg — print-architecture) signed-by=/usr/share/keyrings/helm.gpg] <https://baltocdn.com/helm/stable/debian/> all main” | sudo tee /etc/apt/sources.list.d/helm-stable-debian.list
4. sudo apt-get install helm

Step →4 Install Node Exporter using Helm

To begin monitoring your Kubernetes cluster, you'll install the Prometheus Node Exporter by Using Helm to install Node Exporter in Kubernetes makes it easy to set up and manage the tool for monitoring your servers by providing a pre-packaged, customizable, and consistent way to do it.

1. Add the Prometheus Community Helm repository:

- helm repo add prometheus-community <https://prometheus-community.github.io/helm-charts>

1. Create a Kubernetes namespace for the Node Exporter:

- kubectl create namespace prometheus-node-exporter

1. Install the Node Exporter using Helm:

- helm install prometheus-node-exporter prometheus-community/prometheus-node-exporter — namespace prometheus-node-exporter

```
root@ramalib-Veriton-M430:/home/ramalib#
root@ramalib-Veriton-M430:/home/ramalib# helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
"prometheus-community" has been added to your repositories
root@ramalib-Veriton-M430:/home/ramalib# kubectl create namespace prometheus-node-exporter
namespace/prometheus-node-exporter created
root@ramalib-Veriton-M430:/home/ramalib# helm install prometheus-node-exporter prometheus-community/prometheus-node-exporter --namespace prometheus-node-exporter
NAME: prometheus-node-exporter
LAST DEPLOYED: Thu Nov 2 13:27:30 2023
NAMESPACE: prometheus-node-exporter
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace prometheus-node-exporter -l "app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/instance=prometheus-node-exporter" -o jsonpath="{.items[0].metadata.name}")
  echo "Visit http://127.0.0.1:9100 to use your application"
  kubectl port-forward --namespace prometheus-node-exporter $POD_NAME 9100
root@ramalib-Veriton-M430:/home/ramalib#
```

1. kubectl get pods -n prometheus-node-exporter → run this to verify

```
root@ramalib-Veriton-M430:/home/ramalib# kubectl get pods -n prometheus-node-exporter
NAME                  READY   STATUS    RESTARTS   AGE
prometheus-node-exporter-gm86l  1/1     Running   0          5m14s
root@ramalib-Veriton-M430:/home/ramalib#
```

Add a Job to Scrape Metrics on nodeip:9001/metrics in prometheus.yml:

Update your Prometheus configuration (prometheus.yml) to add a new job for scraping metrics from nodeip:9001/metrics. You can do this by adding

the following configuration to your prometheus.yml file:

```
- job_name: 'Netflix'  
  metrics_path: '/metrics'  
  static_configs:  
    - targets: ['node1Ip:9100']
```

Replace ‘your-job-name’ with a descriptive name for your job. The static_configs section specifies the targets to scrape metrics from, and in this case, it’s set to nodeip:9001.

Don’t forget to reload or restart Prometheus to apply these changes to your configDeploy Application with ArgoCDon.

Step 5→Install Argo cd

Argo CD is a tool that helps software developers and teams manage and deploy their applications to Kubernetes clusters more easily. It simplifies the process of keeping your applications up to date and in sync with your desired configuration by automatically syncing your code with what’s running in your Kubernetes environment. It’s like a traffic cop for your chmod 700 get_helm.shapplications on Kubernetes, ensuring they are always in the right state without you having to manually make changes.

1. Add the Argo CD Helm repository:

- helm repo add argo-cd <https://argoproj.github.io/argo-helm>

2. Update your Helm repositories:

- helm repo update

3. Create a namespace for Argo CD (optional but recommended):

- kubectl create namespace argocd

4. Install Argo CD using Helm:

- helm install argocd argo-cd/argo-cd -n argocd

5. kubectl get all -n argocd

```

root@ramalib-Veriton-M430:/home/ramalib# kubectl get all -n argocd
  NAME                                         READY   STATUS    RESTARTS   AGE
pod/argocd-application-controller-0           1/1     Running   0          3m55s
pod/argocd-applicationset-controller-564df7f665-wbjrs 1/1     Running   0          3m56s
pod/argocd-dex-server-9d6df8f67-zgdbs        1/1     Running   0          3m56s
pod/argocd-notifications-controller-778c7c56dc-844tb 1/1     Running   0          3m57s
pod/argocd-redis-89496b9df-jjt29            1/1     Running   0          3m57s
pod/argocd-repo-server-56fb75c6f5-zz4g9       1/1     Running   0          3m57s
pod/argocd-server-8694dcfd4-m8tws            1/1     Running   0          3m56s

  NAME                           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/argocd-applicationset-controller   ClusterIP  172.20.36.241 <none>        7000/TCP   3m59s
service/argocd-dex-server                 ClusterIP  172.20.115.230 <none>        5556/TCP,5557/TCP   3m59s
service/argocd-redis                     ClusterIP  172.20.107.246 <none>        6379/TCP   3m59s
service/argocd-repo-server                ClusterIP  172.20.47.91  <none>        8081/TCP   3m59s
service/argocd-server                    ClusterIP  172.20.213.207 <none>        80/TCP,443/TCP   3m59s

  NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/argocd-applicationset-controller 1/1     1           1           3m59s
deployment.apps/argocd-dex-server             1/1     1           1           3m59s
deployment.apps/argocd-notifications-controller 1/1     1           1           4m
deployment.apps/argocd-redis                 1/1     1           1           4m
deployment.apps/argocd-repo-server            1/1     1           1           4m
deployment.apps/argocd-server                1/1     1           1           3m59s

  NAME                           DESIRED   CURRENT   READY   AGE
replicaset.apps/argocd-applicationset-controller-564df7f665 1         1         1         4m
replicaset.apps/argocd-dex-server-9d6df8f67        1         1         1         4m
replicaset.apps/argocd-notifications-controller-778c7c56dc 1         1         1         4m1s
replicaset.apps/argocd-redis-89496b9df          1         1         1         4m1s
replicaset.apps/argocd-repo-server-56fb75c6f5       1         1         1         4m1s
replicaset.apps/argocd-server-8694dcfd4          1         1         1         4m

  NAME                               READY   AGE
statefulset.apps/argocd-application-controller   1/1     4m

```

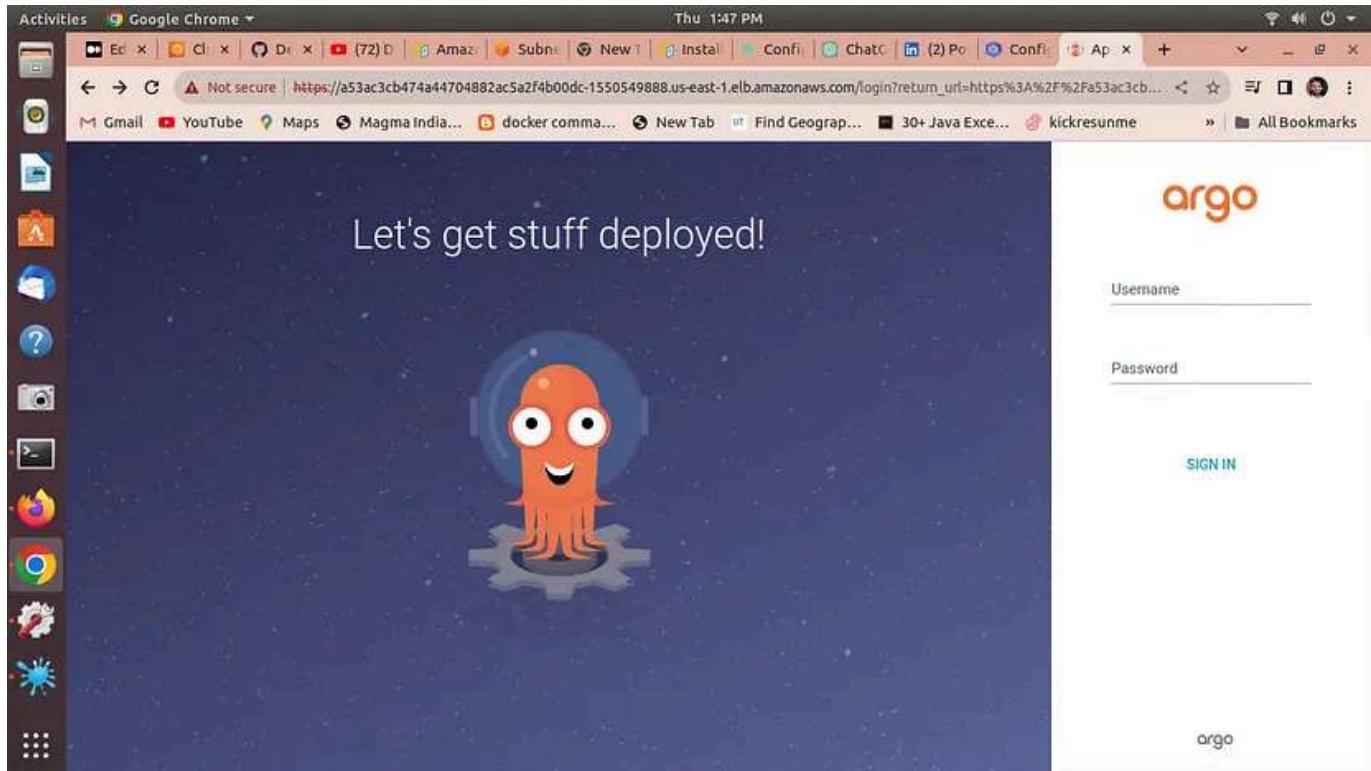
Expose argocd-server

By default argocd-server is not publicaly exposed. For the purpose of this workshop, we will use a Load Balancer to make it usable:

1. `kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "LoadBalancer"}}'`
2. `export ARGOCD_SERVER=`kubectl get svc argocd-server -n argocd -o json | jq --raw-output '.status.loadBalancer.ingress[0].hostname'``
3. `echo $ARGOCD_SERVER`

```
root@ramalib-Veriton-M430:/home/ramalib# export ARGOCD_SERVER=`kubectl get svc argocd-server -n argocd -o json | jq --raw-output '.status.loadBalancer.ingress[0].hostname'`  
root@ramalib-Veriton-M430:/home/ramalib# echo $ARGOCD_SERVER  
a53ac3cb474a44704882ac5a2f4b00dc-1550549888.us-east-1.elb.amazonaws.com
```

you have given a url copy and paste it on your chrome browser



username ==admin

For Password →

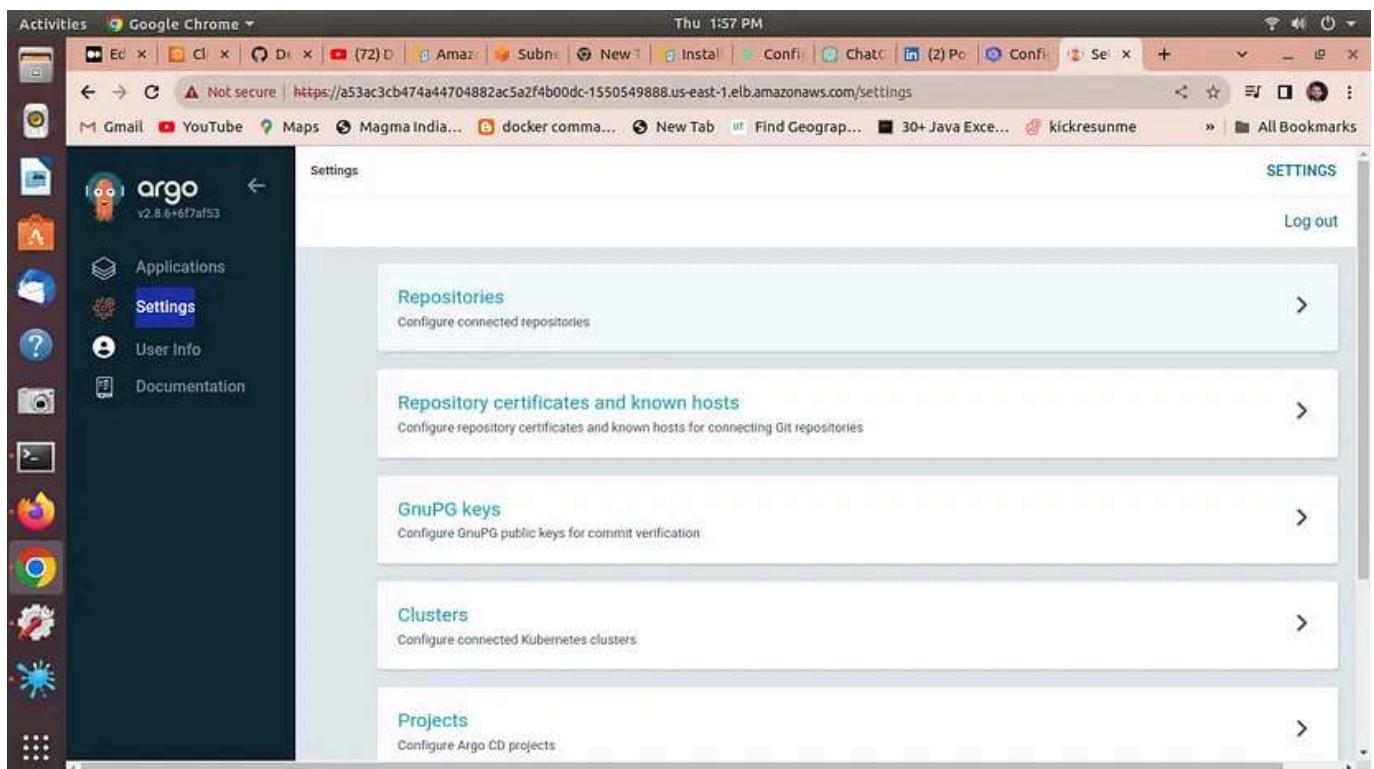
1. export ARGO_PWD=`kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d`
2. echo \$ARGO_PWD
3. output == password for argocd

```
root@ramalib-Veriton-M430:/home/ramalib# export ARGO_PWD=`kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d`
root@ramalib-Veriton-M430:/home/ramalib# echo $ARGO_PWD
stIjy0wLkehdf542
root@ramalib-Veriton-M430:/home/ramalib#
```

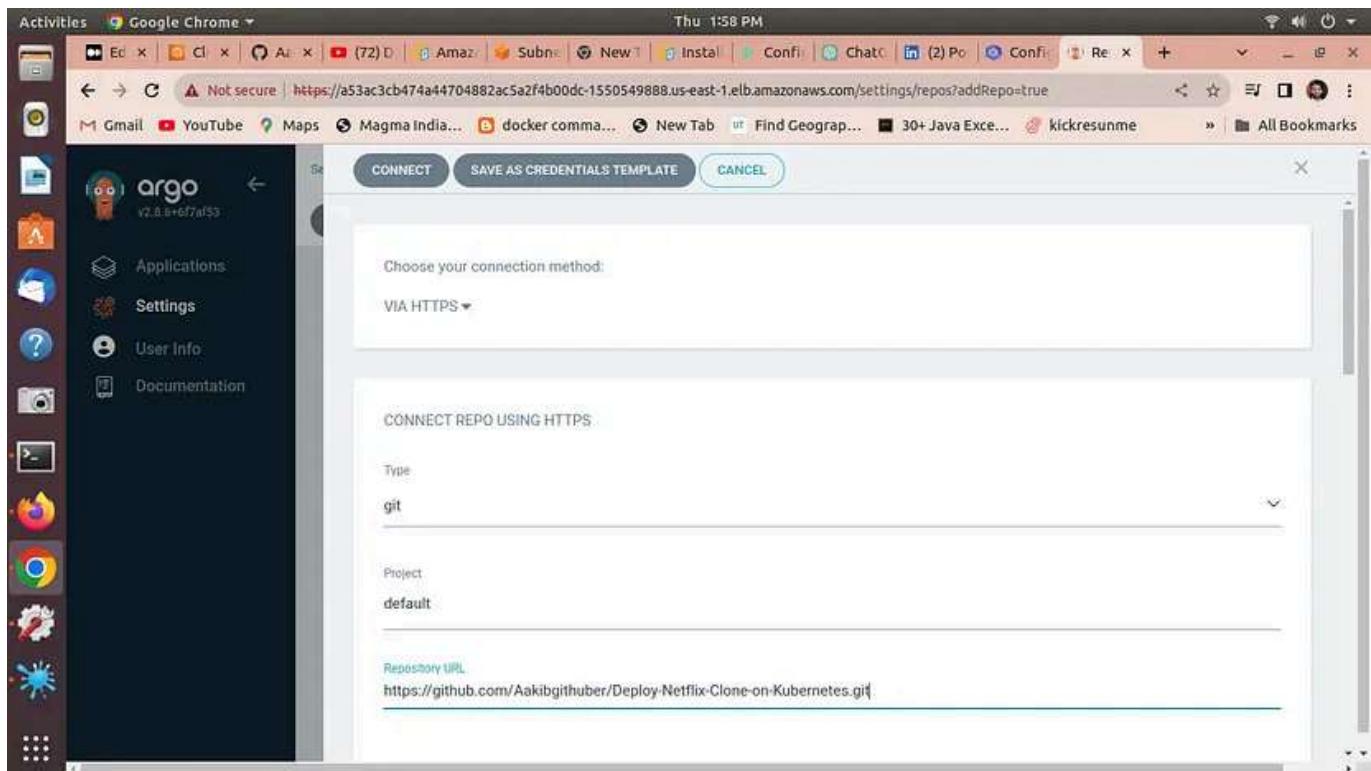
copy and paste the password on argocd page and you are logged in

Step 5 → Deploy Application with ArgoCD

Sign into argo cd using above steps →



1. go to setting → repositories → connect repo



select these options and on repo url copy n paste your github url

2. click on connect output →successful

3. Now go to application →newapp



https://github.com/Aakibgithuber/Deploy-Network-Clone-on-Kubernetes.git GIT✓

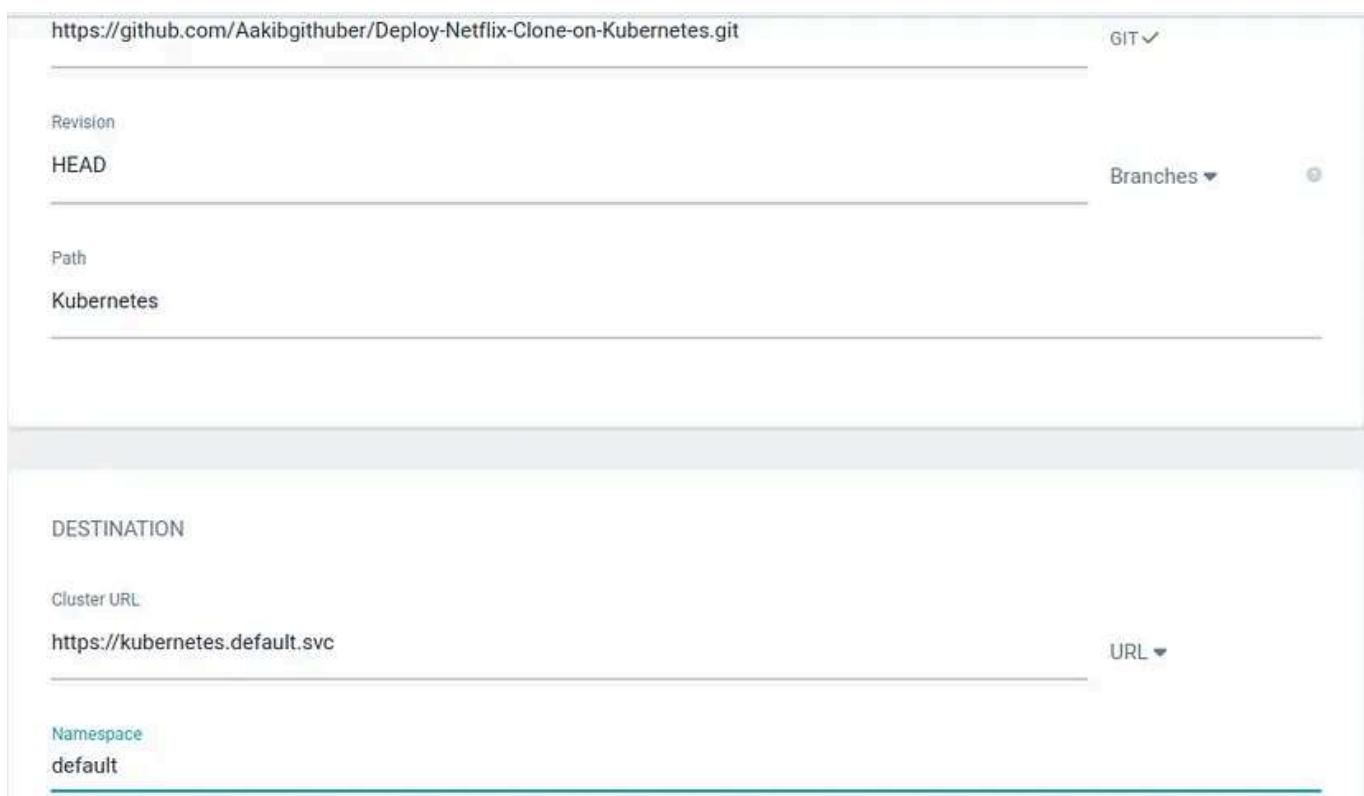
Revision
HEAD Branches ▾

Path
Kubernetes

DESTINATION

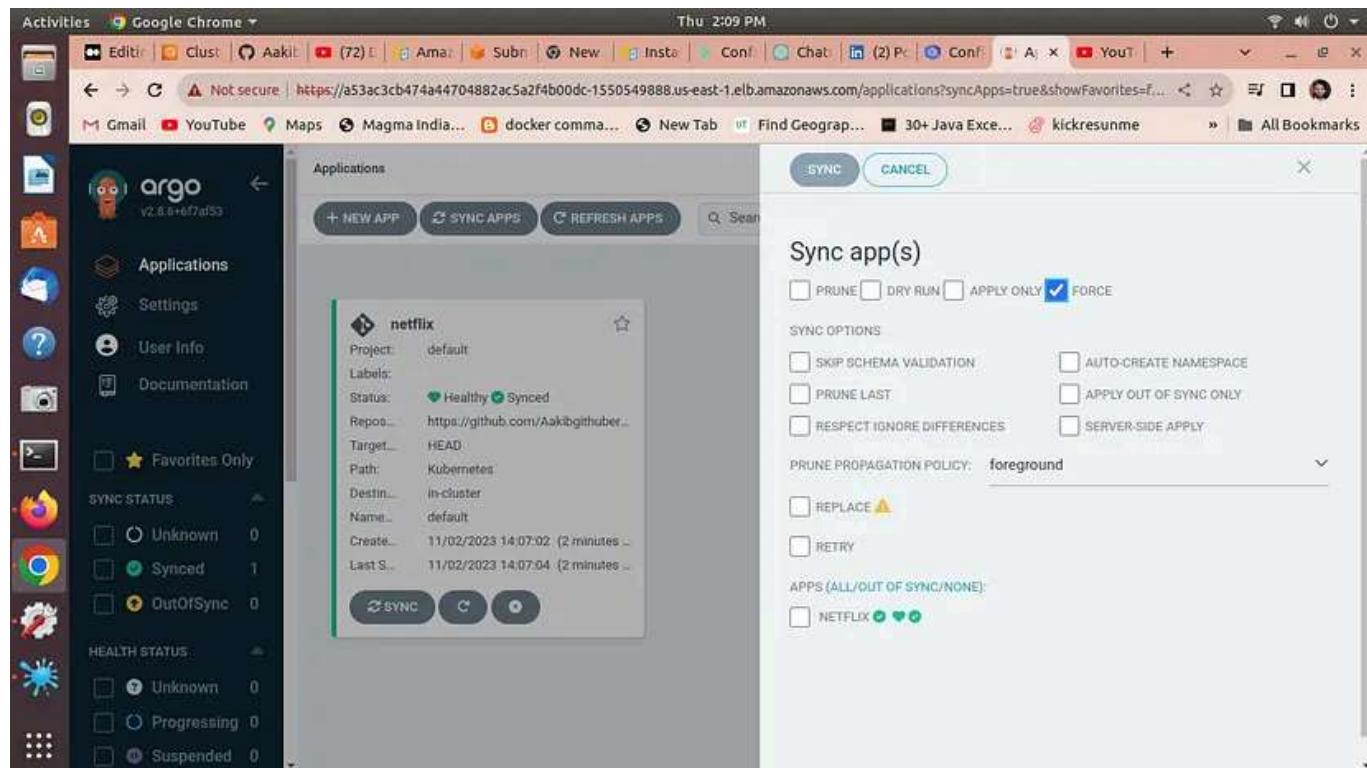
Cluster URL
https://kubernetes.default.svc URL ▾

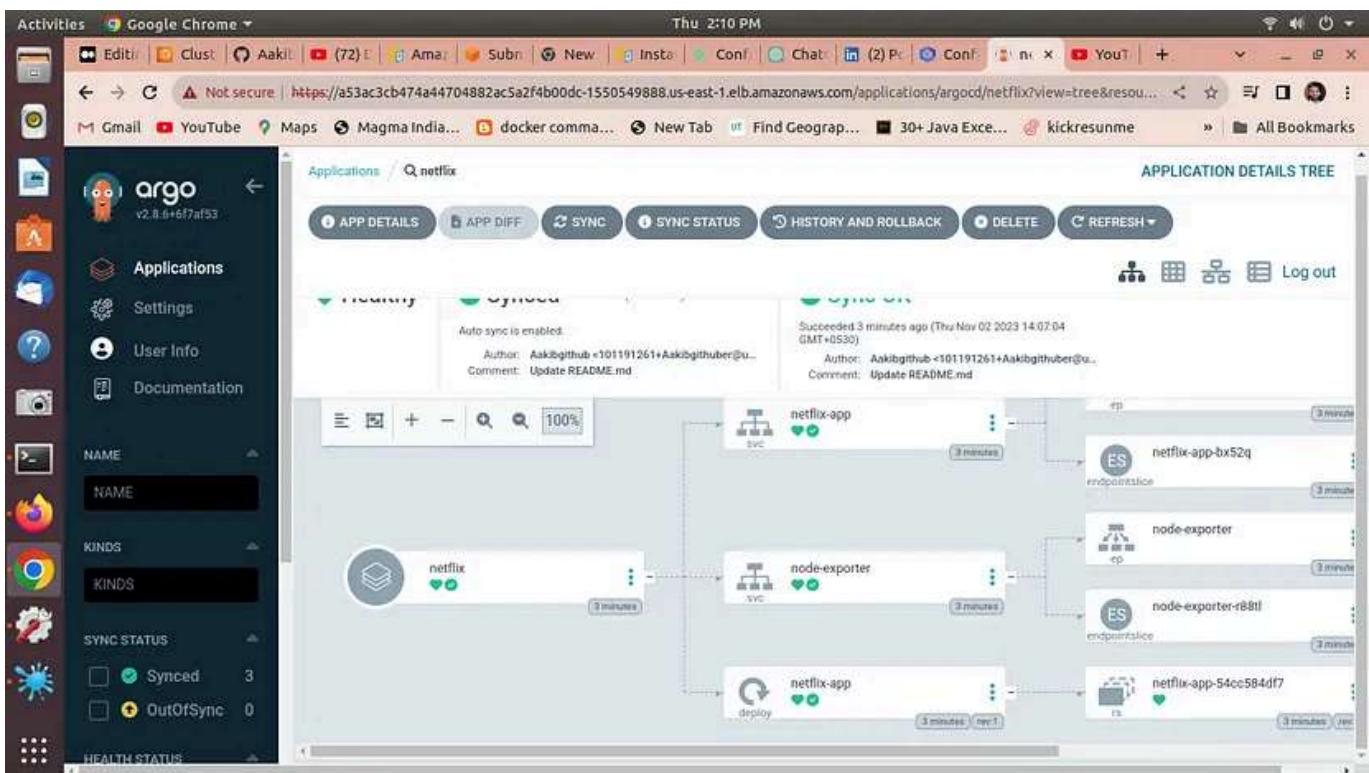
Namespace
default



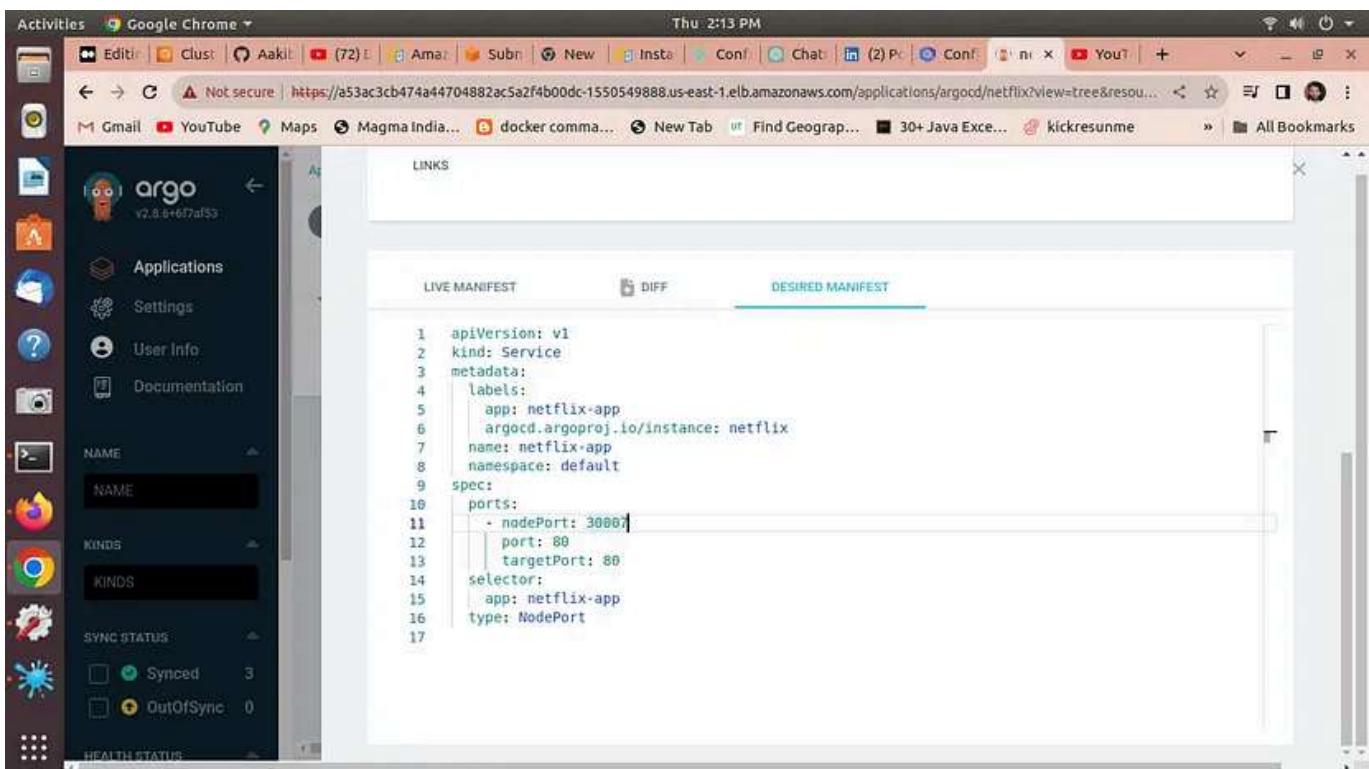
4. select the above options and click on create

5. Now click on sync →force →sync →ok





click on netflix app



6.go to your cluster and select the node created by node group and expose port 30007

Activities Google Chrome Thu 2:15 PM

us-east-1.console.aws.amazon.com/eks/home?region=us-east-1#clusters/Netflix_clus/nodes/ip-10-0-10-59.ec2.internal

Gmail YouTube Maps Magma India... docker comman... New Tab Find Geograph... 30+ Java Exec... kickresumne All Bookmarks

AWS Services Search [Alt+S]

Amazon Elastic Kubernetes Service

EKS > Clusters > Netflix_clus > Node: ip-10-0-10-59.ec2.internal

ip-10-0-10-59.ec2.internal Structured view Raw view C

Clusters New

Related services Amazon ECR AWS Batch

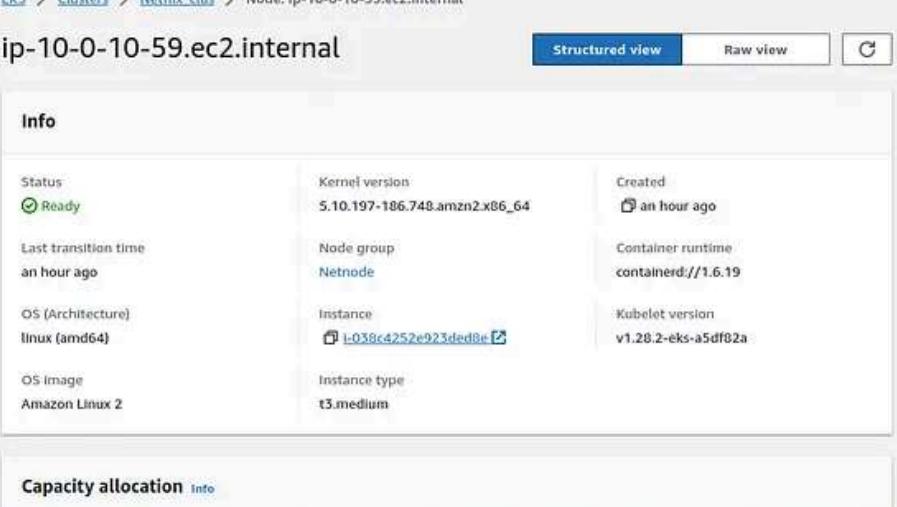
Documentation Submit feedback

Info

Status	Kernel version	Created
Ready	5.10.197-186.748.amzn2.x86_64	an hour ago
Last transition time	Node group	Container runtime
an hour ago	Netnode	containerd://1.6.19
OS (Architecture)	Instance	Kubelet version
linux (amd64)	i-038c4252e923ded8e	v1.28.2-eks-a5df02a
OS Image	Instance type	
Amazon Linux 2	t3.medium	

Capacity allocation Info

CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



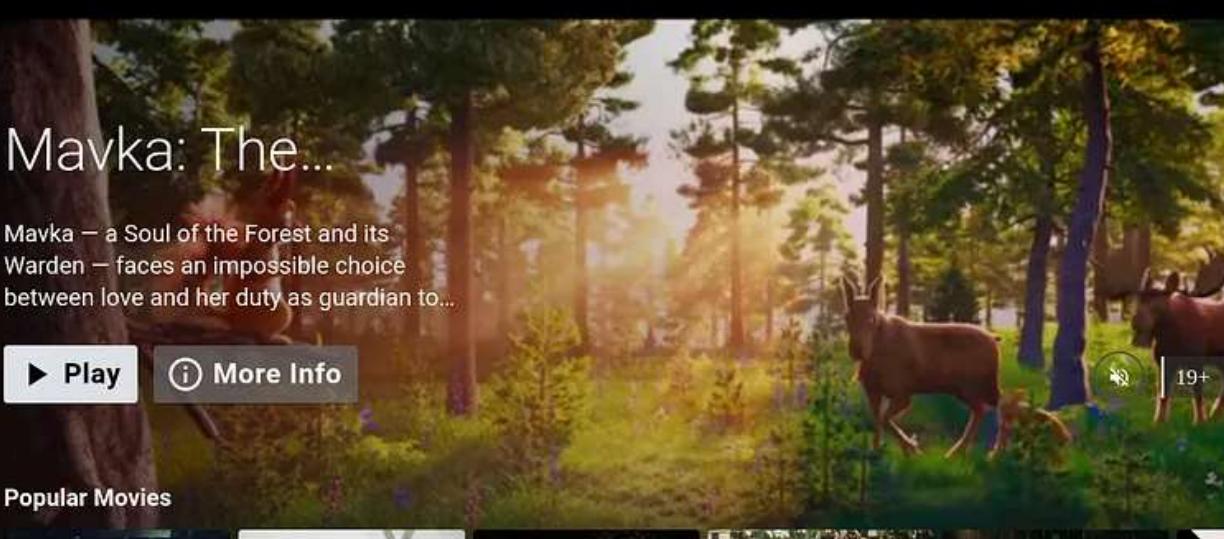
now browse the application by Publicip:30007

Activities Google Chrome Thu 2:18 PM

Not secure | 54.227.14.192:30007/browse

Gmail YouTube Maps Magma India... docker comman... New Tab Find Geograph... 30+ Java Exec... kickresumne All Bookmarks

NFLIX THE FILM LIST MOVIES FFID SHOWAILER Watch later Share



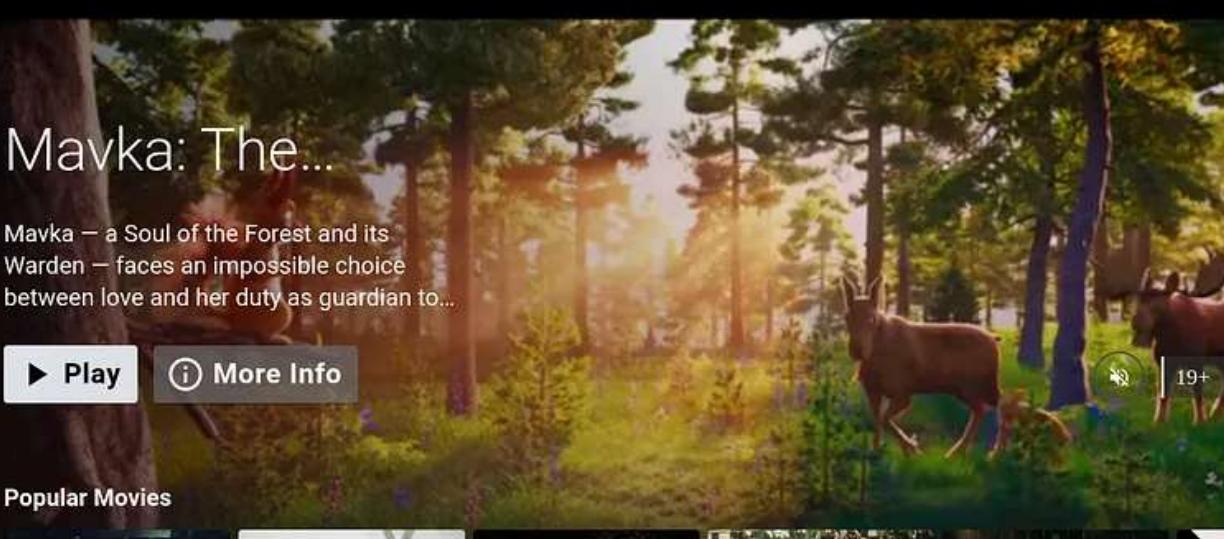
Mavka: The...

Mavka – a Soul of the Forest and its Warden – faces an impossible choice between love and her duty as guardian to...

Play More Info

Popular Movies

19+



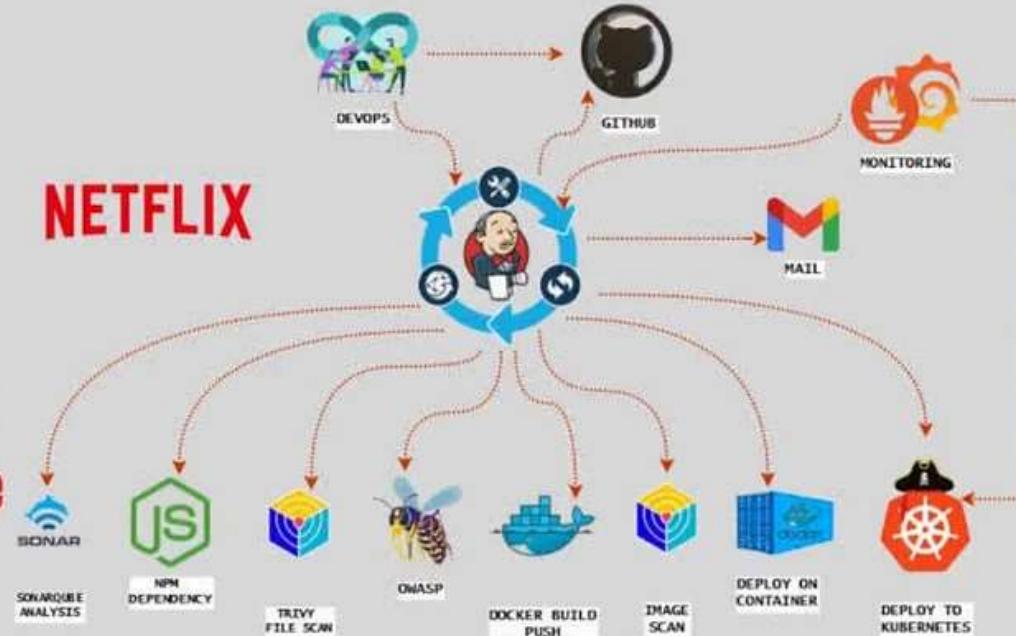
Here is your application is running on port 30007 by EKS

Congrats Here we completed our project go and add this on your resume



NETFLIX

DevSecOps Netflix clone



DevOps

Cloud Computing

Kubernetes

Docker

Jenkins



Written by Aakib

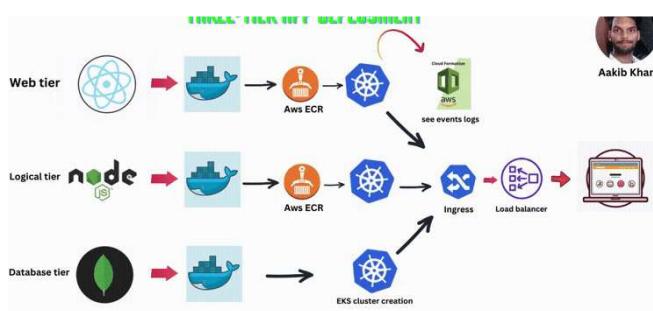
2.2K Followers

Cloud computing and DevOps Engineer and to be as a fresher I am learning and gaining experience by doing some hands on projects on DevOps and in AWS OR GCP

Follow



More from Aakib

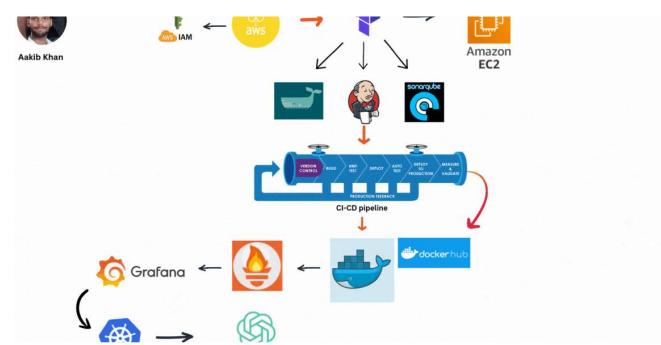


Aakib

Project 8 → Three tier application deployment on Kubernetes

what do you mean by three tier ?

Jan 26 ⚡ 379 🎙 3

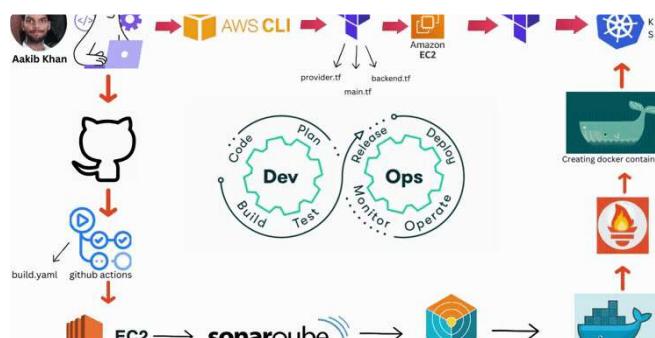


Aakib

Project 11→ Deployment of chat gpt clone app on kubernetes using...

Deployment of chat gpt bot using devSecOps approach towards deployment where we us...

May 19 ⚡ 60 🎙 1

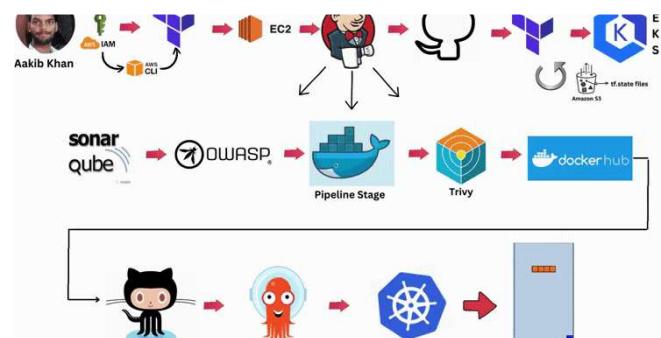


Aakib

Project 10 → Deployment of Swiggy application using github actions

Today we are going to deploy the Swiggy application using github actions as a ci-cd...

Feb 29 ⚡ 120 🎙 3



Aakib

Project 9 → Deployment of Tetris game on Kubernetes and...

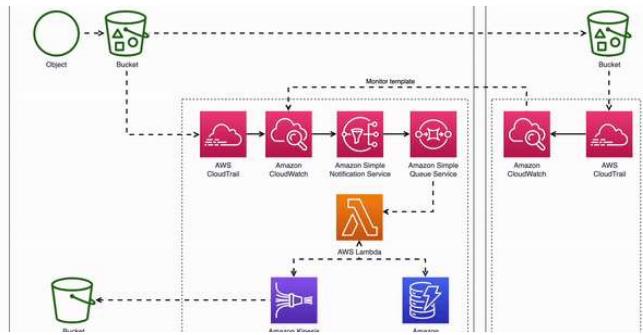
Project Intoduction

Jan 31 ⚡ 144 🎙 4



See all from Aakib

Recommended from Medium

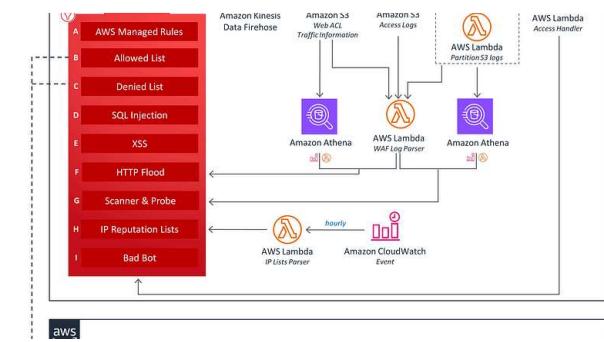


Vishwamitra Mishra

The Power of Flow Animation: Making Your Draw.io Diagrams...

Ever felt lost trying to understand how a website works or how your data gets from...

Feb 7 · 62



Nurunnubi Talukder... in Cloud, DevOps, Security & AI C...

Simplify AWS WAF Configurations: A Step-by-Step Guide for Cloud...

Struggling with AWS WAF Configurations? Here's What to Do...

Jun 27 · 7

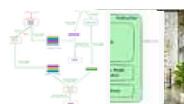


Lists



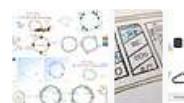
Coding & Development

11 stories · 697 saves



Natural Language Processing

1575 stories · 1121 saves



General Coding Knowledge

20 stories · 1379 saves



Productivity

241 stories · 497 saves

Amazon.com
Software Development Engineer Mar. 2020 – May 2021
• Developed Amazon checkout and payment services to handle traffic of 10 Million daily global transactions
• Integrated Iframes for credit cards and bank accounts to secure 80% of all consumer traffic and prevent CSRF, cross-site scripting, and cookie-jacking
• Led Your Transactions implementation for JavaScript front-end framework to showcase consumer transactions and reduce call center costs by \$25 Million
• Recovered Saudi Arabia checkout failure impacting 4000+ customers due to incorrect GET form redirection

Projects

NinjaPrep.io (React)
Platform to offer coding problem practice with built in code editor and written + video solutions in React
• Utilized Nginx to reverse proxy IP address on Digital Ocean hosts
• Developed using Styled-Components for 95% CSS styling to ensure proper CSS scoping
• Implemented Docker with Seccomp to safely run user submitted code with < 2.2s runtime

HeatMap (JavaScript)
Visualized Google Takeout location data of location history using Google Maps API and Google Maps heatmap code with React
• Included local file system storage to reliably handle 5mb of location history data
• Implemented Express to include routing between pages and jQuery to parse Google Map and implement heatmap overlay



 Alexander Nguyen in Level Up Coding

The resume that got a software engineer a \$300,000 job at Google.

1-page. Well-formatted.

 Jun 1  12.7K  167



 ByteCook

10 Troubleshooting Commands for Linux Systems

1. How to view processes consuming the most CPU?

 May 11  562  2



Screenshot of a Medium article draft titled "Build cloud Architecture Diagrams in 1 Minute (This Tool is Crazy Fast!)". The article discusses the challenges of creating architectural diagrams using traditional tools and introduces Diagram as Code as a solution. It includes a screenshot of the tool's interface showing a diagram generated from code.



 Naveenkumar Murugan in Level Up Coding

Build cloud Architecture Diagrams in 1 Minute (This Tool is Crazy Fast!)

Using ChatGPT and Diagrams python package

 Jan 21  560  4



 Oliver Foster in Stackademic

What's the Difference Between localhost and 127.0.0.1?

My article is open to everyone; non-member readers can click this link to read the full text.

 Feb 1  3.6K  22



[See more recommendations](#)