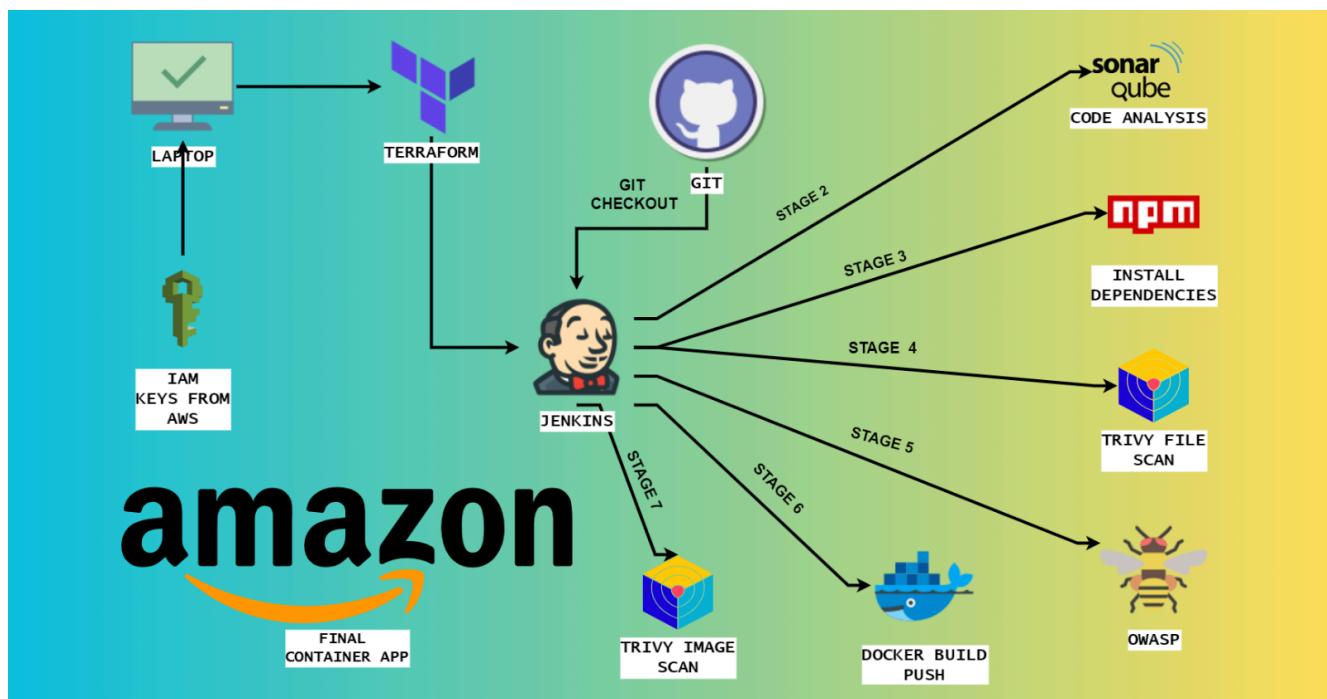


DevOps

Amazon App Deployment: A DevSecOps Approach with Terraform and Jenkins CI/CD



mrcloobook.com · 8 January 2024



Welcome to our in-depth guide on deploying an Amazon app with a strong focus on security through a DevSecOps approach. In today's fast-paced digital landscape, building and deploying applications not only requires speed but also airtight security. That's where DevSecOps comes into play, blending development, security, and operations into a single, unified process. In this blog post, we will embark on a journey where we'll leverage Terraform, Jenkins CI/CD, SonarQube, and Trivy to create a robust and secure pipeline for deploying applications on Amazon Web

Services (AWS). Whether you're an experienced developer looking to enhance your DevSecOps skills or a newcomer eager to explore this exciting intersection of software development and security, this guide has something valuable to offer. Let's dive in and explore the steps to safeguard your Amazon app while ensuring smooth and efficient deployment.

Contents [hide]

COMPLETE PROJECT VIDEO ON YOUTUBE

Step1: create an IAM user

Step2: Aws Configure

Step3: Terraform files and Provision Jenkins,sonar

Step 4 – Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

COMPLETE PROJECT VIDEO ON YOUTUBE

Amazon App Deployment: A DevSecOps Approach with Terraform and J...



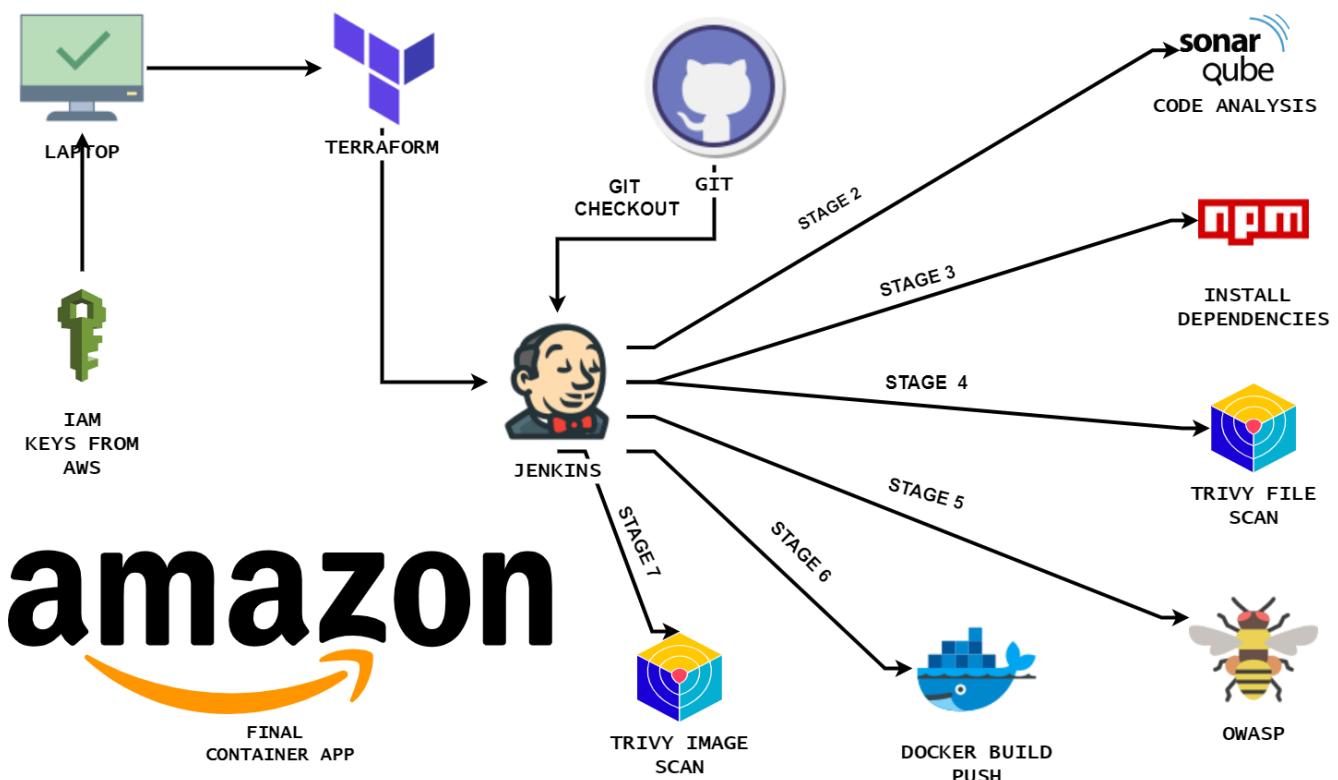
If you have doubts watch this video on YouTube

Terraform Automation: Provisioning EC2 with Jenkins and SonarQube f...



GITHUB LINK

<https://github.com/Aj7Ay/Amazon-FE.git>



Step1: create an IAM user

Navigate to the **AWS console**

Click the “Search” field.

Search for IAM

Click “Users”

Management (IAM)

[Dashboard](#)

▼ Access management

[User groups](#)

[Users](#)
[Roles](#)
[Policies](#)
[Identity providers](#)
[Account settings](#)

▼ Access reports

[Access analyzer](#)
[Archive rules](#)

IAM dashboard

Security recommendations 1

Root user has MFA

Having multi-factor authentication (MFA) for the root user improves security for this account.

Root user has no active access keys

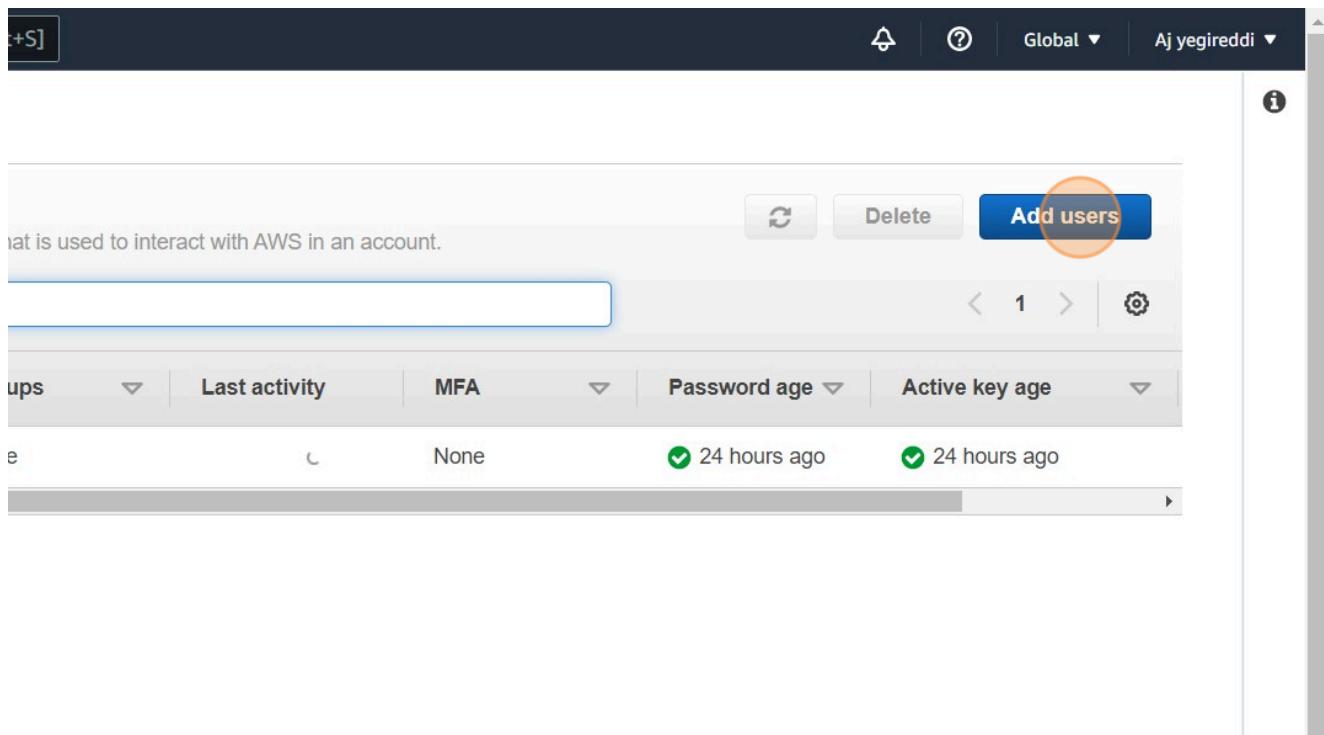
Using access keys attached to an IAM user instead of the root user improves security.

Update your access permissions for AWS Billing, Cost Management, and Account consoles

We are replacing the following IAM actions for Billing, Cost Management, and Account console granular IAM actions: aws-portal:ViewBilling, aws-portal:ModifyBilling, aws-portal:ViewAccount/portal:ModifyAccount, aws-portal:ViewPaymentMethods, aws-portal:ModifyPaymentMethods, aws-portal:ViewUsage, purchase-orders:ViewPurchaseOrders, and purchase-orders:ModifyPurchase. To ensure you don't lose access to AWS Billing, Cost Management, and Account console base update your existing IAM policies to include the new IAM actions before July 2023. Examples impacted include AWS Cost Explorer, AWS Budgets, Billing console, and more. For more information please visit [blog](#).

IAM resources

Click “Add users”



What is used to interact with AWS in an account.

User	Last activity	MFA	Password age	Active key age
me	2 hours ago	None	24 hours ago	24 hours ago

Click the “User name” field.

[Create user](#)

Specify user details

User details

User name

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

Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

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

Type “Terraform” or as you wish about the name

Click Next

acters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

Console - *optional*

[best practice](#) to manage their access in IAM Identity Center.

through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate [Learn more](#)

Cancel

Next

Click “Attach policies directly”



. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

The screenshot shows a step in the AWS Groups creation process. It presents two methods for managing user permissions:

- Copy permissions**
Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly**
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Below this, there is a note: "We recommend using groups to manage user permissions by job permissions." A "Create group" button is visible on the right.

Click this checkbox with Administrator access

The screenshot shows the "Permissions policies (1046)" section of the AWS IAM console. It lists various AWS managed policies:

<input type="checkbox"/>	Policy name	Type
<input type="checkbox"/>	AccessAnalyzerServiceRolePolicy	AWS managed
<input checked="" type="checkbox"/>	AdministratorAccess	AWS managed
<input type="checkbox"/>	AdministratorAccess-Amplify	AWS managed
<input type="checkbox"/>	AdministratorAccess-AWSElasticBe...	AWS managed
<input type="checkbox"/>	AlexaForBusinessDeviceSetup	AWS managed
<input type="checkbox"/>	AlexaForBusinessFullAccess	AWS managed
<input type="checkbox"/>	AlexaForBusinessGatewayExecution	AWS managed

Click “Next”

loud...	AWS managed	0
	AWS managed	0
:cess	AWS managed	0
s	AWS managed	0
is	AWS managed	0
Access	AWS managed	0
:cess	AWS managed	0

Assign permissions for this user. Use this advanced feature used to delegate permission management to others. [Learn more](#)

[Cancel](#)

[Previous](#)

[Next](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click “Create user”

Type	Used as
AWS managed - job function	Permissions policy

to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

[Cancel](#)

[Previous](#)

[Create user](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click newly created user in my case Ajay

IAM > Users

Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS.

Find users by username or access key

User name	Groups	Last active
Ajay	None	Never
mr-cloud-book	None	22 hours

Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzers
- Settings

Click “Security credentials”

Summary

ARN	Console access	Access key 1
arn:aws:iam::677356083070:user/Ajay	Disabled	Not enabled
Created	Last console sign-in	Access key 2
February 28, 2023, 14:41 (UTC+05:30)	-	Not enabled

Permissions **Groups** **Tags** **Security credentials** **Access Advisor**

Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Find policies

Policy name	Type	Attach
AdministratorAccess	AWS managed - job function	Directly

Click “Create access key”

[Dashboard](#)[Access management](#)[User groups](#)[Users](#)[Roles](#)[Policies](#)[Identity providers](#)[Account settings](#)[Access reports](#)[Access analyzer](#)[Archive rules](#)[Analyzers](#)[Settings](#)[Credential report](#)[Navigation activity](#)

No MFA devices. Assign an MFA device to impr

[Assign MFA](#)

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AW inactive) at a time. [Learn more](#)

[Create access key](#)**No access keys**

As a best practice, avoid using long-term credentials like access keys. Ins

[Create acc](#)

Click this radio button with the CLI

The screenshot shows the AWS IAM 'Create access key' wizard. At the top, there's a navigation bar with the AWS logo, 'Services' (selected), a search bar, and a keyboard shortcut '[Alt+S]'. Below the navigation, the path is shown as IAM > Users > Ajay > Create access key.

Step 1: Access key best practices & alternatives

A note says: "Avoid using long-term credentials like access keys to improve your security. Cor".

Step 2 - optional:

- Set description tag**

Step 3:

- Retrieve access keys**

Access key best practices & alternatives

A note says: "Avoid using long-term credentials like access keys to improve your security. Cor".

Usage Scenarios:

- Command Line Interface (CLI)** (Selected): "You plan to use this access key to enable the AWS CLI to access your AWS account."
- Local code**: "You plan to use this access key to enable application code in a local development access your AWS account."
- Application running on an AWS compute service**: "You plan to use this access key to enable application code running on an AWS con Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account."

Agree to terms

Application running outside AWS
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

Other
Your use case is not listed here.

⚠ Alternatives recommended

- Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)
- Use the [AWS CLI V2](#) and enable authentication through a user in IAM Identity Center. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.



Feedback

Language

Click Next

Application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

Use [AWS CloudShell](#), a browser-based CLI, to run commands. [Learn more](#)

Enable authentication through a user in IAM Identity Center. [Learn more](#)

I understand the above recommendation and want to proceed to create an access key.

[Cancel](#) [Next](#)

© 2023, Amazon Web Services India Private Limited or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

Click “Create access key”

onal

ed to this user as a tag and shown alongside the access key.

A good description will help you rotate this access key confidently later.

ers, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Cancel

Previous

Create access key

Download .csv file

Retrieve access keys

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key

Secret access key

 AKIAZ3NMRS57HAYY5GSX

 ***** Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [Best practices for managing AWS access keys](#).

Download .csv file

Done

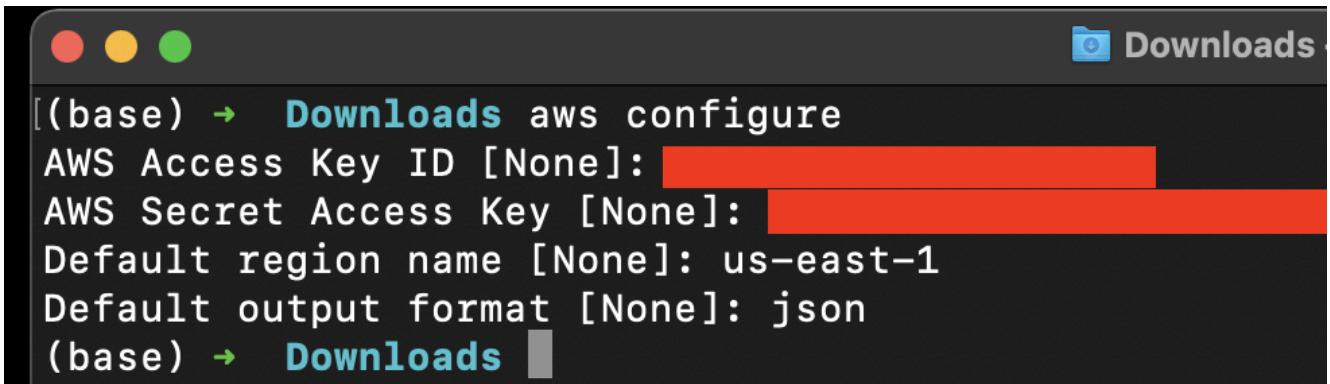
Step2: Aws Configure

Go to vs code or Cmd your wish



```
aws configure
```

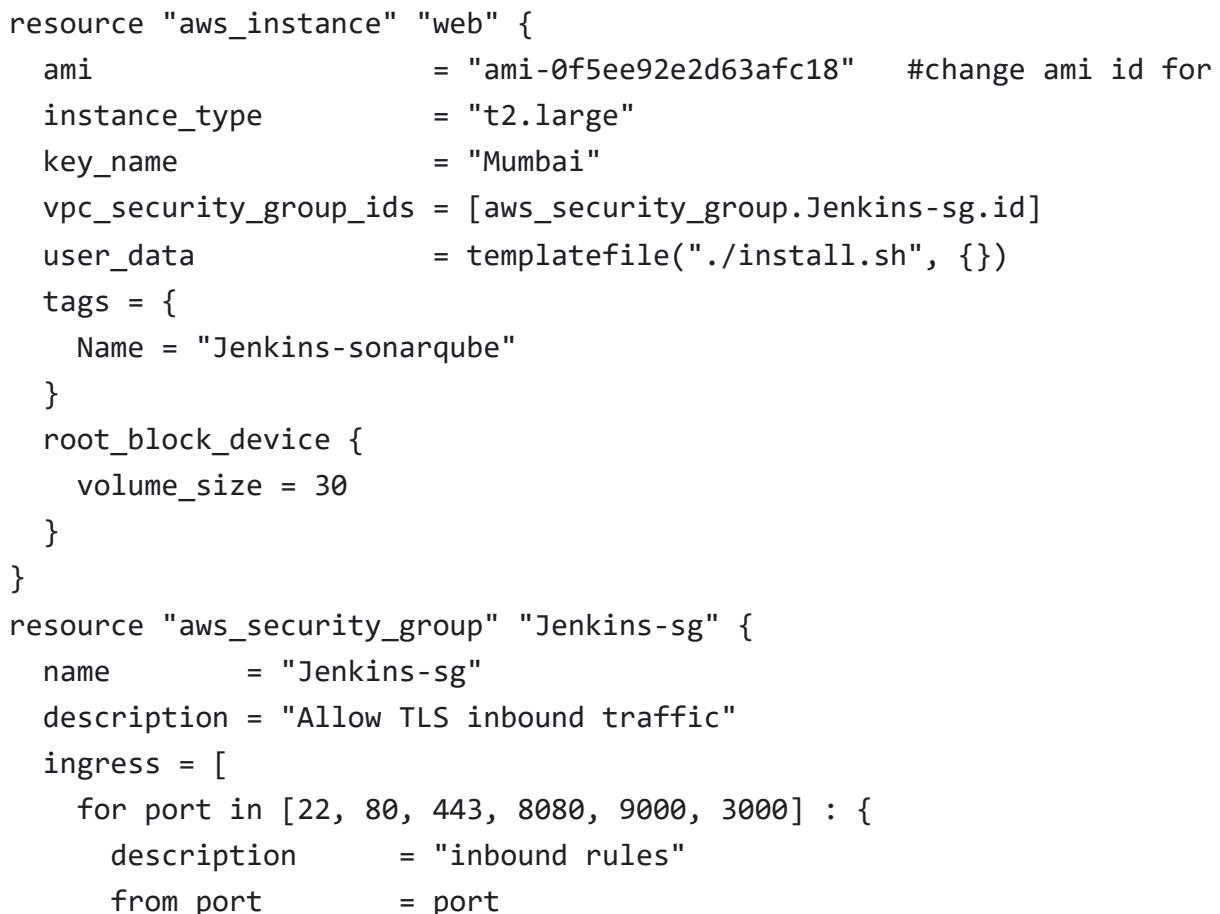
Provide your Aws Access key and Secret Access key



```
(base) → Downloads aws configure
AWS Access Key ID [None]: [REDACTED]
AWS Secret Access Key [None]: [REDACTED]
Default region name [None]: us-east-1
Default output format [None]: json
(base) → Downloads
```

Step3: Terraform files and Provision Jenkins,sonar

main.tf



```
resource "aws_instance" "web" {
    ami                  = "ami-0f5ee92e2d63afc18"      #change ami id for
    instance_type        = "t2.large"
    key_name             = "Mumbai"
    vpc_security_group_ids = [aws_security_group.Jenkins-sg.id]
    user_data            = templatefile("./install.sh", {})
    tags = {
        Name = "Jenkins-sonarqube"
    }
    root_block_device {
        volume_size = 30
    }
}
resource "aws_security_group" "Jenkins-sg" {
    name      = "Jenkins-sg"
    description = "Allow TLS inbound traffic"
    ingress = [
        for port in [22, 80, 443, 8080, 9000, 3000] : {
            description      = "inbound rules"
            from_port        = port
        }
    ]
}
```

```

        to_port          = port
        protocol        = "tcp"
        cidr_blocks     = ["0.0.0.0/0"]
        ipv6_cidr_blocks = []
        prefix_list_ids = []
        security_groups = []
        self            = false
    }
]
egress {
    from_port      = 0
    to_port        = 0
    protocol       = "-1"
    cidr_blocks   = ["0.0.0.0/0"]
}
tags = {
    Name = "jenkins-sg"
}
}

```

provider.tf



```

terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

# Configure the AWS Provider
provider "aws" {
  region = "ap-south-1" #change your region
}

```



install.sh

This will install Jenkins and Docker and Sonarqube and trivy

```
#!/bin/bash
sudo apt update -y
wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/api/debian-stable/jdk/temurin-17-jdk" | sudo tee /etc/apt/sources.list.d/adoptium.list
sudo apt update -y
sudo apt install temurin-17-jdk -y
/usr/bin/java --version
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /etc/apt/trusted.gpg.d/jenkins.io.key
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable/ | sudo tee /etc/apt/sources.list.d/jenkins.list
sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl start jenkins
sudo systemctl status jenkins

#install docker
sudo apt-get update
sudo apt-get install docker.io -y
sudo usermod -aG docker ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
docker run -d --name sonar -p 9000:9000 sonarqube:lts-community

#install trivy
sudo apt-get install wget apt-transport-https gnupg lsb-release -y
wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/trivy.gpg
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb/ | sudo tee /etc/apt/sources.list.d/trivy.list
sudo apt-get update
sudo apt-get install trivy -y
```

Terraform commands to provision

```
terraform init
terraform validate
```

```
terraform plan
```

```
terraform apply
```

```
+ prefix_list_ids = []
+ protocol        = "tcp"
+ security_groups = []
+ self            = false
+ to_port         = 80
},
+ {
+   + cidr_blocks  = [
+     + "0.0.0.0/0",
+   ]
+   + description   = "inbound rules"
+   + from_port     = 9000
+   + ipv6_cidr_blocks = []
+   + prefix_list_ids = []
+   + protocol      = "tcp"
+   + self          = false
+   + to_port       = 9000
},
]
+ name           = "Jenkins-sg"
+ name_prefix    = (known after apply)
+ owner_id       = (known after apply)
+ revoke_rules_on_delete = false
+ tags           = {
+   + "Name" = "jenkins-sg"
}
+ tags_all       = {
+   + "Name" = "jenkins-sg"
}
+ vpc_id         = (known after apply)
}

Plan: 2 to add, 0 to change, 0 to destroy.
aws_security_group.Jenkins-sg: Creating...
aws_security_group.Jenkins-sg: Creation complete after 2s [id=sg-085ec1a9c6e887160]
aws_instance.web: Creating...
aws_instance.web: Still creating... [10s elapsed]
aws_instance.web: Creation complete after 16s [id=i-0b1ace097700e8adf]

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

output

```
<instance-ip:8080> #jenkins
```

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

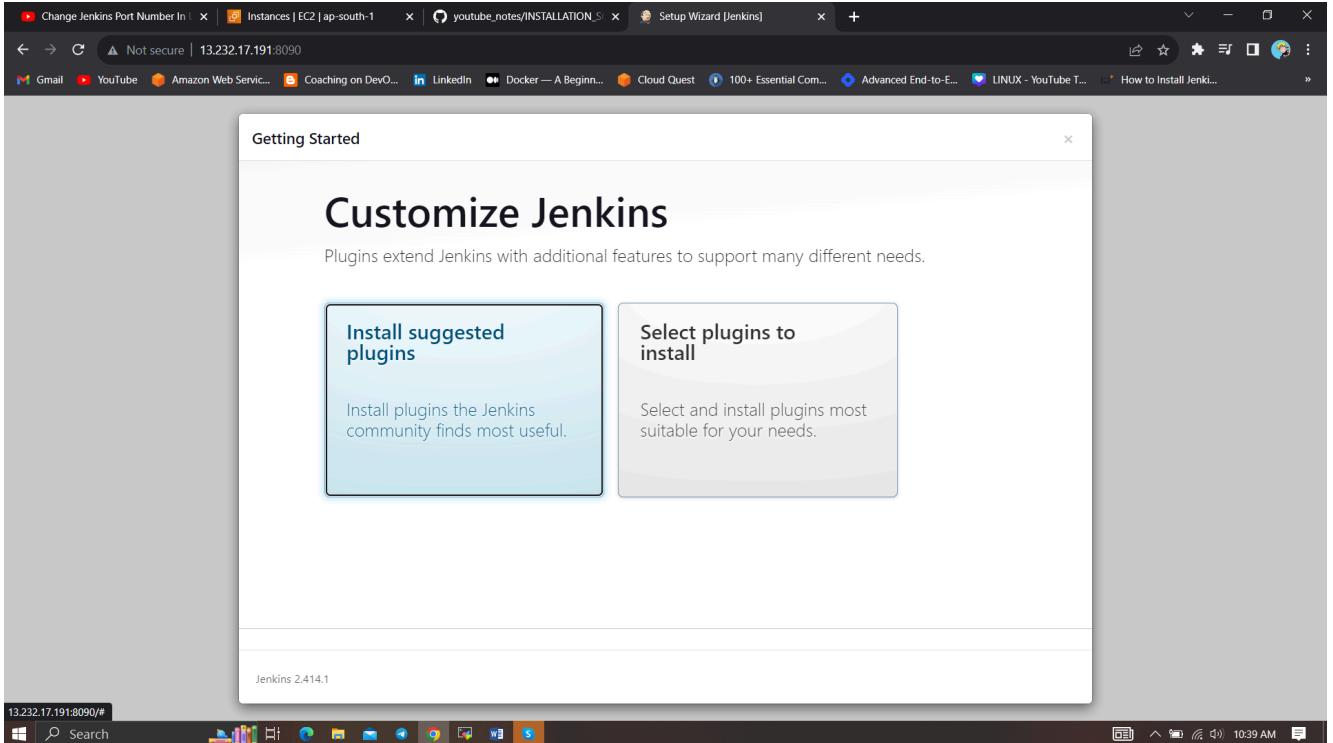
Continue



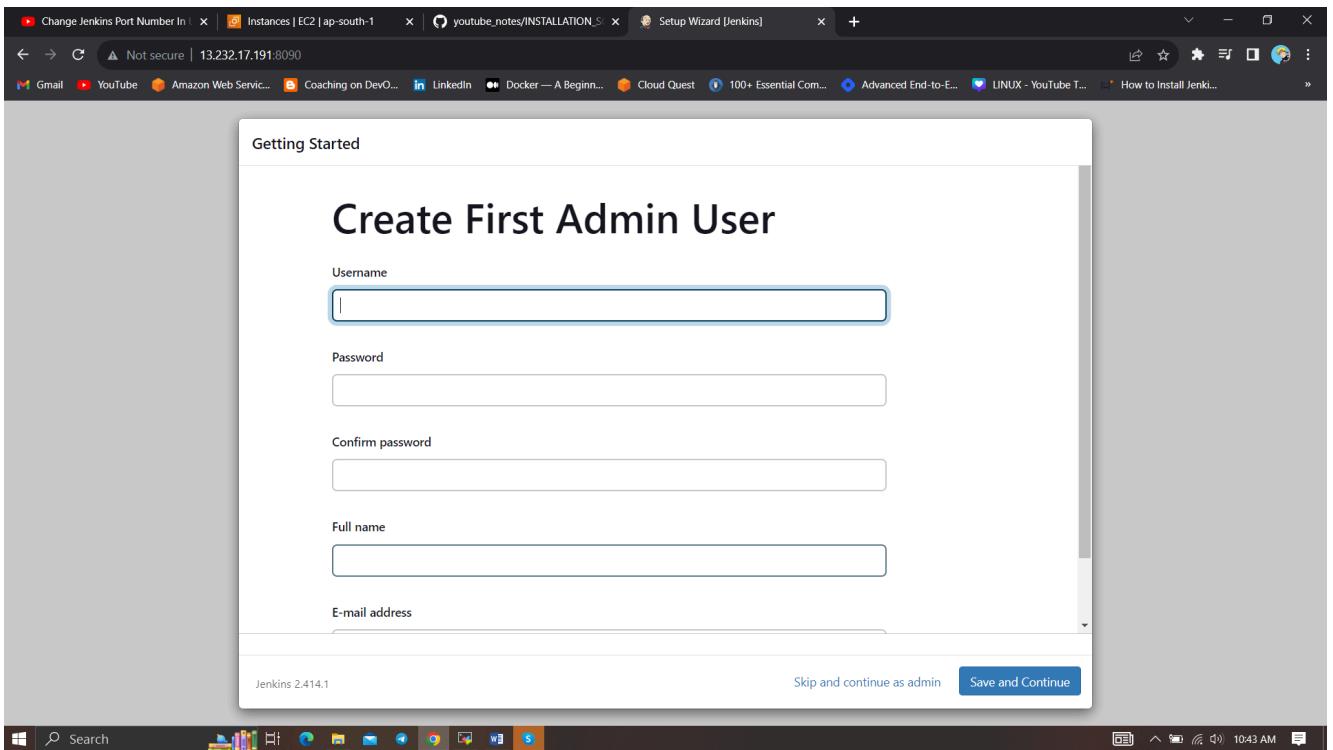
```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



Unlock Jenkins using an administrative password and install the suggested plugins.



Jenkins will now get installed and install all the libraries.



Create a user click on save and continue.

Jenkins Getting Started Screen.

Dashboard >

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Build Queue

No builds in the queue.

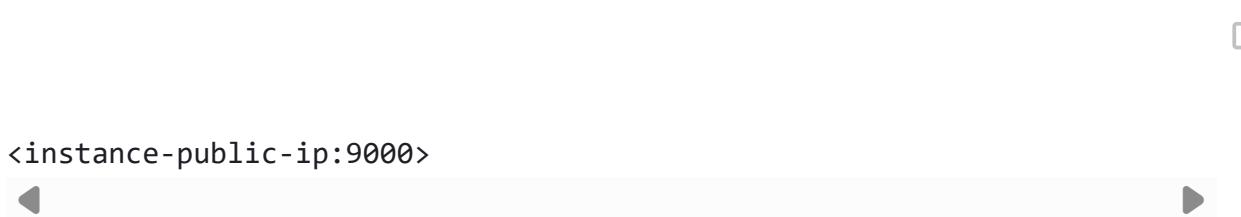
Build Executor Status

1 Idle
2 Idle

Set up a distributed build

Create a job →
Set up an agent →
Configure a cloud →
Learn more about distributed builds ↗

Copy your Public key again and paste it into a new tab



Now our sonarqube is up and running

Instances | EC2 | ap-south-1 petstore Config [Jenkins] SonarQube

52.66.140.95:9000/sessions/new?return_to=%2F

Amazon Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End

Log in to SonarQube

Login

Password

Log in Cancel

Enter username and password, click on login and change password



```
username admin
password admin
```



Instances | EC2 | ap-south-1 | petstore Config [Jenkins] | SonarQube | +

6.140.95:9000/account/reset_password

Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E...

Update your password

This account should not use the default password.

Enter a new password

All fields marked with * are required

Old Password *

New Password *

Confirm Password *

Update

Update New password, This is Sonar Dashboard.

← → ⌂ Not secure | 52.66.140.95:9000/projects/create

Gmail YouTube Amazon Web Servic... Coaching on DevO... LinkedIn Docker — A Beginn... Cloud Quest 100+ Essential Com... Advanced End-to-E... LINUX - YouTube T... How to Install Jenki...

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration ? Search for projects... A

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform. First, you need to set up a DevOps platform configuration.

From Azure DevOps Set up global configuration	From Bitbucket Server Set up global configuration	From Bitbucket Cloud Set up global configuration	From GitHub Set up global configuration	From GitLab Set up global configuration
--	--	---	--	--

Check trivy version

```
trivy --version
```



Step 4 – Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

4A – Install Plugin

Goto Manage Jenkins → Plugins → Available Plugins →

Install below plugins

1 → Eclipse Temurin Installer (Install without restart)

2 → SonarQube Scanner (Install without restart)

3 → NodeJs Plugin (Install Without restart)

Plugin	Description	Last Updated
Eclipse Temurin installer 1.5	Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	11 mo ago
SonarQube Scanner 2.15	External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	9 mo 19 days ago

The screenshot shows the Jenkins Marketplace interface. At the top, there are tabs for 'Install' and 'Name ↓'. Below these, a card for the 'NodeJS 1.6.1' plugin is displayed. The card includes the plugin name, version '1.6.1', a status indicator 'npm', and a note stating 'NodeJS Plugin executes NodeJS script as a build step.' To the right of the card, the word 'Released' is shown. At the bottom right of the card, the text '1 mo 2 days ago' indicates the last update.

4B – Configure Java and Nodejs in Global Tool Configuration

Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(16) → Click on Apply and Save

The screenshot shows the 'JDK installations' configuration page in Jenkins. The URL is 'Dashboard > Manage Jenkins > Tools > JDK installations'. A new entry 'jdk17' is being added under the 'JDK' section. The 'Name' field contains 'jdk17'. The 'Install automatically' checkbox is checked. Under the 'Install from adoptium.net' section, the 'Version' dropdown is set to 'jdk-17.0.8.1+1'. There is also an 'Add Installer' button.

The screenshot shows the 'NodeJS' configuration page in Jenkins. The URL is 'Dashboard > Manage Jenkins > Tools > NodeJS'. A new entry 'node16' is being added under the 'NodeJS' section. The 'Name' field contains 'node16'. The 'Install automatically' checkbox is checked. Under the 'Install from nodejs.org' section, the 'Version' dropdown is set to 'NodeJS 16.2.0'. Below this, there is a note about forcing the 32-bit architecture and a field for global npm packages to install.

Step 5 – Configure Sonar Server in Manage Jenkins

Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server. Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token

The screenshot shows the SonarQube administration interface. The top navigation bar includes links for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. The Administration link is highlighted with a red box. Below the navigation, there's a sub-menu for Security with options: Configuration, Security (selected), Projects, System, and Marketplace. A dropdown menu for 'Users' is open, also highlighted with a red box. Other options in the dropdown include Groups, Global Permissions, and Permission Templates. A search bar labeled 'Find i...' is visible at the bottom left.

click on update Token

The screenshot shows the SonarQube Tokens page for the 'Administrator' user. The top navigation bar includes links for SCM Accounts, Last connection, Groups, and Tokens. The Tokens link is highlighted with a red box. The main table lists two groups: 'sonar-administrators' and 'sonar-users'. For each group, it shows the number of tokens (0) and a 'Tokens' icon. At the bottom right of the table, there is a large 'Update Tokens' button highlighted with a red box.

Create a token with a name and generate

The screenshot shows the 'Tokens of Administrator' page. The top navigation bar includes links for SCM Accounts, Last connection, Groups, and Tokens. The Tokens link is highlighted with a red box. The 'Generate Tokens' section allows setting a token name ('Enter Token Name') and expiration ('Expires in 30 days'). A 'Generate' button is present. 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 'Copy' button is shown next to the token value 'squ_21d162904c1c72cf8b39665f96488185c99dc2f9'. A table lists the generated token: Jenkins, User type, Never last use, September 8, 2023 created, and October 8, 2023 expiration. A 'Revoke' button is at the bottom right of the table.

copy Token

Goto Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this

New credentials

Kind

Secret text

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

Secret
POST THE TOKEN HERE

ID ?
Sonar-token

Description ?
Sonar-token

Create

You will see this page once you click on create

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
 Sonar-token	sonar	Secret text	sonar 

Now, go to Dashboard → Manage Jenkins → System and Add like the below image.

SonarQube servers

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

Environment variables Enable injection of SonarQube server configuration as build environment variables

SonarQube installations

List of SonarQube installations

Name
sonar-server

Server URL
Default is http://localhost:9000
http://13.232.17.191:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
Sonar-token

Add ▾

Save **Apply**

Click on Apply and Save

The Configure System option is used in Jenkins to configure different server

Global Tool Configuration is used to configure different tools that we install using Plugins

We will install a sonar scanner in the tools.

The screenshot shows the Jenkins 'Tools' configuration page under 'Manage Jenkins'. It is specifically for 'SonarQube Scanner installations'. A new configuration is being created with the name 'sonar-scanner'. The 'Install automatically' checkbox is checked. Under 'Install from Maven Central', the version 'SonarQube Scanner 5.0.1.3006' is selected. There is a 'Save' button at the bottom left and an 'Apply' button at the bottom right.

In the Sonarqube Dashboard add a quality gate also

Administration-> Configuration->Webhooks

The screenshot shows the SonarQube 'Administration' page under 'Configuration'. The 'Webhooks' option in the sidebar is highlighted with a red box. The main content area shows a table with one row for 'Administrator admin'. The table includes columns for 'SCM Accounts', 'Last connection', 'Groups', and 'Tokens'. The 'Last connection' column shows '< 1 hour ago'. The 'Groups' column lists 'sonar-administrators' and 'sonar-users'. The 'Tokens' column shows '1' and a settings icon. A 'Create User' button is visible in the top right of the main content area.

Click on Create

The screenshot shows the SonarQube Administration interface under the 'Webhooks' section. It displays a message stating 'No webhook defined.' and a 'Create' button.

Add details



```
#in url section of quality gate
<http://jenkins-public-ip:8080>/sonarqube-webhook/
```



The screenshot shows the 'Create Webhook' dialog box. It has fields for 'Name' (jenkins), 'URL' (http://43.204.36.242:8090/sonarqube-webhook/), and 'Secret'. A note at the bottom left says 'Embedded database should be used...'. A 'Create' button is at the bottom right.

Let's go to our Pipeline and add the script in our Pipeline Script.

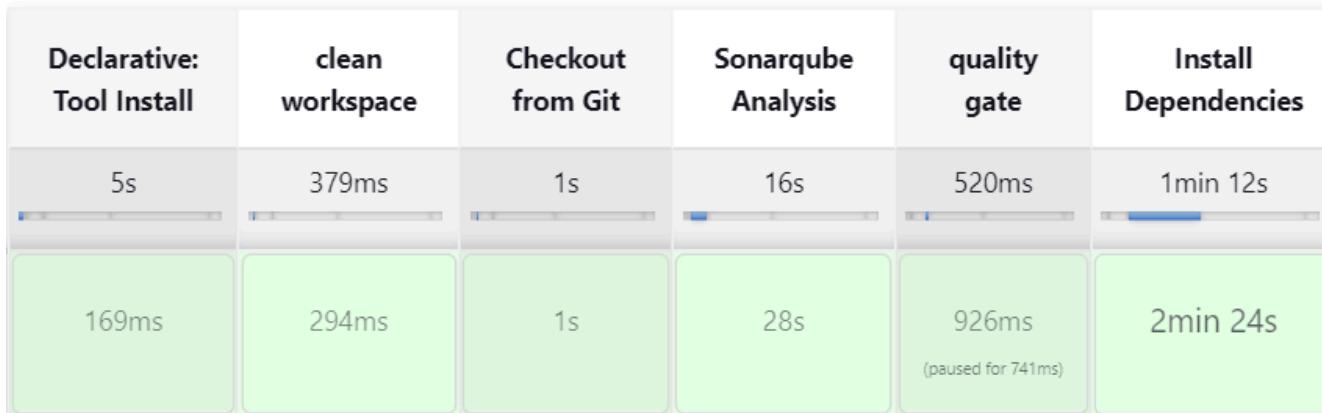


```
pipeline{
    agent any
    tools{
        jdk 'jdk17'
        nodejs 'node16'
    }
}
```

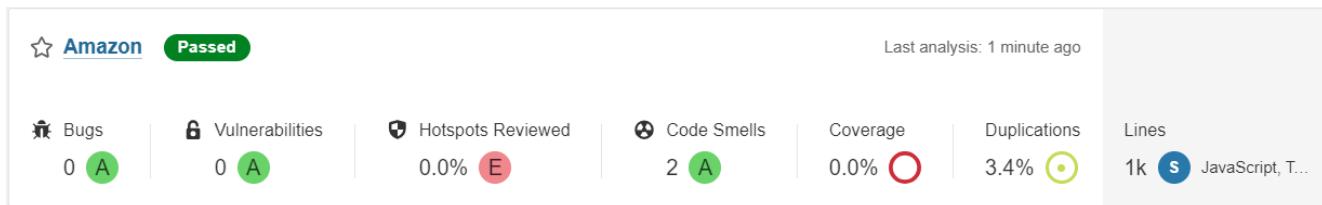
```
environment {
    SCANNER_HOME=tool 'sonar-scanner'
}
stages {
    stage('clean workspace'){
        steps{
            cleanWs()
        }
    }
    stage('Checkout from Git'){
        steps{
            git branch: 'main', url: 'https://github.com/Aj7Ay/Amazon-App-Deployment-A-DevSecOps-Approach-with-Terraform-and-Jenkins-CI-CD.git'
        }
    }
    stage("Sonarqube Analysis"){
        steps{
            withSonarQubeEnv('sonar-server') {
                sh '''$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Amazon -Dsonar.projectKey=Amazon'''
            }
        }
    }
    stage("quality gate"){
        steps {
            script {
                waitForQualityGate abortPipeline: false, credentialsId: 'jenkins-admin'
            }
        }
    }
    stage('Install Dependencies') {
        steps {
            sh "npm install"
        }
    }
}
```

Click on Build now, you will see the stage view like this





To see the report, you can go to Sonarqube Server and go to Projects.



You can see the report has been generated and the status shows as passed. You can see that there are 1k lines it scanned. To see a detailed report, you can go to issues.

Step 6 – Install OWASP Dependency Check Plugins

GotoDashboard → Manage Jenkins → Plugins → OWASP Dependency-Check. Click on it and install it without restart.

Dashboard > Manage Jenkins > Plugins

Plugins

- Updates
- Available plugins (selected)
- Installed plugins
- Advanced settings
- Download progress

Search available plugins /

Install

Install	Name	Released
<input checked="" type="checkbox"/>	OWASP Dependency-Check 5.4.2	8 days 17 hr ago
	Security DevOps Build Tools Build Reports	
	This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.	

First, we configured the Plugin and next, we had to configure the Tool

Goto Dashboard → Manage Jenkins → Tools →

Dependency-Check installations

[Add Dependency-Check](#)

Dependency-Check

Name

DP-Check

 Install automatically [?](#)

Install from github.com

Version

dependency-check 6.5.1

[Add Installer ▾](#)

Click on Apply and Save here.

Now go configure → Pipeline and add this stage to your pipeline and build.

```
stage('OWASP FS SCAN') {  
    steps {  
        dependencyCheck additionalArguments: '--scan ./ --disab'  
        dependencyCheckPublisher pattern: '**/dependency-check-1  
    }  
}  
stage('TRIVY FS SCAN') {  
    steps {  
        sh "trivy fs . > trivyfs.txt"  
    }  
}
```

The stage view would look like this,



You will see that in status, a graph will also be generated and Vulnerabilities.

Dependency-Check Results

SEVERITY DISTRIBUTION			
5	6	6	
			Search <input type="button" value="🔍"/>
File Name	Vulnerability	Severity	Weakness
+ axios:1.3.4	OSSINDEX CVE-2023-45857	🟡 Medium	CWE-352
+ css-what:3.4.2	OSSINDEX CVE-2022-21222	🟠 High	CWE-1333
+ ejs:3.1.9	NVD CVE-2023-29827	🔴 Critical	CWE-74
+ jsonpointer:5.0.1	NVD CVE-2022-4742	🔴 Critical	CWE-1321
+ nth-check:1.0.2	NVD CVE-2021-3803	🟠 High	CWE-1333
+ parseurl:1.3.3	NVD CVE-2022-0722	🟠 High	CWE-200
+ parseurl:1.3.3	NVD CVE-2022-2216	🔴 Critical	CWE-918
+ parseurl:1.3.3	NVD CVE-2022-2217	🟡 Medium	CWE-79
+ parseurl:1.3.3	NVD CVE-2022-2218	🟡 Medium	CWE-79
+ parseurl:1.3.3	NVD CVE-2022-2900	🔴 Critical	CWE-918

Step 7 – Docker Image Build and Push

We need to install the Docker tool in our system, Goto Dashboard → Manage Plugins → Available plugins → Search for Docker and install these plugins

Docker

Docker Commons

Docker Pipeline

Docker API

docker-build-step

and click on install without restart

The screenshot shows the Jenkins Plugins page with a search bar for "docker". The results list three plugins:

- Docker 1.5**: Version 439.va_3cb_0a_6a_fb_29, released 3 days 15 hr ago. This plugin integrates Jenkins with Docker.
- Docker Commons**: Version 439.va_3cb_0a_6a_fb_29, released 1 mo 29 days ago. Provides the common shared functionality for various Docker-related plugins.
- Docker Pipeline**: Version 572.v950f58993843, released 27 days ago. Build and use Docker containers from pipelines.
- Docker API**: Version 3.3.1-79.v20b_53427e041, released 3 mo 4 days ago. This plugin provides docker-java API for other plugins.

Now, goto Dashboard → Manage Jenkins → Tools →

The screenshot shows the Jenkins Tools configuration page under "Docker installations". It has a form to add a new Docker installation:

- Name**: docker
- Install automatically**: checked
- Download from docker.com**
- Docker version**: latest
- Add Installer**: dropdown menu

Add DockerHub Username and Password under Global Credentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Kind: Username with password

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

Username: sevenajay

Treat username as secret

Password:

ID: docker

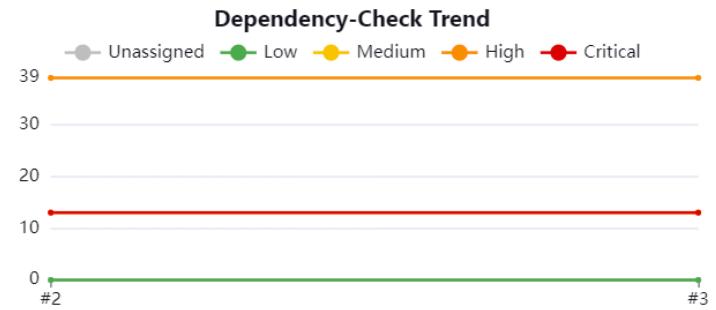
Description: docker

Create

Add this stage to Pipeline Script

```
stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName
                sh "docker build -t amazon ."
                sh "docker tag amazon sevenajay/amazon:latest"
                sh "docker push sevenajay/amazon:latest"
            }
        }
    }
}
stage("TRIVY"){
    steps{
        sh "trivy image sevenajay/amazon:latest > trivyimage.txt"
    }
}
```

You will see the output below, with a dependency trend.



Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY
3s	366ms	1s	19s	451ms	1min 20s	2min 1s	16s	3min 9s	4s
154ms	341ms	1s	25s	315ms	1min 36s	2min 31s	23s	3min 9s	4s

When you log in to Dockerhub, you will see a new image is created

sevenajay / amazon

Description

This repository does not have a description

Last pushed: a minute ago

Docker commands

To push a new tag to this repository:

[Public View](#)

`docker push sevenajay/amazon:tagname`

Now Run the container to see if the game coming up or not by adding the below stage

```
stage('Deploy to container'){
    steps{
        sh 'docker run -d --name amazon -p 3000:3000 sevenajay/:latest'
    }
}
```

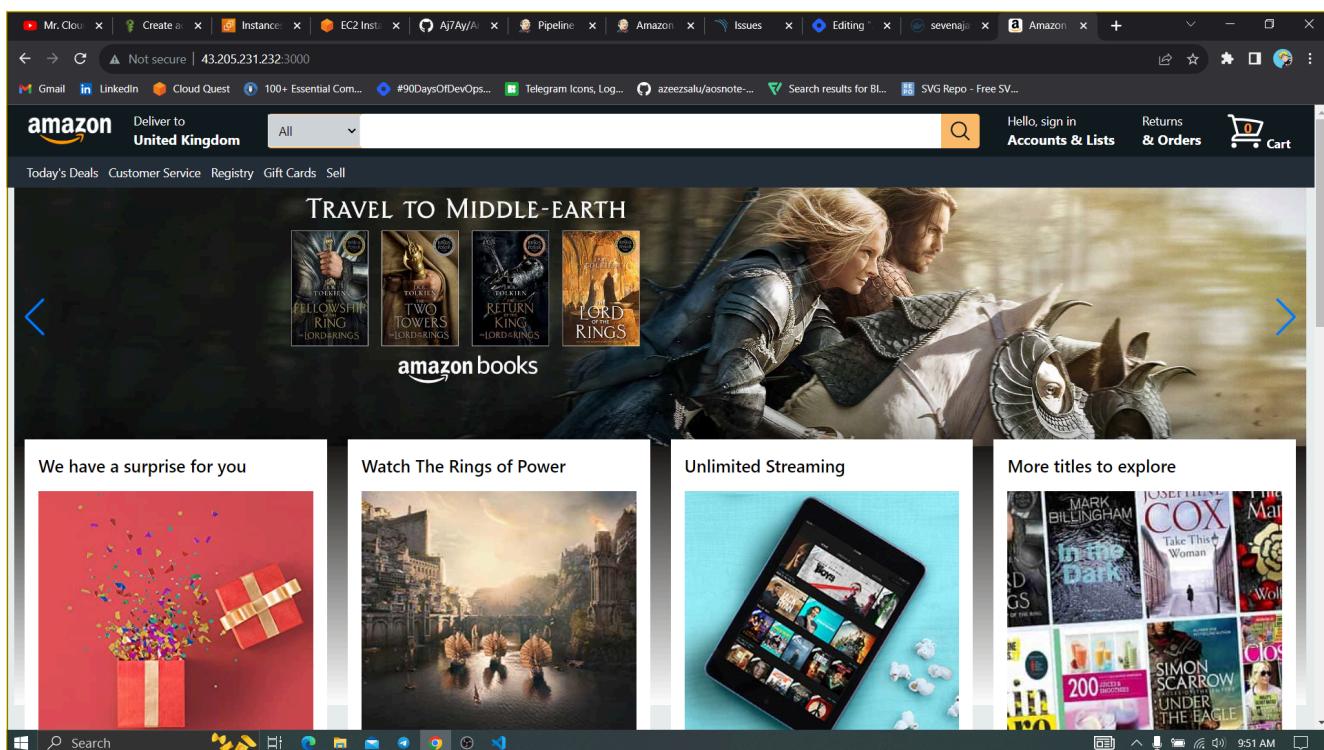
stage view

Amazon - Stage View

	Declarative: Tool Install	CW	GIT CO	SONAR CA	QG	NPM	OWASP FS SCAN	TRIVY FS	DB & Dp	TRIVY IMAGE	AMAZON APP
Average stage times: (Average full run time: ~6min 8s)											
#9 Oct 25 09:40	No Changes	133ms	274ms	1s	25s	466ms	24s	4min 40s	24s	3min 26s	2min 7s
#8 Oct 25 09:27	No Changes	136ms	277ms	1s	24s	379ms	17s	6min 29s	26s		
#7 Oct 25 09:22	No Changes	128ms	266ms	1s	28s	723ms (paused for 1s)	37s				

<Jenkins-public-ip:3000>

You will get this output



Let's destroy everything

Go to VS Code and provide the below command (or) Go to the path where you provisioned the Ec2 instance



```
terraform destroy --auto-approve
```



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT CODEWHISPERER REFERENCE LOG

- {
  - cidr_blocks = [
    - "0.0.0.0/0",
  ]
  - description = "TLS from VPC"
  - from_port = 9000
  - ipv6_cidr_blocks = []
  - prefix_list_ids = []
  - protocol = "tcp"
  - security_groups = []
  - self = false
  - to_port = 9000
},
] -> null
- name = "Jenkins-Security Group" -> null
- owner_id = "672618677785" -> null
- revoke_rules_on_delete = false -> null
- tags = {
  - "Name" = "Jenkins-sg"
} -> null
- tags.all = {
  - "Name" = "Jenkins-sg"
} -> null
- vpc_id = "vpc-0f6bdd74ced5c07c0" -> null
}

Plan: 0 to add, 0 to change, 2 to destroy.
aws_instance.web: Destroying... [id=i-03cbca10897fae32d]
aws_instance.web: Still destroying... [id=i-03cbca10897fae32d, 10s elapsed]
aws_instance.web: Still destroying... [id=i-03cbca10897fae32d, 20s elapsed]
aws_instance.web: Destruction complete after 30s
aws_security_group.Jenkins-sg: Destroying... [id=sg-0a1273b4ae042b673]
aws_security_group.Jenkins-sg: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.

```

In this age of digital transformation, safeguarding your applications is no longer an option but a necessity. The synergy of Terraform, Jenkins, SonarQube, and Trivy empowers us to not only deploy our applications with speed and efficiency but also to do so with an unwavering focus on security. We hope this guide has been a valuable resource in your journey towards embracing DevSecOps principles and securing your Amazon app deployments on AWS. Remember, the world of technology is ever-evolving, and so are the threats that come with it. Stay vigilant, stay informed, and continue to adapt your DevSecOps practices to stay ahead in the realm of secure and efficient application development. Thank you for joining us, and we wish you every success in your DevSecOps endeavours.



Ajay Kumar Yegireddi is a DevSecOps Engineer and System Administrator, with a passion for sharing real-world DevSecOps projects and tasks. **Mr. Cloud Book**, provides hands-on tutorials and practical insights to help others master DevSecOps tools and workflows. Content is designed to bridge the gap between development, security, and operations, making complex concepts easy to understand for both beginners and professionals.

Comments

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

I'm not a robot

reCAPTCHA
[Privacy](#) - [Terms](#)

Post Comment

Uncategorized	AI	AI, DevOps
How to Automate Incident Response : How Q Developer Helped Me Automate a Daily Pain Point	How to Run Docker Model Runner on Ubuntu 24.04	How to Install docker-ai on Ubuntu 24.04
22 July 2025	11 July 2025	15 June 2025

Upskill with Ajay: DevSecOps Mastery

Join Mr Cloud book to master DevSecOps through real-world projects. Learn CI/CD, security integration, automation, and more, gaining hands-on skills for industry-level challenges.



Important Links

[Privacy Policy](#)

[Terms & Conditions](#)

[Contact](#)

Resources

[Blog](#)

[YouTube Channel](#)

