**My Devsecops Project**

**Step:1 Create Jenkins server :**

1. Log in to the **AWS Management Console**.
2. Navigate to **EC2 > Instances > Launch Instances**.
3. Configure the instance:
   * Choose **Amazon Linux 2 AMI**.
   * Select **t3.xlarge** as the instance type.
   * Add storage: **60 GiB gp2**.
   * Configure security group:
     + Allow **SSH (port 22)** and **HTTP (port 8080)**.
   * Assign a key pair.
4. Launch the instance and wait for it to initialize.

**Commands to Run After Launch**

# Update the instance

sudo yum update -y

# Install basic utilities

sudo yum install -y wget git

**step 2; Jenkins Installation**

Run these commands on the **Jenkins server instance**:

1. Add the Jenkins repository:

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo

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

sudo yum upgrade -y

1. Install Java 17:

amazon-linux-extras enable corretto17

sudo yum install -y java-17-amazon-corretto

java -version

1. Install Jenkins:

sudo yum install jenkins -y

1. Enable and start Jenkins:

sudo systemctl enable jenkins

sudo systemctl start jenkins

sudo systemctl status jenkins

**4. Access Jenkins**

1. Open your browser and access Jenkins at:
2. http://<Jenkins-Instance-IP>:8080
3. Unlock Jenkins:
   * Find the initial admin password

cat in /var/lib/jenkins/secrets/initialAdminPassword.

* + Enter the password and proceed.

1. Install the suggested plugins.
2. Create an admin user:
   * Username: Nuthan
   * Password: Panny@123

**Step 3: Configure Jenkins for CI/CD with Additional Tools**

**1. Install Essential Plugins**

1. Go to **Jenkins Dashboard > Manage Jenkins > Manage Plugins**.
2. Navigate to the **Available** tab and search for these plugins:
   * **Git Plugin**: For integrating Git repositories. (pre-installed)
   * **Pipeline Plugin**: For creating declarative or scripted pipelines.

[Pipeline: Stage View](https://plugins.jenkins.io/pipeline-stage-view)

[Pipeline: Declarative Agent AP](https://plugins.jenkins.io/pipeline-model-declarative-agent)

* + **Terraform Plugin**: For running Terraform commands in Jenkins.
  + **HashiCorp Vault Plugin**: To pull secrets from Vault (optional, based on your goals).
  + **SonarQube Scanner Plugin**: For static code analysis integration.
  + **Docker Plugin**: To run Docker-related commands within Jenkins.
  + **Snyk Security Plugin**: For code and dependency scanning.
  + **Ansible Plugin**: To automate configuration management.

**Install Terraform**

Install Terraform by running these commands:

sudo yum install -y yum-utils

sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo

sudo yum -y install terraform

**Install TFscan**

Install TFscan with the following command:

curl -s https://raw.githubusercontent.com/aquasecurity/tfsec/master/scripts/install\_linux.sh | bash

tfsec –version

**Install Trivy**

1. Install Trivy:

curl -sfL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/install.sh | sh

1. Move Trivy to a global directory:

sudo mv /root/bin/trivy /usr/local/bin/trivy

trivy --version

**Why?**: Placing Trivy in /usr/local/bin ensures it is accessible to all users, including Jenkins.

**Updated Steps to Install Snyk CLI**

**Use npm (Recommended)**

If Node.js is installed on your Jenkins server, you can use npm to install Snyk CLI.

1. Install Node.js and npm:

sudo yum install -y nodejs

1. Install Snyk CLI using npm:

npm install -g snyk

1. Verify the installation:

snyk --version

**Configure Global Tools in Jenkins**

1. **Configure Git**:
   * Go to **Manage Jenkins > Global Tool Configuration**.
   * Under **Git**, click **Add Git**.
   * Set the path to /usr/bin/git.
2. **Configure Terraform**:
   * Add Terraform under **Terraform installations**:
     + Go to **Global Tool Configuration**.
     + Add a new Terraform installation and ensure the binary is installed at /usr/bin/
3. **Configure Ansible**:
   * Add Ansible under **Ansible installations**:
     + Go to **Global Tool Configuration**.
     + Add a new Ansible installation and set the path to /usr/bin/

**Create Your First Job to Verify Jenkins**

1. **Create a Freestyle Project**:
   * Go to **Jenkins Dashboard > New Item**.
   * Enter a name (e.g., Verify-Jenkins) and select **Freestyle Project**.
   * Under **Build Steps**, add a shell command:

echo "Jenkins is configured with additional tools!"

tfsec --version

trivy --version

snyk --version

ansible --version

* + Save the job and click **Build Now**.

1. **Check Console Output**:
   * Ensure the console output confirms the installed versions of tfsec, trivy, snyk, and ansible.

**Installing Maven on the Jenkins Server**

**1. Install Maven**

Run the following commands on your Jenkins server to install Maven:

sudo yum install -y maven

**2. Verify the Maven Installation**

Check the Maven version to confirm the installation:

mvn -version

You should see output like this:

Apache Maven 3.x.x

Maven home: /usr/share/maven

Java version: 11, vendor: Amazon.com Inc.

**Configuring Maven in Jenkins**

**1. Add Maven in Jenkins**

1. Go to **Jenkins Dashboard > Manage Jenkins > Global Tool Configuration**.
2. Scroll to the **Maven** section and click **Add Maven**.
3. Configure Maven:
   * **Name**: Maven (or any name you prefer).
   * **Install Automatically**: Uncheck this if Maven is installed manually (as in this case).
   * **Maven Home Directory**: Provide the path where Maven is installed (e.g., /usr/share/maven).

**Create a Jenkins Freestyle Job**

1. Go to **Jenkins Dashboard > New Item**.
2. Enter a job name (e.g., Verify-Jenkins-Maven) and select **Freestyle Project**.
3. Under **Source Code Management**, select **Git**:
   * Enter the **Repository URL**.
   * Provide **Credentials** if required.
   * Specify the branch (e.g., main).

**Add Maven Build Step**

1. Under **Build Steps**, select **Invoke top-level Maven targets**.
2. Configure the Maven build:
   * **Root POM**: pom.xml**(optional)**
   * **Goals and options**: clean install

**Build and Verify**

1. Save the job and click **Build Now**.
2. Check the console output to ensure:
   * The repository is cloned successfully.
   * The pom.xml file is detected.
   * Maven builds the project without errors.

**Integrating SonarQube**

**1. Install SonarQube**

1. Launch a new **EC2 instance** or use an existing one for SonarQube.
2. Instance specifications:
   * **Instance Type**: t3.medium (minimum 4 GiB RAM).
   * **Volume**: 20 GiB.
3. Install and configure SonarQube:
   * **For Amazon Linux**:

sudo yum update -y

install Java:

**amazon-linux-extras enable corretto17**

**sudo yum install -y java-17-amazon-corretto**

wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.9.0.65466.zip

unzip sonarqube-9.9.0.65466.zip

sudo mv sonarqube-9.9.0.65466 /opt/sonarqube

**sudo adduser sonar**

**sudo chown -R sonar:sonar /opt/sonarqube**

**sudo su - sonar**

/opt/sonarqube/bin/linux-x86-64/sonar.sh start

* + Access SonarQube in a browser using: http://<EC2-public-IP>:9000.

**Configuring SonarQube**

**Create a New Project in SonarQube**

1. Log in to SonarQube.
2. Click **Create New Project** and provide the project name (e.g., Sample E-Commerce Project).

**Generate an Authentication Token**

1. Navigate to **My Account > Security**.
2. Under **Generate Tokens**, enter a token name (e.g., SampleProjectToken).
3. Select **Project Analysis** from the dropdown.
4. Click **Generate** and copy the token. (Save it securely; it will not be displayed again.)

**Note :** Exit from sonar user before run any commands

**Installing Sonar Scanner**

**Download and Install Sonar Scanner**

1. **Create a directory for Sonar Scanner:**

mkdir -p /downloads/sonarqube

cd /downloads/sonarqube

1. **Download the latest Sonar Scanner:**

wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-5.0.1.3006-linux.zip

unzip sonar-scanner-cli-5.0.1.3006-linux.zip

sudo mv sonar-scanner-5.0.1.3006-linux /opt/sonar-scanner

1. **Add Sonar Scanner to the PATH:**

vi ~/.bashrc

export PATH="/opt/sonar-scanner/bin:$PATH"

source ~/.bashrc

1. **Verify the installation:**

sonar-scanner --version

Ensure “**SonarQube Scanner for Jenkins**” plugin is installed.

1. **Configure SonarQube in Jenkins**:
   * Go to **Manage Jenkins > Configure System**.
   * Find the **SonarQube Servers** section.
   * Add a new SonarQube server:
     + **Name**: SonarQube.
     + **Server URL**: http://<SonarQube-IP>:9000.
     + **Authentication Token**: Add the token you generated earlier.

**4. Update Jenkins Job**

1. Open the Jenkins job for your project.
2. Add a **SonarQube Analysis Build Step**:
   * Go to **build-steps**
   * Select **"Execute SonarQube Scanner"**.
   * Configure:
     + **Task**: Add the following analysis properties:

sonar.projectKey=<your-project-key>

sonar.projectName=<your-project-name>

sonar.projectVersion=1.0

sonar.sources=src

sonar.java.binaries=target/classes

sonar.login=<token>

* + Replace <your-project-key> and <your-project-name> with the details of your project.

1. **Trigger the Jenkins Job**:
   * Run the job and verify that SonarQube analyzes the code and generates a report.