

DevSecOps Pipeline: Deploying ReactJS on Kubernetes

By: Ansuman Nayak

Email: ansumannayak800@gmail.com

Linkedin: www.linkedin.com/in/ansumannayak-2k

Tools and technologies:

1. AWS EC2 instance (for Jenkins/Sonarqube server):
 - Operating system : Ubuntu(22.04) T2 Large Instance
 - Storage : 15 GB

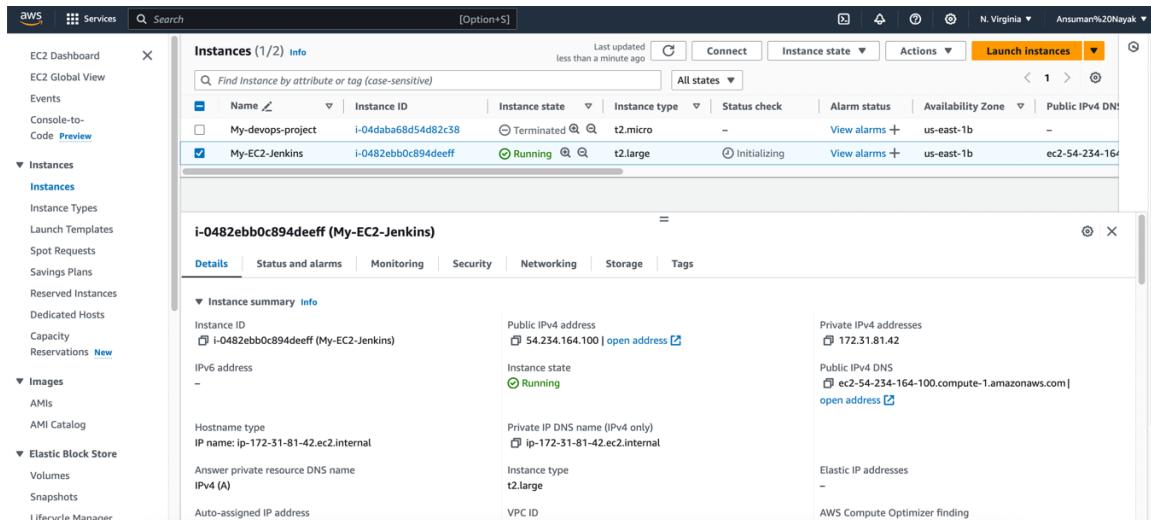
For Kubernetes Master/Worker :

- Operating system : Ubuntu 20.04 T2.Meduim Instance
- Storage : 15 GB

2. Jenkins
3. Sonarqube
4. Trivy
5. Docker
6. Kubernetes

Steps:

1. Launch an Ubuntu(22.04) T2 Large Instance for setting up Jenkins.

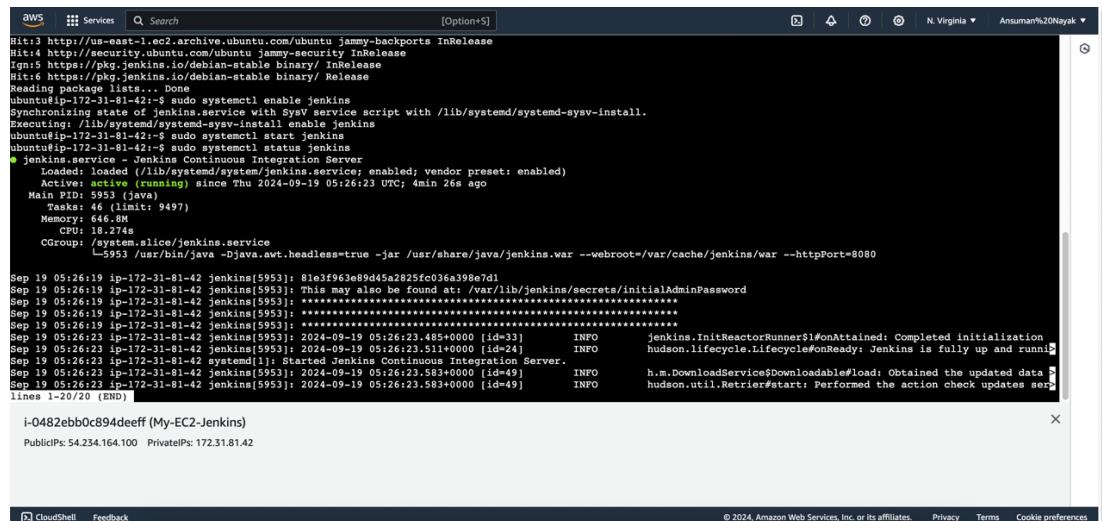


- Install Jenkins on the EC2 instance


```
sudo apt update -y
sudo apt install openjdk-17-jre -y
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

```
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
```

```
sudo apt-get install jenkins -y
sudo systemctl start jenkins
sudo systemctl status Jenkins
```



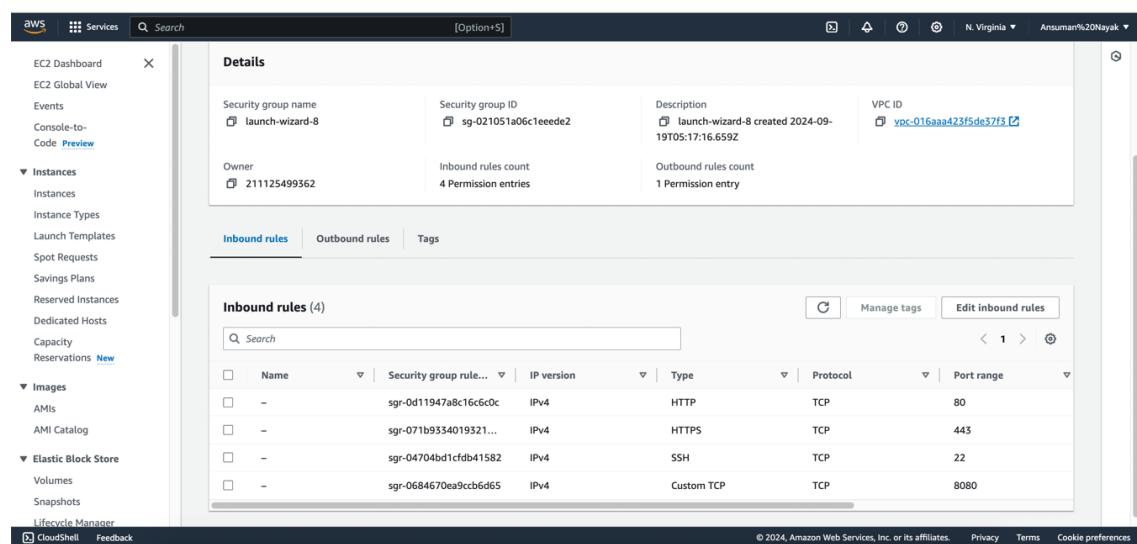
```
aws Services Search [Options+S]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Reading package lists... Done
ubuntu@ip-172-31-81-42:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-81-42:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-81-42:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2024-09-19 05:26:23 UTC; 4min 26s ago
       Main PID: 5953 (java)
          Tasks: 46 (limit: 9497)
            Memory: 646.8M
              CPU: 18.274s
            CGroup: /system.slice/jenkins.service
                    └─5953 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Sep 19 05:26:19 ip-172-31-81-42 jenkins[5953]: 81e3f963e89d45a2825fc036a398b7d1
Sep 19 05:26:19 ip-172-31-81-42 jenkins[5953]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Sep 19 05:26:19 ip-172-31-81-42 jenkins[5953]: ****
Sep 19 05:26:19 ip-172-31-81-42 jenkins[5953]: ****
Sep 19 05:26:19 ip-172-31-81-42 jenkins[5953]: ****
Sep 19 05:26:23 ip-172-31-81-42 jenkins[5953]: 2024-09-19 05:26:23.405+0000 [id=33] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
Sep 19 05:26:23 ip-172-31-81-42 jenkins[5953]: 2024-09-19 05:26:23.511+0000 [id=24] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
Sep 19 05:26:23 ip-172-31-81-42 jenkins[5953]: Started Jenkins Continuous Integration Server.
Sep 19 05:26:23 ip-172-31-81-42 jenkins[5953]: 2024-09-19 05:26:23.583+0000 [id=49] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data
Sep 19 05:26:23 ip-172-31-81-42 jenkins[5953]: 2024-09-19 05:26:23.583+0000 [id=49] INFO hudson.util.Retrier#start: Performed the action check updates ser
Lines 1-20/20 (END)

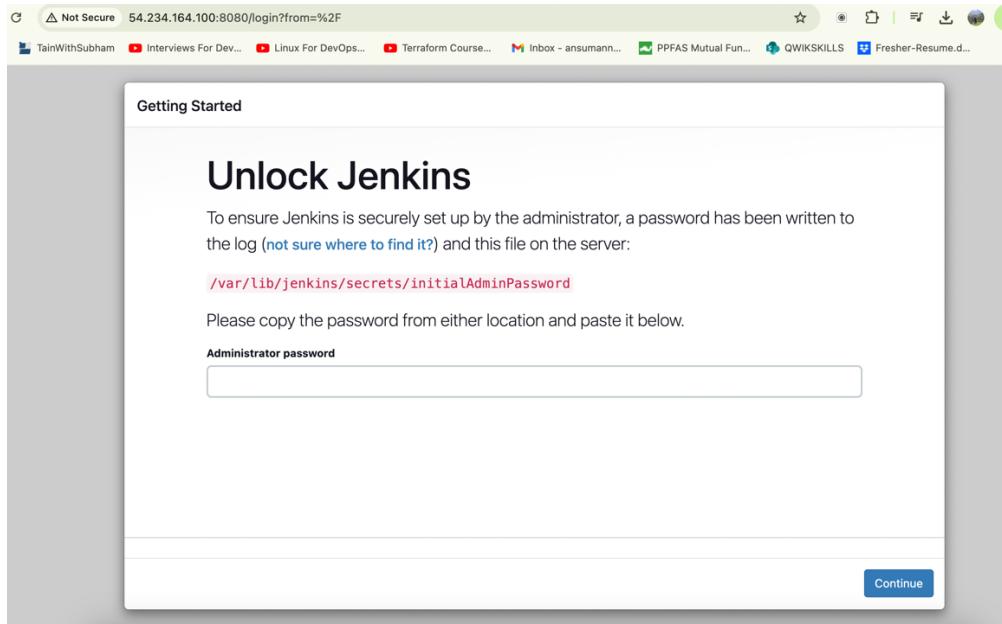
i-0482ebb0c894deeff (My-EC2-Jenkins)
PublicIPs: 54.234.164.100 PrivateIPs: 172.31.81.42
```

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

- Go to AWS EC2, Security Group and open Inbound Port 8080.



Type <EC2 Public IP Address:8080> in web browser to access Jenkins server.



Copy the displayed path and paste it on the EC2 terminal as displayed below.
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Copy the token and paste it in Jenkins administrator password section.

A screenshot of an AWS CloudWatch terminal session. The terminal window title is 'aws' and the tab is 'Services'. The search bar contains 'Search [Option+S]'. The command entered is 'sudo cat /var/lib/jenkins/secrets/initialAdminPassword'. The output shows a long, complex password: '81e3f963e89d45a82825fc036a398e7d1'. The terminal prompt is 'ubuntu@ip-172-31-81-42:~\$'. The status bar at the bottom shows the instance ID 'i-0482ebb0c894deeff (My-EC2-Jenkins)' and the public and private IP addresses 'Public IPs: 54.234.164.100 Private IPs: 172.31.81.42'.

```
aws Services Search [Option+S]
[Output redacted]
ubuntu@ip-172-31-81-42:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
81e3f963e89d45a82825fc036a398e7d1
ubuntu@ip-172-31-81-42:~$
```

i-0482ebb0c894deeff (My-EC2-Jenkins)
Public IPs: 54.234.164.100 Private IPs: 172.31.81.42

4. Install the suggested plugins

The screenshot shows the Jenkins 'Getting Started' page. At the top, there's a 'Getting Started' button. Below it is a large heading 'Getting Started'. A grid of plugin recommendations follows, divided into four columns:

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	⌚ Workspace Cleanup	⌚ Ant	⌚ Gradle
⌚ Pipeline	⌚ GitHub Branch Source	⌚ Pipeline: GitHub Groovy Libraries	⌚ Pipeline Graph View
⌚ Git	⌚ SSH Build Agents	⌚ Matrix Authorization Strategy	⌚ PAM Authentication
⌚ LDAP	⌚ Email Extension	⌚ Mailer	⌚ Dark Theme

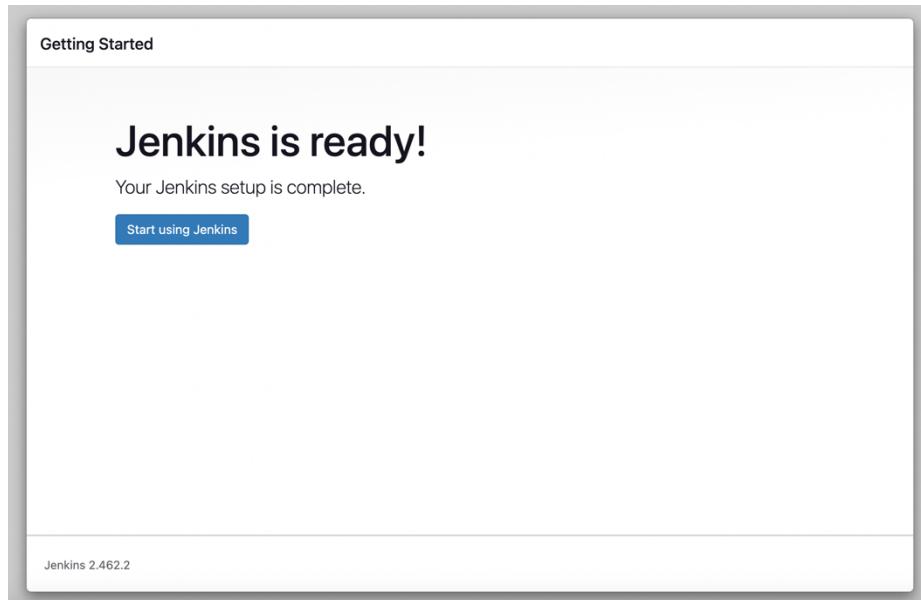
To the right of the grid, a vertical scrollable list of additional Jenkins APIs is visible:

- ** Credentials
- ** Plain Credentials
- ** Variant
- ** SSH Credentials
- Credentials Binding**
- ** SCM API
- ** Pipeline: API
- ** commons-lang3 v3.x Jenkins API
- Timestamper**
- ** Caffeine API
- ** Script Security
- ** JavaBeans Activation Framework (JAF) API
- ** JAXB
- ** SnakeYAML API
- ** JSON API
- ** Jackson 2 API
- ** commons-text API
- ** Pipeline: Supporting APIs
- ** Plugin Utilities API
- ** Font Awesome API

At the bottom left is the text 'Jenkins 2.462.2'.

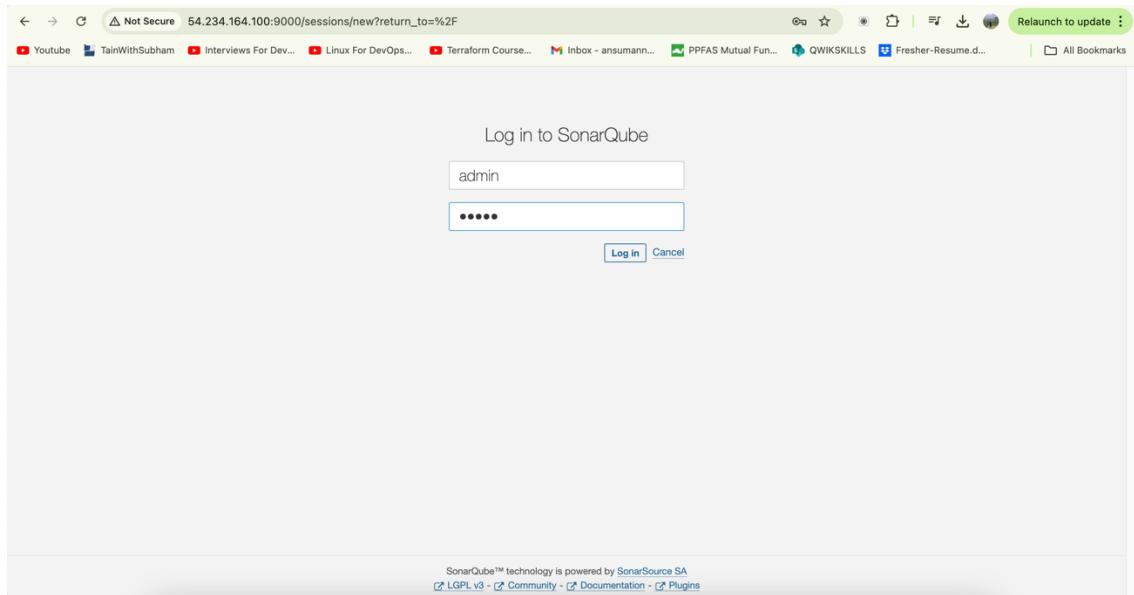
5. Now create a user in Jenkins with own username, password and email.

The screenshot shows the Jenkins 'Instance Configuration' page. At the top, there's a 'Getting Started' button. Below it is a large heading 'Instance Configuration'. A 'Jenkins URL:' input field contains the value 'http://54.234.164.100:8080/'. A tooltip for this field explains its purpose: 'The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.' Another tooltip states: 'The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.' At the bottom right are 'Not now' and 'Save and Finish' buttons. The footer at the bottom left says 'Jenkins 2.462.2'.



6. You're doing really well! Let now install Docker and Sonarqube

```
sudo apt-get update sudo apt-get install docker.io -y sudo usermod -aG docker $USER newgrp docker sudo chmod 777 /var/run/docker.sock
```
7. After the docker installation, we will install sonarqube using a Sonarqube container (Remember to add 9000 ports in the security group as done for Jenkins).
 - `docker run -d --name sonar -p 9000:9000 sonarqube:lts-community`
8. Enter public ip of EC2 instance and the port number in web browser.
`<public-ip-of-EC2:9000>`
9. Enter username and password, click on login and change password
Note: For the first time the default credentials will be username= admin and password =admin, following this update new password.



10. Our infrastructure needs some security, so now we will 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 - darmor | sudo tee /usr/share/keyrings/trivy.gpg > /dev/null
- echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb \$(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list
- sudo apt-get update
- sudo apt-get install trivy -y

11. Now install plugins like JDK, Sonarqube Scanner, NodeJS, OWASP dependency check.

Install plugins

Go to Manage Jenkins → Plugins → Available Plugins → Install below plugins

→ Eclipse Temurin Installer (Install without restart)

→ Sonarqube Scanner (Install without restart)

→ NodeJS Plugin (Install Without restart)

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 2.17.2	7 mo 2 days ago
	External Site/Tool Integrations	
	Build Reports	
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.		

12. 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' section of the Jenkins 'Tools' configuration. A new JDK instance named 'jdk17' is being added. The 'Install automatically' checkbox is checked. Under 'Install from adoptium.net', the version 'jdk-17.0.8.1+1' is selected. There is a 'Save' button at the bottom.

The screenshot shows the 'NodeJS' configuration section. A new NodeJS instance named 'node16' is being added. The 'Install automatically' checkbox is checked. Under 'Install from nodejs.org', the version 'NodeJS 16.2.0' is selected. There is a 'Save' button at the bottom.

13. Create a Job create a job as Devsecops_demo Name, select pipeline and click ok.

14. Configure Sonar Server in Manage Jenkins

Copy the Public IP of your EC2 instance, Sonarqube works on port 9000, <Public IP EC2>:9000.

Goto your Sonarqube Server.

Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → click on Generate Token → Copy the token.

Administration

Configuration ▾ Security ▾ Projects ▾ System Marketplace

Users Create and administer individual users.

Search by login or name...

	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	1

1 of 1 shown

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 - v9.9.6 (build 92038) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)

Goto Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text.

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

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

New credentials

Kind

Secret text

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

Secret

ID ?
Sonar_Token

Description ?
Sonar_Token

Create

Dashboard > Manage Jenkins > System >

SonarQube installations
List of SonarQube installations

Name
Sonar_Server

Server URL
Default is http://localhost:9000
http://54.234.164.100:9000

Server authentication token
SonarQube authentication token. Mandatory when anonymous access is disabled.
Sonar_Token

+ Add ▾

Advanced ▾

Save Apply

15. We will install a sonar scanner in the tools.

The screenshot shows the Jenkins Manage Jenkins > Tools page. Under the SonarQube Scanner installations section, there is a configuration card for 'SonarScanner'. It has a 'Name' field set to 'Sonar_Scanner' and an 'Install automatically' checkbox checked. Below this, under 'Install from Maven Central', a dropdown menu is open, showing 'SonarQube Scanner 5.0.1.3006' as the selected version. There is also a 'Add Installer' button. At the bottom of the card are 'Save' and 'Apply' buttons.

16. In the Sonarqube Dashboard add a quality gate also Administration → Configuration → Webhooks

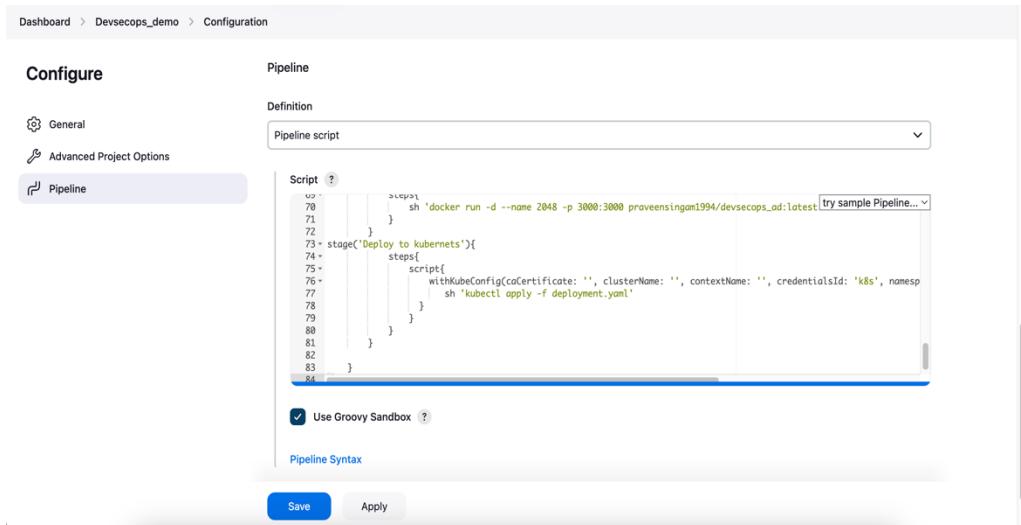
The screenshot shows the SonarQube Administration > Configuration > Webhooks page. A modal window titled 'Create Webhook' is open. It contains fields for 'Name' (set to 'My-EC2-Jenkins'), 'URL' (set to 'http://54.234.164.100:8080/sonarqube-webhook/'), and 'Secret' (left empty). A note at the bottom of the modal says: 'Embedded database should be used for evaluation purposes only. The embedded database will not scale, it will not support upgrading your data out of it into a different database engine.' At the bottom right of the modal are 'Create' and 'Cancel' buttons.

The screenshot shows the SonarQube Administration > Configuration > Webhooks page. The table lists the created webhook:

Name	URL	Has secret?	Last delivery	Actions
My-EC2-Jenkins	http://54.234.164.100:8080/sonarqube-webhook/	No	Never	

A note at the bottom of the page says: '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.'

17. Let's go to our Pipeline and add the script in our Pipeline Script.
<https://github.com/AWS-AZURE-Bootcamp5/Devsecops-Project1/blob/main/Jenkinsfile1>



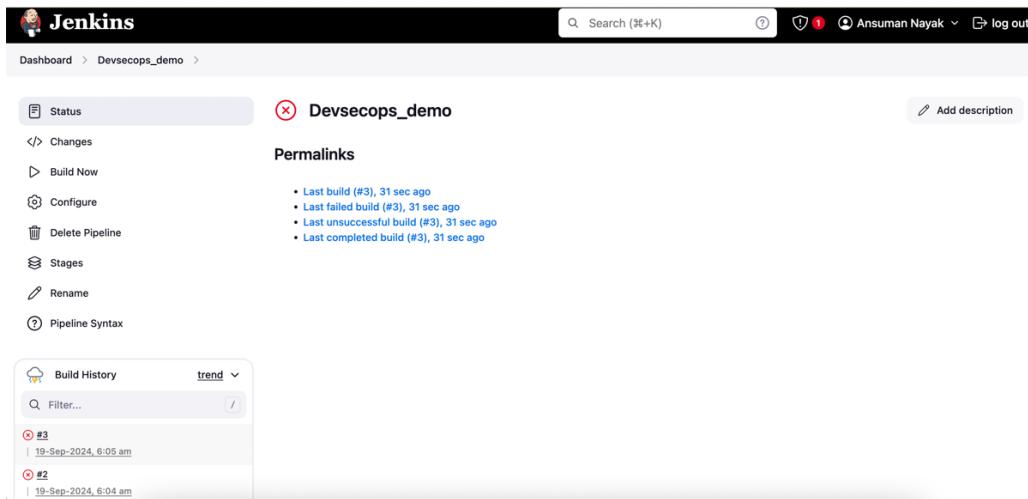
The screenshot shows the Jenkins Pipeline configuration page. The pipeline is named 'Pipeline'. The 'Script' section contains the following Groovy code:

```

1  stages {
2     stage('Build') {
3         steps {
4             sh 'docker run -d --name 2048 -p 3000:3000 proveensingam1994/devsecops_ad:latest'
5         }
6     }
7     stage('Deploy to kubernetes'){
8         steps{
9             script{
10                 withKubeConfig(caCertificate: '', clusterName: '', contextName: '', credentialsId: 'k8s', namespace: '')
11                 sh 'kubectl apply -f deployment.yaml'
12             }
13         }
14     }
15 }
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

```

The 'Use Groovy Sandbox' checkbox is checked. At the bottom are 'Save' and 'Apply' buttons.

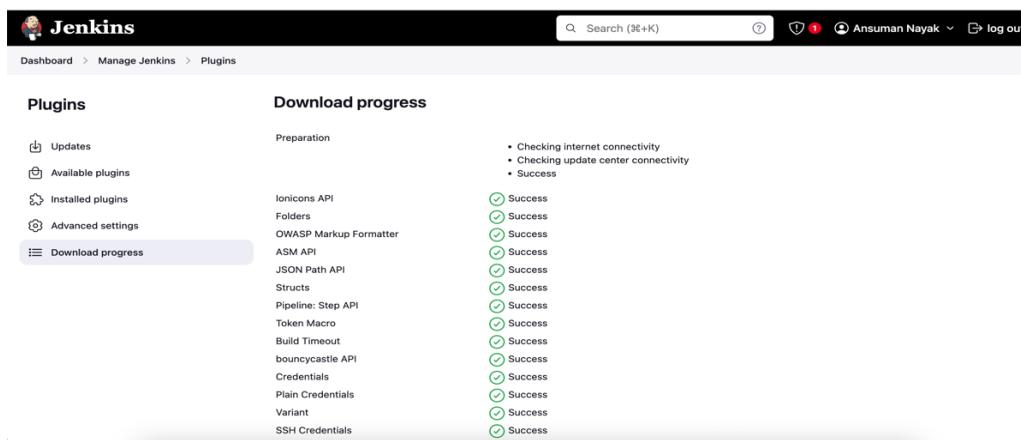


The screenshot shows the Jenkins dashboard for the 'Devsecops_demo' pipeline. The pipeline has four builds:

- Last build (#3), 31 sec ago
- Last failed build (#3), 31 sec ago
- Last unsuccessful build (#3), 31 sec ago
- Last completed build (#3), 31 sec ago

The 'Build History' section shows two builds: #3 (failed) and #2 (completed).

18. Install OWASP Dependency Check Plugins
Go to Dashboard → Manage Jenkins → Plugins → OWASP Dependency-Check.
Click on it and install it without restart.



The screenshot shows the Jenkins Manage Jenkins > Plugins screen. The 'Download progress' section displays the status of plugin installations:

Plugin	Status
Ionicons API	Success
Folders	Success
OWASP Markup Formatter	Success
ASM API	Success
JSON Path API	Success
Structs	Success
Pipeline: Step API	Success
Token Macro	Success
Build Timeout	Success
bouncycastle API	Success
Credentials	Success
Plain Credentials	Success
Variant	Success
SSH Credentials	Success

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

Go to Dashboard → Manage Jenkins → Tools



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

19. 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:

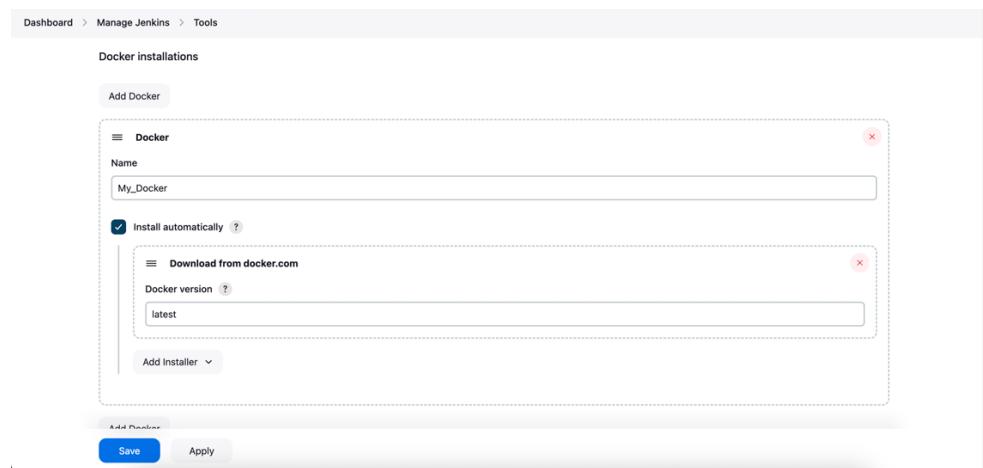
Docker

Docker Commons

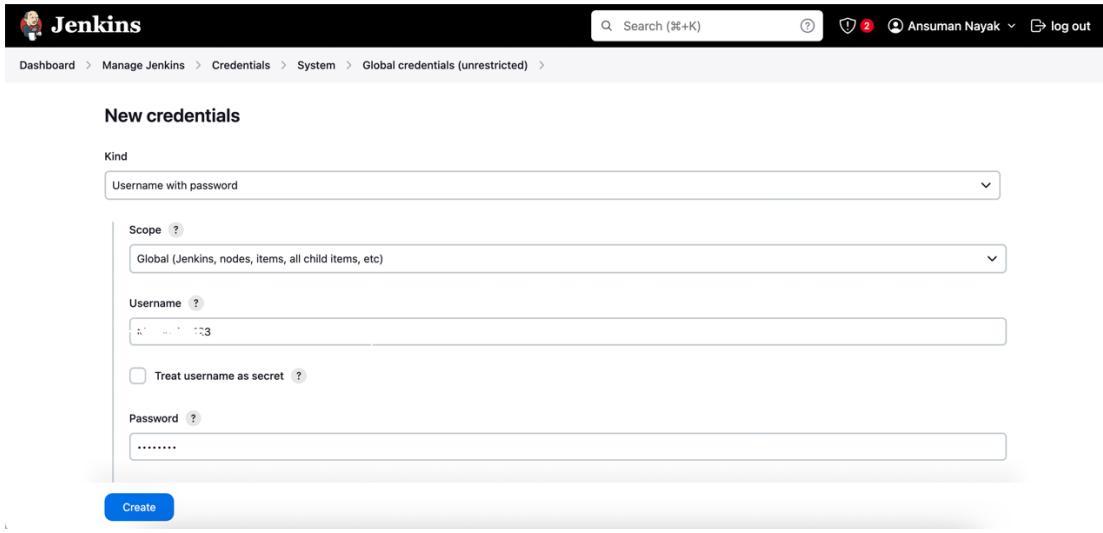
Docker Pipeline

Docker API

Docker-build-step

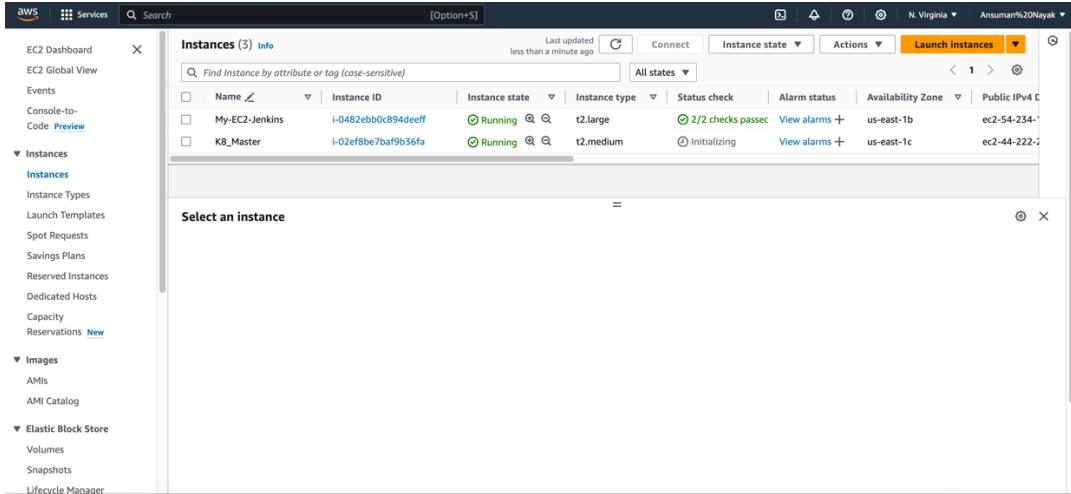


Now, goto Dashboard → Manage Jenkins → Tools
Add Docker Hub Username and Password under Global Credentials



20. Kubernetes setup

Deploy 2 Ubuntu 20.04 instances for kubernetes master and worker. T2.Medium 15 GB
Install Kubectl on Jenkins machine aswell.



- sudo apt update
- sudo apt install curl -y
- curl -LO https://dl.k8s.io/release/\$(curl -L -s <https://dl.k8s.io/release/stable.txt>)/bin/linux/amd64/kubectl
- sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl - kubectl version --client

21. Set hostname for Master node
 - sudo hostnamectl set-hostname K8s-Master
22. Set hostname for Master node
 - sudo hostnamectl set-hostname K8s-Worker
23. Install Kubeadm/Kubelet/kubectl on Master and Worker
 - sudo apt-get update
 - sudo apt-get install -y docker.io
 - sudo usermod -aG docker Ubuntu
 - newgrp docker
 - sudo chmod 777 /var/run/docker.sock
 - sudo curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
 - sudo tee /etc/apt/sources.list.d/kubernetes.list <<EOF deb https://apt.kubernetes.io/ kubernetes-xenial main # 3lines same command
 - EOF
 - sudo apt-get update
 - sudo apt-get install -y kubelet kubeadm kubectl
 - sudo snap install kube-apiserver
24. On Master node
 - sudo kubeadm init --pod-network-cidr=10.244.0.0/16
 - mkdir -p \$HOME/.kube
 - sudo cp -i /etc/kubernetes/admin.conf \$HOME/.kube/config
 - sudo chown \$(id -u):\$(id -g) \$HOME/.kube/config
 - kubectl apply -f <https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kubeflannel.yml>
25. You will get the token and the command like below after executing the above command
 - sudo kubeadm join <master-node-ip>:<master-node-port> --token <token> --discovery-token-ca-cert-hash <hash>

Copy above token and paste in worker node

26. Copy the config file from K8 MASTER to the local laptop and save it with a name secret-file.txt and use this at the Kuberenetes credential section.

27. Install Kubernetes plugin



Jenkins

Dashboard > Manage Jenkins > Plugins

Plugins

[Updates](#)

[Available plugins](#) **Selected**

[Installed plugins](#)

[Advanced settings](#)

Search: kubernetes

Install	Name	Released
<input checked="" type="checkbox"/>	Kubernetes 4288.v1710f9d0c854 Cloud Providers Cluster Management Kubernetes Agent Management	2 days 4 hr ago
<input checked="" type="checkbox"/>	Kubernetes Client API 6.10.0-240.v57880ce8b_0b_2 kubernetes Library plugins (for use by other plugins)	7 mo 27 days ago
<input checked="" type="checkbox"/>	Kubernetes Credentials 190.v03c305394deb_... kubernetes credentials	1 day 13 hr ago
<input checked="" type="checkbox"/>	Kubernetes CLI 1.12.1 kubernetes	1 yr 0 mo ago
<input type="checkbox"/>	Kubernetes Credentials Provider 1.262.v2670ef7ea_0c5	

28. Go to manage Jenkins → manage credentials → Click on Jenkins global → add credentials.

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

New credentials

Kind

Secret file ▼

Scope ?

Global (Jenkins, nodes, items, all child items, etc) ▼

File

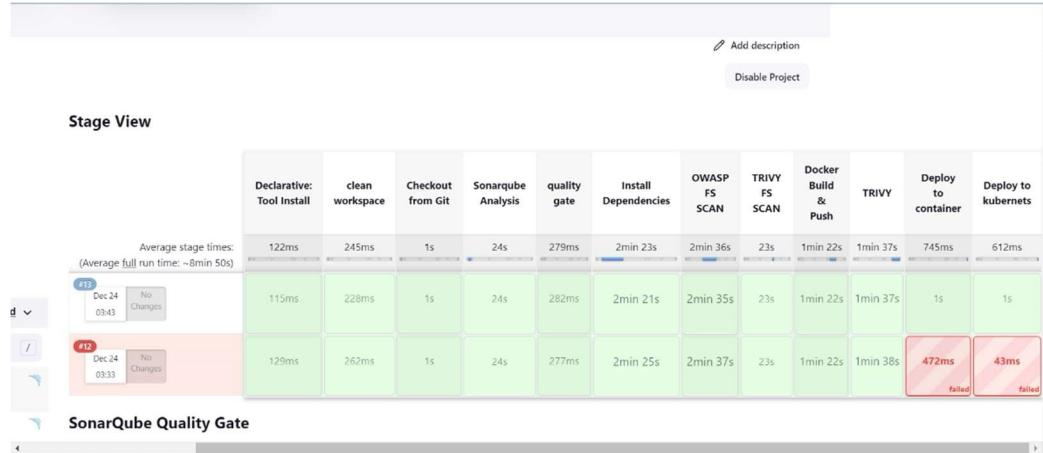
secret-file.txt

ID ?

k8s

Create

29. Run the pipeline



Hurray!!! We have successfully deployed the CI/CD pipeline. Congratulations, our efforts have paid off.

30. In the Kubernetes cluster give this command

- `kubectl get all`
- `kubectl get svc`

31. Access from a Web browser with

- `<public-ip-of-slave:service port>`

Please terminate your instance to avoid extra billing cost.

References: <https://www.youtube.com/live/mdbS5Hu1NnQ?si=T3uu4VBVhspEQ6PR>