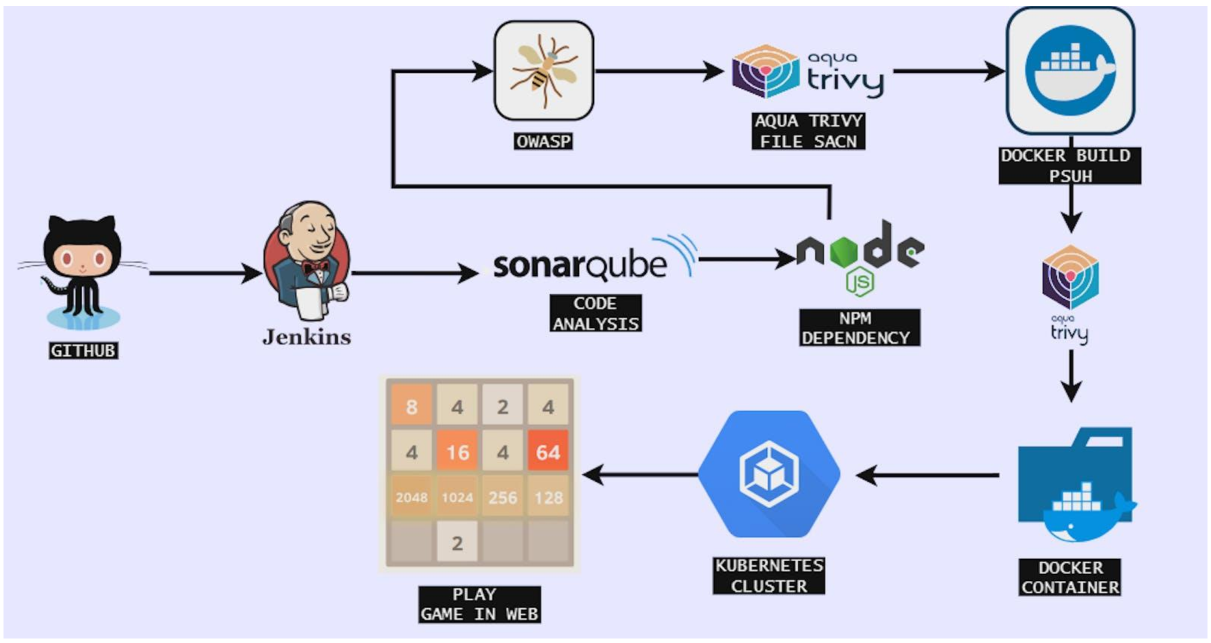
DevSecOps: Deploying the 2048 Game on Docker and Kubernetes with Jenkins CI/CD



Step 1 — Launch an Ubuntu(22.04) T2 Large Instance

Step 2 — Install Jenkins, Docker and Trivy. Create a Sonarqube Container using Docker.

Step 3 — Install Plugins like JDK, Sonarqube Scanner, Nodejs, and OWASP Dependency Check.

Step 4 — Create a Pipeline Project in Jenkins using a Declarative Pipeline

Step 5 — Install OWASP Dependency Check Plugins

Step 6 — Docker Image Build and Push

Step 7 — Deploy the image using Docker

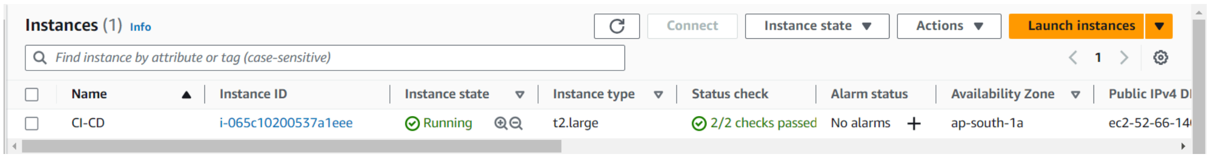
Step 8 — Kubernetes master and slave setup on Ubuntu (20.04)

Step 9 — Access the Game on Browser.

Step 10 — Terminate the AWS EC2 Instances

STEP1:Launch an Ubuntu(22.04) T2 Large Instance

Launch an AWS T2 Large Instance. Use the image as Ubuntu. You can create a new key pair or use an existing one. Enable HTTP and HTTPS settings in the Security Group and open all ports (not best case to open all ports but just for learning purposes it's okay).



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DI
CI-CD	i-065c10200537a1eee	Running	t2.large	2/2 checks passed	No alarms	ap-south-1a	ec2-52-66-14

Step 2 — Install Jenkins, Docker and Trivy

2A — To Install Jenkins

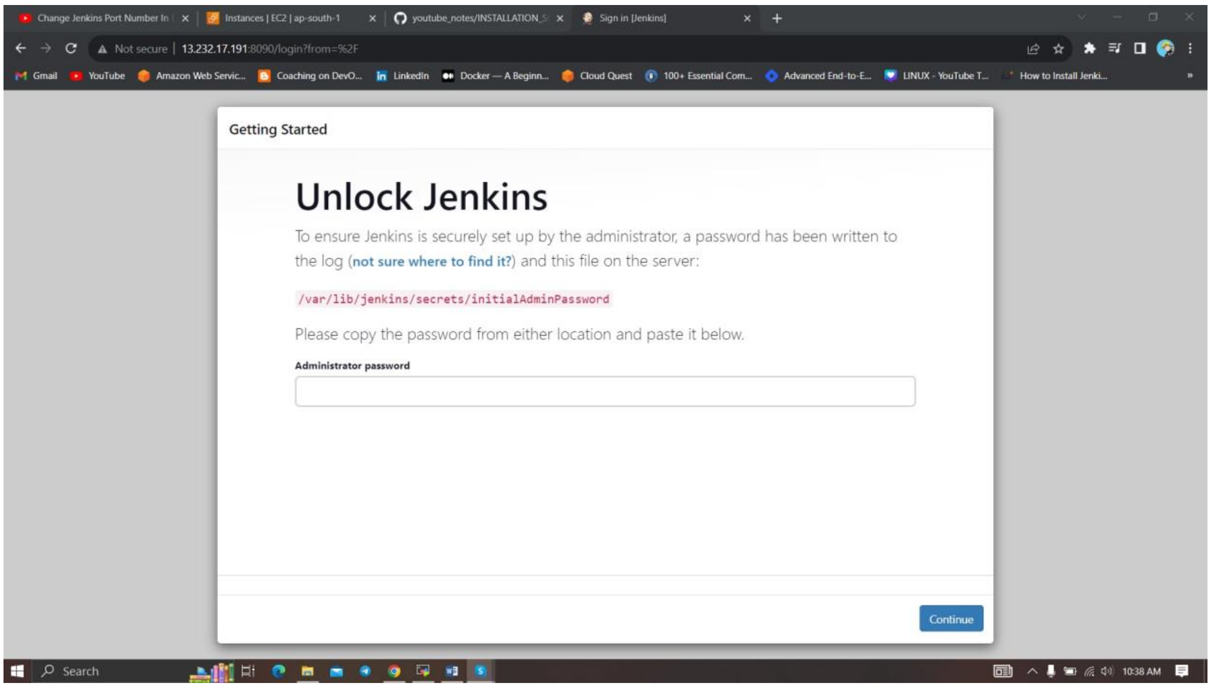
Connect to your console, and enter these commands to Install Jenkins

```
vi jenkins.sh

#!/bin/bash

sudo apt update -y

#sudo apt upgrade -y

wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public | tee
/etc/apt/keyrings/adoptium.asc

echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc]
https://packages.adoptium.net/artifactory/deb $(awk -F= '/^VERSION_CODENAME/{print$2}'
/etc/os-release) main" | 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 \
          /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 update -y

sudo apt-get install jenkins -y

sudo systemctl start jenkins

sudo systemctl status jenkins
```

Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

<EC2 Public IP Address:8080>

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



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:

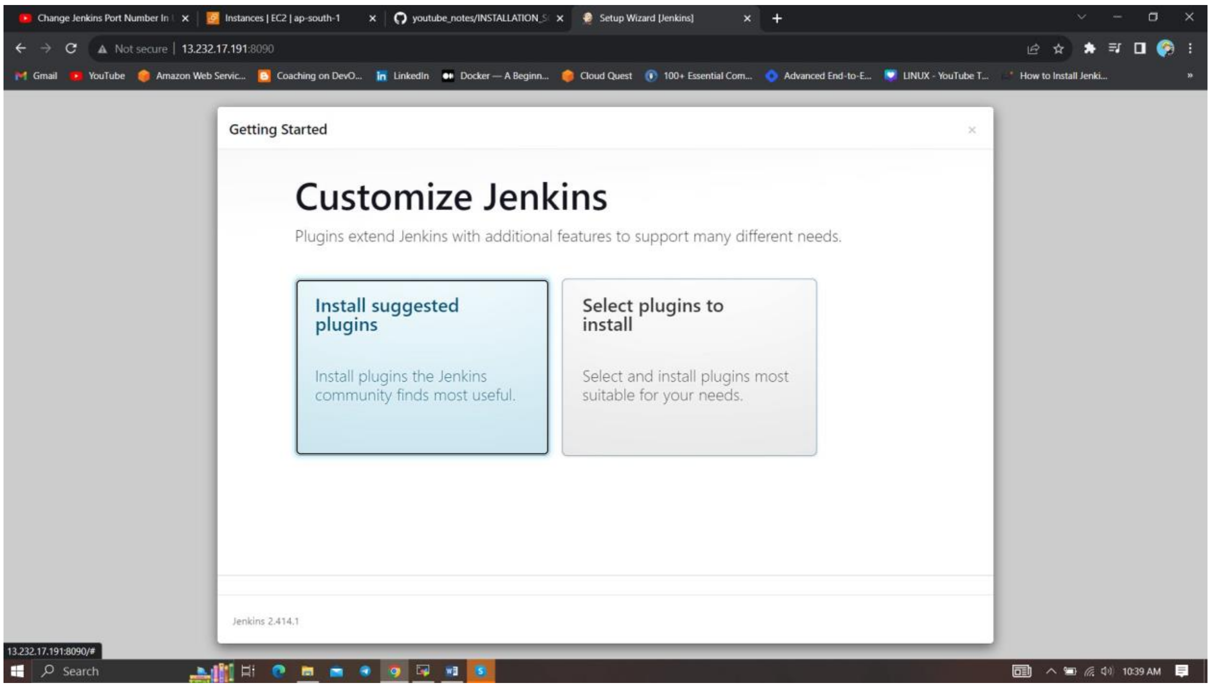
/var/lib/jenkins/secrets/initialAdminPassword

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

Administrator password

Continue

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



Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

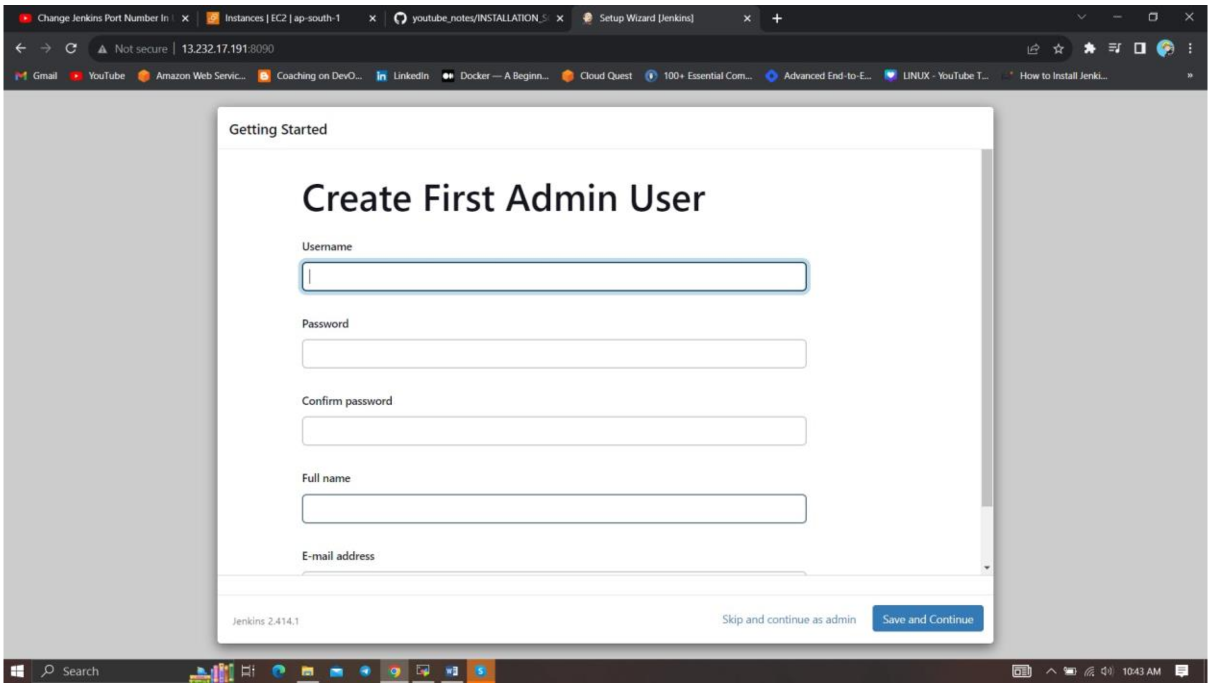
Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

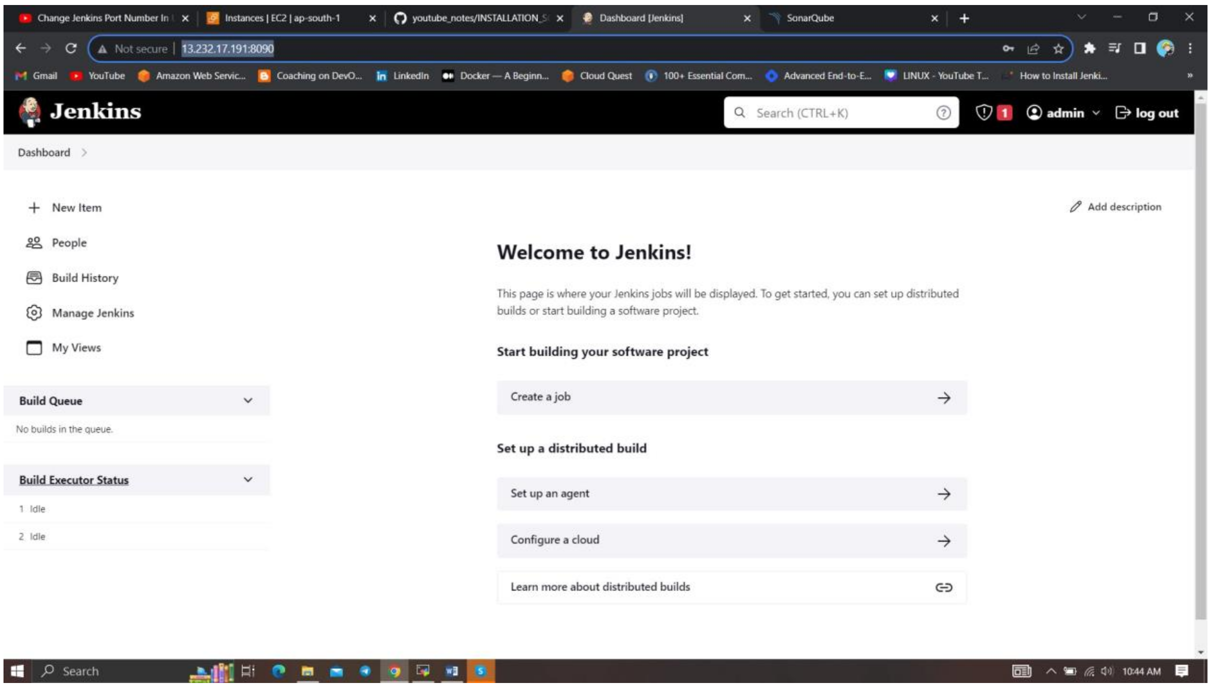
Jenkins 2.414.1

Jenkins will now get installed and install all the libraries.



Create a user click on save and continue.

Jenkins Getting Started Screen.



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

Set up a distributed build

No builds in the queue.

1 Idle

2 Idle

2B — Install Docker

sudo apt-get update

sudo apt-get install docker.io -y

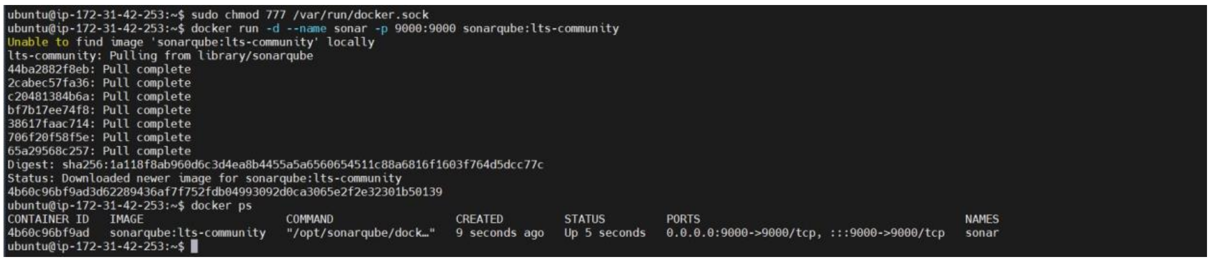
sudo usermod -aG docker $USER   #my case is ubuntu

newgrp docker

sudo chmod 777 /var/run/docker.sock

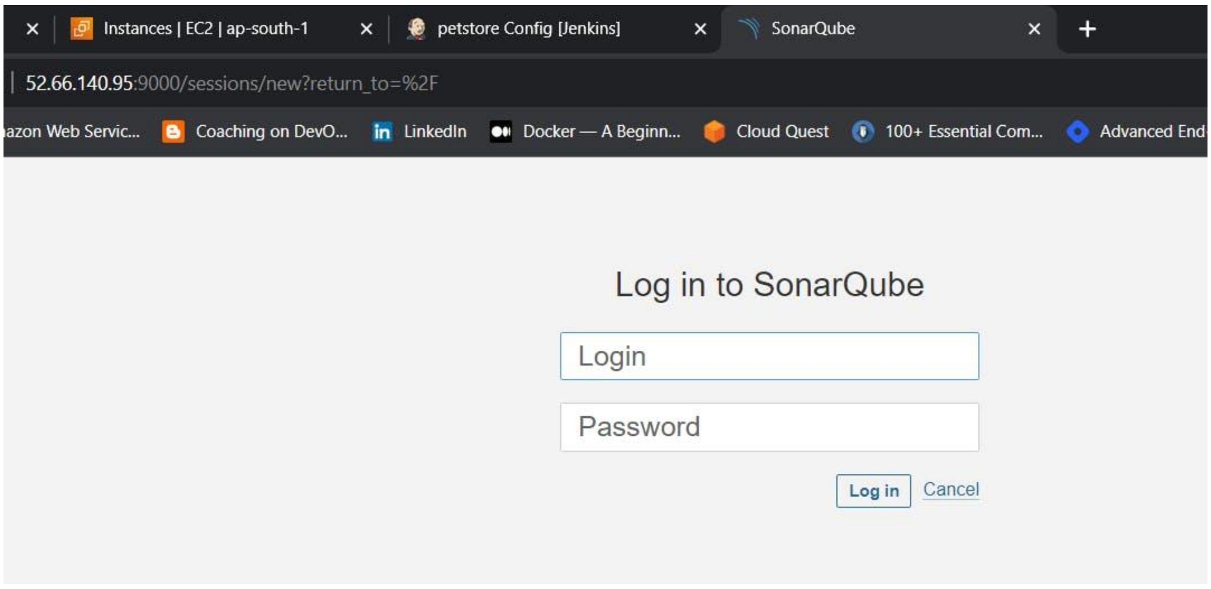
After the docker installation, we create a sonarqube container (Remember to add 9000 ports in the security group).

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



```
ubuntu@ip-172-31-42-253:~$ sudo chmod 777 /var/run/docker.sock
ubuntu@ip-172-31-42-253:~$ 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
44ba2882f8eb: Pull complete
2cabec57fa36: Pull complete
c20481384b6a: Pull complete
bf7b17ee74f8: Pull complete
38617faac714: Pull complete
706f20f58f5e: Pull complete
65a29568c257: Pull complete
Digest: sha256:1a118f8ab960d6c3d4ea8b4455a5a6560654511c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b60c96bf9ad3d62289436af7f752fdb04993092d0ca3065e2f2e32301b50139
ubuntu@ip-172-31-42-253:~$ docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS                                       NAMES
4b60c96bf9ad   sonarqube:lts-community   "/opt/sonarqube/dock…"   9 seconds ago   Up 5 seconds   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   sonar
ubuntu@ip-172-31-42-253:~$ 
```

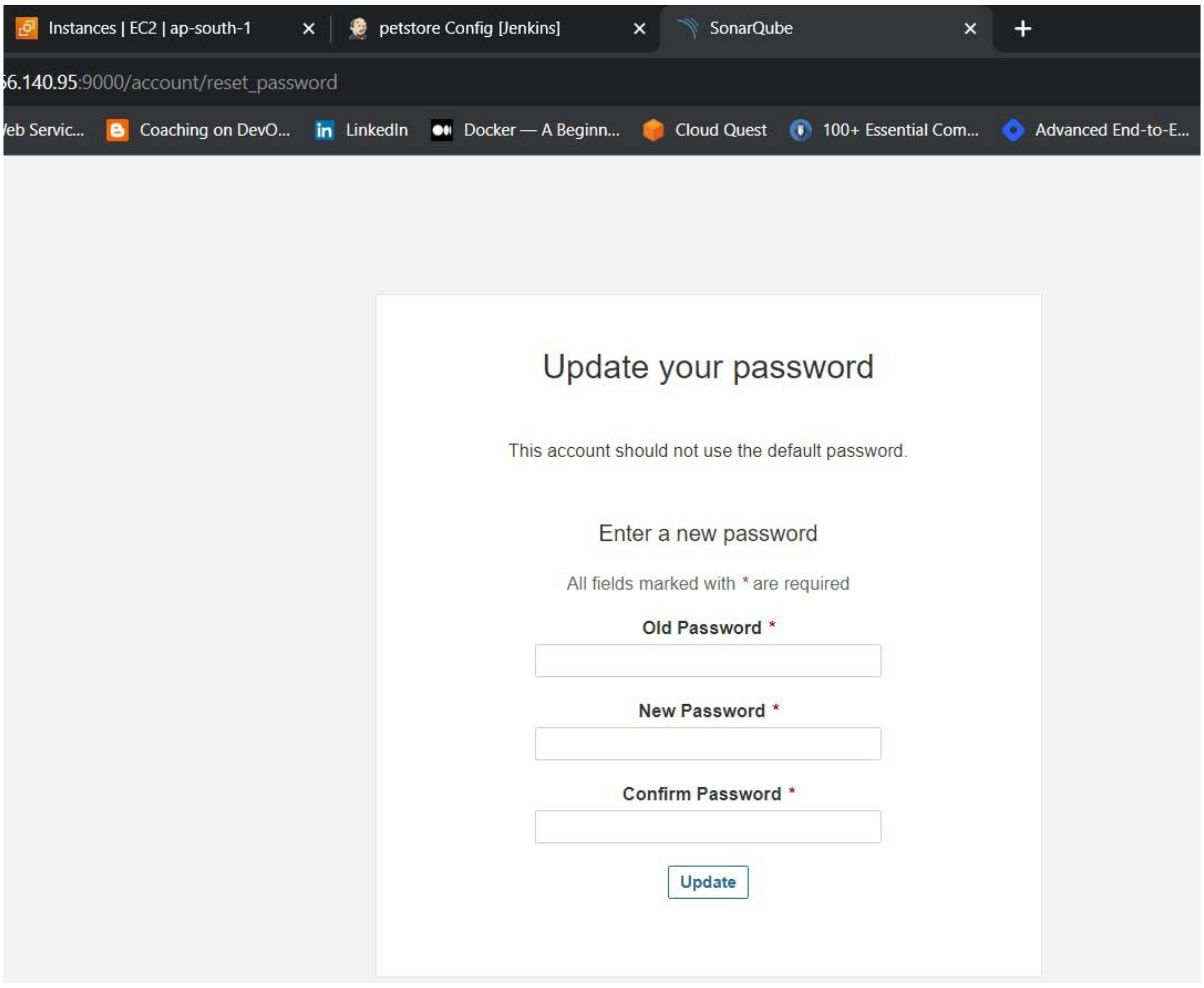
Now our sonarqube is up and running



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
eb 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

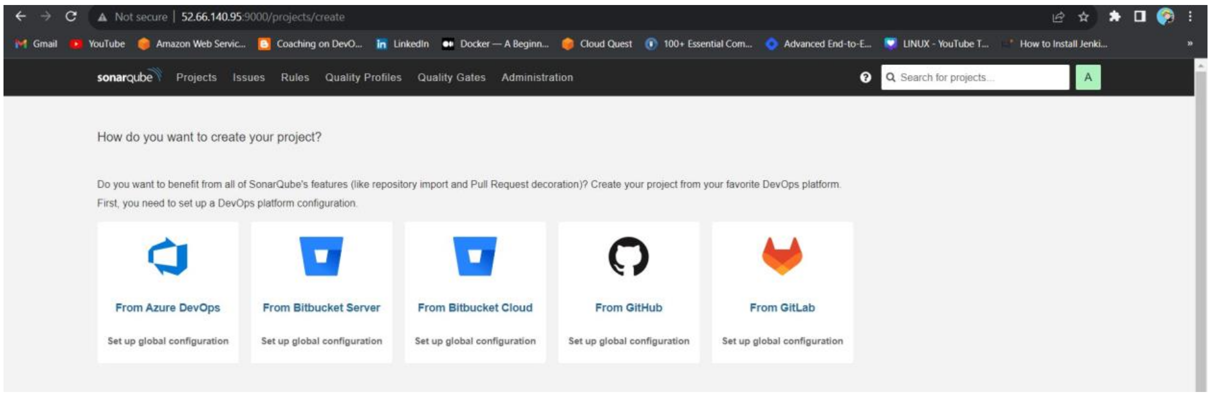
Old Password *

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

2C — Install Trivy

vi trivy.sh

```
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
/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
```

Next, we will log in to Jenkins and start to configure our Pipeline in Jenkins

Step 3 — Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

3A — Install Plugin

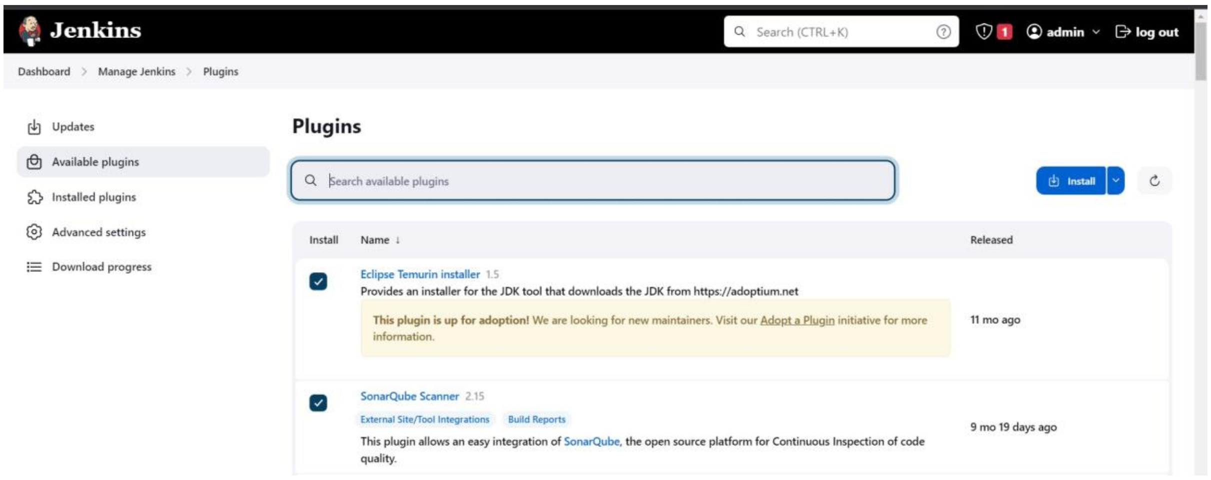
Goto Manage Jenkins →Plugins → Available Plugins →

Install below plugins

1 → Eclipse Temurin Installer (Install without restart)

2 → SonarQube Scanner (Install without restart)

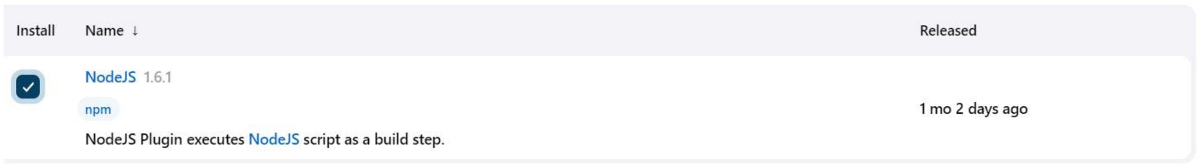
3 → NodeJs Plugin (Install Without restart)



Plugins

Search available plugins

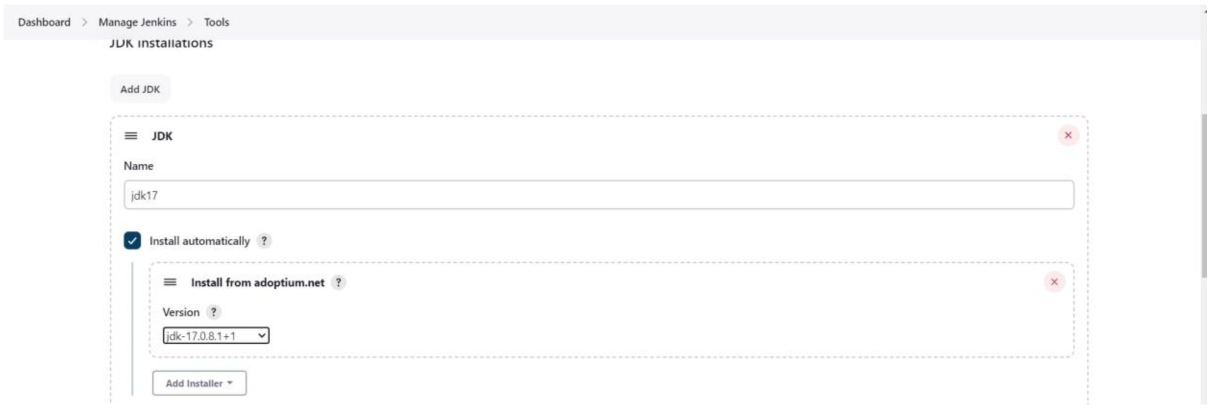
Install	Name ↓	Released
✓	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	11 mo ago
✓	SonarQube Scanner 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



Install	Name ↓	Released
✓	NodeJS 1.6.1 npm NodeJS Plugin executes NodeJS script as a build step.	1 mo 2 days ago

3B — Configure Java and Nodejs in Global Tool Configuration

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

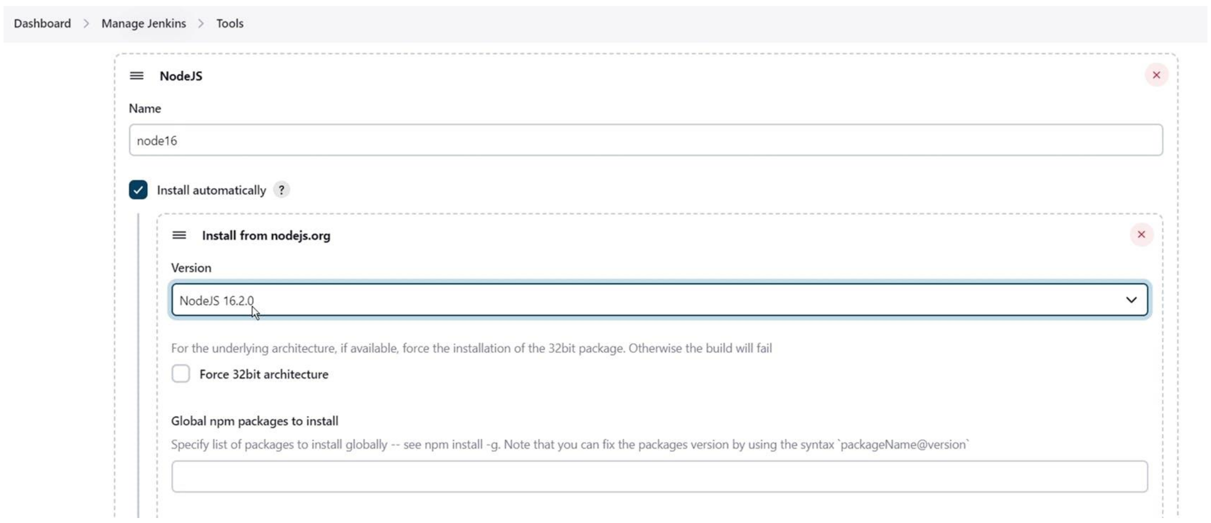


Name: jdk17

✓ Install automatically

Install from adoptium.net

Version: jdk-17.0.8.1+1



Name: node16

Install automatically ✓

Version: NodeJS 16.2.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

Global npm packages to install

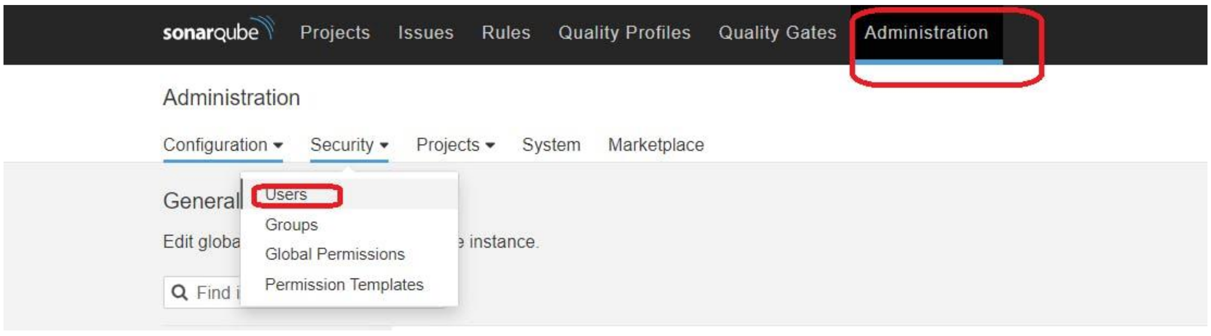
Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

3C — Create a Job

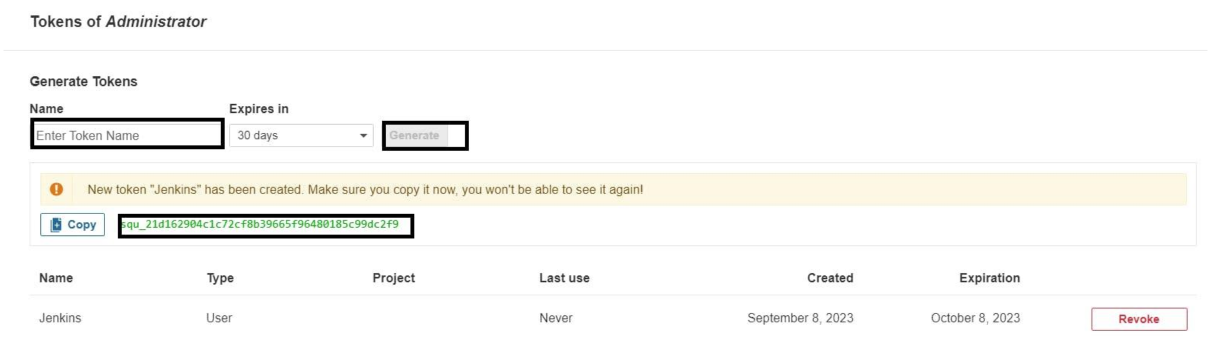
create a job as 2048 Name, select pipeline and click on ok

Step 4 — 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



Create a token with a name and generate


Tokens of *Administrator*

Generate Tokens

Name	Expires in	
Enter Token Name	30 days	Generate

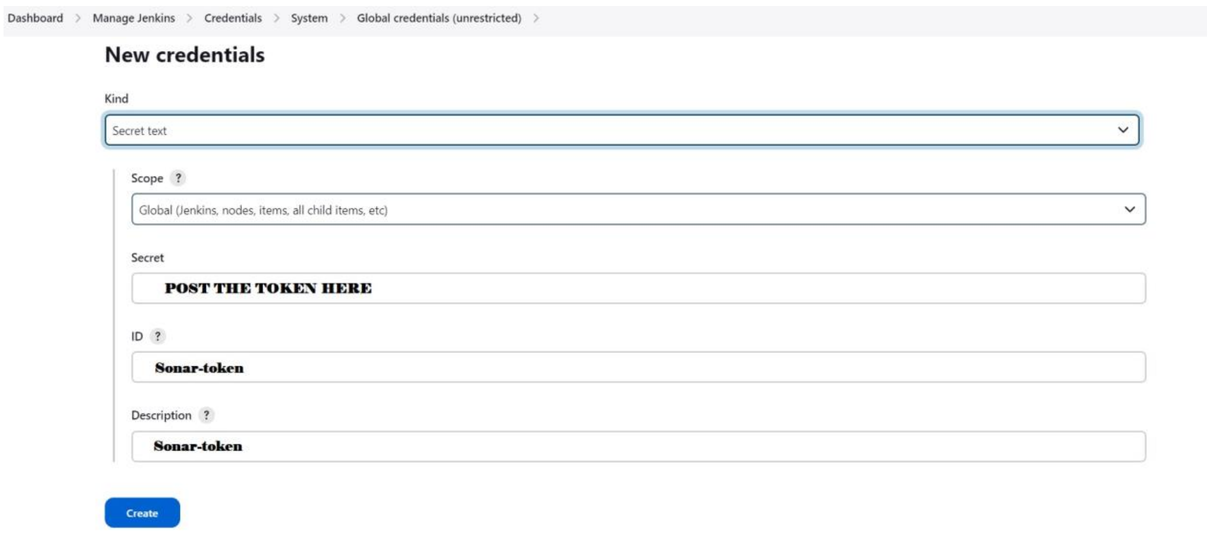
New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

Copy `squ_21d162904c1c72cf8b39665f96480185c99dc2f9`

Name	Type	Project	Last use	Created	Expiration	
Jenkins	User		Never	September 8, 2023	October 8, 2023	Revoke

copy Token

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



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

New credentials

Kind: Secret text

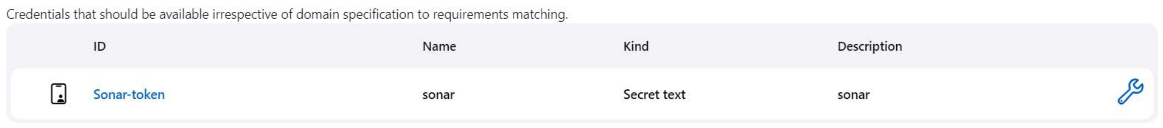
Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: **POST THE TOKEN HERE**

ID: **Sonar-token**

Description: **Sonar-token**

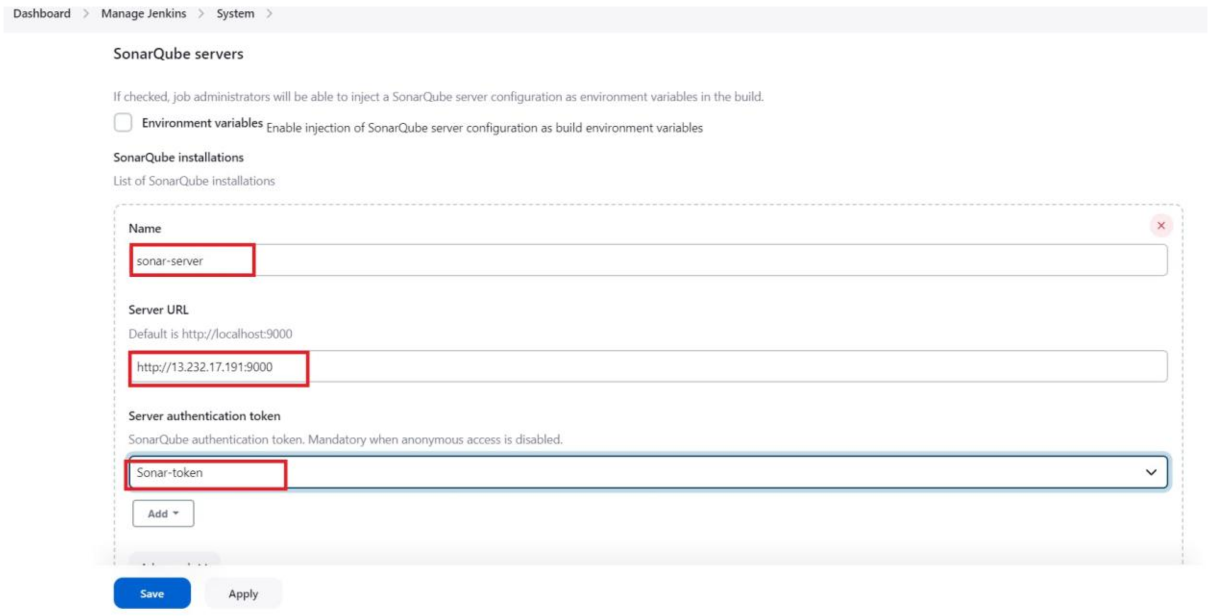
You will this page once you click on create



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.



Dashboard > Manage Jenkins > System >

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.

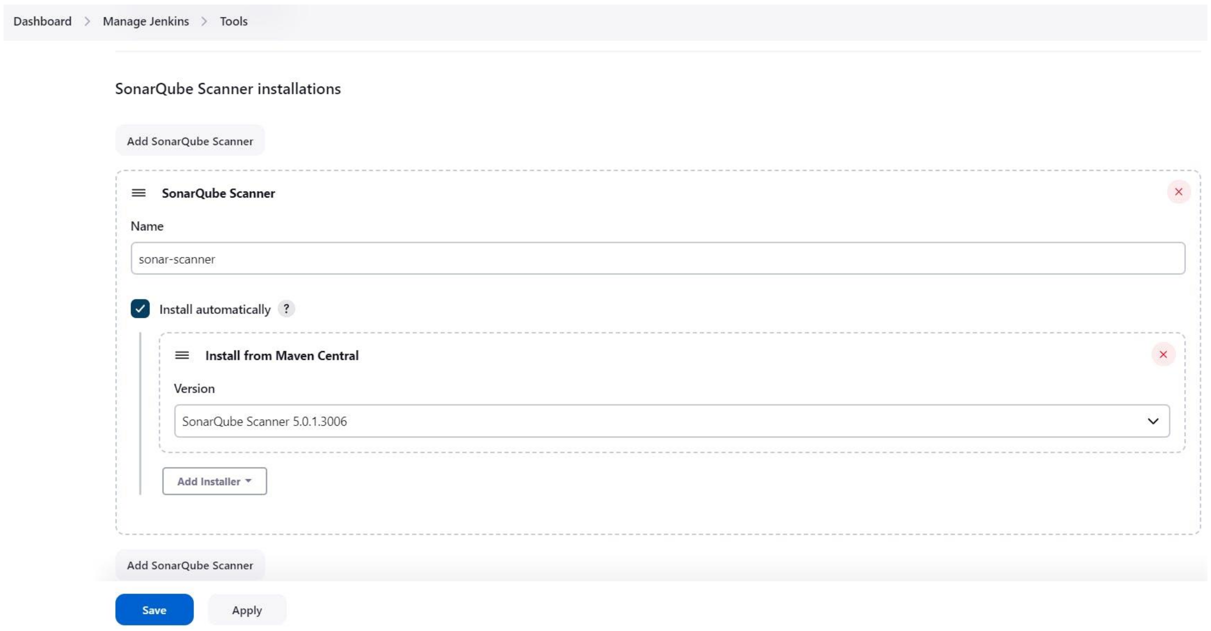
Sonar-token

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.


Dashboard > Manage Jenkins > Tools

SonarQube Scanner installations

SonarQube Scanner

Name: sonar-scanner

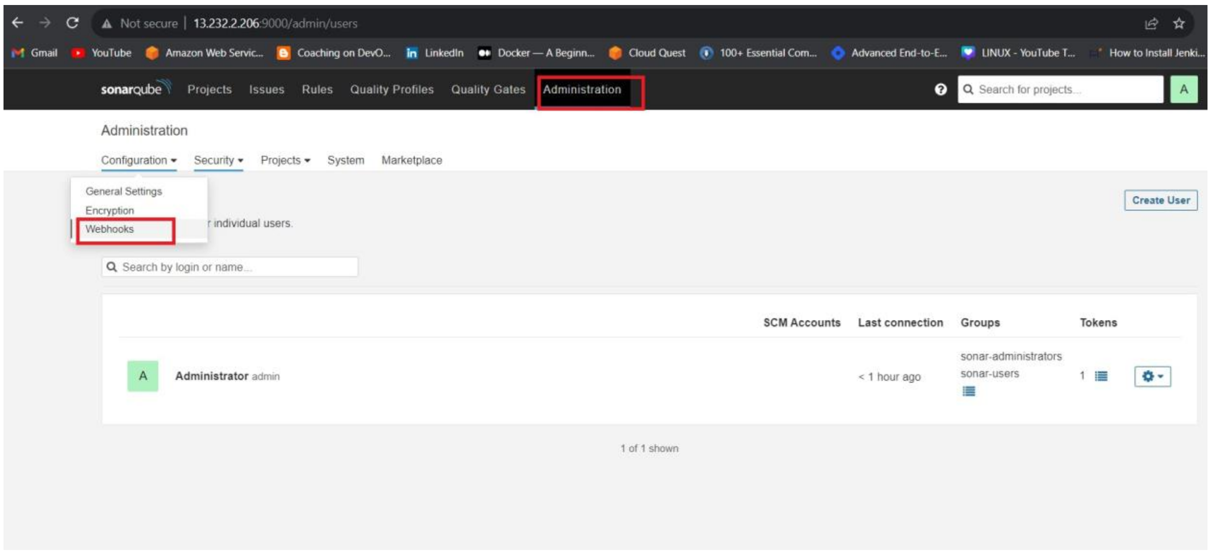
Install automatically ✓

Install from Maven Central

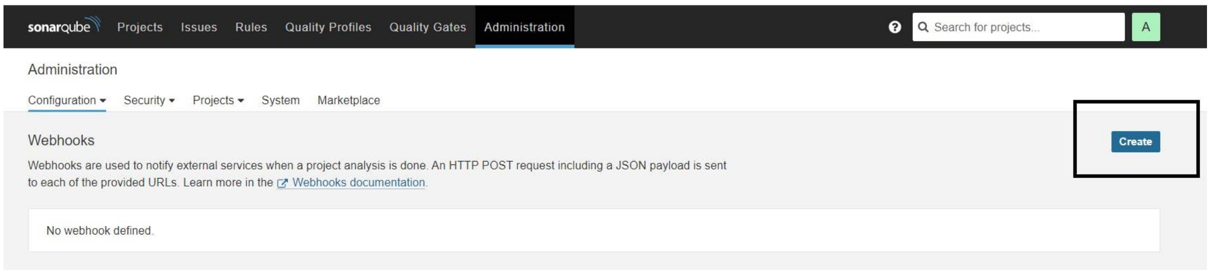
Version: SonarQube Scanner 5.0.1.3006

In the Sonarqube Dashboard add a quality gate also

Administration--> Configuration-->Webhooks



Click on Create

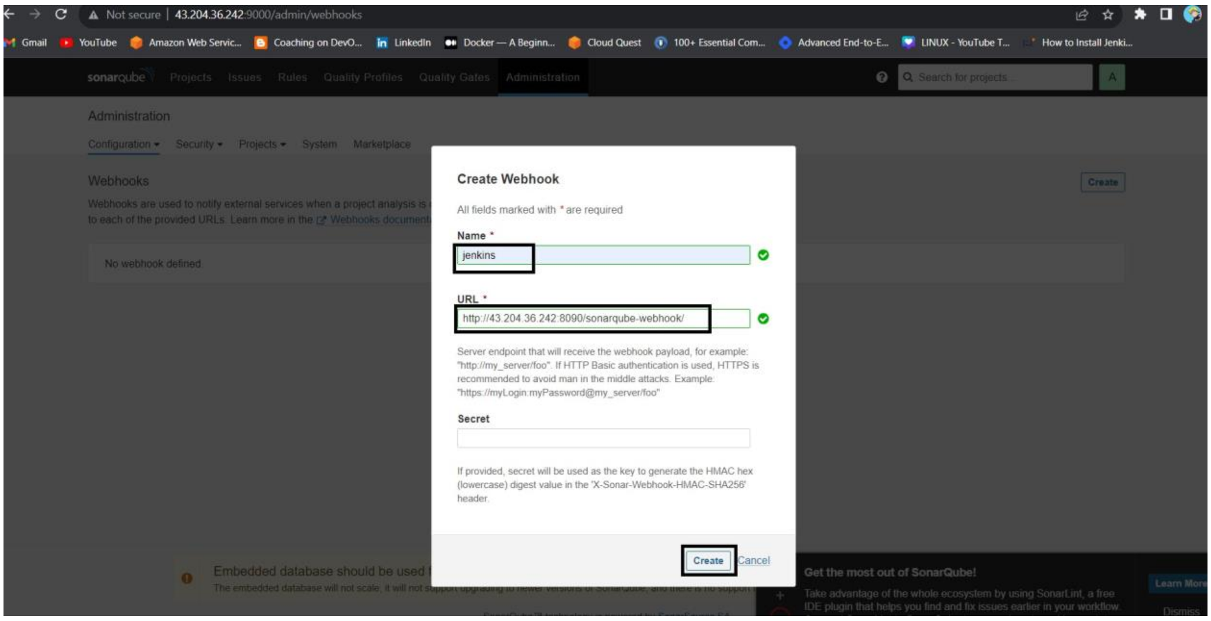


Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the Webhooks documentation.

Add details

#in url section of quality gate

<http://jenkins-public-ip:8080>/sonarqube-webhook/


Not secure | 43.204.36.242:9000/admin/webhooks
Create Webhook
Name *
URL *
Secret
Create
Cancel
Embedded database should be used for evaluation purposes only
Get the most out of SonarQube!
Learn More
Dismiss

All fields marked with * are required

Name *: jenkins

URL *: http://43.204.36.242:8090/sonarqube-webhook/

Server endpoint that will receive the webhook payload, for example: "http://my_server/foo". If HTTP Basic authentication is used, HTTPS is recommended to avoid man in the middle attacks. Example: "https://myLogin:myPassword@my_server/foo"

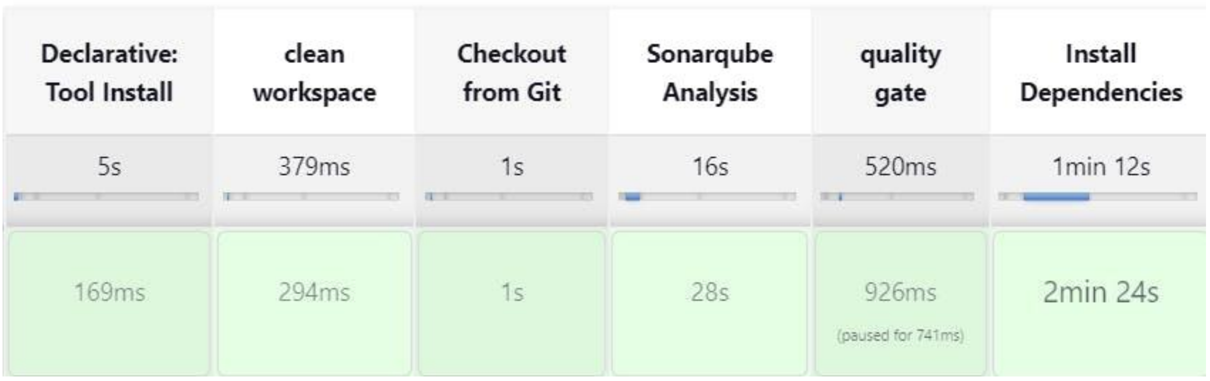
If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

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: 'master', url: 'https://github.com/Milky19/2048-React-CICD.git'
      }
    }
    stage("Sonarqube Analysis "){
      steps{
        withSonarQubeEnv('sonar-server') {
          sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Game \
          -Dsonar.projectKey=Game '''
        }
```

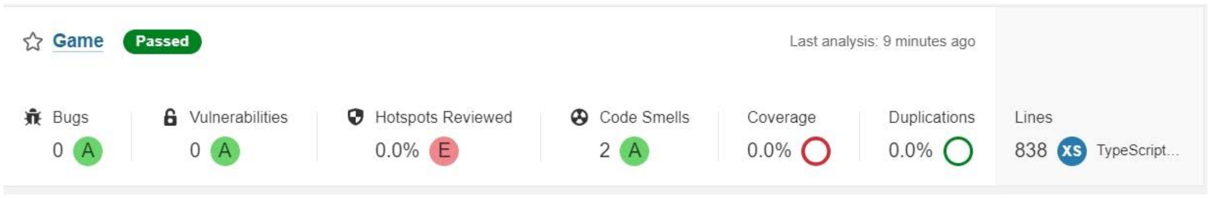
```
            }
        }
        stage("quality gate"){
          steps {
              script {
                  waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
              }
            }
        }
        stage('Install Dependencies') {
            steps {
              sh "npm install"
            }
        }
    }
}
```

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



Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies
5s	379ms	1s	16s	520ms	1min 12s
169ms	294ms	1s	28s	926ms (paused for 741ms)	2min 24s

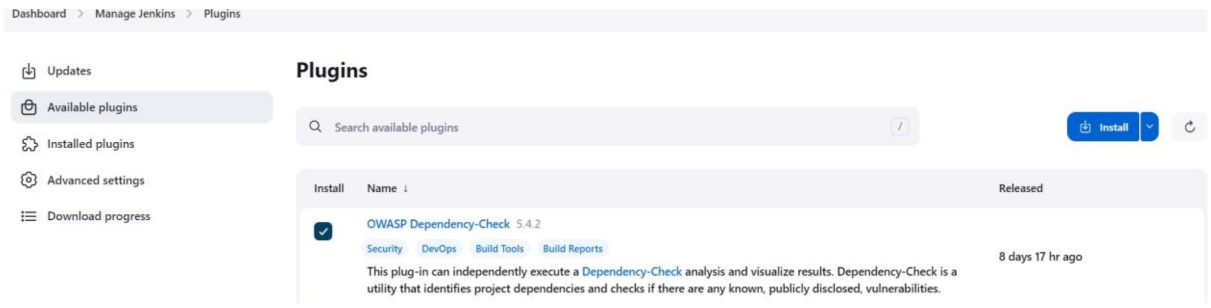
To see the report, you can go to Sonarqube Server and go to Projects.



You can see the report has been generated and the status shows as passed. You can see that there are 838 lines. To see a detailed report, you can go to issues.

Step 5 — Install OWASP Dependency Check Plugins

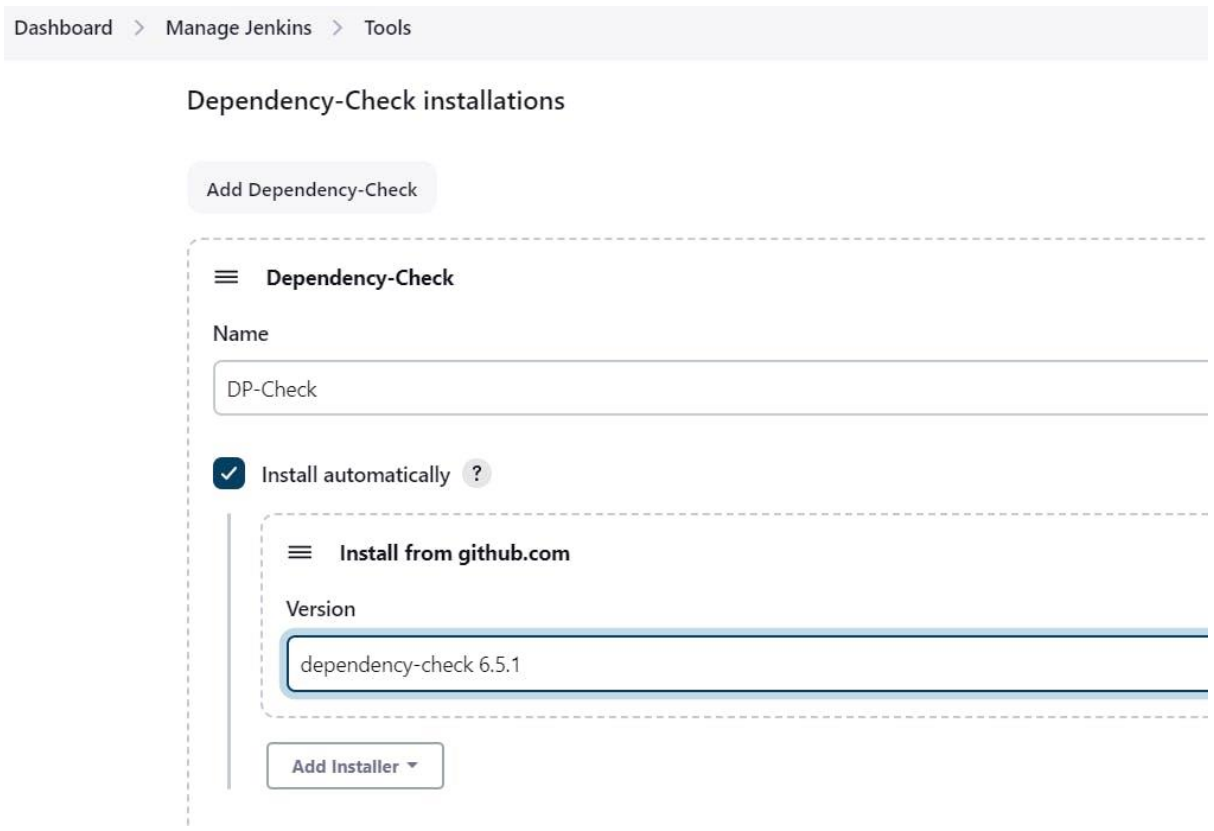
GotoDashboard → Manage Jenkins → Plugins → OWASP Dependency-Check. Click on it and install it without restart.



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 →


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

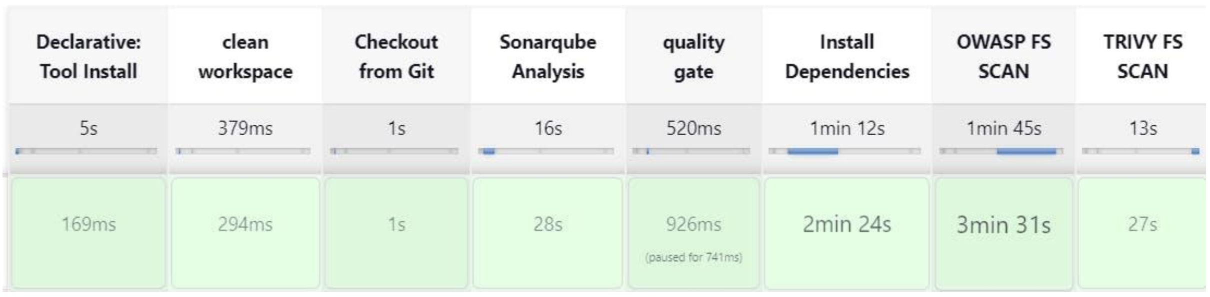
Add Installer

Click on Apply and Save here.

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

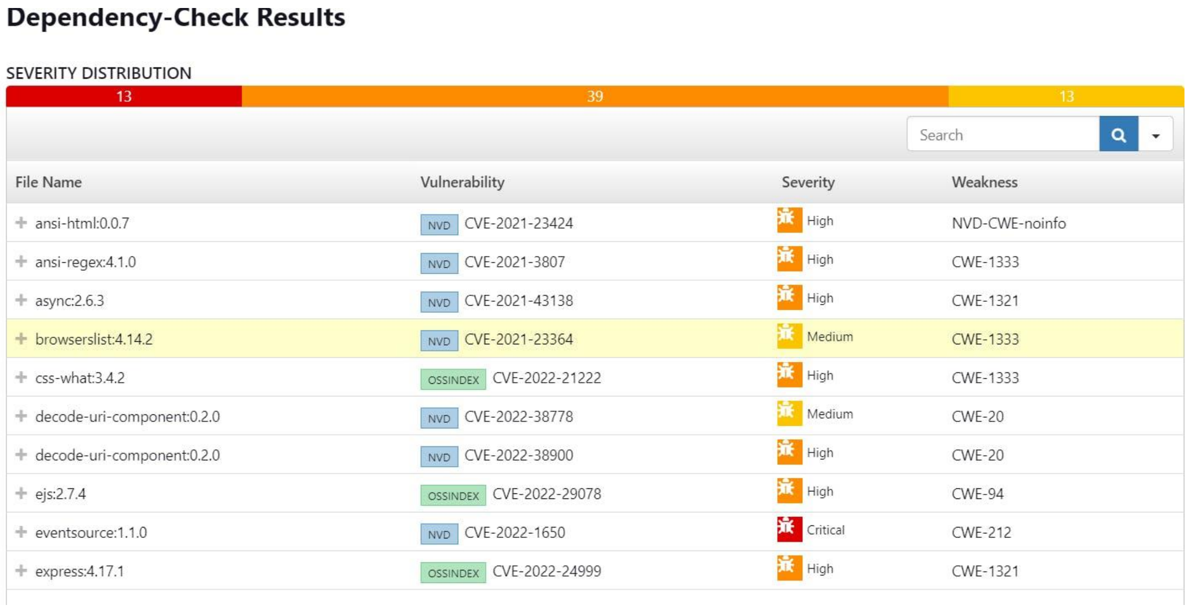
```
stage('OWASP FS SCAN') {
        steps {
            dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
            dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
        }
    }
    stage('TRIVY FS SCAN') {
        steps {
            sh "trivy fs . > trivyfs.txt"
        }
    }
```

The stage view would look like this,



Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN
5s	379ms	1s	16s	520ms	1min 12s	1min 45s	13s
169ms	294ms	1s	28s	926ms (paused for 741ms)	2min 24s	3min 31s	27s

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



Dependency-Check Results

SEVERITY DISTRIBUTION

13	39	13

File Name	Vulnerability	Severity	Weakness
ansi-html:0.0.7	NVD CVE-2021-23424	High	NVD-CWE-noinfo
ansi-regex:4.1.0	NVD CVE-2021-3807	High	CWE-1333
async:2.6.3	NVD CVE-2021-43138	High	CWE-1321
browserslist:4.14.2	NVD CVE-2021-23364	Medium	CWE-1333
css-what:3.4.2	OSSINDEX CVE-2022-21222	High	CWE-1333
decode-uri-component:0.2.0	NVD CVE-2022-38778	Medium	CWE-20
decode-uri-component:0.2.0	NVD CVE-2022-38900	High	CWE-20
ejs:2.7.4	OSSINDEX CVE-2022-29078	High	CWE-94
eventsource:1.1.0	NVD CVE-2022-1650	Critical	CWE-212
express:4.17.1	OSSINDEX CVE-2022-24999	High	CWE-1321

Step 6 — 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`

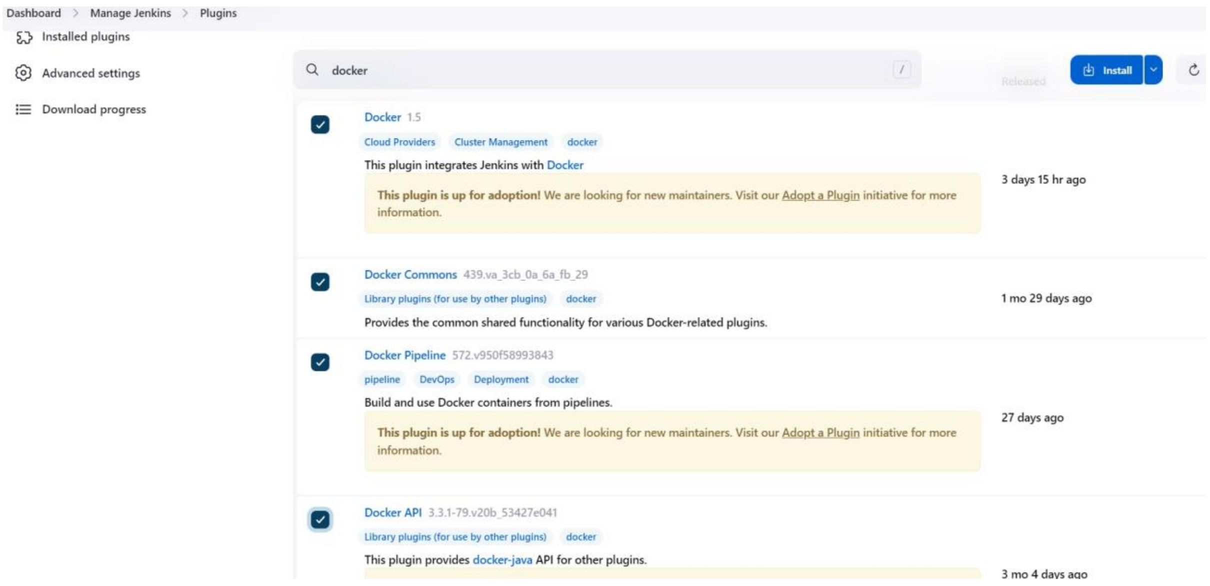
`Docker Commons`

`Docker Pipeline`

`Docker API`

`docker-build-step`

and click on install without restart


Dashboard > Manage Jenkins > Plugins

Installed plugins
Advanced settings
Download progress

docker

Install

Released

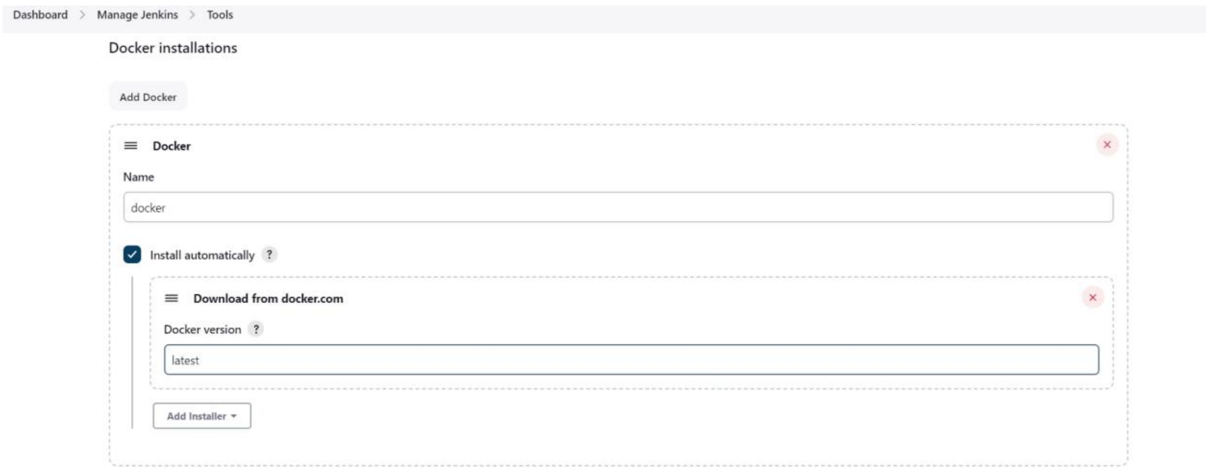
[✓] Docker 1.5
Cloud Providers · Cluster Management · docker
This plugin integrates Jenkins with Docker
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
3 days 15 hr ago

[✓] Docker Commons 439.va_3cb_0a_6a_fb_29
Library plugins (for use by other plugins) · docker
Provides the common shared functionality for various Docker-related plugins.
1 mo 29 days ago

[✓] Docker Pipeline 572.v950f58993843
pipeline · DevOps · Deployment · docker
Build and use Docker containers from pipelines.
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
27 days ago

[✓] Docker API 3.3.1-79.v20b_53427e041
Library plugins (for use by other plugins) · docker
This plugin provides docker-java API for other plugins.
3 mo 4 days ago

Now, goto Dashboard → Manage Jenkins → Tools →

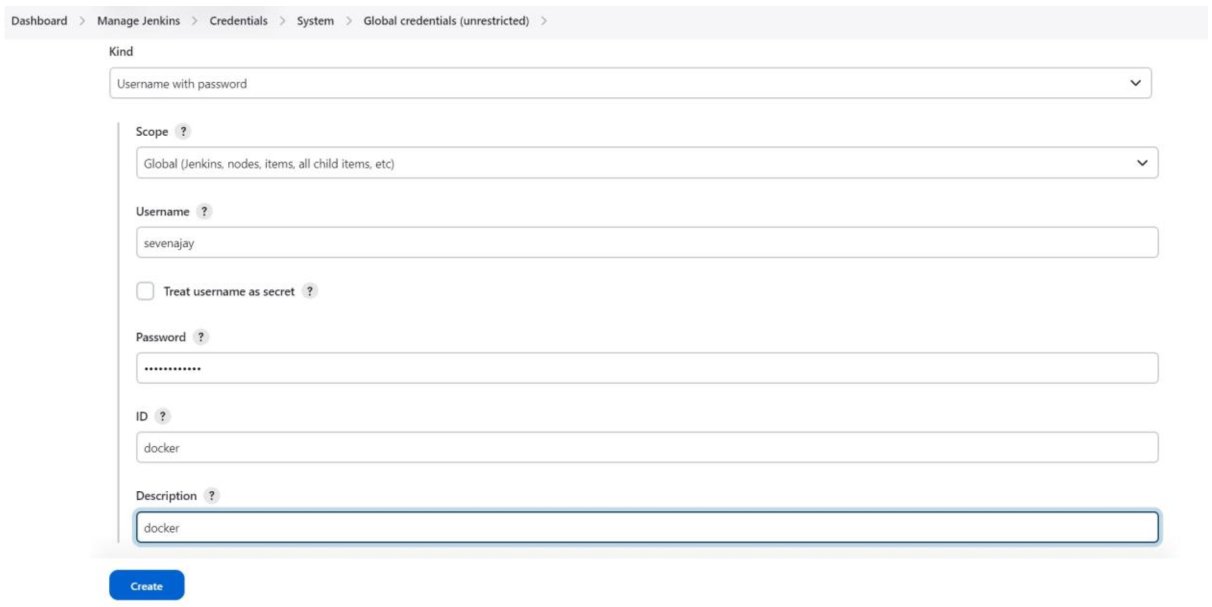

Dashboard > Manage Jenkins > Tools

Docker installations

Add Docker

Docker
Name: docker
[✓] Install automatically
Download from docker.com
Docker version: latest
Add Installer

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

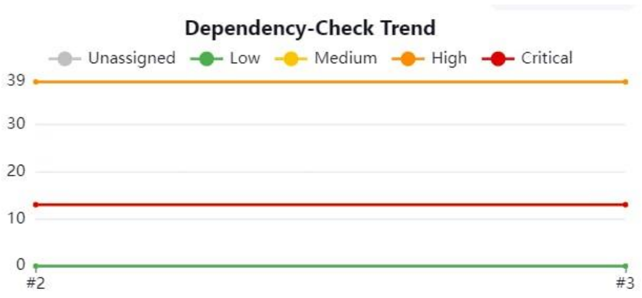
Description: docker

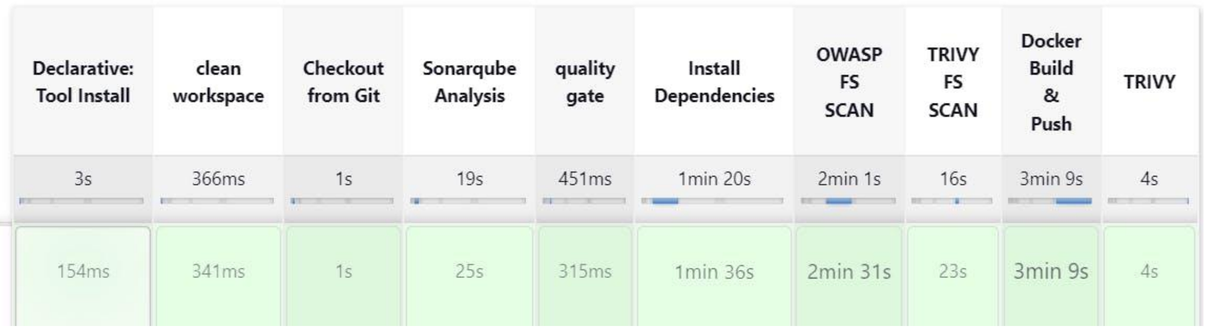
Add this stage to Pipeline Script

```
stage("Docker Build & Push"){
            steps{
                script{
                   withDockerRegistry(credentialsId: 'docker', toolName:
'docker'){
                       sh "docker build -t 2048 ."
                       sh "docker tag 2048 hanvitha/2048:latest "
                       sh "docker push hanvitha/2048:latest "
                    }
                }
            }
        }
        stage("TRIVY"){
```

```
        steps{
            sh "trivy image hanvitha/2048:latest > trivy.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

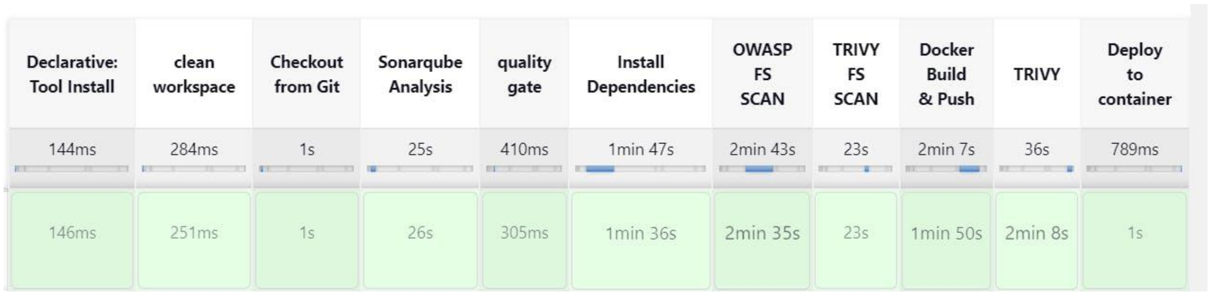
When you log in to Dockerhub, you will see a new image is created

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

```
stage('Deploy to container'){
        steps{
            sh 'docker run -d --name 2048 -p 3000:3000
sevenajay/2048:latest'
        }
```

}

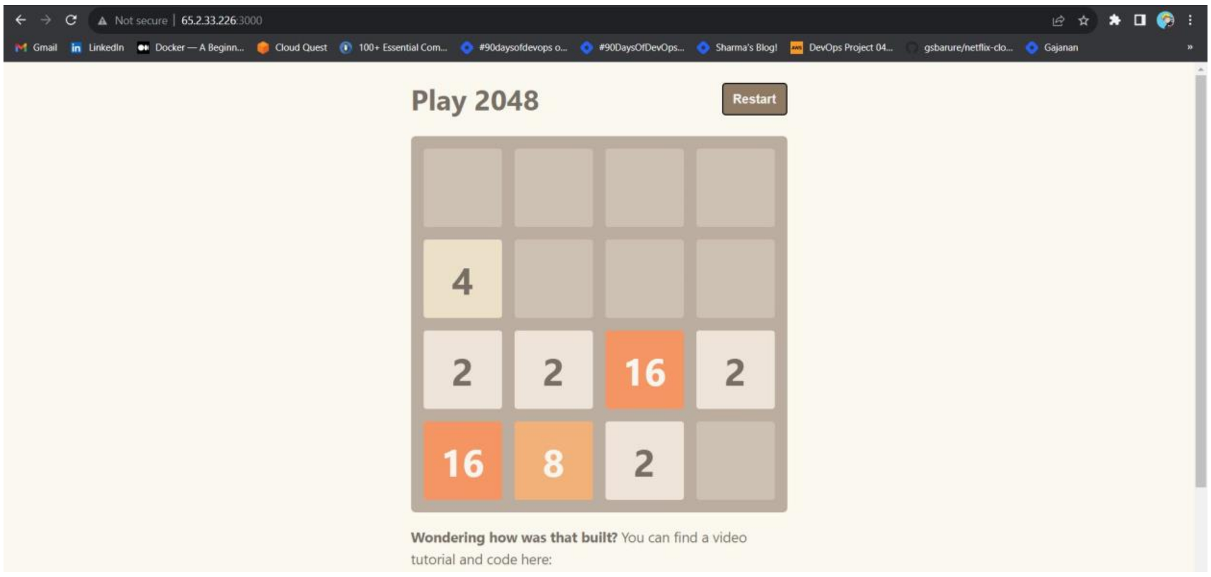
stage view



Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Deploy to container
144ms	284ms	1s	25s	410ms	1min 47s	2min 43s	23s	2min 7s	36s	789ms
146ms	251ms	1s	26s	305ms	1min 36s	2min 35s	23s	1min 50s	2min 8s	1s

`<Jenkins-public-ip:3000>`

You will get this output



Wondering how was that built? You can find a video tutorial and code here:

Step 8 — Kuberenetes Setup

Connect your machines to Putty or Mobaxtreme

Take-Two Ubuntu 20.04 instances one for k8s master and the other one for worker.

Install Kubectl on Jenkins machine also.

Kubectl is to be installed on Jenkins also

sudo apt update

sudo apt install curl

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

Part 1 ----------Master Node------------

sudo hostnamectl set-hostname K8s-Master

----------Worker Node------------

sudo hostnamectl set-hostname K8s-Worker

sudo hostnamectl set-hostname K8s-Worker

Part 2 ------------Both Master & Node ------------

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

EOF


sudo apt-get update


sudo apt-get install -y kubelet kubeadm kubectl


sudo snap install kube-apiserver
```

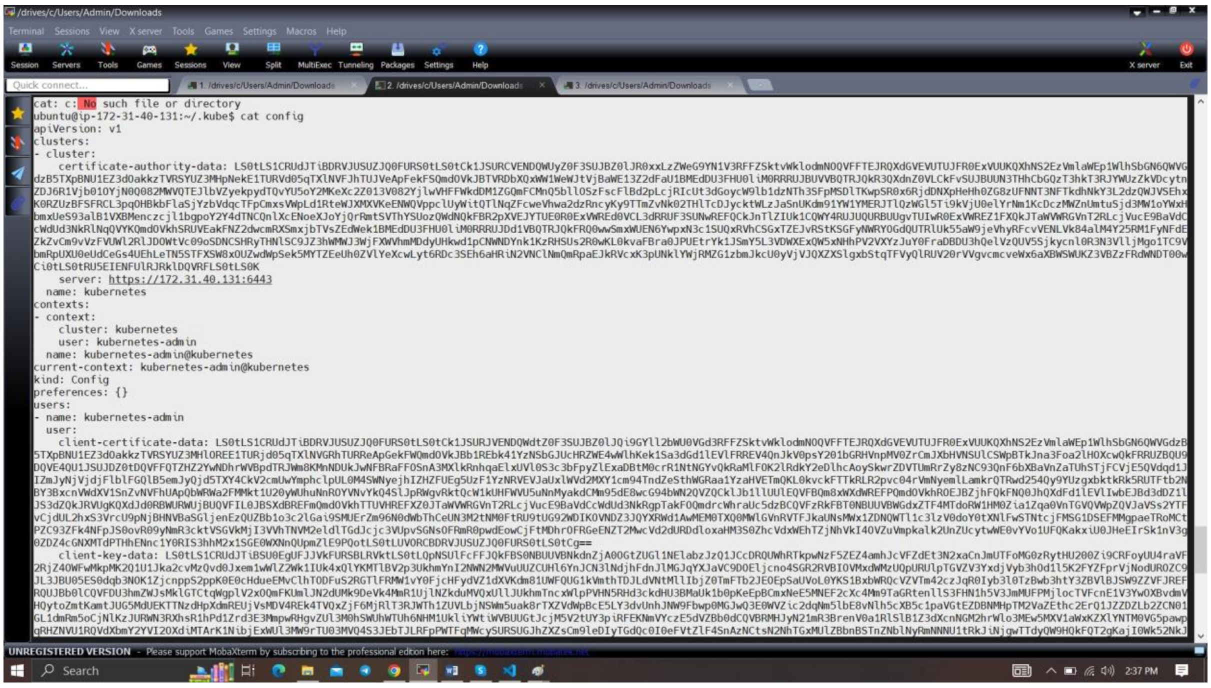
Part 3 --------------- Master ---------------

```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16

# in case your in root exit from it and run below commands

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/kube-flannel.yml
```

----------Worker Node------------

sudo kubeadm join <master-node-ip>:<master-node-port> --token <token> --discovery-token-ca-cert-hash <hash>

Copy the config file to Jenkins master or the local file manager and save it

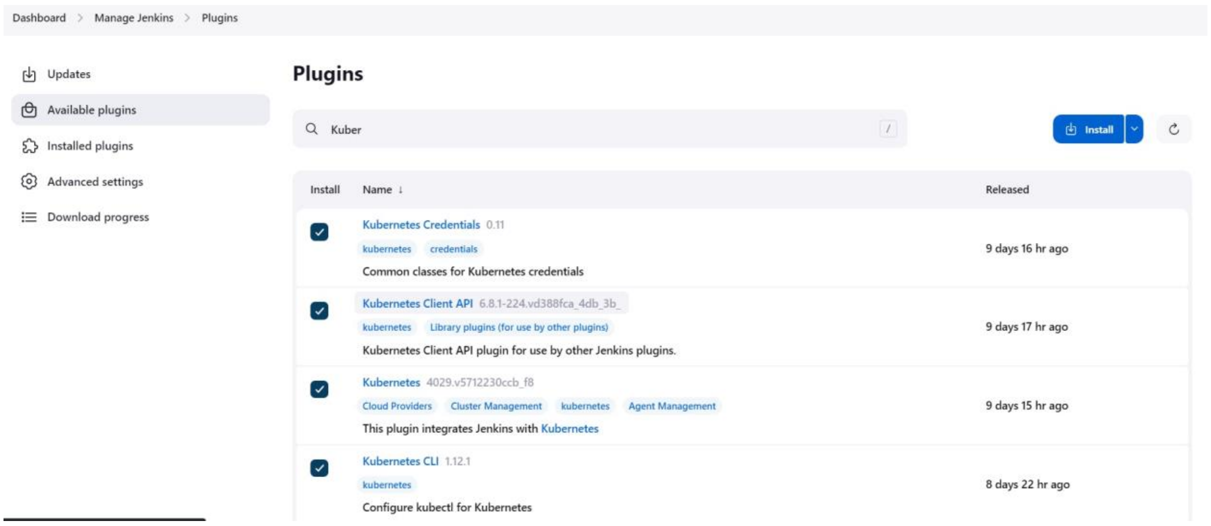

/drives/c/Users/Admin/Downloads
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
1. /drives/c/Users/Admin/Downloads
2. /drives/c/Users/Admin/Downloads
3. /drives/c/Users/Admin/Downloads
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here:
Search
2:37 PM

```
cat: c: No such file or directory
ubuntu@ip-172-31-40-131:~/.kube$ cat config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURCVENDQWUyZ0F3SUJBZ0lJR0xxLzZWeG9YN1V3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVG
dzB5TXpBNU1EZ3dOakkzTVRSYUZ3MHpNekE1TURVd05qTXlNVFJhTUJVeApFekFSQmdOVkJBTVRDbXQxWW1WeWJtVjBaWE13Z2dFaU1BMEdDU3FHU0liM0RRRUJBUVVBQTRJQkR3QXdnZ0VLCkFvSUJBUUN3THhCbGQzT3hkT3RJYWUzZkVDcytn
ZDJ6R1Vjb01OYjN0Q082MWVQTEJlbVZyekpydTQvYU5oY2MKeXc2Z013V082YjlwVHFFWkdDM1ZGQmFCMnQ5bllOSzFscFlBd2pLcjRIcUt3dGoycW9lb1dzNTh3SFpMSDlTKwpSR0x6RjdDNXpHeHh0ZG8zUFNNT3NFTkdhNkY3L2dzQWJVSEhx
K0RZUzBFSFRCL3pqOHBkbFlaSjYzbVdqcTFpCmxsVWpLd1RteWJXMXVKeENWQVppclUyWitQTlNqZFcweVhwa2dzRncyKy9TTmZvNk02THlTcDJycktWLzJaSnUKdm91YW1YMERJTlQzWGl5Ti9kVjU0elYrNm1KcDczMWZnUmtuSjd3MW1oYWxH
bmxUeS93alB1VXBMenczcjl1bgpoY2Y4dTNCQnlXcENoeXJoYjQrRmtSVThYSUozQWdNQkFBR2pXVEJYTUE0R0ExVWREd0VCL3dRRUF3SUNwREFQQkJnTlZIUk1CQWY4RUJUQURBUUgvTUIwR0ExVWREZ1FXQkJTaWVWRGVnT2RLcjVucE9BaVdD
cWdUd3NkRlNqQVYKQmdOVkhSRUVEakFNZ2dwcmRXSmxjbTVsZEdWek1BMEdDU3FHU0liM0RRRUJDd1VBQTRJQkFRQ0wwSmxWUEN6YwpxN3c1SUQxRVhCSGxTZEJvRStKSGFyNWRYOGdQUTRlUk55aW9jeVhyRFcvVENLVk84alM4Y25RM1FyNFdE
ZkZvCm9vVzFVUWl2RlJDOWtVc09oSDNCSHRyTHNlSC9JZ3hWMWJ3WjFXWVhmMDdyUHkwd1pCNWNDYnk1KzRHSUs2R0wKL0kvaFBra0JPUEtrYk1JSmY5L3VDWXExQW5xNHhPV2VXYzJuY0FraDBDU3hQelVzQUV5Sjkycnl0R3N3VlljMgo1TC9V
bmRpUXU0eUdCeGs4UEhLeTN5STFXSW8xOUZwdWpSek5MYTZEeUh0ZVlYeXcwLyt6RDc3SEh6aHRiN2VNClNmQmRpaEJkRVcxK3pUNklYWjRMZG1zbmJkcU0yVjVJQXZXSlgxbStqTFVyQlRUV20rVVgvcmcveWx6aXBWSWUKZ3VBZzFRdWNDT00w
Ci0tLS0tRU5EIENFUlRJRklDQVRFLS0tLS0K
    server: https://172.31.40.131:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURJVENDQWdtZ0F3SUJBZ0lJQi9GYll2bWU0VGd3RFFZSktvWklodmNOQVFFTEJRQXdGVEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB
5TXpBNU1EZ3dOakkzTVRSYUZ3MHlOREE1TURjd05qTXlNVGRhTURReApGekFWQmdOVkJBb1REbk41YzNSbGJUcHRZWE4wWlhKek1Sa3dGd1lEVlFRREV4QnJkV0psY201bGRHVnpMV0ZrCmJXbHVNSUlCSWpBTkJna3Foa2lHOXcwQkFRRUZBQU9
DQVE4QU1JSUJDZ0tDQVFFQTZHZ2YwNDhrWVBpdTRJWm8KMnNDUkJwNFBRaFFOSnA3MXlkRnhqaElxUVl0S3c3bFpyZlExaDBtM0crR1NtNGYvQkRaMlFOK2lRdkY2eDlhcAoySkwrZDVTUmRrZy8zNC93QnF6bXBaVnZaTUhSTjFCVjE5QVdqd1J
IZmJyNjVjdjFlblFGQlB5emJyQjd5TXY4CkV2cmUwYmphclpUL0M4SWNyejhIZHZFUEg5UzF1YzNRVEVJaUxlWVd2MXY1cm94TndZeSthWGRaa1YzaHVETmQKL0kvckFTTkRLR2pvc04rVmNyemlLamkrQTRwd254Qy9YUzgxbktkRk5RUTFtb2N
BY3BxcnVWdXV1SnZvNVFhUApQbWRWa2FMMkt1U20yWUhuNnROYVNvYkQ4SlJpRWgvRktQcW1kUHFWVU5uNnMyakdCMm95dE8wcG94bWN2QVZQCklJb1llUUlEQVFBQm8xWXdWREFPQmdOVkhROEJBZjhFQkFNQ0JhQXdFd1lEVlIwbEJBd3dDZ1l
JS3dZQkJRVUgKQXdJd0RBWURWUjBUQVFIL0JBSXdBREFmQmdOVkhTTUVHREFXZ0JTaWVWRGVnT2RLcjVucE9BaVdDcWdUd3NkRgpTakFOQmdrcWhraUc5dzBCQVFzRkFBT0NBUUVBWGdxZTF4MTdoRW1HM0Zia1Zqa0VnTGVQVWpZQVJaVSs2YTF
vCjdUL2hxS3VrcU9pNjBHNVBaSGljenEzQUZBb1o3c2lGai9SMUErZm96N0dWbThCeUN3M2tNM0FtRU9tUG92WDIKOVNDZ3JQYXRWd1AwMEM0TXQ0MWlGVnRVTFJkaUNsMWx1ZDNQWTl1c3lzV0doY0tXNlFwSTNtcjFMSG1DSEFMMgpaeTRoMCt
PZC93ZFk4NFpJS0ovR09yNmR3cktVSGVkMjI3VVhTNVM2eldlTGdJcjc3VUpvSGNsOFRmR0pwdEowCjFtMDhrOFRGeENZT2MwcVd2dURDdloxaHM3S0ZhcVdxWEhTZjNhVkI4OVZuVmpkalk2UnZUcytwWE0vYVo1UFQKakxiU0JHeEIrSk1nV3g
0ZDZ4cGNXMTdPTHhENnc1Y0RIS3hhM2x1SGE0WXNnQUpmZlE9PQotLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0tCg==
    client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFcFFJQkFBS0NBUUVBNkdnZjA0OGtZUGl1NElabzJzQ1JCcDRQUWhRTkpwNzF5ZEZ4amhJcVFZdEt3N2xaCnJmUTFoMG0zRytHU200Zi9CRFoyUU4raVF
2RjZ4OWFwMkpMK2Q1U1Jka2cvMzQvd0Jxem1wWlZ2Wk1IUk4xQlYKMTlBV2p3UkhmYnI2NWN2MWVuUUZCUHl6YnJCN3lNdjhFdnJlMGJqYXJaVC9DOEljcno4SGR2RVBIOVMxdWMzUQpURUlpTGVZV3YxdjVyb3h0d1l5K2FYZFprVjNodURNZC9
JL3JBU05ES0dqb3NOK1ZjcnppS2ppK0E0cHdueEMvClhTODFuS2RGTlFRMW1vY0FjcHFydVZ1dXVKdm81UWFQUG1kVmthTDJLdVNtMllIbjZ0TmFTb2JEOEpSaUVoL0YKS1BxbWRQcVZVTm42czJqR0Iyb3l0TzBwb3htY3ZBVlBJSW9ZZVFJREF
RQUJBb0lCQVFDU3hmZWJsMklGTCtqWgplV2xOQmFKUmlJN2dUMk9DeVk4MmR1UjlNZkduMVQxUllJUkhmTncxWlpPVHN5RHd3ckdHU3BMaUk1b0pKeEpBCmxNeE5MNEF2cXc4Mm9TaGRtenllS3FHN1h5V3JmMUFPMjlocTVFcnE1V3YwOXBvdmV
HQytoZmtKamtJUG5MdUEKTTNzdHpXdmREUjVsMDV4REk4TVQxZjF6MjRlT3RJWTh1ZUVLbjNSWm5uak8rTXZVdWpBcE5LY3dvUnhJNW9Fbwp0MGJwQ3E0WVZic2dqNm5lbE8vNlh5cXB5c1paVGtEZDBNMHpTM2VaZEthc2ErQ1JZZDZLb2ZCN01
GL1dmRm5oCjNlKzJURWN3RXhsR1hPd1Zrd3E3MmpwRHgvZUl3M0hSWUhWTUh6NHM1UkliYWtiWVBUUGtJcjM5V2tUY3piRFEKNmVYczE5dVZBb0dCQVBRMHJyN21mR3BrenV0a1RlSlB1Z3dXcnNGM2hrWlo3MEw5MXV1aWxKZXlYNTM0VG5wawp
qRHZNVU1RQVdXbmY2YVI2OXdiMTArK1NibjExWUl3MW9rTU03MVQ4S3JEbTJLRFpPWTFqMWcySURSUGJhZXZsCm9leDIyTGdQc0I0eFVtZlF4SnAzNCtsN2NhTGxMUlZBbnBSTnZNblNyRmNNNU1tRkJiNjgwTTdyQW9HQkFQT2gKajI0Wk52NkpJ
```

copy it and save it in documents or another folder save it as secret-file.txt

Note: create a secret-file.txt in your file explorer save the config in it and use this at the kubernetes credential section.

Install Kubernetes Plugin, Once it's installed successfully

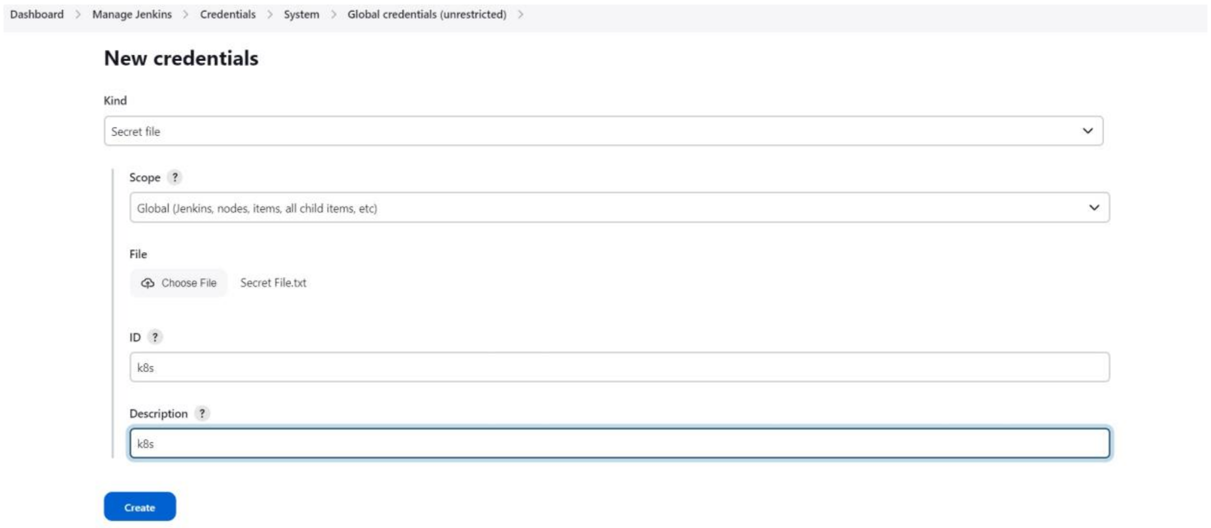

Dashboard > Manage Jenkins > Plugins

Plugins

Search: Kuber

Install	Name ↓	Released
✓	Kubernetes Credentials 0.11 — kubernetes, credentials — Common classes for Kubernetes credentials	9 days 16 hr ago
✓	Kubernetes Client API 6.8.1-224.vd388fca_4db_3b_ — kubernetes, Library plugins (for use by other plugins) — Kubernetes Client API plugin for use by other Jenkins plugins.	9 days 17 hr ago
✓	Kubernetes 4029.v5712230ccb_f8 — Cloud Providers, Cluster Management, kubernetes, Agent Management — This plugin integrates Jenkins with Kubernetes	9 days 15 hr ago
✓	Kubernetes CLI 1.12.1 — kubernetes — Configure kubectl for Kubernetes	8 days 22 hr ago

goto 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: Choose File — Secret File.txt

ID: k8s

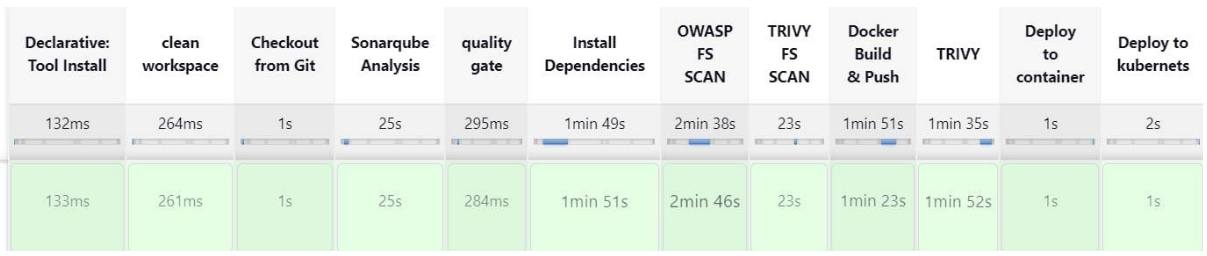
Description: k8s

final step to deploy on the Kubernetes cluster

```
stage('Deploy to kubernets'){
      steps{
        script{
```

```
                    withKubeConfig(caCertificate: '', clusterName: '', contextName: '', credentialsId: 'k8s', namespace: '', restrictKubeConfigAccess: false, serverUrl: '') {
                        sh 'kubectl apply -f deployment.yaml'
                    }
                }
            }
        }
```

stage view

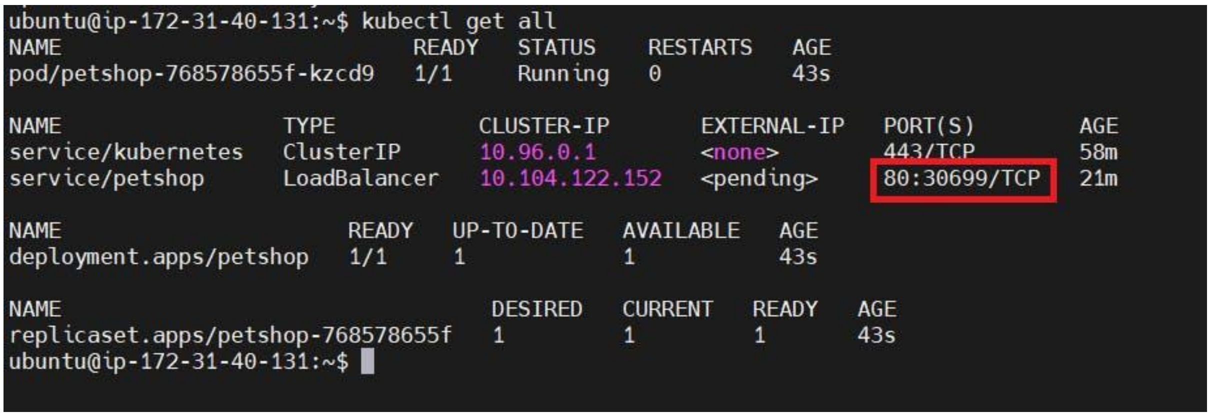


Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY	Deploy to container	Deploy to kubernets
132ms	264ms	1s	25s	295ms	1min 49s	2min 38s	23s	1min 51s	1min 35s	1s	2s
133ms	261ms	1s	25s	284ms	1min 51s	2min 46s	23s	1min 23s	1min 52s	1s	1s

In the Kubernetes cluster give this command

kubectl get all

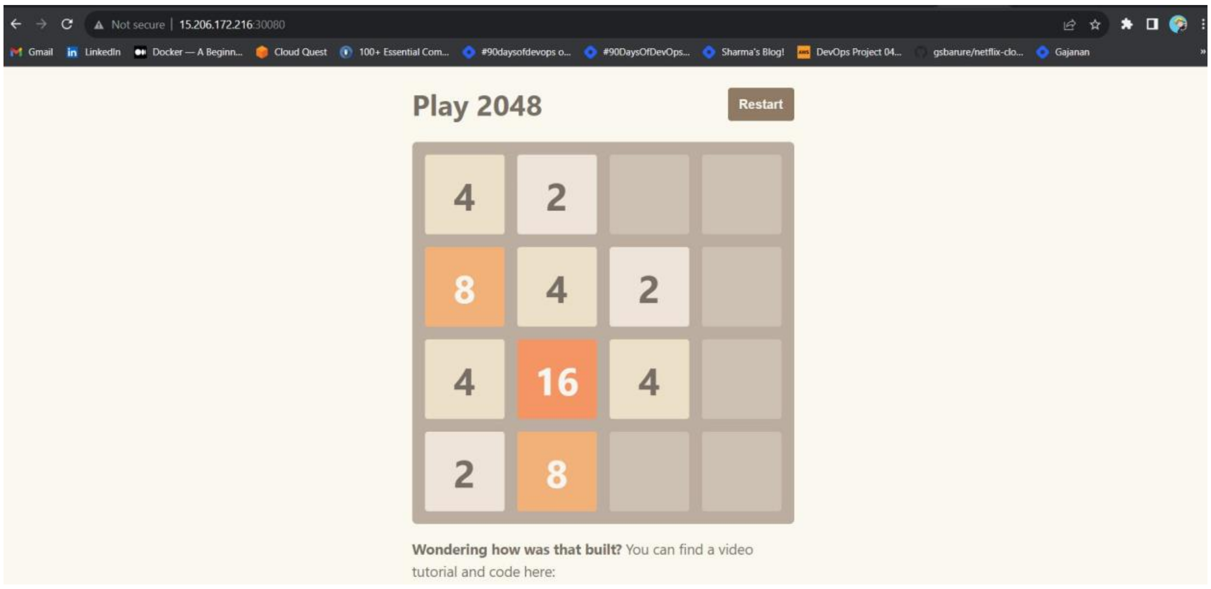
kubectl get svc #use anyone



```
ubuntu@ip-172-31-40-131:~$ kubectl get all
NAME                           READY   STATUS    RESTARTS   AGE
pod/petshop-768578655f-kzcd9   1/1     Running   0          43s

NAME                 TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
service/kubernetes   ClusterIP      10.96.0.1        <none>        443/TCP        58m
service/petshop      LoadBalancer   10.104.122.152   <pending>     80:30699/TCP   21m

NAME                      READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/petshop   1/1     1            1           43s

NAME                                 DESIRED   CURRENT   READY   AGE
replicaset.apps/petshop-768578655f   1         1         1       43s
ubuntu@ip-172-31-40-131:~$
```

STEP9:Access from a Web browser with

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



Wondering how was that built? You can find a video tutorial and code here:

COMPLETE 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: 'master', url:
'https://github.com/Milky19/2048-React-CICD.git'

            }

        }

        stage("Sonarqube Analysis "){

            steps{

                withSonarQubeEnv('sonar-server') {

                    sh ''' $SCANNER_HOME/bin/sonar-scanner -
Dsonar.projectName=Game \

                    -Dsonar.projectKey=Game '''

                }

            }

        }

        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 -t 2048 ."
                       sh "docker tag 2048 hanvitha/2048:latest "
                       sh "docker push hanvitha/2048:latest "
                    }
                }
            }
        }
        stage("TRIVY"){
            steps{
                sh "trivy image hanvitha/2048:latest > trivy.txt"
            }
        }
        stage('Deploy to container'){
            steps{
                sh 'docker run -d --name 2048 -p 3000:3000
hanvitha/2048:latest'
            }
        }
        stage('Deploy to kubernets'){
            steps{
                script{
                    withKubeConfig(caCertificate: '', clusterName: '',
contextName: '', credentialsId: 'k8s', namespace: '',
restrictKubeConfigAccess: false, serverUrl: '') {
                        sh 'kubectl apply -f deployment.yaml'
                    }
                }
            }
        }
    }
}
```