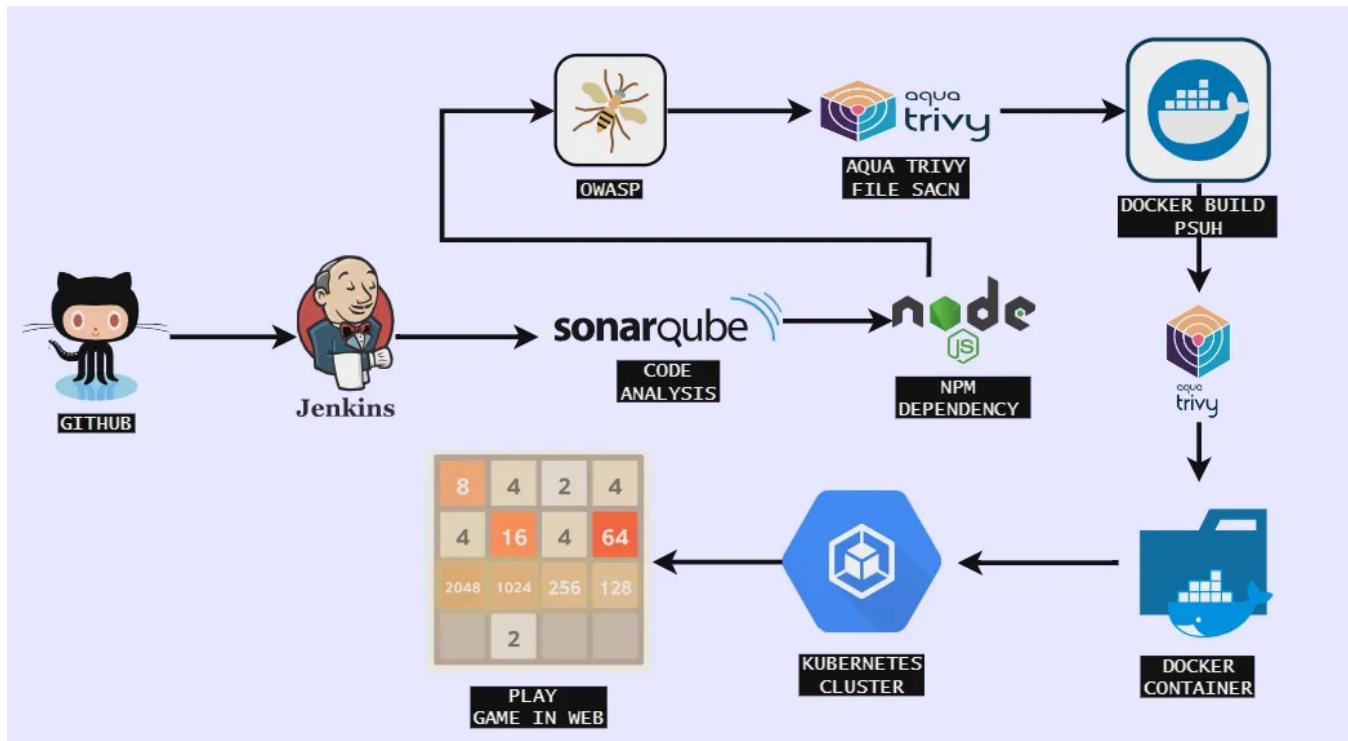


# DevSecOps: Deploying the 2048 Game on Docker and Kubernetes with Jenkins CI/CD



Hello friends, we will be deploying a React Js 2048 Game. We will be using Jenkins as a CICD tool and deploying our application on a Docker container and Kubernetes Cluster. I Hope this detailed blog is useful.

Youtube video : [https://youtu.be/21Z-u8Fd\\_Mk?si=Ux4WabhFAxtYH9-g](https://youtu.be/21Z-u8Fd_Mk?si=Ux4WabhFAxtYH9-g)

GitHub Repo : <https://github.com/Aj7Ay/2048-React-CICD.git>

**Steps:-**

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.

Now, let's get started and dig deeper into each of these steps:-

## **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).

Instances (1) <a href="#">Info</a>										
<a href="#">Launch instances</a>										
<a href="#">Find instance by attribute or tag (case-sensitive)</a>										
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 D		
<input type="checkbox"/>	CI-CD	i-065c10200537a1eee	<span>Running</span>	t2.large	<span>2/2 checks passed</span>	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/trusted.gpg.d/adoptium.asc
echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/api/debian stable main" | 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
```

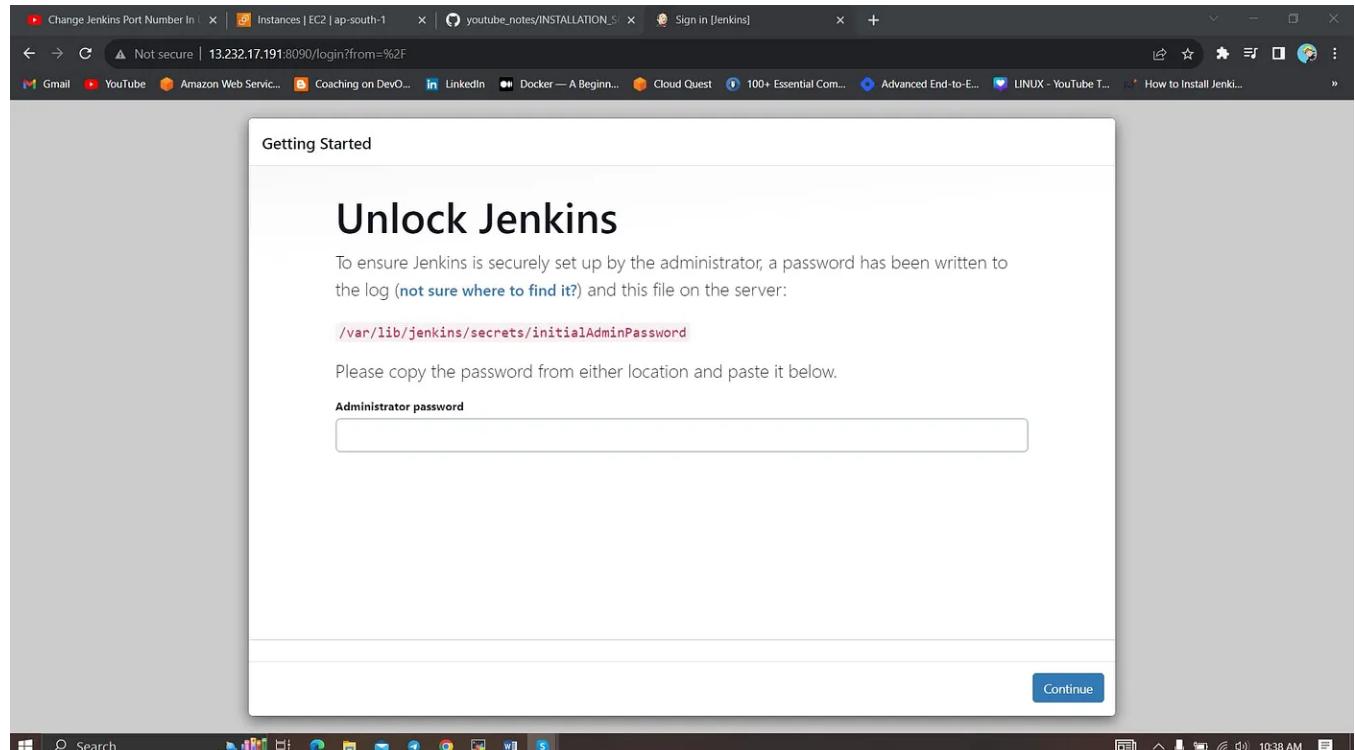
```
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
      https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
      /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update -y
sudo apt-get install jenkins -y
sudo systemctl start jenkins
sudo systemctl status jenkins
```

```
sudo chmod 777 jenkins.sh
./jenkins.sh    # this will install Jenkins
```

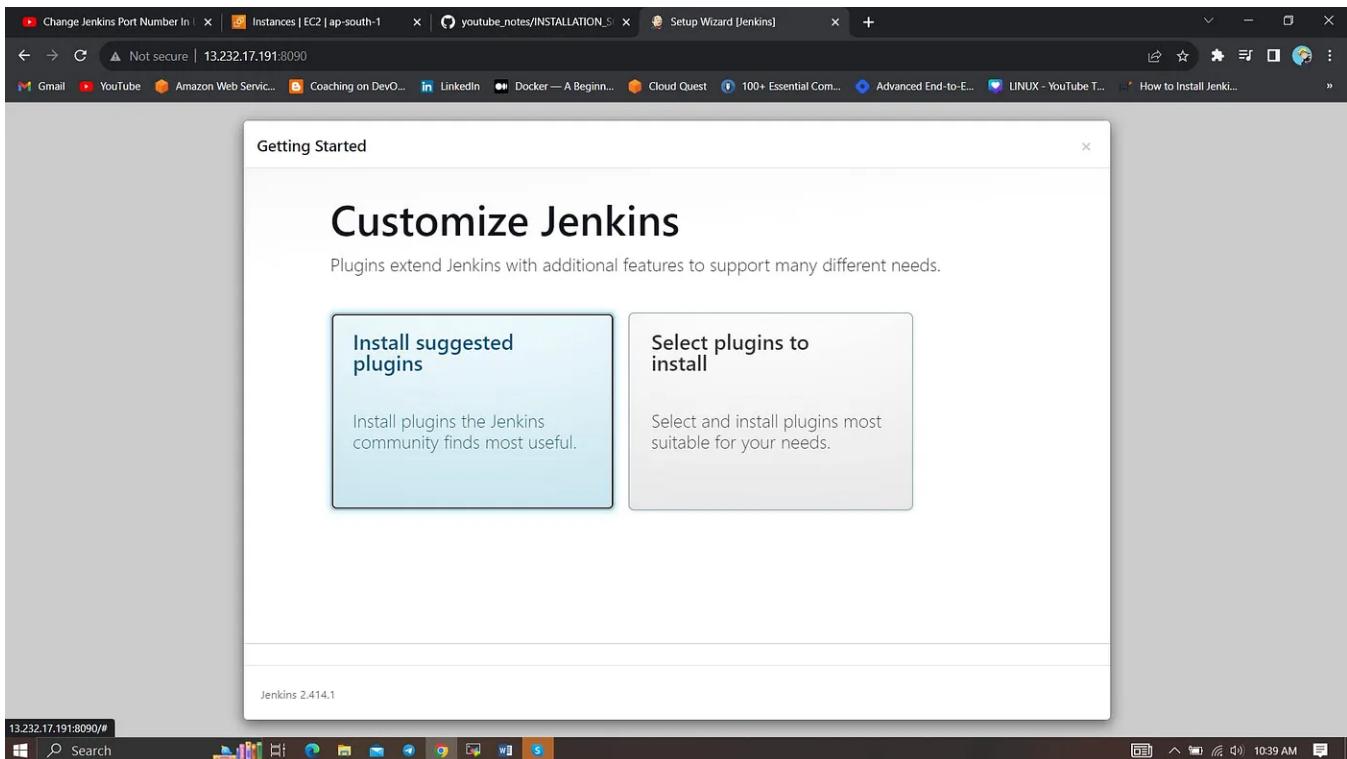
Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

Now, grab your Public IP Address

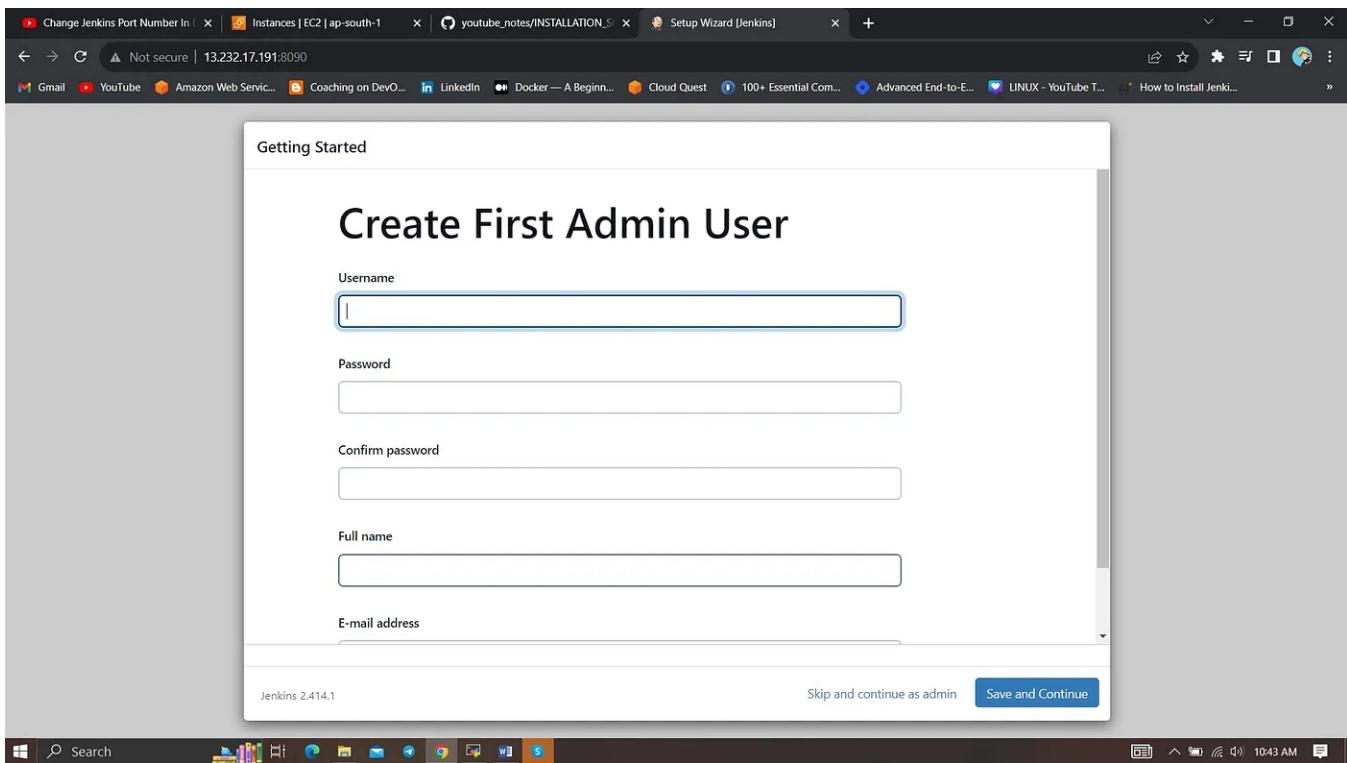
```
<EC2 Public IP Address:8080>
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



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.

## 2B — Install Docker

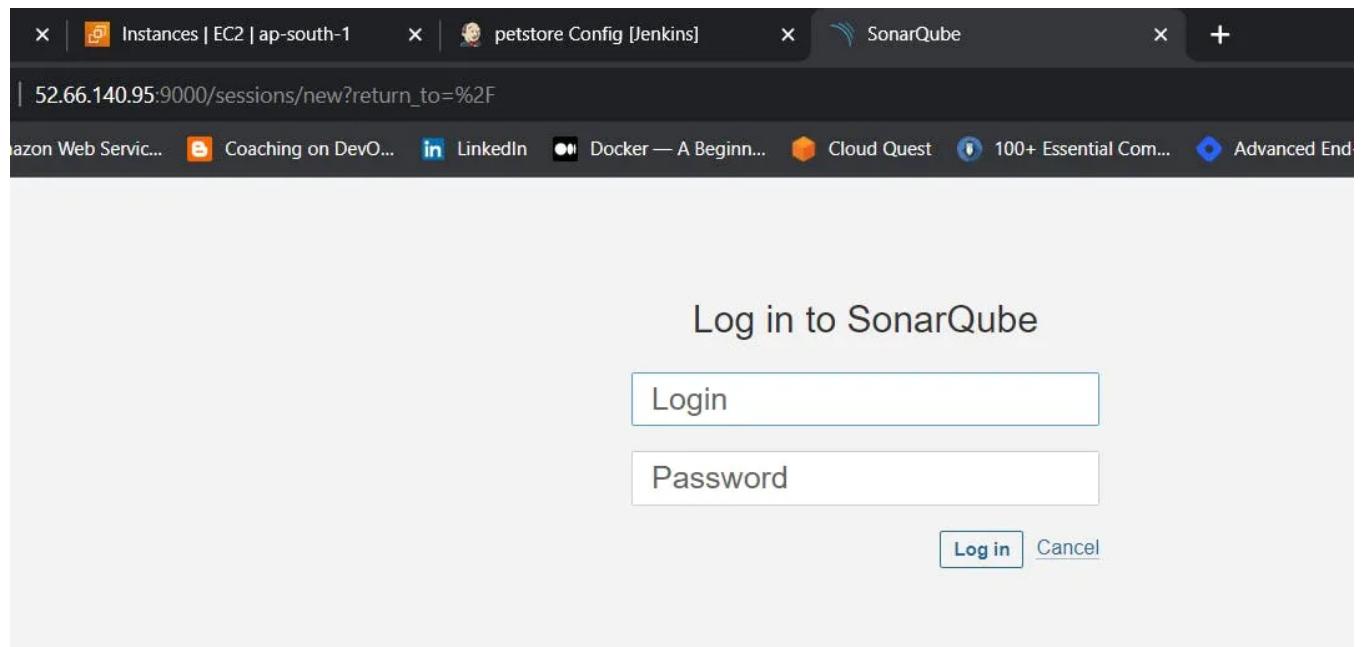
```
sudo apt-get update
sudo apt-get install docker.io -y
sudo usermod -aG docker $USER    #my case is ubuntu
newgrp docker
sudo chmod 777 /var/run/docker.sock
```

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: sha256:1a118f8ab960d6c3d4ea8b4455a5a656065451c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b60c96bf9ad3d62289436af7f752fdb4993092d0ca3065e2f2e32301b50139
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

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?

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 --dear
echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github
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

Go to 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)

The screenshot shows the Jenkins Plugins page. The left sidebar has links for Updates, Available plugins (which is selected), Installed plugins, Advanced settings, and Download progress. The main area is titled 'Plugins' and shows a search bar. Below it, there's a table with columns for 'Install', 'Name', and 'Released'. Two plugins are highlighted with yellow boxes: 'Eclipse Temurin installer 1.5' and 'SonarQube Scanner 2.15'. Both have a note below them: 'This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.' The 'Eclipse Temurin installer' was released 11 months ago, and the 'SonarQube Scanner' was released 9 months and 19 days ago. At the bottom, the 'NodeJS' plugin is listed with version 1.6.1, released 1 month and 2 days ago. A note says: 'NodeJS Plugin executes [NodeJS](#) script as a build step.'

Install	Name	Released
<input checked="" type="checkbox"/>	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from <a href="https://adoptium.net">https://adoptium.net</a>	11 mo ago
<input checked="" type="checkbox"/>	SonarQube Scanner 2.15 <small>External Site/Tool Integrations Build Reports</small> This plugin allows an easy integration of <a href="#">SonarQube</a> , the open source platform for Continuous Inspection of code quality.	9 mo 19 days ago
<input checked="" type="checkbox"/>	NodeJS 1.6.1 <small>npm</small> NodeJS Plugin executes <a href="#">NodeJS</a> 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

The screenshot shows the 'JDK installations' section of the Jenkins 'Tools' configuration. A new entry named 'jdk17' is being added. The 'Install automatically' option is selected. The 'Version' dropdown is set to 'jdk-17.0.8.1+1'. There is a 'Add Installer' button at the bottom.

The screenshot shows the 'NodeJS' configuration page. A new entry named 'node16' is being added. The 'Install automatically' option is selected. The 'Version' dropdown is set to 'NodeJS 16.2.0'. A note states: 'For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail' with a 'Force 32bit architecture' checkbox. A 'Global npm packages to install' section is present.

### 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

The screenshot shows the SonarQube administration menu. The 'Administration' tab is selected and highlighted with a red box. Below it, the 'Security' tab is also highlighted with a red box. A dropdown menu is open under the 'Security' tab, with the 'Users' option highlighted by a red box. Other options in the dropdown include 'Groups', 'Global Permissions', and 'Permission Templates'. The main menu bar includes 'Projects', 'Issues', 'Rules', 'Quality Profiles', 'Quality Gates', and 'Administration'.

click on update Token

The screenshot shows a user profile page for 'Administrator'. The 'Tokens' tab is selected and highlighted with a red box. Below the tabs are sections for 'SCM Accounts', 'Last connection', 'Groups', and 'Tokens'. Under 'Tokens', there are two entries: 'sonar-administrators' and 'sonar-users'. To the right of these entries is a button labeled 'Update Tokens'.

Create a token with a name and generate

The screenshot shows the 'Generate Tokens' page for the 'Administrator' user. It has fields for 'Name' (containing 'Jenkins') and 'Expires in' (set to '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\_21d162904c1c72cf8b39665f96480185c99dc2f9'. A table below lists the generated token details: Name (Jenkins), Type (User), Project (empty), Last use (Never), Created (September 8, 2023), Expiration (October 8, 2023), and a 'Revoke' button.

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 'Manage Jenkins' interface under the 'Tools' section. It is titled 'SonarQube Scanner installations'. A button 'Add SonarQube Scanner' is visible. Below it, a configuration card for 'SonarQube Scanner' is displayed. The 'Name' field contains 'sonar-scanner'. The 'Install automatically' checkbox is checked. Under the 'Install from Maven Central' section, the 'Version' dropdown is set to 'SonarQube Scanner 5.0.1.3006'. A 'Save' button is at the bottom left, and an 'Apply' button is at the bottom right.

In the Sonarqube Dashboard add a quality gate also

Administration → Configuration → Webhooks

The screenshot shows the Sonarqube administration interface at the URL <http://13.232.2.206:9000/admin/users>. The 'Administration' tab is selected in the top navigation bar. In the main content area, there is a sidebar with 'General Settings', 'Encryption', and 'Webhooks' (which is highlighted with a red box). Below the sidebar is a search bar 'Search by login or name...'. A table lists users, showing one entry: 'Administrator admin' (with a green 'A' icon), last connected ' $< 1$  hour ago', and group 'sonar-administrators'. There are columns for 'SCM Accounts', 'Last connection', 'Groups', and 'Tokens'. A 'Create User' button is located in the top right corner of the user list area.

Click on Create

The screenshot shows the SonarQube Administration interface. The top navigation bar includes links for sonarqube, Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and a search bar. The Administration tab is selected. Below it, the Webhooks section is visible, stating "No webhook defined." A "Create" button is located in the top right corner of the main content area.

## Add details

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

The screenshot shows the SonarQube Administration interface with the "Create Webhook" dialog box open. The dialog has fields for "Name" (set to "jenkins") and "URL" (set to "http://43.204.36.242:8090/sonarqube-webhook/"). Both fields have validation icons indicating they are correct. The "Secret" field is empty. At the bottom, there are "Create" and "Cancel" buttons, with "Create" being highlighted.

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 {
```

```

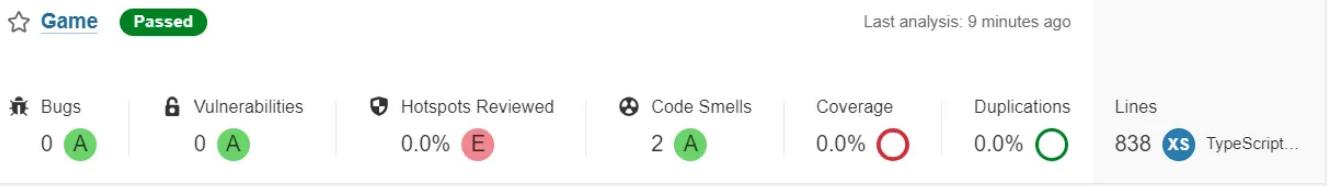
stage('clean workspace'){
    steps{
        cleanWs()
    }
}
stage('Checkout from Git'){
    steps{
        git branch: 'master', url: 'https://github.com/Aj7Ay/2048-React'
    }
}
stage("Sonarqube Analysis"){
    steps{
        withSonarQubeEnv('sonar-server') {
            sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=
-Dsonar.projectKey=Game '''
        }
    }
}
stage("quality gate"){
    steps {
        script {
            waitForQualityGate abortPipeline: false, credentialsId: 'Sc
        }
    }
}
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

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

The screenshot shows the Jenkins Plugins page with the following interface elements:

- Sidebar menu: Updates, Available plugins (selected), Installed plugins, Advanced settings, Download progress.
- Search bar: Search available plugins.
- Install button: A large blue "Install" button with a dropdown arrow.
- Plugin list table:
 

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 <a href="#">Dependency-Check</a> 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

Go to 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'
        dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
    }
}
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			
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

Search bar: docker

Install button

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.

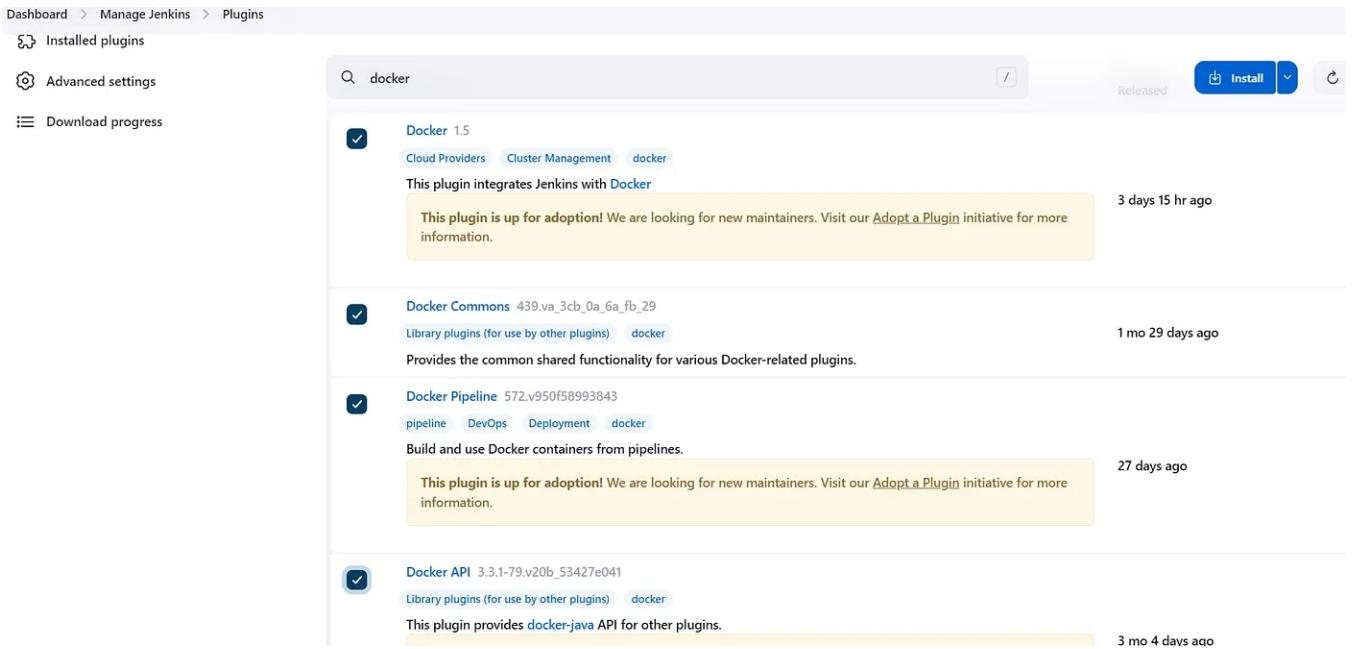
27 days ago

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

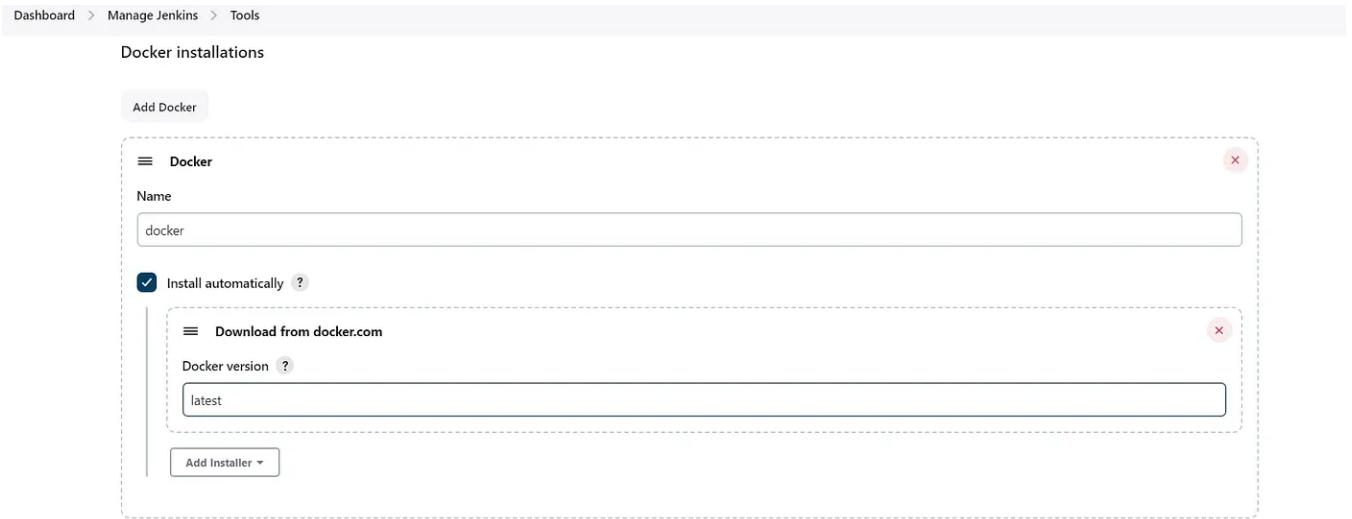
Name: docker

Install automatically: checked

Download from docker.com

Docker version: latest

Add Installer



Add Docker Hub Username and Password under Global Credentials

Kind

Username with password

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

Username ?  
sevenajay

Treat username as secret ?

Password ?  
.....

ID ?  
docker

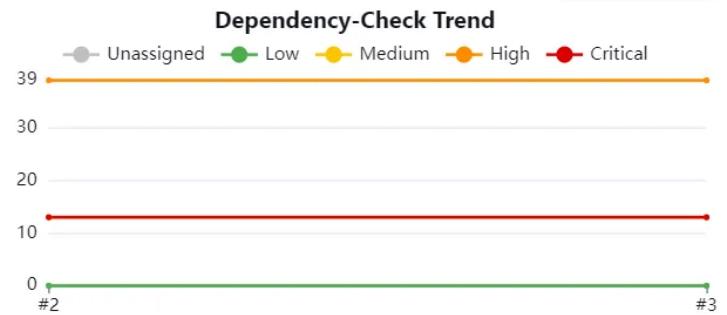
Description ?  
docker

**Create**

Add this stage to Pipeline Script

```
stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName: 'docke
                sh "docker build -t 2048 ."
                sh "docker tag 2048 sevenajay/2048:latest "
                sh "docker push sevenajay/2048:latest "
            }
        }
    }
}
stage("TRIVY"){
    steps{
        sh "trivy image sevenajay/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 Docker hub, you will see a new image is created

sevenajay / 2048

**Description**  
This repository does not have a description

Last pushed: a few seconds ago

---

**Tags**  
This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	---	a few seconds ago

**Docker commands**  
To push a new tag to this repository:

```
docker push sevenajay/2048:tagname
```

**Automated Builds**  
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#)

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

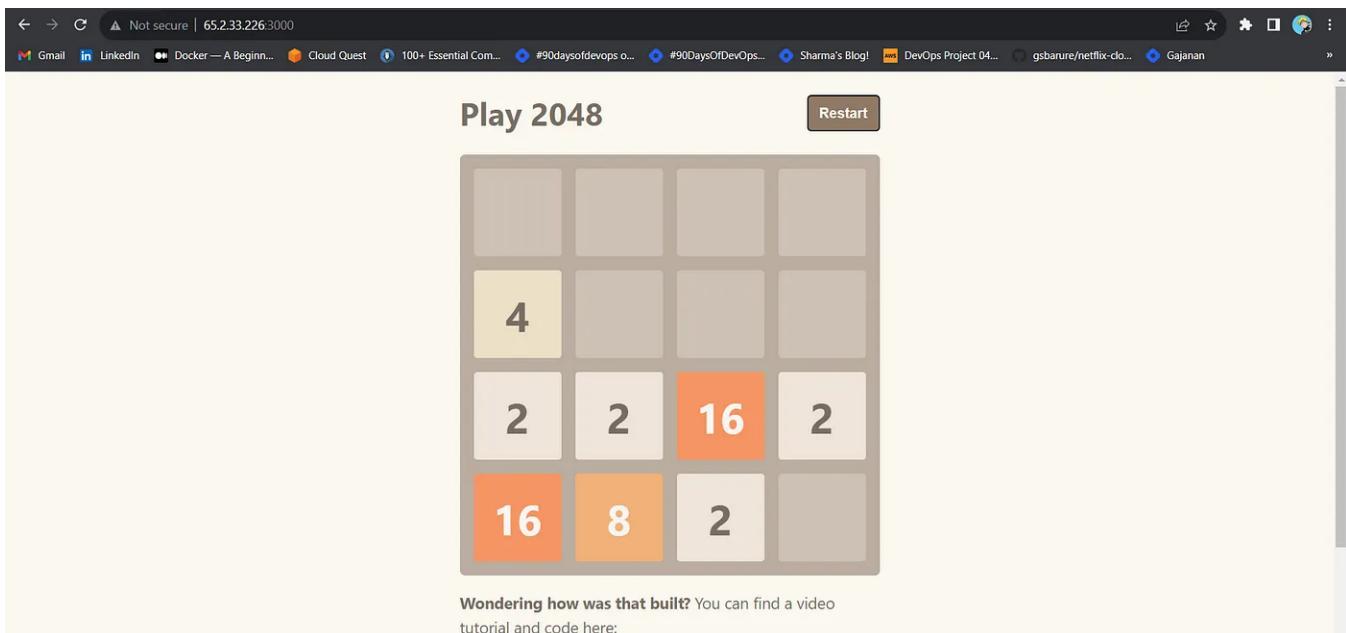
```
stage('Deploy to container'){
    steps{
        sh 'docker run -d --name 2048 -p 3000:3000 sevenajay/2048:latest'
    }
}
```

stage view



<Jenkins-public-ip:3000>

You will get this output



Play the game and make it 2048

## 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 -y
curl -LO https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stabl
```

```
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
```

## 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      # 3lines same command  
EOF  
  
sudo apt-get update  
sudo apt-get install -y kubelet kubeadm kubectl  
sudo snap install kube-apiserver
```

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

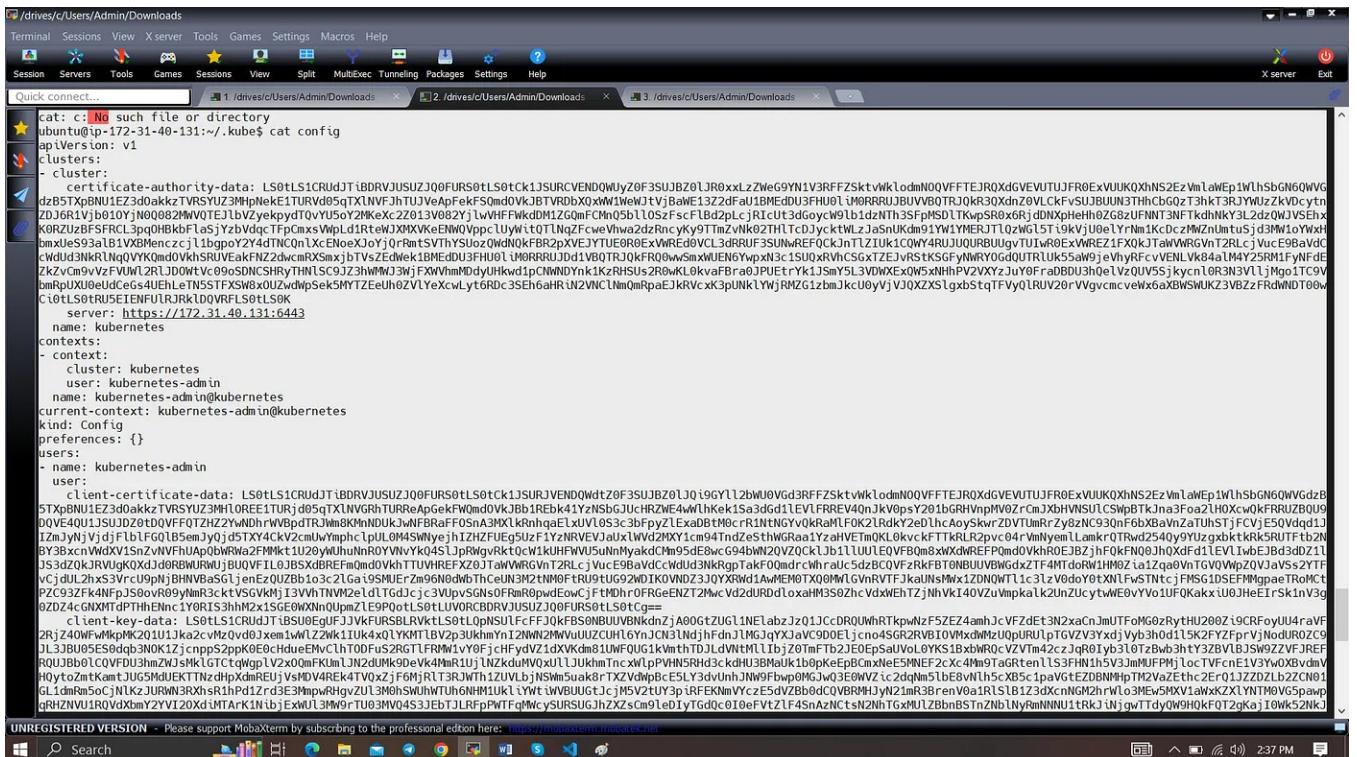
```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16  
# in case you're in root exit from it and run below commands  
mkdir -p $HOME/.kube  
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```

```
sudo chown $(id -u):$(id -g) $HOME/.kube/config  
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Docume
```

## - Worker Node -----

```
sudo kubeadm join <master-node-ip>:<master-node-port> --token <token> --discover
```

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



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 Kuberenetes credential section.

Install Kubernetes Plugin, Once it's installed successfully

**Plugins**

Available plugins

Install	Name	Released
<input checked="" type="checkbox"/>	Kubernetes Credentials 0.11 kubernetes credentials Common classes for Kubernetes credentials	9 days 16 hr ago
<input checked="" type="checkbox"/>	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
<input checked="" type="checkbox"/>	Kubernetes 4029.v5712230ccb_f8 Cloud Providers Cluster Management kubernetes Agent Management This plugin integrates Jenkins with Kubernetes	9 days 15 hr ago
<input checked="" type="checkbox"/>	Kubernetes CLI 1.12.1 kubernetes Configure kubectl for Kubernetes	8 days 22 hr ago

go to manage Jenkins → manage credentials → Click on Jenkins global → add credentials

### New credentials

Kind

Scope ?

File

Choose File Secret File.txt

ID ?

Description ?

**Create**

final step to deploy on the Kubernetes cluster

```
stage('Deploy to kubernetes'){
    steps{
        script{
            withKubeConfig(caCertificate: '', clusterName: '', contextName: '')
            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

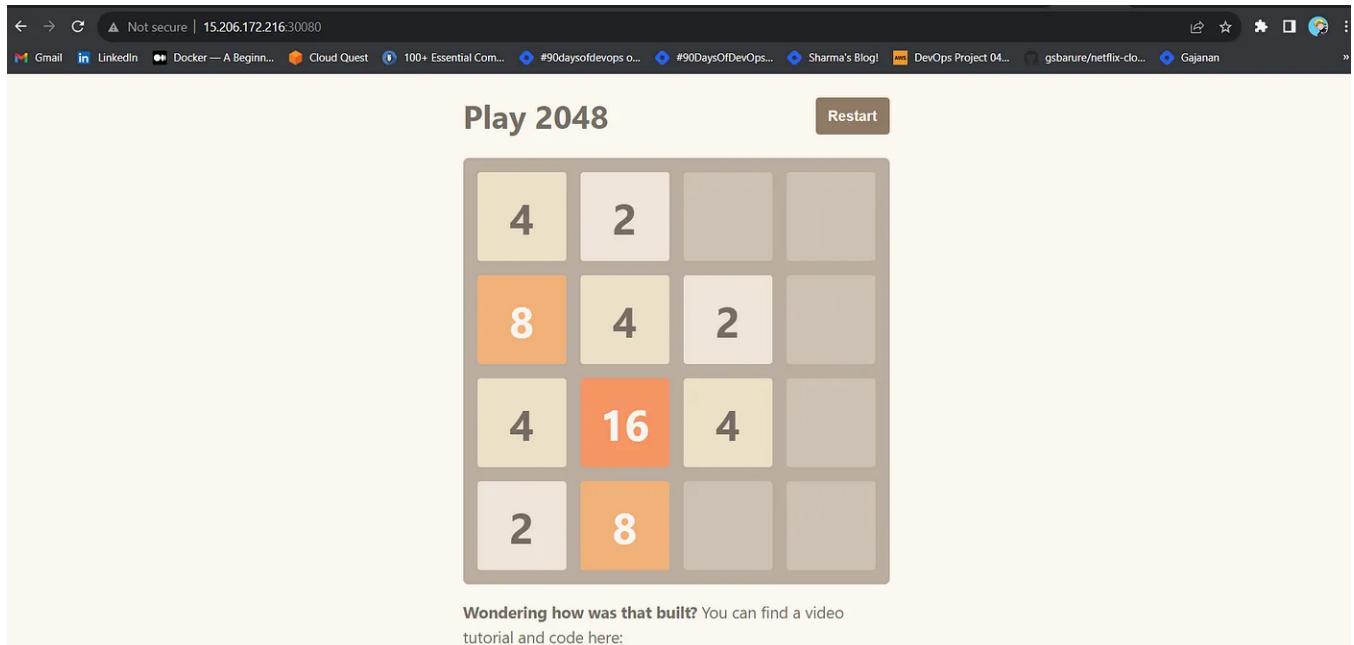
```
kubectl get all  
kubectl get svc #use anyone
```

```
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>

output:



## Step 10: Terminate instances.