

Deploying Code to Github -> Jenkins CI-CD Pipeline On AWS

Using Jenkins/SonarCube/Docker:

- Create a GitHub Repository and push your code to the Repository.

Requirements:

3-EC2 Instances: one for Jenkins, one for SonarCube, and another for Docker.

Go to your Amazon Console:

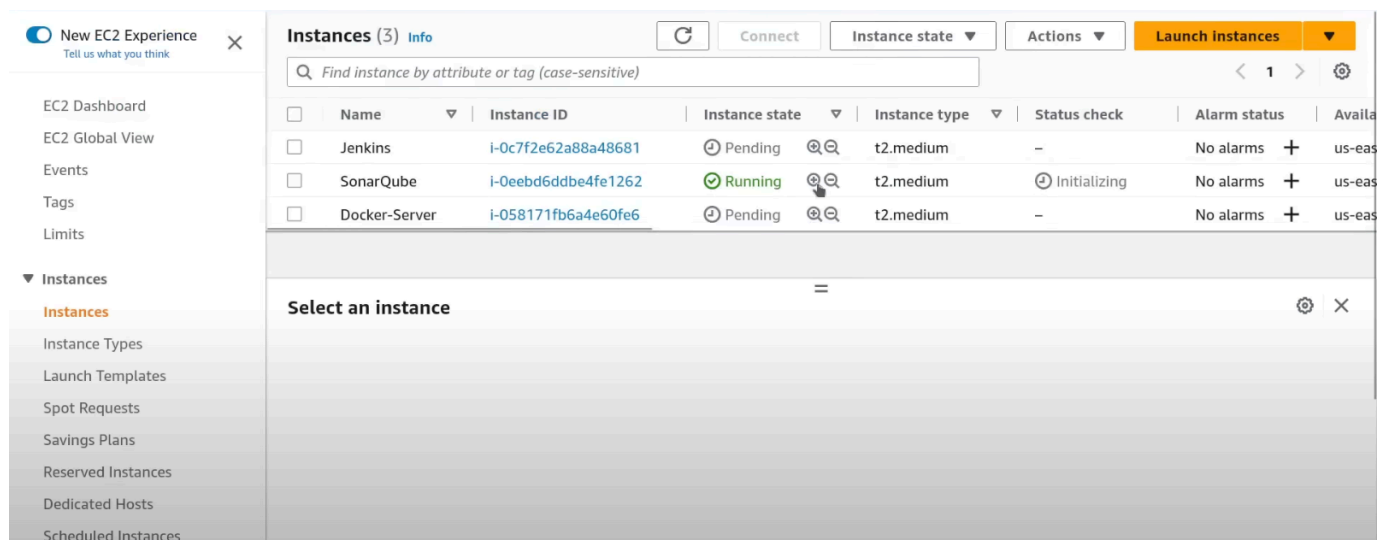
Launched 1st EC2 Instance -> Chosen Name as Jenkins -> Chosen Ubuntu as Operating System

-> Chosen Instance type as t2.micro -> Created a new key pair (helps to SSH into the instance, downloads your SSH key. pem file) -> Launch Instance.

Launched 2nd EC2 Instance -> Chosen Name as SonarCube -> Chosen Ubuntu as Operating System -> Chosen Instance type as t2.medium(it Consumes a lot of memory) -> Used earlier created key pair -> Launch Instance.

Launched 3rd EC2 Instance -> Chosen Name as Docker-Server -> Chosen Ubuntu as Operating System -> Chosen Instance type as t2.micro -> Used earlier created key pair -> Launch Instance.

Output:



The screenshot shows the AWS Management Console 'Instances' page. The table lists three instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Jenkins	i-0c7f2e62a88a48681	Pending	t2.medium	-	No alarms	us-east-1
SonarQube	i-0eebd6ddbe4fe1262	Running	t2.medium	Initializing	No alarms	us-east-1
Docker-Server	i-058171fb6a4e60fe6	Pending	t2.medium	-	No alarms	us-east-1

Below the table, a modal window titled 'Select an instance' is open, showing a search bar and a list of instances.

SSH Into EC2 Instances:

Jenkins:

Open your terminal and navigate to the SSH-Key-Pair folder. Before SSH, Check for permissions and Give permissions using the below command, if required.

- `chmod 400 SSH-Key-Pair.pem`
- `ssh -i SSH-Key-Pair.pem ubuntu@ip`
- hostname change: `sudo hostnamectl set-hostname Jenkins`

- sudo apt update (It will update)

Installing Jenkins:

To install Jenkins, we need to have a Java 11 runtime environment. We can install this using the below command.

- sudo apt install openjdk-11-jre(it will make your environment ready).

Go to the below link(Jenkins.io) and copy the bash command to install Jenkins:

- <https://www.jenkins.io/doc/book/installing/linux/#debianubuntu>

The screenshot shows the Jenkins website's 'Debian/Ubuntu' installation page. The header includes the Jenkins logo and navigation links like 'Blog', 'Success Stories', 'Documentation', 'Plugins', 'Community', 'Subprojects', 'Security', 'About', and 'Download'. A search bar is on the right. The left sidebar contains a 'User Handbook' menu with links to 'User Handbook Overview', 'Installing Jenkins' (which is highlighted), 'Docker', 'Kubernetes', 'Linux', 'macOS', 'Windows', 'Other Systems', 'WAR file', 'Other Servlet Containers', 'Offline Installations', 'Initial Settings', 'Using Jenkins', 'Pipeline', 'Blue Ocean', and 'Managing Jenkins'. The main content area is titled 'Debian/Ubuntu' and contains an information icon, a note about OpenJDK 11 being replaced by OpenJDK 17, a 'Long Term Support release' section, and a terminal code block for installation instructions.

```
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
sudo apt-get install jenkins
```

```
- 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
sudo apt-get install jenkins
```

This will install updated Jenkins on your Ec2 Instance.

To verify whether Jenkins is installed or not use the below command

- systemctl status Jenkins

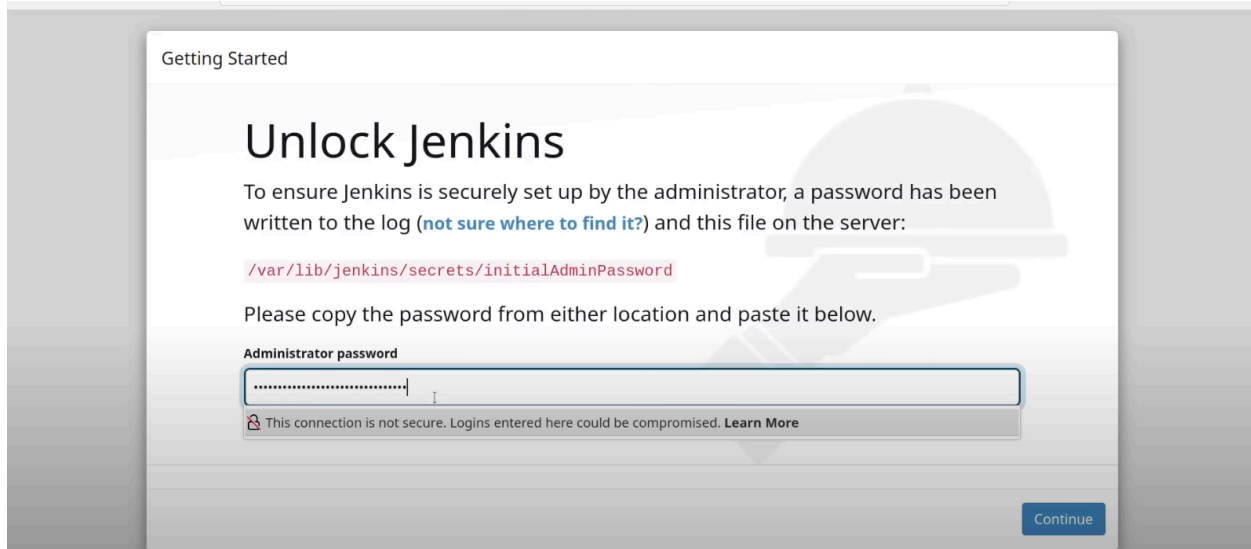
To access this instance, you need to enable default port 8080.

- EC2 instance for Jenkins -> Security Groups -> Edit Inbound Rules -> Add Rule with 8080 as shown below.

The screenshot shows the 'Inbound rules' section of the AWS Management Console. At the top, it says 'Inbound rules control the incoming traffic that's allowed to reach the instance.' Below this is a table of rules. The first rule has a 'Security group rule ID' of 'sgr-0cc6f6de3ce358754', 'Type' of 'SSH', 'Protocol' of 'TCP', 'Port range' of '22', 'Source' of 'Custom', and a 'Description - optional' field. The second rule has a 'Type' of 'Custom TCP', 'Protocol' of 'TCP', 'Port range' of '8080', 'Source' of 'Anywh...', and a 'Description - optional' field. Both rules have '0.0.0.0/0' in the source field. At the bottom, there are buttons for 'Add rule', 'Cancel', 'Preview changes', and 'Save rules'.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0cc6f6de3ce358754	SSH	TCP	22	Custom	
-	Custom TCP	TCP	8080	Anywh...	

- Manage Jenkins -> Manage Plugins -> Search & Install SonarCube Scanner, SSH2 easy plugin.(it will be used later for sonar qube).
- One can copy the IP address of the instance & port, and paste it in the browser. x.x.x.x:8080:

The image shows the 'Getting Started' page of Jenkins with the title 'Unlock Jenkins'. It explains that a password has been written to the log and a file on the server. The file path `/var/lib/jenkins/secrets/initialAdminPassword` is shown in red. It instructs the user to copy the password from either location and paste it into the 'Administrator password' field. A security warning states: 'This connection is not secure. Logins entered here could be compromised. Learn More'. A 'Continue' button is at the bottom right.

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

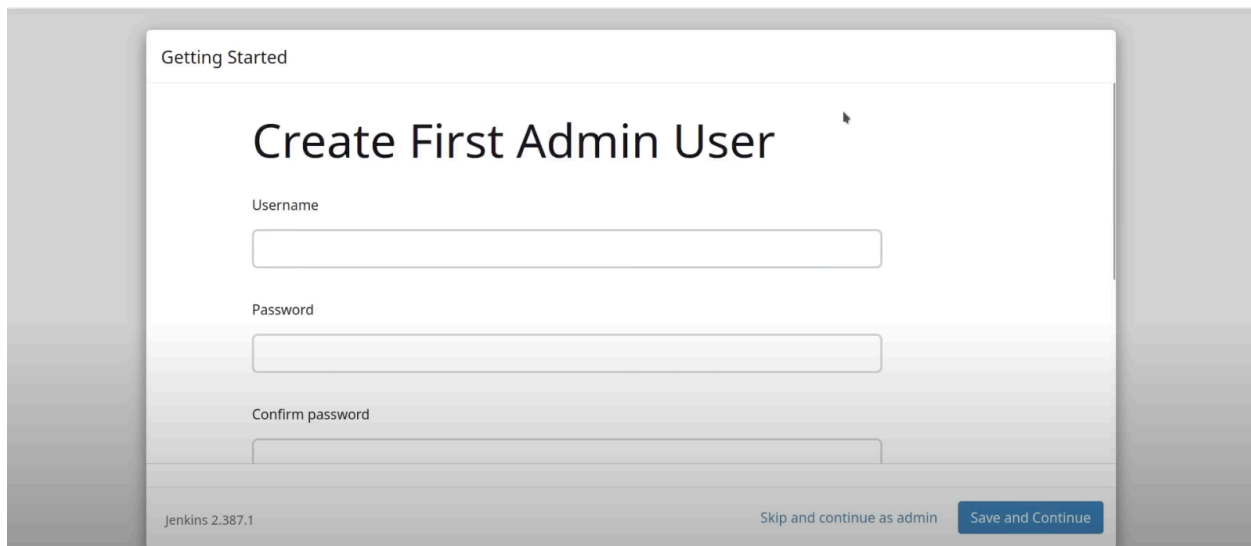
 This connection is not secure. Logins entered here could be compromised. [Learn More](#)

Continue

As suggested, navigate through the path and copy the encrypted key/password & paste it here.

- `sudo cat /var/lib/jenkins/secrets/initialAdminPassword`
- Install all the suggested plugins.

Now Create a User with the necessary details and start Jenkins as shown below:

The image shows the 'Getting Started' page of Jenkins with the title 'Create First Admin User'. It contains three input fields: 'Username', 'Password', and 'Confirm password'. At the bottom left, it says 'Jenkins 2.387.1'. At the bottom right, there are two buttons: 'Skip and continue as admin' and 'Save and Continue' (highlighted in blue).

Getting Started

Create First Admin User

Username

Password

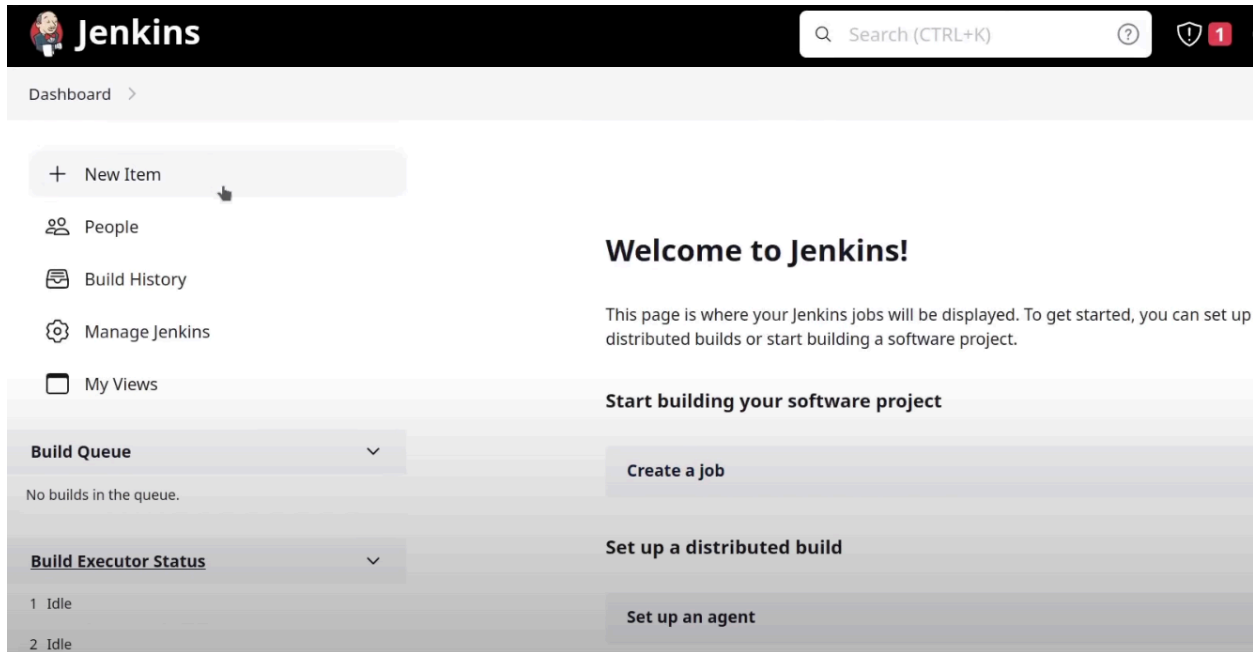
Confirm password

Jenkins 2.387.1

Skip and continue as admin

Save and Continue

Jenkins UI:



Create a pipeline:

- New Item -> Name(ex: Automated-Pipeline) -> Select Project(ex:- Freestyle Project).
- Once the project is created -> Source Code Management -> Git -> Give the git details.
- After that go to Configure -> enable GitHub hook triggers for GITScm polling (Enabling hooks triggers whenever there is a change in your git hub repository).
- Go to Repository settings, Enable WebHooks using Jenkins ip: port & push, pull events.

SonarCube:

Open your terminal and navigate to the SSH-Key-Pair folder.

- `ssh -i SSH-Key-Pair.pem ubuntu@ip`
- hostname change: `sudo hostnamectl set-hostname Jenkins`
- `sudo apt update` (It will update)

Installing SonarCube:

To install SonarCube, we need to have a Java 17 runtime environment. We can install this using the below command.


- `sudo apt install openjdk-17-jre`(it will make your environment ready).

Go to the below link(SonarCube) and copy the link to install SonarCube Community edition:

- <https://www.sonarsource.com/products/sonarcube/downloads/success-download-community-edition/>
- `wget url` -> will download the sonar cube.
- `unzip sonarcube.zip`

Version 10.2 | Released September 2023

Community Edition	Developer Edition	Enterprise Edition	Data Center Edition
Used and loved by 400,000+ companies.	Built for developers by developers	Designed to meet Enterprise requirements	Designed for high availability
DOWNLOAD FOR FREE	DOWNLOAD	DOWNLOAD	DOWNLOAD
All of the following features: <ul style="list-style-type: none"> Static code analysis for 19 languages: Java, C#, JavaScript, TypeScript, CloudFormation, Terraform, Docker, Kubernetes, Kotlin 	Community Edition plus: <ul style="list-style-type: none"> Support for C, C++, Obj-C, Swift, ABAP, T-SQL and PL/SQL Detection of advanced vulnerabilities including 	Developer Edition plus: <ul style="list-style-type: none"> Support for Apex, COBOL, PL/I, RPG and VB6 Portfolio Management & PDF Executive Reports 	Enterprise edition plus: <ul style="list-style-type: none"> Component redundancy Data r Horizon

Hey! Thanks for your interest in SonarQube. What are you looking for today?


- Go to the bin folder and choose OS to download.
- ./sonar.sh, Console(To see logs of working Sonar).

This will install the updated SonarCube on your Ec2 Instance.
To verify whether SonarCube is installed or not use the below command.

- systemctl status SonarCube.

To access this instance, you need to enable default port 9000.

- EC2 instance for SonarCube -> Security Groups -> Edit Inbound Rules -> Add Rule with 9000 as shown below.

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

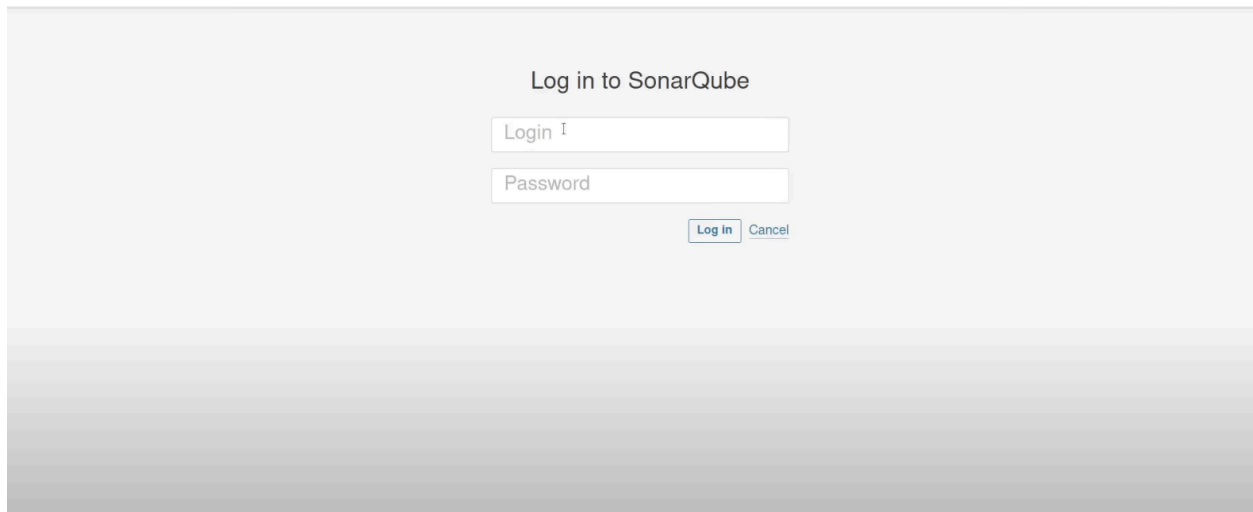
Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>	
sgr-016ca6f0ac5e9f015	SSH	TCP	22	Custom <input type="text" value="0.0.0.0/0"/>		Delete
-	Custom TCP	TCP	9000	Custom <input type="text" value=""/>		Delete

[Add rule](#)

Cancel [Preview changes](#) [Save rules](#)

One can copy the IP address of the instance & port, and paste it in the browser. x.x.x.x:9000:

- By default username: admin, password: admin & then create a new password.

The image shows the SonarQube login interface. At the top, it says "Log in to SonarQube". Below this, there are two input fields: "Login I" and "Password". At the bottom right of the form, there are two buttons: "Log In" and "Cancel". The background is a light gray gradient.

Create Manual Project:

- Give the project name, key as you like, and corresponding branch.
- Now Using Continuous Integration CI Jenkins, Choose the DevOps platform as GitHub.
- Configure Analysis -> Continue -> Choose Build based on your requirements(Other).
- In Security, create a SonarCube token.
- Go to Jenkins -> Manage Jenkins -> Global Tool Configuration -> Add Sonar Qube Scanner.
- Go to Configure System of Manage Jenkins -> Add SonarQube Server with IP: Port.
- Try Build Now

DOCKER:

Docker:

Open your terminal and navigate to the SSH-Key-Pair folder. Before SSH, Check for permissions and Give permissions using the below command, if required.

- `chmod 400 SSH-Key-Pair.pem`
- `ssh -i SSH-Key-Pair.pem ubuntu@ip`
- hostname change: `sudo hostnamectl set-hostname docker`
- `sudo apt update` (It will update)

Installing Docker:

To install Docker, Use the below command.

- Switch to another user: `sudo su Jenkins`
- # Add Docker's official GPG key:

```
sudo apt-get update
```

```
sudo apt-get install ca-certificates curl gnupg
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/
```

```
keyrings/docker.gpg
```

Add the repository to Apt sources:

```
echo \
```

```
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://
```

```
download.docker.com/linux/ubuntu \
```

```
"$(. /etc/os-release && echo "$VERSION_CODENAME)" stable" | \
```

```
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt-get update
```

- `sudo apt-get install docker-ce docker-ce-cli containerd.io docker-build-plugin docker-compose-plugin.`
- Create a password for your docker user. (- passwd ubuntu)

The screenshot shows the Docker documentation website for Ubuntu. The left sidebar contains a navigation menu with categories like Docker Desktop, Docker Extensions, Docker Engine, Overview, Install, Overview, CentOS, Debian, Fedora, RHEL, SLES, Ubuntu (highlighted), Raspbian, Binaries, Post-installation steps, Troubleshoot installation, and Storage. The main content area is titled 'Install using the Apt repository' and includes a note about setting up the repository for new machines. It lists the steps to install Docker, including adding the GPG key and updating the package list. The right sidebar shows a 'Contents' section with links to prerequisites, OS requirements, uninstalling old versions, installation methods, and related content. A 'Give feedback' button is visible on the far right.

docker docs Guides Manuals Reference Samples FAQ

Search [K]

Docker Desktop >
Docker Extensions >
Docker Engine >
Overview >
Install >
Overview >
CentOS >
Debian >
Fedora >
RHEL >
SLES >
Ubuntu >
Raspbian >
Binaries >
Post-installation steps >
Troubleshoot installation >
Storage >

Install using the Apt repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

1. Set up Docker's Apt repository.

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/ap
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] ht
"${. /etc/os-release && echo "$VERSION_CODENAME)" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

8 minute read
Edit this page
Request changes

Contents

- Prerequisites
- OS requirements
- Uninstall old versions
- Installation methods
- Install using the Apt repository**
- Install from a package
- Uninstall Docker Engine
- Next steps

Related content

- Install Docker Desktop on Ubuntu
- Install Docker Desktop on Mac
- Install the Compose plugin
- Install Docker Desktop on Windows
- Install Docker Engine on CentOS

Give feedback

This will install an updated Docker on your Ec2 Instance.

To verify whether Docker is installed or not use the below command

- `systemctl status docker`

To access this instance and website, you need to enable default port 22, 8085.

- EC2 instance for Docker -> Security Groups -> Edit Inbound Rules -> Add Rule with 22, 8085 as shown below.

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sgr-078ed2a15a3c8803c	SSH	TCP	22	Custom		Delete
-	Custom TCP	TCP	8085	Anywhere		Delete

[Add rule](#) [Cancel](#) [Preview changes](#) [Save rules](#)

Integrating Jenkins & Docker:

- Check, if you can ssh from Jenkins to Docker.
- Edit sshd config in docker, to give permissions for Jenkins to access.
- `nano /etc/ssh/sshd_config`
- Make sure your `PubkeyAuthentication` Yes uncommented & `PasswordAuthentication` YES
- restart ssh service. `systemctl restart sshd`.
- Now you can ssh from Jenkins to Docker. (Your in Jenkins Console).
- Generate Keypair public/private (`ssh-keygen`).
- `ssh-copy-id ubuntu@dockerIP` from Jenkins Console.

Dashboard > Manage Jenkins > Configure System >

Server Group List :

Create the server groups for your projects

Group Name ?

Docker-Servers

SSH Port ?

22

User Name ?

ubuntu

Password ?

[Save](#) [Apply](#)

- Now, Jenkins -> Manage Jenkins -> configure system -> Under server group center -> create a Docker Server Group(port 22) -> Add the server under server list.

Add Server under Server list:

Dashboard > Manage Jenkins > Configure System >

add the server under this server group for your projects

Server Group: ×
 Docker-Servers

Server Name ?
 Docker-1

Server IP ?

- Go to Post-build Actions and look for Remote Shell, Select the Docker server and create a file using touch to check whether everything is working correctly or not.
- Add the current user to the docker group, so that it will. Give permissions to Execute Shell.
- sudo usermod -aG docker ubuntu & then newgrp docker & then docker ps should work.

Docker post build Actions:

Dashboard > Automated-Pipeline > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions**

Target Server ?
 Docker-Servers--Docker-1--54.163.4.0

shell ?

```
cd /home/ubuntu/website
docker build -t mywebsite .
docker run -d -p 8085:80 --name=Only-Website mywebsite
```

Post-build Actions

- Under Excuse shell -> give a path to your directory -> docker build -t name -> docker run as shown above.
- you should able to see your website with dockerIP:8085.