
CI/CD

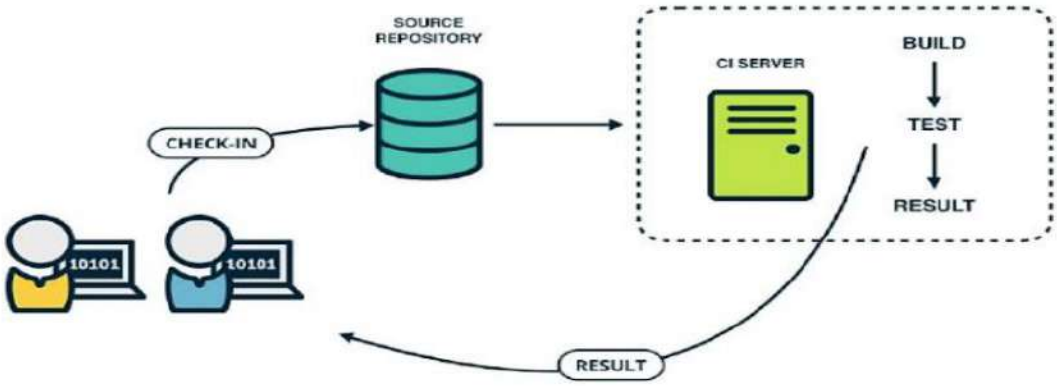
WHAT IS CI/CD

- **CI/CD** is a method to frequently deliver apps to customers by automation into the stages of app development
- Main concepts of CI/CD are **continuous integration**, **continuous delivery**, and **continuous deployment.**
- **CI/CD** is a solution to the problems integrating new code can cause for development and operations teams.

DIFFERENCE BETWEEN CI AND CD (AND OTHER CD)

CONTINUOUS INTEGRATION (CI):

- Developers regularly merge their code changes into a central repository, after which automated builds and tests are run.
- CI refers to the build or integration stage of the software release process and entails both an automation component and a cultural component.
- The key goals of CI are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.



- Developers check out code into their own workspaces. When done, commit the changes to the repository.
- CI server monitors the repository and checks out changes when they occur.
- CI server builds the system and runs unit and integration tests.
- CI server releases deployable artefacts for testing.
- CI server assigns a build label to the version of the code it just built.
- CI server informs the team of the successful build.
- If the build or tests fail, the CI server alerts the team.
- The team fixes the issue at the earliest opportunity.

CONTINUOUS DELIVERY (CD):

- CD is the automated delivery of completed code to environments like testing and development.
- CD provides an automated and consistent way for code to be delivered to these environments.

CONTINUOUS DEPLOYMENT (CD):

- CD is the next step of continuous delivery.
- Every change that passes the automated tests is automatically placed in production, resulting in many production deployments.
- In practice, continuous deployment means that a developer's change to a cloud application could go live within minutes of writing it (assuming it passes automated testing).

CI/CD TOOLS:

- The popular CI / CD tools are:
    - Jenkins
    - Drone CI (CD Platform written in GO)
    - TeamCity (JetBrains)
    - Wercker
    - CircleCI
    - CodeShip
    - SemaPhoreCI

Jenkins

JENKINS

- Jenkins is an **Open-Source CI/CD tool** written in **Java.**
- It is an Automation Tool, used to **Build** and **Deliver** the Software Product.
- Jenkins was forked from Another Project called **Hudson,** after dispute with Oracle.
- It is a server-based application and requires a web server like **Apache Tomcat.**
- Jenkins monitoring of repeated tasks which arise during the development of a project.

E.g.: Your team is developing a project; Jenkins will continuously test your project builds and show you the errors in early stages of your development.

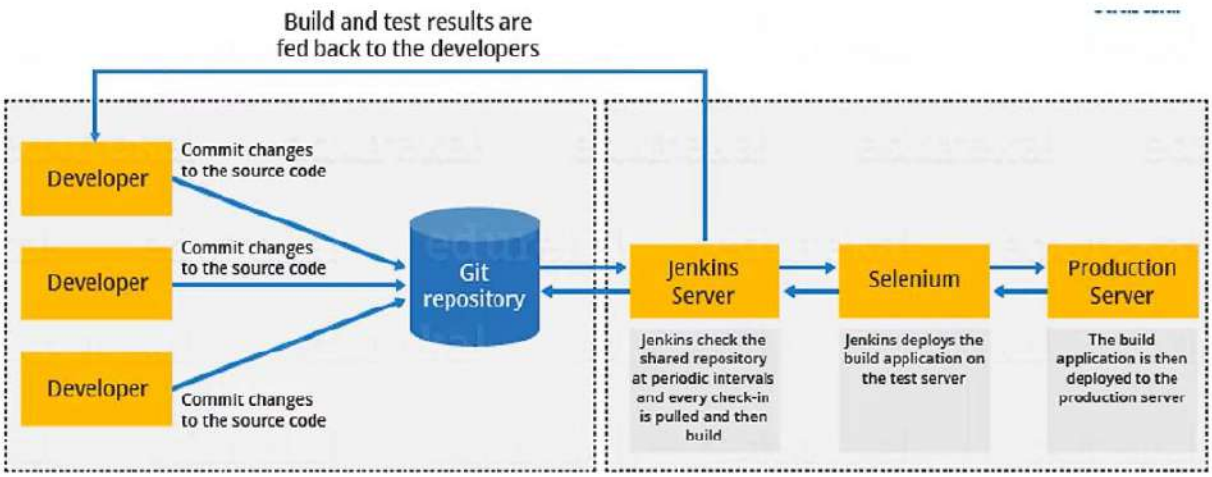
WHY USE CONTINUOUS INTEGRATION WITH JENKINS:

- Code is built and test as soon as Developer commits code.
- Jenkins will build and test code many times during the day.
- On Successful build, Jenkins will deploy the source into test server and notifies the deployment team.
- On Build Failures, Jenkins will notify the errors to the developer team.
- Code is built immediately after any of the Developer commits.
- Automated build and test process saving timing and reducing defects.

ADVANTAGE OF USING JENKINS:

- Jenkins is being managed by the community which is very open.
- Jenkins has around 320 plugins published in its plugins database.
- It supports cloud-based architecture so that you can deploy Jenkins in cloud-based platforms.
- It Support Docker Containers, you can containerize Jenkins Service.

JENKINS ARCHITECTURE



This single Jenkins server was not enough to meet certain requirements like:

- Sometimes you might need several different environments to test your builds. This cannot be done by a single Jenkins server.
- If larger and heavier projects get built on a regular basis then a single Jenkins server cannot simply handle the entire load.
- A standalone Jenkins instance can grow fairly quickly into a disk-munching, CPU-eating monster.

To prevent this from happening, we can scale Jenkins by implementing a slave node architecture, which can help us offload some of the responsibilities of the master Jenkins instance.

JENKINS SERVER:

- Jenkins checks the shared repository at periodic intervals and every check-in is pilled and then build.

SELENIUM:

- Jenkins deploys the build application on the Test Server.

PRODUCTION SERVER:

- The Build application is the deployed to the Production server.

INSTALLING JENKINS ON CENTOS / RHEL 9

PREREQUISITES:

- Jenkins installers are available for several Linux distributions.
    - Debian/Ubuntu, Fedora, Red Hat / Centos
    - **Running Port:** Tomcat-8080

MINIMUM HARDWARE REQUIREMENTS:

- 256 MB of RAM
- 1 GB of drive space (10 GB is a recommended minimum if running Jenkins as a Docker container)

Recommended hardware configuration for a small team:

- 4 GB+ of RAM
- 50 GB+ of drive space

SOFTWARE REQUIREMENTS:

- Java 8 / 11 runtime environments are supported in both 32-bit and 64-bit versions

STEP 1: Setting System Hostname

#hostname Jenkins-Master

#vim /etc/hostname

Jenkins-Master

#bash

STEP2: Security-Enhanced Linux is being disabled or in permissive mode.

#sed -i 's/SELINUX=.*/SELINUX=disabled/g' /etc/selinux/config

#setenforce 0

STEP 3: Installing Java

#yum install java-11-openjdk -y

#java –version

STEP 4: Adding Jenkins repository and import the repository GPG key

#wget -O /etc/yum.repos.d/jenkins.repo \

https://pkg.jenkins.io/redhat-stable/jenkins.repo

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

#yum upgrade

STEP 5: Installing Jenkins

#yum install jenkins -y

#systemctl daemon-reload

#systemctl start jenkins

#systemctl enable jenkins

#systemctl status jenkins

#netstat -pantl

STEP 6: Testing through web user interface

Open a Web browser type

http://IP-Address:8080

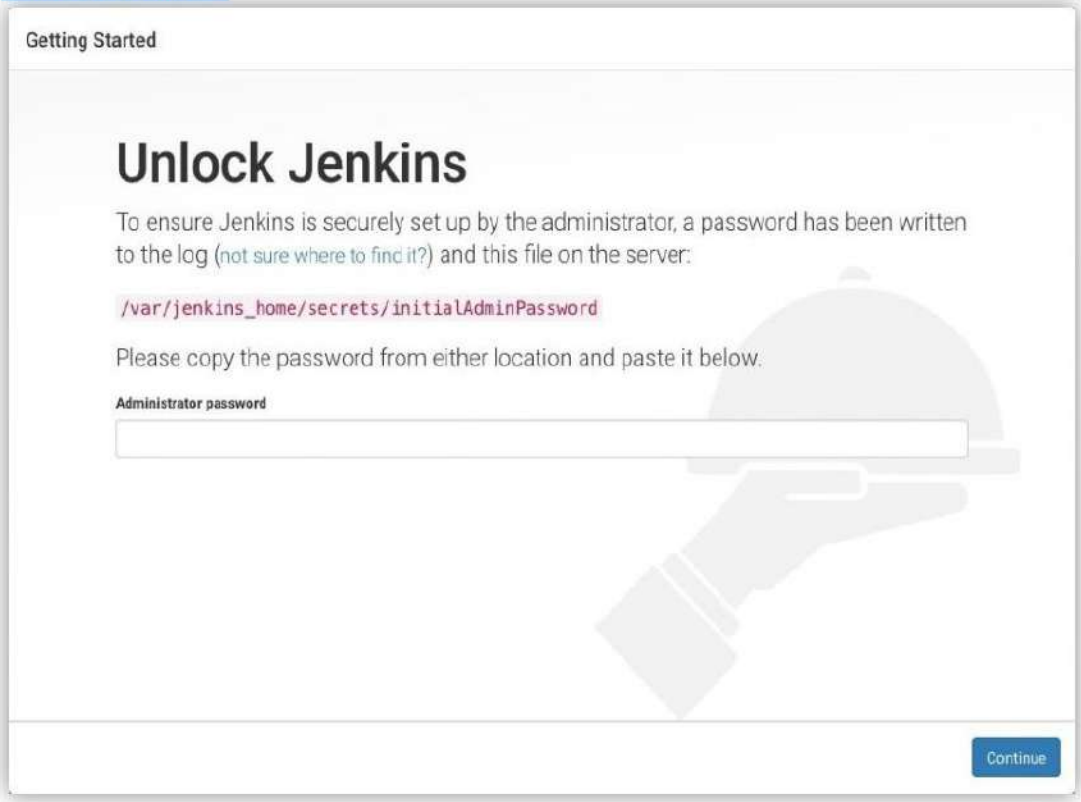
Post-installation setup wizard:

- After downloading, installing and running Jenkins using one of the procedures above (except for installation with Jenkins Operator), the post-installation setup wizard begins.
- This setup wizard takes you through a few quick "one-off" steps to unlock Jenkins, customize it with plugins and create the first administrator user through which you can continue accessing Jenkins.

Unlocking Jenkins:

- Browse to http://IP-Address:8080 and wait until the Unlock Jenkins page appears.

Jenkins page appears.



Collect automatic password from Jenkins Server:

2. From the Jenkins console log output, copy the automatically-generated alphanumeric password (between the 2 sets of asterisks).

```
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@24cf7484: defining b
eans [filter,legacy]; root of factory hierarchy
Sep 30, 2017 7:18:39 AM jenkins.install.SetupWizard init
INFO:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

2f064d2563814887864b6839405725 67

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

--> setting agent port for jnlp
--> setting agent port for jnlp... done
Sep 30, 2017 7:18:51 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
Sep 30, 2017 7:18:53 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Sep 30, 2017 7:18:59 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tools.JDKInstaller
Sep 30, 2017 7:18:59 AM hudson.model.AsyncPeriodicWork$1 run
INFO: Finished Download metadata. 25,543 ms
```

Note:

- The command: `sudo cat /var/lib/jenkins/secrets/initialAdminPassword` will print the password at console.
- If you are running Jenkins in Docker using the official `jenkins/jenkins` image you can use `sudo docker exec ${CONTAINER_ID or CONTAINER_NAME} cat /var/jenkins_home/secrets/initialAdminPassword` to print the password in the console without having to exec into the container.

3. On the **Unlock Jenkins** page, paste this password into the **Administrator password** field and click **Continue**.

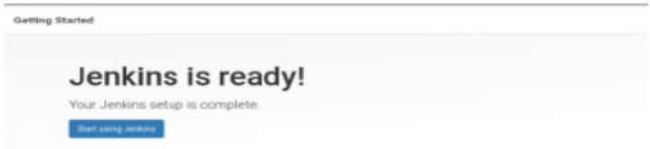
Creating the First administrator user:

Finally, after customizing Jenkins with plugins, Jenkins asks you to create your first administrator user.

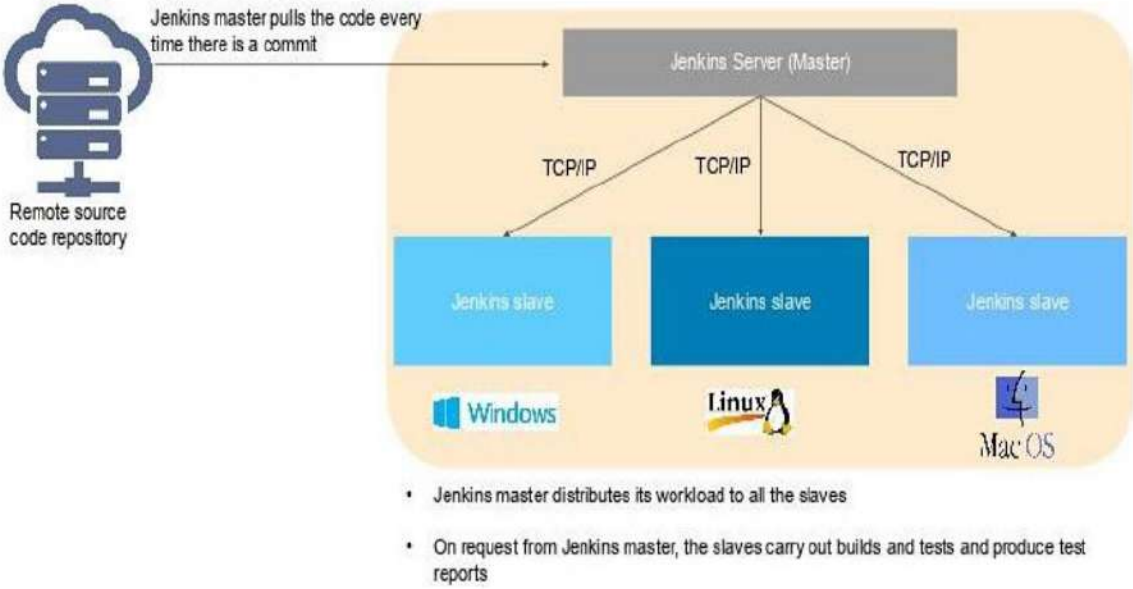
1. When the **Create First Admin User** page appears, specify the details for your administrator user in the respective fields and click **Save and Finish**.
2. When the **Jenkins is ready** page appears, click **Start using Jenkins**.

   Notes:

   - This page may indicate **Jenkins is almost ready!** instead and if so, click **Restart**.
   - If the page does not automatically refresh after a minute, use your web browser to refresh the page manually.
3. If required, log in to Jenkins with the credentials of the user you just created and you are ready to start using Jenkins!



JENKINS MASTER-AGENT ARCHITECTURE



JENKINS MASTER:

- The Jenkins master simply represents the base installation of Jenkins.
- The master will continue to perform basic operations and serve the user interface, while the slaves do the heavy lifting.
- Master will be scheduling build jobs.
- Dispatching builds to the slaves for the actual execution.
- Monitor the slaves.
- Recording and presenting the build results.
- A Master instance of Jenkins can also execute build jobs directly.

JENKINS SLAVE

- A Slave is a Java executable that runs on a remote machine.
- Slaves can run on a variety of operating systems.
- It hears requests from the Jenkins Master instance.
- You can configure a project to always run on a particular Slave machine or simply let Jenkins pick the next available Slave.

JENKINS AND AGENTS

- Jenkins runs its jobs on agents, choosing them based on availability. You can add a Jenkins agent manually, and Jenkins doesn't know or care whether an agent is a physical or virtual machine.
- To set this up, you must configure a Instance before adding it as a Jenkins agent.

STEP 1: Setting Up Hostname
#hostname Jenkins-Agent
#vim /etc/hostname
Jenkins-Agent
#bash

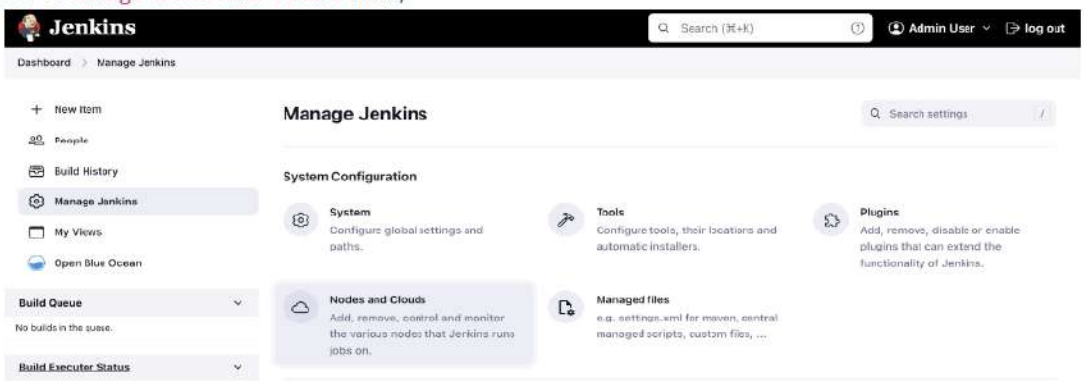
STEP 2: Installing Java
#yum install java-11-openjdk
#java –version

STEP 3: Create directory for saving Jobs
#mkdir /opt/jenkins

ADD AN INSTANCE AS A JENKINS AGENT:

To add your VM as a node in Jenkins, go to the Manage Jenkins panel and select Manage Nodes.

1. Go to your Jenkins dashboard;
2. Go to `Manage Jenkins` option in main menu;
3. Go to `Manage Nodes and clouds` item;



Provide a name for the node, select **Permanent Agent**, and then click **OK**.

Add Remote root directory: /opt/Jenkins

Label: Jenkins-Slave

Launch method: Launch agent by connecting it to the controller

Select Use WebSocket option

Finally: Save

NOTE: By default, Node has been Offline

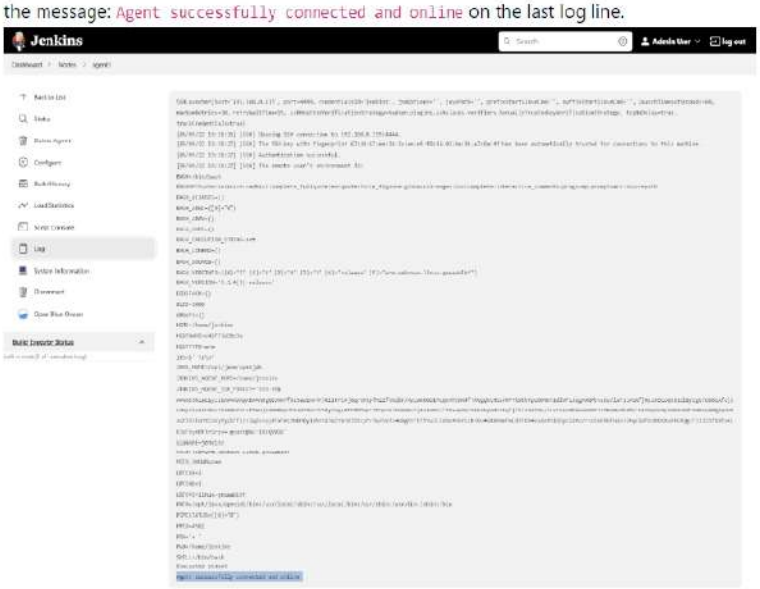
Connecting Jenkins-Agent to Jenkins-Master:

Go to Jenkins-agent Machine
Open Command Line Interface
Download and Run the Following commands

STEP 1: Download agent.jar file
#wget **agent,ar** file url adddres

STEP 2: Run the command
#java -jar **agent.jar** -jnlpUrl http://54.80.2.90:8080 /computer/…… etc

8. You should now see `This node is being launched.`. If that's not the case, you can now press the `Relaunch agent` button and wait a few seconds. You can now click on the `Log` button on the left, then you should receive the message: `Agent successfully connected and online` on the last log line.

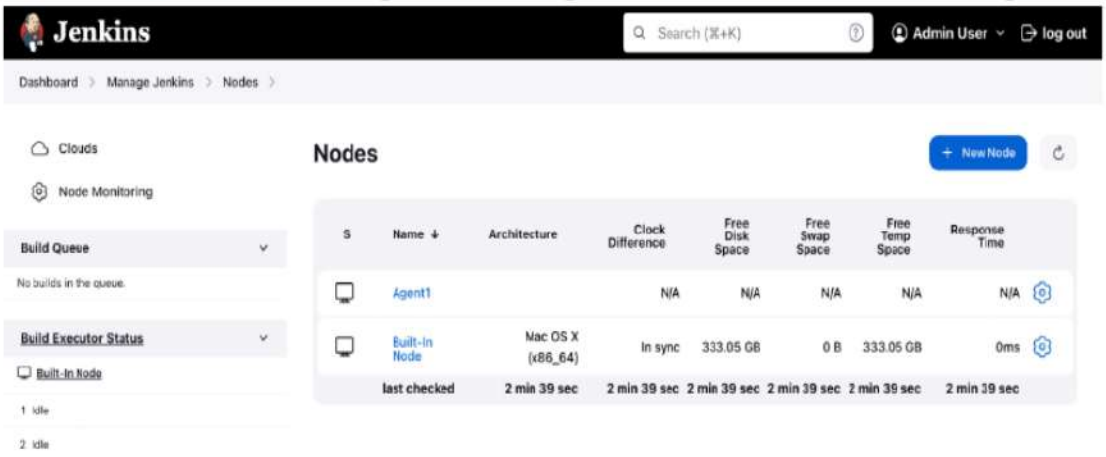


. . .

If your Jenkins controller does not start the agent via ssh, please check the port you configured on your agent. Copy it, and then click on the `Advanced...` button. You will then be able to paste the port number into the `Port` textfield.

Now Agent Will be Online

Press the `Save` button and the agent1 will be registered, but offline for the time being. Click on it.



Delegating the first job to Jenkins-Agent:

Start your build: In your local instance of Jenkins, start a build. Jenkins executes all commands on yo+ur Instance.