# Jenkins Master/Slave Setup

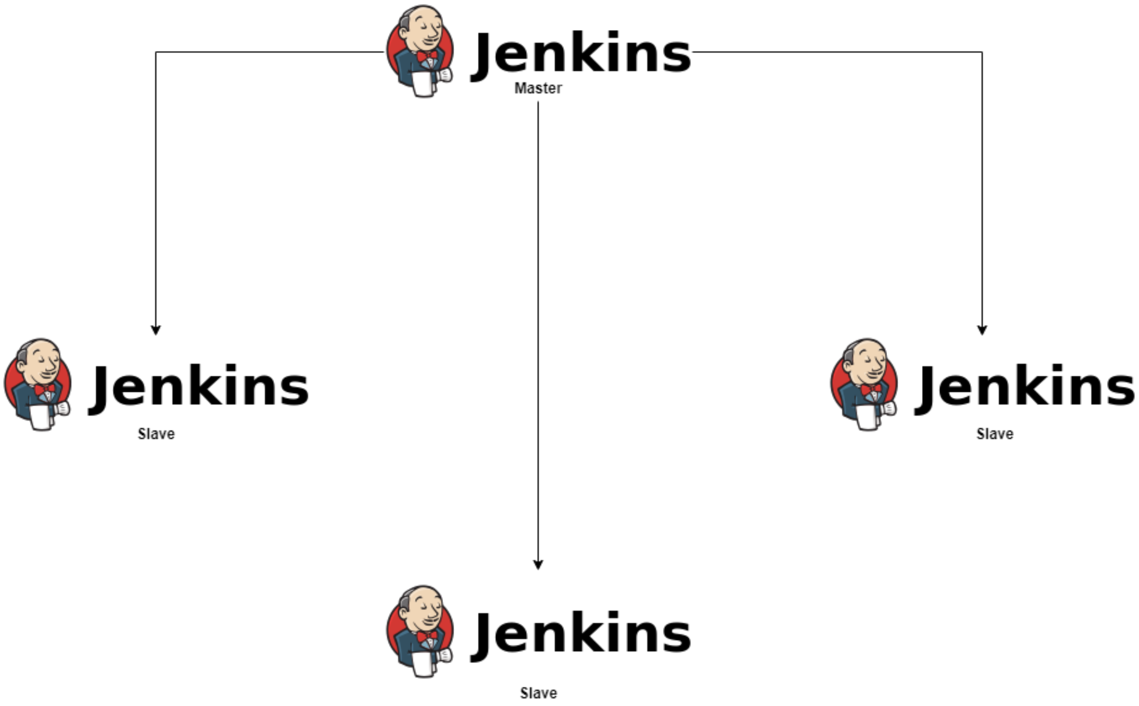
In this article we will learn how to setup Devops tool Jenkins in master/slave mode.

With the growing popularity of Microservices, it is becoming necessary for any company to develop and launch several services at the same time. It's at this point that the CI tools must provide a way for sharing the load across several machines/servers.

Jenkins also has the capability of distributing build jobs across a large number of systems, which is known as Jenkins distributed builds. We can set up a farm of build servers using Jenkins to distribute the burden or perform the build jobs in different settings.

When we have a large project to build and the load has to be distributed across several machines on the network, distributed builds improve the efficiency of the build process. Jenkins implements the Master/Slave architecture to manage distributed builds.

We may associate a project with the slave node and then schedule it for the build when we've created and configured it.



**Jenkins Master** is the primary Jenkins server and is responsible for the following tasks:

- It distributes the builds among the numerous slaves for execution.
- It organises the build projects.
- It keeps an eye on the slaves at all times.
- Master can also run build jobs directly if necessary.
- It keeps track of the build outcomes and shows them.

**Jenkins Slave** runs on a remote machine. A slave is responsible for the following tasks:

- Slaves can be operated on a number of different operating systems.

- It responds to the Jenkins Master's demands.
- Apart from the fact that Jenkins executes the build task on the next available save,
- we can always arrange the project to run on a certain sort of slave computer.
- It also completes construction operations that the Master has dispatched.

**Steps -**

1. Install Java on master node
2. Install Jenkins on master node
3. Install java on slave node
4. Create a user and ssh keys on slave node
5. Copy keys on master node
6. Join slave node to master
7. Test the setup

**Procedure -**

**1. Install java on master node**

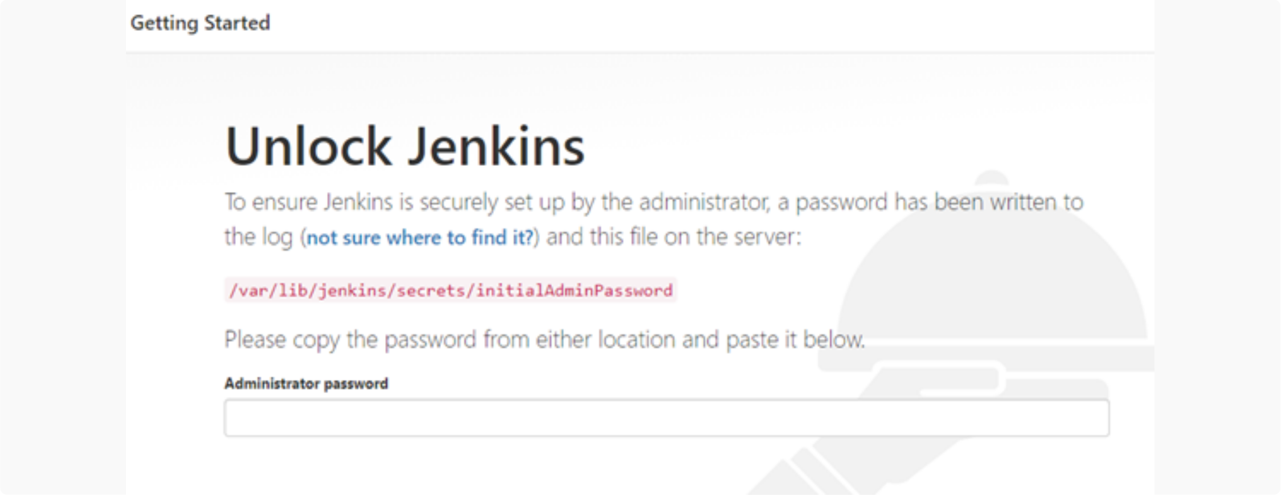
```
[root@jenkins-master ~]# dnf install java-11-openjdk -y
```

**2. Install jenkins on master node**

```
[root@jenkins-master ~]# dnf install wget -y

[root@jenkins-master ~]# wget -O /etc/yum.repos.d/jenkins.repo  https://pkg.jenkins.io/redhat-stable/jenkins.repo

[root@jenkins-master ~]# rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

[root@jenkins-master ~]# dnf install jenkins -y

[root@jenkins-master ~]# systemctl status jenkins

[root@jenkins-master ~]# systemctl enable --now  jenkins

[root@jenkins-master ~]# systemctl status jenkins

[root@jenkins-master ~]# firewall-cmd --permanent --add-port=8080/tcp

[root@jenkins-master ~]# firewall-cmd --reload
```

## Access URL http://192.168.0.103:8080

We will be asked to enter the password, like below -



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

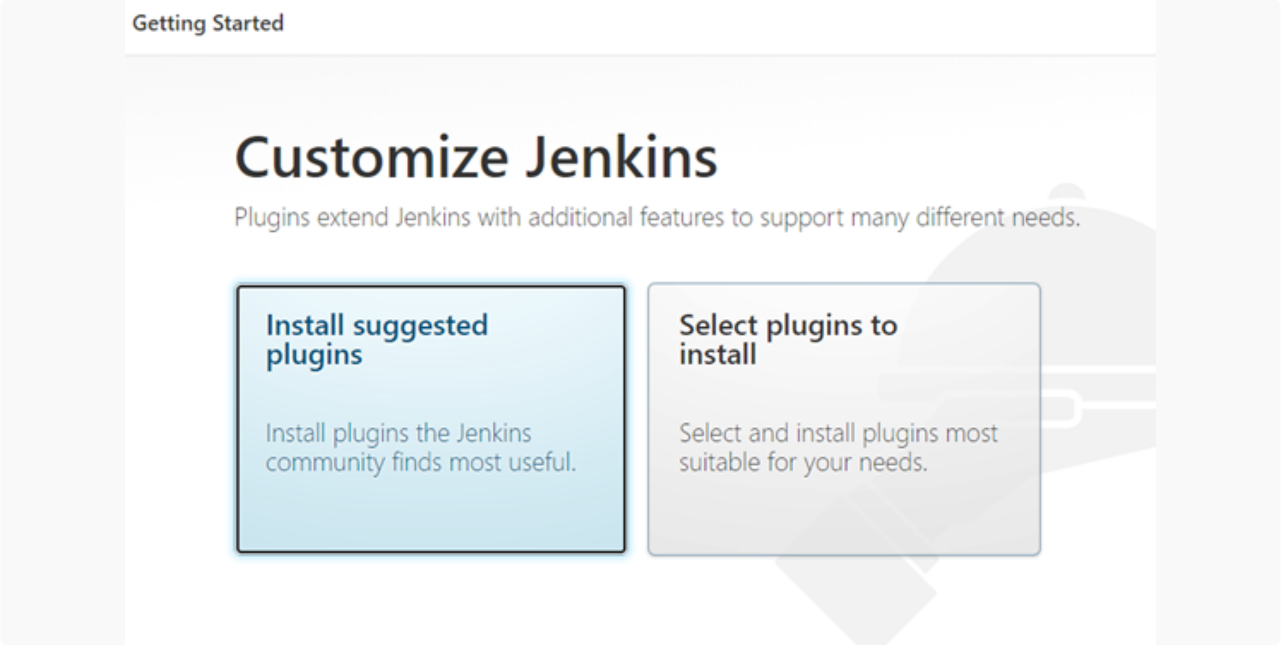
When prompted get the password from below location and enter it -

[root@jenkins-master ~]# cat /var/lib/jenkins/secrets/initialAdminPassword

8913aa206f044edb83ca0ddf52cbe085

Install the select the suggested plugins

On next screen you can see its asking to install the suggested plugins -

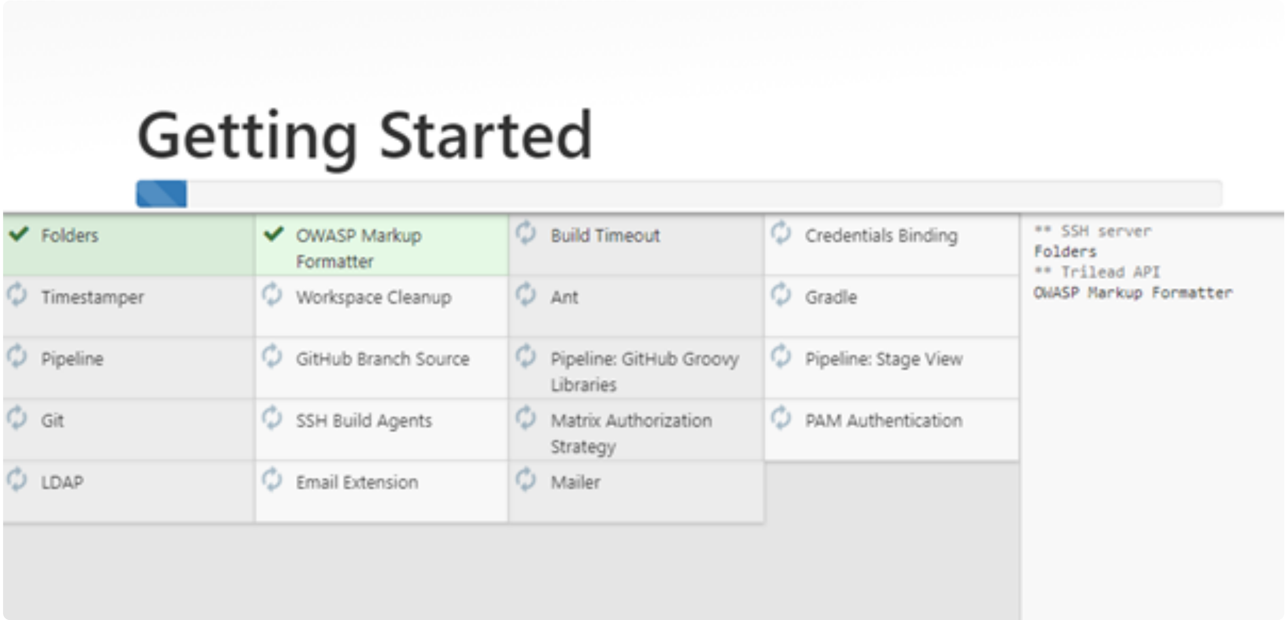


Getting Started

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.

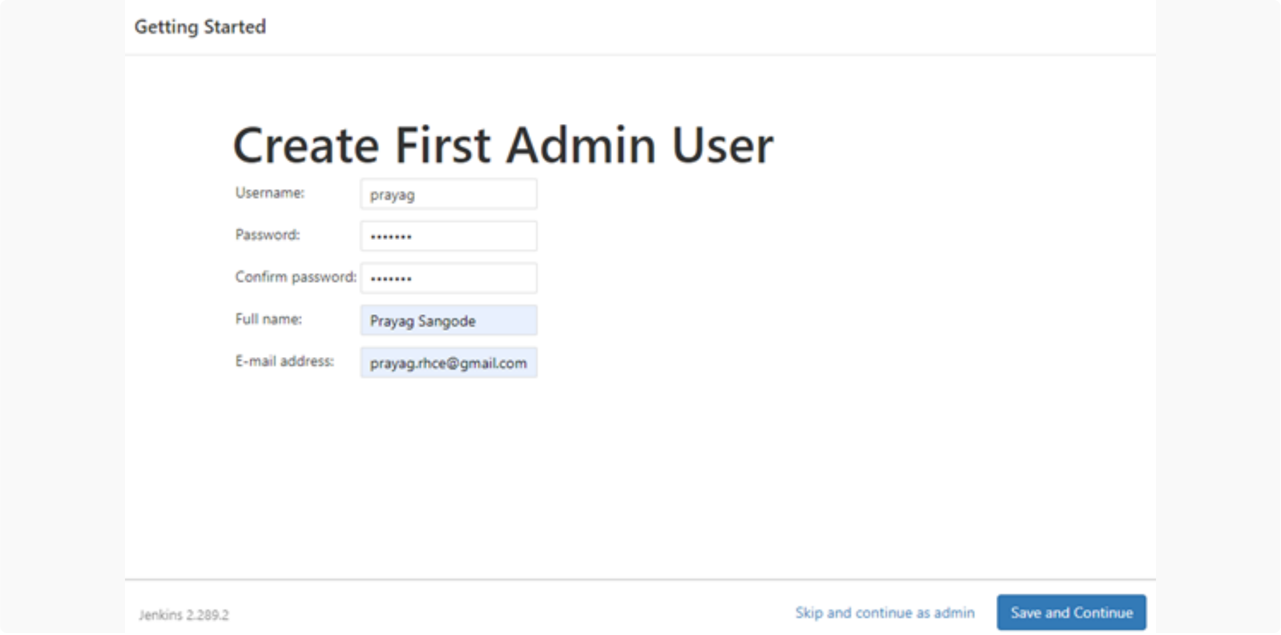


## Getting Started

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

```
** SSH server
Folders
** Trilead API
OWASP Markup Formatter
```

On next screen you will be prompted to create Jenkins user -

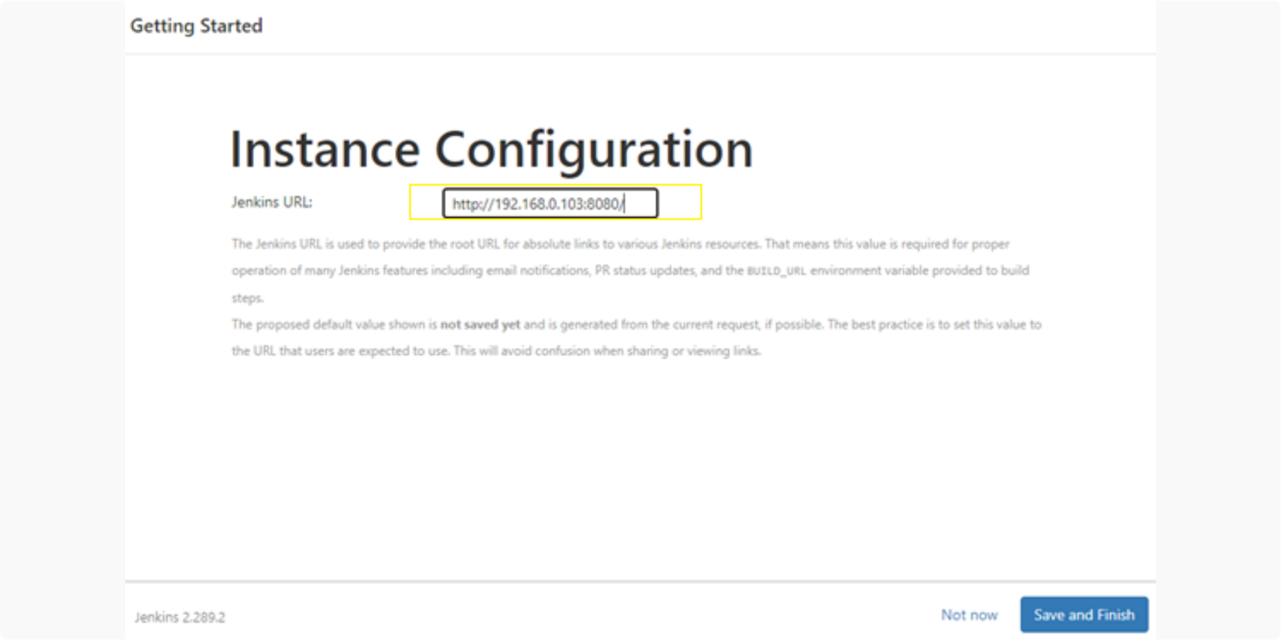


Getting Started

## Create First Admin User

| | |
|---|---|
| Username: | prayag |
| Password: | ••••••• |
| Confirm password: | ••••••• |
| Full name: | Prayag Sangode |
| E-mail address: | prayag.rhce@gmail.com |

Jenkins 2.289.2



Getting Started

Instance Configuration

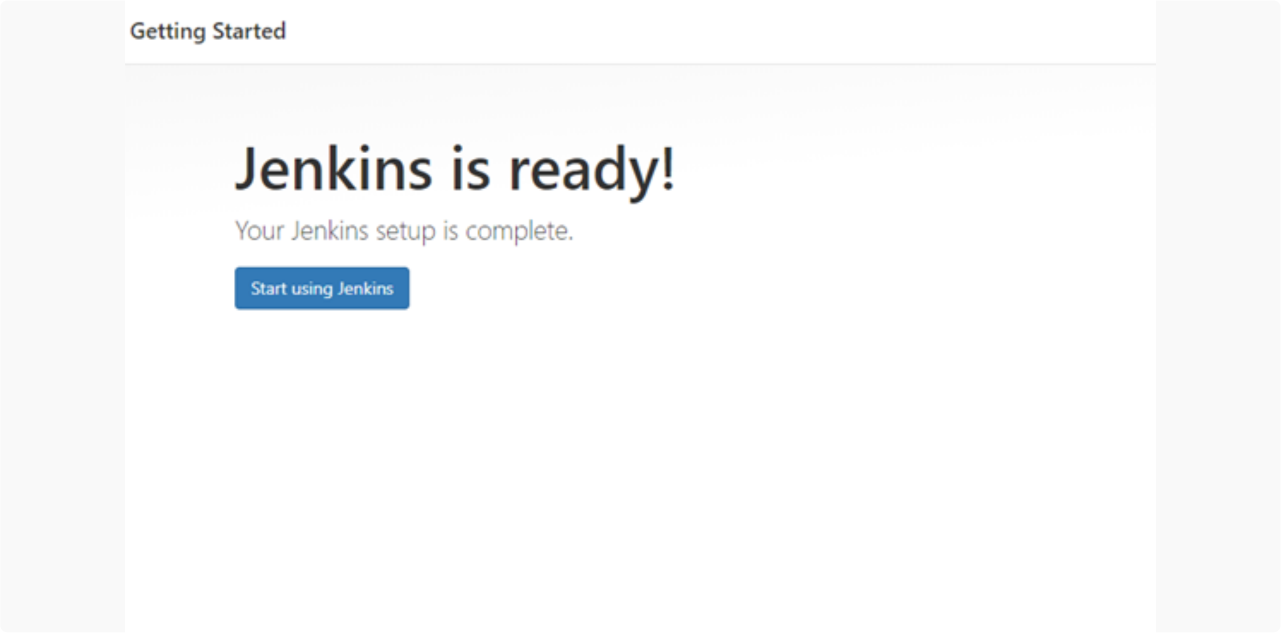
Jenkins URL: http://192.168.0.103:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.289.2

Installation is completed

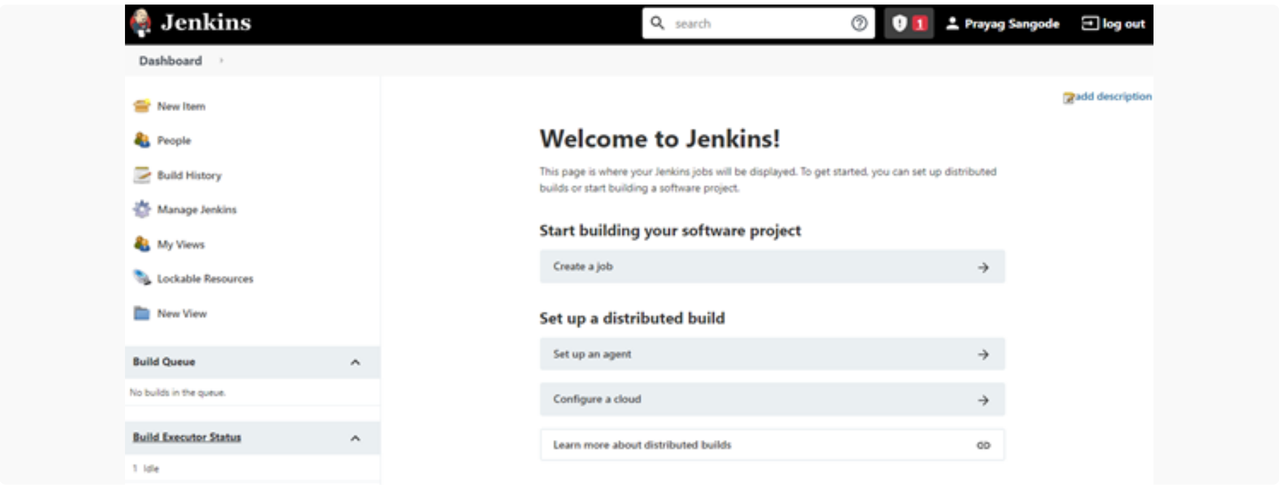


Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

We get default Jenkins Dashboard



No builds in the queue.

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

### 3. Install java on slave node

```
[root@jenkins-slave1 ~]# dnf install java-11-openjdk -y
```

### 4. Create a user and ssh keys on slave node

```
[root@jenkins-slave1 ~]# useradd prayag
```

Generate ssh-key for user prayag and note down private key - to be added in Jenkins credentials -

```
[prayag@jenkins-slave ~]$ ssh-keygen -t rsa
```

```
[prayag@jenkins-slave ~]$ cd .ssh/
```

```
[prayag@jenkins-slave ~]$ cat id_rsa.pub > authorized_keys
```

```
[prayag@jenkins-slave ~]$ chmod 700 authorized_keys
```

## [prayag@jenkins-slave ~]$ cat authorized_keys

### 5. Copy keys on master node

On master - copy public key in /var/lib/jenkins/.ssh/known_hosts - used when selecting option Know hosts key strategy

```
[root@jenkins-master ~]# mkdir /var/lib/jenkins/.ss
```

```
[root@jenkins-master ~]# chown jenkins:jenkins /var/lib/jenkins/.ssh
```

```
[root@jenkins-master ~]# echo "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQC2djObC0ahzBSVDKy82N4G/
+orCAqwspGcPBZne0HL1f9S3ZJxZm+UJtCbbL0MjeXmJOJ3y1qRxn5f2e9+RLXADT7RDiGU1DOp3ZEqjET
PHHZrM3fJEeu/kada7ivUoj3o82Uid7cBeAUtEUJ2L4uWPjdskjtOVi6AIpe/
XTHr+uQ6zprqZovpeRJjJNeDyT4hXzH0587kCPyYxVSGkz5n2+/c2WV8QTrSiEb9fENiOE231AoRc38uN1f/
eoernJ6lqO8PFLoOTp0qhcmOLa3Op1OJqikNa09yIfLZ1/
CzAjJC8p19YiWei50ypmm0zcNrmxdtguOU1hKS0ie0UGLXpMhbsBI2YA6dXosTqCzQzB9PmOa+nLrwu6iFE
34PN5UsvnmqQvhAWVsNLFzg+Uqok8fSd6pqxgvgNmYU4TSQJa+/
K09DCj+xdAszOvWyuGRs6D3oSogvH0cjPa5qqXoMSecK/81NU44i37Mj6/cY5Q2AHj/
LFKJEbvST9dSoPmU= prayag@jenkins-slave.example.com" > /var/lib/jenkins/.ssh/known_hosts
```

```
[root@jenkins-master ~]# cat /var/lib/jenkins/.ssh/known_hosts
```

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC2djObC0ahzBSVDKy82N4G/
+orCAqwspGcPBZne0HL1f9S3ZJxZm+UJtCbbL0MjeXmJOJ3y1qRxn5f2e9+RLXADT7RDiGU1DOp3ZEqjET
PHHZrM3fJEeu/kada7ivUoj3o82Uid7cBeAUtEUJ2L4uWPjdskjtOVi6AIpe/
XTHr+uQ6zprqZovpeRJjJNeDyT4hXzH0587kCPyYxVSGkz5n2+/c2WV8QTrSiEb9fENiOE231AoRc38uN1f/
eoernJ6lqO8PFLoOTp0qhcmOLa3Op1OJqikNa09yIfLZ1/
CzAjJC8p19YiWei50ypmm0zcNrmxdtguOU1hKS0ie0UGLXpMhbsBI2YA6dXosTqCzQzB9PmOa+nLrwu6iFE
34PN5UsvnmqQvhAWVsNLFzg+Uqok8fSd6pqxgvgNmYU4TSQJa+/
K09DCj+xdAszOvWyuGRs6D3oSogvH0cjPa5qqXoMSecK/81NU44i37Mj6/cY5Q2AHj/
LFKJEbvST9dSoPmU= prayag@jenkins-slave.example.com
```

```
[root@jenkins-master ~]# ssh-keyscan -H jenkins-slave.example.com >>/var/lib/jenkins/.ssh/known_hostsh
```

### 6. Join slave node to master

**To join the Jenkins slave node to Jenkins Master, perform below steps -**

Select Build Executor Status > New Node > Type - Permanent

Select below values -

Name - jenkins-slave1

Description - jenkins-slave1

Number of executors - 1

Remote root directory - /home/prayag

Labels - jenkins-slave1

Usage - Use this mode as much as possible

Launch method - Launch agents via SSH

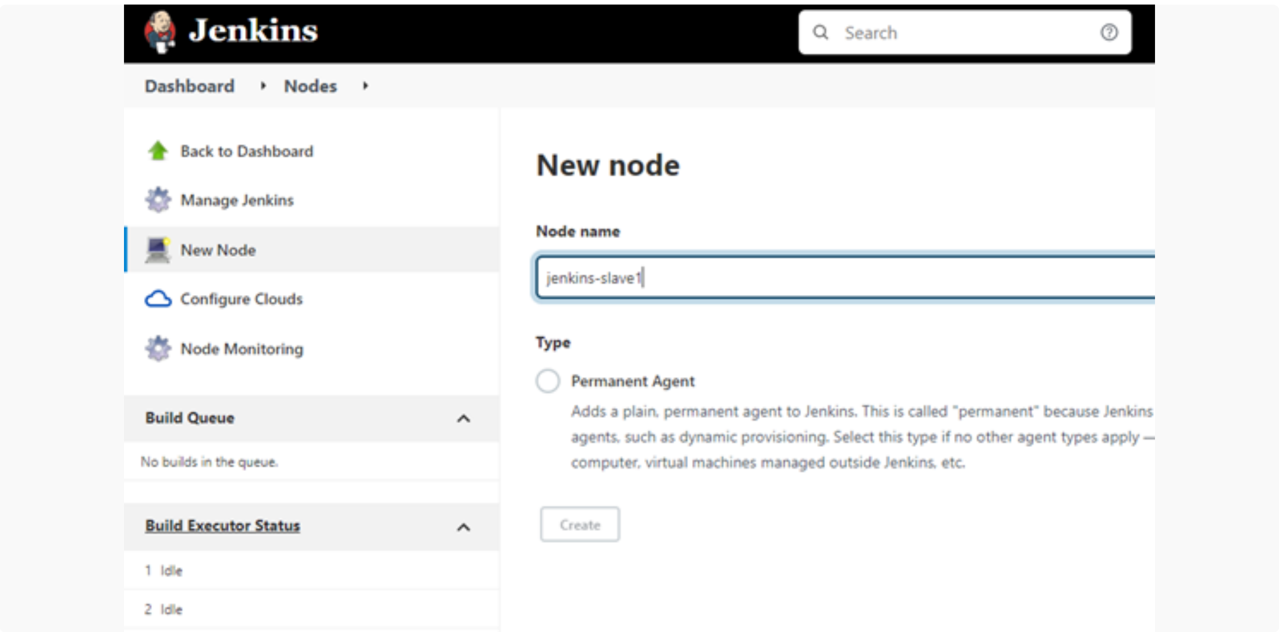
Host - 192.168.0.103

Credentials - use ssh username / private key options

Host Key Verification Strategy - Know hosts key strategy

Save and check that new slave node is added and is in sync

When prompted give node name



## New node

Node name: jenkins-slave1

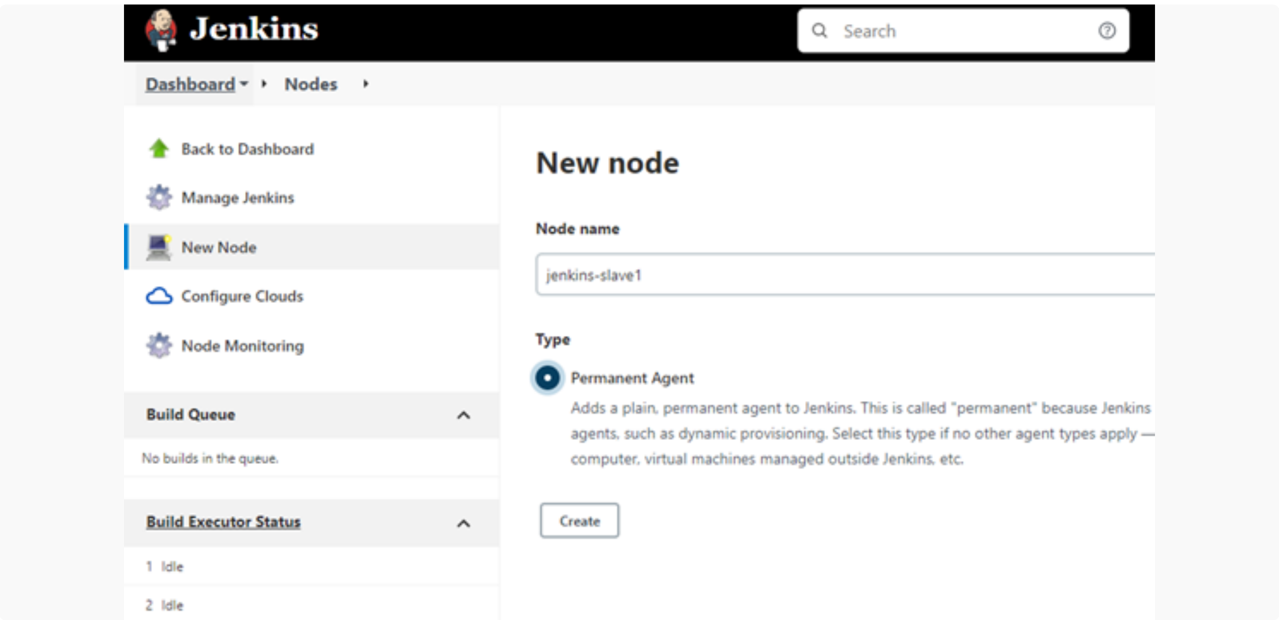
Type

○ Permanent Agent
Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins agents, such as dynamic provisioning. Select this type if no other agent types apply — computer, virtual machines managed outside Jenkins, etc.

No builds in the queue.

1 Idle
2 Idle

Select Permanent Agent



## New node

Node name: jenkins-slave1

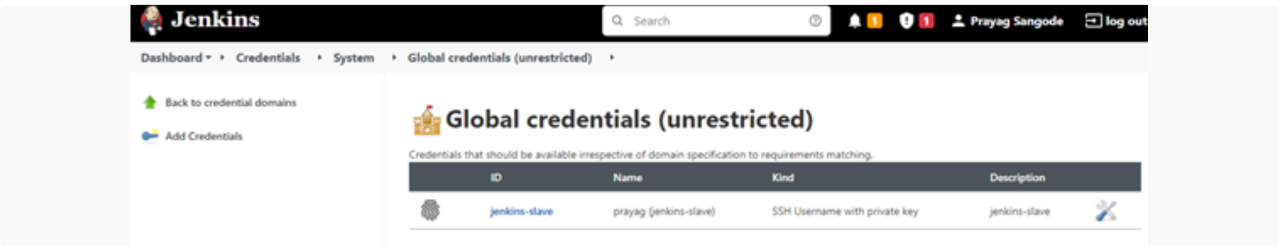
Type

● Permanent Agent
Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins agents, such as dynamic provisioning. Select this type if no other agent types apply — computer, virtual machines managed outside Jenkins, etc.

No builds in the queue.

1 Idle
2 Idle

Create ssh credentials with Private key -

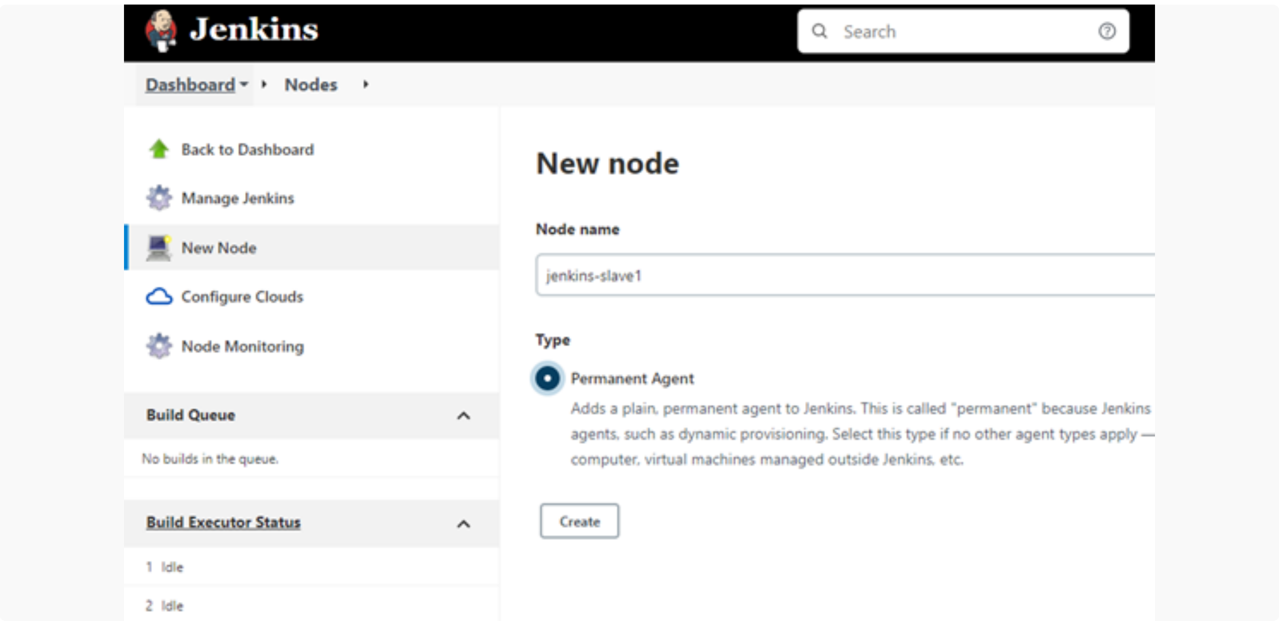


## Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

| ID | Name | Kind | Description |
|---|---|---|---|
| jenkins-slave | prayag (jenkins-slave) | SSH Username with private key | jenkins-slave |

Enter node name and select Permanent Agent



No builds in the queue.

1 Idle

2 Idle

## New node

Node name

jenkins-slave1

Type

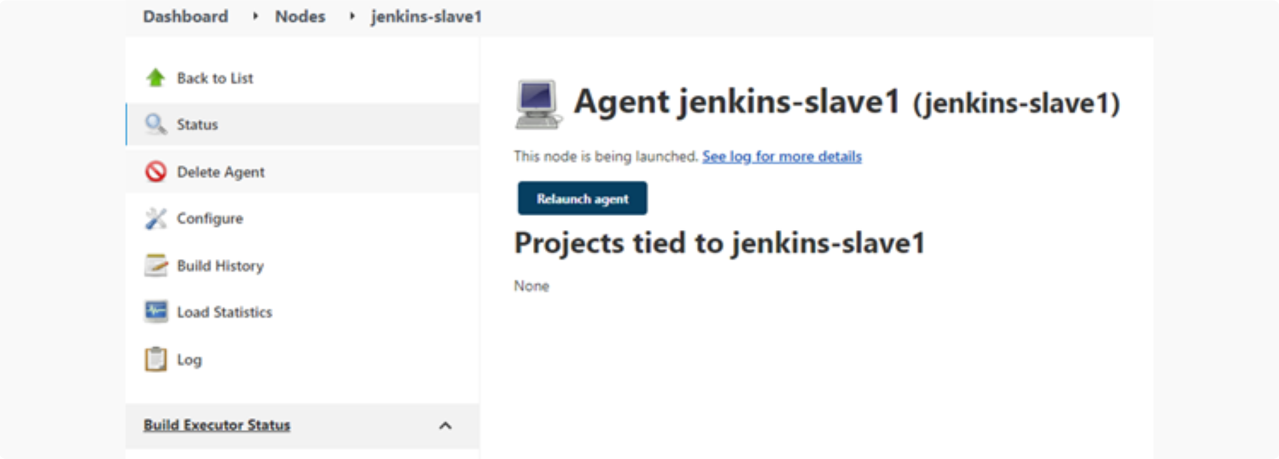
● Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins agents, such as dynamic provisioning. Select this type if no other agent types apply — computer, virtual machines managed outside Jenkins, etc.

Create

We can see Jenkins Slave is launching ..

Agent status - "is being launched"



## Agent jenkins-slave1 (jenkins-slave1)

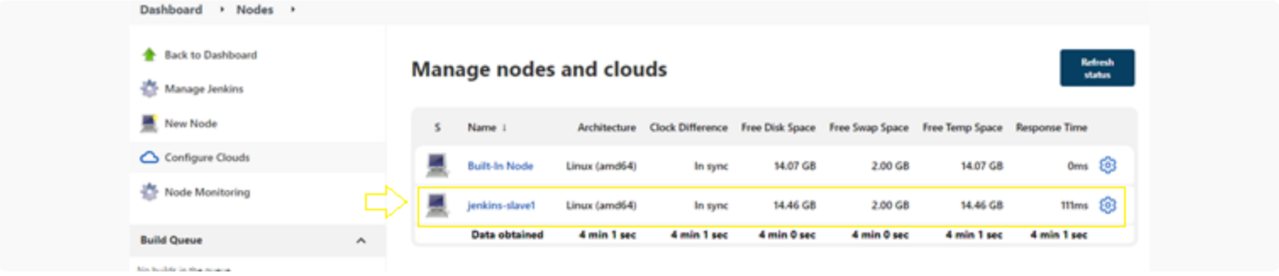
This node is being launched. See log for more details

Relaunch agent

## Projects tied to jenkins-slave1

None

Jenkins Slave node is now is "In Sync"



## Manage nodes and clouds

| S | Name ↓ | Architecture | Clock Difference | Free Disk Space | Free Swap Space | Free Temp Space | Response Time |
|---|---|---|---|---|---|---|---|
| | Built-In Node | Linux (amd64) | In sync | 14.07 GB | 2.00 GB | 14.07 GB | 0ms |
| | jenkins-slave1 | Linux (amd64) | In sync | 14.46 GB | 2.00 GB | 14.46 GB | 111ms |
| | Data obtained | 4 min 1 sec | 4 min 1 sec | 4 min 0 sec | 4 min 0 sec | 4 min 1 sec | 4 min 1 sec |

### 7. Test Jenkins Jobs -

- Create "new item"
- Enter an item name – Job1

Chose Freestyle project

- Under General Section

Choose Restrict where this project can be run

Update your jenkins slave label jenkins-slave1

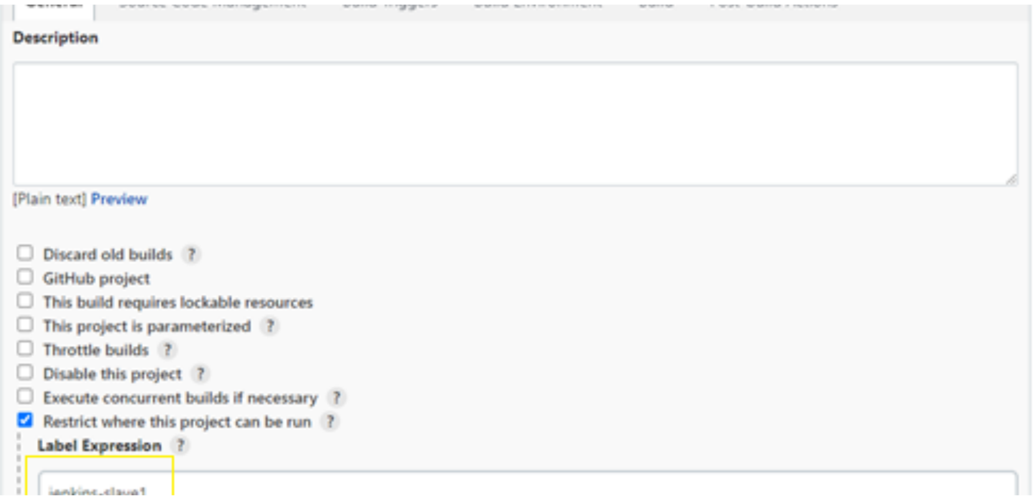
- Under Build section Execute shell

#!/bin/bash

echo "Testing Jenkins Master Slave Setup"

- Save your job
- Build job
- Check "console output"

Create new item and in General tab - **select "Restrict where this project can be run"**



Description

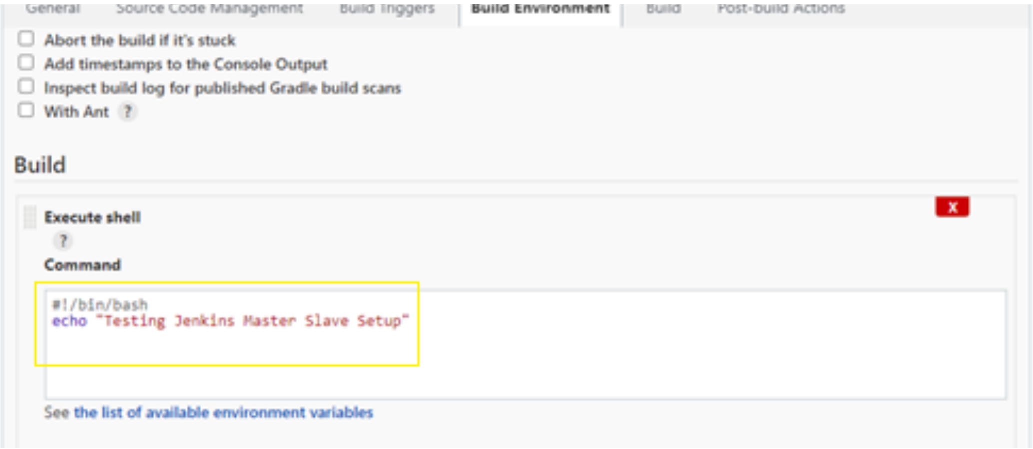
[Plain text] Preview

□ Discard old builds
□ GitHub project
□ This build requires lockable resources
□ This project is parameterized
□ Throttle builds
□ Disable this project
□ Execute concurrent builds if necessary
☑ Restrict where this project can be run

Label Expression

jenkins-slave1

Select Build as "Execute shell" and run echo command - "Testing Jenkins Master Slave Setup"



□ Abort the build if it's stuck
□ Add timestamps to the Console Output
□ Inspect build log for published Gradle build scans
□ With Ant

Build

Execute shell

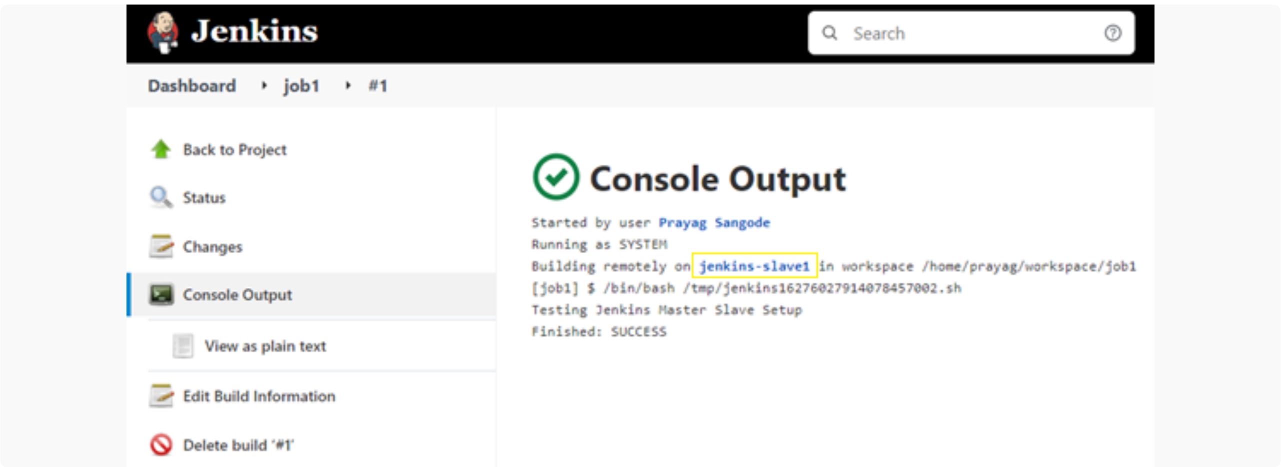
Command

```
#!/bin/bash
echo "Testing Jenkins Master Slave Setup"
```

See the list of available environment variables

Save and run build - "Build Now"

Once the build is completed we can check the console output - "Building remotely jenkins-slave1" in workspace....



Console Output

```
Started by user Prayag Sangode
Running as SYSTEM
Building remotely on jenkins-slave1 in workspace /home/prayag/workspace/job1
[job1] $ /bin/bash /tmp/jenkins16276027914078457002.sh
Testing Jenkins Master Slave Setup
Finished: SUCCESS
```

I hope you found this article to be useful in some way. I'll be back with some more interesting new articles on DEVOPS tools.