

Jenkins Master - Slave Configuration

So far we have learned how to create Jenkins freestyle Job, Integration of Jenkins with maven and git, also learned auto trigger (Poll SCM & Build periodically) and GitHub webhooks along with sequential and parallel Jobs. But everything was done on Jenkins master only (git checkout, build and deploy was done on single machine), so now onwards we will learn how to connect other ec2 machines with jenkins as a Jenkins slave and build jobs on slaves-

TCP Connection- over TCP port

Two methods to achieve Jenkins slave configuration:

1. **SSH** - Port 22
2. **JNLP** - Java Network Launch Protocol - with any random TCP port (0 - 65535)

SLAVE / NODE / AGENT / LABEL - all are same in jenkins

Dashboard → Manage Jenkins → Nodes

You can see all your nodes here,
Jenkins master called as- [Build-In Node](#)

Let's launch and prepare a slave machine: (slave-1)

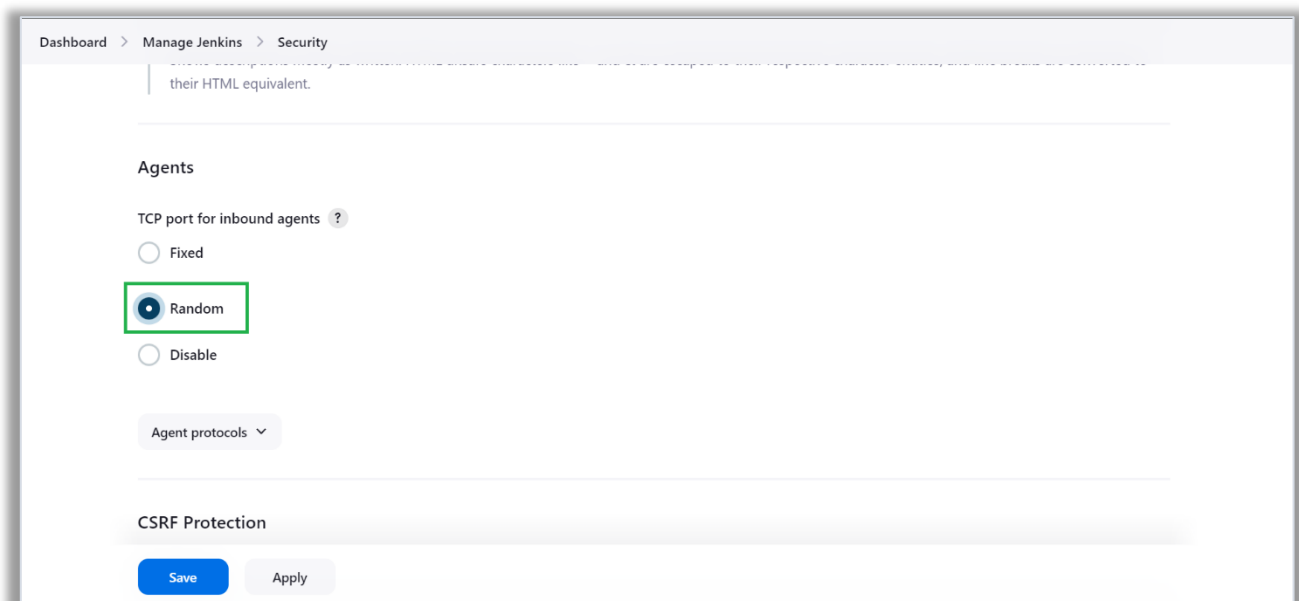
Launch an Ec2 instance → name (slave-1) → login to machine →
\$ sudo su -
amazon-linux-extras install java-openjdk11 -y
(install java on slave machine *must have java on slave machine)

Create one directory as a slave's remote root directory where we will download our agent.jar file later in this practical.

```
cd /mnt  
mkdir jenkins-slave  
chmod -R 777 /mnt/jenkins-slave
```

JNLP Method:

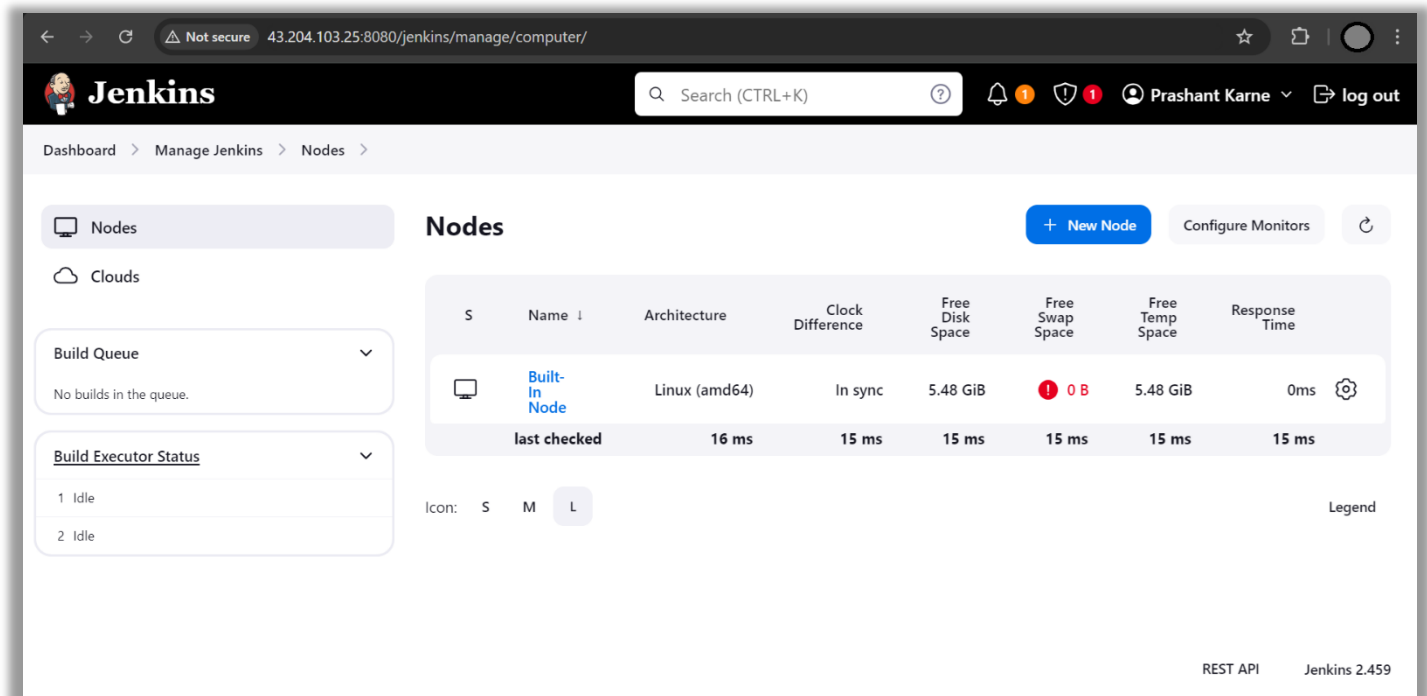
Go to Manage Jenkins → Security → Agents → select TCP port for inbound agents as **Random** (we will make it **Fixed** later)



→ **Apply + Save**

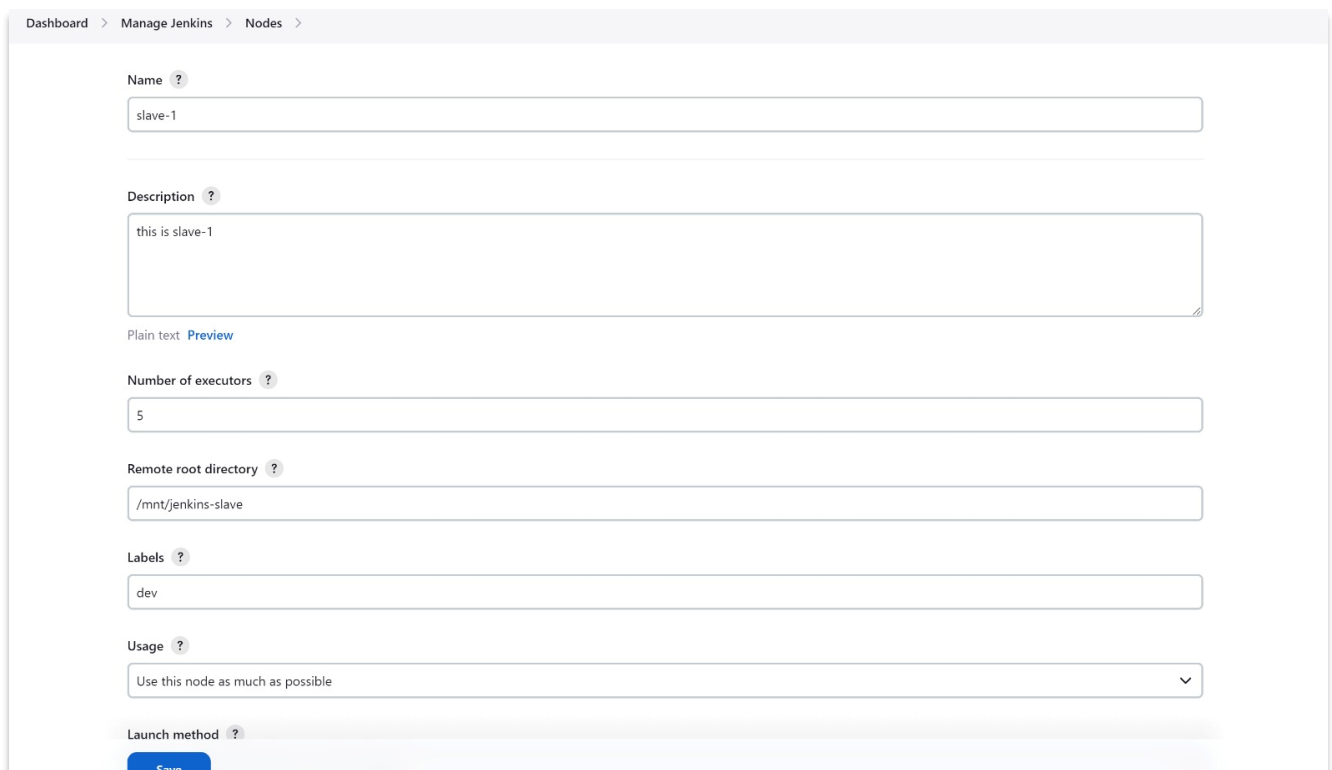
We will create a slave now

Go to - Dashboard → Manage Jenkins → Nodes →



The screenshot shows the Jenkins 'Nodes' page. The left sidebar contains a 'Nodes' tab, a 'Build Queue' section with 'No builds in the queue.', and a 'Build Executor Status' section showing '1 Idle' and '2 Idle' executors. The main area is titled 'Nodes' and features a '+ New Node' button, a 'Configure Monitors' button, and a refresh icon. Below these is a table with columns: S, Name, Architecture, Clock Difference, Free Disk Space, Free Swap Space, Free Temp Space, and Response Time. The table lists one node, 'Built-In Node', with architecture 'Linux (amd64)', 'In sync' clock difference, and 5.48 GiB of free disk, swap, and temp space. Below the table is an 'Icon' section with 'S', 'M', and 'L' options, and a 'Legend' link. The bottom right corner shows 'REST API' and 'Jenkins 2.459'.

Click on New node → add node name (slave-1) → select Type Permanent Agent → click Create → add description → Number of executors (eg: 5) → add Remote root directory (Eg: /mnt/jenkins-slave ; this directory has to be present on machine Jenkins will not create for us) → add label (must have, eg: dev) → Launch method → Launch agent by connecting it to the connector (this is JNLP) → click on **save**



The screenshot shows the 'New Node' form in Jenkins. The form fields are: 'Name' (slave-1), 'Description' (this is slave-1), 'Number of executors' (5), 'Remote root directory' (/mnt/jenkins-slave), 'Labels' (dev), 'Usage' (Use this node as much as possible), and 'Launch method' (Launch agent by connecting it to the connector). A 'Save' button is at the bottom.

Dashboard > Manage Jenkins > Nodes >

Usage ?
 Use this node as much as possible

Launch method ?
 Launch agent by connecting it to the controller

☐ Disable WorkDir ?

Custom WorkDir path ?

Internal data directory ?

☐ Fail if workspace is missing ?




☐ Use WebSocket ?

Advanced ▾

Availability ?
 Keep this agent online as much as possible

Save

You can see new slave-1 is created but it's disconnected,

Nodes								+ New Node		Configure Monitors	↻
S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time				
	Built-In Node	Linux (amd64)	In sync	5.48 GiB	 0 B	5.48 GiB	0ms				
	slave-1		N/A	N/A	N/A	N/A	N/A				
last checked		5 min 46 sec	5 min 46 sec	5 min 46 sec	5 min 46 sec	5 min 46 sec	5 min 46 sec				

now open slave-1 node you will see a few commands.

We will connect our slave to master with the help of this commands:

Run from agent command line: (Unix)

```
curl -sO http://43.204.103.25:8080/jenkins/jnlpJars/agent.jar
java -jar agent.jar -url http://43.204.103.25:8080/jenkins/ -secret
ce68df0e569a658e46a0185b09d6f5bb7b415de748fda8310013c366668366d1 -name "slave-1" -workDir "/mnt/jenkins-slave"
```

Agent slave-1

Run from agent command line: (Unix)

```
curl -sO http://43.204.103.25:8080/jenkins/jnlpJars/agent.jar
java -jar agent.jar -url http://43.204.103.25:8080/jenkins/ -secret
ce68df0e569a658e46a0185b09d6f5bb7b415de748fda8310013c366668366d1 -name "slave-1" -workDir "/mnt/jenkins-slave"
```

Run from agent command line: (Windows)

```
curl.exe -sO http://43.204.103.25:8080/jenkins/jnlpJars/agent.jar
java -jar agent.jar -url http://43.204.103.25:8080/jenkins/ -secret
ce68df0e569a658e46a0185b09d6f5bb7b415de748fda8310013c366668366d1 -name "slave-1" -workDir "/mnt/jenkins-slave"
```

(Note - also make sure if Jenkins URL is updated by going to - Dashboard – Manage Jenkins – System – Jenkins URL)

Now download agent.jar on slave machine, inside slave's Remote root directory (Eg: /mnt/jenkins-slave) using curl command given by jenkins for node connection.

Again, run the second command starting with - java -jar from the location where you have downloaded the agent.jar file.

It will try to connect but will give an error, in the logs we can see why it failed to connect over a specific port number. The reason is that the port number is not open in our ec2 instance's Security Group from where it will attempt to connect with our Jenkins Master. we can also add this port number in Jenkins security as well to make it fixed.

```
INFO: Both error and output logs will be printed to /mnt/jenkins-slave/remoting
Oct 11, 2024 6:40:04 PM hudson.remoting.Launcher createEngine
INFO: Setting up agent: slave-1
Oct 11, 2024 6:40:04 PM hudson.remoting.Engine startEngine
INFO: Using Remoting version: 3248.v65ecb_254c298
Oct 11, 2024 6:40:04 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /mnt/jenkins-slave/remoting as a remoting work directory
Oct 11, 2024 6:40:05 PM hudson.remoting.Launcher$CuiListener status
INFO: Locating server among [http://43.204.103.25:8080/jenkins/]
Oct 11, 2024 6:40:05 PM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver resolve
INFO: Remoting server accepts the following protocols: [JNLP4-connect, Ping]
Oct 11, 2024 6:40:10 PM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver isPortVisible
WARNING: connect timed out
Oct 11, 2024 6:40:10 PM hudson.remoting.Launcher$CuiListener status
INFO: Could not locate server among [http://43.204.103.25:8080/jenkins/]; waiting 10 seconds before retry
java.io.IOException: http://43.204.103.25:8080/jenkins/ provided port:38757 is not reachable on host 43.204.103.25
    at org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver.resolve(JnlpAgentEndpointResolver.java:311)
    at hudson.remoting.Engine.innerRun(Engine.java:829)
    at hudson.remoting.Engine.run(Engine.java:574)
Oct 11, 2024 6:40:20 PM hudson.remoting.Launcher$CuiListener status
INFO: Locating server among [http://43.204.103.25:8080/jenkins/]
Oct 11, 2024 6:40:20 PM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver resolve
INFO: Remoting server accepts the following protocols: [JNLP4-connect, Ping]
Oct 11, 2024 6:40:25 PM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver isPortVisible
WARNING: connect timed out
Oct 11, 2024 6:40:25 PM hudson.remoting.Launcher$CuiListener status
INFO: Could not locate server among [http://43.204.103.25:8080/jenkins/]; waiting 10 seconds before retry
java.io.IOException: http://43.204.103.25:8080/jenkins/ provided port:38757 is not reachable on host 43.204.103.25
    at org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver.resolve(JnlpAgentEndpointResolver.java:311)
    at hudson.remoting.Engine.innerRun(Engine.java:829)
    at hudson.remoting.Engine.run(Engine.java:574)
```

For that go to, Dashboard → Manage Jenkins → Security → Inside Agents → tcp port for inbound → select Fixed and add port number here.

Dashboard > Manage Jenkins > Security

Shows descriptions mostly as written. HTML unsafe characters like < and & are escaped to their respective character entities, and line breaks are converted to their HTML equivalent.

Agents

TCP port for inbound agents ?

☒ Fixed

☐ Random

☐ Disable

38757

Agent protocols ▾

Save Apply

We need to open the same port on the machine's security group's inbound rule, after then run the same command again and we can see that our slave will now connect to our Master.

```

Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Handshaking
Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Connecting to 43.204.103.25:38757
Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Server reports protocol JNLP4-connect-proxy not supported, skipping
Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Trying protocol: JNLP4-connect
Oct 11, 2024 6:45:58 PM org.jenkinsci.remoting.protocol.impl.BIONetworkLayer$Reader run
INFO: Waiting for ProtocolStack to start.
Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Remote identity confirmed: d8:bf:1d:ea:87:7e:27:93:c1:32:0e:69:39:d9:6a
Oct 11, 2024 6:45:58 PM hudson.remoting.Launcher$CuiListener status
INFO: Connected

```

Check by going to → Dashboard → Manage Jenkins → Nodes → slave-1

NOTE: If we press CTRL+C on terminal it will lose the connection with master, hence to run the command in background

Use same command starting with 'nohup' and ending with '&'

```
$ nohup <Connection-command> &
```

Run the command now and press enter key 2 - 3 times. our command is running in the background.

ALL COMMANDS HISTORY FOR SLAVE-1

```

# amazon-linux-extras install java-openjdk11
# cd /mnt
# mkdir jenkins-slave
# cd jenkins-slave/
# curl -sO http://43.205.178.42:8080/jenkins/jnlpJars/agent.jar
# nohup java -jar agent.jar -jnlpUrl
http://43.205.178.42:8080/jenkins/computer/slave%2D2/jenkins-agent.jnlp
-secret

```

```
07fea906afed6aaca3d88a5338c67b6d50b9bc31eb40e4d13908592aaa658cf8 -  
workDir "/mnt/jenkins-slave" &  
tail -100f c  
tail -100f nohup.out
```

Now, create one more machine (slave-2) by repeating above steps.

Let's create and run a Job

New Item → add name (test-slave) → Freestyle Project → OK → Add build steps → execute shell

```
Sudo yum install httpd -y  
echo "Job run on slave-1" > /var/www/html/index.html  
chmod -R 777 /var/www/html/  
systemctl start httpd
```

→ select Delete workspace before build start → Discard Old builds → **Apply + Save**

You can see we haven't specified on which slave we want to build the job so by default Jenkins will run jobs on any of the available free slaves, Job can be build on any available machine either it can be master or slave.

To run build on specific slave or node:

So, for that,

Open Job and go to configure → select restrict where this project can be run → mention Label Expression > add 'dev' (it will run job on any free machine which has label dev) → **Apply + Save**

The screenshot shows the Jenkins Configuration page for a job named 'slae-1-test-job'. The 'Configure' tab is active, and the 'General' section is selected in the left sidebar. In the 'Restrict where this project can be run' section, the checkbox is checked, and the 'Label Expression' field contains 'slave-1'. Below the field, a message states: 'Label slave-1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.' The 'Advanced' dropdown is visible. At the bottom, there are 'Save' and 'Apply' buttons.

Let's say if we want to run job on particular slave - just write name of the slave instead of label (Eg: slave-1) next time when you build a job it will execute on that particular machine.