

Install Jenkins slaves in the cloud and form a Jenkins cluster



For this let us start by creating two instances(redhat AMI) in AWS cloud environment . Consider One as master and another as slave machine.

Please make a note of the Public IP's of the Instances.

Install Java in master machine.

Install Jenkins in Master machine.

Start the slave agent

- Master node will start the slave agent on the slave machine via SSH .
- Automatic SSH login without password from master node to the slave node is needed.
- Master node will be running as a specific user called **Jenkins** to start the slave agent.

Using User Name and Password

- ❖ Create Ubuntu EC2 Instance ubuntu in AWS.
- ❖ Login with pem file using git bash or putty
 - Go to Downloads
 - ssh -i "Jenkins-Mast.pem" [ubuntu@ec2-52-53-156-213.us-west-1.compute.amazonaws.com](https://1.compute.amazonaws.com)
 - sudo -i

Jenkins Setup

Java Installation :

- ❖ sudo add-apt-repository ppa:openjdk-r/ppa
- ❖ sudo apt-get update
- ❖ sudo apt-get install openjdk-8-jdk

Jenkins Installation:

- ❖ sudo apt-get install wget
- ❖ wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -
- ❖ sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
- ❖ sudo apt-get update
- ❖ sudo apt-get install Jenkins
- ❖ go to browser and type IPAddress:8080 ex : 9.180.345.67:8080
- ❖ cat /var/lib/jenkins/secrets/initialAdminPassword
- ❖ Give the password to browse
- ❖ And click on Install suggested plug ins
- ❖ And configure your own user id there.
- ❖ And able to login our Jenkins successfully.

Now we just our Jenkins Master server setup and we are going to create slave now.

- ❖ Create Ubuntu EC2 Instance centos in AWS.
- ❖ Login with pem file using git bash or putty
 - Go to Downloads
 - ssh -i "Jenkins-Mast.pem" [ubuntu@ec2-52-53-156-213.us-west-1.compute.amazonaws.com](https://1.compute.amazonaws.com)
 - sudo -i

Java Installation :

- ❖ sudo yum search java
- ❖ sudo yum install **java-1.8.0-openjdk.x86_64**

- ❖ Now we have to create one folder for slave agent and slave Jenkins workspace
- ❖ `mkdir /opt/Jenkins`
- ❖ `useradd jennode`
- ❖ `passwd jennode`
- ❖ Enter password 2 times here and it says successfully user created.
- ❖ Now we have to own the created directory for created user
- ❖ `chown jennode:jennode /opt/Jenkins`

Now we have to update as password authentication as yes

- ❖ `vi /etc/ssh/sshd_config`

```
# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication no
#PermitEmptyPasswords no
PasswordAuthentication yes
```

- ❖ And restart the sshd service
- ❖ `service sshd restart`

```
[root@ip-172-31-12-197 ~]# mkdir /opt/jenkins
[root@ip-172-31-12-197 ~]# useradd jennode
[root@ip-172-31-12-197 ~]# passwd jennode
Changing password for user jennode.
New password:
BAD PASSWORD: it is based on a dictionary word
BAD PASSWORD: is too simple
Retype new password:
passwd: all authentication tokens updated successfully.
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]#
[root@ip-172-31-12-197 ~]# chown jennode /opt/jenkins
[root@ip-172-31-12-197 ~]# chown jennode:jennode /opt/jenkins
[root@ip-172-31-12-197 ~]# vi /etc/ssh/sshd_config
[root@ip-172-31-12-197 ~]# service sshd restart
Stopping sshd:
Starting sshd:
[root@ip-172-31-12-197 ~]#
```

Our required slave set up is done and now we have to create node at Jenkins master server.

- ❖ Go to Jenkins Master server browser
- ❖ Go to Manage Jenkins
- ❖ Click on Manage Nodes
- ❖ Create new node.

Using Private key and Public key setup :

Go to the Jenkins user by typing the below command.

```
$sudo -iu Jenkins /bin/bash
```

```
root@ubuntu-512mb-nyc3-01:~# sudo -iu jenkins
jenkins@ubuntu-512mb-nyc3-01:~$
```

master node

Now we need to generate the public and private key pair for authentication purpose. For this we need to type the below command in master as below.

```
$ ssh-keygen -t rsa
```

And then press enter for all the default options.

```
root@ubuntu-512mb-nyc3-01:~# sudo -iu jenkins
jenkins@ubuntu-512mb-nyc3-01:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa.
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:YfdmILCSWhrru08Hm9A02gdSzk+7WnqyRgzScYNQt4c jenkins@ubuntu-512mb-nyc3-01
The key's randomart image is:
+---[RSA 2048]-----+
|..o...|
| oo.oo |
| .+oE.o o .|
|o..o o + + o|
| = B . S . + |
| * % = o |
|o B * .|
| .+. .|
|==*==|
+---[SHA256]-----+
jenkins@ubuntu-512mb-nyc3-01:~$
```

master node

The public key is saved in the location as shown in the below picture

```

root@ubuntu-512mb-nyc3-01:~# sudo -iu jenkins
jenkins@ubuntu-512mb-nyc3-01:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa.
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:YfdmILCSWhrru08Hm9A0ZgdSzK+7WnqyRgzScYNQt4c jenkins@ubuntu-512mb-nyc3-01
The key's randomart image is:
+---[RSA 2048]-----+
|.O...|
| oo.oo|
|+oE.o.o|

```

master node

The private key is saved in the location as shown in below picture.

```

root@ubuntu-512mb-nyc3-01:~# sudo -iu jenkins
jenkins@ubuntu-512mb-nyc3-01:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa.
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:YfdmILCSWhrru08Hm9A0ZgdSzK+7WnqyRgzScYNQt4c jenkins@ubuntu-512mb-nyc3-01
The key's randomart image is:
+---[RSA 2048]-----+
|.O...|
| oo.oo|

```

master node

Now from the master node let us create .ssh folder in slave machine as shown below.

```
$ ssh root@<slave-ip> mkdir -p .ssh
```

It will ask the root password of the slave machine . enter the root password of slave machine.

```

jenkins@ubuntu-512mb-nyc3-01:~$ ssh root@45.55.229.105 mkdir -p .ssh
root@45.55.229.105's password:
jenkins@ubuntu-512mb-nyc3-01:~$ █

```

master node

Now we have to copy the public key from the master machine and to .ssh folder in slave machine.

```
$ cat .ssh/id_rsa.pub | ssh root@<slave-ip> 'cat >> .ssh/authorized-keys'
```

Press enter and give the root password of slave machine.

```

jenkins@ubuntu-512mb-nyc3-01:~$ ssh root@45.55.229.105 mkdir -p .ssh
root@45.55.229.105's password:
jenkins@ubuntu-512mb-nyc3-01:~$ cat .ssh/id_rsa.pub | ssh root@45.55.229.105 'cat >> .ssh/authorized_keys'
root@45.55.229.105's password:
jenkins@ubuntu-512mb-nyc3-01:~$ █

```

master node

Now we can switch to slave machine from master machine without root password of slave machine.

```
jenkins@ubuntu-512mb-nyc3-01:~$ ssh root@45.55.229.105 mkdir -p .ssh
root@45.55.229.105's password:
jenkins@ubuntu-512mb-nyc3-01:~$ cat .ssh/id_rsa.pub | ssh root@45.55.229.105 'cat >> .ssh/authorized_keys'
root@45.55.229.105's password:
jenkins@ubuntu-512mb-nyc3-01:~$ ssh root@45.55.229.105
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-38-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Wed Oct  5 20:42:13 2016 from 45.55.86.3
root@ubuntu-512mb-nyc3-01:~#
```

master node

Now login to slave machine and we have to download slave agent from the master where Jenkins has been installed.

In the slave machine create directory bin and change to that directory as below.

```
root@ubuntu-512mb-nyc3-01:~# mkdir bin
root@ubuntu-512mb-nyc3-01:~# cd bin/
root@ubuntu-512mb-nyc3-01:~/bin# pwd
/root/bin
root@ubuntu-512mb-nyc3-01:~/bin#
```

Slave node

Now install the wget command first in the slave by typing the below command.

\$ sudo yum install wget (since it is a redhat machine, use apt-get if it is ubuntu)

After installation of wget , download the slave agent by typing the following command.

\$wget <http://<master-ip>:8080/jnlpJars/slave.jar>

```
root@ubuntu-512mb-nyc3-01:~# mkdir bin
root@ubuntu-512mb-nyc3-01:~# cd bin/
root@ubuntu-512mb-nyc3-01:~/bin# pwd
/root/bin
root@ubuntu-512mb-nyc3-01:~/bin# wget http://45.55.86.3:8080/jnlpJars/slave.jar
--2016-10-05 20:46:54--  http://45.55.86.3:8080/jnlpJars/slave.jar
Connecting to 45.55.86.3:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 535167 (523K) [application/java-archive]
Saving to: 'slave.jar'

slave.jar                               100%[=====] 522.62K  --.-KB/s   in 0.007s

2016-10-05 20:46:54 (69.5 MB/s) - 'slave.jar' saved [535167/535167]

root@ubuntu-512mb-nyc3-01:~/bin#
```

Slave node

After the download is completed check whether the slave.jar is there or not.

```
root@ubuntu-512mb-nyc3-01:~# mkdir bin
root@ubuntu-512mb-nyc3-01:~# cd bin/
root@ubuntu-512mb-nyc3-01:~/bin# pwd
/root/bin
root@ubuntu-512mb-nyc3-01:~/bin# wget http://45.55.86.3:8080/jnlpJars/slave.jar
--2016-10-05 20:46:54-- http://45.55.86.3:8080/jnlpJars/slave.jar
Connecting to 45.55.86.3:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 535167 (523K) [application/java-archive]
Saving to: 'slave.jar'

slave.jar          100%[=====>] 522.62K  --.-KB/s   in 0.007s

2016-10-05 20:46:54 (69.5 MB/s) - 'slave.jar' saved [535167/535167]

root@ubuntu-512mb-nyc3-01:~/bin# ls
slave.jar
root@ubuntu-512mb-nyc3-01:~/bin#
```

Slave node

Now we have to install java on slave machine because the slave.jar is a java program which needs JRE.

\$sudo yum install default-jre (if it is ubuntu use apt-get instead of yum)

```
root@ubuntu-512mb-nyc3-01:~# mkdir bin
root@ubuntu-512mb-nyc3-01:~# cd bin/
root@ubuntu-512mb-nyc3-01:~/bin# pwd
/root/bin
root@ubuntu-512mb-nyc3-01:~/bin# wget http://45.55.86.3:8080/jnlpJars/slave.jar
--2016-10-05 20:46:54-- http://45.55.86.3:8080/jnlpJars/slave.jar
Connecting to 45.55.86.3:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 535167 (523K) [application/java-archive]
Saving to: 'slave.jar'

slave.jar          100%[=====>] 522.62K  --.-KB/s   in 0.007s

2016-10-05 20:46:54 (69.5 MB/s) - 'slave.jar' saved [535167/535167]

root@ubuntu-512mb-nyc3-01:~/bin# ls
slave.jar
root@ubuntu-512mb-nyc3-01:~/bin# sudo apt-get install default-jre
```

Slave node

Now we have to configure the slave machine in Jenkins.

Goto Jenkins home page.

Click on Manage Jenkins --> click on Manage nodes

Jenkins > ENABLE AUTO REFRESH

New Item

People

Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Manage Jenkins

- Configure System**
Configure global settings and paths.
- Configure Global Security**
Secure Jenkins; define who is allowed to access/use the system.
- Global Tool Configuration**
Configure tools, their locations and automatic installers.
- Reload Configuration from Disk**
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- System Information**
Displays various environmental information to assist trouble-shooting.
- System Log**
System log captures output from `java.util.logging` output related to Jenkins.
- Load Statistics**
Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**
Access/manage Jenkins from your shell, or from your script.
- Script Console**
Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- About Jenkins**
See the version and license information.

15.55.86.3-8080/computer

Now the below page will be opened.

Jenkins > Nodes > ENABLE AUTO REFRESH

Back to Dashboard

Manage Jenkins

New Node

Configure

Build Queue

No builds in the queue.

Build Executor Status

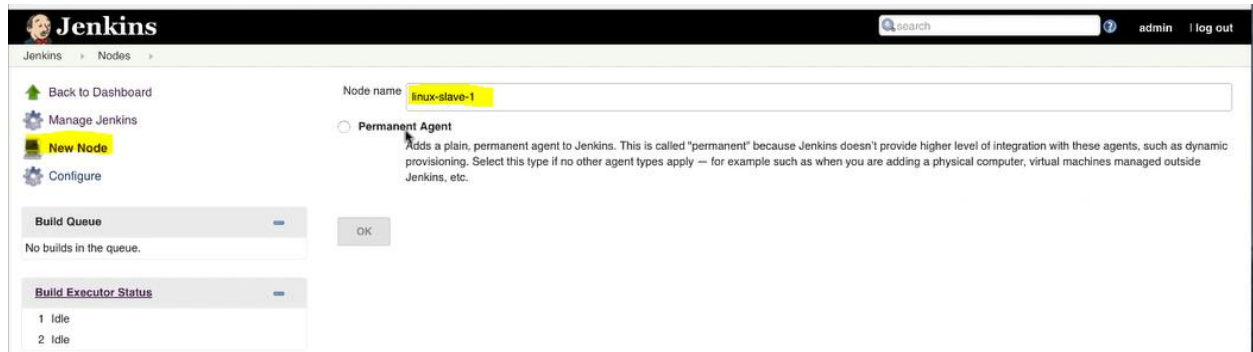
1 Idle

2 Idle

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Linux (amd64)	In sync	17.08 GB	0 B	17.08 GB	0ms
	Data obtained	35 sec	35 sec	35 sec	35 sec	35 sec	35 sec

Refresh status

Now click on New Node and give the name to the new node as shown below. Select Permanent Agent and then click on OK



The image shows the Jenkins 'New Node' configuration page. The left sidebar contains links: 'Back to Dashboard', 'Manage Jenkins', 'New Node' (highlighted), and 'Configure'. Below these are two expandable sections: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing 1 Idle and 2 Idle executors). The main form area has a 'Node name' field with 'linux-slave-1' entered. Below it is a radio button for 'Permanent Agent' with a tooltip explaining it. An 'OK' button is at the bottom right.

Jenkins

search admin log out

Jenkins > Nodes >

Back to Dashboard

Manage Jenkins

New Node

Configure

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

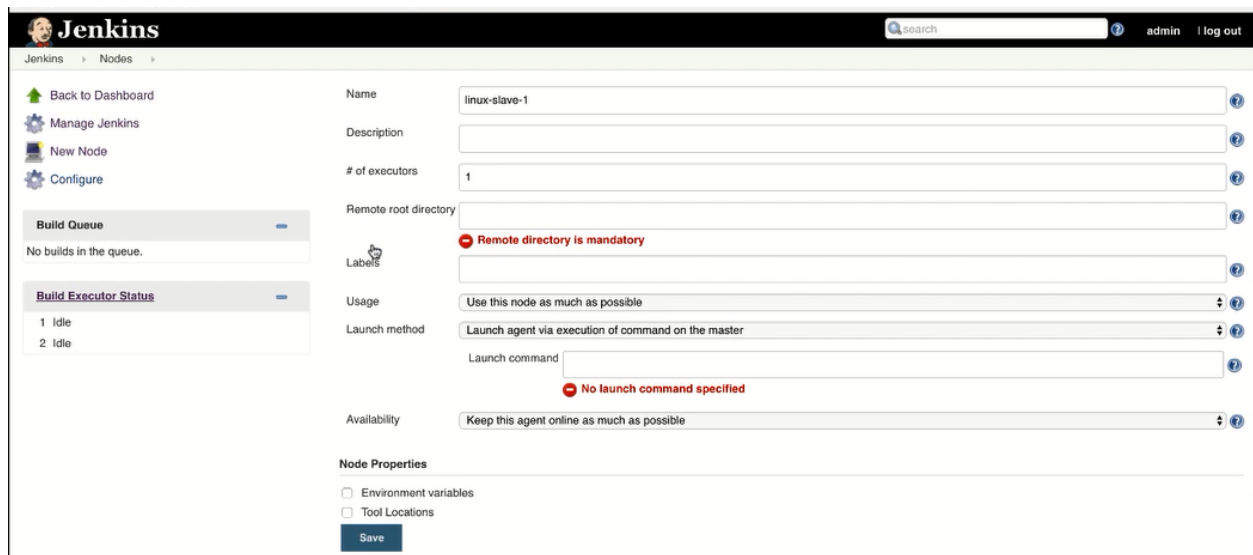
Node name linux-slave-1

☐ Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

OK

Now another page will be opened as below.



The image shows the Jenkins 'Node Configuration' page for 'linux-slave-1'. The left sidebar is identical to the previous page. The main form area contains fields for 'Name' (linux-slave-1), 'Description', '# of executors' (1), 'Remote root directory', 'Labels', 'Usage' (Use this node as much as possible), 'Launch method' (Launch agent via execution of command on the master), 'Launch command', and 'Availability' (Keep this agent online as much as possible). There are red error messages: 'Remote directory is mandatory' and 'No launch command specified'. At the bottom, there are checkboxes for 'Environment variables' and 'Tool Locations', and a 'Save' button.

Jenkins

search admin log out

Jenkins > Nodes >

Back to Dashboard

Manage Jenkins

New Node

Configure

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Name linux-slave-1

Description

of executors 1

Remote root directory

Labels

Usage Use this node as much as possible

Launch method Launch agent via execution of command on the master

Launch command

Availability Keep this agent online as much as possible

Node Properties

☐ Environment variables

☐ Tool Locations

Save

Remote directory is mandatory

No launch command specified

Here specify No. of executors.

Executor

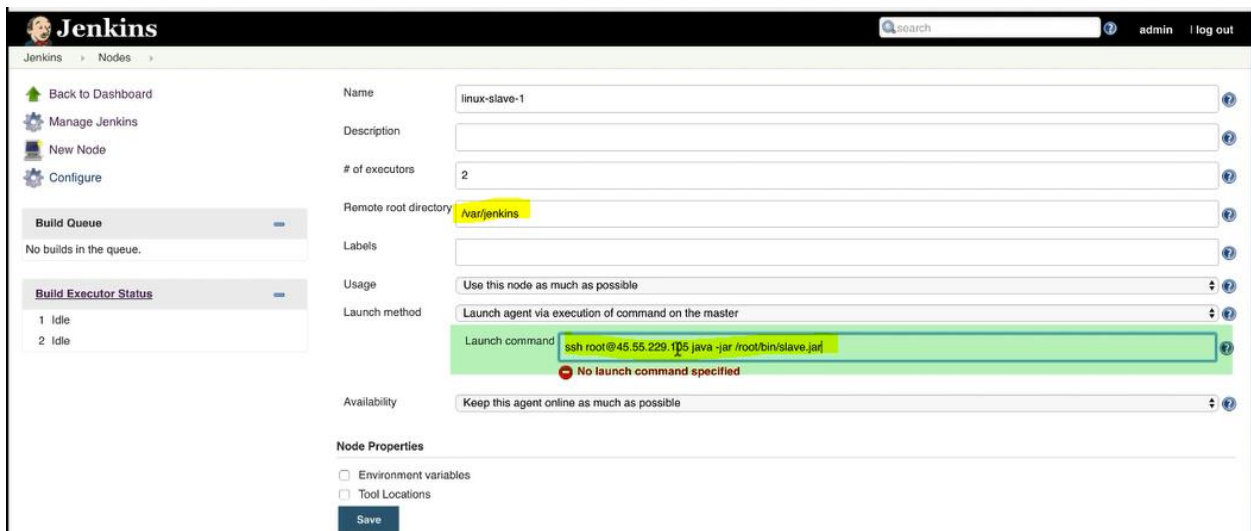
- A Jenkins executor is one of the basic building blocks which allow a build to run on a node.
- Think of an executor as a single “process ID”, or as the basic unit of resource that Jenkins executes on your machine to run a build.
- This number executors basically specifies the maximum number of concurrent builds that Jenkins may perform on this agent.
- A good value for the number of executors to start with would be the number of CPU cores on the machine.
- Setting a higher value would cause each build to take longer, but could increase the overall throughput.
- For example, one build might be CPU-bound, while a second build running at the same time might be I/O-bound — so the second build could take advantage of the spare I/O capacity at that moment.

Now in the same page we need to specify the root directory where the builds has to be stored and specify the launch method.

Root directory : `/var/jenkins`

Launch method : `ssh root@<slave-ip> java -jar /root/bin/slave.jar`

Now click on Save button.



The screenshot shows the Jenkins 'New Node' configuration page for a node named 'linux-slave-1'. The page is divided into a left sidebar with navigation links (Back to Dashboard, Manage Jenkins, New Node, Configure) and a main configuration area. The configuration area includes fields for Name, Description, # of executors (set to 2), Remote root directory (set to /var/jenkins), Labels, Usage (set to 'Use this node as much as possible'), Launch method (set to 'Launch agent via execution of command on the master'), and Availability (set to 'Keep this agent online as much as possible'). The 'Launch command' field is highlighted in green and contains the command 'ssh root@45.55.229.115 java -jar /root/bin/slave.jar'. Below this field, a red error message states 'No launch command specified'. At the bottom, there are checkboxes for 'Environment variables' and 'Tool Locations', and a 'Save' button.

Jenkins

Back to Dashboard

Manage Jenkins

New Node

Configure

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Name: linux-slave-1

Description:

of executors: 2

Remote root directory: /var/jenkins

Labels:

Usage: Use this node as much as possible

Launch method: Launch agent via execution of command on the master

Launch command: ssh root@45.55.229.115 java -jar /root/bin/slave.jar

No launch command specified

Availability: Keep this agent online as much as possible

Node Properties

☐ Environment variables

☐ Tool Locations

Save

The screenshot shows the Jenkins 'Nodes' page. On the left sidebar, there are links for 'Back to Dashboard', 'Manage Jenkins', 'New Node', and 'Configure'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing 'master' with '1' idle). The main table lists two nodes:

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	linux-slave-1		N/A	N/A	N/A	N/A	N/A
	master	Linux (amd64)	In sync	17.08 GB	0 B	17.08 GB	0ms
Data obtained			1 min 20 sec	1 min 20 sec	1 min 20 sec	1 min 20 sec	1 min 20 sec

A 'Refresh status' button is located at the bottom right of the table. The 'linux-slave-1' node is highlighted in yellow, indicating it is not active.

The above picture indicates that the slave machine is not active. After start the slave machine click on Refresh status (or) refresh the page.

The screenshot shows the Jenkins 'Nodes' page after refreshing. The 'linux-slave-1' node is now active and online. The 'Build Executor Status' section shows 'master' with '1' idle and 'linux-slave-1' with '1' idle. The main table shows updated data:

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	linux-slave-1	Linux (amd64)	In sync	N/A	0 B	17.21 GB	2549ms
	master	Linux (amd64)	In sync	17.08 GB	0 B	17.08 GB	0ms
Data obtained			3.1 sec	3 sec	3 sec	3 sec	3 sec

A 'Refresh status' button is still present at the bottom right of the table.

The above indicates that slave machine is configured and clustered with the master machine successfully.