Master – Slave- Jenkins:

Jenkins uses a Master-Slave architecture to manage distributed builds. In this architecture, Master and Slave communicate through TCP/IP protocol.

In master-slave architecture of Jenkins, master represents basic installation of Jenkins and it handles all tasks for build system.

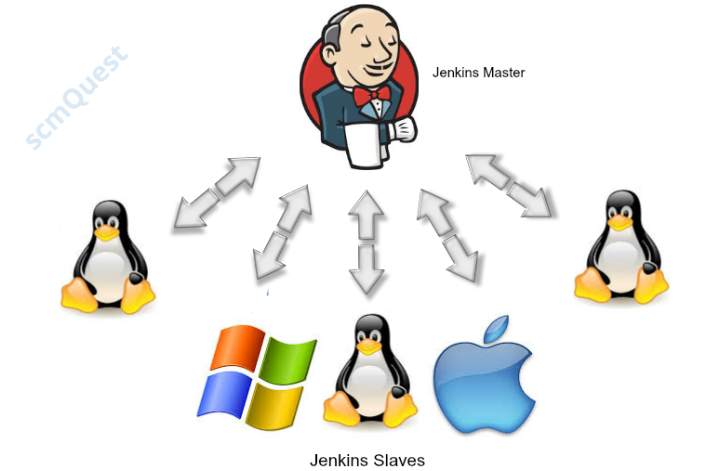
Jenkins master is used to handle following things:

* Scheduling build jobs.
* Dispatching builds to the slaves for the actual execution.
* Monitor the slaves (possibly taking them online and offline as required).
* Recording and presenting the build results.
* A Master instance of Jenkins can also execute build jobs directly.

A slave is a computer that is set up to offload build projects from the master and once connection is established between master and slave, tasks distribution is done automatic. Each slave runs a separate program called a “slave agent”. There is no need to install the full Jenkins on a slave. There are various ways to start slave agents, but in the end the “slave agent” and “Jenkins master” needs to establish a bi-directional communication link (for example a TCP/IP socket.) in order to operate.

Following are the characteristics of Jenkins Slaves:

* It hears requests from the Jenkins Master instance.
* Slaves can run on a variety of operating systems.
* The job of a Slave is to do as they are told to, which involves executing build jobs dispatched by the Master

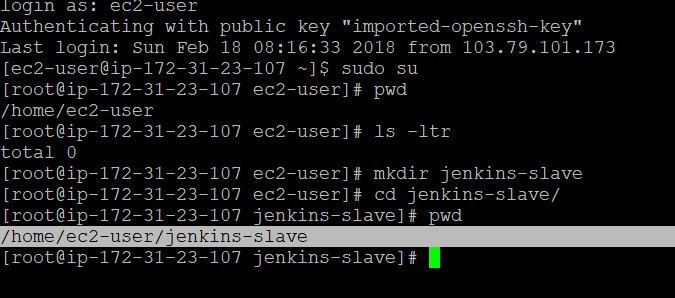


**Lab:**

Create a new ec2 instance for slave:

Step 1: Create new directory:

/home/ec2-user/jenkins-slave



sudo yum install -y git java-1.8.0-openjdk-devel aws-cli

sudo alternatives --config java

(We need to install required build tool)

sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo

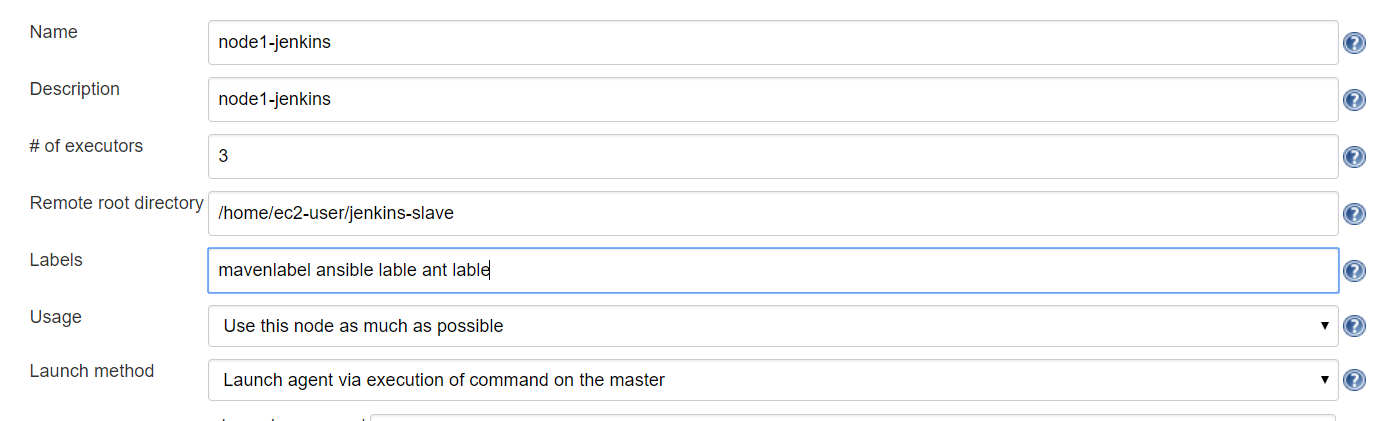
sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo

yum install -y apache-maven

mvn --version

Step 3: Manage Jenkins->Manage node.

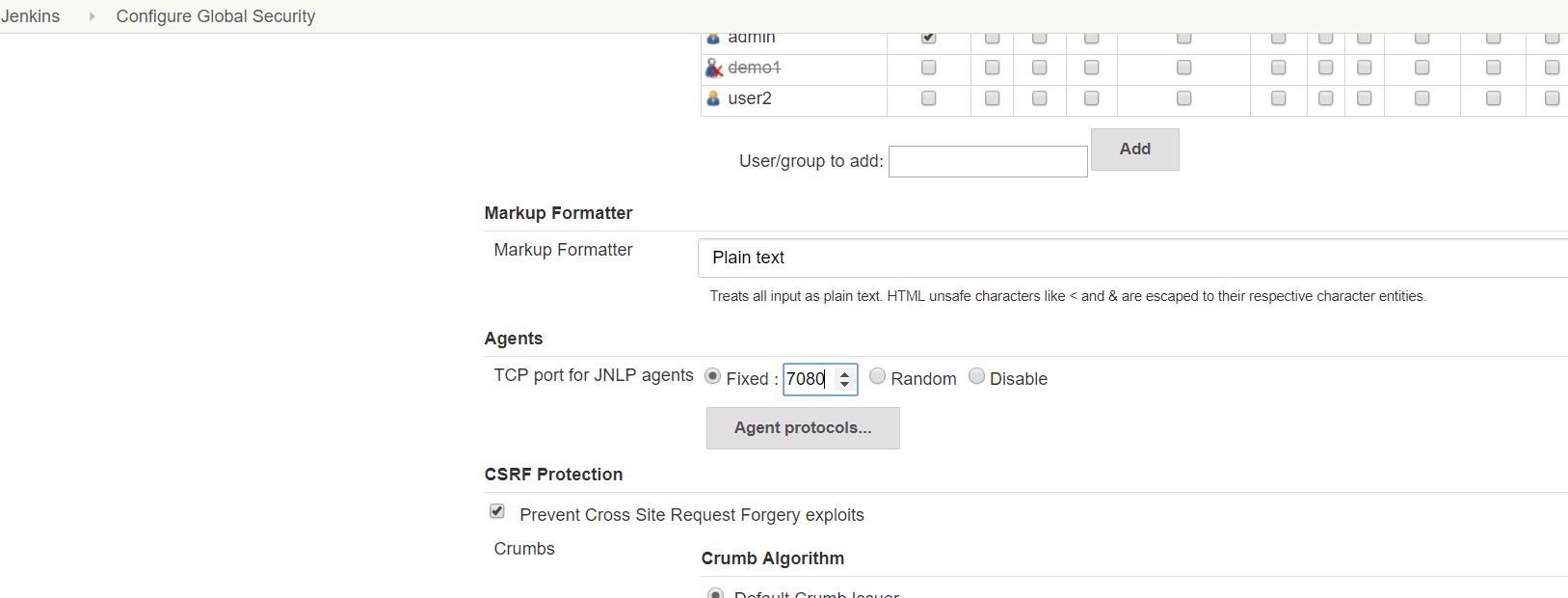
Copy slave vm’s path (step 1) and update Remote root directory.



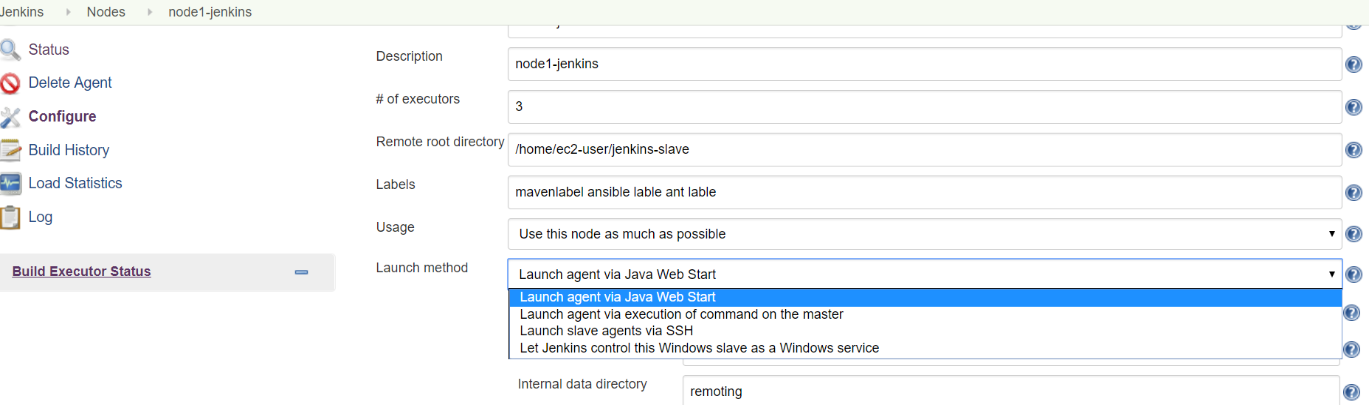
In labels section: you can provide labels (if you have multiple slave machines, similar to AWS ELB just like cluster)

And save it. Now you can see it is not connected, so we have to provide some methods to connect our slave. You can provide methods in manage node section. But as a best practice we will enable some port for our slave.

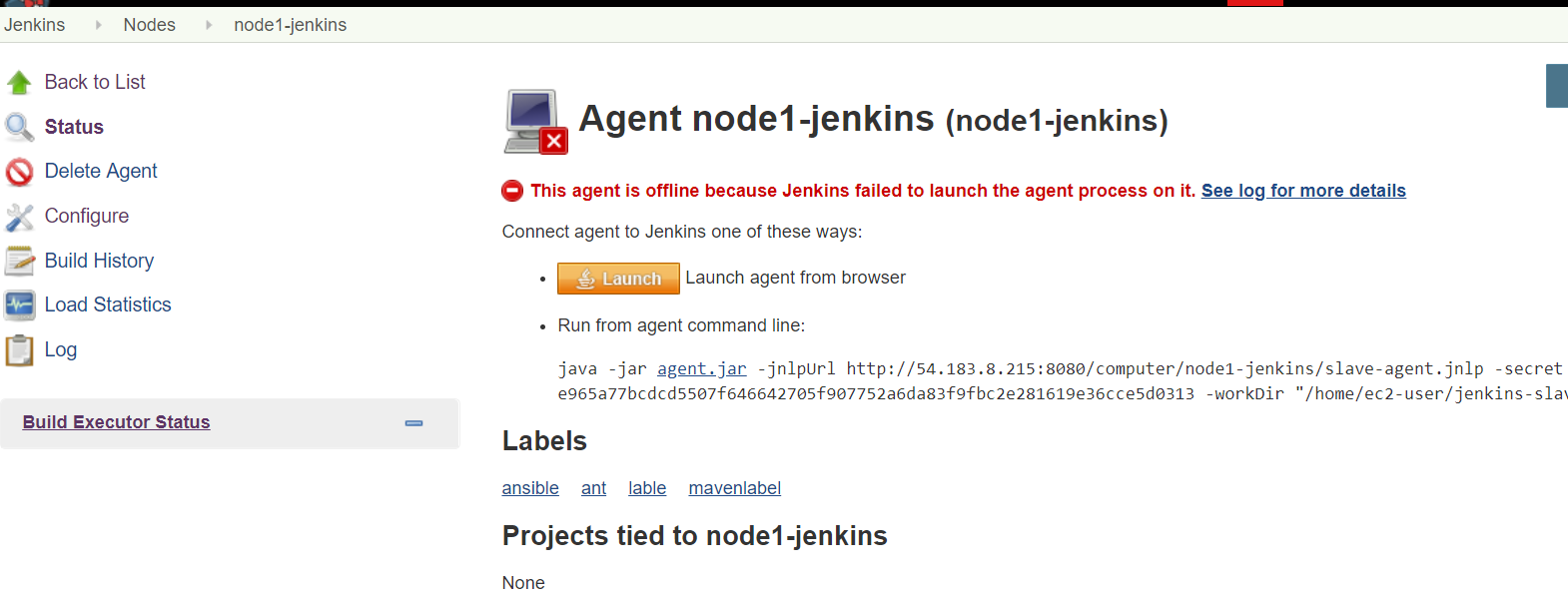
Step 3: Go to manage Jenkins -> configuration global security->in agent section provide any port number and save. Java Network Launch Protocol (JNLP)



Now go to manage Jenkins-manage node-go to the node🡪 agent java web start option will come automatically, and save.



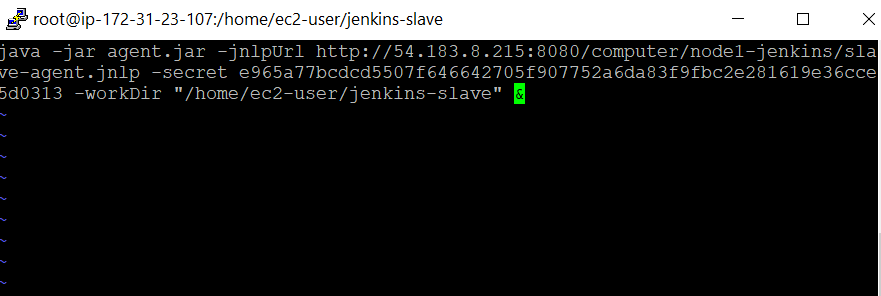
It will give you output, copy output command and execute on slave machine.



For best practice you can create a file called startup.sh and paste output command. (So that slave will always run in backed.) add & at the end of command.

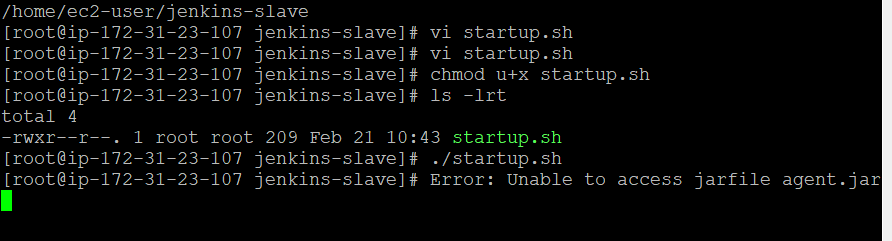
NOTE: UPDATE IP in SCRIPT of MASTER

**vi startup.sh**



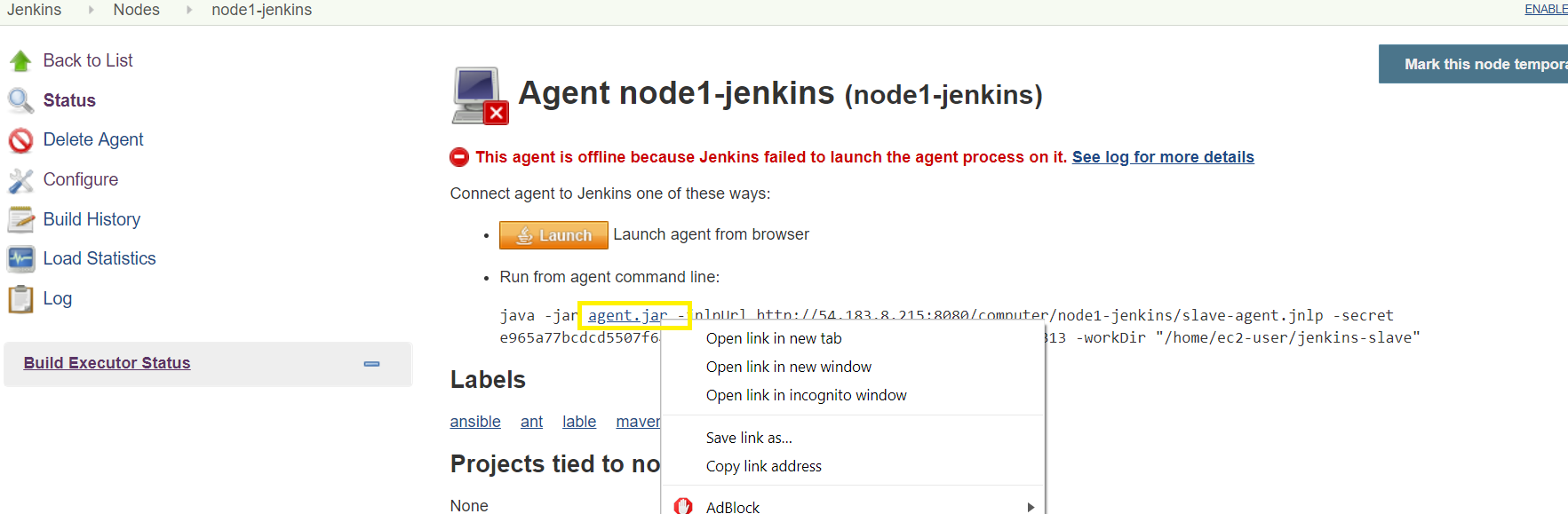
And save. (Note: you can also paste output command without creating startup.sh file)

Change the permission and execute.



It will through you an error agent.jar is missing.

Now go to Manage node section click on agent.jar and copy link access. Go to your slave machine and do wget and paste (make sure you have wget package if not installed then install **yum install wget**)





Once you execute. You will get output like connected.



If you get any error check for java version. Install java 8

(Note: Build tool should be there on Node for building your job)

And execute command **./startup.sh** (wait for some time, command is executing in backend)

Step 5: Now configure any job. Under General section: Restrict where the project can be build

Choose label name

Build now

Check workspace on your slave.

Done