# add PPA repo

sudo add-apt-repository ppa:openjdk-r/ppa

# Install OpenJDK

sudo apt-get update

sudo apt-get install openjdk-8-jdk

# Switch to Java 8

sudo update-alternatives --config java

# Check Java installation

java -version

# Add PPA repo for Jenkins stable LTS

wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.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

# Install Jenkins

sudo apt-get install jenkins

# Start Jenkins at boot time

sudo systemctl start jenkins.service

sudo systemctl enable jenkins.service

#Re-directing Port

sudo iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080  
sudo sh -c "iptables-save > /etc/iptables.rules"

--------------------------------------------------------------------------------------------------------------------------------

# Pull official Docker image for Jenkins LTS

docker pull jenkins/jenkins:lts

# Start Docker container

docker run --rm -dit -p 80:8080 -p 50000:50000 --name jenkins --hostname jenkins -v jenkins\_home:/var/jenkins jenkins/jenkins:lts

docker run -p 8080:8080 -p 50000:50000 -v jenkins\_home:/var/jenkins jenkins/jenkins:lts

# Start Docker container in detached mode

docker run -p 8080:8080 -p 50000:50000 -v jenkins\_home:/var/jenkins -d jenkins/jenkins:lts

**Jenkins Docker Integration:**

**On Docker Host as below:**

Jenkins master must be installed with Docker engine . We need to edit the Docker host docker file to allow jenkins server to run the images.

nano /lib/systemd/system/docker.service and add following line.

ExecStart=/usr/bin/dockerd -H unix:// -H tcp://0.0.0.0:2375

systemctl daemon-reload

service docker restart

Run curl http://localhost:2375/images/json to confirm.

**On jenkins Server:**

sudo usermod -a -G docker jenkins

sudo service jenkins restart

*If above dont work:*

usermod -aG root jenkins

chmod 664 /var/run/docker.sock

chmod 777 /var/run/docker.sock

From the remote host run :

docker -H tcp://<DockerHost>:2375 ps

docker -H tcp://10.1.1.130:2375 ps

docker -H tcp://10.1.1.254:2375 run --rm -dit --name jenkins2 --hostname jenkins2 --network ansible\_nw sreeharshav/jenkins:v2

**Jenkins Build Script using the webhooks:**

#!/bin/bash

VERSION=$(date +%H-%M-%S)

docker build -t sreeharshav/testingbuild:${VERSION} .

docker push sreeharshav/testingbuild:${VERSION}

docker -H tcp://10.1.1.200:2375 stop nginx

docker -H tcp://10.1.1.200:2375 run --rm -dit -p 8000:80 --name nginx --hostname nginx sreeharshav/testingbuild:${VERSION}

**Using AMAZON EC2 as jenkins slave:**

1. Create a AMI with JDK 1.8 and note down the AMI ID.
2. Create a users in AWS IAM with Access Key and Secret Key. This user must have permission to deploy EC2 machines.
3. Create a Jenkins Credentials using the above access and secret key.
4. Install Amazon EC2 plugin in Jenkins.
5. From Manage Jenkins -> Configure System, go down and select Amazon EC2 Cloud.
6. Enter the Access details and AMI Details and make sure proper labeling is done.
7. Create a job and provide the label and run the job.
8. This should deploy a new EC2 instance and run the job.
9. Based on the idle minutes selected, the EC2 machine will be terminated.

**Using Docker Container as jenkins slave:**

1. We are going to use the Docker Image jenkinsci/ssh-slave:latest
2. Install the Docker Plugin from Manage Plugins page.
3. Perform the following steps in the Docker Host. In our case Jenkins and Docker host are the same.

nano /lib/systemd/system/docker.service and add following line.

ExecStart=/usr/bin/dockerd -H unix:// -H tcp://0.0.0.0:2375

systemctl daemon-reload

service docker restart

Run curl http://localhost:2375/images/json to confirm.

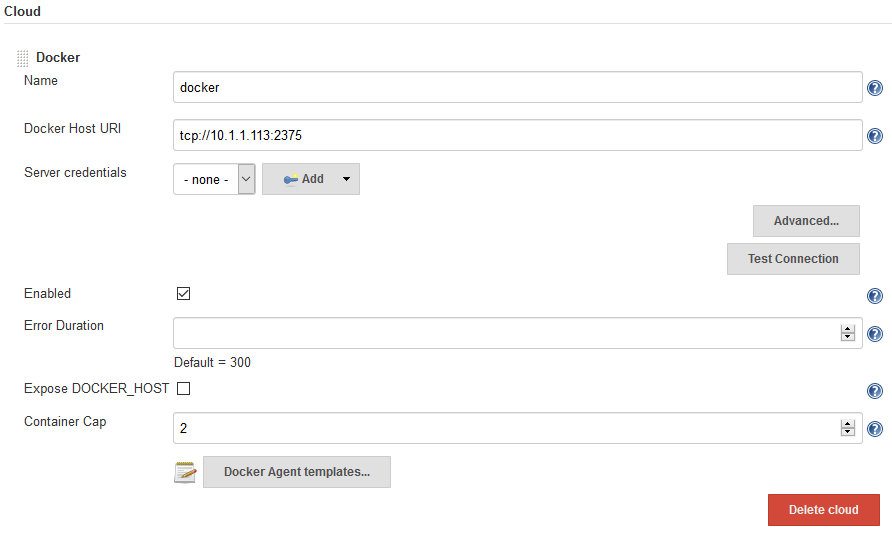
**On jenkins Server:**

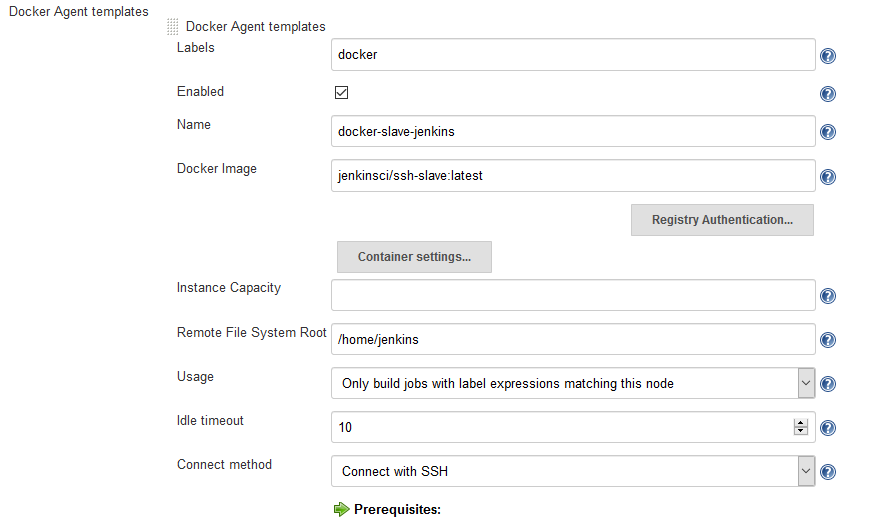
sudo usermod -a -G docker jenkins

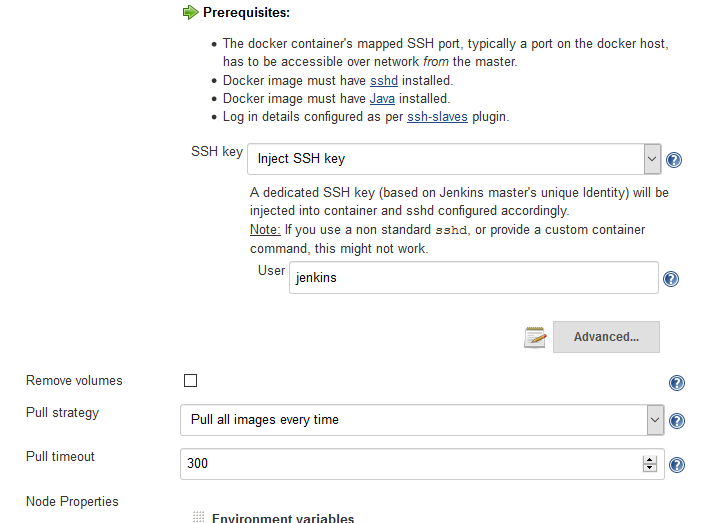
sudo service jenkins restart

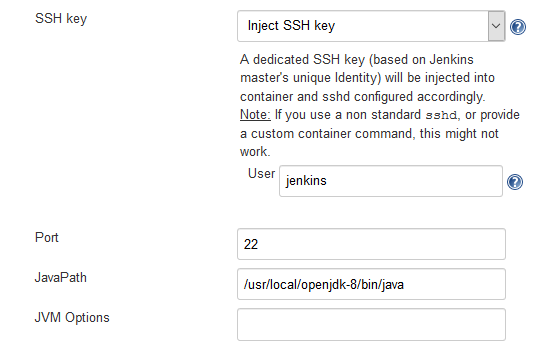
1. Similar to Amazon EC2 Cloud , we need to add the Docker Cloud. We also need to have a ssh key-pair.
2. Change user to jenkin and perform a ssh-keygen which generates public(id\_rsa.pub) and private(id\_rsa) key. Copy the contents of public key.
3. Add the docker cloud and enter all necessary details and make sure you give proper label. Dont forget to add the Enviroment Variable JENKINS\_SLAVE\_SSH\_PUBKEY.
4. Create a job with above labe and run it and it must deploy a container and run the job and once the job is done, container will be terminated.

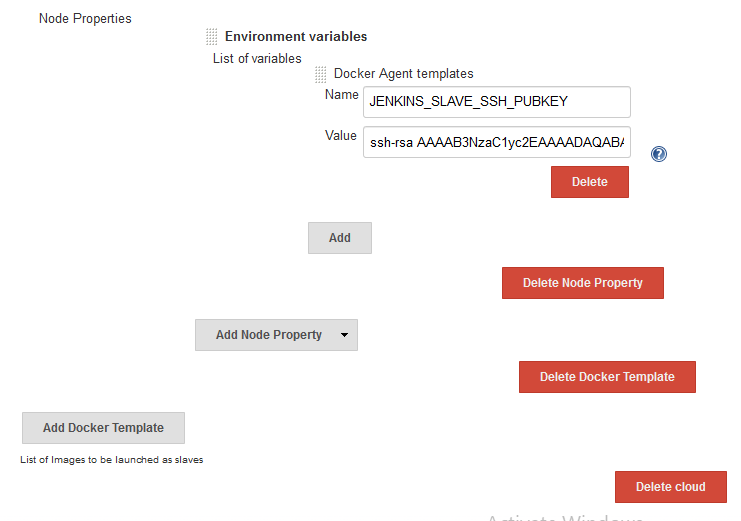
**FOLLOWING PIC IS VALID IF YOU USE SSH INJECT KEY:**



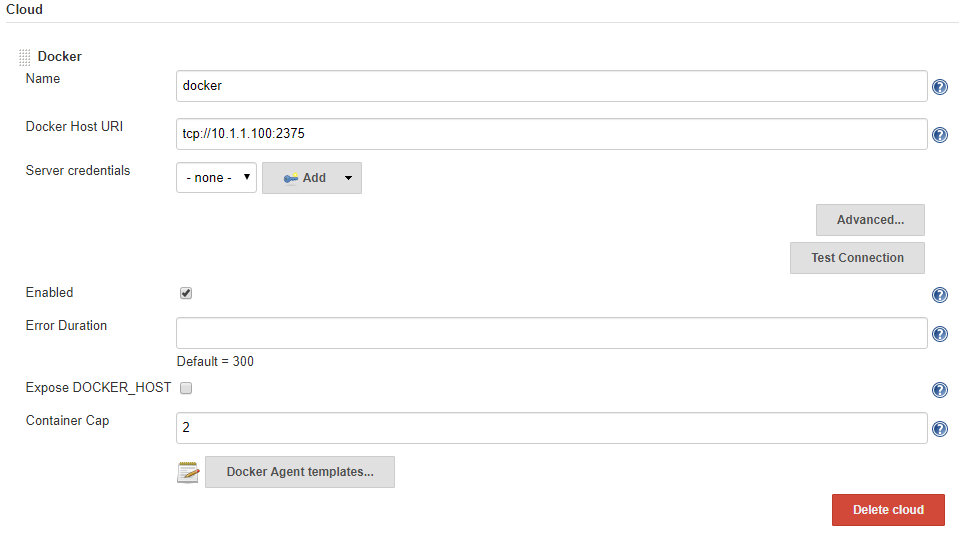


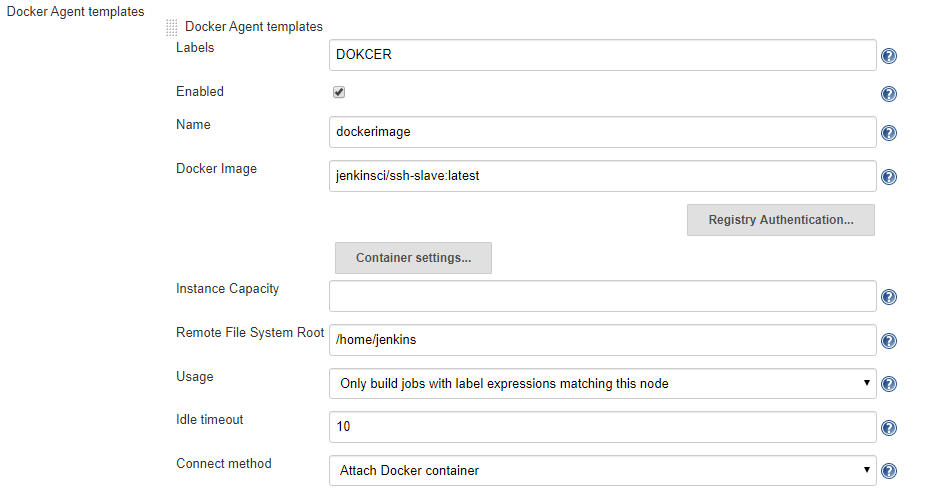


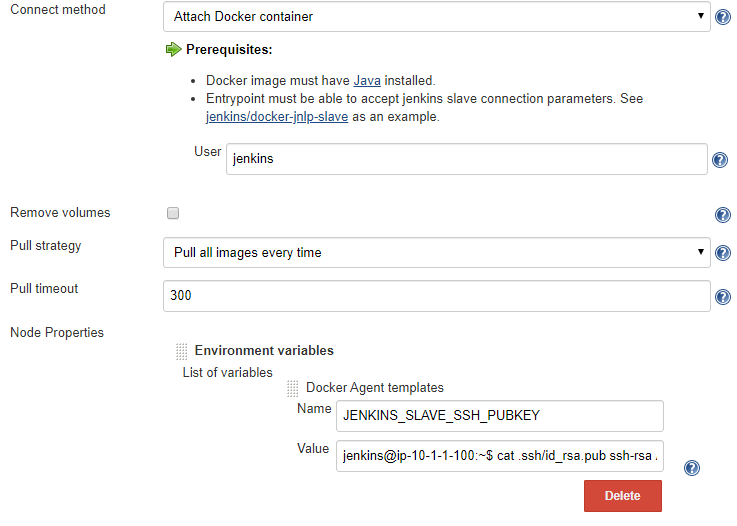




FOLLOWING PIC IS VALID IF YOU USE ATTACH CONTAINER (NOT SSH):







**Jenkins RBAC:**

1. Install [Role-based Authorization Strategy](https://wiki.jenkins.io/display/JENKINS/Role+Strategy+Plugin) plugin.
2. Got to Manage Jenkins - > Global Security-> Select “Role-Based Strategy”.
3. Check if the Manage and Assign Roles are visible.
4. Create two users, Allan and Bob. Allan works on Java and Bob works on Python.
5. We want Allan access only Java Projects and Bob Python projects only.
6. Create Global Roles with total read access and project roles for java and python.
7. Assign the Java role to Allan and Python role to Bob.
8. Create test projects with names java and python.
9. Login with Allan and Bob and check if they are able to run.

Project Role Pattern:

java.\*

python.\*

**WHEN USING JENKINS WITH EFS USE THE FOLLOWING MOUNT:**

(Works with Amazon Linux and Ubuntu 16.04)

fs-fa0cc77a.efs.us-east-1.amazonaws.com:/ /var/lib/jenkins nfs defaults 0 0

**Jenkins version upgrade:**

https://www.thegeekstuff.com/2016/06/upgrade-jenkins-and-plugins/