Install Jenkins in offline mode on CentOS / RHEL

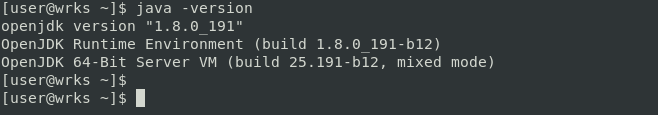
There are times when you would need to do an offline installation of the Jenkins. This may be to meet certain requirements of the workplace that you are operating in. I came across this issue some time back and internet is woefully out of articles for proper steps on this one. In this blog post, we’ll learn how to do an offline installation of Jenkins.

Install OpenJDK Java

First, we need to check what version of java is installed on the machine using:

java -version

If OpenJDK version of JDK is installed, you would see something like below:



If your version of installed java is different, then uninstall it using:

$ sudo yum uninstall java -y

sudo yum remove <package-name>

sudo yum autoremove

After this you can install it using:

$ sudo yum install java-1.8.0-openjdk.x86\_64 -y

Add Environment Variables for Java

In order to help Jenkins locate the JVM properly, you need to set two environment variables: “JAVA\_HOME” and “JRE\_HOME”:

$ export JAVA\_HOME=/usr/lib/jvm/jre-1.8.0-openjdk

$ export JRE\_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre

You will also need this to add to /etc/profile or .bash\_profile so as to make it persistent.

Install Jenkins from RPM Release Package

For the purpose of offline installation, RPM packages for jenkins are available at <https://pkg.jenkins.io/redhat/>. You can also download specific versions of Jenkins from the repository. Once you have downloaded the requisite rpm version, you can install it using:

Jenkins Redhat Packages

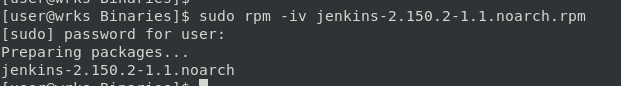
* sudo wget -O /etc/yum.repos.d/jenkins.repo <https://pkg.jenkins.io/redhat-stable/jenkins.repo>
* sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key

If you've previously imported the key from Jenkins, the rpm --import will fail because you already have a key. Please ignore that and move on

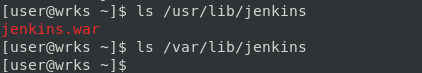
* yum install fontconfig java-17-openjdk
* yum install jenkins

$ sudo rpm -iv jenkins-2.150.2-1.1.noarch.rpm

You should be able to see output like below:



Once you are done with the above steps, Jenkins will be installed in /var/lib/jenkins folder. You will also be able to see the jenkins.war file in /usr/lib/jenkins folder:



Do note that /var/lib/jenkins will be empty at this time.

Install Jenkins from WAR file

This step is not required, if you have performed previous step already. If you are using WAR file to install jenkins, you would not be able to configure it as a service. The location to download WAR files is <http://mirrors.jenkins.io/war-stable>. Thereafter you can run jenkins using:

$ java -jar jenkins.war

So we need to configure it in a way, where we’ll be able to treat it like a service, starting and stopping it and letting it run in the background with ease. The service will essentially be working as a wrapper.

To do this, first we need to make sure the WAR file you’ve downloaded is sitting in a location convenient for long-term storage and use:

$ sudo cp jenkins.war /usr/local/bin/jenkins.war

Then, go to your /etc/systemd/system/ directory, and create a new file called jenkins.service. Then add the following lines to the new jenkins.service file:

[Unit]

Description=Jenkins Service

After=network.target

[Service]

Type=simple

User=root

ExecStart=/usr/bin/java -jar /usr/local/bin/jenkins.war

Restart=on-abort

[Install]

WantedBy=multi-user.target

If you’ve seen configuration files before (INI files or similar), you’ll recognize the structure being used here. The bracketed text denotes a section heading. This means that, for instance, [Service] declares a section called “Service,” and all the assignments below it contain relevant information that the system will then know how to find and relate to the section header.

A configuration file, this one included, is typically a text file – meaning it has no intrinsic meaning to the computer. Rather, the text file will be parsed by some process, and that process will use the headings and other information to find its way around. For this reason, it’s technically irrelevant how a given configuration file is laid out – as long as the program which reads it can understand what everything means.

The first section, Unit, contains only two configuration directives. The first is simply a name. It can be whatever name you’d like, but ideally it should be one that uniquely identifies the new process. The second directive states what service, if any, is necessary for the current service to start.

In the next section, the Type directive allows you to select what type of startup this service will use. The value simple indicates that the process noted in the later directive ExecStart will be the primary process of the service being created. Really, type is unnecessary, as simple is assumed when type is unspecified, but we are leaving it in for clarity.

User specifies which user has control over this process, and Restart is used to indicate that, in this case, if the process terminates but the exit code implies error, the service will be restarted. This is useful in maintaining the continuity of the service in case of unexpected crashes.

As mentioned, ExecStart is the directive where we indicate what process is to become the main action of the service. This directive represents the main wrapper for Jenkins – the service will run the WAR through Java rather than treating it a foreground process.

Finally, in the Install section, multi-user.target indicates a target, called a runlevel prior to CentOS 7. It provides for the system a sense of what resources to provide this service and what amount of intensity will be required by the user.

Enable Jenkins Service

If you have installed Jenkins from rpm repo, you can enable Jenkins service using below command:

$ sudo systemctl enable jenkins.service

If you have are Jenkins WAR file, then you need to write service configuration file as mentioned above and then run below command:

$ sudo systemctl daemon-reload

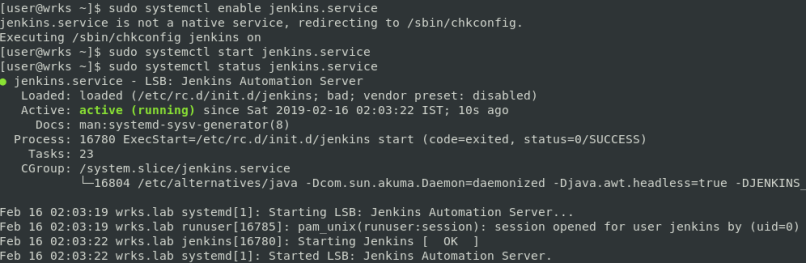
Start and Check Status of Jenkins

We can now start and monitor status of Jenkins using below command:

$ sudo systemctl start jenkins.service

$ sudo systemctl status jenkins.service

If everything goes well, you should be able to see below like status:



Configure firewall exceptions

With CentOS and RHEL, you will also need to configure firewall to allow jenkins connections. By default, Jenkins operates on TCP Port 8080. So, we can use below commands:

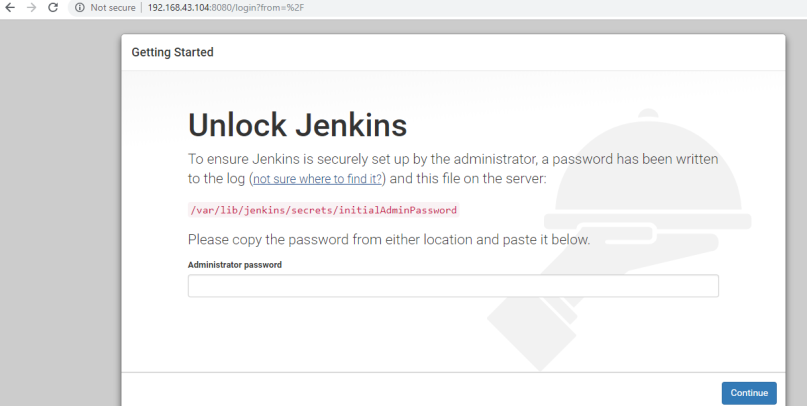
$ sudo firewall-cmd --zone=public --add-port=8080/tcp --permanent

$ sudo firewall-cmd --zone=public --add-service=http --permanent

$ sudo firewall-cmd --reload

Access Jenkins Web Portal

As mentioned above, by default Jenkins web portal is available on TCP Port 8080. So to access web portal, open browser of your choice and type in the address of the Jenkins server followed by port:



Setup Jenkins Plugins in Offline Mode

To install plugins in offline mode, you have to download the respetive .hpi plugin file from <https://updates.jenkins-ci.org/download/plugins/>. However, if you go this route, you will have to download the dependencies too, which can make this task near impossible. The easier way is to:  
1. Install Jenkins and the related plugins on your workstation, or any other machine (It can be Windows machine too).  
2. Once all the plugins are downloaded, you can copy the plugins folder from the workstation to the Jenkins server /var/lib/jenkins folder.  
3. After you copy, just stop and start Jenkins, and you will see that all your plugins are installed.

……………………………………………………………………..path……………………………………………………………………………………..

**Install OpenJDK 11:**

* sudo yum install java-11-openjdk
* java -version
* sudo alternatives --config java
* export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk
* export PATH=$PATH:$JAVA\_HOME/bin
* source ~/.bashrc

**2. Set JAVA\_HOME:**

export JAVA\_HOME=/path/to/your/java/11

3. Update PATH:

export PATH=$PATH:$JAVA\_HOME/bin

4. Apply Changes:

source ~/.bashrc # or source ~/.bash\_profile

**5. Verify Environment Variables:**

echo $JAVA\_HOME

echo $PATH