AWS EC2 Jenkins & Sonar Server Configs

Install steps on Linux **RHEL/Centos Distributions**:

```
sudo yum upgrade
```

Jenkins Installation and Configuration

Server Details - Jenkins

- IP address public jenkins: 3.6.93.46
- IP address private jenkins: 10.20.20.50
- Port: 8080
- Accessible URL: http://<ipaddress>:<port> (example: http://3.6.93.46:8080)
- un: admin pass: Sludi@2023

We can install Jenkins two ways on RHEL/Centos through a repo or repositroy and the WAR file.

Before going to install and configure the Jenkins first need to setup JAVA on server.

```
[root@lab ~]# yum -y install java-11-openjdk wget

Total download size: 32 M
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
(1/3): wget-1.14-15.el7.x86_64.rpm | 547 kB 00:00
(2/3): java-1.8.0-openjdk-1.8.0.144-0.b01.el7_4.x86_64.rpm | 238 kB 00:00
(3/3): java-1.8.0-openjdk-headless-1.8.0.144-0.b01.el7_4.x | 32 MB 00:01
--------------------------------------------------------------------------------
------
Complete!
```

Check java version

```
[root@lab ~]# java -version
```

By default Jenkins package is not available on the RHEL/Centos repositories. So that need to add and import the jenkins repository on machine by using below commands.

```
[root@lab ~]# cd /etc/yum.repos.d/
```

```
[root@lab yum.repos.d]# wget
https://pkg.jenkins.io/redhat-stable/jenkins.repo
```

```
[root@lab yum.repos.d]# rpm --import
```

```
https://pkg.jenkins.io/redhat-stable/jenkins.io.key
```

Install and configure Jenkins

```
[root@lab ~]# yum -y install jenkins
```

```
Installed:
 jenkins.noarch 0:2.60.3-1.1
```

```
Complete!
```

Start and enable the Jenkins service

```
[root@lab ~]# systemctl start jenkins
[root@lab ~]# systemctl enable jenkins
```

Jenkins runs on port 8080, Enable/Allow port 8080 from firewall by following below commands

```
[root@lab ~]# firewall-cmd --zone=public --add-port=8080/tcp --permanent
success
```

```
(OR)
[root@lab ~]# firewall-cmd --zone=public --add-service=http --permanent

success

[root@lab ~]# firewall-cmd --reload
```

To fix the **firewall-cmd** command not found error, you need to install **firewalld** on RHEL/CentOS 7 like below:

```
$ sudo yum install firewalld
```

You may ignore the Warning: ALREADY_ENABLED: 8080:tcp

Go to web browser and type “http://<ip address>:8080” .

Follow the below screenshots.

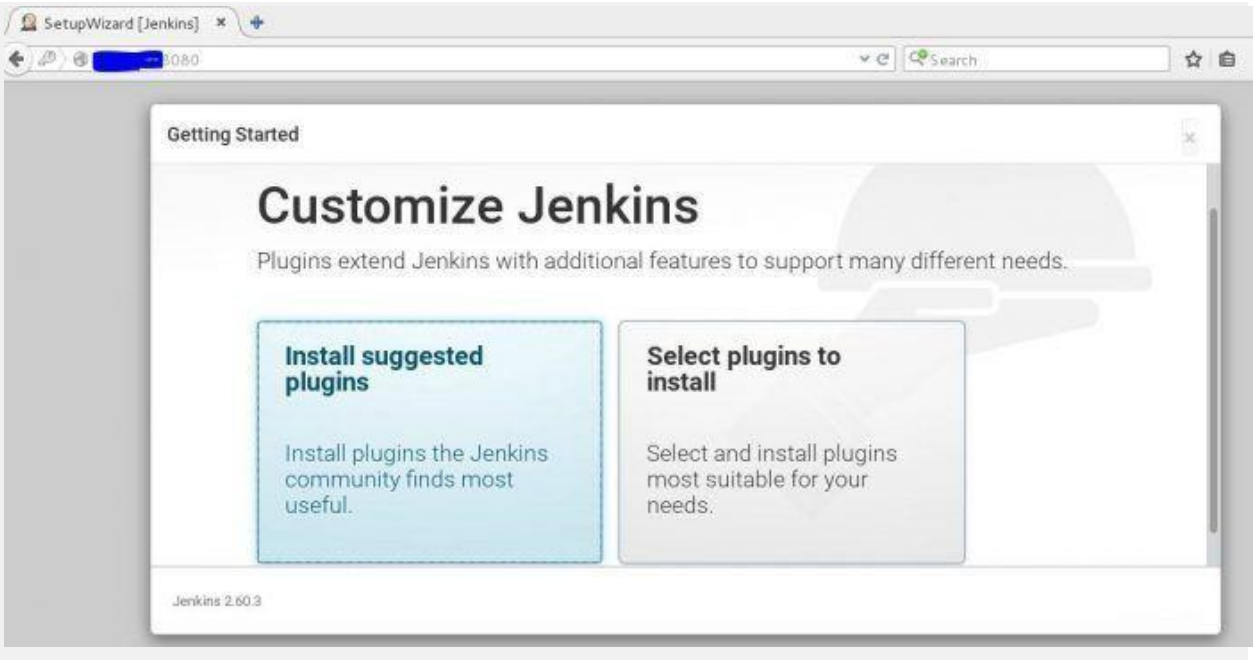
The Admin password is created and stored in the below file.

```
[root@lab ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
```

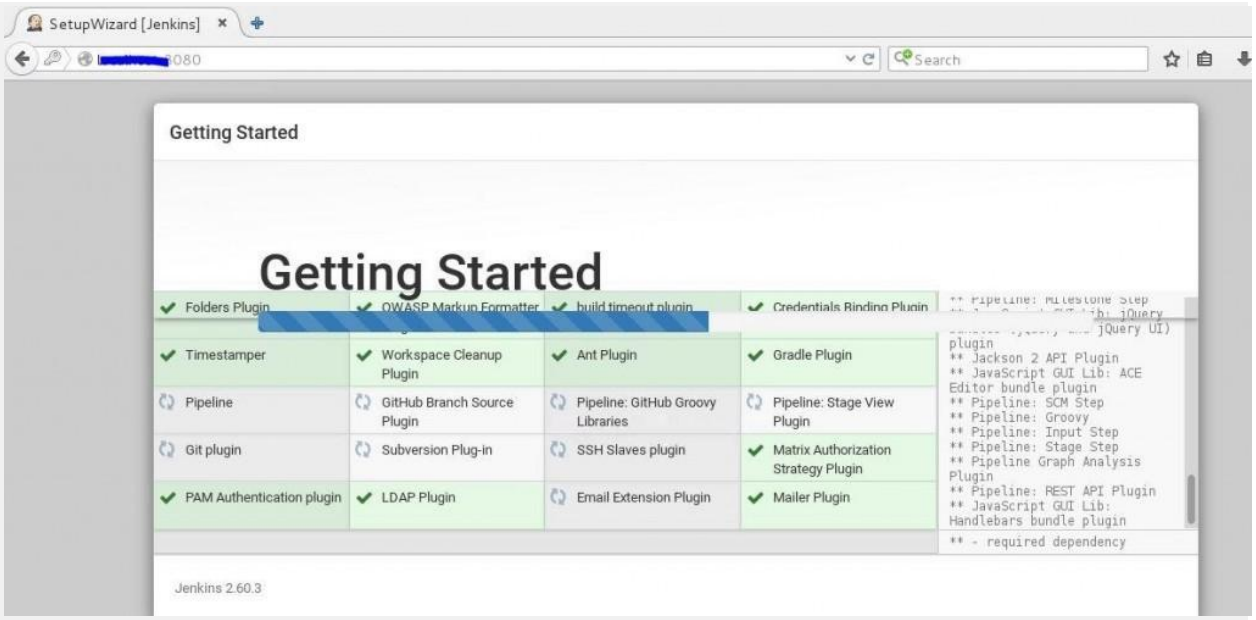
(OR)

```
[root@lab ~]# grep -A 5 passwd /var/log/jenkins/jenkins.log
```

Copy the password and insert , click on continue button. Select the option “Install suggested plugins”.

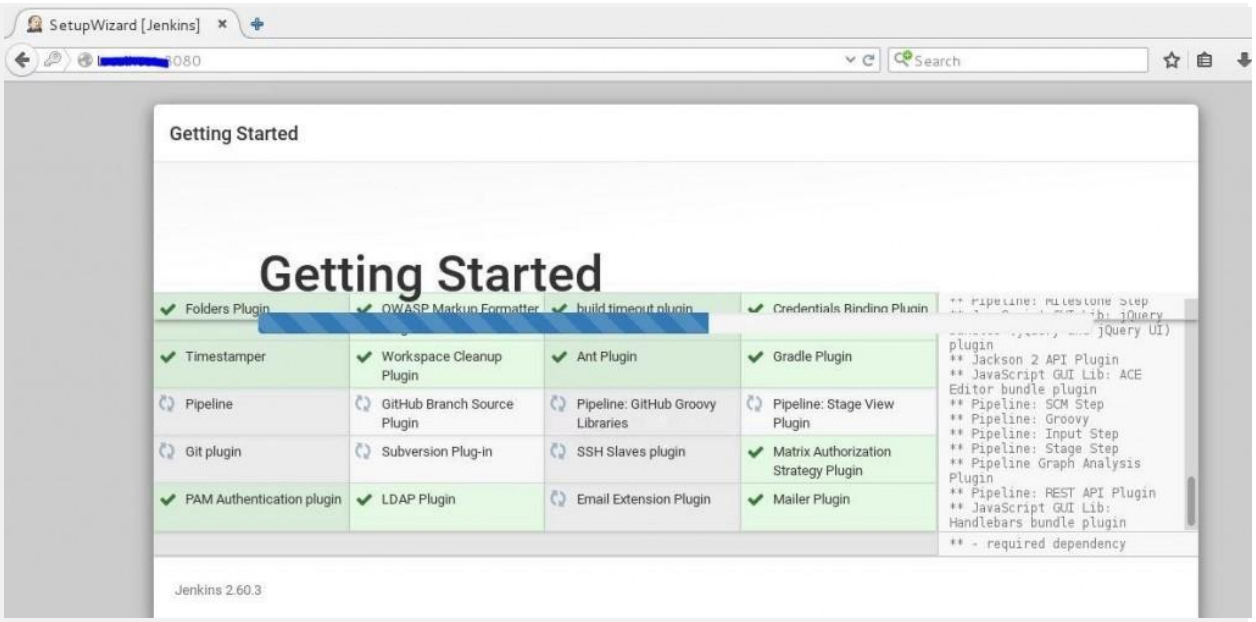


Install Jenkins Plugins

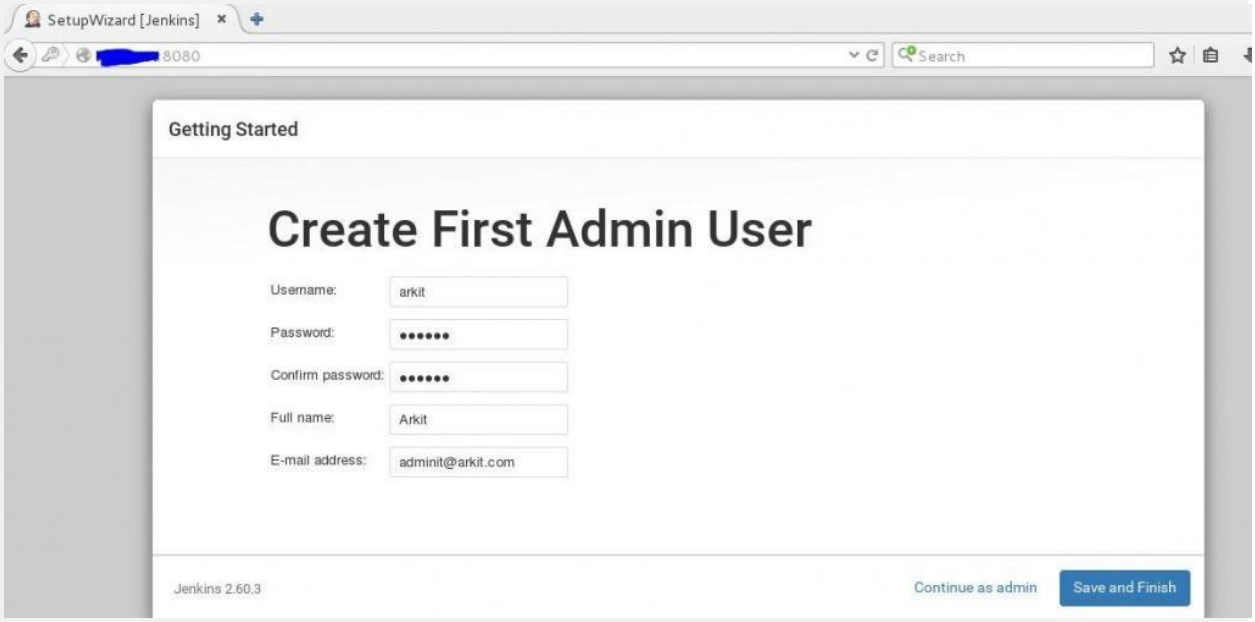


Getting Started with jenkins

After done with plugin installation it will ask to create Admin user.

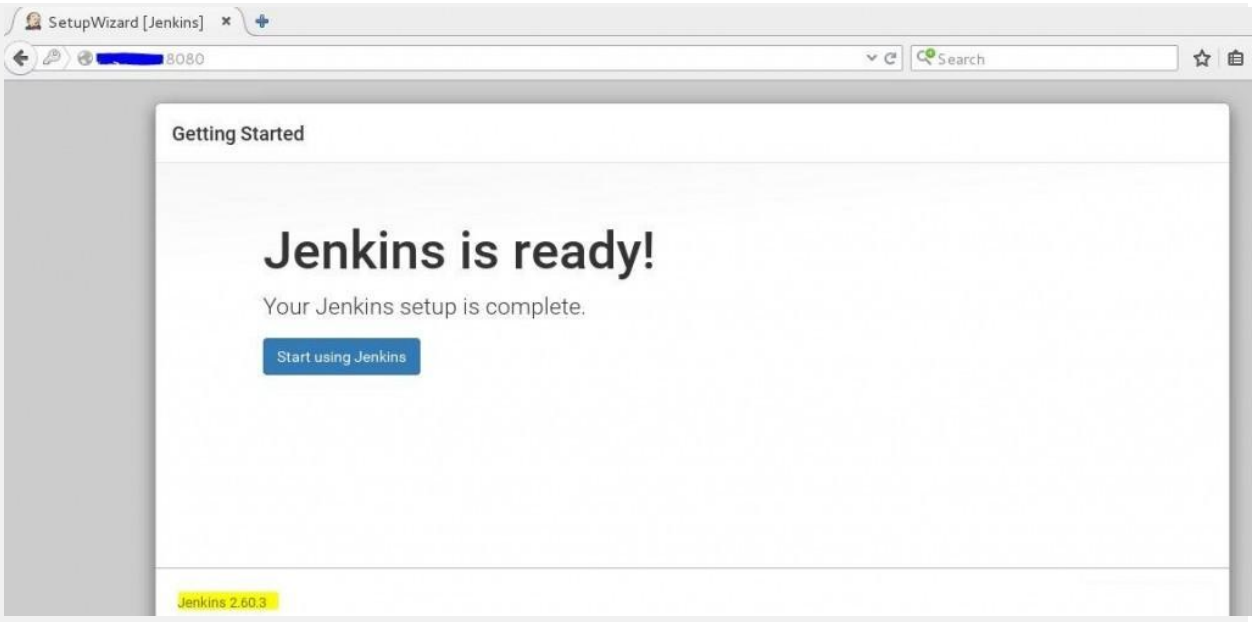


Getting Started



Create jenkins Admin User

Click on “Save and Finish” button.



Jenkins Ready to Use

Before starting to use Jenkins

Also install Git, Nodejs/Npm, Angular and Maven on this instance..

Install Git:

```
yum install git
```

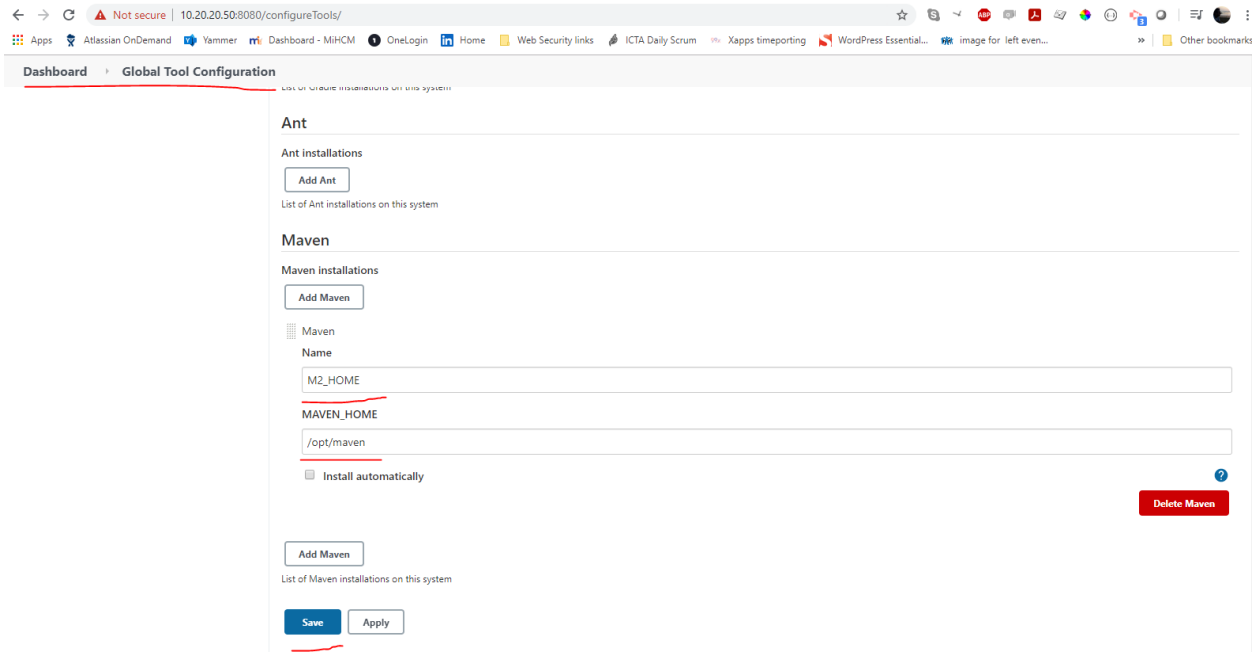
Install node/npm and Angular:

```
-------------- As root user --------------
# curl -sL https://rpm.nodesource.com/setup_14.x | bash -
```

```
yum -y install nodejs
```

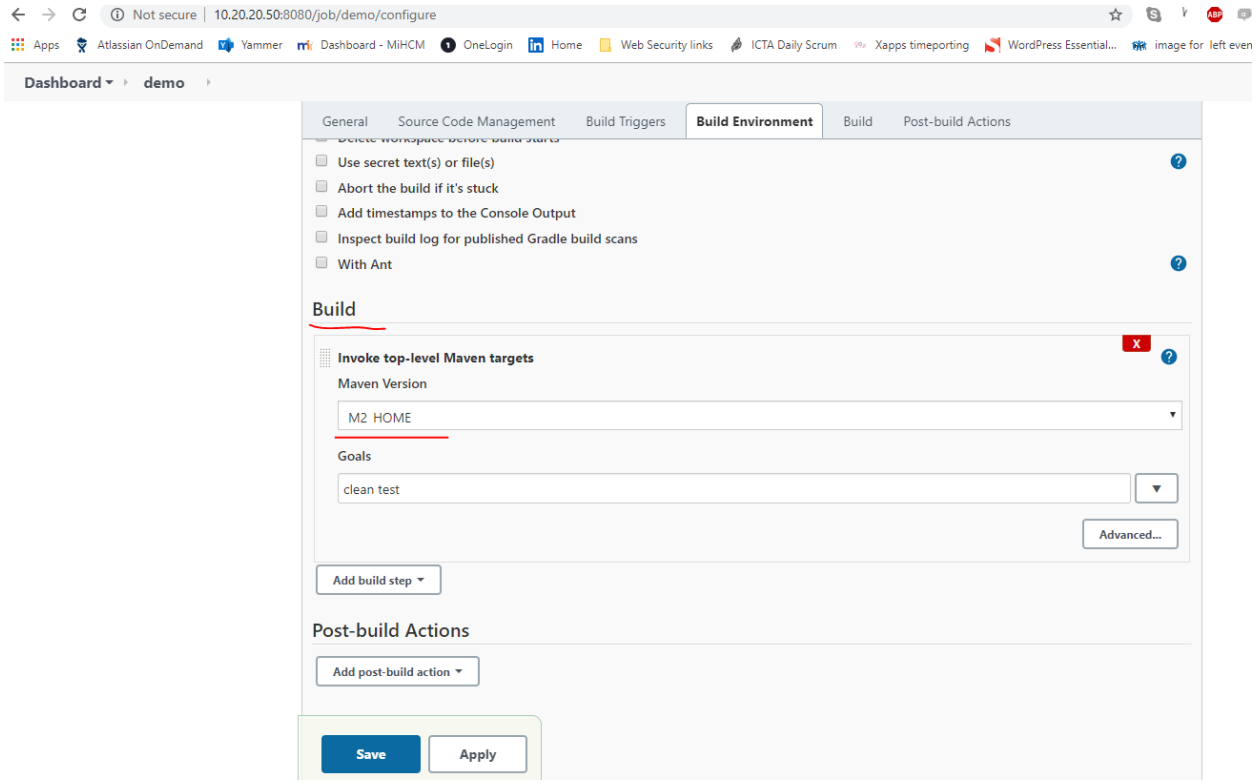
```
npm install -g @angular/cli
```

Then install Maven - https://tecadmin.net/install-apache-maven-on-centos/



After installing maven we need to set the Maven HOME within jenkins.. **Dashboard** -> **Global Tool Configuration**

Finally go to the Jenkins project configuration UI and set the version for maven which was created in the above step.



Securing our jenkins server

As mentioned in the official jenkins wiki, there are 2 aspects to look into when securing our Jenkins server instance.

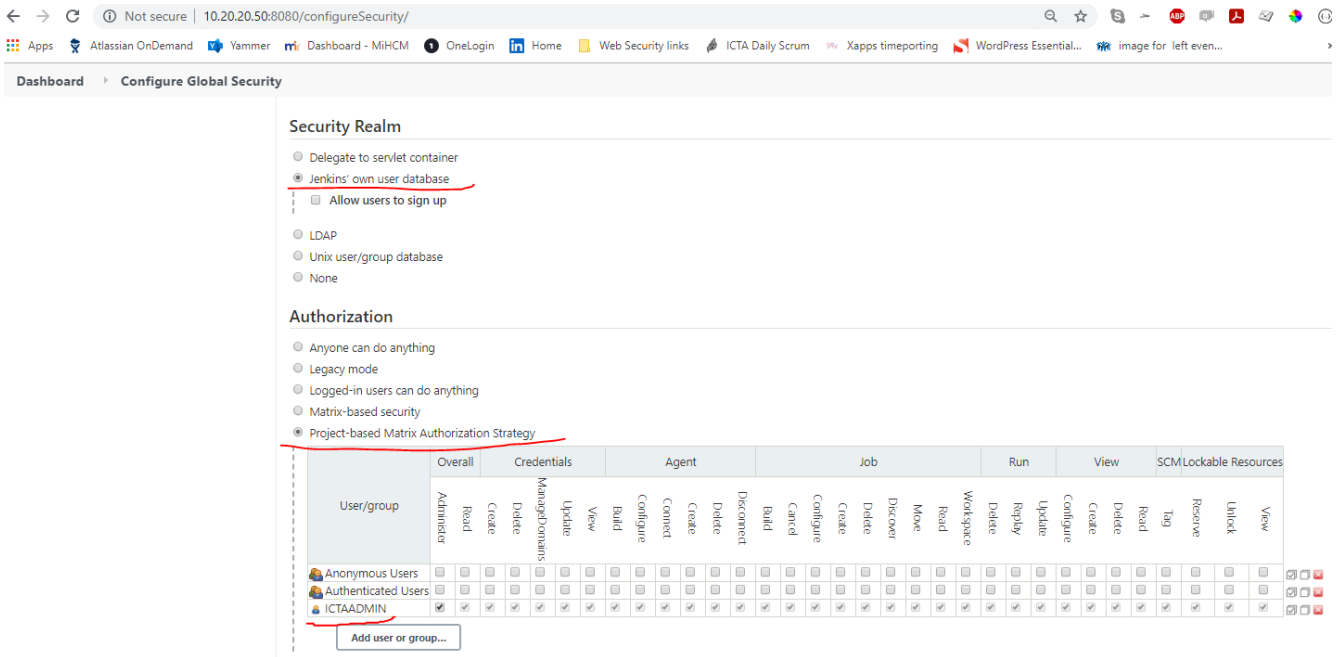
1.Access control, which ensures users are authenticated when accessing Jenkins and their activities are authorized.
2. Protecting Jenkins against external threats

Configuring Access Control

As described in the official jenkins wiki this setting is controlled mainly by two axes:

- ***Security Realm***, which determines users and their passwords, as well as what groups the users belong to.
- ***Authorization Strategy***, which determines who has access to what.

Here for our server setup the chosen security realm was to let **Jenkins run its own user database** on our ec2 instance. And the authorization strategy was set to **Project-based Matrix Authorization Strategy.** Refer below setup



Also the other best practices to secure our jenkins instance from external threats as mentioned in the official jenkins wiki page was followed ideally. Ex:- (CSRF protection is enabled by DEFAULT, Agent To Controller Access was set, Secured JENKINS_HOME etc...)

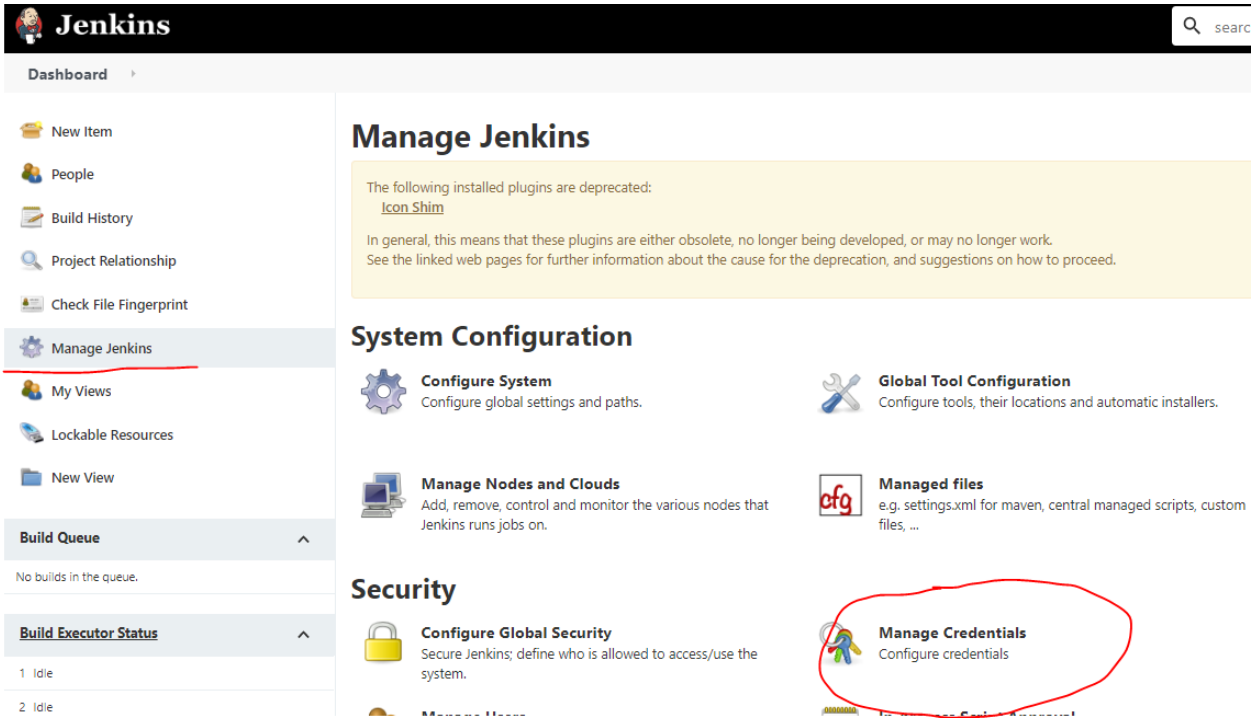
Integrate with Github

First we need to create a personal access token for the github user who is trying to clone our git repo. For this follow the instructions in the below link to create your own PAT (Personal access token)

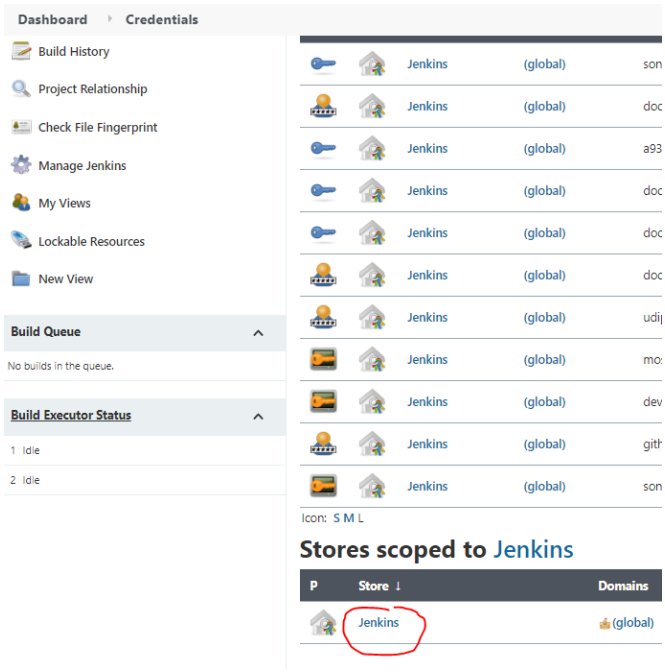
https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token

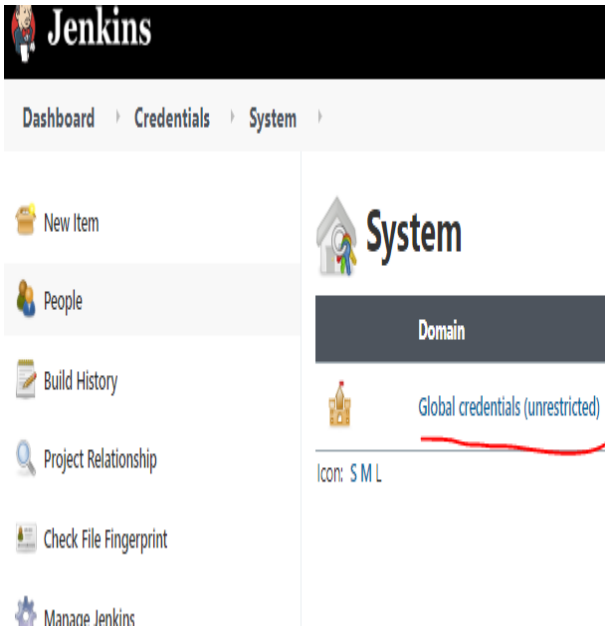
Generate the token by selecting the permissions you'd like to grant this token. (granted all permissions since our jenkins instance has been secured in the previous section)

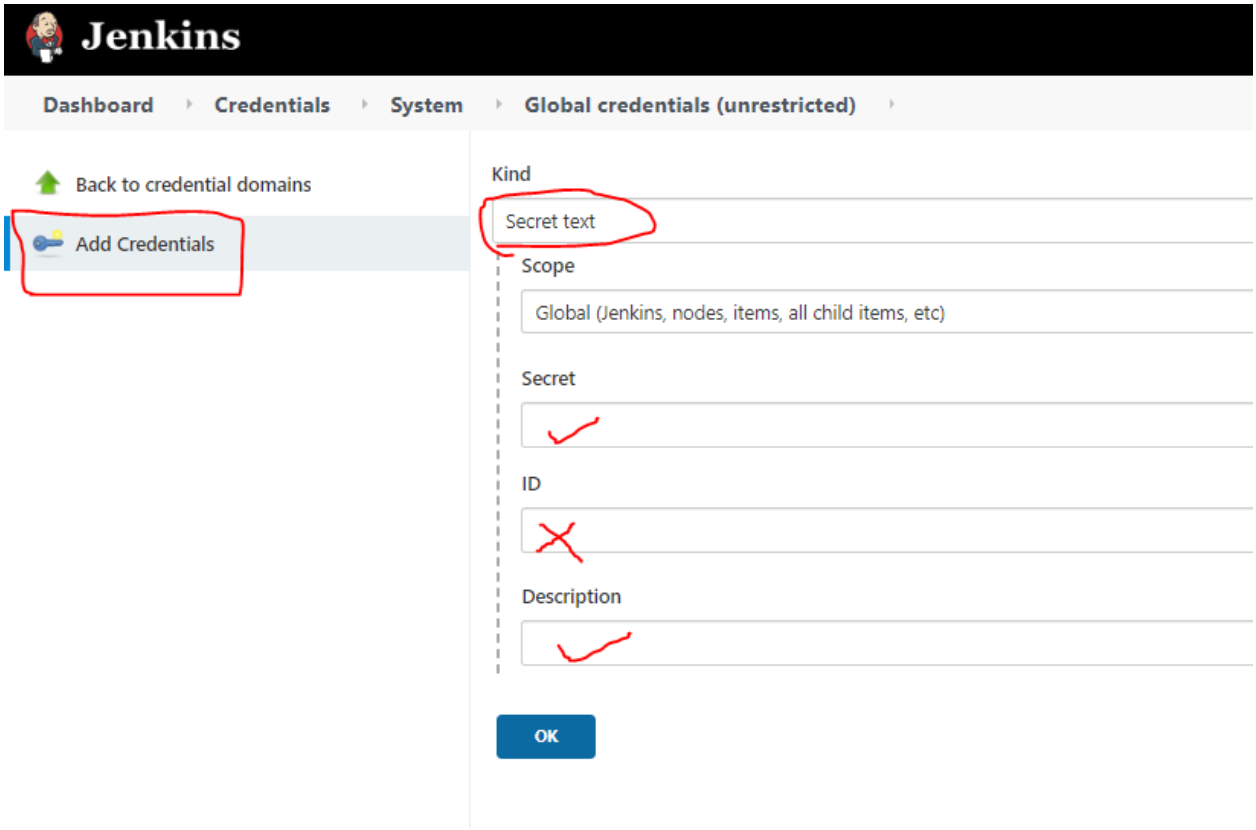
Once the PAT has been generated from the Github side we need to assign this as a credential within our Jenkins UI. Copy the PAT to clipboard which we generated. (Copy it to a notepad for reference) On our Jenkins UI go to “Manage Jenkins” => “Manage Credentials “



Next click on “Jenkins” as shown below. And click “Global credentials (unrestricted)”


Dashboard
Credentials
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Jenkins (global) son
Jenkins (global) doc
Jenkins (global) a93
Jenkins (global) doc
Jenkins (global) doc
Jenkins (global) doc
Jenkins (global) udi
Jenkins (global) mo
Jenkins (global) dev
Jenkins (global) gith
Jenkins (global) son
Icon: S M L
Stores scoped to Jenkins
P
Store
Domains
Jenkins
(global)

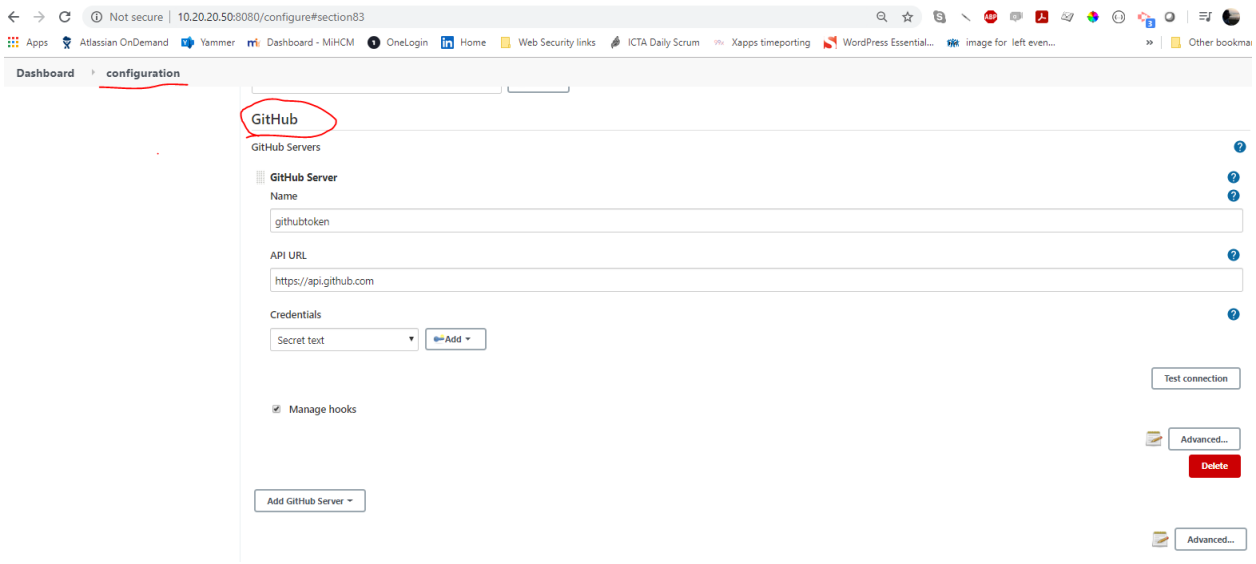

Jenkins
Dashboard
Credentials
System
New Item
People
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
System
Domain
Global credentials (unrestricted)
Icon: S M L



Click on "Add Credentials" and select "Secret text" for kind. Paste our Github PAT to the Secret textfield. Leave the ID field empty. (as it will be auto-generated). Give some description to easily identify our credential. Click OK.

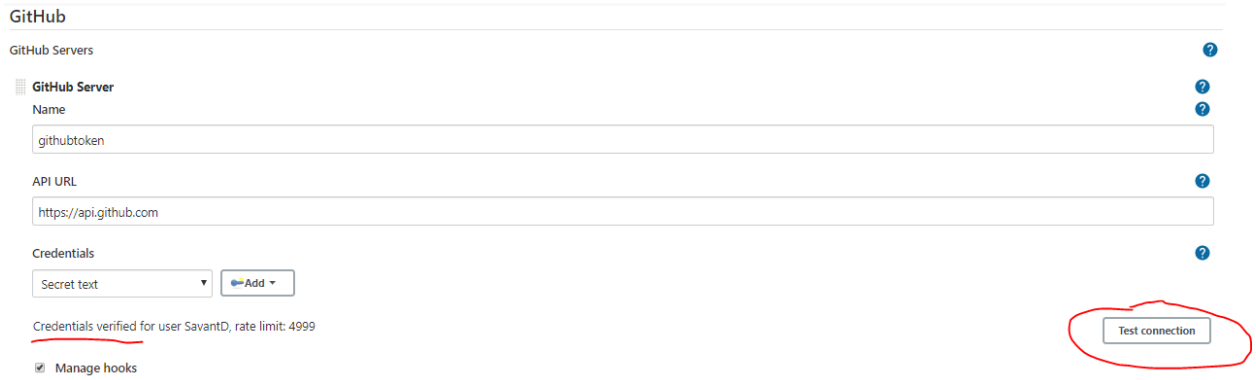
Click on 'Manage Jenkins' => 'Configure System' icon. This should be the top icon in the list. The resulting configuration menu will feature many options. We need to enable the github webhook for our jenkins to communicate with Github. So scroll down to the GitHub section of the menu.(As shown below) If a GitHub section is not present, ensure that the GitHub plugin is installed (Add the plugin through 'Manage Jenkins' => 'Manage Plugins' type 'github' in the available section https://plugins.jenkins.io/github/).

Note: as mentioned here, We have followed the Automatic Mode (where Jenkins manages hooks for jobs by itself) . To setup the Manual mode, we will need our jenkins ec2's public ip 3.6.93.46:8080 to be accessible globally via internet.



In the GitHub section, Under the GitHub Servers area provide below details for our github server.

Name - a preferred name,
API URL - https://api.github.com
Credentials - select the credential we created previously from the dropdown
Also check the **Manage hooks** checkbox.
Now Click 'test connection' to confirm hook's connection as depicted below.



That's it.. Done..

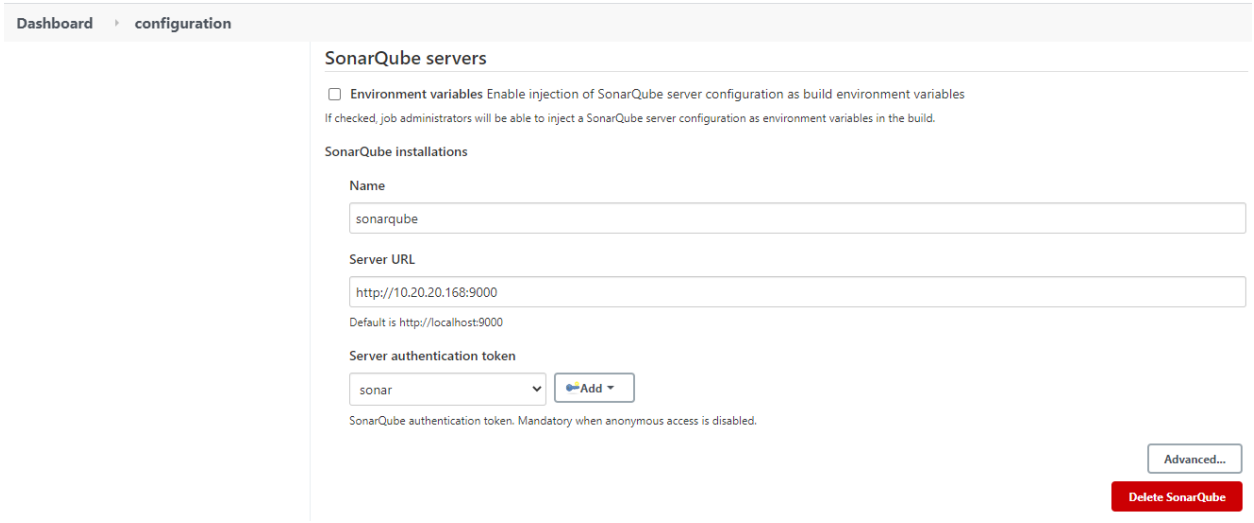
Jenkins uninstallation steps

sudo service jenkins stop
sudo yum clean all
sudo yum -y remove jenkins
sudo rm -rf /var/cache/jenkins
sudo rm -rf /var/lib/jenkins/

Additional settings

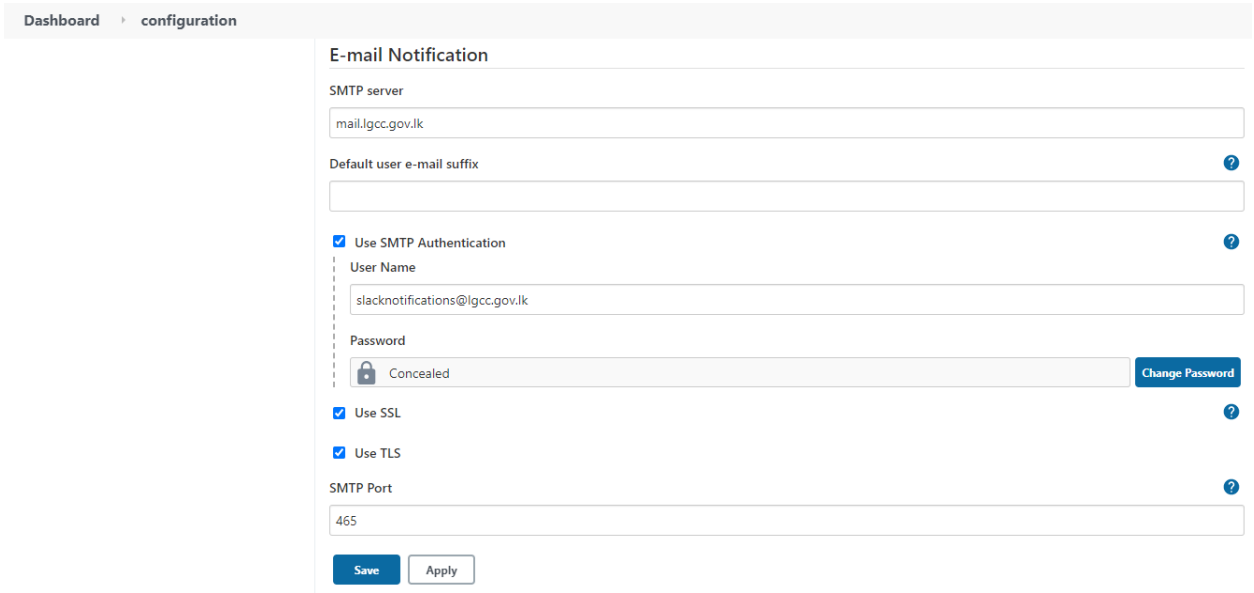
Add Sonarqube url to jenkins

Go to Dashboard -> Configuration -> scroll down to SonarQube servers



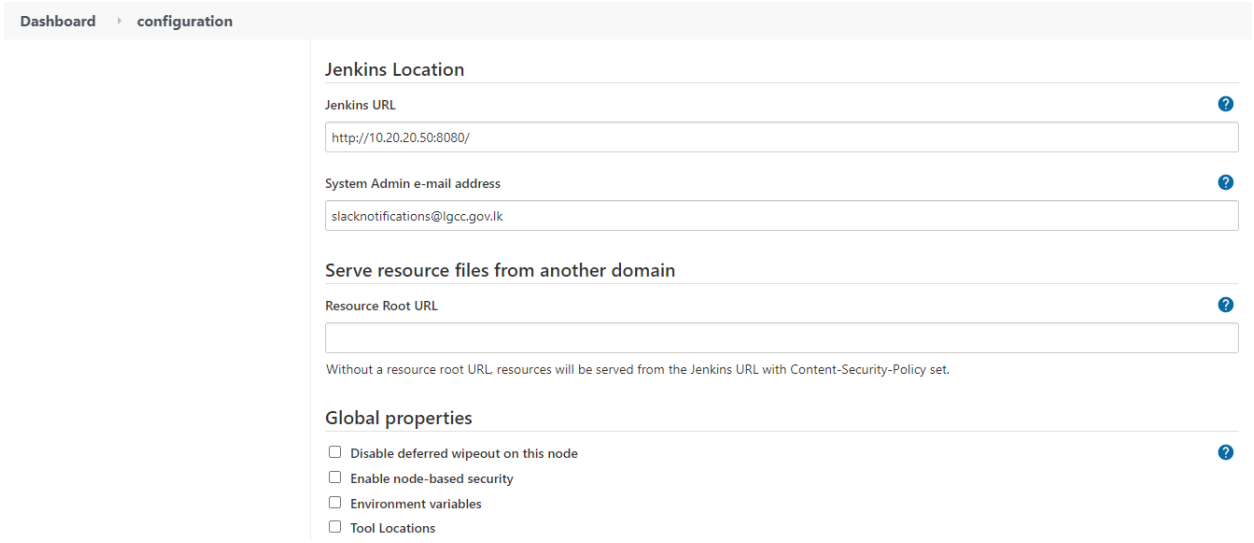
Configure the Email notification to jenkins

Go to Dashboard -> Configuration -> scroll down to Email Notification



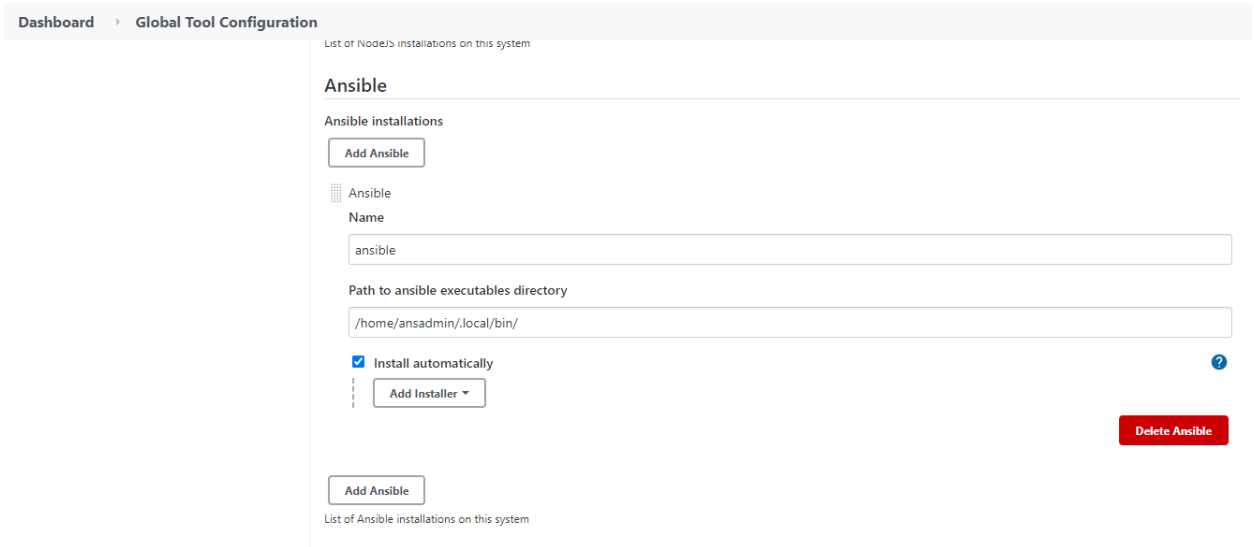
Configure the Additional Configuration

Go to Dashboard -> Configuration -> scroll down to Jenkins Location



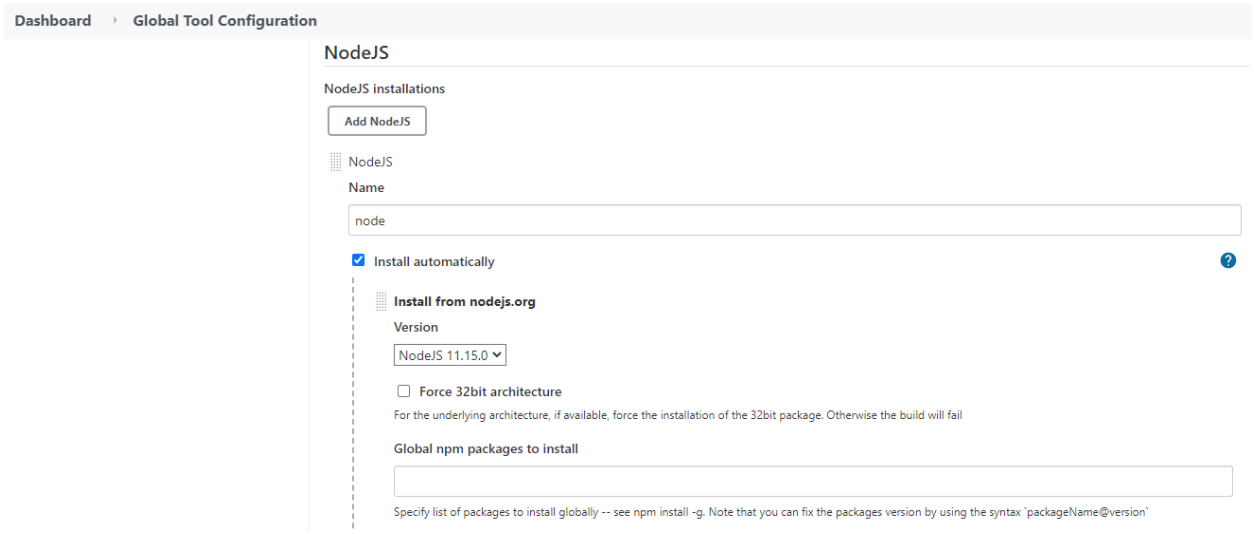
Configure the Ansible

Go to Dashboard -> Configuration -> scroll down to Ansible
(This need to install ansible on jenkins server is a must.)



Configure the Node JS

Go to Dashboard -> Configuration -> scroll down to NodeJS
(This is not required to install nodeJS on the server..)



SonarQube Installation and Configuration

Server Details - Sonar

- IP address public: 13.126.65.117
- IP address private: 10.20.20.168
- Port: 9000
- Accessible URL: http://<ipaddress>:<port> (example: http://13.126.65.117:9000)
- un: admin pass: SLudi2021Icta

Server Specifications

- Hosted service: Amazon Web Services (AWS)
- OS: CentOS 7
- Server username: centos
- Hosted Server Spec Name: t2.medium (General Purpose)
- Hosted server spec: 2 Core, 4GB Memory, 20GB Storage

Running services

PostgreSQL 10.16

- Start service: systemctl start postgresql-10.service
- Stop service: systemctl stop postgresql-10.service
- Check status: systemctl status postgresql-10.service
- Path: /var/lib/pgsql/10

SonarQube 7.9 (LTS)

- Start service: systemctl start sonar.service
- Stop service: systemctl stop sonar.service
- Check status: systemctl status sonar.service
- JVM allocation 512m
- Path: /opt/sonarqube/

ElasticSearch

- Bind with SonarQube
- JVM allocation 1024m
- Path: /opt/sonarqube/elasticsearch/

Java OpenJDK 11.0.10 (LTS)

- OpenJDK 64-Bit Server VM
- Path: /usr/lib/jvm/java-11-openjdk-11.0.10.0.9-1.el7_9.x86_64

Installation:

https://docs.sonarqube.org/7.9/requirements/requirements/

https://docs.sonarqube.org/7.9/setup/install-server/

Integrate with Jenkins:

Token Generate.

1. Login to Sonar server
2. Go to Administrator -> Security -> Users -> Tokens -> Generate Tokens
3. Then enter token name click Generate button.
4. Copy the generated token.

Add Sonar Token to Jenkins

1. Login to Jenkins server
2. Go to Manage Jenkins -> Configure System -> SonarQube servers
3. Then enter sonarquber server url (example http://localhost:9000)
4. Then add copied sonar token to Server authentication token.

Add Web Hook to Sonar Qulity Gate

1. Login to Sonar server
2. Go to Administrator -> Configuration -> Webhooks
3. Then create new webhook as below format (http://<jenkins-server-ip>:<port>/sonarqube-webhook/)

References

1. https://arkit.co.in/jenkins-installation-on-rhel-7/
2. https://www.tecmint.com/fix-firewall-cmd-command-not-found-error/
3. https://gist.github.com/springaki/1add83b255c2d38800e5c3d864cbb737
4. https://www.jenkins.io/doc/book/installing/linux/#red-hat-centos
5. https://www.tecmint.com/install-git-centos-fedora-redhat/
6. https://tecadmin.net/install-apache-maven-on-centos/
7. https://www.tecmint.com/install-nodejs-npm-in-centos-ubuntu/
8. https://www.jenkins.io/doc/book/system-administration/security/