**JAVA 8 installation in ubuntu 16.04**

* sudo apt-get -y install default-jdk

**Java 11 ( open JDK )**

* sudo add-apt-repository ppa:openjdk-r/ppa
* sudo apt-get update
* sudo apt install openjdk-11-jdk

**JAVA 11 installation in ubuntu 16.04**

* sudo add-apt-repository ppa:linuxuprising/java
* sudo apt update
* sudo apt-get install oracle-java11-installer
* sudo apt-get install oracle-java11-set-default ( to set java 11 as default )
* java -version ( verify java installation )

for more details - <https://tecadmin.net/install-oracle-java-11-on-ubuntu-16-04-xenial/>

**JAVA 11 installation in GCP machine**

* download **jdk-11.0.12\_linux-x64\_bin.tar.gz** from <https://www.oracle.com/java/technologies/downloads/#java11> and place it under **/var/cache/oracle-jdk11-installer-local**
* sudo apt-get install oracle-java11-installer-local
* sudo apt install oracle-java11-set-default-local

**git installation in ubuntu 16.04**

* apt-get update
* apt-get install git
* git --version ( to verify git version )

**Jenkins installation in ubuntu 16.04**

* wget <https://updates.jenkins-ci.org/download/war/2.162/jenkins.war> ( installs 2.162 version, if you want any other version to be installed visit <https://updates.jenkins-ci.org/download/war/> download particular version )
* java -jar jenkins.war ( default runs on 8080 port )
* java -jar jenkins.war --httpPort=5000 ( if you want run on any other port use this, in my case its 5000 port )
* nohup java -jar jenkins.jar & ( to run jenkins process in background )

**Jenkins as a service**

* wget -q -O - <https://pkg.jenkins.io/debian/jenkins-ci.org.key> | sudo apt-key add -
* echo deb <https://pkg.jenkins.io/debian-stable> binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list
* apt-get update
* apt-get install jenkins
* systemctl start jenkins
* systemctl status jenkins

**maven installation in ubuntu 16.04**

* cd /usr/local
* wget <https://downloads.apache.org/maven/maven-3/3.6.3/binaries/apache-maven-3.6.3-bin.tar.gz>
* sudo tar xvf apache-maven-3.6.3-bin.tar.gz
* ln -s apache-maven-3.6.3 apache-maven
* sudo vi /etc/profile.d/apache-maven.sh
* please insert below lines to /etc/profile.d/apache-maven.sh
* export JAVA\_HOME=/usr/lib/jvm/java-11-oracle
* export M2\_HOME=/usr/local/apache-maven
* export MAVEN\_HOME=/usr/local/apache-maven
* export PATH=${M2\_HOME}/bin:${PATH}
* source /etc/profile.d/apache-maven.sh
* mvn -version ( to verify maven version )

For more details <https://tecadmin.net/install-apache-maven-on-ubuntu/>

easy way to install - sudo apt install maven

**Docker installation in ubuntu 16.04**

* sudo apt-get update
* sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
* curl -fsSL <https://download.docker.com/linux/ubuntu/gpg> | sudo apt-key add -
* sudo add-apt-repository "deb [arch=amd64] <https://download.docker.com/linux/ubuntu> $(lsb\_release -cs) stable"
* sudo apt-key fingerprint 0EBFCD88
* sudo apt-get update
* sudo apt-get install docker-ce docker-ce-cli containerd.io ( to install latest version )
* sudo docker run hello-world

if you want all the things in a script use <https://github.com/DeekshithSN/cheatsheet/blob/master/docker-install.sh>

for more details refer <https://docs.docker.com/install/linux/docker-ce/ubuntu/#install-docker-ce-1>

If you face below problem which is same as mentione below the you can refer this link

https://stackoverflow.com/questions/25183063/docker-on-rhel-6-cgroup-mounting-failing

Starting cgconfig service: Error: cannot create directory /cgroup/blkio

/sbin/cgconfigparser; error loading /etc/cgconfig.conf: Cgroup, operation not allowed

Failed to parse /etc/cgconfig.conf [FAILED]

After doing this you need run all the commands with sudo concatenated to it.

- To solve this issue we need to add current user to docker group , to do the same follow the below commands

sudo groupadd docker

sudo usermod -aG docker $USER ( got a error while runing docker commands with the current user)

sudo usermod -aG docker jenkins (got a error while runing docker commands with jenkins user )

getent group <groupname> (to check the list of users in particular group)

sudo passwd jenkins to change password of jenkins user

- even after following the above commands if you face any issue in ruuning commands then run below command

chmod 777 /var/run/docker.sock

**SonarQube Installation in ubuntu 16.04**

Use apt-get to install the required packages.

* apt-get update
* apt-get install unzip software-properties-common wget default-jdk

Install the PostgreSQL database service.

* apt-get install postgresql postgresql-contrib

Access the Postgres database service command-line.

* su - postgres
* psql

Create a Postgres user named sonarqube,Create a Postgres database named sonarqube. Give the PostgreSQL user named sonarqube permission over the database named sonarqube

* CREATE USER sonarqube WITH PASSWORD 'password';
* CREATE DATABASE sonarqube OWNER sonarqube;
* GRANT ALL PRIVILEGES ON DATABASE sonarqube TO sonarqube;
* \q

Download the Sonarqube package and move it to the OPT directory.

* mkdir /downloads/sonarqube -p
* cd /downloads/sonarqube
* wget <https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-7.9.1.zip>
* unzip sonarqube-7.9.1.zip
* mv sonarqube-7.9.1 /opt/sonarqube

Create a new Linux account named sonarqube, Set the correct file permission on the sonarqube directory.

* adduser --system --no-create-home --group --disabled-login sonarqube
* chown -R sonarqube:sonarqube /opt/sonarqube

Edit the sonar.sh configuration file.

* vi /opt/sonarqube/bin/linux-x86-64/sonar.sh

Configure the following options:

* RUN\_AS\_USER=sonarqube

Edit the sonar.properties configuration file.

* vi /opt/sonarqube/conf/sonar.properties

Configure the following options:

sonar.jdbc.username=sonarqube

sonar.jdbc.password=password

sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube

sonar.web.javaAdditionalOpts=-server

sonar.web.host=0.0.0.0

Create a Linux configuration file named 99-sonarqube.conf

* vi /etc/security/limits.d/99-sonarqube.conf

Here is the content of the 99-sonarqube.conf file.

sonarqube - nofile 65536

sonarqube - nproc 4096

Edit the sysctl.conf configuration file.

* vi /etc/sysctl.conf

Add the following lines at the end of the sysctl.conf file.

vm.max\_map\_count=262144

fs.file-max=65536

Reboot your computer to enable the new configuration

* reboot

Start the Sonarqube service.

* /opt/sonarqube/bin/linux-x86-64/sonar.sh start

Use the following command to monitor the SonarQube log.

* tail -f /opt/sonarqube/logs/sonar.log

for more deatils refer - <https://techexpert.tips/sonarqube/sonarqube-installation-ubuntu-linux/>

**you want to create sonarqube through docker then use below command**

docker run -d -p 9000:9000 sonarqube:lts

**JFROG Artifactory installation in ubuntu 16.04 using docker container**

* export JFROG\_HOME= /set/some/path
* mkdir -p $JFROG\_HOME/artifactory/var/etc/
* cd $JFROG\_HOME/artifactory/var/etc/
* touch ./system.yaml
* chown -R 1030:1030 $JFROG\_HOME/artifactory/var

run below docker command

docker run --name artifactory -v $JFROG\_HOME/artifactory/var/:/var/opt/jfrog/artifactory -d -p 8081:8081 -p 8082:8082 docker.bintray.io/jfrog/artifactory-oss:latest

**Ansible Installation in ubuntu 16.04**

* sudo apt-add-repository ppa:ansible/ansible
* sudo apt-get update
* sudo apt-get install ansible

For more details - <https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-16-04>

**kubernetes installation on ubuntu 16.04 ( kubeadm - single master multi nodes )**

Make sure docker installed in master and nodes, make sure master has 2 cpu's

Execute below commands in both master and node

* sudo apt-get update && sudo apt-get install -y apt-transport-https curl
* curl -s <https://packages.cloud.google.com/apt/doc/apt-key.gpg> | sudo apt-key add -
* cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list  
  deb <https://apt.kubernetes.io/> kubernetes-xenial main  
  EOF
* sudo apt-get update
* sudo apt-get install -y kubelet kubeadm kubectl
* sudo apt-mark hold kubelet kubeadm kubectl

for more details <https://kubernetes.io/docs/setup/independent/install-kubeadm/>

Execute below commands in master

after executing this command you will get node's joining command, copy and paste it somewhere

* kubeadm init --pod-network-cidr=10.244.0.0/16 ( if you have forget to do then use kubeadm token create --print-join-command )
* export KUBECONFIG=/etc/kubernetes/admin.conf
* mkdir -p $HOME/.kube
* sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
* sudo chown $(id -u):$(id -g) $HOME/.kube/config;mkdir -p $HOME/.kube
* sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
* kubectl apply -f <https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/canal/rbac.yaml>
* kubectl apply -f <https://raw.githubusercontent.com/coreos/flannel/2140ac876ef134e0ed5af15c65e414cf26827915/Documentation/kube-flannel.yml>

Execute join command in node, which may look like as mentioned below

kubeadm join 10.128.0.8:6443 --token q915fe.do2ty6a8ow6qjixt \

--discovery-token-ca-cert-hash sha256:acd137106e6b763d1ca6b5a4f7c1b1538c2ee8af81e47f9ea3f385c66cd710b3

Then to verify use below commands

* kubectl get nodes
* kubectl get pods
* Kubectl get service

**Nexus Installation in ubuntu 16.04**

* apt-get install wget ( install if you dont have wget )
* java -version ( make sure java is installed which should be java 8 or higher version )
* wget <https://download.sonatype.com/nexus/3/latest-unix.tar.gz>
* tar -xvf latest-unix.tar.gz
* cd nexus-3.35.0-02/bin
* ./nexus start ( starts the nexus artifactory )
* ./nexus status ( by this you check the status of nexus artifactory )
* To access this use http://ip\_Address:8081 ( by deafault which will be running on 8081)

intial password will be present in /opt/sonatype-work/nexus3/admin.password

**Helm Installation in ubuntu 16.04**

* curl -fsSL -o get\_helm.sh <https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3>
* chmod 700 get\_helm.sh
* ./get\_helm.sh

**helm uninstallation**

* + which helm ( to see which folder its installed )
  + rm -rf /usr/local/bin/helm

**installing kubeadm on GCP machines**

**Install kubelet, kubeadm and kubectl**

add Kubernetes repository for Ubuntu 20.04 to all the servers.

sudo apt update

sudo apt -y install curl apt-transport-https

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

Then install required packages.

sudo apt update

sudo apt -y install vim git curl wget kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

Confirm installation by checking the version of kubectl.

kubectl version --client && kubeadm version

Client Version: version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.2", GitCommit:"8b5a19147530eaac9476b0ab82980b4088bbc1b2", GitTreeState:"clean", BuildDate:"2021-09-15T21:38:50Z", GoVersion:"go1.16.8", Compiler:"gc", Platform:"linux/amd64"}

kubeadm version: &version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.2", GitCommit:"8b5a19147530eaac9476b0ab82980b4088bbc1b2", GitTreeState:"clean", BuildDate:"2021-09-15T21:37:34Z", GoVersion:"go1.16.8", Compiler:"gc", Platform:"linux/amd64"}

**Disable Swap**

Turn off swap.

sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

sudo swapoff -a

Enable kernel modules and configure sysctl.

Enable kernel modules

sudo modprobe overlay

sudo modprobe br\_netfilter

Add some settings to sysctl

sudo tee /etc/sysctl.d/kubernetes.conf<<EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

EOF

Reload sysctl

sudo sysctl --system

**Install Container runtime**

**Installing Docker runtime:**

Add repo and Install packages

sudo apt update

sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

sudo apt update

sudo apt install -y containerd.io docker-ce docker-ce-cli

**Create required directories**

sudo mkdir -p /etc/systemd/system/docker.service.d

**Create daemon json config file**

sudo tee /etc/docker/daemon.json <<EOF

{

"exec-opts": ["native.cgroupdriver=systemd"],

"log-driver": "json-file",

"log-opts": {

"max-size": "100m"

},

"storage-driver": "overlay2"

}

EOF

**Start and enable Services**

sudo systemctl daemon-reload

sudo systemctl restart docker

sudo systemctl enable docker

**Ensure you load modules**

sudo modprobe overlay

sudo modprobe br\_netfilter

**Set up required sysctl params**

sudo tee /etc/sysctl.d/kubernetes.conf<<EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

EOF

**Reload sysctl**

sudo sysctl --system

**Initialize master node**

Login to the server to be used as master and make sure that the br\_netfilter module is loaded:

lsmod | grep br\_netfilter

Enable kubelet service.

sudo systemctl enable kubelet

Initialize kubeadm

kubeadm init

Configure kubectl using commands in the output:

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

Additional nodes can be added using the command in installation output:

kubeadm join k8s-cluster.computingforgeeks.com:6443 --token sr4l2l.2kvot0pfalh5o4ik \

--discovery-token-ca-cert-hash sha256:c692fb047e15883b575bd6710779dc2c5af8073f7cab460abd181fd3ddb29a18 \

--control-plane

**Install network plugin on Master**

In this we’ll use Calico. You can choose any other supported network plugins.

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml