14 Steps to Install kubernetes on Ubuntu 20.04(bento/ubuntu-20.04), 18.04(hashicorp/bionic64)

Apr 24, 2020 · 8 min read · KUBERNETES · Share on:

(Note : - This article has been updated for installing kubernetes on ubuntu 20.04)

In this article on **How to Install kubernetes on Ubuntu 20.04 and18.04** we are going to setup Kubernetes cluster along with kubeadm on Ubuntu 20.04 (The same steps can be used for the Ubuntu 18.04 as well as 16.04)

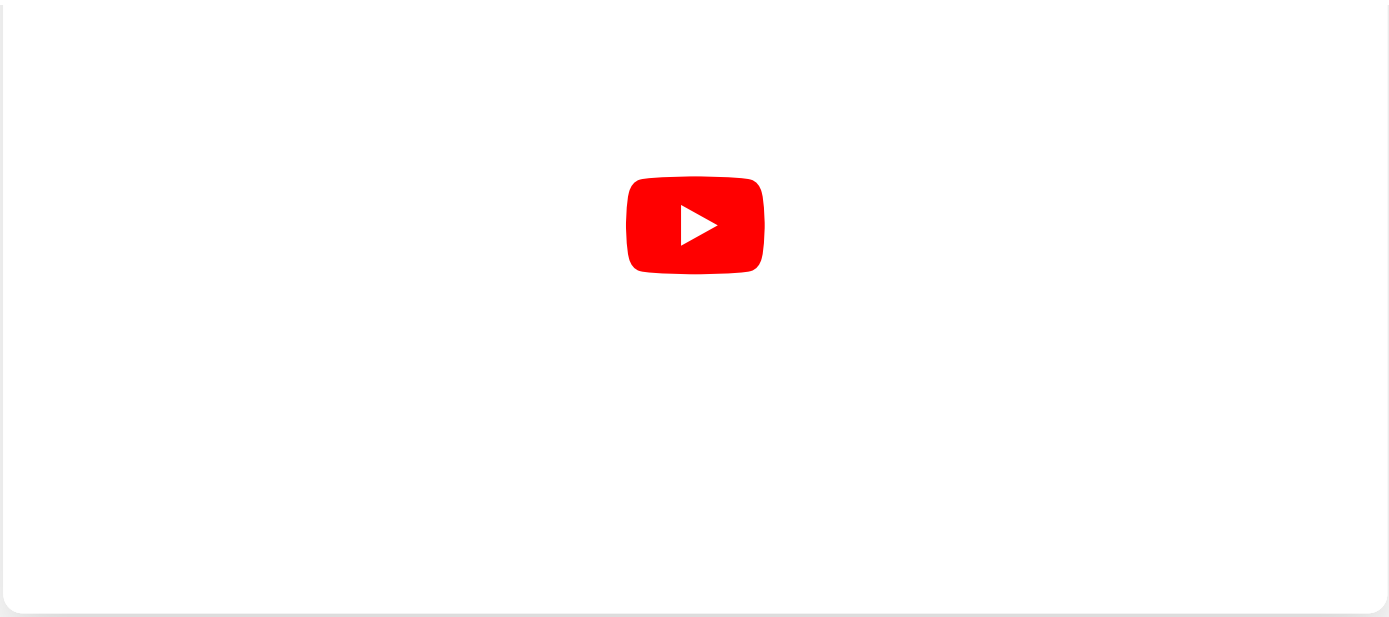
We will start setting up everything from the scratch on virtual machine. Since we are setting up kubernetes cluster, so we will be provisioning two virtual machine - `kubernetes master(kmaster)` and `kubernetes kworker(kworker)`. The only pre-requisite which you need to do is to install ***Vagrant*** and ***Virtual Box*** before.

We are going to setup two nodes for kubernetes cluster

1. Master node
2. Worker node

Prerequisites

1. Reading time is about 20 minutes
2. Vagrant 2.2.15 or latest - For installation instruction click here
3. VM VirtualBox - For installation instruction click here



Step 1 - Start your vagrant box

As a minimum requirement for kubernetes installation we need -

1. Master Node - 2 cpus, 2 GB Memory
2. Worker Node - 1 cpu, 1 GB Memory

Use following Vagrantfile or at least create a `Vagrantfile` and copy the following configuration into it -

BASH

```
Vagrant.configure("2") do |config|
  config.vm.define "master" do |master|
    master.vm.box_download_insecure = true
    master.vm.box = "bento/ubuntu-20.04"        ## For ubuntu 18.04 use - hashic
    master.vm.network "private_network", ip: "100.0.0.1"
    master.vm.hostname = "master"
    master.vm.provider "virtualbox" do |v|
      v.name = "master"
      v.memory = 2048
```

```
  config.vm.define "worker" do |worker|
    worker.vm.box_download_insecure = true
    worker.vm.box = "bento/ubuntu-20.04"        ## For ubuntu 18.04 use - hashic
    worker.vm.network "private_network", ip: "100.0.0.2"
    worker.vm.hostname = "worker"
    worker.vm.provider "virtualbox" do |v|
      v.name = "worker"
      v.memory = 1024
      v.cpus = 1
    end
  end

end
```

Start you virtual boxes by starting up your vagrant box -

BASH

```
vagrant up
```

***Note - If you are interested in setting the kubernetes cluster on Ubuntu 18.04 then replace `worker.vm.box = "bento/ubuntu-20.04"` with `worker.vm.box = "hashicorp/bionic64"` ***

Step 2 - Update host files on both master and worker node

After starting the vagrant box now we need to login into the virtual machine using the command `vagrant ssh master`

BASH

```
vagrant ssh master
```

Add host entry for master as well as worker node

BASH

```
sudo vi /etc/hosts
```

BASH

```
100.0.0.1 master.jhooq.com master
100.0.0.2 worker.jhooq.com worker
```

worker node - SSH into the master node

BASH

```
vagrant ssh worker
```

Add host entry for master as well as worker node

BASH

```
sudo vi /etc/hosts
```

BASH

```
100.0.0.1 master.jhooq.com master
100.0.0.2 worker.jhooq.com worker
```

BASH

```
ping worker
```

BASH

```
PING worker.jhooq.com (100.0.0.2) 56(84) bytes of data.
64 bytes from worker.jhooq.com (100.0.0.2): icmp_seq=1 ttl=64 time=0.462 ms
64 bytes from worker.jhooq.com (100.0.0.2): icmp_seq=2 ttl=64 time=0.686 ms
```

Test the master node by sending from worker

BASH

```
ping master
PING master.jhooq.com (100.0.0.1) 56(84) bytes of data.
64 bytes from master.jhooq.com (100.0.0.1): icmp_seq=1 ttl=64 time=0.238 ms
64 bytes from master.jhooq.com (100.0.0.1): icmp_seq=2 ttl=64 time=0.510 ms
```

Step 3 - Install Docker on both master and worker node

You need to install Docker on both the node.

So run the following installation command on both the nodes

BASH

```
sudo apt-get update
```

Enable and start docker

BASH

```
sudo systemctl enable docker
```

BASH

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /li
```

BASH

```
sudo systemctl start  docker
```

Check the docker service status

BASH

```
sudo systemctl status docker
```

BASH

```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: e
   Active: active (running) since Thu 2020-04-23 19:10:59 UTC; 4s ago
     Docs: https://docs.docker.com
```

"swapping"

We need to disable firewall as well as swapping on master as well as worker node. Because to install kubernetes we need to disable the swapping on both the nodes

BASH

```
sudo ufw disable
```

BASH

```
Firewall stopped and disabled on system startup
```

BASH

```
sudo swapoff -a
```

Step 5 - Install "apt-transport-https" package

To download the kubernetes and its public we need to install "apt-transport-https" package on both master as well as worker node

BASH

```
sudo apt-get update && sudo apt-get install -y apt-transport-https
```

Step 6 - Download the public keys

So run the following command to get the public keys on both master as well as worker node

BASH

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

BASH

```
OK
```

Step 7 - Add kubernetes repo

As a next step we need to add the kubernetes repo to both master as well as worker node

BASH

```
sudo bash -c 'echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" > /etc/
```

Step 8 - Install kubernetes

Now after adding the kubernetes repo we need to install the kubernetes on both mater as well as worker node

BASH

```
sudo apt-get update && sudo apt-get install -y kubelet kubeadm kubectl
```

Step 9 - Enable and Start Kubelet

Alright now we have installed the kubernetes, now we need to enable the kubelet support for both master as well worker node

BASH

```
sudo systemctl enable kubelet
```

BASH

```
sudo systemctl start kubelet
```

Step 10 - Initialize the kubernetes cluster

Okay now we have reach to point where we have done all the prerequisite for initializing the kubernetes cluster.

Let's run the kubernetes initialization command on only on master

BASH

```
sudo kubeadm init --apiserver-advertise-address=100.0.0.1
```

Note down kubeadm join command which we are going to use from worker node to join the master node using token. (Note : - Followig command will be different for you, do not try copy the following command)

BASH

```
sudo kubeadm join 100.0.0.1:6443 --token g2bsw7.5xr3bqc21eqyc6r7 --discovery-tok
```

Step 11 - Move kube config file to current user (only run on master)

To interact with the kubernetes cluster and to user kubectl command, we need to have the kube config file with us.

Use the following command to get the kube config file and put it under working directory.

BASH

```
mkdir -p $HOME/.kube
```

BASH

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

BASH

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

Step 12 - Apply CNI from kube-flannel.yml(only run on master)

After the master of the cluster is ready to handle jobs and the services are running, for the purpose of making containers accessible to each other through networking, we need to set up the network for container communication.

Get the CNI(container network interface) configuration from flannel

Step 13 - Join worker nodes to master(only run on worker)

In the **Step 10** we generated the token and kubeadm join command.

Now we need to use that join command from our worker node

BASH

```
sudo kubeadm join 100.0.0.1:6443 --token g2bsw7.5xr3bqc21eqyc6r7     --discovery
```

BASH

```
W0423 19:27:00.344480   18268 join.go:346] preflight] WARNING: JoinControlPane.c
preflight] Running pre-flight checks
        WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup
preflight] Reading configuration from the cluster...
preflight] FYI: You can look at this config file with 'kubectl -n kube-system ge
kubelet-start] Downloading configuration for the kubelet from the "kubelet-confi
kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.ya
kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kub
kubelet-start] Starting the kubelet
kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

master)

To check the status of the nodes use

BASH

```
kubectl get nodes
```

BASH

```
NAME     STATUS   ROLES    AGE   VERSION
master   Ready    master   26m   v1.18.2
worker   Ready    <none>   63s   v1.18.2
```

(Note - In case you see the status of the nodes `not ready` then I would recommend to check the these troubleshooting step as well as flannel settings)

Summary

So this was our beginner tutorial which involves around - 14 Steps to Install kubernetes on Ubuntu 18.04 and 16.04.

With that we can conclude what we have achieved -

- Congratulations we have successfully installed kubernetes on Ubuntu 18.04
- After installing kubernetes we are successfully join the worker nodes with master.

Troubleshooting kube-flannel.yml

Look out for the Ethernet i.e. eth1 which has a ip address 100.0.0.1(this is the ip address which we used in vagrant file)

BASH

```
ip a s
```

BASH

```
1: lo: <LOOPBACK,UP,LOWER_UP>
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP grou
    link/ether 08:00:27:bb:14:75 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP grou
    link/ether 08:00:27:fb:48:77 brd ff:ff:ff:ff:ff:ff
    inet 100.0.0.1
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP>
```

Now we need to add the extra args for eth1 in kube-flannel.yml

BASH

```
vi kube-flannel.yml
```

Searche for - "flanneld"

In the args section add : - -iface=eth1

BASH

```
- --iface=eth1
        args:
        - --ip-masq
        - --kube-subnet-mgr
        - --iface=eth1
```

BASH

```
kubectl apply -f kube-flannel.yml
```

BASH

```
podsecuritypolicy.policy/psp.flannel.unprivileged created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds-amd64 created
daemonset.apps/kube-flannel-ds-arm64 created
daemonset.apps/kube-flannel-ds-arm created
daemonset.apps/kube-flannel-ds-ppc64le created
daemonset.apps/kube-flannel-ds-s390x created
```

Troubleshooting Error

How to fix – [ERROR Swap]: running with swap on is not supported. Please disable swap .

> **Learn more On Kubernetes -**
>
> 1. Setup kubernetes on Ubuntu
> 2. Setup Kubernetes on CentOs
> 3. Setup HA Kubernetes Cluster with Kubespray
> 4. Setup HA Kubernetes with Minikube
> 5. Setup Kubernetes Dashboard for local kubernetes cluster

8. Deploy Spring Boot Microservice on local Kubernetes cluster
9. Deploy Spring Boot Microservice on Cloud Platform(GCP)
10. Setting up Ingress controller NGINX along with HAproxy inside Kubernetes cluster
11. CI/CD Kubernetes | Setting up CI/CD Jenkins pipeline for kubernetes
12. kubectl export YAML | Get YAML for deployed kubernetes resources(service, deployment, PV, PVC....)
13. How to setup kubernetes jenkins pipeline on AWS?
14. Implementing Kubernetes liveness, Readiness and Startup probes with Spring Boot Microservice Application?
15. How to fix kubernetes pods getting recreated?
16. How to delete all kubernetes PODS?
17. How to use Kubernetes secrets?
18. Share kubernetes secrets between namespaces?
19. How to Delete PV(Persistent Volume) and PVC(Persistent Volume Claim) stuck in terminating state?
20. Delete Kubernetes POD stuck in terminating state?

Posts in this Series

- Kubernetes Cheat Sheet for day to day DevOps operations?
- Delete Kubernetes POD stuck in terminating state?
- How to Delete PV(Persistent Volume) and PVC(Persistent Volume Claim) stuck in terminating state?
- Share kubernetes secrets between namespaces?
- How to use Kubernetes secrets?
- How to delete all kubernetes PODS?
- kubernetes pods getting recreated?

- kubectl export yaml OR How to generate YAML for deployed kubernetes resources
- Kubernetes Updates
- CI/CD Kubernetes | Setting up CI/CD Jenkins pipeline for kubernetes
- Kubernetes cluster setup with Jenkins
- How to use Persistent Volume and Persistent Claims | Kubernetes
- How to fix ProvisioningFailed persistentvolume controller no volume plugin matched
- Fixing – Cannot bind to requested volume: storageClasseName does not match
- Fixing – pod has unbound immediate persistentvolumeclaims or cannot bind to requested volume incompatible accessmode
- How to fix kubernetes dashboard forbidden 403 error – message services https kubernetes-dashboard is forbidden User
- How to fix Kubernetes – error execution phase preflight [preflight]
- Deploy Spring Boot microservices on kubernetes?
- How to fix – ansible_memtotal_mb minimal_master_memory_mb
- How to use kubespray – 12 Steps for Installing a Production Ready Kubernetes Cluster
- How to setup kubernetes on CentOS 8 and CentOS 7
- How to fix – How to fix - ERROR Swap running with swap on is not supported. Please disable swap
- 14 Steps to Install kubernetes on Ubuntu 20.04(bento/ubuntu-20.04), 18.04(hashicorp/bionic64)
- Kubernetes Dashboard | Kubernetes Admin GUI | Kubernetes Desktop Client
- Install Kubernetes with Minikube

Search...

Categories

AWS 6 BLOGGING 6 SPRING-BOOT 5 SSL 5 QUARKUS 4

GITHUB 3 KUBESPRAY 3 PROMETHEUS-GRAFANA 3 VAGRANT 3

NGINX 2

ALL CATEGORIES

Series

TERRAFORM 34 KUBERNETES 27 DOCKER 22 HELM-CHART 11

AWS 1 LINUX-COMMANDS 1

Tags

KUBERNETES 18 HELM-CHART 10 BLOGGING 4 QUARKUS 4

DOCKER 3 GITHUB 3 SSL 3 KUBESPRAY 2

SPRING-BOOT 2 HADOOP 1 INDEX 1 NGINX 1

TERRAFORM 1

- How to fix - AWS ECR docker login error storing credentials - err exit status 1?
- Terraform Guide - Docker Containers & AWS ECR(elastic container registry)?
- How to generate SSH key in Terraform using tls_private_key?
- How to fix-Terraform Error acquiring the state lock ConditionalCheckFiledException?
- Terraform Template - A complete guide?
- Fixing terraform x509 certificate signed by unknown authority?
- Learn $ grep command with 15 examples
- AWS S3 - How to host static website?

Rahul Wagh

Its all about Open Source and DevOps, here I talk about Kubernetes, Docker, Java, Spring boot and practices.

READ MORE

Copyright 2022 JHOOQ. All Rights Reserved