

How to Install Kubernetes Cluster on Ubuntu 22.04 (Step-by-Step Guide)

Introduction



Hakan Bayraktar · [Follow](#)

6 min read · Nov 3, 2023



Listen



Share

Kubernetes is a powerful container orchestration platform used for automating the deployment, scaling, and management of containerized applications. In this guide, we will walk you through the step-by-step process of installing Kubernetes on Ubuntu 22.04. This cluster configuration includes a master node and worker nodes, allowing you to harness the full power of Kubernetes.

Kubernetes Nodes

In a Kubernetes cluster, you will encounter two distinct categories of nodes:

Master Nodes: These nodes play a crucial role in managing the control API calls for various components within the Kubernetes cluster. This includes overseeing pods, replication controllers, services, nodes, and more.

Worker Nodes: Worker nodes are responsible for providing runtime environments for containers. It's worth noting that a group of container pods can extend across multiple worker nodes, ensuring optimal resource allocation and management.

Prerequisites

Before diving into the installation, ensure that your environment meets the following prerequisites:

- An Ubuntu 22.04 system.
- Privileged access to the system (root or sudo user).
- Active internet connection.

- Minimum 2GB RAM or more.
- Minimum 2 CPU cores (or 2 vCPUs).
- 20 GB of free disk space on /var (or more).

Step 1: Update and Upgrade Ubuntu (all nodes)

Begin by ensuring that your system is up to date. Open a terminal and execute the following commands:

```
sudo apt update
sudo apt upgrade
```

Step 2: Disable Swap (all nodes)

To enhance Kubernetes performance, disable swap and set essential kernel parameters. Run the following commands on all nodes to disable all swaps:

```
sudo swapoff -a
sudo sed -i 's/^(\s*)swap\s*/\1# \1/g' /etc/fstab
```

Step 3: Add Kernel Parameters (all nodes)

Load the required kernel modules on all nodes:

```
sudo tee /etc/modules-load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
```

Configure the critical kernel parameters for Kubernetes using the following:

```
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
```

Then, reload the changes:

```
sudo sysctl --system
```

Step 4: Install Containerd Runtime (all nodes)

We are using the containerd runtime. Install containerd and its dependencies with the following commands:

```
ll -y curl gnupg2 software-properties-common apt-transport-https ca-certificates
```

Enable the Docker repository:

```
.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/docker.gpg
4] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

Update the package list and install containerd:

```
sudo apt update
sudo apt install -y containerd.io
```

Configure containerd to start using systemd as cgroup:

```
config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1
```

```
i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml
```

Restart and enable the containerd service:

```
sudo systemctl restart containerd
sudo systemctl enable containerd
```

Step 5: Add Apt Repository for Kubernetes (all nodes)

Kubernetes packages are not available in the default Ubuntu 22.04 repositories. Add the Kubernetes repositories with the following commands:

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo gpg --dearmor | sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
```

Step 6: Install Kubectl, Kubeadm, and Kubelet (all nodes)

After adding the repositories, install essential Kubernetes components, including kubectl, kubelet, and kubeadm, on all nodes with the following commands:

```
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```

Step 7: Initialize Kubernetes Cluster with Kubeadm (master node)

With all the prerequisites in place, initialize the Kubernetes cluster on the master node using the following Kubeadm command:

```
sudo kubeadm init
```

```

root@master:~# sudo kubeadm init
[init] Using Kubernetes version: v1.28.3
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your int
[preflight] You can also perform this action in beforehand using 'kubeadm confi
W1102 19:06:53.288119 10840 checks.go:835] detected that the sandbox image "r
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [kubernetes kubernetes.c
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [localhost master] and
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [localhost master] and I
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifest
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/k
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.
[kubelet-start] Starting the kubelet
[wait-control-plane] Waiting for the kubelet to boot up the control plane as st
[apiclient] All control plane components are healthy after 8.002720 seconds
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in
[kubelet] Creating a ConfigMap "kubelet-config" in namespace kube-system with t
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node master as control-plane by adding the lab
[mark-control-plane] Marking the node master as control-plane by adding the tai
[bootstrap-token] Using token: flh95l.u4nkex9cw8d0g63w
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Ro
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get r
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller aut
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all r
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" na
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotata
[addons] Applied essential addon: CoreDNS

```

[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Alternatively, if you are the root user, you can run:

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

Then you can join any number of worker nodes by running the following on each a

```
kubeadm join 146.190.135.86:6443 --token flh95l.u4nkex9cw8d0g63w \
--discovery-token-ca-cert-hash sha256:6d15f2a79bdb38d1666af50c85f060b9f
```

After the initialization is complete make a note of the `kubeadm join` command for future reference.

Run the following commands on the master node:

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Next, use `kubectl` commands to check the cluster and node status:

```
kubectl get nodes
```

```
root@master:~# kubectl get nodes
NAME        STATUS    ROLES    AGE   VERSION
master      NotReady  control-plane  3m21s  v1.28.2
root@master:~#
```

Step 8: Add Worker Nodes to the Cluster (worker nodes)

On each worker node, use the `kubeadm join` command you noted down earlier:

```
kubeadm join 146.190.135.86:6443 --token f1h95l.u4nkex9cw8d0g63w --disc
```

```
root@worker:~# kubeadm join 146.190.135.86:6443 --token f1h95l.u4nkex9cw8d0g63w --discovery-token-ca-cert-hash sha256:6d15f2a79bdb38d1666af50c85f060b9fadc73f13c932e0e2a9eeef08f51f91a
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[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 apiserer 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.
```

Step :9 Install Kubernetes Network Plugin (master node)

To enable communication between pods in the cluster, you need a network plugin. Install the Calico network plugin with the following command from the master node:

Open in app ↗

Sign up

Sign in

 Medium

 Search



Step 10: Verify the cluster and test (master node)

Finally, we want to verify whether our cluster is successfully created.

```
kubectl get pods -n kube-system
kubectl get nodes
```



```
root@master:~# kubectl get nodes
NAME        STATUS    ROLES    AGE     VERSION
master      Ready     control-plane  2d6h   v1.28.2
worker      Ready     <none>      2d6h   v1.28.2
root@master:~# kubectl get po -n kube-system
NAME                                            READY   STATUS    RESTARTS   AGE
calico-kube-controllers-658d97c59c-xqj9p      1/1     Running   0           2d6h
calico-node-kp5kh                             1/1     Running   0           2d6h
calico-node-t6csv                             1/1     Running   0           2d6h
coredns-5dd5756b68-klbdw                     1/1     Running   0           2d6h
coredns-5dd5756b68-wxgx8                     1/1     Running   0           2d6h
etcd-master                                   1/1     Running   0           2d6h
kube-apiserver-master                         1/1     Running   0           2d6h
kube-controller-manager-master               1/1     Running   0           2d6h
kube-proxy-4gm7j                             1/1     Running   0           2d6h
kube-proxy-fwdnv                             1/1     Running   0           2d6h
kube-scheduler-master                        1/1     Running   0           2d6h
root@master:~#
```

Step 11: Deploy test application on cluster (master node)

```
kubectl run nginx --image=nginx
```

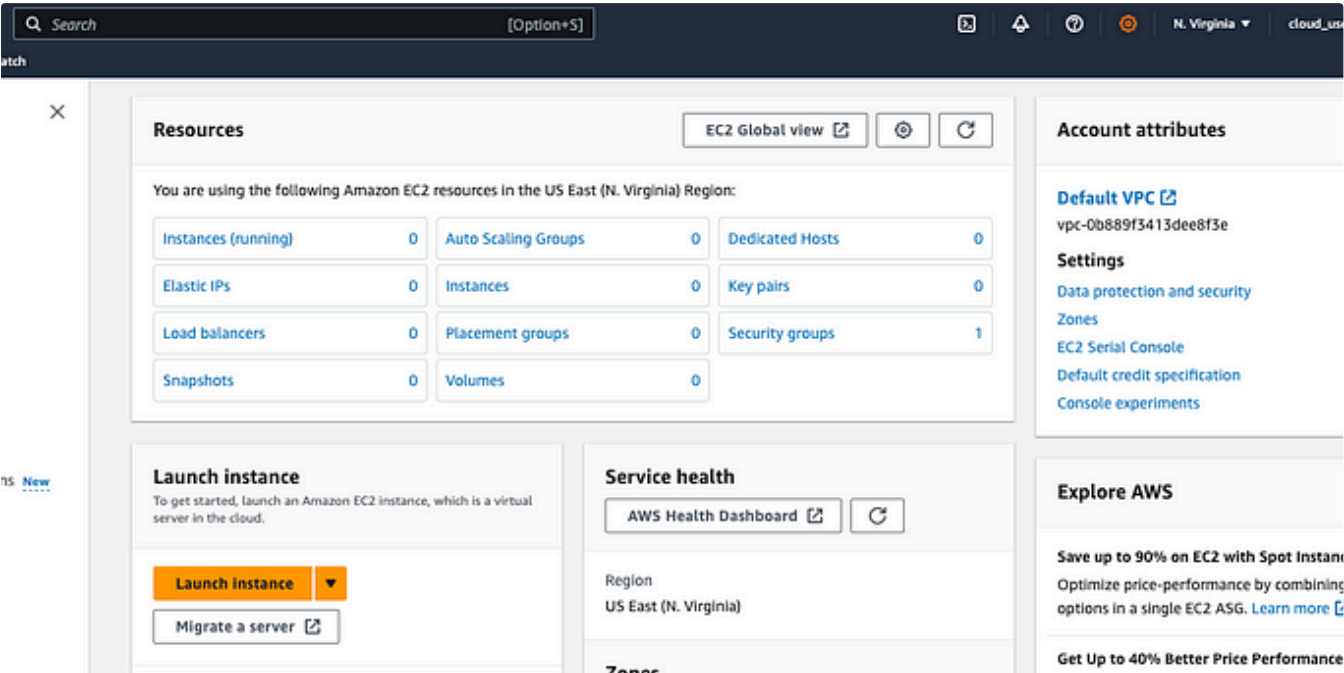
```
root@master:~# kubectl get po
NAME        READY   STATUS    RESTARTS   AGE
nginx       1/1     Running   0           46s
root@master:~#
```


[Kubernetes Cluster](#)[Ubuntu](#)[Kubernetes Installation](#)[Kubeadm](#)[Containers](#)[Follow](#)

Written by Hakan Bayraktar

265 Followers

More from Hakan Bayraktar



 Hakan Bayraktar

How to Install PostgreSQL 15 on Amazon Linux 2023: A Step-by-Step Guide

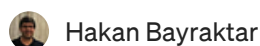
Introduction

6 min read · Nov 9, 2023

 80

 4

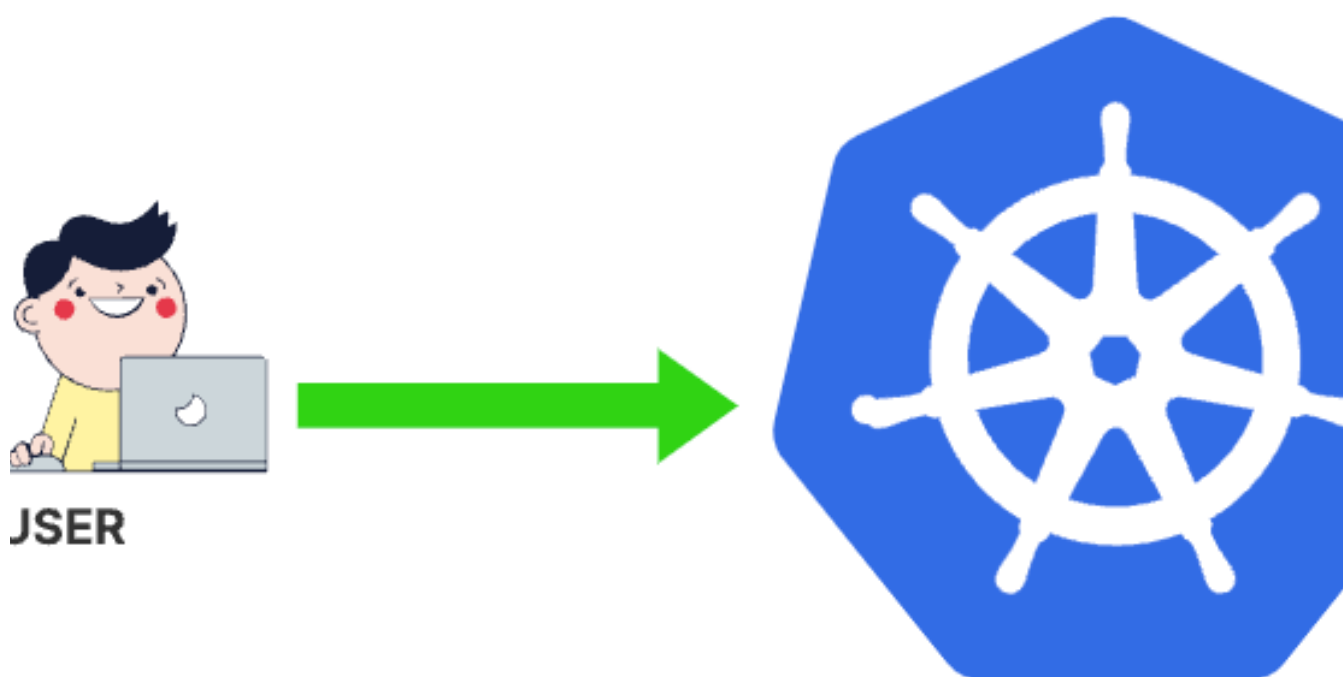




How to Setup Dynamic NFS Provisioning in a Kubernetes Cluster

Dynamic NFS storage provisioning in Kubernetes streamlines the creation and management of NFS volumes for your Kubernetes applications. It...

4 min read · Nov 3, 2023



How to Create a User in a Kubernetes Cluster and Grant Access

In this detailed guide, we'll illustrate the steps required to create a user, generate necessary certificates, and configure access using a...

4 min read · Nov 18, 2023



17



Hakan Bayraktar

Merging Multiple kubeconfig Files into One: A Comprehensive Guide

Introduction

2 min read · Nov 7, 2023



2



See all from Hakan Bayraktar

Recommended from Medium



Ashish Singh in DevOps.dev

Installing Kubernetes on Ubuntu 22.04

Introduction

4 min read · Jan 10, 2024



304



2



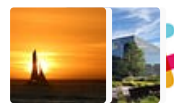
Kamitha Vihanga Bodhinayake

Complete Guide to Kubernetes Cluster Setup on Ubuntu 22.04 LTS : A Step-by-Step Tutorial for DevOps

6 min read · Dec 17, 2023

 ω^+

655 stories · 1016 saves



19 stories · 638 saves



20 stories · 1971 saves



20 stories · 1778 saves

A cartoon penguin character with a yellow beak and feet, standing next to a terminal window. The terminal window displays network configuration commands for a system named 'theo@theo-Lenovo-Flex-2-15-'. The commands include setting up loopback and ethernet interfaces, configuring IP addresses, and setting up network namespaces. The penguin is positioned to the right of the terminal, looking towards the viewer.



Linux command line for effective server management, automation...

5 min read · Jan 20, 2024



405



5



Lubomir Tobek

Kubernetes Multi-Master Node Cluster

Creating and operating a highly available Kubernetes cluster requires multiple Kubernetes control plane nodes and “Master Nodes”. To...

10 min read · Dec 13, 2023



13





Akriotis Kyriakos in ITNEXT

Install Kubernetes 1.29 using Vagrant in under 10 minutes

Step by step installation of a Kubernetes 1.29 Cluster, with 1 master and 3 worker nodes, on Ubuntu virtual machines using Vagrant

9 min read · Feb 1, 2024



69



ypik



Ubuntu

minikube



Suresh yadav in Cypik

Installing Minikube on Linux

Install Minikube on Linux , Ubuntu, and Server

5 min read · Apr 23, 2024



551

[See more recommendations](#)