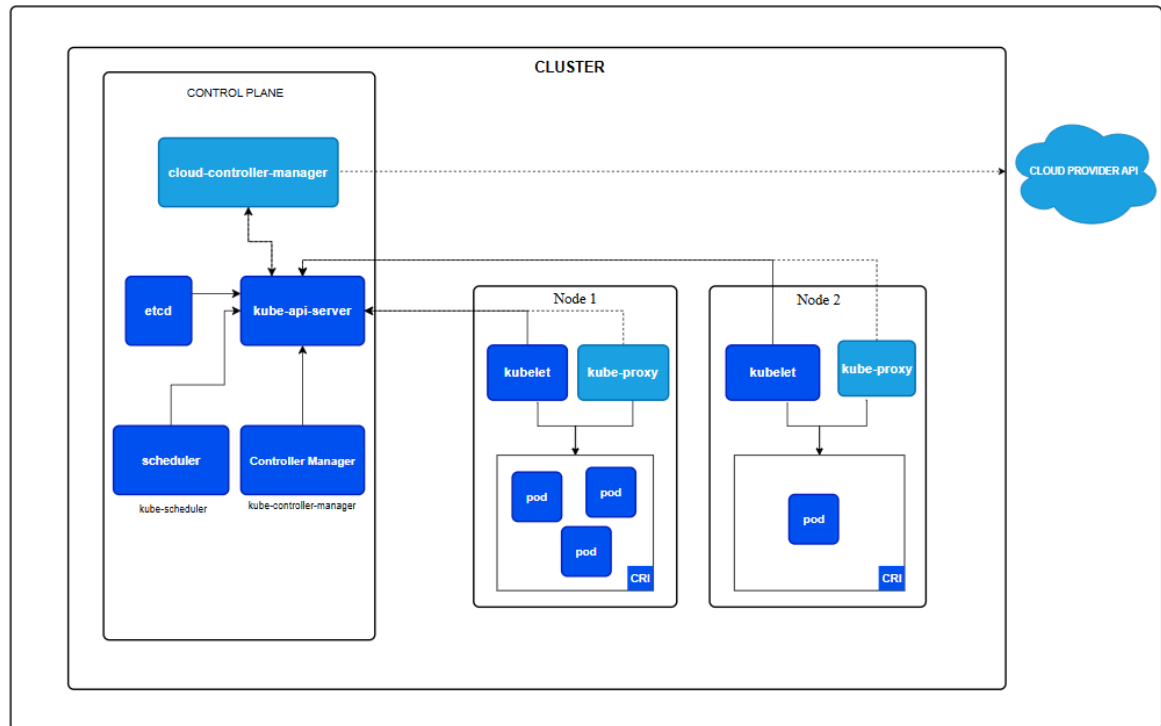


How to set up a K8s cluster with 1 control-plane and 1 data node using kubeadm



Author: Lam Thanh Bui

Created: 9/11/2024

Last update: 11/11/2024

Document:

- Kubernetes architecture: [Kubernetes cluster components](#)
- What is kubeadm, kubectl, minikube?
 - kubeadm: is a tool used to build Kubernetes clusters
 - kubectl: command line tool to run commands and interact with Kubernetes clusters
 - minikube: tool to run a single-node cluster Kubernetes on a local machine
- What is flannel?

Step by step:

1. Launch 2 instance (t2.medium) in AWS
2. Install container runtime in each node
3. Install Kubernetes tools (kubeadm, kubectl, kubelet)
4. Pull Kubernetes config images
5. Control - plane init
6. Install network plugin (flannel)
7. Add worker node to cluster
8. Bugs usually happen when setup
9. References

Launch 2 instance (t2.medium) in AWS

EC2 name

Name

[thanh.bl] control-plane

[Add additional tags](#)

Application and OS images (Select Ubuntu 22.04)

Recents

My AMIs

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu


ubuntu

Windows

Microsoft

Red Hat

Red Ha



[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-005fc0f236362e99f (64-bit (x86)) / ami-07ee04759daf109de (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Instance type (t2.medium with 2vCPUs)

Instance type

t2.medium

Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0499 USD per Hour
On-Demand Linux base pricing: 0.0464 USD per Hour
On-Demand RHEL base pricing: 0.0752 USD per Hour
On-Demand Windows base pricing: 0.0644 USD per Hour
On-Demand SUSE base pricing: 0.1464 USD per Hour

☐ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

Key pair to SSH

Key pair name - *required*

[thanh.bl] key-pair



[Create new key pair](#)

Set up network (Make sure that all ports of Kubernetes tools work)

▼ Network settings [Info](#)

VPC - *required* [Info](#)

vpc-0ae4f763f59f9ec40 ([thanh.bl] VPC)
10.0.0.0/16



Subnet [Info](#)

subnet-02e586ca5c584e443 [thanh.bl] public-subnet
VPC: vpc-0ae4f763f59f9ec40 Owner: 541253215789
Availability Zone: us-east-1a Zone type: Availability Zone
IP addresses available: 247 CIDR: 10.0.1.0/24



[Create new subnet](#) [↗](#)

Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group

☒ Select existing security group

Common security groups [Info](#)

Select security groups

[thanh.bl] frontend-sg sg-070ef8d3e171320fc ✕
VPC: vpc-0ae4f763f59f9ec40



[Compare security group rules](#)

Security groups that you add or remove here will be added to or removed from all your network interfaces.

► [Advanced network configuration](#)

Configure Storage

1x GiB ▼ Root volume (Not encrypted)

Launch instances

2 instances running:

Running ▾

<input type="checkbox"/>	Name	Instance ID	Instance state	Instanc...	Status check
<input type="checkbox"/>	[thanh.bl] control-plane	i-02b42a33...	Running	t2.medium	Initializing
<input type="checkbox"/>	[thanh.bl] worker-node	i-011db0f5...	Running	t2.medium	Initializing

Try to SSH:

```
C:\> ssh -i key.pem ubuntu@54.81.38.20
ubuntu@control-plane:~$
```

Connect to instance success.

Disable swap and add kernel modules

```
ubuntu@control-plane:~$ sudo swapoff -a
```

```
ubuntu@control-plane:~$ sudo tee /etc/modules-
load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
```

```
ubuntu@control-plane:~$ sudo modprobe overlay
```

```
ubuntu@control-plane:~$ sudo modprobe br_netfilter
```

```
ubuntu@control-plane:~$ 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

```
ubuntu@control-plane:~$ sudo systemctl --system
```

Install container runtime in each node (containerd)

Install container runtime (containerd)

```
ubuntu@control-plane:~$ sudo apt install containerd
```

```
ubuntu@control-plane:~$ containerd -version
```

```
containerd github.com/containerd/containerd 1.7.12
```

Containerd installed success!

Configure containerd

```
ubuntu@control-plane:~$ sudo mkdir /etc/containerd
```

Create the configurations with command

```
ubuntu@control-plane:~$ containerd config default |  
sudo tee /etc/containerd/config.toml
```

Enable SystemdCgroup with command

```
ubuntu@control-plane:~$ sudo sed -i  
's/SystemdCgroup = false/SystemdCgroup = true/g'  
/etc/containerd/config.toml
```

Check with command:

```
ubuntu@control-plane:~$ sudo cat  
/etc/containerd/config.toml | grep true
```

Download the required systemd with command

```
ubuntu@control-plane:~$ sudo curl -  
L https://raw.githubusercontent.com/containerd/containerd/main/containerd.service -o  
/etc/systemd/system/containerd.service
```

Reload the systemd daemon

```
ubuntu@control-plane:~$ sudo systemctl daemon-  
reload  
ubuntu@control-plane:~$ sudo systemctl enable --now  
containerd
```

Check containerd status

```
ubuntu@control-plane:~$ sudo systemctl status  
containerd
```

```
ubuntu@ip-10-0-1-87:~$ sudo systemctl status containerd  
● containerd.service - containerd container runtime  
   Loaded: loaded (/etc/systemd/system/containerd.service; enabled; vendor preset: enabled)  
   Active: active (running) since Fri 2024-11-08 23:48:45 UTC; 7min ago  
     Docs: https://containerd.io  
  Main PID: 1678 (containerd)  
    Tasks: 8  
   Memory: 13.4M  
      CPU: 1.206s  
   CGroup: /system.slice/containerd.service  
           └─1678 /usr/bin/containerd
```

You better see its active :D

Note: I'm working with control - plane, everything same with data - node

Install Kubernetes tools (kubeadm, kubectl, kubelet)

```
ubuntu@control-plane:~$ sudo apt-get update
```

```
ubuntu@control-plane:~$ sudo apt-get install -y  
apt-transport-https ca-certificates curl gpg
```

```
ubuntu@control-plane:~$ sudo mkdir -p -m 755  
/etc/apt/keyrings
```

```
ubuntu@control-plane:~$ curl -fsSL  
https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release  
.key | sudo gpg --dearmor -o  
/etc/apt/keyrings/kubernetes-apt-keyring.gpg  
ubuntu@control-plane:~$ echo 'deb [signed-  
by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]  
https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' |  
sudo tee /etc/apt/sources.list.d/kubernetes.list
```

Update package, install kubernetes tools (kubelet, kubeadm, kubectl)

```
ubuntu@control-plane:~$ sudo apt-get update
```

```
ubuntu@control-plane:~$ sudo apt-get install -y  
kubelet kubeadm kubectl
```

```
ubuntu@control-plane:~$ sudo apt-mark hold kubelet  
kubeadm kubectl
```

Pull Kubernetes config images

```
ubuntu@control-plane:~$ sudo mkdir -p -m 755  
/etc/apt/keyrings
```

Check image need to install

```
ubuntu@control-plane:~$ sudo kubeadm config images  
list
```

Install images:

```
ubuntu@control-plane:~$ for i in $(sudo kubeadm  
config images list); do sudo ctr -n k8s.io images  
pull $i -k; done
```

Control - plane init

Init control-plane using command

```
ubuntu@control-plane:~$ sudo kubeadm init --  
control-plane-endpoint=10.0.1.87:6443 --pod-  
network-cidr=10.244.0.0/16 --cri-  
socket=unix:///var/run/containerd/containerd.sock
```

Note: CIDR of pod can different and based on network plugin (now i'm using flannel)

Install CNI (Flannel)

```
ubuntu@control-plane:~$ kubectl apply -f  
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```


Wait a minute, we will see that out Control - plane is ready!

Add worker node to cluster

In master node, run command:

```
ubuntu@control-plane:~$ kubeadm token create --print-join-command
```

Copy the output and run in machine you want to add to be a worker node:

Result will be like:

```
ubuntu@ip-10-0-1-87:~$ kubectl label nodes ip-10-0-1-55 node-role.kubernetes.io/worker=worker
node/ip-10-0-1-55 labeled
ubuntu@ip-10-0-1-87:~$ kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-1-55	Ready	worker	111s	v1.31.2
ip-10-0-1-87	Ready	control-plane	16m	v1.31.2

Bugs happen when setup and how to fix

- network plugin not installed (coredns pods pending status)

```
ubuntu@ip-10-0-1-87:~$ kubectl get pods --all-namespaces
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-flannel	kube-flannel-ds-5x5b5	1/1	Running	1 (6m28s ago)	3h38m
kube-flannel	kube-flannel-ds-l77mt	1/1	Running	1 (6m27s ago)	3h38m
kube-system	coredns-7c65d6cfc9-t6z7n	1/1	Running	3 (6m27s ago)	7h59m
kube-system	coredns-7c65d6cfc9-w4fqc	1/1	Running	3 (6m27s ago)	7h59m
kube-system	etcd-ip-10-0-1-87	1/1	Running	9 (6m27s ago)	8h
kube-system	kube-apiserver-ip-10-0-1-87	1/1	Running	11 (6m27s ago)	7h59m
kube-system	kube-controller-manager-ip-10-0-1-87	1/1	Running	10 (6m27s ago)	7h58m
kube-system	kube-proxy-75jsc	1/1	Running	5 (6m27s ago)	7h59m
kube-system	kube-proxy-cms9g	1/1	Running	5 (6m28s ago)	7h45m
kube-system	kube-scheduler-ip-10-0-1-87	1/1	Running	9 (6m27s ago)	7h59m

- containerd not active: create a symlink

When you change cgroup=true but you forgot to restart, you will get this bug:

Solution: Create a symlink with below command:

```
ubuntu@control-plane:~$ sudo ln -s  
/usr/bin/containerd /usr/local/bin/containerd
```

```
ubuntu@control-plane:~$ sudo systemctl restart  
containerd
```

- You would get bug when read (most) documents in internet when install kubernetes tools with command:

```
ubuntu@control-plane:~$ echo "deb [signed-  
by=/usr/share/keyrings/kubernetes-archive-  
keyring.gpg] https://apt.kubernetes.io/ kubernetes-  
xenial main" | sudo tee  
/etc/apt/sources.list.d/kubernetes.list
```

```
ubuntu@control-plane:~$ sudo apt update
```

If you do like below, you will get some refuse connect from an old IP.
Change step like in above step.
You can read more information in [here](#).

References

1. [How to setup Kubernetes Cluster with Kubeadm on Ubuntu 22.04](#)
2. [How to Install Kubernets Cluster \(kubeadm setup\) on Ubuntu 24.04 LTS \(Step-by-step Guide\)](#)
3. [Free Kubernetes Lab](#)