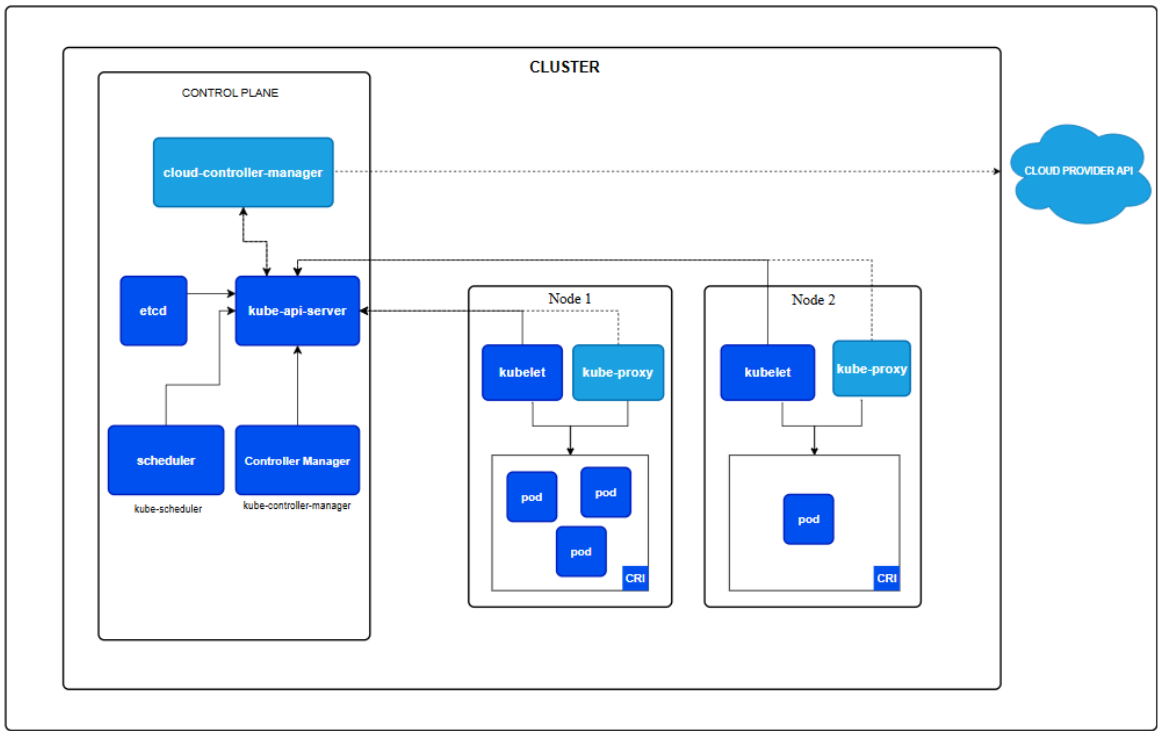


How to set up a K8s cluster with 1 control-plane and 1 data node using kubeadm

Author: Lam Thanh Bui
Created: 9/11/2024
Last update: 9/11/2024

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. Install Flannel
5. Pull Kubernetes config images
6. Control – plane init
7. Add worker node to cluster
8. Bugs usually happen when setup
9. References

Launch 2 instance (t2.medium) in AWS

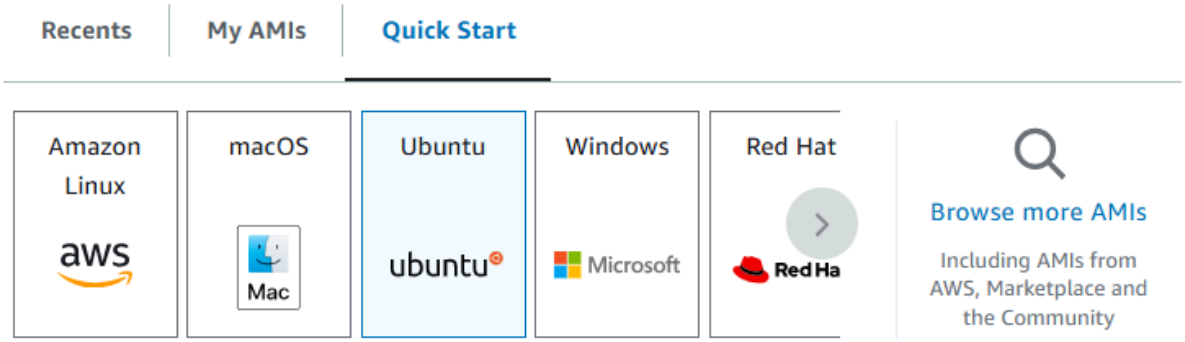
EC2 name

Name

[thanh.bl] control-plane

Add additional tags

Application and OS images (Select Ubuntu 22.04)



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

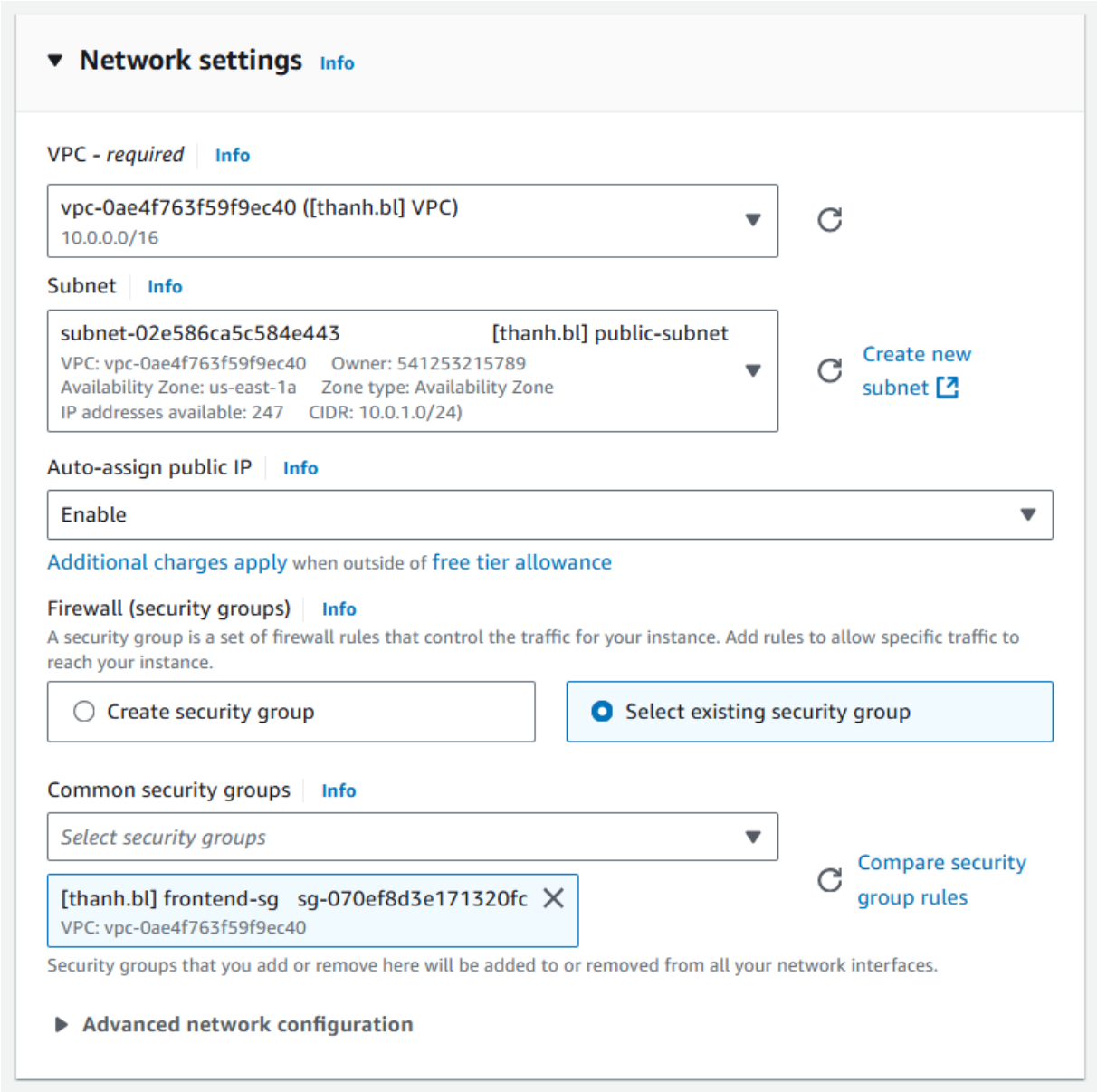
Key pair to SSH

Key pair name - *required*

[thanh.bl] key-pair

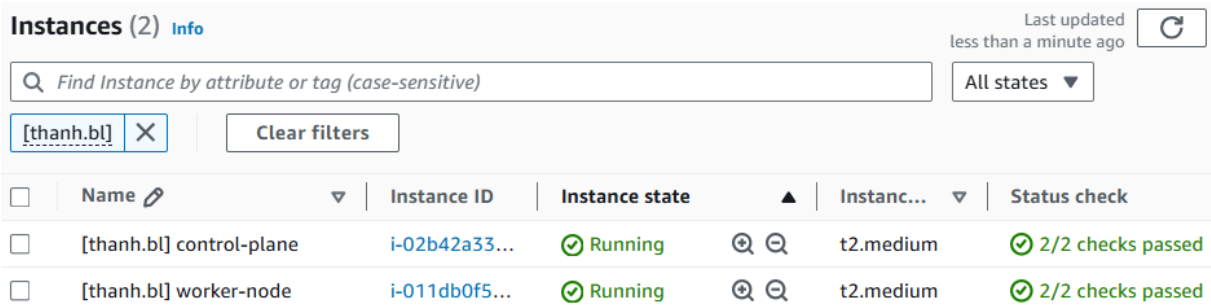
Create new key pair

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



Configure Storage

Launch instances:



Try to SSH:

```
C:\> ssh -i key.pem ubuntu@54.81.38.20
```

```
ubuntu@ip-10-0-1-87:~$
```

Connect to instance success.
Disable swap and add kernel modules

```
ubuntu@ip-10-0-1-87:~$ sudo swapoff -a
ubuntu@ip-10-0-1-87:~$ sudo tee /etc/modules-load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
ubuntu@ip-10-0-1-87:~$ sudo modprobe overlay
ubuntu@ip-10-0-1-87:~$ sudo modprobe br_netfilter
ubuntu@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ sudo sysctl --system
```

Install container runtime in each node (containerd)

Install container runtime (containerd)

```
ubuntu@ip-10-0-1-87:~$ sudo apt install containerd
ubuntu@ip-10-0-1-87:~$ containerd –version
containerd github.com/containerd/containerd 1.7.12
```

Containerd installed success!

Configure containerd

```
ubuntu@ip-10-0-1-87:~$ sudo mkdir /etc/containerd
```

Create the configurations with command

```
ubuntu@ip-10-0-1-87:~$ containerd config default |
sudo tee /etc/containerd/config.toml
```

Enable SystemdCgroup with command

```
ubuntu@ip-10-0-1-87:~$ sudo sed -i 's/SystemdCgroup
= false/SystemdCgroup = true/g'
/etc/containerd/config.toml
```

Check with command:

```
ubuntu@ip-10-0-1-87:~$ sudo cat
/etc/containerd/config.toml | grep true
```

Dowload the required systemd with command

```
ubuntu@ip-10-0-1-87:~$ sudo curl -
L https://raw.githubusercontent.com/containerd/cont
ainerd/main/containerd.service -o
/etc/systemd/system/containerd.service
```

Reload the systemd daemon

```
ubuntu@ip-10-0-1-87:~$ sudo systemctl daemon-reload
ubuntu@ip-10-0-1-87:~$ sudo systemctl enable --now
containerd
```

Check containerd status

```
ubuntu@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ sudo apt-get update
ubuntu@ip-10-0-1-87:~$ sudo apt-get install -y apt-transport-https ca-certificates curl gpg
ubuntu@ip-10-0-1-87:~$ sudo mkdir -p -m 755 /etc/apt/keyrings
ubuntu@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ sudo apt-get update
```

```
ubuntu@ip-10-0-1-87:~$ sudo apt-get install -y
kubelet kubeadm kubectl
ubuntu@ip-10-0-1-87:~$ sudo apt-mark hold  kubelet
kubeadm kubectl
```

Pull Kubernetes config images

```
ubuntu@ip-10-0-1-87:~$ sudo mkdir -p -m 755
/etc/apt/keyrings
```

Install images:

```
ubuntu@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/ma
ster/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@ip-10-0-1-87:~$ 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
- You would get bug when read (most) documents in internet when install kubernetes tools with command:

```
ubuntu@ip-10-0-1-87:~$ 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@ip-10-0-1-87:~$ 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