Here’s your content properly formatted:

**Install Docker**

**Update the Package List:**

sudo apt-get update

2. Install Required Packages:

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

3. Add Docker’s Official GPG Key:

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

4. Add Docker APT Repository:

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

5. Update the Package List Again:

sudo apt-get update

6. Install Docker:

sudo apt-get install -y docker-ce

7. Verify Docker Installation:

sudo systemctl status docker

**Install Kind**

Execute the below script on the host to install the kind command. This script is inspired by the official Kind installation.

#!/bin/bash

# For AMD64 / x86\_64

[ $(uname -m) = x86\_64 ] && curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.20.0/kind-linux-amd64

chmod +x ./kind

sudo cp ./kind /usr/local/bin/kind

rm -rf kind

After executing the above script, the kind command will be accessible on the host.

[amit@test-kubernetes]# kind --version

kind version 0.20.0

[amit@test-kubernetes]#

**Bring Up a Multi-Node Cluster**

Create a configuration file (config.yml) with the following content:

# 4 node (3 workers) cluster config

kind: Cluster

apiVersion: kind.x-k8s.io/v1alpha4

nodes:

- role: control-plane

- role: worker

- role: worker

- role: worker

Start the 4-node cluster:

[amit@test-kubernetes]# kind create cluster --config=config.yml

Creating cluster "kind" ...

✓ Ensuring node image (kindest/node:v1.28.0) 🖼

✓ Preparing nodes 📦 📦 📦

✓ Writing configuration 📜

✓ Starting control-plane 🕹️

✓ Installing CNI 🔌

✓ Installing StorageClass 💾

✓ Joining worker nodes 🚜

Set kubectl context to "kind-kind"

You can now use your cluster with:

kubectl cluster-info --context kind-kind

**Check Cluster Status**

**Using Kind**

[amit@test-kubernetes]# kind get clusters

kind

**Using kubectl**

[amit@test-kubernetes]# kubectl cluster-info

Kubernetes control plane is running at https://127.0.0.1:41273

CoreDNS is running at https://127.0.0.1:41273/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

**Check Node Status**

[amit@test-kubernetes]# kubectl get nodes

NAME STATUS ROLES AGE VERSION

kind-control-plane Ready control-plane 22m v1.28.0

kind-worker Ready <none> 21m v1.28.0

kind-worker2 Ready <none> 21m v1.28.0

kind-worker3 Ready <none> 21m v1.28.0

**Check Kubernetes Version**

[amit@test-kubernetes]# kubectl version

Client Version: v1.28.3

Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3

Server Version: v1.28.0

From the output, we can see that the cluster is running on version **1.28.0**, whereas the client is on **1.28.3**.

This maintains the structure without any extra content while keeping it clean and readable.