## Darshan Nikam Date: 12/04/2024

# How to Install Minikube on Ubuntu 20.04 LTS

## What is Minikube in Kubernetes ?

**Minikube**creates a single node cluster inside a VM or Cloud Instance. It is good for beginners to learn Kubernetes since you don’t have to create a master and worker node to create a cluster and we can practice basic Kubernetes functions and can also install the Kubernetes dashboard on it.

## Minikube System Requirements

* Minimum 2 CPU’s or more
* Minimum 2GB of free memory
* Minimum 20GB of free disk space
* Internet connection
* Container or virtual machine manager, such as: Docker, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMware Fusion/Workstation

Update the system packages on Ubuntu EC2

sudo apt update

## #1.Install kubectl on Ubuntu 20.04 LTS

Step 1: Download kubectl Binary

Command:

bash

Copy code

curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl

Explanation:

This command uses curl to download the kubectl binary from the Kubernetes release page.

The $(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt) part fetches the latest stable version of Kubernetes and inserts it into the URL.

The -LO flags with curl are used to download the file without saving the output and to follow redirects.

Step 2: Make kubectl Binary Executable

Command:

bash

Copy code

chmod +x ./kubectl

Explanation:

The chmod +x command makes the downloaded kubectl binary executable.

This step is necessary to allow the binary to be run as a command.

Step 3: Move kubectl Binary to /usr/local/bin/kubectl

Command:

bash

Copy code

sudo mv ./kubectl /usr/local/bin/kubectl

Explanation:

The mv command is used to move the kubectl binary from the current directory to /usr/local/bin/kubectl.

This directory is included in the system's PATH, allowing you to run kubectl from anywhere without specifying its full path.

Using sudo ensures that the move operation has the necessary permissions since /usr/local/bin is typically owned by root.

Step 4: Check kubectl Version

Command:

bash

Copy code

kubectl version

Explanation:

This command checks the version of kubectl installed on the system.

It displays both the client version (which you use to interact with the Kubernetes cluster) and the server version (which indicates the version of the Kubernetes control plane).

Notes:

Version Check Output:

The output you provided confirms that kubectl is installed and displays the client version information.

This information can be useful for verifying the installation and ensuring compatibility with the Kubernetes cluster you intend to manage.

Permissions:

Using sudo in the mv command ensures that you have the necessary permissions to move the kubectl binary to /usr/local/bin.

Always be cautious when using sudo to perform operations as root, especially when moving files around in system directories.

Updating kubectl:

You can use the same process to update kubectl by downloading the latest binary and replacing the existing one in /usr/local/bin.

Troubleshooting:

If you encounter any errors during these steps, double-check the commands for typos and ensure that you have internet connectivity to download the kubectl binary.

## Install Docker on Ubuntu 20.04 LTS

Step 1: Install Required Packages

Command:

bash

Copy code

sudo apt-get install ca-certificates curl gnupg lsb-release

Explanation:

This command installs the necessary packages required for adding Docker's official repository and for package management.

Step 2: Add Docker Official GPG Key

Command:

bash

Copy code

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

Explanation:

The first command creates a directory /etc/apt/keyrings where Docker's GPG key will be stored.

The second command downloads Docker's official GPG key and saves it as /etc/apt/keyrings/docker.gpg.

Step 3: Setup Docker Repository

Command:

bash

Copy code

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

Explanation:

This command adds the Docker repository to the system's package sources (/etc/apt/sources.list.d/docker.list).

It uses $(dpkg --print-architecture) to determine the system's architecture and $(lsb\_release -cs) to get the Ubuntu codename (e.g., focal, bionic).

Step 4: Update Package Manager

Command:

bash

Copy code

sudo apt-get update

Explanation:

This command updates the package manager's cache to include the newly added Docker repository and its packages.

Step 5: Install Docker and Docker Compose

Command:

bash

Copy code

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

Explanation:

This command installs Docker Engine, Docker CLI, Containerd, and Docker Compose on the system.

Step 6: Check Docker Service Status

Command:

bash

Copy code

sudo systemctl status docker

Explanation:

This command checks the status of the Docker service to verify that it's running.

It provides information about whether the service is active, its uptime, and any errors or warnings.

Step 7: Create Docker Group

Command:

bash

Copy code

sudo groupadd docker

Explanation:

This command creates a group named docker. Users added to this group will be able to run Docker commands without using sudo.

Step 8: Configure Docker to Run Without sudo

Command:

bash

Copy code

sudo usermod -aG docker $USER && newgrp docker

Explanation:

This command adds the current user to the docker group, enabling them to run Docker commands without sudo.

The newgrp docker command refreshes the current shell session to apply the group membership changes immediately.

Step 9: Enable Docker Service at System Startup

Command:

bash

Copy code

sudo systemctl enable docker

Explanation:

This command enables the Docker service to start automatically at system startup.

Step 10: Check Docker Service Status (Again)

Command:

bash

Copy code

sudo systemctl status docker

Explanation:

This command checks the status of the Docker service again to ensure that it's still active after enabling it for startup.

Step 11: Start/Stop Docker Service

Commands:

bash

Copy code

sudo systemctl start docker

sudo systemctl stop docker

Explanation:

These commands start and stop the Docker service, respectively. They are useful for manually controlling the Docker service if needed.

Notes:

Docker Compose: It seems like there's a typo in the command to install Docker Compose. It should be docker-compose instead of docker-compose-plugin. You might want to correct this to ensure Docker Compose is installed correctly.

Post-Installation Steps: After installing Docker, it's a good practice to configure Docker according to your needs, such as setting up Docker network, storage, security, etc. You might also want to pull Docker images and start containers to test Docker functionality.

## Install cri-dockerd on Ubuntu 20.04 LTS

Purpose: To install cri-dockerd, a CRI plugin for Docker, enabling Docker as a container runtime in Kubernetes.

Commands:

bash

Copy code

git clone https://github.com/Mirantis/cri-dockerd.git

cd cri-dockerd

mkdir bin

go build -o bin/cri-dockerd

sudo install -o root -g root -m 0755 bin/cri-dockerd /usr/local/bin/cri-dockerd

sudo cp -a packaging/systemd/\* /etc/systemd/system

sudo sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-dockerd,' /etc/systemd/system/cri-docker.service

sudo systemctl daemon-reload

sudo systemctl enable cri-docker.service

sudo systemctl enable --now cri-docker.socket

Step 4: Verify cri-dockerd Installation

Purpose: To confirm that cri-dockerd is properly installed and configured.

Verification Steps:

Check cri-dockerd Binary:

bash

Copy code

ls /usr/local/bin/cri-dockerd

Check Systemd Unit Files:

bash

Copy code

ls /etc/systemd/system/cri-docker.\*

Verify Systemd Service Status:

bash

Copy code

sudo systemctl status cri-docker.service

Verify Socket Status:

bash

Copy code

sudo systemctl status cri-docker.socket

Test Docker Integration with Kubernetes:

Deploy and manage Docker containers using kubectl commands.

Check Docker version:

bash

Copy code

docker version

Additional Suggestions:

Documentation:

Keep detailed notes of each step, including command explanations and potential errors.

Version Compatibility:

Ensure compatibility between kubectl, Docker, cri-dockerd, and Kubernetes versions.

Security:

Implement security measures such as RBAC and TLS encryption.

Monitoring and Logging:

Set up monitoring and logging solutions for Kubernetes clusters and Docker containers.

Backup and Recovery:

Establish backup procedures for Kubernetes resources and Docker images.

## #4.Install conntrack package on Ubuntu 20.04 LTS

Purpose: To install the conntrack package, which is required for Minikube on Ubuntu 20.04 LTS.

Commands:

bash

Copy code

sudo apt-get update

sudo apt-get install -y conntrack

Explanation:

The apt-get update command ensures that the package list is up to date before installation.

The apt-get install -y conntrack command installs the conntrack package, which is necessary for Minikube to manage network connections effectively.

After executing these commands, conntrack should be installed on your Ubuntu 20.04 LTS system, allowing you to proceed with your Minikube setup.

## #5.Install crictl package on Ubuntu 20.04 LTS

Install crictl Package on Ubuntu 20.04 LTS

Purpose: To install crictl, a CLI tool for interacting with the kubelet's Container Runtime Interface (CRI), on Ubuntu 20.04 LTS.

Commands:

bash

Copy code

VERSION="v1.24.2"

wget https://github.com/kubernetes-sigs/cri-tools/releases/download/$VERSION/crictl-$VERSION-linux-amd64.tar.gz

sudo tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin

rm -f crictl-$VERSION-linux-amd64.tar.gz

Explanation:

The wget command downloads the crictl binary archive from the official GitHub release page of cri-tools, using the specified version.

The tar command extracts the crictl binary from the downloaded archive and places it in the /usr/local/bin directory.

The rm command removes the downloaded archive file to clean up the system.

After executing these commands, crictl should be installed on your Ubuntu 20.04 LTS system, allowing you to interact with the kubelet's CRI.

## 6.Download and Install Minikube on Ubuntu 20.04 LTS

Installation of Minikube on EC2 Instance

Download Minikube Binary:

ruby

Copy code

curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

Explanation: This command downloads the Minikube binary file from the official Minikube release page.

Make Minikube Binary Executable:

bash

Copy code

chmod +x minikube

Explanation: This command makes the downloaded Minikube binary file executable, allowing you to run it as a command.

Move Minikube Binary to /usr/local/bin Directory:

bash

Copy code

sudo mv minikube /usr/local/bin/

Explanation: Moving the Minikube binary to /usr/local/bin ensures that it's accessible from anywhere on your system.

Check Minikube Version:

Copy code

minikube version

Explanation: This command verifies the installation and displays the installed Minikube version.

Start Minikube Kubernetes Cluster:

css

Copy code

minikube start --network-plugin=cni --cni=calico

Explanation: Initiates the Minikube Kubernetes cluster with the Calico CNI plugin for network configuration.

Handle Error (GUEST\_MISSING\_CONNTRACK):

arduino

Copy code

sudo apt-get install -y conntrack

Explanation: Resolves the error related to missing conntrack package by installing it using apt.

Retry Starting Minikube:

css

Copy code

minikube start --vm-driver=none

Explanation: Restarts Minikube after resolving the conntrack issue.

Check Minikube Status:

lua

Copy code

minikube status

Explanation: Verifies the status of the Minikube cluster, ensuring it's running properly.

Phase 2: Basic Operations and Verification

View Minikube Cluster Information:

Copy code

kubectl cluster-info

Explanation: Provides information about the Minikube cluster, including the Kubernetes control plane and CoreDNS status.

Check Minikube Cluster Status:

arduino

Copy code

kubectl get nodes

Explanation: Displays the status of nodes in the Minikube cluster, ensuring they are ready and available.

View Minikube Cluster Events:

arduino

Copy code

kubectl get events

Explanation: Shows recent events in the Minikube cluster, helping to diagnose any issues or errors.

View Kubectl Configuration:

arduino

Copy code

kubectl config view

Explanation: Displays the current Kubectl configuration, including cluster, context, and user information.

Deploy a Container on Minikube Cluster:

arduino

Copy code

kubectl run hello-minikube --image=gcr.io/google\_containers/echoserver:1.4 --port=8080

Explanation: Creates a deployment named "hello-minikube" using a sample container image.

Check Pods on Minikube:

arduino

Copy code

kubectl get pods

Explanation: Lists the pods running in the Minikube cluster, verifying the successful deployment of the container.

Error Handling and Troubleshooting

Error Handling Strategy:

Take note of any errors encountered during the practice.

Search for error messages online or refer to official documentation for troubleshooting steps.

Ask for assistance if needed, providing detailed error messages and steps already attempted.

Additional Suggestions:

Documentation:

Maintain organized notes documenting each step, command, and its purpose.

Include any errors encountered and their resolutions for future reference.

Keep notes concise but informative, making them easy to follow during future practices or troubleshooting.

Exploration and Experimentation:

Experiment with different Minikube configurations, options, and commands to deepen your understanding.

Explore additional Kubernetes features and concepts beyond basic operations.

By following these steps and suggestions, you'll not only successfully practice Minikube on your EC2 instance but also build a valuable resource for future reference and learning.