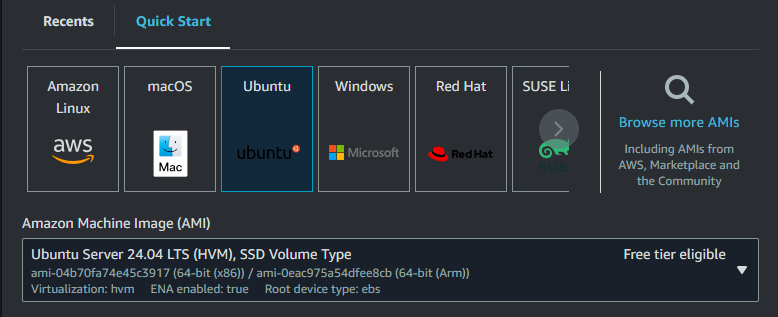
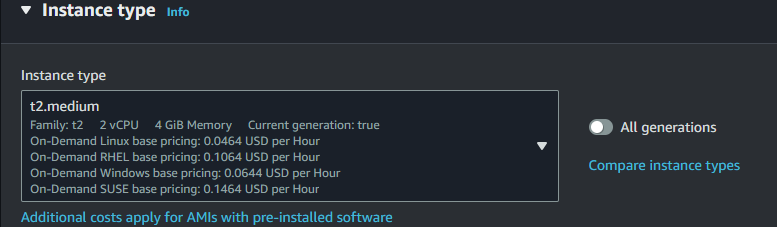
Kubernetes – Minikube Installation and Configuration

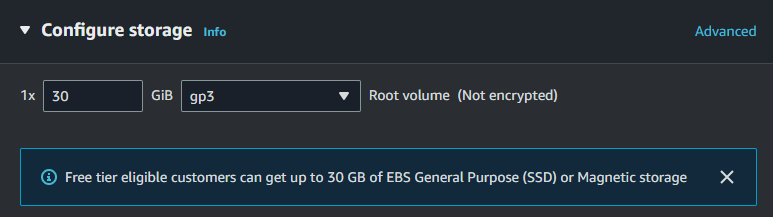
* First we need an Ubuntu instance with 2cpu.
* Select the Ubuntu AMI.



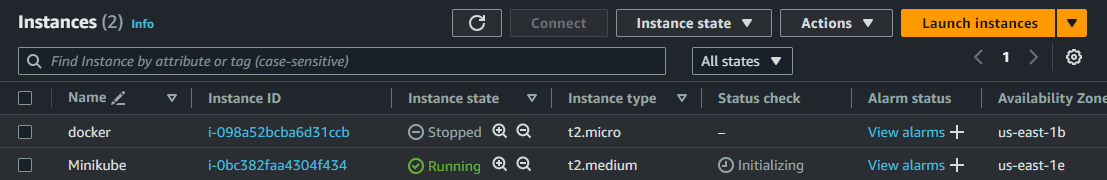
* In Instance Type select t2.medium.



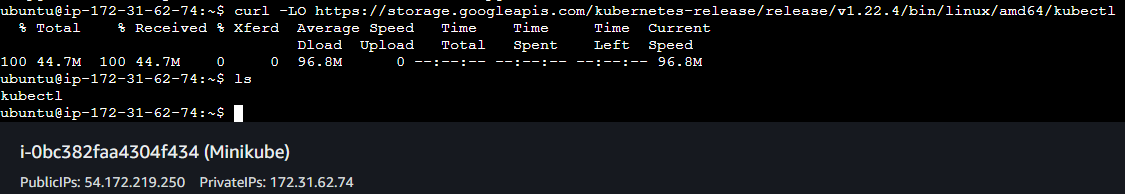
* Enter the Storage to 30 GB as we need 20 GB free storage.



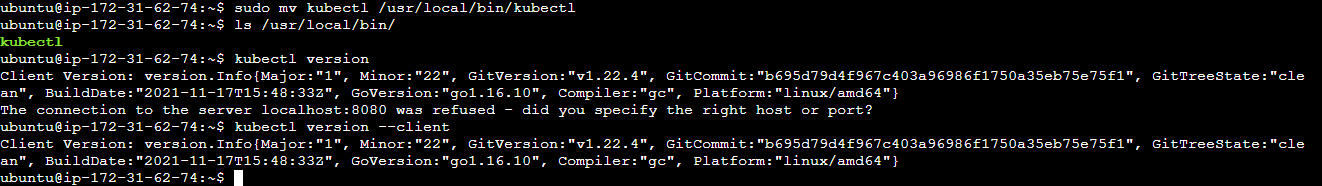
* Click Launch Instance.
* Instance Created.



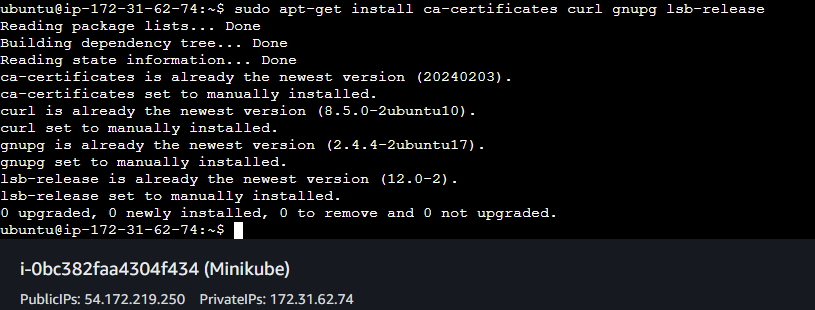
* Now connect the Instance.
* Hit command “sudo apt update && sudo apt upgrade”.
* Now we need to download and install the Kubectl command line tool.
* kubectl is the command-line interface (CLI) tool used to interact with Kubernetes clusters. It allows users to perform various tasks, such as deploying applications, managing resources, inspecting cluster state, and debugging issues within Kubernetes environments.
* Hit command “curl -LO <https://storage.googleapis.com/kubernetes-release/release/v1.22.4/bin/linux/amd64/kubectl>” to download the kubectl binary file.



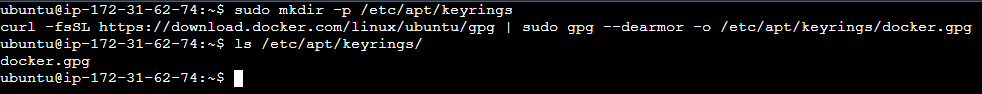
* Now we need to give the execute permission to the kubectl file.
* Hit command “chmod +x ./kubectl” to give the execute permission to the file.
* Now we need to move the kubectl file to the location /usr/local/bin/ as it is the common location for the executable files.
* Hit command “sudo mv kubectl /usr/local/bin/kubectl” to move the file.
* Hit command “kubectl version –client” to check the kubectl version.



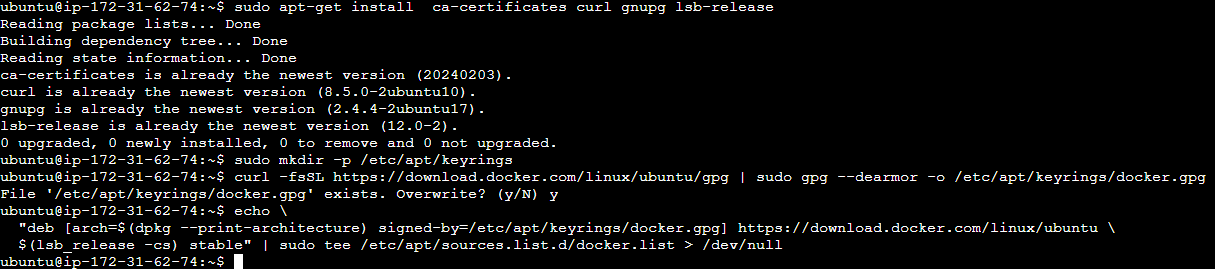
* Now we need to install the packages “ca-certificates”, “gnupg” and “lsb-release” as these packages are required for installing and setting up Docker.
* Hit command “sudo apt-get install ca-certificates curl gnupg lsb-release” to download the packages.



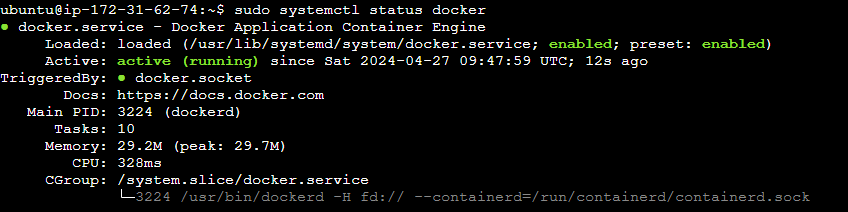
* Hit command “sudo mkdir -p /etc/apt/keyrings
* curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg” to make the directory keyrings if not present and download the docker GPG key.
* sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg: Imports Docker's GPG key into the system's keyring.



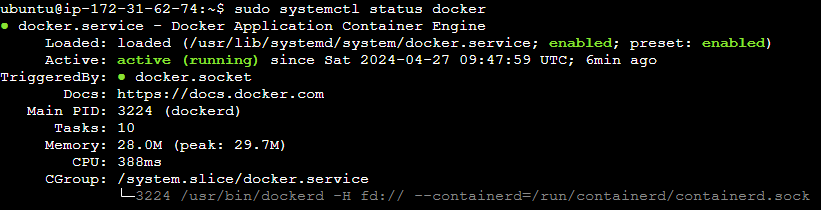
* Docker's official GPG key is a trusted entity that users can rely on to verify the authenticity of Docker packages. By importing this key into the system's keyring, users establish a trust relationship with Docker's software distribution channel.
* Now we need to configure the Docker repository.
* Hit command “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”
* echo ...: Prints the Docker repository configuration to standard output.
* sudo tee /etc/apt/sources.list.d/docker.list > /dev/null: Writes the Docker repository configuration to /etc/apt/sources.list.d/docker.list.



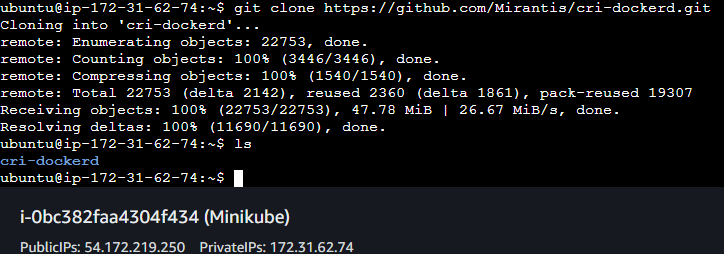
* Hit command “sudo apt-get update” to refresh the list of available packages from the repositories.
* Hit command “sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin” this will install the components required for Docker on Ubuntu.
* Hit command “sudo systemctl status docker” to check the docker status.



* Now we need a group named docker.
* Hit command “sudo groupadd docker” to create a group.
* Hit command “sudo usermod -aG docker $USER && newgrp docker” to modify user.
* usermod -aG docker $USER: This command adds the current user ($USER) to the "docker" group (-aG docker). Adding a user to the "docker" group grants them permissions to execute Docker commands without requiring sudo privileges.
* newgrp docker: This command activates the group membership for the current user in the current shell session. After running this command, any Docker commands executed in this session will be executed with the permissions granted to the "docker" group.
* Now we need to enable the docker.
* Hit command “sudo systemctl enable docker”.
* Hit command “sudo systemctl status docker” to check the docker status.



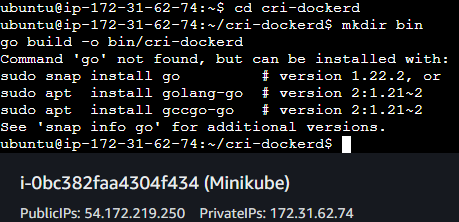
* Now we need to clone the cri-dockerd github repository.
* Hit command “git clone <https://github.com/Mirantis/cri-dockerd.git>”.



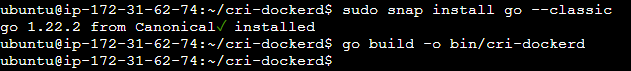
* Hit command “wget <https://storage.googleapis.com/golang/getgo/installer_linux>” to download the installer\_linux file.
* Hit command “chmod +x ./installer\_linux” to give the execute permission to the file.
* Hit command “./installer\_linux” to execute the file.
* Hit command “source ~/.profile” to refresh the profile.



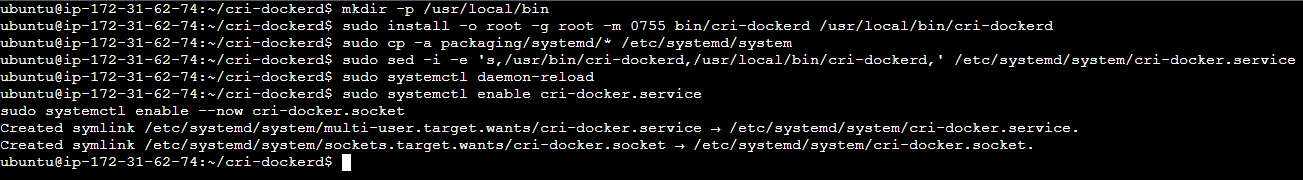
* Hit command “cd cri-dockerd” to change the directory.
* Hit command “mkdir bin” to create a directory called bin.
* Hit command “go build -o bin/cri-dockerd” to build the cri-dockerd.



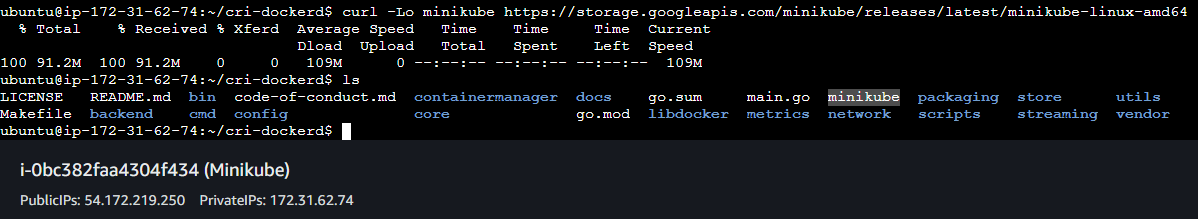
* Here the Go is not installed so we have to install it.
* Hit command “sudo snap install go --classic”.
* Now try again to build.
* Hit command “go build -o bin/cri-dockerd” and wait for some time.



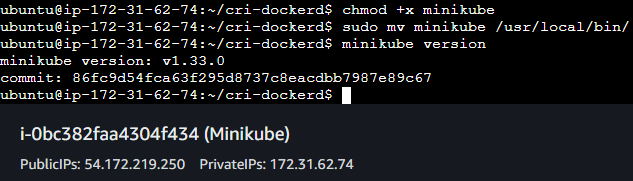
* Hit command “mkdir -p /usr/local/bin” to create directory if not present.
* Hit command “sudo install -o root -g root -m 0755 bin/cri-dockerd /usr/local/bin/cri-dockerd”.
* This command installs the cri-dockerd binary to /usr/local/bin, making it accessible system-wide. It sets the correct permissions (owner, group, and mode) for the binary.
* sudo cp -a packaging/systemd/\* /etc/systemd/system
* This command copies the systemd service files from the cri-dockerd repository to the appropriate directory (/etc/systemd/system). These service files define how the cri-dockerd service should be managed by systemd.
* Hit command “sudo sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-dockerd,' /etc/systemd/system/cri-docker.service”.
* This command updates the systemd service configuration file to point to the installed cri-dockerd binary. It ensures that systemd knows the correct path to the cri-dockerd executable.
* Hit command “sudo systemctl daemon-reload”.
* This command reloads systemd to read the updated service files. It ensures that systemd recognizes any changes made to the service configuration.
* Hit command “sudo systemctl enable cri-docker.service”.
* Hit command “sudo systemctl enable --now cri-docker.socket”.
* These commands enable and start the cri-dockerd service and socket. They ensure that cri-dockerd is automatically started at boot and that it can accept incoming connections through the socket.



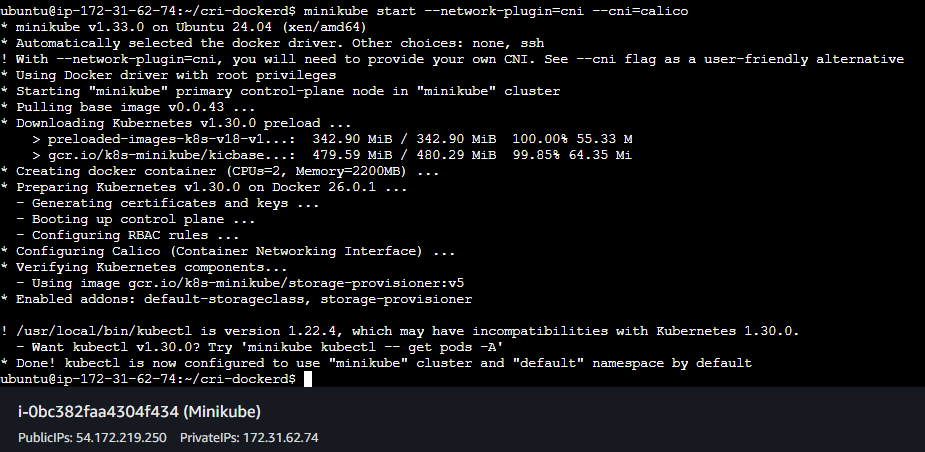
* Now we need to install the package conntrack.
* This package is required for Networking and Load Balancing.
* Hit command “sudo apt-get install -y conntrack” to install the package.
* Now we need to download the cri package.
* The crictl package is a command-line interface (CLI) tool used for interacting with the Container Runtime Interface (CRI) of the kubelet, which is a Kubernetes component responsible for managing containers on a node.
* Hit command “VERSION="v1.24.2"
* Hit command “wget <https://github.com/kubernetes-sigs/cri-tools/releases/download/$VERSION/crictl-$VERSION-linux-amd64.tar.gz>” to download the package.
* Hit command “sudo tar zxvf crictl-$VERSION-linux-amd64.tar.gz -C /usr/local/bin” to extract the tar file.
* Hit command “rm -f crictl-$VERSION-linux-amd64.tar.gz” to remove the tar.gz file.
* Now we need to download the Minikube.
* Hit command “curl -Lo minikube <https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64>”.



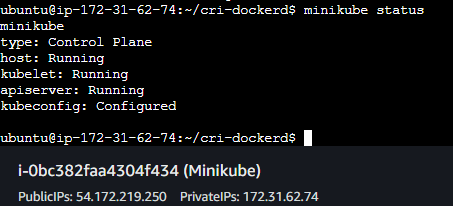
* Now we need to give the execute permission to the minikube file.
* Hit command “chmod +x minikube”.
* Now we need to move the minikube to the /usr/local/bin as all the executable files are located there.
* Hit command “sudo mv minikube /usr/local/bin/”.
* Hit command “minikube version”.



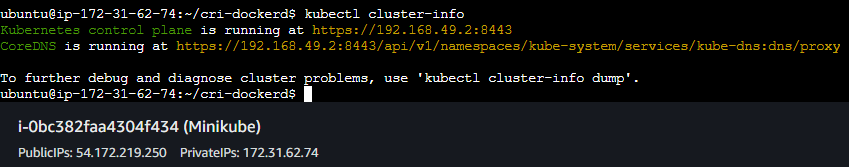
* Now hit command “minikube start --network-plugin=cni --cni=calico” to start the minikube with the calico network plugin.



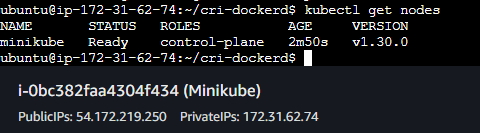
* Minikube started.
* Hit command “minikube status” to check the minikube status.



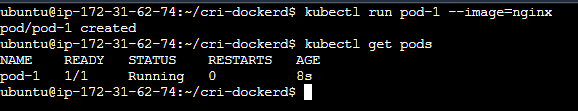
* Now our minikube is configured and running.
* Now we can run the kubectl commands here.
* Hit command “kubectl cluster-info”.



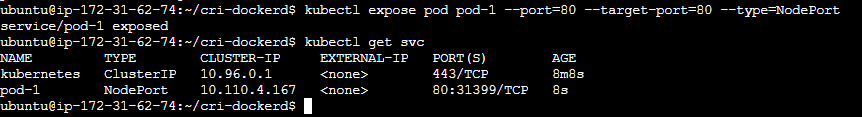
* Hit command “kubectl get nodes”.



* Creating a pod.
* Hit command “kubectl run pod-1 --image=nginx”.



* Expose the port to the service type NodePort.
* Hit command “kubectl expose pod pod-1 --port=80 --target-port=80 --type=NodePort”.



* Now allow all tcp in the security group.
* Try to hit the instance IP with the port number in incognito mode.