Terms in Kuberneties:

* Distributed System: Distributed System is a collection of autonomous computer systems that are physically separated but are connected by a centralized computer network that is equipped with distributed system software. The autonomous computers will communicate among each system by sharing resources and files and performing the tasks assigned to them.
* Node: [In the context of distributed systems, a node refers to an independent component located on a different system that communicates with other nodes in order to operate as a single unit](https://www.confluent.io/learn/distributed-systems/)
* Cluster: [In the context of distributed systems, a cluster refers to the connection of computers or servers to each other over a network to form a larger “computer”, which is based on the distributed computing architecture](https://www.gigabyte.com/Article/cluster-computing-an-advanced-form-of-distributed-computing-a-tech-guide-by-gigabyte).
* Stateful Applications: The data that is saved is called the application’s state. Apps can be stateful or stateless. When an app is stateful, client data is either stored locally or on a remote host until the user logs out or the session expires
* Stateless Applications: A stateless app is an **application program that does not save client data generated in one session for use in the next session with that client**. Each session is carried out as if it was the first time and responses are not dependent upon data from a previous session.
* Monolith:
* Microservices : Microservices are a type of distributed system, since the architecture entails decoupling a large application into a suite of small services. These services communicate with each other, often through APIs and are designed to be deployed independently.

Installing Kuberneties on Virtual machines

Step1:

* Install Docker on all machine’s

**$ curl -fsSL https://get.docker.com -o install-docker.sh**

**$ sudo sh install-docker.sh**

* Now add user to docker group

sudo usermod -aG docker ubuntu

* Now exit from machines and relogin.
* Run the below commands as root user in all the nodes
* To become root user execute “sudo –i”

**# Run these commands as root**

* wget https://storage.googleapis.com/golang/getgo/installer\_linux
* chmod +x ./installer\_linux
* ./installer\_linux
* source /root/.bash\_profile
* git clone https://github.com/Mirantis/cri-dockerd.git
* cd cri-dockerd
* mkdir bin
* go build -o bin/cri-dockerd
* mkdir -p /usr/local/bin
* install -o root -g root -m 0755 bin/cri-dockerd /usr/local/bin/cri-dockerd
* cp -a packaging/systemd/\* /etc/systemd/system
* sed -i -e 's,/usr/bin/cri-dockerd,/usr/local/bin/cri-dockerd,' /etc/systemd/system/cri-docker.service
* systemctl daemon-reload
* systemctl enable cri-docker.service
* systemctl enable --now cri-docker.socket

**---**

* now go to root user “ cd ~”
* execute below commands
* Installing kubadm, kubectl, kubelet on all machines
* sudo apt-get update
* sudo apt-get install -y apt-transport-https ca-certificates curl
* curl -fsSL https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-archive-keyring.gpg
* echo "deb [signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list
* sudo apt-get update
* sudo apt-get install -y kubelet kubeadm kubectl
* sudo apt-mark hold kubelet kubeadm kubectl
* Now create a cluster from a master node
* use the command kubeadm init --pod-network-cidr "10.244.0.0/16" --cri-socket "unix:///var/run/cri-dockerd.sock"
* Now exit from root user and run below command as normal user in master node
* mkdir -p $HOME/.kube
* sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
* sudo chown $(id -u):$(id -g) $HOME/.kube/config
* install flannel kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml

Now to add worker nodes to master node

* In worker node as a root user execute below command
* kubeadm join 172.31.14.184:6443 --token bxshtg.51w2pxgp9u3jqtr6 \

--cri-socket "unix:///var/run/cri-dockerd.sock" --discovery-token-ca-cert-hash sha256:8593ab70510b50f84c4d7ace392d8e39c3e0860258fdbcade1b2ff8a9814f714

Commands in kuberneties

* kubectl get nodes (to see the list of nodes in the cluster)
* kubectl get nodes -w (to see the changes happening)
* kubectl api-resources (to see the list and short names of resources)
* kubectl apply -f <name-of-manifestfile> (to create a pod)
* kubectl get pods (to get list of pods)
* kubectl get pods –o wide (this will give more information about pods)
* kubectl describe pods <name-of-pods>
* kubectl delete –f <name-of-pod> (to delete the pod)
* kubectl delete pods <name-of-pod> (to delete pod)
* kubectl exec <name-of-pod> -- <command> (to execute command inside a containers>
* kubectl exec <name-of-pod> -it -- /bin/sh (to get inside a container)
* kubectl get rs (to see the list of replica sets)
* kubectl get po --show-labels (to see the labels of the pods)
* kubectl get po --selector “<label>” --show-labels (to see thepods with specific labels )

Alias

alias kgp=‘kubectl get pods ’

alias kgn=‘kubectl get nodes ’

alias kar=’ kubectl api-resources ’

alias ka=’ kubectl apply –f ’

alias kdp=’ kubectl describe pods ’

alias kd=’ kubectl delete –f ‘

alias krs=’kubectl rollout status deployments/’

alias krh=’kubectl rollout history deployments/’

alias kdd=’kubectl describe deploy ‘