Multi-Master setup using Kubeadm

We need 4 servers.

1 Ec2 – Haproxy will act as load balancer.

2 Ec2 – Will act as master nodes.

1 Ec2- Will act as worker nodes.

Install and configure HA proxy

Text

Description automatically generated

Check the status

Graphical user interface, text

Description automatically generated

Edit the configuration file

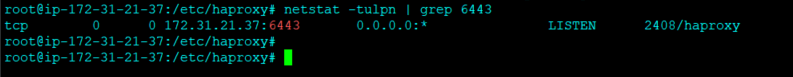
Text

Description automatically generated

Reload the services



Check 6443 port is listening



Below needs to be on Master and worker nodes

Check n/w connectivity between all 3 nodes

Graphical user interface, text

Description automatically generated

Disable firewall in all 3 machines

Text

Description automatically generated

Disable swap in all 3 machines

Text

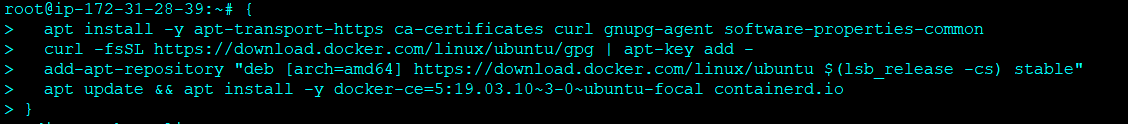
Description automatically generated

Let IP tables see bridge traffic

Text

Description automatically generated

Install Docker



Make the cgroup driver as systemd for container runtime and kubelet

Text

Description automatically generated

Restart docker

A picture containing text

Description automatically generated

Add repository for k8s

A screenshot of a computer

Description automatically generated

Install kubeadm/kubelet/kubectl

Graphical user interface, text

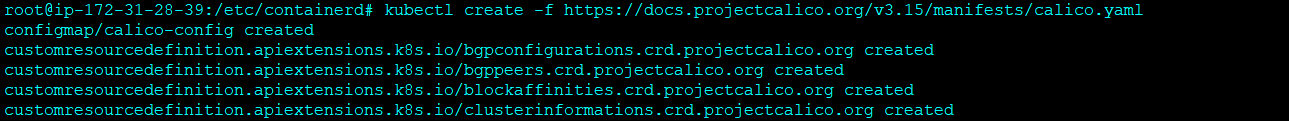
Description automatically generated

Initialize the cluster in any one of the master nodes

Graphical user interface, application

Description automatically generated

Deploy the calico network on any of the master nodes

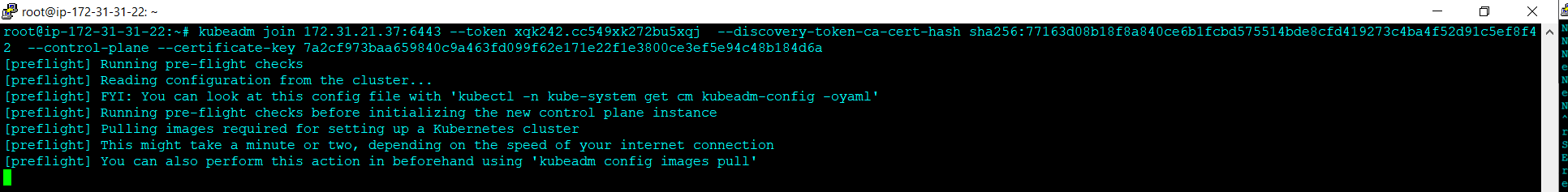


Join the worker and additional master nodes

Text

Description automatically generated

Join the 2nd master nodes

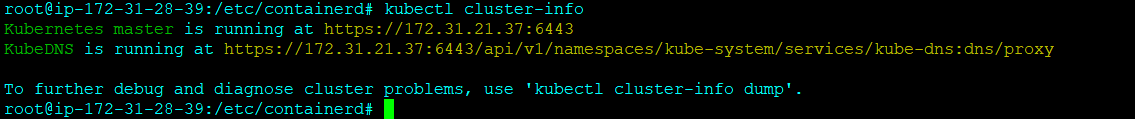


Join the worker nodes

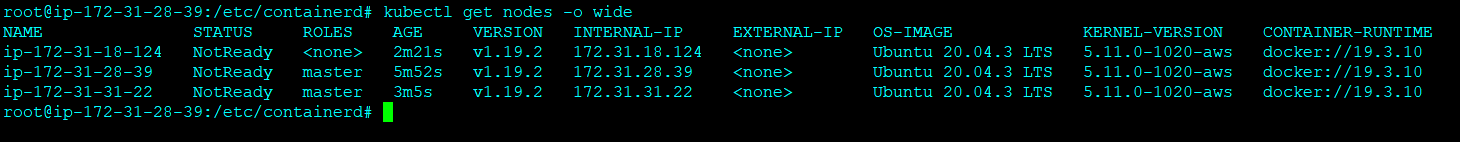
Graphical user interface, text

Description automatically generated

Check cluster status



Check the cluster node status. When you have both docker/containerd runtime, docker takes the precedence



Look the HA proxy logs

Graphical user interface, text, application

Description automatically generated

Check the POD status in

Graphical user interface

Description automatically generated