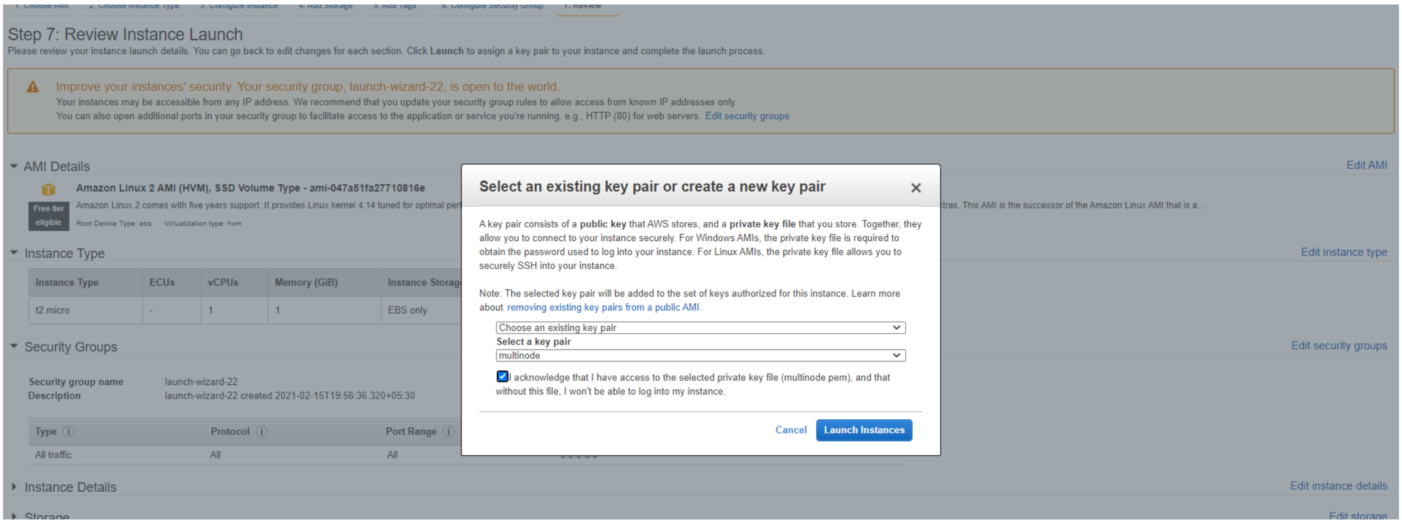
configuring one more as worker node..



1. Choose AMI · 2. Choose Instance Type · 3. Configure Instance · 4. Add Storage · 5. Add Tags · 6. Configure Security Group · 7. Review

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

Improve your instances' security. Your security group, launch-wizard-22, is open to the world.
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.
You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups

AMI Details — Edit AMI

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-047a51fa27710816e
Free tier eligible
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal per… …tras. This AMI is the successor of the Amazon Linux AMI that is a...
Root Device Type: ebs Virtualization type: hvm

Instance Type — Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storag
t2.micro	-	1	1	EBS only

Security Groups — Edit security groups

Security group name: launch-wizard-22
Description: launch-wizard-22 created 2021-02-15T19:56:36.320+05:30

Type	Protocol	Port Range
All traffic	All	All

Instance Details — Edit instance details

Storage — Edit storage

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

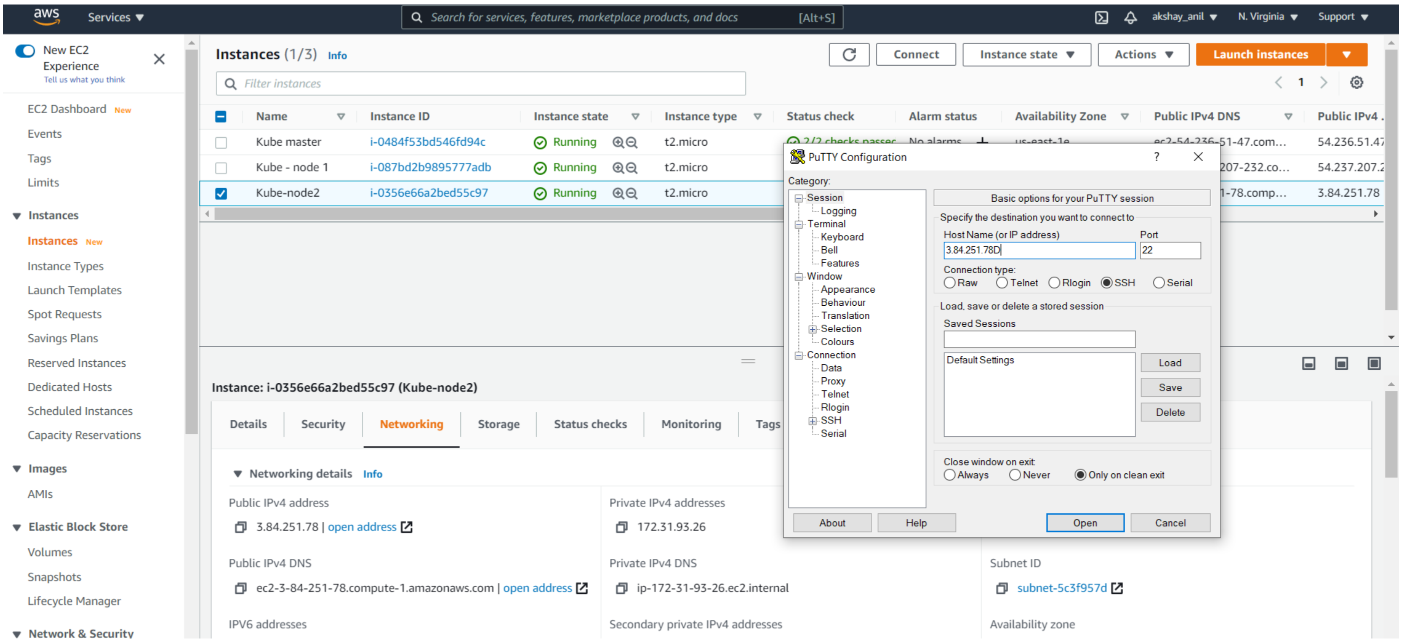
Choose an existing key pair

Select a key pair: multinode

☑ I acknowledge that I have access to the selected private key file (multinode.pem), and that without this file, I won't be able to log into my instance.

launching one more….

- ⇨ connect to putty



Instances (1/3) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 …
□	Kube master	i-0484f53bd546fd94c	Running	t2.micro	2/2 checks passed	No alarms	us-east-1e	ec2-54-236-51-47.com…	54.236.51.47
□	Kube - node 1	i-087bd2b9895777adb	Running	t2.micro				…207-232.co…	54.237.207.2
☑	Kube-node2	i-0356e66a2bed55c97	Running	t2.micro				…1-78.comp…	3.84.251.78

Instance: i-0356e66a2bed55c97 (Kube-node2)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Networking details Info

Public IPv4 address: 3.84.251.78 | open address

Private IPv4 addresses: 172.31.93.26

Public IPv4 DNS: ec2-3-84-251-78.compute-1.amazonaws.com | open address

Private IPv4 DNS: ip-172-31-93-26.ec2.internal

Subnet ID: subnet-5c3f957d

IPV6 addresses

Secondary private IPv4 addresses

Availability zone

PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 3.84.251.78D

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

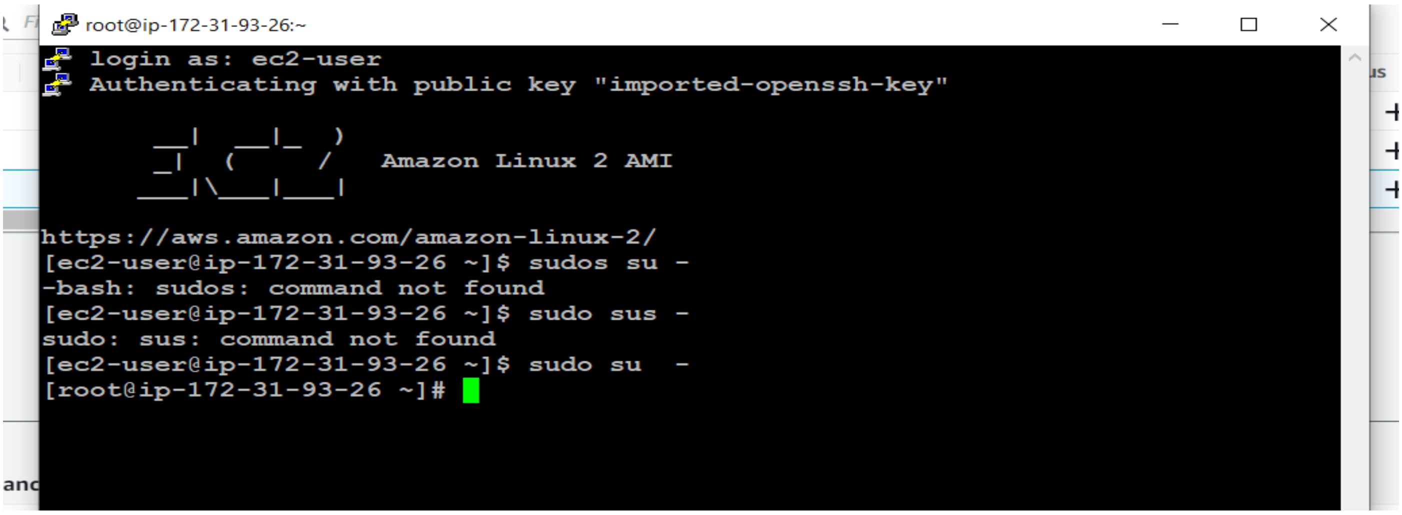
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

- ⇨ login into root

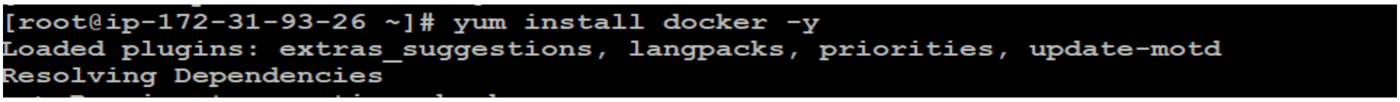


root@ip-172-31-93-26:~

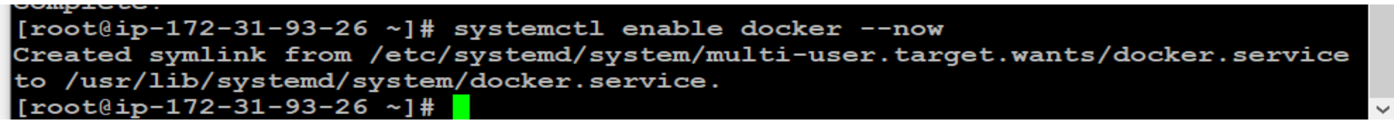
```
login as: ec2-user
Authenticating with public key "imported-openssh-key"

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-93-26 ~]$ sudos su -
-bash: sudos: command not found
[ec2-user@ip-172-31-93-26 ~]$ sudo sus -
sudo: sus: command not found
[ec2-user@ip-172-31-93-26 ~]$ sudo su  -
[root@ip-172-31-93-26 ~]#
```

installation of docker

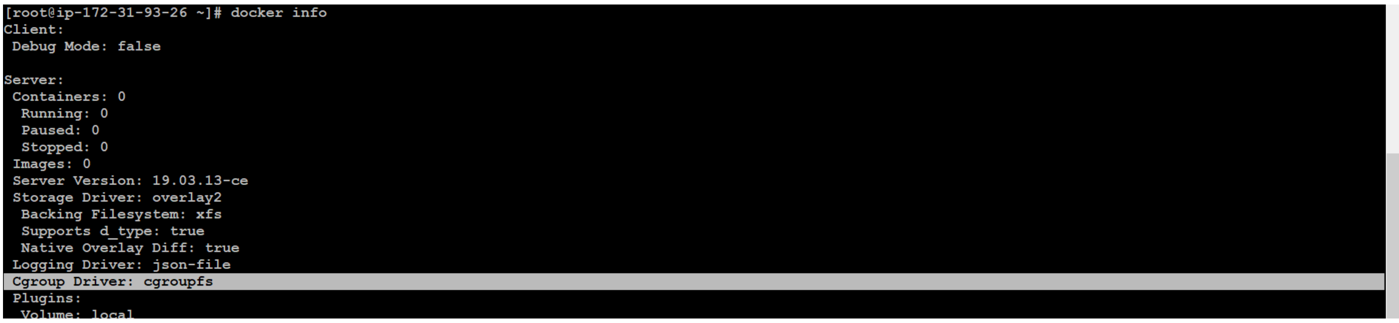
1. yum install docker -y
2. systemctl enable docker --now
3. docker info



```
[root@ip-172-31-93-26 ~]# yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
```



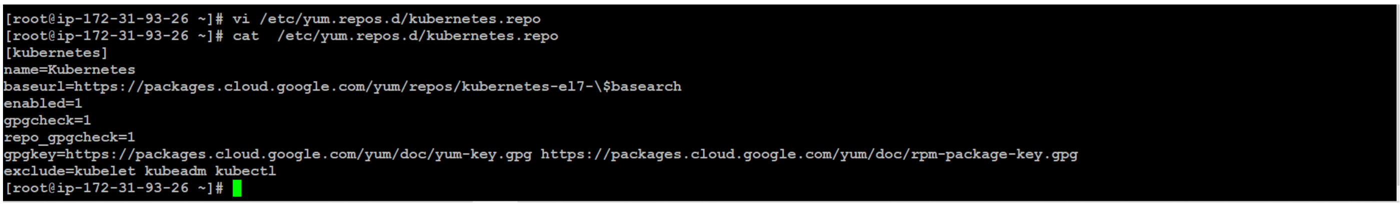
```
Complete!
[root@ip-172-31-93-26 ~]# systemctl enable docker --now
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service
to /usr/lib/systemd/system/docker.service.
[root@ip-172-31-93-26 ~]#
```



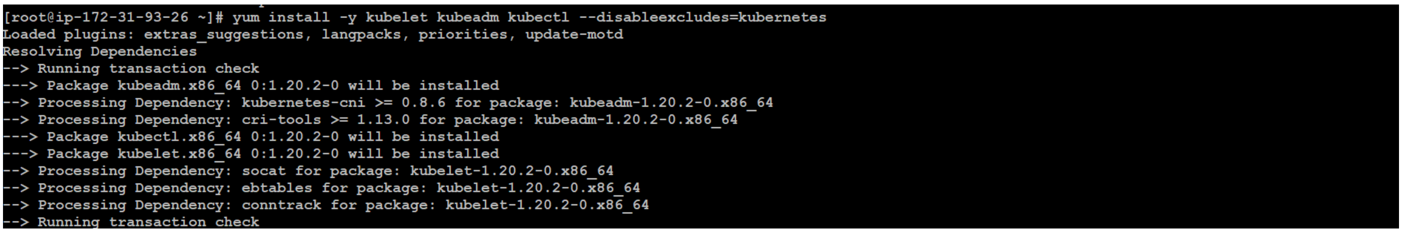
```
[root@ip-172-31-93-26 ~]# docker info
Client:
 Debug Mode: false

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 19.03.13-ce
 Storage Driver: overlay2
  Backing Filesystem: xfs
  Supports d_type: true
  Native Overlay Diff: true
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Plugins:
  Volume: local
```

installation of kubeadm

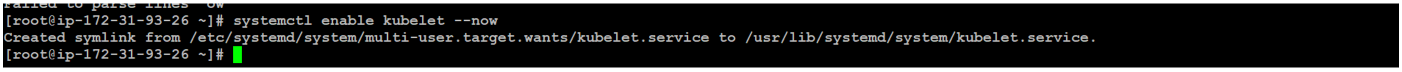
1. yum install kubeadm - fails
2. vi /etc/yum.repos.d/kubernetes.repo   - need to setup the repo
3. yum repolist
4. yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
5. systemctl status kubelet
6. systemctl enable kubelet –now



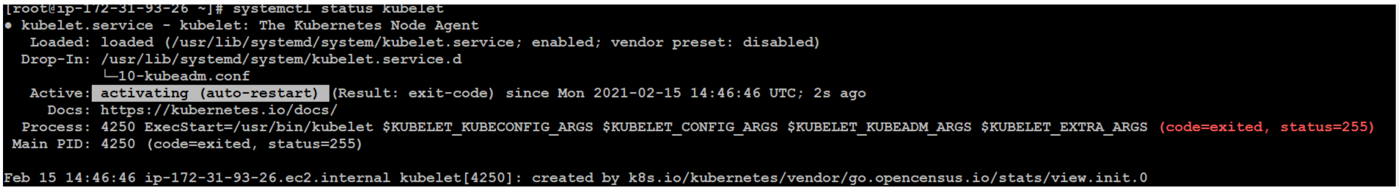
```
[root@ip-172-31-93-26 ~]# vi /etc/yum.repos.d/kubernetes.repo
[root@ip-172-31-93-26 ~]# cat  /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-\$basearch
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
exclude=kubelet kubeadm kubectl
[root@ip-172-31-93-26 ~]#
```



```
[root@ip-172-31-93-26 ~]# yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package kubeadm.x86_64 0:1.20.2-0 will be installed
--> Processing Dependency: kubernetes-cni >= 0.8.6 for package: kubeadm-1.20.2-0.x86_64
--> Processing Dependency: cri-tools >= 1.13.0 for package: kubeadm-1.20.2-0.x86_64
---> Package kubectl.x86_64 0:1.20.2-0 will be installed
---> Package kubelet.x86_64 0:1.20.2-0 will be installed
--> Processing Dependency: socat for package: kubelet-1.20.2-0.x86_64
--> Processing Dependency: ebtables for package: kubelet-1.20.2-0.x86_64
--> Processing Dependency: conntrack for package: kubelet-1.20.2-0.x86_64
--> Running transaction check
```



```
failed to parse lines  ow
[root@ip-172-31-93-26 ~]# systemctl enable kubelet --now
Created symlink from /etc/systemd/system/multi-user.target.wants/kubelet.service to /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-93-26 ~]#
```

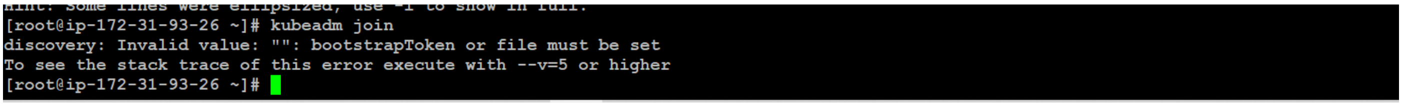


```
[root@ip-172-31-93-26 ~]# systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; vendor preset: disabled)
  Drop-In: /usr/lib/systemd/system/kubelet.service.d
           └─10-kubeadm.conf
   Active: activating (auto-restart) (Result: exit-code) since Mon 2021-02-15 14:46:46 UTC; 2s ago
     Docs: https://kubernetes.io/docs/
  Process: 4250 ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS $KUBELET_CONFIG_ARGS $KUBELET_KUBEADM_ARGS $KUBELET_EXTRA_ARGS (code=exited, status=255)
 Main PID: 4250 (code=exited, status=255)

Feb 15 14:46:46 ip-172-31-93-26.ec2.internal kubelet[4250]: created by k8s.io/kubernetes/vendor/go.opencensus.io/stats/view.init.0
```

..

now, we have to join this to master

Kubeadm join.



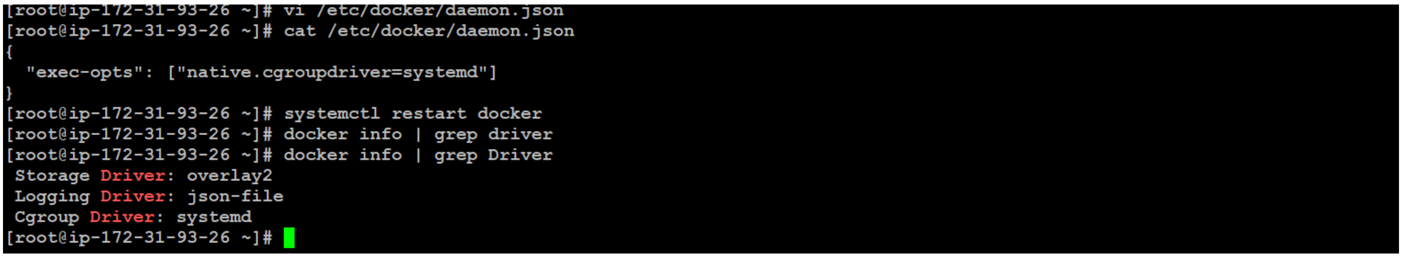
```
Hint: Some lines were ellipsized, use -l to show in full.
[root@ip-172-31-93-26 ~]# kubeadm join
discovery: Invalid value: "": bootstrapToken or file must be set
To see the stack trace of this error execute with --v=5 or higher
[root@ip-172-31-93-26 ~]# 
```

right now, it fails, it requires token from master,

since we know, preflight check error will come,

1st error: [WARNING IsDockerSystemdCheck]: we have to use system instead of cgroups.

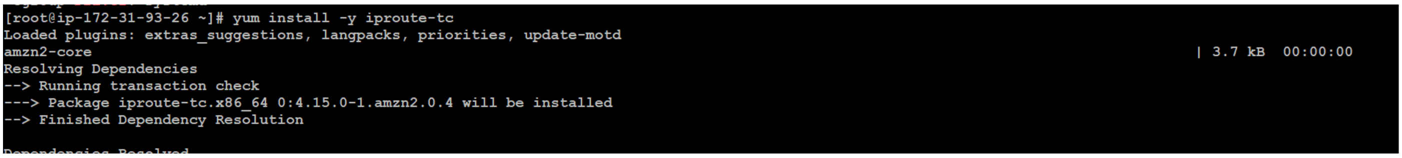
```
vi /etc/docker/daemon.json
      {
        "exec-opts": ["native.cgroupdriver=systemd"]
      }
cat /etc/docker/daemon.json
systemctl restart docker
docker info | grep Driver
```



```
[root@ip-172-31-93-26 ~]# vi /etc/docker/daemon.json
[root@ip-172-31-93-26 ~]# cat /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
[root@ip-172-31-93-26 ~]# systemctl restart docker
[root@ip-172-31-93-26 ~]# docker info | grep driver
[root@ip-172-31-93-26 ~]# docker info | grep Driver
 Storage Driver: overlay2
 Logging Driver: json-file
 Cgroup Driver: systemd
[root@ip-172-31-93-26 ~]# 
```

2nd error: [WARNING FileExisting-tc]: tc not found in system path

```
yum install -y iproute-tc
```



```
 Cgroup Driver: systemd
[root@ip-172-31-93-26 ~]# yum install -y iproute-tc
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                                                          | 3.7 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package iproute-tc.x86_64 0:4.15.0-1.amzn2.0.4 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

3rd error: FileContent—proc-sys-net-bridge-bridge-nf-call-iptables

vim /etc/sysctl.d/k8s.conf

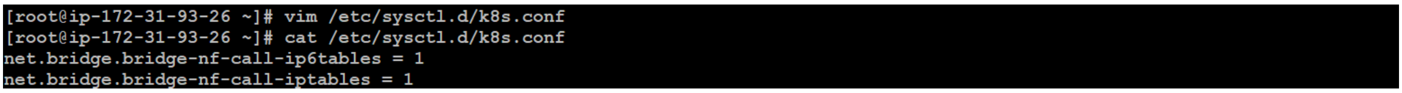
net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

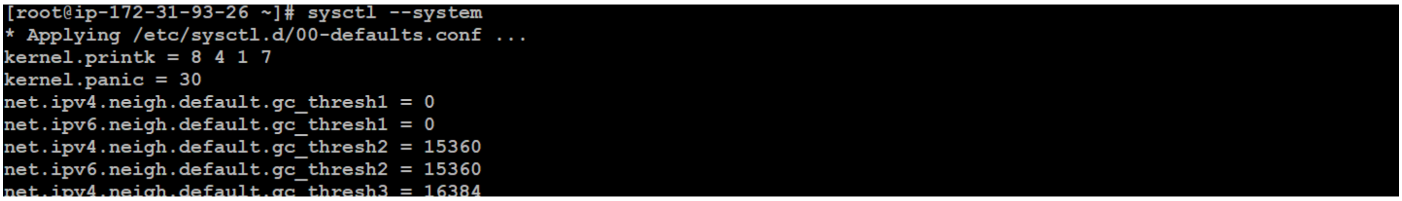
cat /etc/sysctl.d/k8s.conf

sysctl –system

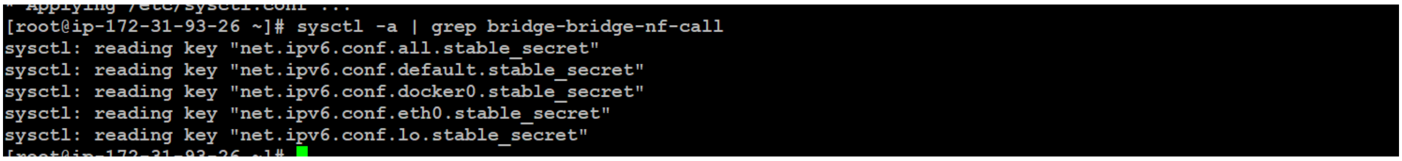
sysctl -a | grep bridge-bridge-nf-call



```
[root@ip-172-31-93-26 ~]# vim /etc/sysctl.d/k8s.conf
[root@ip-172-31-93-26 ~]# cat /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
```



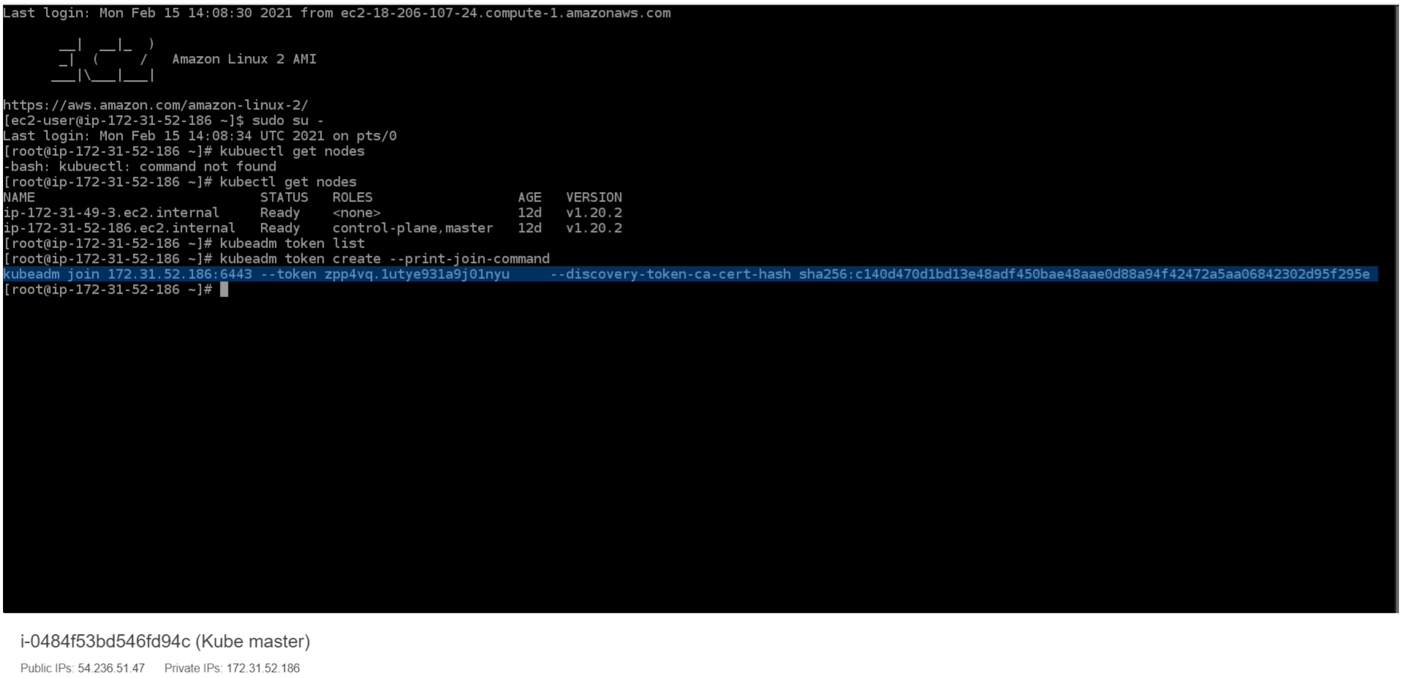
```
[root@ip-172-31-93-26 ~]# sysctl --system
* Applying /etc/sysctl.d/00-defaults.conf ...
kernel.printk = 8 4 1 7
kernel.panic = 30
net.ipv4.neigh.default.gc_thresh1 = 0
net.ipv6.neigh.default.gc_thresh1 = 0
net.ipv4.neigh.default.gc_thresh2 = 15360
net.ipv6.neigh.default.gc_thresh2 = 15360
net.ipv4.neigh.default.gc_thresh3 = 16384
```



```
* Applying /etc/sysctl.conf ...
[root@ip-172-31-93-26 ~]# sysctl -a | grep bridge-bridge-nf-call
sysctl: reading key "net.ipv6.conf.all.stable_secret"
sysctl: reading key "net.ipv6.conf.default.stable_secret"
sysctl: reading key "net.ipv6.conf.docker0.stable_secret"
sysctl: reading key "net.ipv6.conf.eth0.stable_secret"
sysctl: reading key "net.ipv6.conf.lo.stable_secret"
[root@ip-172-31-93-26 ~]#
```

now, we can join this worker to the master

go to master create the token



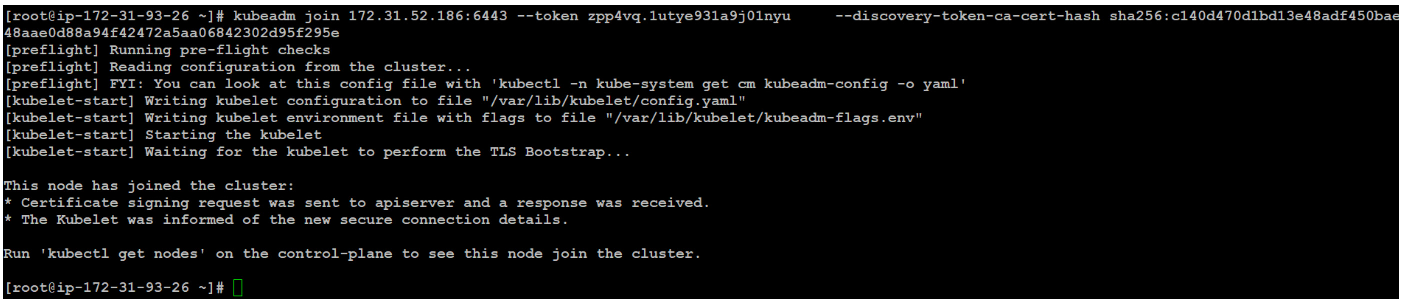
```
Last login: Mon Feb 15 14:08:30 2021 from ec2-18-206-107-24.compute-1.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-52-186 ~]$ sudo su -
Last login: Mon Feb 15 14:08:34 UTC 2021 on pts/0
[root@ip-172-31-52-186 ~]# kubuectl get nodes
-bash: kubuectl: command not found
[root@ip-172-31-52-186 ~]# kubectl get nodes
NAME                           STATUS   ROLES                  AGE   VERSION
ip-172-31-49-3.ec2.internal    Ready    <none>                 12d   v1.20.2
ip-172-31-52-186.ec2.internal  Ready    control-plane,master   12d   v1.20.2
[root@ip-172-31-52-186 ~]# kubeadm token list
[root@ip-172-31-52-186 ~]# kubeadm token create --print-join-command
kubeadm join 172.31.52.186:6443 --token zpp4vq.1utye931a9j01nyu     --discovery-token-ca-cert-hash sha256:c140d470d1bd13e48adf450bae48aae0d88a94f42472a5aa06842302d95f295e
[root@ip-172-31-52-186 ~]#
```

i-0484f53bd546fd94c (Kube master)

Public IPs: 54.236.51.47   Private IPs: 172.31.52.186

come to worker node again.

enter the above token directly in worker node.

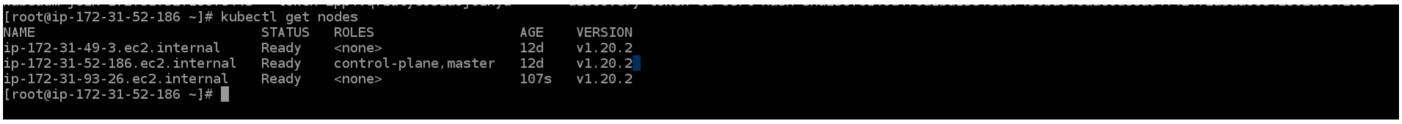


```
[root@ip-172-31-93-26 ~]# kubeadm join 172.31.52.186:6443 --token zpp4vq.1utye931a9j01nyu     --discovery-token-ca-cert-hash sha256:c140d470d1bd13e48adf450bae
48aae0d88a94f42472a5aa06842302d95f295e
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

[root@ip-172-31-93-26 ~]#
```

verifying…

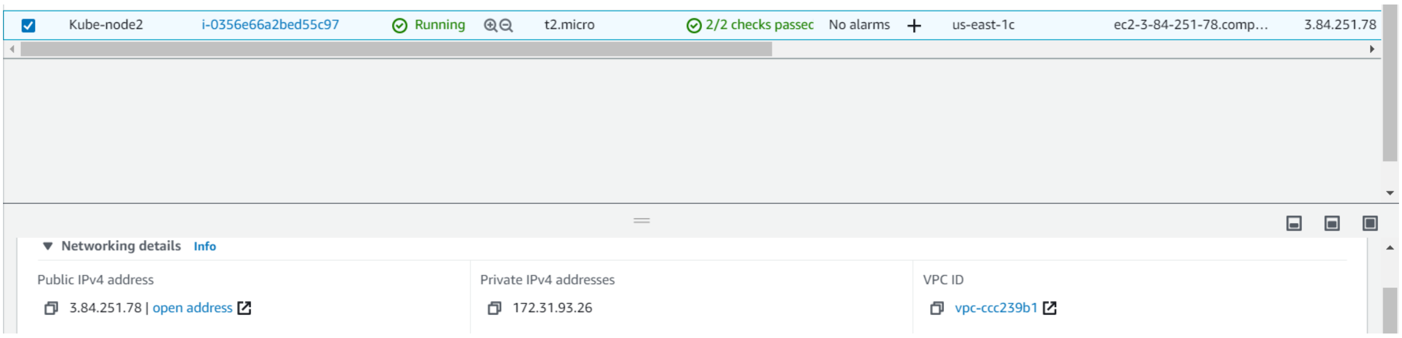
go to master

kubectl get node



```
[root@ip-172-31-52-186 ~]# kubectl get nodes
NAME                           STATUS   ROLES                  AGE    VERSION
ip-172-31-49-3.ec2.internal    Ready    <none>                 12d    v1.20.2
ip-172-31-52-186.ec2.internal  Ready    control-plane,master   12d    v1.20.2
ip-172-31-93-26.ec2.internal   Ready    <none>                 107s   v1.20.2
[root@ip-172-31-52-186 ~]#
```

Ip of node 2 is now mention as node in mater



☑ Kube-node2 i-0356e66a2bed55c97 Running t2.micro 2/2 checks passec No alarms + us-east-1c ec2-3-84-251-78.comp… 3.84.251.78

Public IPv4 address: 3.84.251.78 | open address

Private IPv4 addresses: 172.31.93.26

VPC ID: vpc-ccc239b1

done………………………………