Installing a User-Provisioned Openshift cluster on bare metal

Pre-requisites:

8 Machines:

RHEL image:

MACHINE	NAME	TOTAL
Jumphost	bastion.vcet.citiuscloud.com	1
Nginx	{nginx} *(Load balance)*	1

Rest all are RHCOS image:

MACHINE	NAME	TOTAL
Master	master1.vcet.citiuscloud.com master2.vcet.citiuscloud.com master3.vcet.citiuscloud.com	3
Worker	worker1.vcet.citiuscloud.com worker2.vcet.citiuscloud.com	2
Bootstrap	bootstrap.vcet.citiuscloud.com	1 *(Temporary machine for booting)*

nmap -to get free ip

```
nmap -v -sn 10.48.70.0/23
```

Copy the free ips with big range.

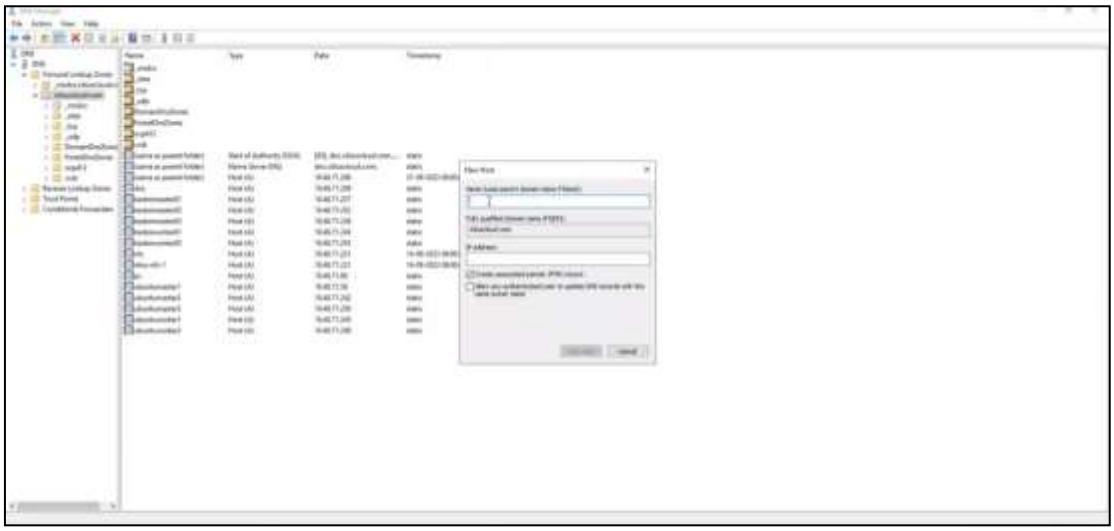
Output:

```
Nmap scan report for 10.48.71.184 [host down]
Nmap scan report for 10.48.71.185 [host down]
Nmap scan report for 10.48.71.186 [host down]
Nmap scan report for 10.48.71.187 [host down]
Nmap scan report for 10.48.71.188 [host down]
Nmap scan report for 10.48.71.189 [host down]
Nmap scan report for 10.48.71.190 [host down]
Nmap scan report for 10.48.71.191 [host down]
Nmap scan report for 10.48.71.192 [host down]
Nmap scan report for 10.48.71.193 [host down]
Nmap scan report for 10.48.71.194 [host down]
Nmap scan report for 10.48.71.195 [host down]
```

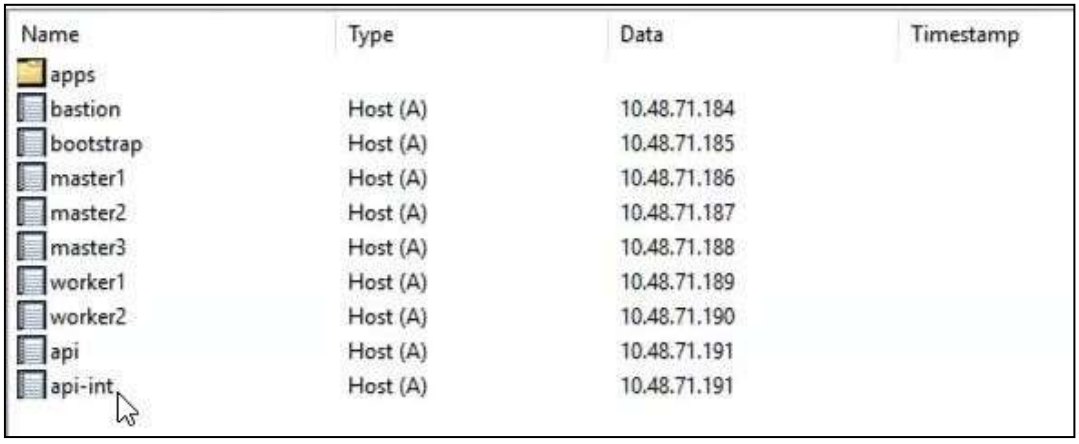
ENTRIES IN DNS

10.48.71.184 | bastion.vcet.citiuscloud.com
10.48.71.185 | bootstrap.vcet.citiuscloud.com
10.48.71.186 | master1.vcet.citiuscloud.com
10.48.71.187 | master2.vcet.citiuscloud.com
10.48.71.188 | master3.vcet.citiuscloud.com
10.48.71.189 | worker1.vcet.citiuscloud.com
10.48.71.190 | worker2.vcet.citiuscloud.com
10.48.71.191 | *.apps.vcet -> type this on DNS server *(nginx)*
10.48.71.191 | api.vcet -> type this on DNS server *(nginx)*
10.48.71.191 | api-int.vcet -> type this on DNS server *(nginx)*

1. Login to your DNS Server and go to DNS Manager
2. Right click on the domain name and click on add host(A AAAA)
3. Enter all the entries of all the machines.



4. At last, the DNS Manager will consist of these many entries.

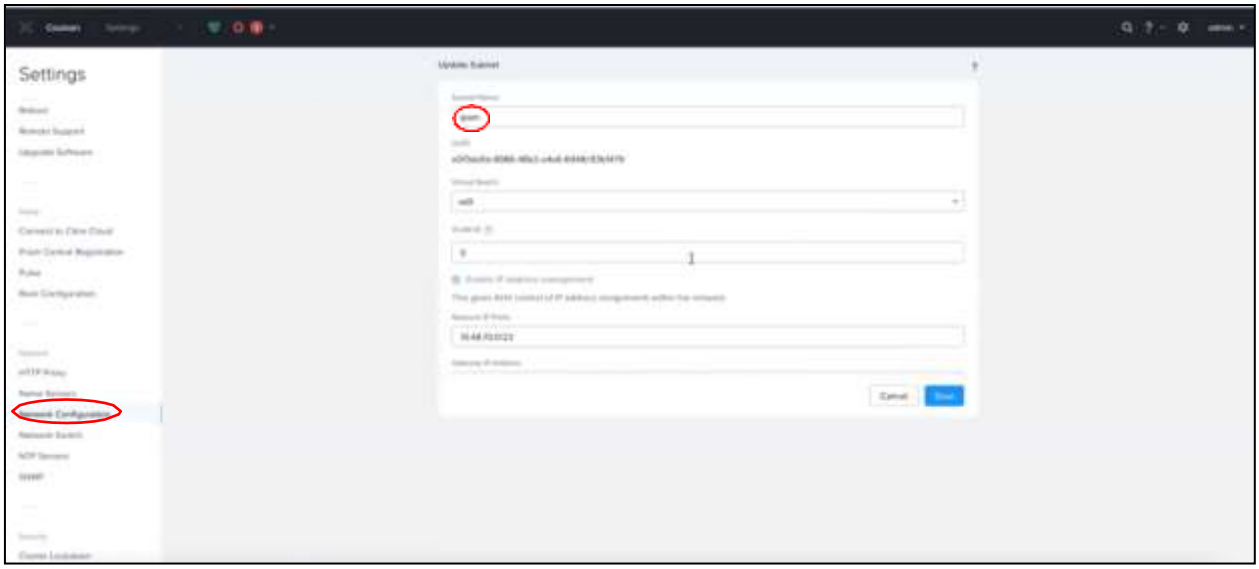


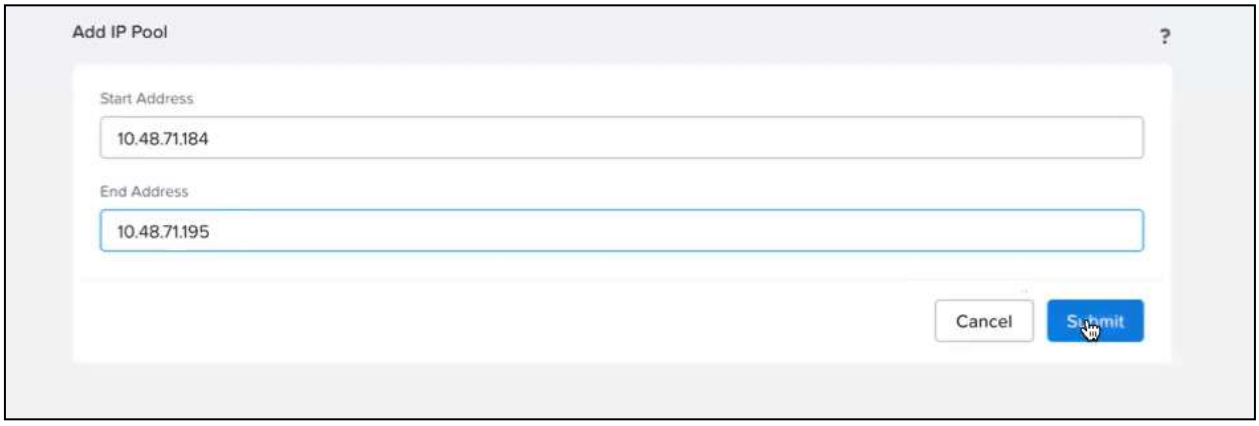
Name	Type	Data	Timestamp
apps			
bastion	Host (A)	10.48.71.184	
bootstrap	Host (A)	10.48.71.185	
master1	Host (A)	10.48.71.186	
master2	Host (A)	10.48.71.187	
master3	Host (A)	10.48.71.188	
worker1	Host (A)	10.48.71.189	
worker2	Host (A)	10.48.71.190	
api	Host (A)	10.48.71.191	
api-int	Host (A)	10.48.71.191	

Make virtual machines on prism dashboard

Let's configure ipam(IP Address Management) first,

1. Go to prism Central and login using your credentials.
2. Once logged in, click on settings and go to Network Configuration.
3. Click on edit ipam and scroll down to create new pool.
4. Click on create pool and add the Ip range of your cluster and click on submit.



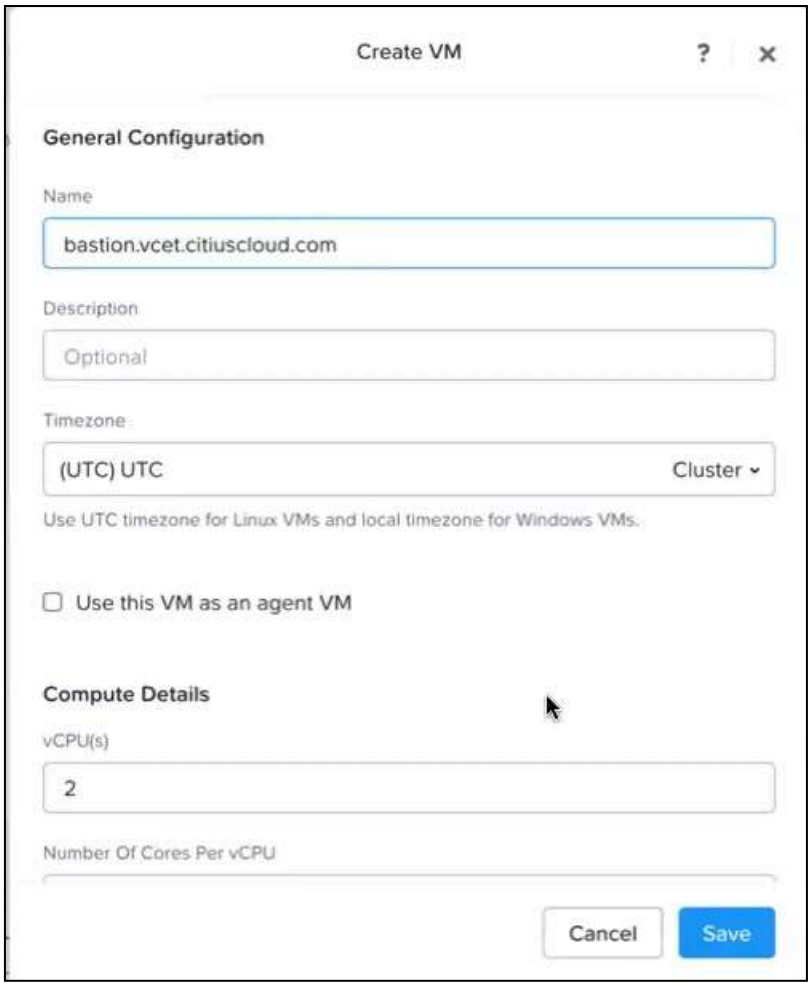


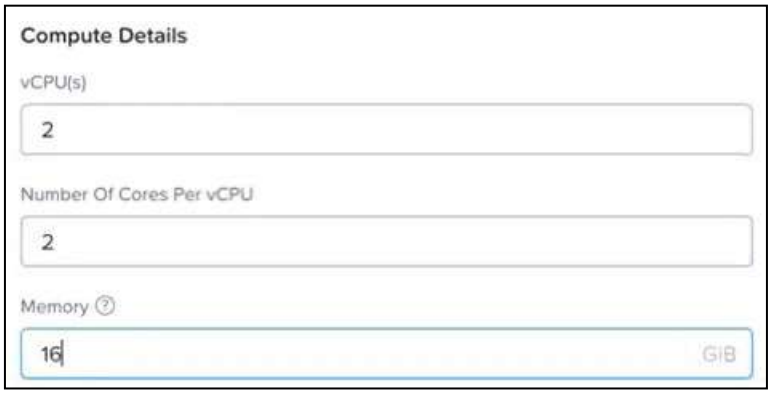
Creating virtual machines on Prism Central

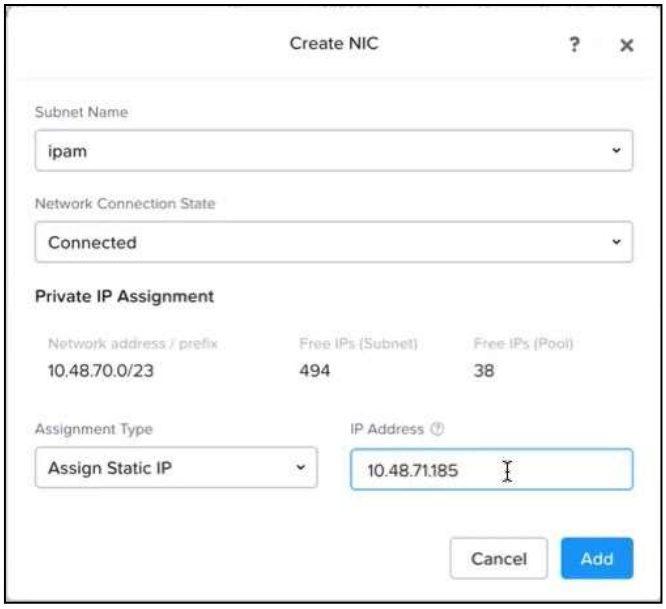
1. Click on Home, drop down menu will be appeared
2. Click on VM and click on create VM

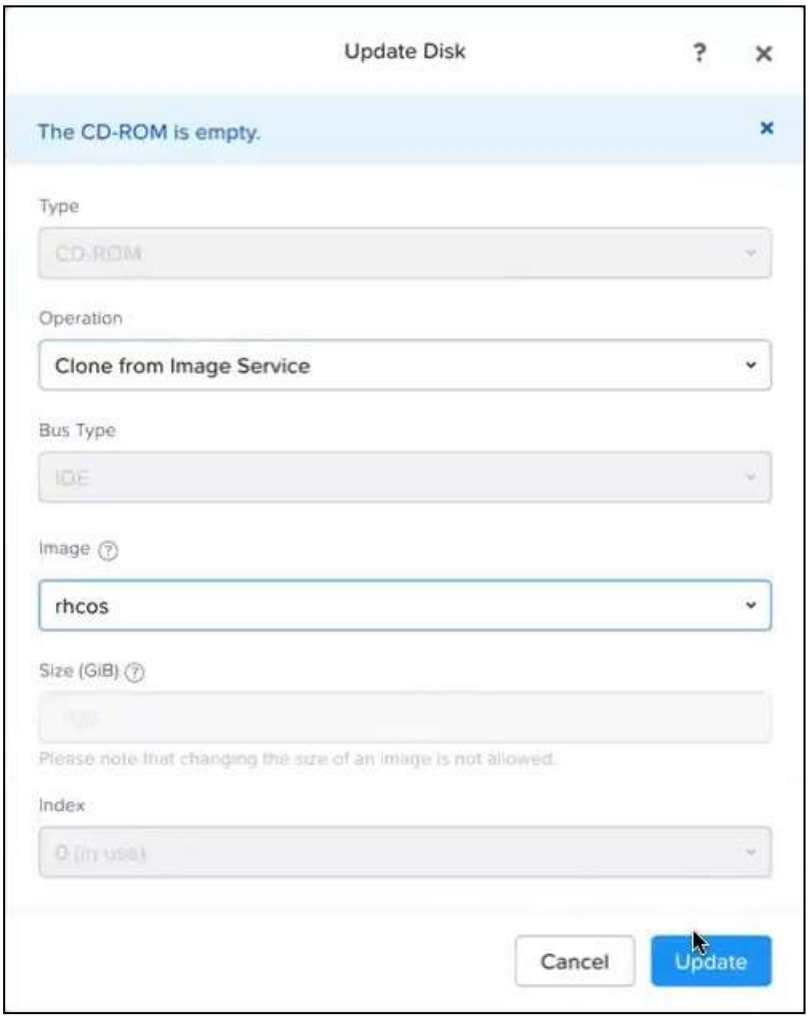
Role	Operating System	vCPU	Cores per Vcpu	Memory	Storage
Bastion	RHEL	2	2	16	250GB
Nginx	RHEL	2	2	16	250GB
Bootstrap	RHCOS	2	2	16	250GB
Master*3	RHCOS	2	2	16	250GB
Worker*2	RHCOS	2	2	16	250GB

- Firstly, we will make only two machines: Bastion and Bootstrap. (*Make sure you select the correct OS image for both the machines.*)



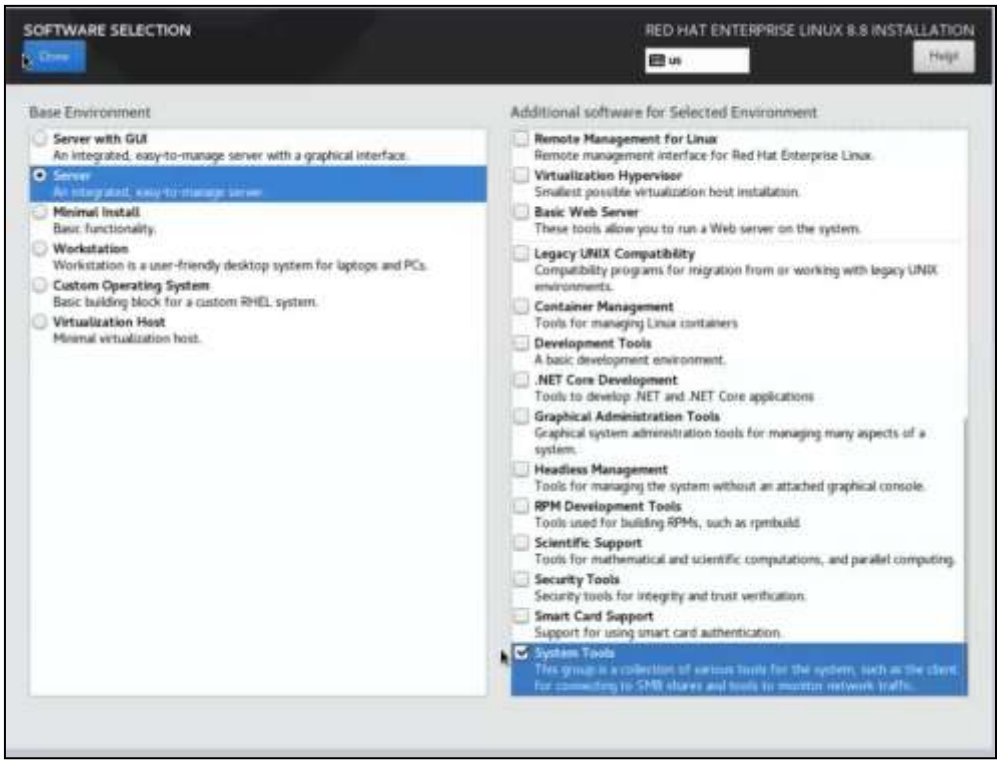

Compute Details
vCPU(s)
2
Number Of Cores Per vCPU
2
Memory
16
GiB


Create NIC
Subnet Name
ipam
Network Connection State
Connected
Private IP Assignment
Network address / prefix
10.48.70.0/23
Free IPs (Subnet)
494
Free IPs (Pool)
38
Assignment Type
Assign Static IP
IP Address
10.48.71.185
Cancel
Add

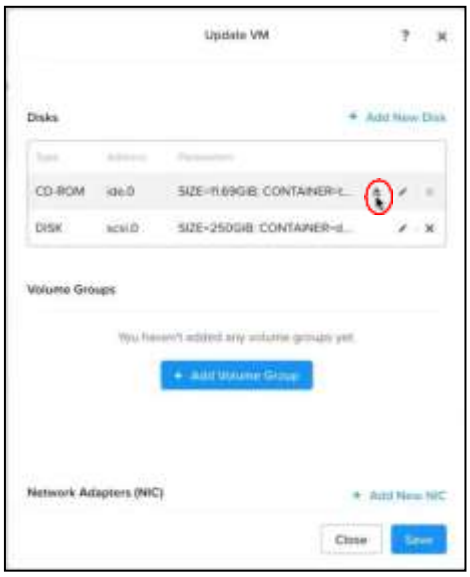


- Once both the machines are ready, clone bootstrap machine to create 3 master and 2 worker machines.
- After the machine creation, edit the name and assign required IPs to it.

Let’s now power on the bastion machine and do the necessary installation:



- Before rebooting unmount the disk.



- Now clone the Bastion machine for nginx machine

➢ Go to CLI and run the following commands:

```
ssh root@<bastion-ip>
cat /etc/resolv.conf
nsloopup
> master1        OR        > 10.48.71.186        **See if the ip is resolved in nslookup
```

Output:

```
[root@bastion ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search citiuscloud.com vcet.citiuscloud.com
nameserver 10.48.70.221
[root@bastion ~]# nslookup
> master1
Server:         10.48.70.221
Address:        10.48.70.221#53

Name:   master1.vcet.citiuscloud.com
Address: 10.48.71.186
> 10.48.71.191
191.71.48.10.in-addr.arpa       name = *.apps.vcet.citiuscloud.com.
191.71.48.10.in-addr.arpa       name = api.vcet.citiuscloud.com.
191.71.48.10.in-addr.arpa       name = api-int.vcet.citiuscloud.com.
```

#IF NOT: check for reverse backlookup or check if entries are correctly configured

ping all the machines and check if correct name is present in FQDN format

- Login to the nginx machine now

```
ssh root@<ngnix-ip>
```

Ping all the machines and check if correct name is present in FQDN format

- let's install nginx now:

```
subscription-manager register

subscription-manager auto-attach

yum install nginx
```

```
systemctl restart nginx

systemctl status nginx          #should be running
```

Generate self-signed key for nginx and stored in /etc/ssl/private/:

```
mkdir /etc/ssl/private

openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/nginx-selfsigned.key -out /etc/ssl/certs/nginx-selfsigned.crt
```

Output:

```
[root@nginx ~]# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/nginx-selfsigned.key -out /etc/ssl/certs/nginx-selfsigned.crt
Generating a RSA private key
.......+++++
...+++++
writing new private key to '/etc/ssl/private/nginx-selfsigned.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:IN
State or Province Name (full name) []:MH
Locality Name (eg, city) [Default City]:THANE
Organization Name (eg, company) [Default Company Ltd]:CITIUSCLOUD
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:
Email Address []:
```

Then create a file self-signed.conf under /etc/nginx/snippets, and mention nginx Self-signed.crt and nginx-selfsigned.key path inside it.

```
mkdir /etc/nginx/snippets
vi /etc/nginx/snippets/self-signed.conf
ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
```

```
ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
~
~
```

```
vi /etc/nginx/snippets/ssl-params.conf
```

#paste on notepad and format the documents

#Make sure proper intend are followed, no extra space must be present

```
ssl_protocols TLSv1.2;
ssl_prefer_server_ciphers on;
ssl_dhparam /etc/ssl/certs/dhparam.pem;
ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384;
ssl_ecdh_curve secp384rl; # Requires nginx > 1.1.0
ssl_session_timeout 10m;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off; # Requires nginx >= 1.5.9
# ssl stapling on; # Requires nginx > 1.3.7
# ssl_stapling_ verify on; # Requires nginx => 1.3.7 resolver 8.8.8.8 8.8.4.4 valid 300s;
resolver_timeout 5s;
add_header X-Frame-Options DENY;
add_header X-Content-Type-Options nosniff;
add_header X-XSS-Protection "1: mode=block";
```

vim /etc/nginx/nginx.conf *#edit the file*

#paste on notepad and make changes in the file and paste it at the end of the file

#Make sure proper intend are followed, no extra space must be present

```
stream {
    server
        {
        listen 6443;
        proxy_pass openshift_api_server;
        ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
        ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers HIGH:!aNULL:MD5;
    }
    server
        {
        listen 22623;
        proxy_pass machine_config_server;
        ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
        ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers HIGH:!aNULL:MD5;
    }
    server
        {
```

```
        listen 80;
        proxy_pass ingress_http;
        ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
        ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers HIGH:!aNULL:MD5;
    }
    server
        {
        listen 443;
        proxy_pass ingress_https;
        ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
        ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers HIGH:!aNULL:MD5;
    }
    upstream openshift_api_server
        {
        server bootstrap.vcet.citiuscloud.com:6443;
        server master1.vcet.citiuscloud.com:6443;
        server master2.vcet.citiuscloud.com:6443;
        server master3.vcet.citiuscloud.com:6443;
        server worker1.vcet.citiuscloud.com:6443;
        server worker2.vcet.citiuscloud.com:6443;
    }
    upstream machine_config_server
        {
        server bootstrap.vcet.citiuscloud.com:22623;
        server master1.vcet.citiuscloud.com:22623;
        server master2.vcet.citiuscloud.com:22623;
```

```
        server  master3.vcet.citiuscloud.com:22623;
        server  worker1.vcet.citiuscloud.com:22623;
        server worker2.vcet.citiuscloud.com:22623;
    }
     upstream ingress_http
        {
        server bootstrap.vcet.citiuscloud.com:80;
        server master1.vcet.citiuscloud.com:80;
        server master2.vcet.citiuscloud.com:80;
        server master3.vcet.citiuscloud.com:80;
        server worker1.vcet.citiuscloud.com:80;
        server worker2.vcet.citiuscloud.com:80;
    }
     upstream ingress_https
        {
        server bootstrap.vcet.citiuscloud.com:443;
        server master1.vcet.citiuscloud.com:443;
        server master2.vcet.citiuscloud.com:443;
        server master3.vcet.citiuscloud.com:443;
        server worker1.vcet.citiuscloud.com:443;
        server worker2.vcet.citiuscloud.com:443;
    }
}
```

```
#
#       error_page 404 /404.html;
#           location = /40x.html {
#       }
#
#       error_page 500 502 503 504 /50x.html;
#           location = /50x.html {
#       }
#   }
}
stream {
        server
                {
                listen 6443;
                proxy_pass openshift_api_server;
                ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
                ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
                ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
                ssl_ciphers HIGH:!aNULL:MD5;
      }
        server
                {
                listen 22623;
                proxy_pass machine_config_server;
                ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
                ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
```

sed -i 's/<old-text>/<new-text>/g' <filename> # to change certain words in the file by using sed command

setenforce 0

systemctl stop firewalld.service

systemctl disable firewalld.service

systemctl restart nginx.service

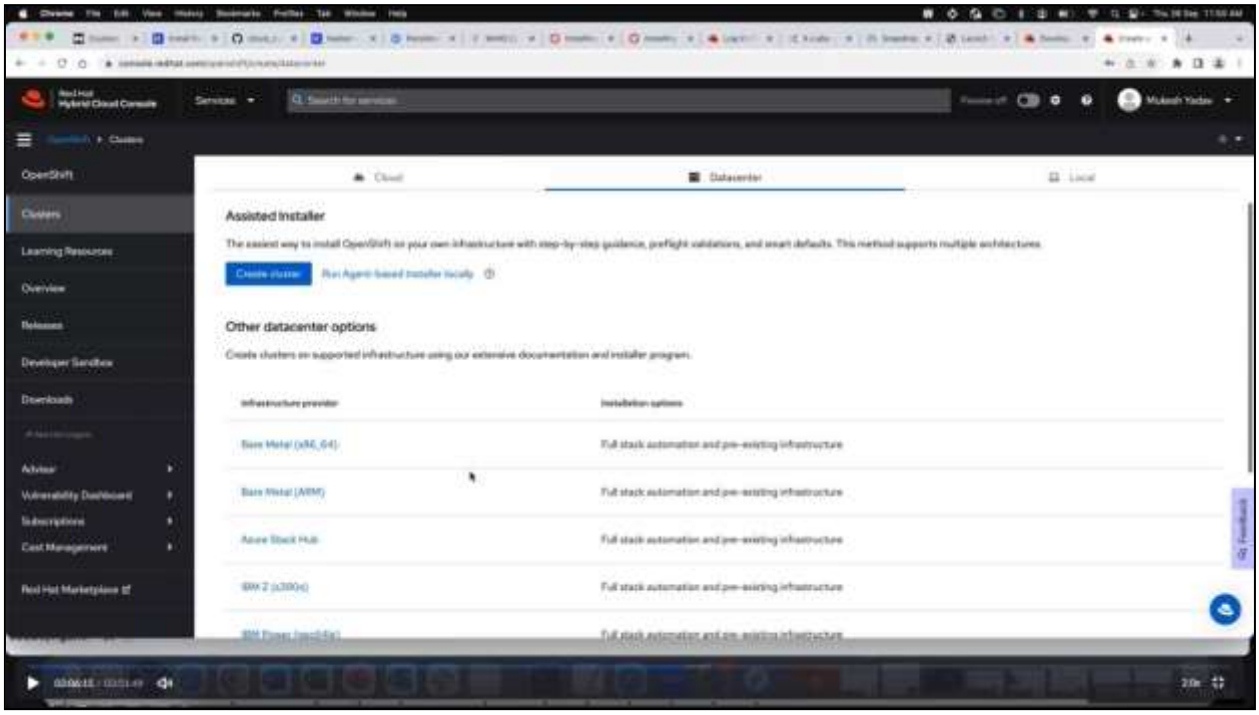
netstat -tulnp *#to see if ports are listening*

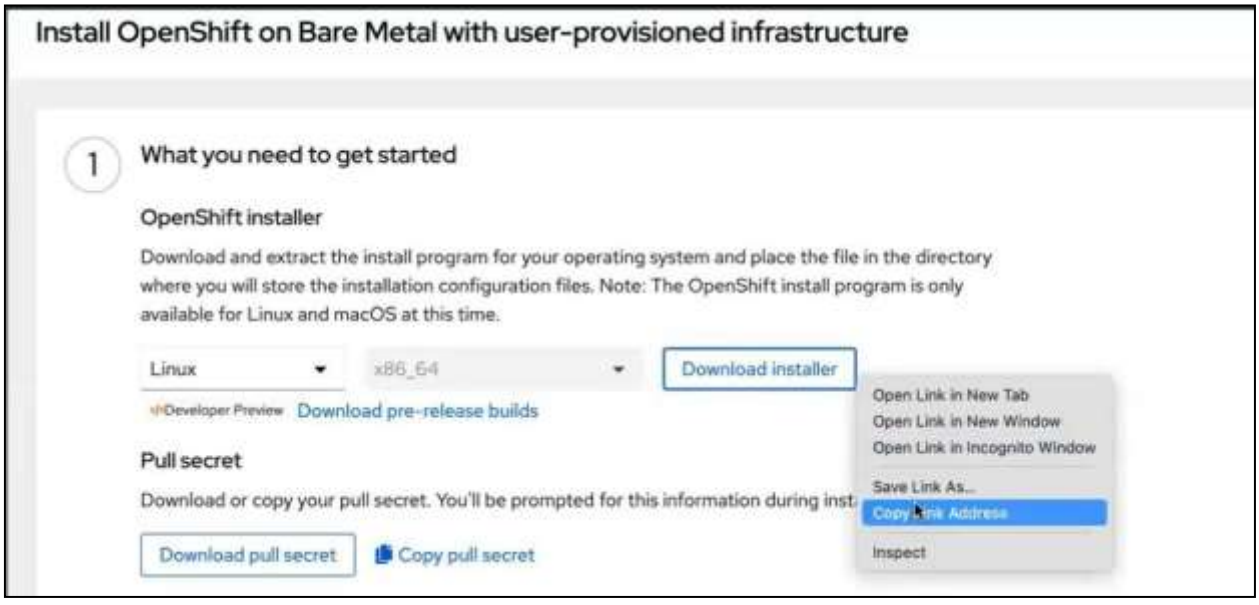
```
[root@nginx ~]# systemctl restart nginx.service
[root@nginx ~]# systemctl status  nginx.service
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-09-28 02:13:40 EDT; 8s ago
  Process: 33132 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
  Process: 33130 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
  Process: 33128 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
 Main PID: 33133 (nginx)
    Tasks: 5 (limit: 100190)
   Memory: 7.5M
   CGroup: /system.slice/nginx.service
           ├─33133 nginx: master process /usr/sbin/nginx
           ├─33134 nginx: worker process
           ├─33135 nginx: worker process
           ├─33136 nginx: worker process
           └─33137 nginx: worker process

Sep 28 02:13:40 nginx.vcet.citiuscloud.com systemd[1]: Starting The nginx HTTP and reverse proxy server...
Sep 28 02:13:40 nginx.vcet.citiuscloud.com nginx[33130]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Sep 28 02:13:40 nginx.vcet.citiuscloud.com nginx[33130]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Sep 28 02:13:40 nginx.vcet.citiuscloud.com systemd[1]: Started The nginx HTTP and reverse proxy server.
[root@nginx ~]# netstat  -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      33133/nginx: master
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      33133/nginx: master
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      930/sshd
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      33133/nginx: master
tcp        0      0 0.0.0.0:22623           0.0.0.0:*               LISTEN      33133/nginx: master
tcp        0      0 0.0.0.0:6443            0.0.0.0:*               LISTEN      33133/nginx: master
tcp6       0      0 :::8080                 :::*                    LISTEN      33133/nginx: master
tcp6       0      0 :::22                   :::*                    LISTEN      930/sshd
udp        0      0 127.0.0.1:323           0.0.0.0:*                           880/chronyd
udp6       0      0 ::1:323                 :::*                                880/chronyd
```

Go to https://cloud.redhat.com/ and login to the console

1. Click on service, drop down menu will be visible
2. Now, click on Infrastructure and click on Cluster
3. All the clusters can be accessed through the dashboard
4. Now, click on create cluster, in the datacenter section
5. Choose any Baremetal and select any of the installation type, here we are using Full control.





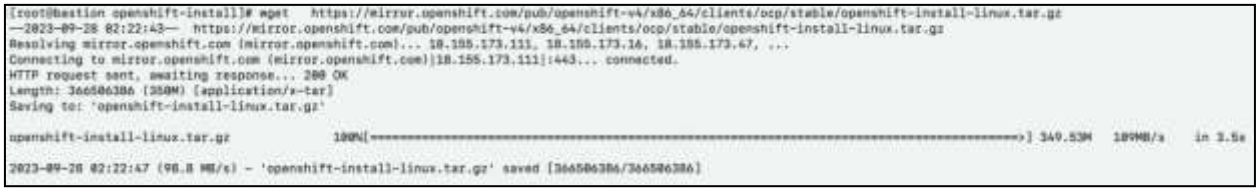
- Go to bastion machine

```
mkdir openshift-install

mkdir openshift-deployment

cd openshift-install

wget <installer link which we copied before>
```

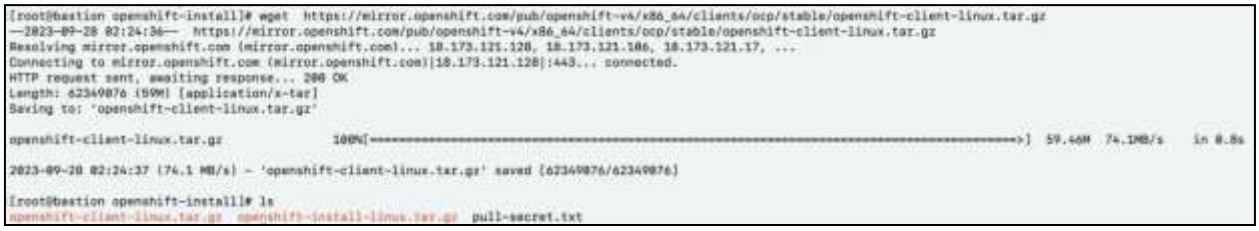


- Go to console again and copy pull secret
- Go to bastion machine

```
vim pull-secret.txt                    #paste the secret copied.
```

- Go to console again and copy command line tools installer address

```
wget <installer link which we copied>
```



- Go to bastion machine

```
ls

tar -xvf openshift-client-linux.tar.gz

tar -xvf openshift-install-linux.tar.gz
```

```
[root@bastion openshift-install]# tar  -xvf  openshift-client-linux.tar.gz
README.md
oc
kubectl
[root@bastion openshift-install]# tar  -xvf  openshift-install-linux.tar.gz
README.md
openshift-install
```

```
mv oc kubectl /usr/local/bin
oc version
```

***search on google >> rhcos iso -> ..miror > select required version of rhcos if needed*

openshift upi installation bare metal: https://docs.openshift.com/container-platform/4.13/installing/installing_bare_metal/installing-bare-metal.html#installation-bare-metal-config-yaml_installing-bare-metal

##MAKE SURE YOU ARE SELECTING CORRECT VERSION

Make changes in the file according to our requirements: {baseDomain, metadata(vcet), keep rest default}

copy and paste pull secrets in

```
>> pullsecret: '{<your-secret-file>}'
```

```
ssh keygen -t rsa
cat /root/.ssh/id-rsa.conf
```

copy the contents and paste in the same file in

```
>> sshKey: '<your-ssh-key>
```

copy sample install-config.yaml and make the file in the openshift-install folder with the same name(install-config.yaml)

```
apiVersion: v1
baseDomain: example.com  #change according to your domain
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: test  #your sub-domain here
networking:
  clusterNetwork:
```

```
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN serviceNetwork:
  - 172.30.0.0/16
  platform:
  none: {} fips: false
  pullSecret: '<your pull secret file contents here>'
  sshKey: ‘<paste public key contents here>’
```

→To get the public key:


```
cp install-config.yaml /root/openshift-deployment
./openshift-install create manifests --dir <openshift-deployment-directory>
```

```
cd /root/openshift-deployment/openshift
ls -ltrh
cd ..
rm -rf openshift/99/_openshift-cluster-api_master-machines-*.yaml
rm -rf openshift/99/_openshift-cluster-api_worker-machineset-*.yaml
vim manifests/cluster-schedular-02-config.yaml          #make changes in the file.
>>masterScheduable: false
```

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: false
  policy:
    name: ""
status: {}
```

3 Iginition Files - {Master, Worker, Bootstrap}

```
cd openshift-install
./openshift-install create iginition-configs --dir /root/openshift-deployment
cd openshift-deployment
ls
```

```
[root@bastion openshift-install]# ./openshift-install create ignition-configs --dir /root/openshift-deployment/
INFO Consuming Common Manifests from target directory
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Openshift Manifests from target directory
INFO Consuming Worker Machines from target directory
INFO Consuming Master Machines from target directory
INFO Ignition-Configs created in: /root/openshift-deployment and /root/openshift-deployment/auth
[root@bastion openshift-install]# cd /root/openshift-deployment/
[root@bastion openshift-deployment]# ls
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

- Bastion Machine:

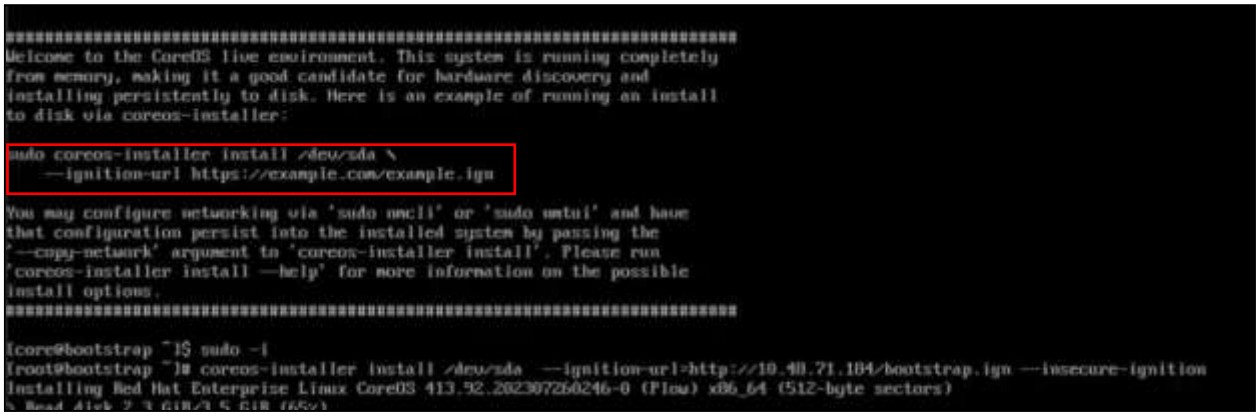
```
subscription-manager register
subscription-manager auto-attach
yum install httpd*
cd openshift-deployment
cp -a *.ign /var/www/html/
cd /var/www/html/
ls
chmod 777 *
ls          #files displayed in green colour
```

```
[root@bastion openshift-deployment]# cp -a  *.ign  /var/www/html/
[root@bastion openshift-deployment]# cd /var/www/html/
[root@bastion html]# ls
bootstrap.ign  master.ign  worker.ign
[root@bastion html]# chmod 777 *
[root@bastion html]# ls
bootstrap.ign  master.ign  worker.ign
```

```
setenforce 0
systemctl stop firewalld.service
systemctl disable firewalld.service
systemctl restart httpd.service
systemctl status httpd.service
```

- Go to Prism central and power on the bootstrap machine and launch console.

```
sudo -i
coreos-installer install /dev/sda --ignition-url=http://<bastion-ip>/bootstrap.ign --insecure-ignition
##The above command is already generated when we launch the console just modify it by adding bastion machine IP.
```



*****repeat same process for all the master node and worker node*****

FOR MASTER:

```
coreos-installer install /dev/sda --ignition-url=http://<bastion-ip>/master.ign --insecure-ignition
```

SHUTDOWN MACHINES AFTER INSTALLATION USING >> `shutdown -P now`

##after shutdown unmount the disk (CD-ROM)

Power on bootstrap machine

- Go to bastion machine

try to ping bootstrap from bastion machine, if pinged do the further steps

ssh core@bootstrap
journalctl -b -f -u release-image.service -u bootkube.service ***{command in the output after ssh is done}***

after api is up power on all the master machines

→power on worker machine

sudo -i
FOR WORKER:
coreos-installer install /dev/sda --ignition-url=http://<bastion-ip>/worker.ign --insecure-ignition

SHUTDOWN MACHINES AFTER INSTALLATION USING >> shutdown -P now

##after shutdown unmount the disk (CD-ROM)

Go to redhat portal: https://docs.openshift.com/container-platform/4.13/installing/installing_bare_metal/installing-bare-metal.html#installation-bare-metal-config-yaml_installing-bare-metal

ctrl-f and search → wait-for.. {command}

- Go to bastion machine

```
cd openshift-install

./openshift-install --dir <openshift-deployment-directory>  wait-for bootstrap-complete --log-level=info
```

Add another tab for bastion machine

```
ssh core@master1

oc get node

sudo find / -name kubeconfig

export KUBECONFIG=/etc/kubernetes/kubeconfig

oc get node

exit
```

```
[root@bastion ~]# export  KUBECONFIG=/root/openshift-deployment/auth/kubeconfig
[root@bastion ~]# oc get nodes
NAME                           STATUS   ROLES                  AGE   VERSION
master1.vcet.citiuscloud.com   Ready    control-plane,master   16m   v1.26.7+c7ee51f
master2.vcet.citiuscloud.com   Ready    control-plane,master   15m   v1.26.7+c7ee51f
master3.vcet.citiuscloud.com   Ready    control-plane,master   15m   v1.26.7+c7ee51f
```

- Power on both the worker node
- Go to bastion machine

```
export KUBECONFIG=/root/openshift-deployment/auth/kubeconfig

oc get node

oc get co

oc get csr | grep -i pending

oc get csr -o name | xargs oc adm certificate approve          ##approve the pending request
```

oc get nodes

```
Every 2.0s: oc get nodes

NAME                              STATUS   ROLES                  AGE    VERSION
master1.vcet.citiuscloud.com      Ready    control-plane,master   21m    v1.26.7+c7ee51f
master2.vcet.citiuscloud.com      Ready    control-plane,master   21m    v1.26.7+c7ee51f
master3.vcet.citiuscloud.com      Ready    control-plane,master   21m    v1.26.7+c7ee51f
worker1.vcet.citiuscloud.com      Ready    worker                 116s   v1.26.7+c7ee51f
worker2.vcet.citiuscloud.com      Ready    worker                 110s   v1.26.7+c7ee51f
```

##kill the bootstrap cli process going on

- Go to Bastion machine

oc get node	*#3 master 2 worker in ready condition*
watch oc get co	*#check if all are true in available column and wait till all are true*

```
[root@bastion ~]# oc get co
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
authentication                             4.13.13   True        False         False      52s
baremetal                                  4.13.13   True        False         False      27m
cloud-controller-manager                   4.13.13   True        False         False      30m
cloud-credential                           4.13.13   True        False         False      32m
cluster-autoscaler                         4.13.13   True        False         False      27m
config-operator                            4.13.13   True        False         False      28m
console                                    4.13.13   True        False         False      6m14s
control-plane-machine-set                  4.13.13   True        False         False      28m
csi-snapshot-controller                    4.13.13   True        False         False      28m
dns                                        4.13.13   True        False         False      28m
etcd                                       4.13.13   True        False         False      26m
image-registry                             4.13.13   True        False         False      18m
ingress                                    4.13.13   True        False         False      9m5s
insights                                   4.13.13   True        False         False      21m
kube-apiserver                             4.13.13   True        False         False      24m
kube-controller-manager                    4.13.13   True        False         False      24m
kube-scheduler                             4.13.13   True        False         False      24m
kube-storage-version-migrator              4.13.13   True        False         False      28m
machine-api                                4.13.13   True        False         False      27m
machine-approver                           4.13.13   True        False         False      28m
machine-config                             4.13.13   True        False         False      27m
marketplace                                4.13.13   True        False         False      27m
monitoring                                 4.13.13   True        False         False      8m15s
network                                    4.13.13   True        False         False      28m
node-tuning                                4.13.13   True        False         False      27m
openshift-apiserver                        4.13.13   True        False         False      22m
openshift-controller-manager               4.13.13   True        False         False      22m
openshift-samples                          4.13.13   True        False         False      21m
operator-lifecycle-manager                 4.13.13   True        False         False      28m
operator-lifecycle-manager-catalog         4.13.13   True        False         False      28m
operator-lifecycle-manager-packageserver   4.13.13   True        False         False      22m
service-ca                                 4.13.13   True        False         False      28m
storage                                    4.13.13   True        False         False      28m
```

- Add another tab for bastion machine

```
Ssh root@<bastion-ip>

ssh core@<master-node>

sudo crictl pods

exit
```

- Bastion machine:

```
oc get pods -n kube-system

oc get pods -A | grep api                    **openshift has their own dedicated namespace for the nodes same like how kubectl has kube-system,etc.**

oc get routes -A
```

- Copy the console host/port link {console-openshift-console.apps.vcet.citiuscloud.com}
- Do the entry in the local machine to get dashboard access from the local machine

```
sudo vi /etc/hosts

<nginx-ip> console-openshift-console.apps.vcet.citiuscloud.com
```

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
10.48.71.88  pc.citiuscloud.com
10.48.71.191 console-openshift-console.apps.vcet.citiuscloud.com
10.48.71.191 oauth-openshift.apps.vcet.citiuscloud.com
10.48.71.191 k10-route-kasten-io.apps.vcet.citiuscloud.com
```

➔Go to chrome and search >> console-openshift-console.apps.vcet.citiuscloud.com

#TO GET THE PASSWORD

➔*Go to bastion machine*

```
cd openshift-install

./openshift-install --dir <openshift-deployment-directory>  wait-for install-complete
```

```
[root@bastion openshift-install]# ./openshift-install --dir /root/openshift-deployment/ wait-for install-complete
INFO Waiting up to 40m0s (until 4:43AM) for the cluster at https://api.vcet.citiuscloud.com:6443 to initialize...
INFO Checking to see if there is a route at openshift-console/console...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/root/openshift-deployment/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.vcet.citiuscloud.com
INFO Login to the console with user: "kubeadmin", and password: "hJyTJ-nFiLo-FddvB-ZRtAt"
INFO Time elapsed: 0s
```

OR

cat /root/openshift-deployment/auth/kubeadmin-password

```
[root@bastion openshift-install]# cat /root/openshift-deployment/auth/kubeadmin-password
hJyTJ-nFiLo-FddvB-ZRtAt[root@bastion openshift-install]#
```