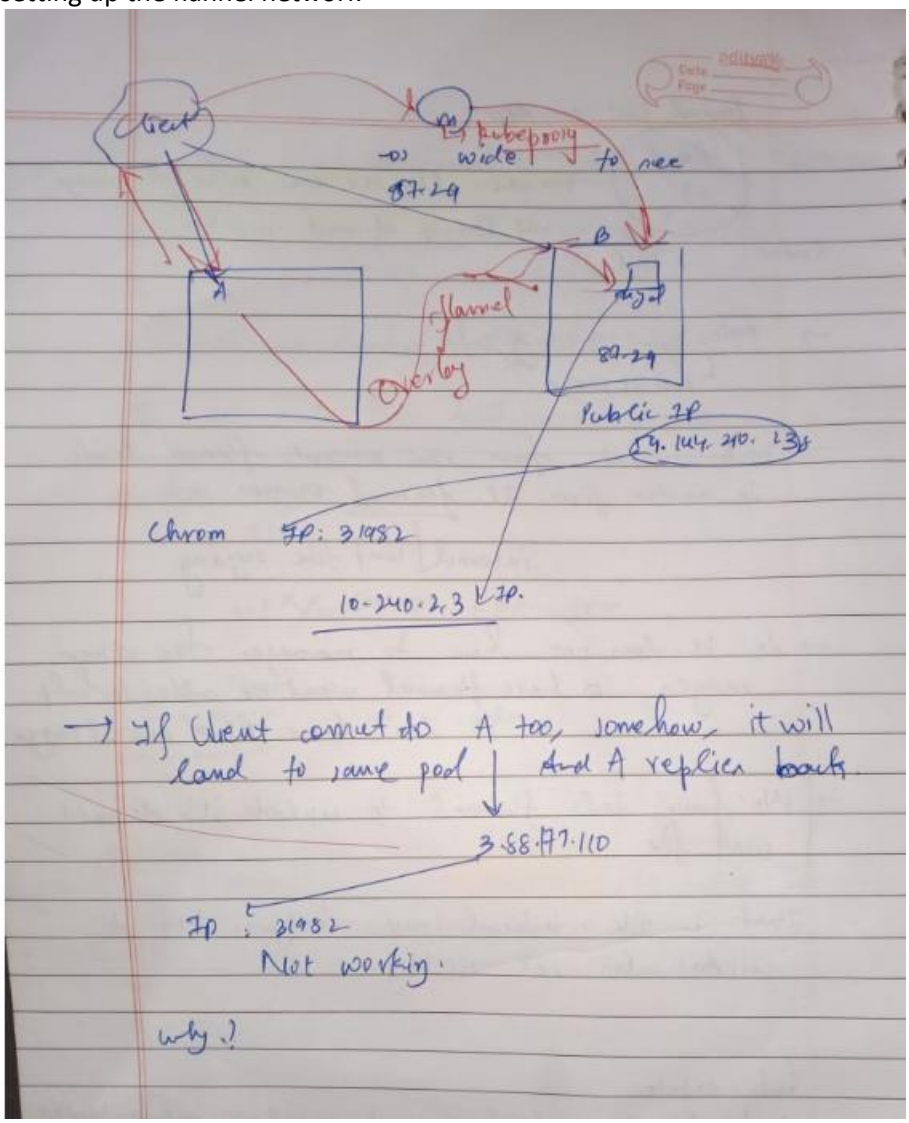
setting up the flannel network`



Scenario

- ➔ now, there is one master(M) and two Worker Node(W1 and W2)
- ⇨ we have to launch one web server in W2 and its very obvious that the server can be accessed with W2 ip.
- ⇨ Challenge: I want to access the server setup in W2 with W1 Ip and Master IP.
- ⇨ why the above requirement exists? because at times we may have several say 100 of worker nodes running, so providing the particular IP of distinct services is not good. we have to provide just one IP to the client.

Intuition:

- ⇨ in cluster, WN's works as team, so on behalf of client, the one IP provided to the client of this cluster will come inside the cluster and will retrieve the info from the respective server. This is possible due to the program called kubeproxy.
- ⇨ Kubeproxy behind the scenes uses flannel (uses backend tunnelling of VXLAN to setup the overlay).

AIM - The only thing we have to do – is to change the conf file of flannel and give the ip range of the pods that we have provided to kubeadm init while setting up the multi node cluster.

edit the conf file of flannel with:

kubectl edit configmap kube-flannel-cfg -n kube-system

and changing the conf of any service requires a restart for the changes to take place.w

setting up the flannel network`

current status….

```
[root@ip-172-31-52-186 ~]# kubectl get nodes
NAME                          STATUS   ROLES                  AGE   VERSION
ip-172-31-49-3.ec2.internal   Ready    <none>                 13d   v1.20.2
ip-172-31-52-186.ec2.internal Ready    control-plane,master   14d   v1.20.2
ip-172-31-93-26.ec2.internal  Ready    <none>                 45h   v1.20.2
[root@ip-172-31-52-186 ~]#
```

a. deploy one server in one WN..

⇨ creating a namespace

⇨ launch a deployment of pod

Commands:

⇨ kubectl create namespace tech - create a namespace called “tech”

⇨ kubectl get namespace - check if created or not.

⇨ kubectl create deployment --image=vimal13/apache-webserver-php -n tech - launch a deployment

⇨ kubectl get pods -n tech - the pod myd is deployed.

⇨ ifconfig -a - show only real network card.

⇨ kubectl get pod -o wide -n tech - determine in which WN , above pod is launched.

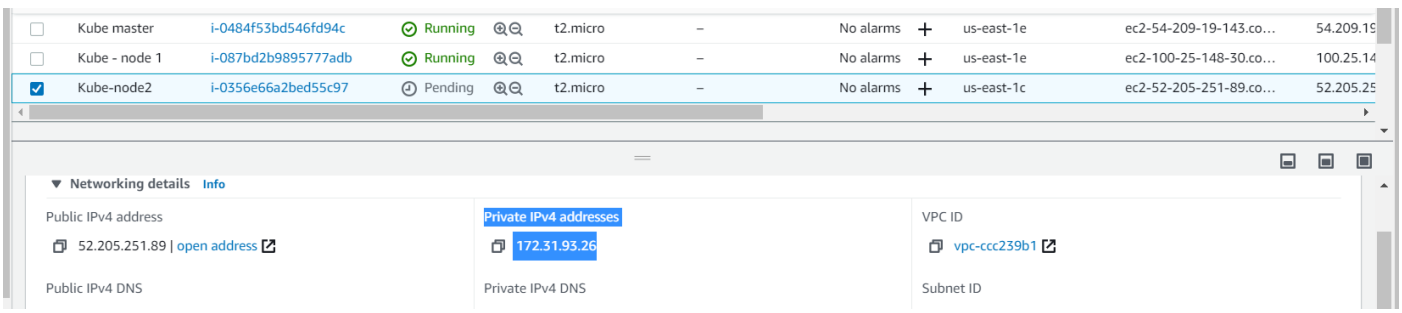
```
[root@ip-172-31-52-186 ~]# kubectl create namespace tech
namespace/tech created
[root@ip-172-31-52-186 ~]# kubectl get namespace
NAME              STATUS   AGE
default           Active   14d
kube-node-lease   Active   14d
kube-public       Active   14d
kube-system       Active   14d
tech              Active   10s
```

```
[root@ip-172-31-52-186 ~]# kubectl create deployment myd --image=vimal13/apache-
webserver-php -n tech
deployment.apps/myd created
[root@ip-172-31-52-186 ~]# kubectl get pods -n tech
```

```
[root@ip-172-31-52-186 ~]# kubectl get pods -n tech
NAME                   READY   STATUS    RESTARTS   AGE
myd-5f55596db4-mh2mc   1/1     Running   0          14s
[root@ip-172-31-52-186 ~]#
```

```
[root@ip-172-31-52-186 ~]# kubectl  get pod -o wide -n tech
NAME                   READY   STATUS    RESTARTS   AGE     IP           NODE
                                NOMINATED NODE   READINESS GATES
myd-5f55596db4-mh2mc   1/1     Running   0          3m31s   10.240.2.3   ip-172-
31-93-26.ec2.internal   <none>           <none>
[root@ip-172-31-52-186 ~]#
```

launched in kube node2



GO TO KUBE NODE 2 and verify:

setting up the flannel network`

- ⇨ docker ps

```
root@ip-172-31-93-26:~
[ec2-user@ip-172-31-93-26 ~]$ sudo su -
Last login: Mon Feb 15 14:34:47 UTC 2021 on pts/0
[root@ip-172-31-93-26 ~]# docker ps
CONTAINER ID        IMAGE                           COMMAND                  CREA
TED             STATUS              PORTS               NAMES
e75b017fb52b        vimal13/apache-webserver-php    "/usr/sbin/httpd -DF…"   12 m
inutes ago      Up 12 minutes                           k8s_apache-webserver-php
_myd-5f55596db4-mh2mc_tech_fc880348-9b85-467c-b27a-82c68b9496a8_0
ab017b3fd23b        k8s.gcr.io/pause:3.2            "/pause"                 12 m
inutes ago      Up 12 minutes                           k8s_POD_myd-5f55596db4-m
h2mc_tech_fc880348-9b85-467c-b27a-82c68b9496a8_0
546ec6178843        f03a23d55e57                    "/opt/bin/flanneld -…"   28 m
```

- ⇨ by def, the containers in docker are in a private isolated LAN, attached to virtual network card in docker "cni0"

```
[root@ip-172-31-93-26 ~]#  brctl show
bridge name     bridge id               STP enabled     interfaces
cni0            8000.4aad38fe3ef0       no              vethcf9098c9
docker0         8000.0242b7335758       no
[root@ip-172-31-93-26 ~]#
```

- ⇨ individual network card is given to each container launched by docker engine , form container perspective, it seems a real N/W card but from the base OS perspective, that is a virtual N/W card and starts from "veth…"

ifconfig -a

```
vethcf9098c9: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 8951
        inet6 fe80::3446:cdff:fe05:749e  prefixlen 64  scopeid 0x20<link>
        ether 36:46:cd:05:74:9e  txqueuelen 0  (Ethernet)
        RX packets 1  bytes 42 (42.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 668 (668.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- ⇨ exposing this container…..

come to master…

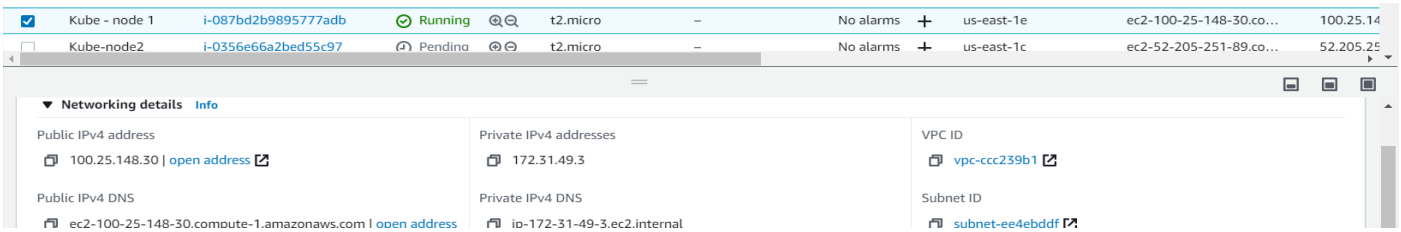
```
root@ip-172-31-52-186:~
[root@ip-172-31-52-186 ~]# kubectl expose deploy myd --port=80 --type=NodePort -
n tech
service/myd exposed
[root@ip-172-31-52-186 ~]# kubectl get svc
NAME         TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        14d
myd          NodePort    10.110.49.182   <none>        80:31022/TCP   13d
[root@ip-172-31-52-186 ~]#
```

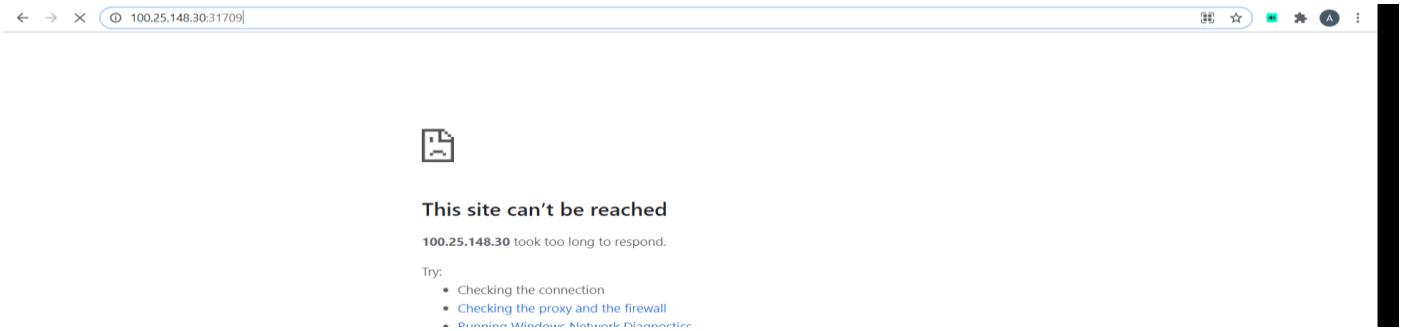
open: http://ip.of.WN.where.pod.is.launched:31022

Since the pod was launched in kube node2 and IP : 52.205.251.89 => http://52.205.251.89:31022

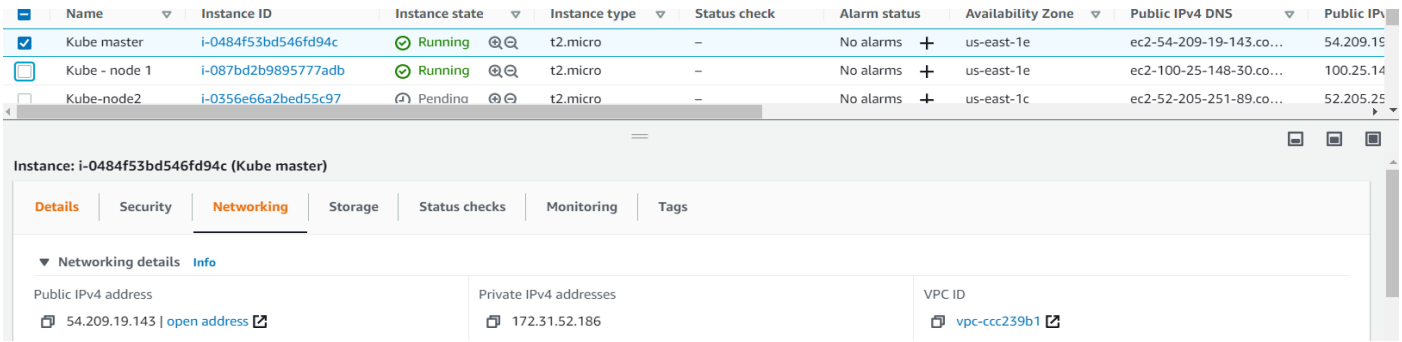
setting up the flannel network`

➔ now, trying with WN node 1 IP, it fails…

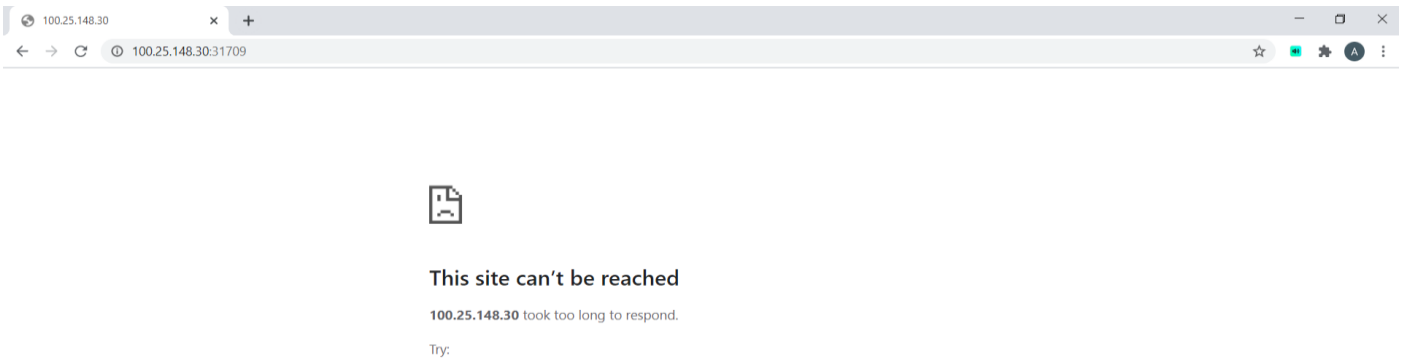




➔ opening with master IP, it will fail…



54.209.19.143



setting up the flannel network`

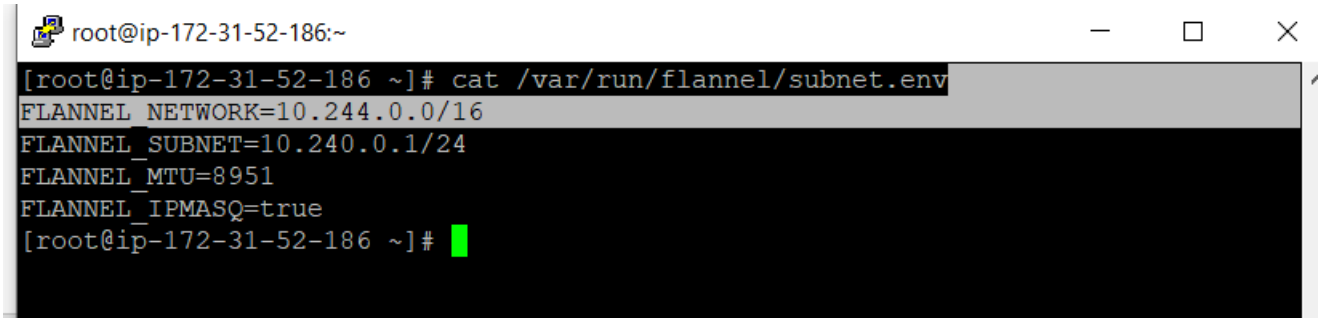
Solution: have to edit the conf file of flannel and provide the IP range which is provide to kubeadm init while setting up the multi node cluster.

GO to master:

cat /var/run/flannel/subnet.env

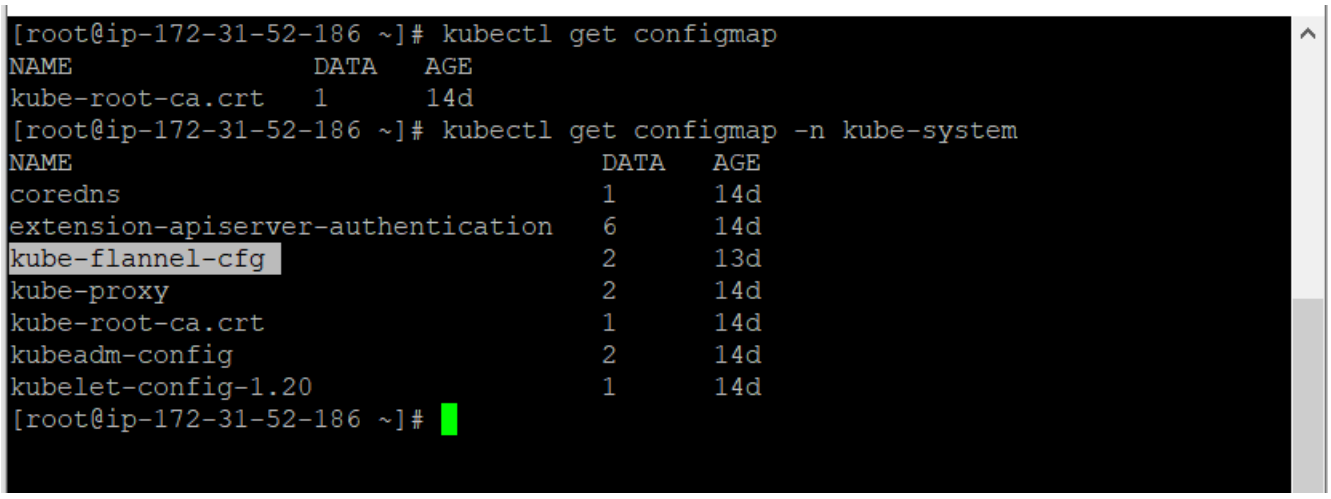
FLANNEL_NETWORK: the IP we have provided

FLANNEL_SUBNET: the IP range provided by the developer.



have to change the subnet to our provided network range ie. 10.240.0.0/16

- ⇨ kubectl get configmap - searching fir conf file
- ⇨ kubeclt get configmap -n kube-system - conf file in kube-system namespace , yes kube-flannel-cfg
- ⇨ kubectl edit configmap kube-flannel-cfg -n kube-system – edit the conf file of flannel
- ⇨ kubectl get pod -l app=flannel -n kube-system – check the status of flannel
- ⇨ kubectl delete pod -l app=flannel -n kube-system - deleting to restart the pod



setting up the flannel network`
also, the coredns is also not working now…

```
[root@ip-172-31-52-186 ~]# kubectl get pod -n kube-system
NAME                                                READY   STATUS    RESTAR
TS   AGE
coredns-74ff55c5b-xpncd                             0/1     Running   3
     14d
coredns-74ff55c5b-z45dq                             0/1     Running   3
     14d
etcd-ip-172-31-52-186.ec2.internal                  1/1     Running   3
     14d
kube-apiserver-ip-172-31-52-186.ec2.internal        1/1     Running   3
     14d
kube-controller-manager-ip-172-31-52-186.ec2.internal   1/1     Running   3
     14d
kube-flannel-ds-7qthx                               1/1     Running   3
     13d
kube-flannel-ds-gjtvv                               1/1     Running   2
     46h
kube-flannel-ds-qd4tz                               1/1     Running   3
     13d
kube-proxy-9gjm7                                    1/1     Running   3
     13d
kube-proxy-9ngv2                                    1/1     Running   2
     46h
kube-proxy-gbwqc                                    1/1     Running   3
     14d
kube-scheduler-ip-172-31-52-186.ec2.internal        1/1     Running   3
     14d
[root@ip-172-31-52-186 ~]# kubectl get pod -l app=flannel -n kube-system
```

```
[root@ip-172-31-52-186 ~]# kubectl edit configmap kube-flannel-cfg -n kube-syste
m
configmap/kube-flannel-cfg edited
```

```
    }
  net-conf.json: |
    {
      "Network": "10.244.0.0/16",
      "Backend": {
        "Type": "vxlan"
      }
    }
kind: ConfigMap
metadata:
```

change to 10.240.0.0/16

```
    }
  net-conf.json: |
    {
      "Network": "10.240.0.0/16",
      "Backend": {
        "Type": "vxlan"
      }
    }
```

edited….

```
[root@ip-172-31-52-186 ~]# kubectl get pod -l app=flannel -n kube-system
NAME                    READY   STATUS    RESTARTS   AGE
kube-flannel-ds-7qthx   1/1     Running   3          13d
kube-flannel-ds-gjtvv   1/1     Running   2          46h
kube-flannel-ds-qd4tz   1/1     Running   3          13d
[root@ip-172-31-52-186 ~]#
```

setting up the flannel network`
need to restart the service for the changes to take place. so lets delete the flannel pod, deployment will launch it again with update settings..

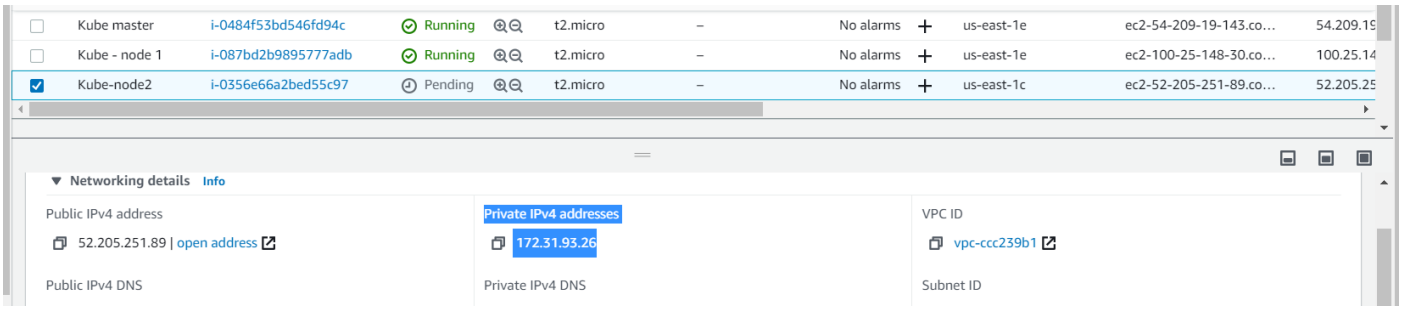
```
[root@ip-172-31-52-186 ~]# cat /var/run/flannel/subnet.env
FLANNEL_NETWORK=10.240.0.0/16
FLANNEL_SUBNET=10.240.0.1/24
FLANNEL_MTU=8951
FLANNEL_IPMASQ=true
[root@ip-172-31-52-186 ~]#
```

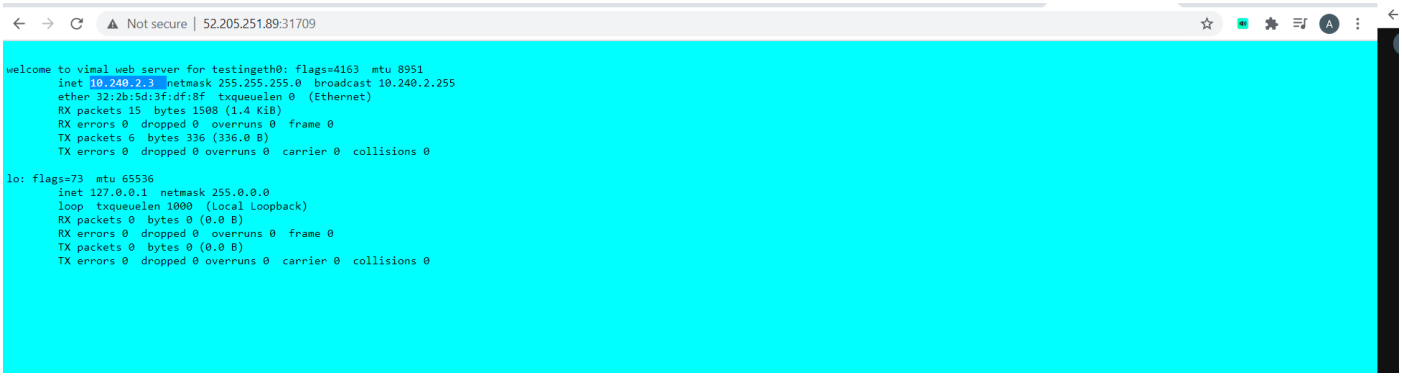
done.

⇨ Also, core dns starts too….

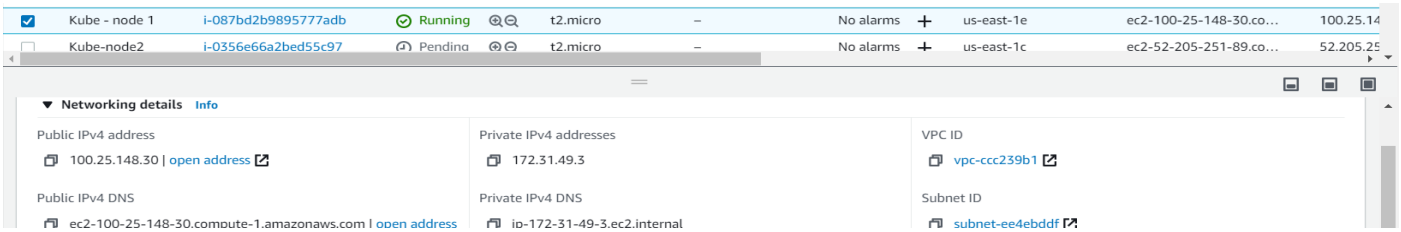
```
[root@ip-172-31-52-186 ~]# kubectl get pod -n kube-system
NAME                                                READY   STATUS    RESTARTS   AGE
coredns-74ff55c5b-xpncd                             1/1     Running   3          14d
coredns-74ff55c5b-z45dq                             1/1     Running   3          14d
etcd-ip-172-31-52-186.ec2.internal                  1/1     Running   3          14d
kube-apiserver-ip-172-31-52-186.ec2.internal        1/1     Running   3          14d
kube-controller-manager-ip-172-31-52-186.ec2.internal   1/1     Running   3          14d
kube-flannel-ds-92vpl                               1/1     Running   0          3m17s
kube-flannel-ds-gc2jl                               1/1     Running   0          3m15s
```

➔ now the server hosted in WN2 can be opened by IP of WN1 too.
⇨ from WN2



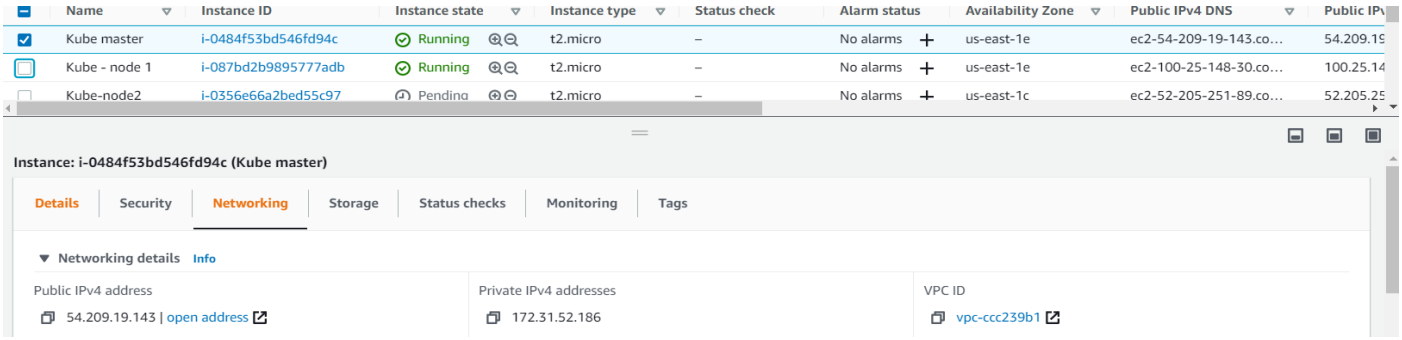


⇨ from WN1



setting up the flannel network`

⇨ from master IP.



so, now we can provide one single IP to the client….

setting up the flannel network`
ADDITIONAL INFO ON FLANNEL

- ⇨ Who manages flannel pod?

Ans. daemon set

kubectl describe pod -l app=flannel -n kube-system

```
[root@ip-172-31-52-186 ~]# kubectl describe pod -l app=flannel -n kube-system
```

```
Annotations:          <none>
Status:               Running
IP:                   172.31.93.26
IPs:
  IP:           172.31.93.26
Controlled By:  DaemonSet/kube-flannel-ds
Init Containers:
  install-cni:
    Container ID:  docker://e80311feaa4918d409ff6d06371e0023501c8f321e23dd193d6f61f0adfa431f
    Image:         quay.io/coreos/flannel:v0.13.1-rc1
```

- ⇨ how many replicas are configured for flannel ?

Ans. 3

kubectl get ds -n kube-system

```
NAME              DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
kube-flannel-ds   3         3         3       3            3           <none>                   13d
kube-proxy        3         3         3       3            3           kubernetes.io/os=linux   14d
[root@ip-172-31-52-186 ~]#
```