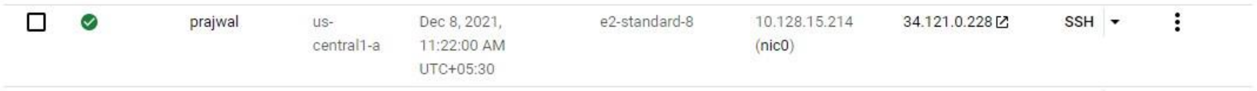
Test report - Deployment of Arktos Cluster with Mizar CNI on GCE

This document captures the steps to deploy an Arktos cluster lab with mizar cni. The machine in this lab used are GCE e2-standard-8 (8 vCPUs, 32 GB memory) and the storage size is 128GB), Ubuntu 18.04 LTS.

Date-24.09.2021

Create an instance on GCE

Created instance on GCE



☐	✓	prajwal	us-central1-a	Dec 8, 2021, 11:22:00 AM UTC+05:30	e2-standard-8	10.128.15.214 (nic0)	34.121.0.228	SSH

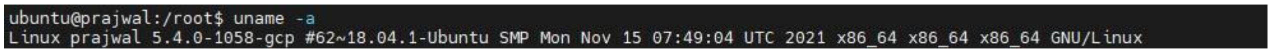
SSH instance with credentials.

Step-1: Update kernel (If required)

To check kernel, run following command

```
uname -a
```

output:

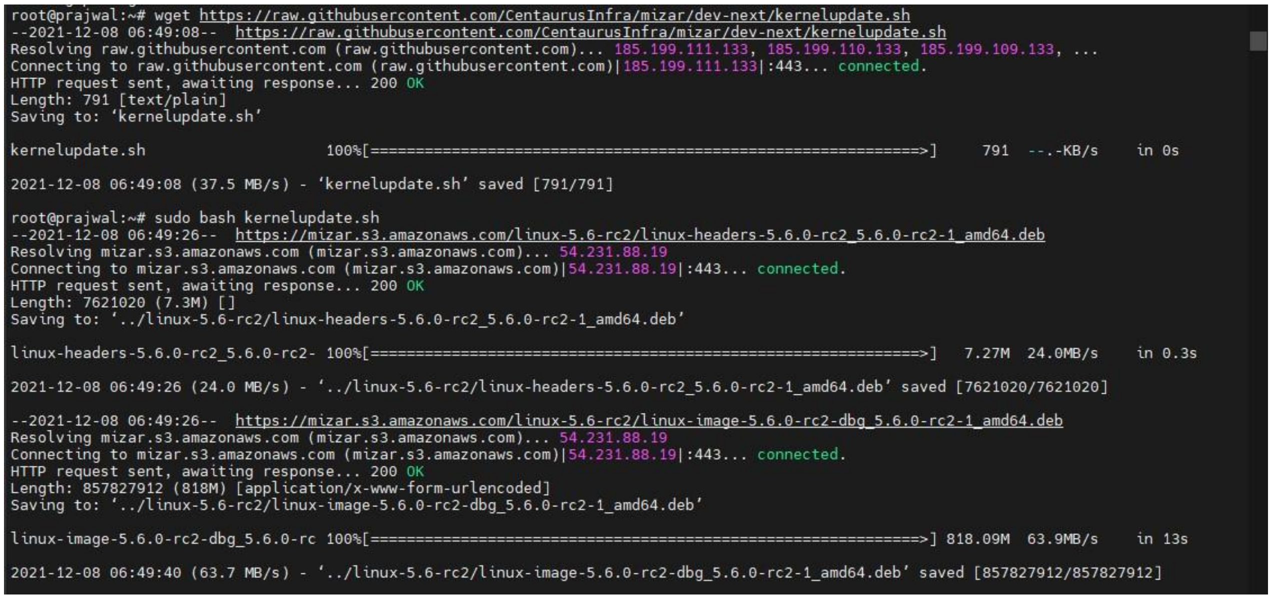


```
ubuntu@prajwal:/root$ uname -a
Linux prajwal 5.4.0-1058-gcp #62~18.04.1-Ubuntu SMP Mon Nov 15 07:49:04 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

Here kernel version is 5.4.0-1051-gcp which is less than the required kernel version, so to update the kernel version to 5.6.0-rc2, we used the following steps :

```
wget https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
sudo bash kernelupdate.sh
```

output:



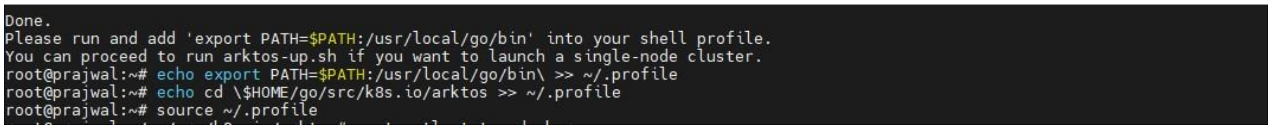
```
root@prajwal:~# wget https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
--2021-12-08 06:49:08--  https://raw.githubusercontent.com/CentaurusInfra/mizar/dev-next/kernelupdate.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.111.133, 185.199.110.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 791 [text/plain]
Saving to: 'kernelupdate.sh'

kernelupdate.sh                100%[=============================================================>]     791  --.-KB/s    in 0s

2021-12-08 06:49:08 (37.5 MB/s) - 'kernelupdate.sh' saved [791/791]

root@prajwal:~# sudo bash kernelupdate.sh
--2021-12-08 06:49:26--  https://mizar.s3.amazonaws.com/linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb
Resolving mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)... 54.231.88.19
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|54.231.88.19|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7621020 (7.3M) []
Saving to: '../linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb'

linux-headers-5.6.0-rc2_5.6.0-rc2- 100%[=============================================================>]   7.27M  24.0MB/s    in 0.3s

2021-12-08 06:49:26 (24.0 MB/s) - '../linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb' saved [7621020/7621020]

--2021-12-08 06:49:26--  https://mizar.s3.amazonaws.com/linux-5.6-rc2/linux-image-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64.deb
Resolving mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)... 54.231.88.19
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|54.231.88.19|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 857827912 (818M) [application/x-www-form-urlencoded]
Saving to: '../linux-5.6-rc2/linux-image-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64.deb'

linux-image-5.6.0-rc2-dbg_5.6.0-rc 100%[=============================================================>] 818.09M  63.9MB/s    in 13s

2021-12-08 06:49:40 (63.7 MB/s) - '../linux-5.6-rc2/linux-image-5.6.0-rc2-dbg_5.6.0-rc2-1_amd64.deb' saved [857827912/857827912]
```

Step-2: Install dependencies

Run the following steps to install dependencies required for arktos deployment:

```
git clone https://github.com/Click2Cloud-Centaurus/arktos.git ~/go/src/k8s.io/arktos
-b default-cni-mizar  sudo bash
$HOME/go/src/k8s.io/arktos/hack/setup-dev-node.sh  echo
export PATH=$PATH:/usr/local/go/bin\ >> ~/.profile  echo cd
\$HOME/go/src/k8s.io/arktos >> ~/.profile  source ~/.profile
```

output:



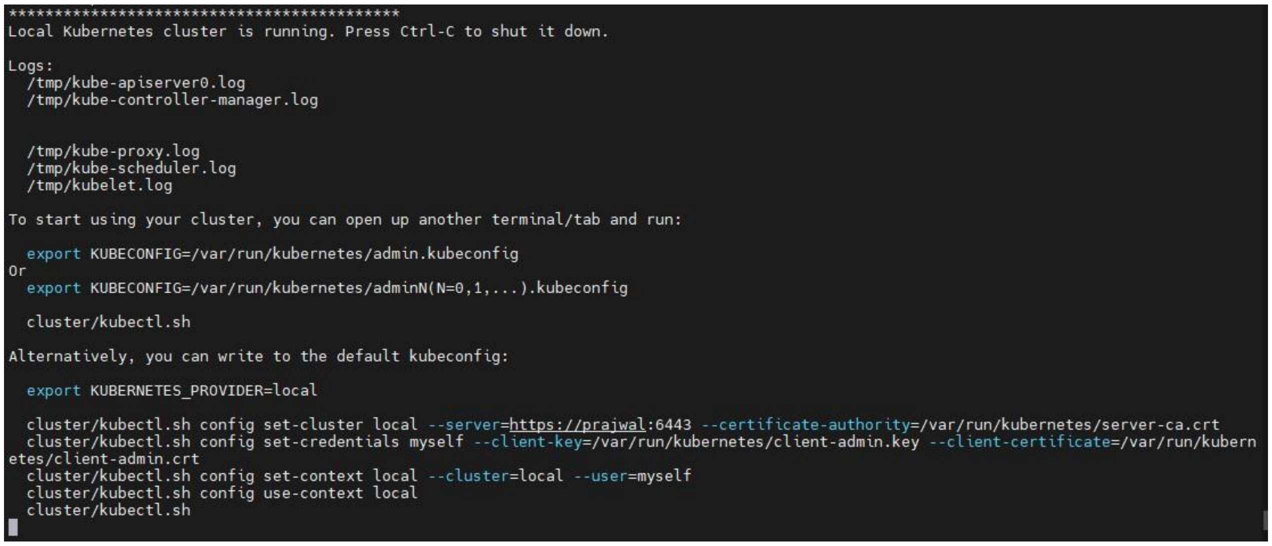
```
Done.
Please run and add 'export PATH=$PATH:/usr/local/go/bin' into your shell profile.
You can proceed to run arktos-up.sh if you want to launch a single-node cluster.
root@prajwal:~# echo export PATH=$PATH:/usr/local/go/bin\ >> ~/.profile
root@prajwal:~# echo cd \$HOME/go/src/k8s.io/arktos >> ~/.profile
root@prajwal:~# source ~/.profile
```

Step-3: Start Arktos cluster

Login to instance and run following steps to deploy arktos cluster with Mizar as CNI:

```
CNIPLUGIN=mizar ./hack/arktos-up.sh
```

Finally we got following output, which indicates that arktos cluster created successfully with Mizar as CNI output

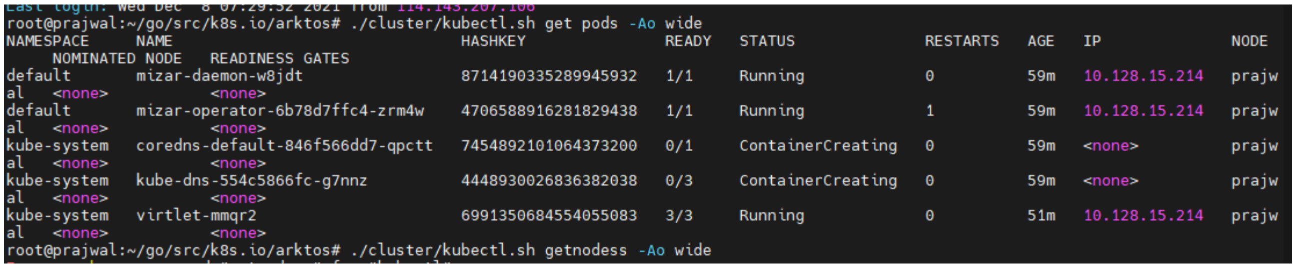


```
*******************************************
Local Kubernetes cluster is running. Press Ctrl-C to shut it down.

Logs:
  /tmp/kube-apiserver0.log
  /tmp/kube-controller-manager.log


  /tmp/kube-proxy.log
  /tmp/kube-scheduler.log
  /tmp/kubelet.log

To start using your cluster, you can open up another terminal/tab and run:

  export KUBECONFIG=/var/run/kubernetes/admin.kubeconfig
Or
  export KUBECONFIG=/var/run/kubernetes/adminN(N=0,1,...).kubeconfig

  cluster/kubectl.sh

Alternatively, you can write to the default kubeconfig:

  export KUBERNETES_PROVIDER=local

  cluster/kubectl.sh config set-cluster local --server=https://prajwal:6443 --certificate-authority=/var/run/kubernetes/server-ca.crt
  cluster/kubectl.sh config set-credentials myself --client-key=/var/run/kubernetes/client-admin.key --client-certificate=/var/run/kubern
etes/client-admin.crt
  cluster/kubectl.sh config set-context local --cluster=local --user=myself
  cluster/kubectl.sh config use-context local
  cluster/kubectl.sh
```

Leave this terminal here as it is (do not close the terminal) and open new terminal of same instance

1) Check pod status

./cluster/kubectl.sh get pods -Ao wide



```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get pods -Ao wide
NAMESPACE     NAME                             HASHKEY               READY   STATUS              RESTARTS   AGE   IP              NODE
     NOMINATED NODE   READINESS GATES
default       mizar-daemon-w8jdt               8714190335289945932   1/1     Running             0          59m   10.128.15.214   prajw
al    <none>           <none>
default       mizar-operator-6b78d7ffc4-zrm4w  4706588916281829438   1/1     Running             1          59m   10.128.15.214   prajw
al    <none>           <none>
kube-system   coredns-default-846f566dd7-qpctt 7454892101064373200   0/1     ContainerCreating   0          59m   <none>          prajw
al    <none>           <none>
kube-system   kube-dns-554c5866fc-g7nnz        4448930026836382038   0/3     ContainerCreating   0          59m   <none>          prajw
al    <none>           <none>
kube-system   virtlet-mmqr2                    6991350684554055083   3/3     Running             0          51m   10.128.15.214   prajw
al    <none>           <none>
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh getnodess -Ao wide
```

Pods kube-dns and coredns are in container creating state for long time

After re-running the script, we have below outputs

`CNIPLUGIN=mizar ./hack/arktos-up.sh -O`

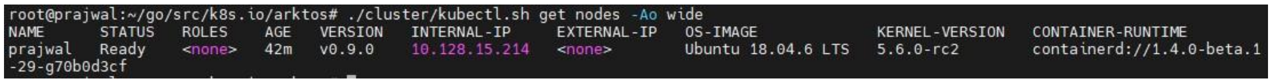
Step-4 Check Cluster health

Open new terminal for same instance and run following commands:

1) Check node status

```
./cluster/kubectl.sh get nodes -Ao wide
```

Output

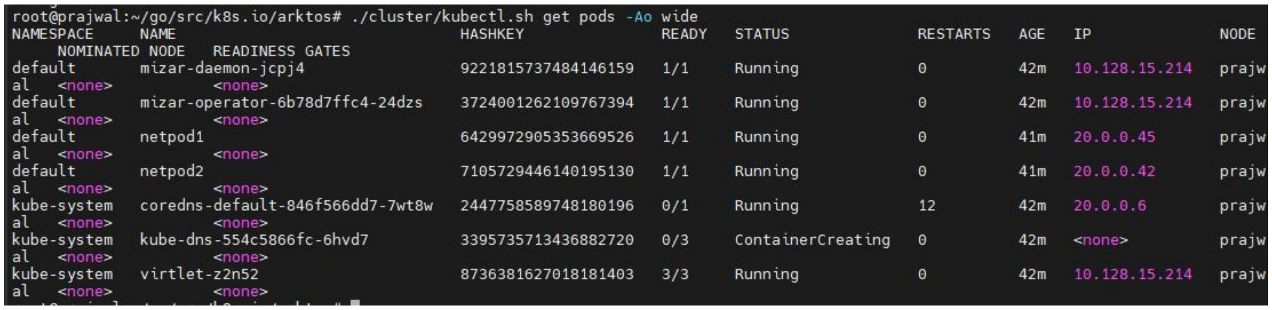


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get nodes -Ao wide
NAME      STATUS   ROLES    AGE   VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
prajwal   Ready    <none>   42m   v0.9.0    10.128.15.214   <none>        Ubuntu 18.04.6 LTS   5.6.0-rc2        containerd://1.4.0-beta.1
-29-g70b0d3cf
```

2) Check pods status

```
./cluster/kubectl.sh get pods -Ao wide
```

Output

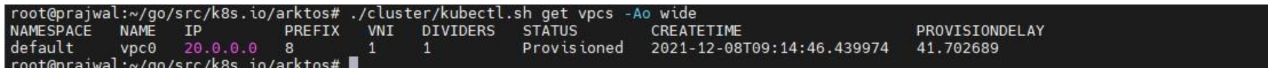


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get pods -Ao wide
NAMESPACE     NAME                                  HASHKEY               READY   STATUS              RESTARTS   AGE   IP              NODE
      NOMINATED NODE   READINESS GATES
default       mizar-daemon-jcpj4                    9221815737484146159   1/1     Running             0          42m   10.128.15.214   prajw
al    <none>           <none>
default       mizar-operator-6b78d7ffc4-24dzs       3724001262109767394   1/1     Running             0          42m   10.128.15.214   prajw
al    <none>           <none>
default       netpod1                               6429972905353669526   1/1     Running             0          41m   20.0.0.45       prajw
al    <none>           <none>
default       netpod2                               7105729446140195130   1/1     Running             0          41m   20.0.0.42       prajw
al    <none>           <none>
kube-system   coredns-default-846f566dd7-7wt8w      2447758589748180196   0/1     Running             12         42m   20.0.0.6        prajw
al    <none>           <none>
kube-system   kube-dns-554c5866fc-6hvd7             3395735713436882720   0/3     ContainerCreating   0          42m   <none>          prajw
al    <none>           <none>
kube-system   virtlet-z2n52                         8736381627018181403   3/3     Running             0          42m   10.128.15.214   prajw
al    <none>           <none>
```

3) Check vpc status

```
./cluster/kubectl.sh get vpc -Ao wide
```

Output

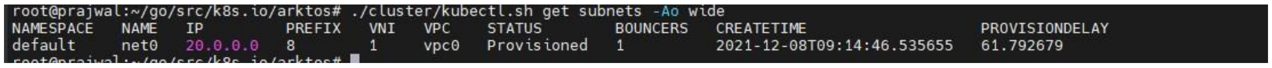


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get vpcs -Ao wide
NAMESPACE   NAME   IP         PREFIX   VNI   DIVIDERS   STATUS        CREATETIME                   PROVISIONDELAY
default     vpc0   20.0.0.0   8        1     1          Provisioned   2021-12-08T09:14:46.439974   41.702689
```

4) Check subnets

```
./cluster/kubectl.sh get subnets -Ao wide
```

Output

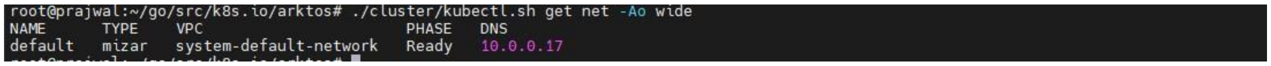


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get subnets -Ao wide
NAMESPACE   NAME   IP         PREFIX   VNI   VPC    STATUS        BOUNCERS   CREATETIME                   PROVISIONDELAY
default     net0   20.0.0.0   8        1     vpc0   Provisioned   1          2021-12-08T09:14:46.535655   61.792679
```

5) Check net

./cluster/kubectl.sh get net -Ao wide

Output

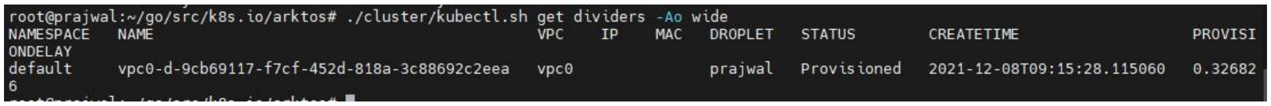


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get net -Ao wide
NAME      TYPE    VPC                      PHASE   DNS
default   mizar   system-default-network   Ready   10.0.0.17
```

6) Check dividers

./cluster/kubectl.sh get dividers -Ao wide

Output

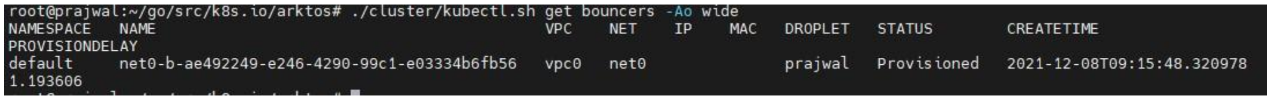


```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get dividers -Ao wide
NAMESPACE   NAME                                         VPC    IP    MAC   DROPLET   STATUS        CREATETIME                   PROVISI
ONDELAY
default     vpc0-d-9cb69117-f7cf-452d-818a-3c88692c2eea  vpc0               prajwal   Provisioned   2021-12-08T09:15:28.115060   0.32682
6
```

7) Check bouncers

./cluster/kubectl.sh get bouncers -Ao wide

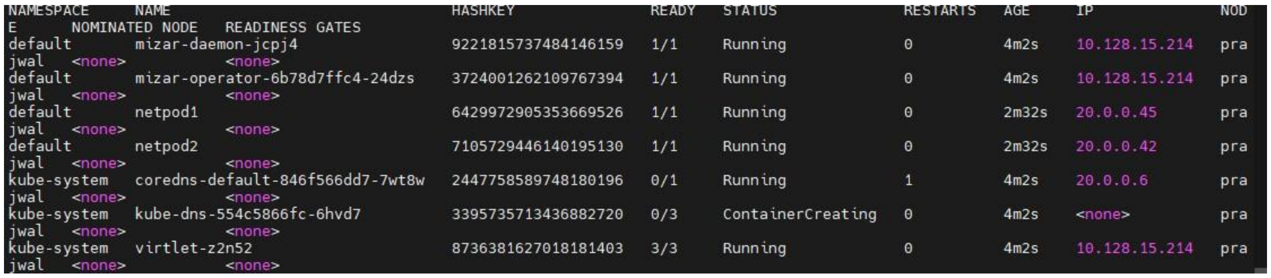
Output



```
root@prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get bouncers -Ao wide
NAMESPACE   NAME                                         VPC    NET    IP    MAC   DROPLET   STATUS        CREATETIME
PROVISIONDELAY
default     net0-b-ae492249-e246-4290-99c1-e03334b6fb56  vpc0   net0               prajwal   Provisioned   2021-12-08T09:15:48.320978
1.193606
```

8) Pod deployment:

Output



```
NAMESPACE     NAME                                HASHKEY               READY   STATUS              RESTARTS   AGE     IP              NOD
E       NOMINATED NODE   READINESS GATES
default       mizar-daemon-jcpj4                  9221815737484146159   1/1     Running             0          4m2s    10.128.15.214   pra
jwal    <none>           <none>
default       mizar-operator-6b78d7ffc4-24dzs     3724001262109767394   1/1     Running             0          4m2s    10.128.15.214   pra
jwal    <none>           <none>
default       netpod1                             6429972905353669526   1/1     Running             0          2m32s   20.0.0.45       pra
jwal    <none>           <none>
default       netpod2                             7105729446140195130   1/1     Running             0          2m32s   20.0.0.42       pra
jwal    <none>           <none>
kube-system   coredns-default-846f566dd7-7wt8w    2447758589748180196   0/1     Running             1          4m2s    20.0.0.6        pra
jwal    <none>           <none>
kube-system   kube-dns-554c5866fc-6hvd7           3395735713436882720   0/3     ContainerCreating   0          4m2s    <none>          pra
jwal    <none>           <none>
kube-system   virtlet-z2n52                       8736381627018181403   3/3     Running             0          4m2s    10.128.15.214   pra
jwal    <none>           <none>
```

*Pod getting stuck in **ContainerCreating** state.*