Deploying KubeEdge On premises

User will need to run following steps on workstation machine(the recommended instance size should be atleast 16 CPU and 32GB RAM and the storage size should be 150GB or more)

CentOS distribution of linux is preferred.

Limitation

- Currently support of `keadm` is available for Ubuntu and CentOS OS. RaspberryPi supports is in-progress.
- Need super user rights (or root rights) to run.

Dependencies

For cloud side, we need:

- Kubernetes cluster (preferred 1.21 version)

For edge side, we need:

- Container runtimes, now we support:
  - Docker
  - Containerd
  - Cri-o
  - Virtlet

Docker installation:-

Please follow the link for installation:-

https://docs.docker.com/engine/install/centos/

Kubernetes Installation (1.21 version recommended)

Please follow the link for installation

https://kubernetes.io/docs/tasks/tools/

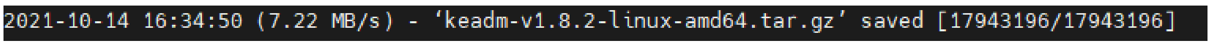
Lets start kube-edge installation process

Install keadm

Command

wget https://github.com/kubeedge/kubeedge/releases/download/v1.8.2/keadm-v1.8.2-linux-amd64.tar.gz

output:

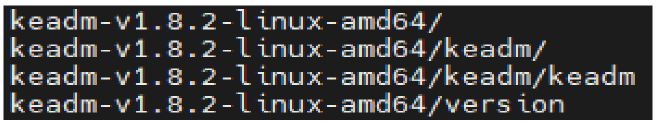


```
2021-10-14 16:34:50 (7.22 MB/s) - ‘keadm-v1.8.2-linux-amd64.tar.gz’ saved [17943196/17943196]
```

Command

tar -xvzf keadm-v1.8.2-linux-amd64.tar.gz

Output:



```
keadm-v1.8.2-linux-amd64/
keadm-v1.8.2-linux-amd64/keadm/
keadm-v1.8.2-linux-amd64/keadm/keadm
keadm-v1.8.2-linux-amd64/version
```

Command

cd /root/keadm-v1.8.2-linux-amd64/keadm

cp keadm /usr/local/sbin

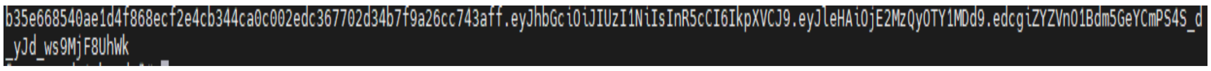
Command

keadm init --advertise-address=”ip addresse of node”

Command

`keadm gettoken`

Output



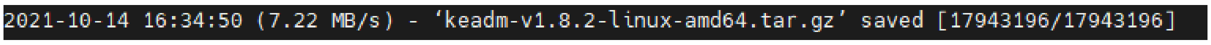
```
b35e668540ae1d4f868ecf2e4cb344ca0c002edc367702d34b7f9a26cc743aff.eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE2MzQyOTY1MDd9.edcgiZYZVnO1Bdm5GeYCmPS4S_d
_yJd_ws9MjF8UhWk
```

ON Edgecore:

Command

wget https://github.com/kubeedge/kubeedge/releases/download/v1.8.2/keadm-v1.8.2-linux-amd64.tar.gz

output

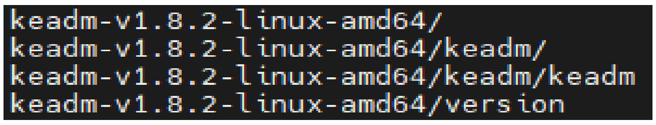


```
2021-10-14 16:34:50 (7.22 MB/s) - ‘keadm-v1.8.2-linux-amd64.tar.gz’ saved [17943196/17943196]
```

Command

tar -xvzf keadm-v1.8.2-linux-amd64.tar.gz

Output:



```
keadm-v1.8.2-linux-amd64/
keadm-v1.8.2-linux-amd64/keadm/
keadm-v1.8.2-linux-amd64/keadm/keadm
keadm-v1.8.2-linux-amd64/version
```

Command

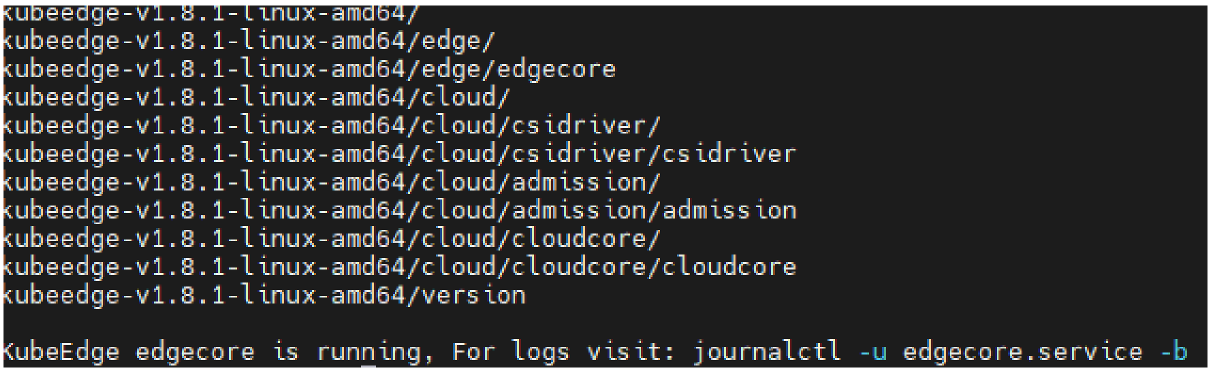
cd /root/keadm-v1.8.2-linux-amd64/keadm

cp keadm /usr/local/sbin

Command

```
keadm join --cloudcore-ipport=”ip of cloudcore node”:10000 --
token=”paste token from cloudcore node”
```

output



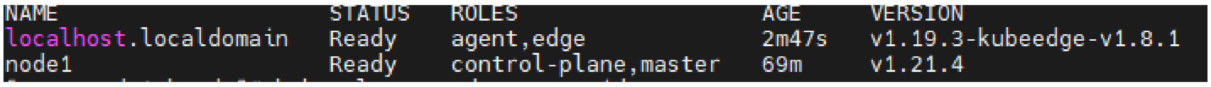
```
kubeedge-v1.8.1-linux-amd64/
kubeedge-v1.8.1-linux-amd64/edge/
kubeedge-v1.8.1-linux-amd64/edge/edgecore
kubeedge-v1.8.1-linux-amd64/cloud/
kubeedge-v1.8.1-linux-amd64/cloud/csidriver/
kubeedge-v1.8.1-linux-amd64/cloud/csidriver/csidriver
kubeedge-v1.8.1-linux-amd64/cloud/admission/
kubeedge-v1.8.1-linux-amd64/cloud/admission/admission
kubeedge-v1.8.1-linux-amd64/cloud/cloudcore/
kubeedge-v1.8.1-linux-amd64/cloud/cloudcore/cloudcore
kubeedge-v1.8.1-linux-amd64/version

KubeEdge edgecore is running, For logs visit: journalctl -u edgecore.service -b
```

On cloudcore node check whether the kubeedge node is ready

Command

kubectl get nodes

output:



```
NAME                    STATUS   ROLES                  AGE     VERSION
localhost.localdomain   Ready    agent,edge             2m47s   v1.19.3-kubeedge-v1.8.1
node1                   Ready    control-plane,master   69m     v1.21.4
```

Edge node is joined successfully.

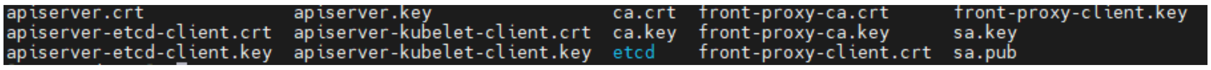
Enable `kubectl logs` Feature

Before metric server deployed kubectl logs feature must be activated

Command:

```
ls /etc/kubernetes/pki/
```

output:

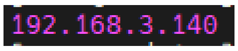


```
apiserver.crt                 apiserver.key                 ca.crt  front-proxy-ca.crt      front-proxy-client.key
apiserver-etcd-client.crt     apiserver-kubelet-client.crt  ca.key  front-proxy-ca.key      sa.key
apiserver-etcd-client.key     apiserver-kubelet-client.key  etcd    front-proxy-client.crt  sa.pub
```

Command:

```
export CLOUDCOREIPS="Cloudcore Ip addresse"

echo $CLOUDCOREIPS
```

output:

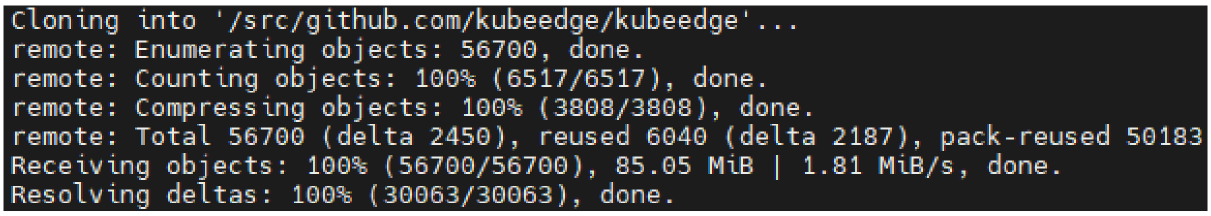


```
192.168.3.140
```

Command:

git clone https://github.com/kubeedge/kubeedge.git $GOPATH/src/github.com/kubeedge/kubeedge

Output:



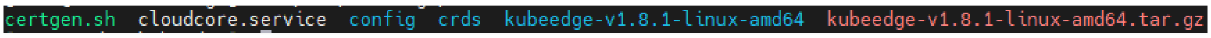
```
Cloning into '/src/github.com/kubeedge/kubeedge'...
remote: Enumerating objects: 56700, done.
remote: Counting objects: 100% (6517/6517), done.
remote: Compressing objects: 100% (3808/3808), done.
remote: Total 56700 (delta 2450), reused 6040 (delta 2187), pack-reused 50183
Receiving objects: 100% (56700/56700), 85.05 MiB | 1.81 MiB/s, done.
Resolving deltas: 100% (30063/30063), done.
```

Command:

```
cp $GOPATH/src/github.com/kubeedge/kubeedge/build/tools/certgen.sh
/etc/kubeedge/
```

```
ls /etc/kubeedge/
```

output:

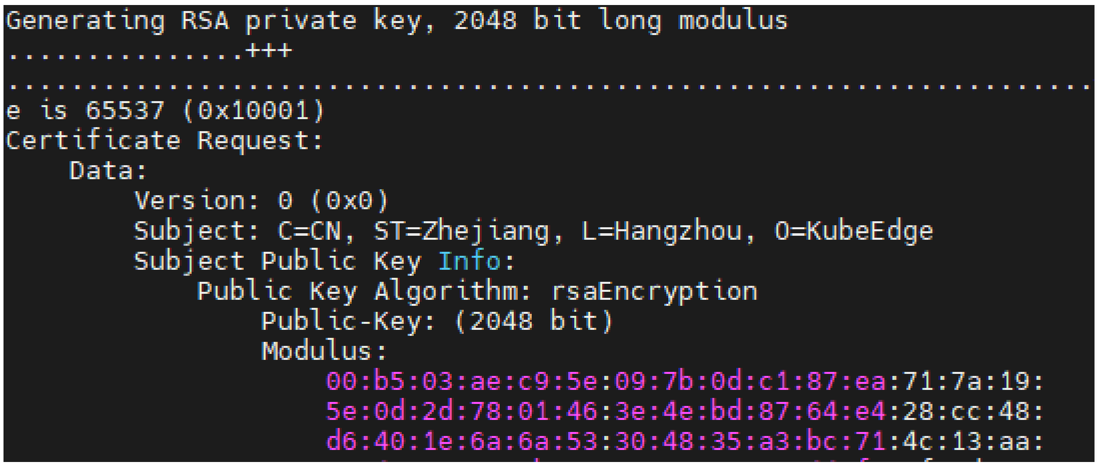


```
certgen.sh  cloudcore.service  config  crds  kubeedge-v1.8.1-linux-amd64  kubeedge-v1.8.1-linux-amd64.tar.gz
```

Command:

```
/etc/kubeedge/certgen.sh stream
```

Output:



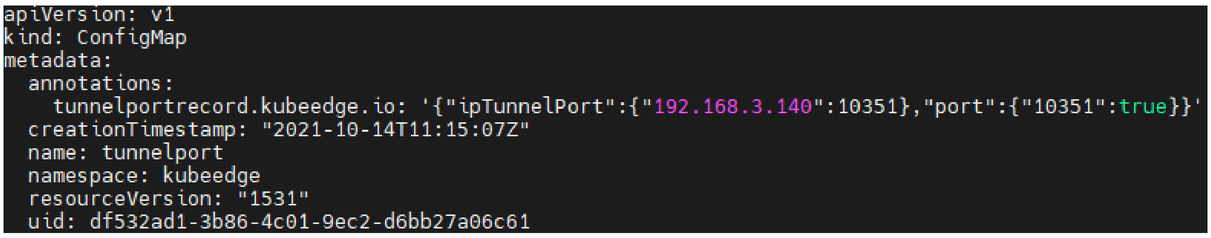
```
Generating RSA private key, 2048 bit long modulus
...............+++
.......................................................................
e is 65537 (0x10001)
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=CN, ST=Zhejiang, L=Hangzhou, O=KubeEdge
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b5:03:ae:c9:5e:09:7b:0d:c1:87:ea:71:7a:19:
                    5e:0d:2d:78:01:46:3e:4e:bd:87:64:e4:28:cc:48:
                    d6:40:1e:6a:6a:53:30:48:35:a3:bc:71:4c:13:aa:
```

Note:You need to get the configmap first, which contains all the cloudcore ips and tunnel ports.

Command:

```
kubectl get cm tunnelport -nkubeedge -o yaml
```

Output:

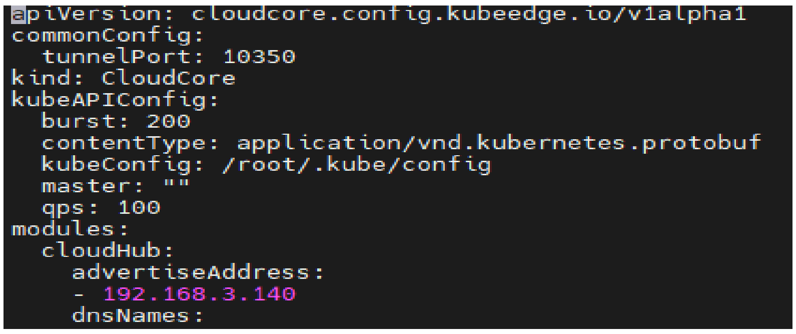


```
apiVersion: v1
kind: ConfigMap
metadata:
  annotations:
    tunnelportrecord.kubeedge.io: '{"ipTunnelPort":{"192.168.3.140":10351},"port":{"10351":true}}'
  creationTimestamp: "2021-10-14T11:15:07Z"
  name: tunnelport
  namespace: kubeedge
  resourceVersion: "1531"
  uid: df532ad1-3b86-4c01-9ec2-d6bb27a06c61
```

`Command:`

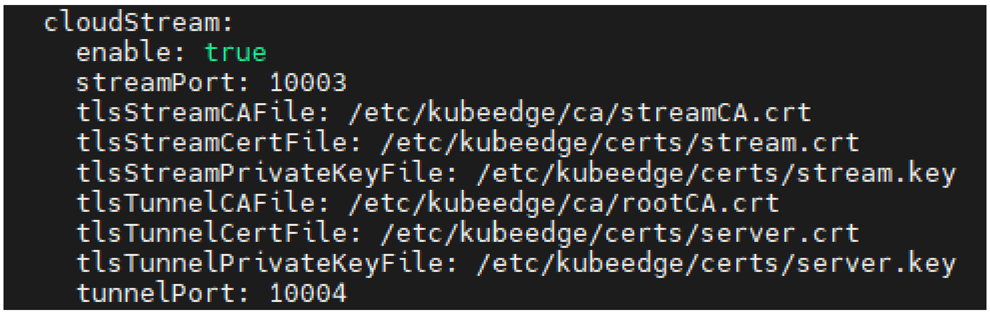
Vi /etc/kubeedge/config/cloudcore.yaml

Output:



```
apiVersion: cloudcore.config.kubeedge.io/v1alpha1
commonConfig:
  tunnelPort: 10350
kind: CloudCore
kubeAPIConfig:
  burst: 200
  contentType: application/vnd.kubernetes.protobuf
  kubeConfig: /root/.kube/config
  master: ""
  qps: 100
modules:
  cloudHub:
    advertiseAddress:
    - 192.168.3.140
    dnsNames:
```

Modify the file in the following part (`enable: true`):



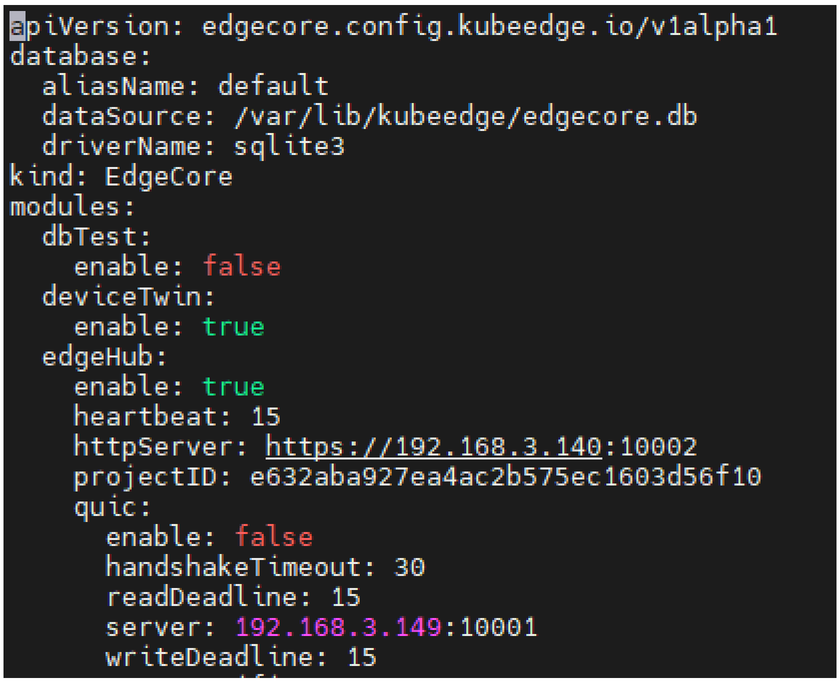
```
  cloudStream:
    enable: true
    streamPort: 10003
    tlsStreamCAFile: /etc/kubeedge/ca/streamCA.crt
    tlsStreamCertFile: /etc/kubeedge/certs/stream.crt
    tlsStreamPrivateKeyFile: /etc/kubeedge/certs/stream.key
    tlsTunnelCAFile: /etc/kubeedge/ca/rootCA.crt
    tlsTunnelCertFile: /etc/kubeedge/certs/server.crt
    tlsTunnelPrivateKeyFile: /etc/kubeedge/certs/server.key
    tunnelPort: 10004
```

Now login to edgecore node

Command:

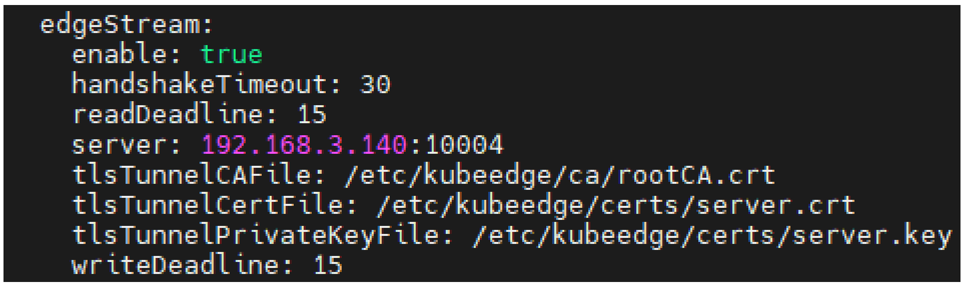
`vi /etc/kubeedge/config/edgecore.yaml`

output:



```
apiVersion: edgecore.config.kubeedge.io/v1alpha1
database:
  aliasName: default
  dataSource: /var/lib/kubeedge/edgecore.db
  driverName: sqlite3
kind: EdgeCore
modules:
  dbTest:
    enable: false
  deviceTwin:
    enable: true
  edgeHub:
    enable: true
    heartbeat: 15
    httpServer: https://192.168.3.140:10002
    projectID: e632aba927ea4ac2b575ec1603d56f10
    quic:
      enable: false
      handshakeTimeout: 30
      readDeadline: 15
      server: 192.168.3.149:10001
      writeDeadline: 15
```

`Note:` Modify the file in the following part (`enable: true`), (`server:cloudcoreIP:10004`) :



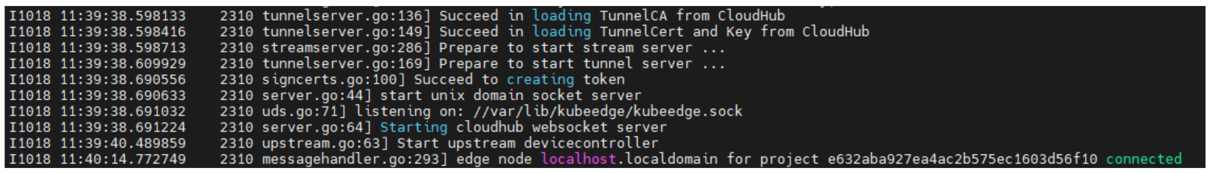
```
  edgeStream:
    enable: true
    handshakeTimeout: 30
    readDeadline: 15
    server: 192.168.3.140:10004
    tlsTunnelCAFile: /etc/kubeedge/ca/rootCA.crt
    tlsTunnelCertFile: /etc/kubeedge/certs/server.crt
    tlsTunnelPrivateKeyFile: /etc/kubeedge/certs/server.key
    writeDeadline: 15
```

`Now on Cloudcore node:`

`Command:`

```
pkill cloudcore
nohup cloudcore > cloudcore.log 2>&1 &
tail -f cloudcore.log
```

`Output:`

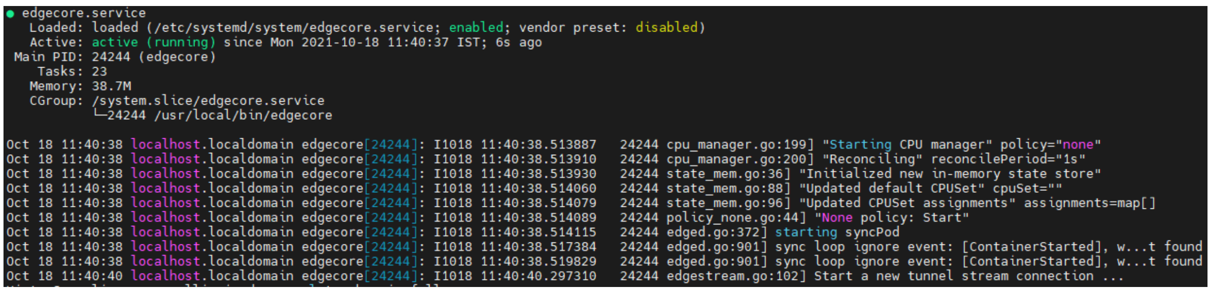


```
I1018 11:39:38.598133    2310 tunnelserver.go:136] Succeed in loading TunnelCA from CloudHub
I1018 11:39:38.598416    2310 tunnelserver.go:149] Succeed in loading TunnelCert and Key from CloudHub
I1018 11:39:38.598713    2310 streamserver.go:286] Prepare to start stream server ...
I1018 11:39:38.609929    2310 tunnelserver.go:169] Prepare to start tunnel server ...
I1018 11:39:38.690556    2310 signcerts.go:100] Succeed to creating token
I1018 11:39:38.690633    2310 server.go:44] start unix domain socket server
I1018 11:39:38.691032    2310 uds.go:71] listening on: //var/lib/kubeedge/kubeedge.sock
I1018 11:39:38.691224    2310 server.go:64] Starting cloudhub websocket server
I1018 11:39:40.489859    2310 upstream.go:63] Start upstream devicecontroller
I1018 11:40:14.772749    2310 messagehandler.go:293] edge node localhost.localdomain for project e632aba927ea4ac2b575ec1603d56f10 connected
```

`Restart edgecore service on edgecore node:`

`Command:_systemctl restart edgecore.service`

Output:



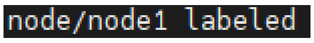
```
● edgecore.service
   Loaded: loaded (/etc/systemd/system/edgecore.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2021-10-18 11:40:37 IST; 6s ago
 Main PID: 24244 (edgecore)
    Tasks: 23
   Memory: 38.7M
   CGroup: /system.slice/edgecore.service
           └─24244 /usr/local/bin/edgecore

Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.513887   24244 cpu_manager.go:199] "Starting CPU manager" policy="none"
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.513910   24244 cpu_manager.go:200] "Reconciling" reconcilePeriod="1s"
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.513930   24244 state_mem.go:36] "Initialized new in-memory state store"
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.514060   24244 state_mem.go:88] "Updated default CPUSet" cpuSet=""
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.514079   24244 state_mem.go:96] "Updated CPUSet assignments" assignments=map[]
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.514089   24244 policy_none.go:44] "None policy: Start"
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.514115   24244 edged.go:372] starting syncPod
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.517384   24244 edged.go:901] sync loop ignore event: [ContainerStarted], w...t found
Oct 18 11:40:38 localhost.localdomain edgecore[24244]: I1018 11:40:38.519829   24244 edged.go:901] sync loop ignore event: [ContainerStarted], w...t found
Oct 18 11:40:40 localhost.localdomain edgecore[24244]: I1018 11:40:40.297310   24244 edgestream.go:102] Start a new tunnel stream connection ...
```

Metric server Deployment:

Command:

kubectl label node node1 app=engine

Output:

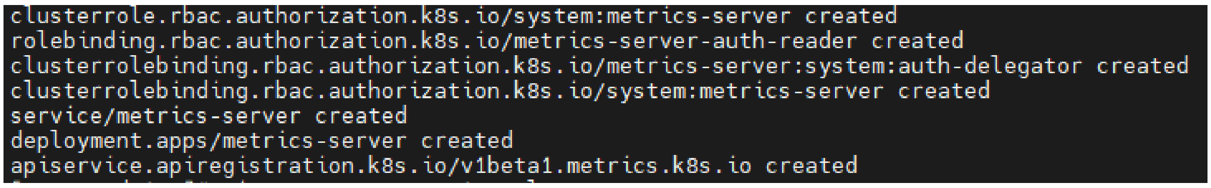


```
node/node1 labeled
```

Command:

Kubectl apply -f components-0.5.1.yaml

Output:

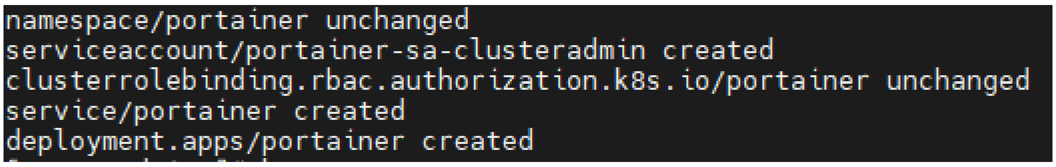


```
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
```

Dashboard Deployment:

Kubectl apply -f portainer.yaml

Output:



```
namespace/portainer unchanged
serviceaccount/portainer-sa-clusteradmin created
clusterrolebinding.rbac.authorization.k8s.io/portainer unchanged
service/portainer created
deployment.apps/portainer created
```

Visit the link below for kubeedge dashboard:

http://cloudcoreip:30777/