Test report - Deployment of Arktos Cluster without Mizar CNI on GCP

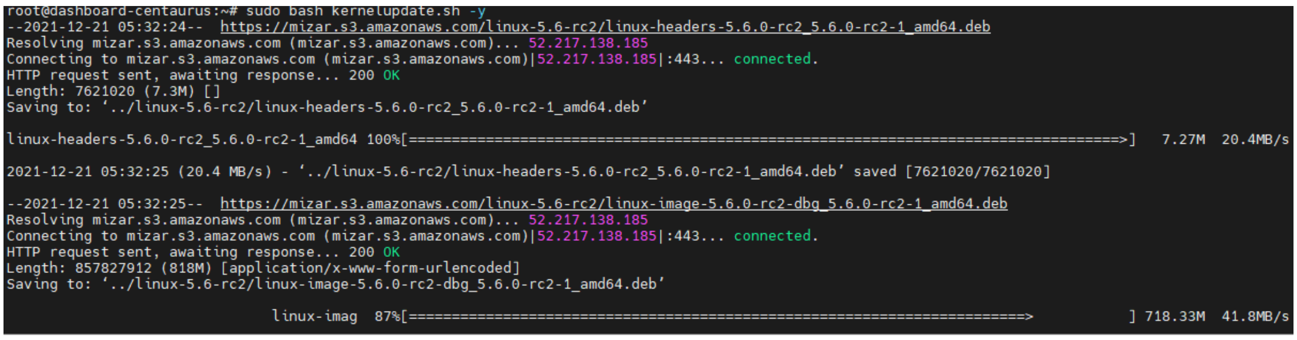
This document captures the steps to deploy an Arktos cluster lab without Mizar CNI. The machine in this lab used are **e2-standard-8 (8 vCPUs,32 GB Memory) and Ubuntu 18.04 LTS.**

Date-21 Dec. 2021

Step-1: Update kernel (If required)

To check kernel, run the following command

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



```
root@dashboard-centaurus:~# sudo bash kernelupdate.sh -y
--2021-12-21 05:32:24--  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)... 52.217.138.185
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|52.217.138.185|: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-rc2-1_amd64 100%[=============================================================================>]   7.27M  20.4MB/s

2021-12-21 05:32:25 (20.4 MB/s) - ‘../linux-5.6-rc2/linux-headers-5.6.0-rc2_5.6.0-rc2-1_amd64.deb’ saved [7621020/7621020]

--2021-12-21 05:32:25--  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)... 52.217.138.185
Connecting to mizar.s3.amazonaws.com (mizar.s3.amazonaws.com)|52.217.138.185|: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-imag  87%[=====================================================================>          ] 718.33M  41.8MB/s
```

Step-2: Install dependencies

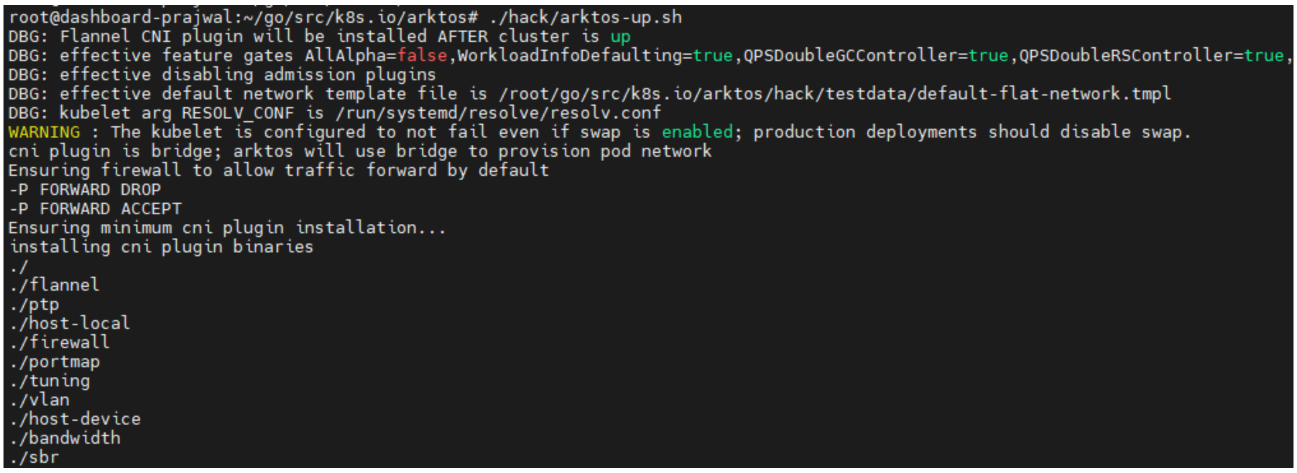
Run the following steps to install dependencies required for arktos deployment:

```
git clone https://github.com/Click2Cloud-Centaurus/arktos.git   ~/go/src/k8s.io/arktos
cd ~/go/src/k8s.io/arktos
sudo bash ./hack/setup-dev-node.sh
```

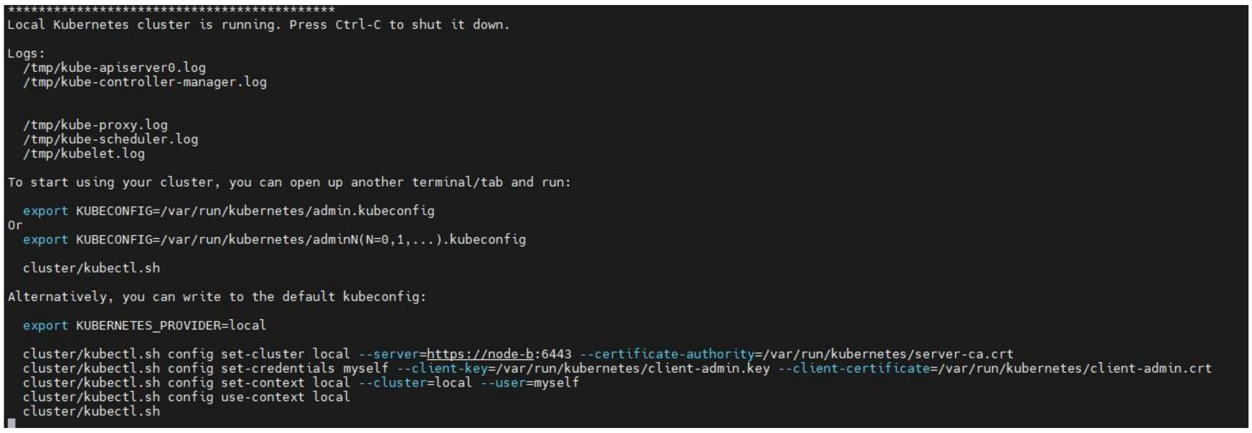
Run Arktos

The easiest way to run Arktos is to bring up a single-node cluster in your local development box:

```
echo export PATH=$PATH:/usr/local/go/bin\ >> ~/.profile
echo cd \$HOME/go/src/k8s.io/arktos >> ~/.profile
source ~/.profile

./hack/arktos-up.sh
```



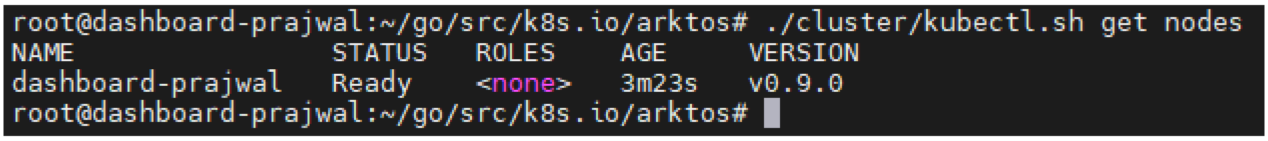
```
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./hack/arktos-up.sh
DBG: Flannel CNI plugin will be installed AFTER cluster is up
DBG: effective feature gates AllAlpha=false,WorkloadInfoDefaulting=true,QPSDoubleGCController=true,QPSDoubleRSController=true,
DBG: effective disabling admission plugins
DBG: effective default network template file is /root/go/src/k8s.io/arktos/hack/testdata/default-flat-network.tmpl
DBG: kubelet arg RESOLV_CONF is /run/systemd/resolve/resolv.conf
WARNING : The kubelet is configured to not fail even if swap is enabled; production deployments should disable swap.
cni plugin is bridge; arktos will use bridge to provision pod network
Ensuring firewall to allow traffic forward by default
-P FORWARD DROP
-P FORWARD ACCEPT
Ensuring minimum cni plugin installation...
installing cni plugin binaries
./
./flannel
./ptp
./host-local
./firewall
./portmap
./tuning
./vlan
./host-device
./bandwidth
./sbr
```



```
*******************************************
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://node-b: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/kubernetes/client-admin.crt
  cluster/kubectl.sh config set-context local --cluster=local --user=myself
  cluster/kubectl.sh config use-context local
  cluster/kubectl.sh
```

1) Check nodes status:

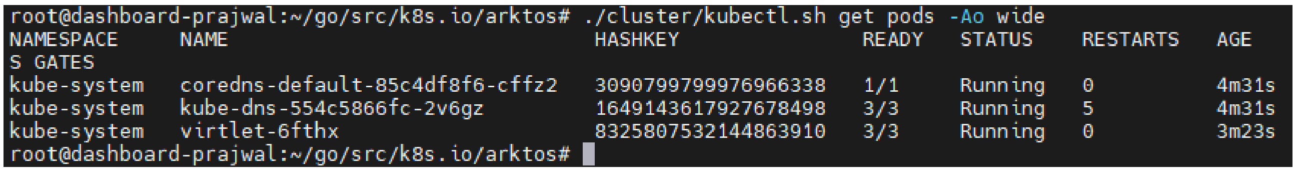
./cluster/kubectl.sh get nodes



```
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get nodes
NAME                STATUS   ROLES    AGE     VERSION
dashboard-prajwal   Ready    <none>   3m23s   v0.9.0
root@dashboard-prajwal:~/go/src/k8s.io/arktos#
```

2) Check pods status:

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



```
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get pods -Ao wide
NAMESPACE     NAME                                    HASHKEY               READY   STATUS     RESTARTS   AGE
S GATES
kube-system   coredns-default-85c4df8f6-cffz2         3090799799976966338   1/1     Running    0          4m31s
kube-system   kube-dns-554c5866fc-2v6gz               1649143617927678498   3/3     Running    5          4m31s
kube-system   virtlet-6fthx                           8325807532144863910   3/3     Running    0          3m23s
root@dashboard-prajwal:~/go/src/k8s.io/arktos#
```

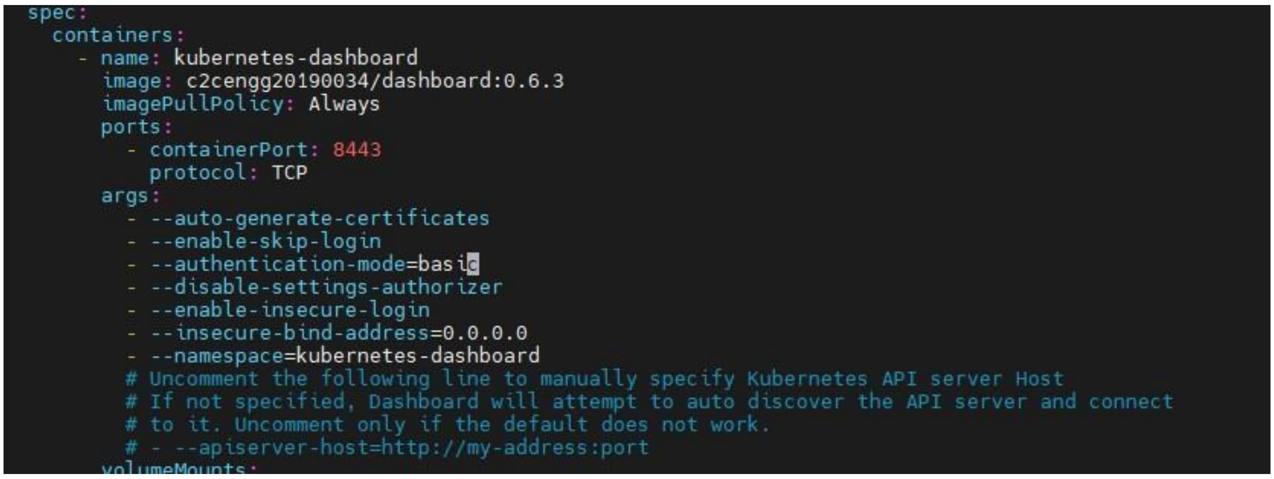
Deployment of Centaurus dashboard:

Link for YAML file of the dashboard:

https://click2cloud-my.sharepoint.com/personal/amit_nagpure_click2cloud_net/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Famit%5Fnagpure%5Fclick2cloud%5Fnet%2FDocuments%2FMicrosoft%20Teams%20Chat%20Files%2Fkubernetes%2Ddashboard%2Eyaml&parent=%2Fpersonal%2Famit%5Fnagpure%5Fclick2cloud%5Fnet%2FDocuments%2FMicrosoft%20Teams%20Chat%20Files

Create YAML file naming 'kubernetes-dashboard.yaml' change image c2c/…..0.6.3

and in args input '—authentication-mode=basic'



```
spec:
  containers:
    - name: kubernetes-dashboard
      image: c2cengg20190034/dashboard:0.6.3
      imagePullPolicy: Always
      ports:
        - containerPort: 8443
          protocol: TCP
      args:
        - --auto-generate-certificates
        - --enable-skip-login
        - --authentication-mode=basic
        - --disable-settings-authorizer
        - --enable-insecure-login
        - --insecure-bind-address=0.0.0.0
        - --namespace=kubernetes-dashboard
        # Uncomment the following line to manually specify Kubernetes API server Host
        # If not specified, Dashboard will attempt to auto discover the API server and connect
        # to it. Uncomment only if the default does not work.
        # - --apiserver-host=http://my-address:port
      volumeMounts:
```

Create the Centaurus Dashboard password file.

```
mkdir /etc/kubernetes/auth -p
```

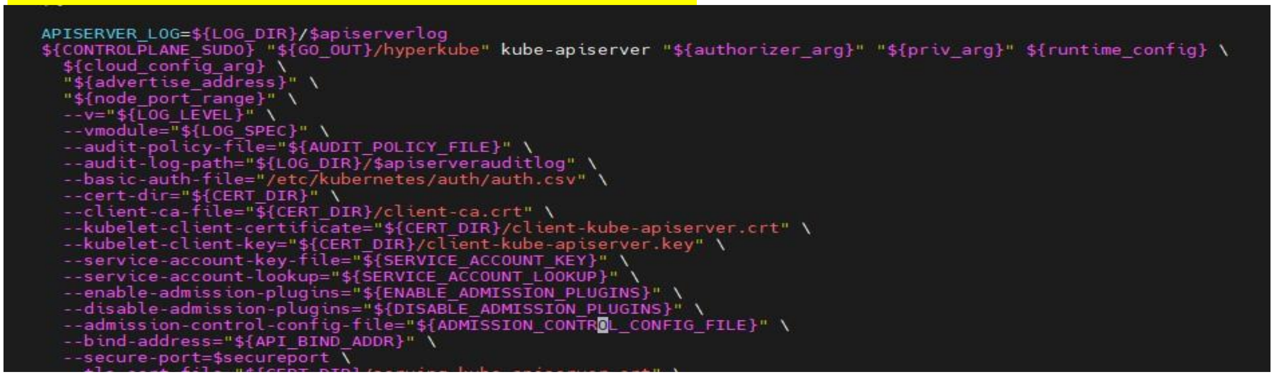
```
vim /etc/kubernetes/auth/auth.csv
```

add following text which has password/token,username,userid, groups respectively

```
password,admin,admin,system:masters
```

we need to configure while deploying the arktos the following entry in 'common.sh'

```
vi /hack/lib/common.sh
350    - --basic-auth-file=/etc/kubernetes/auth/auth.csv
```

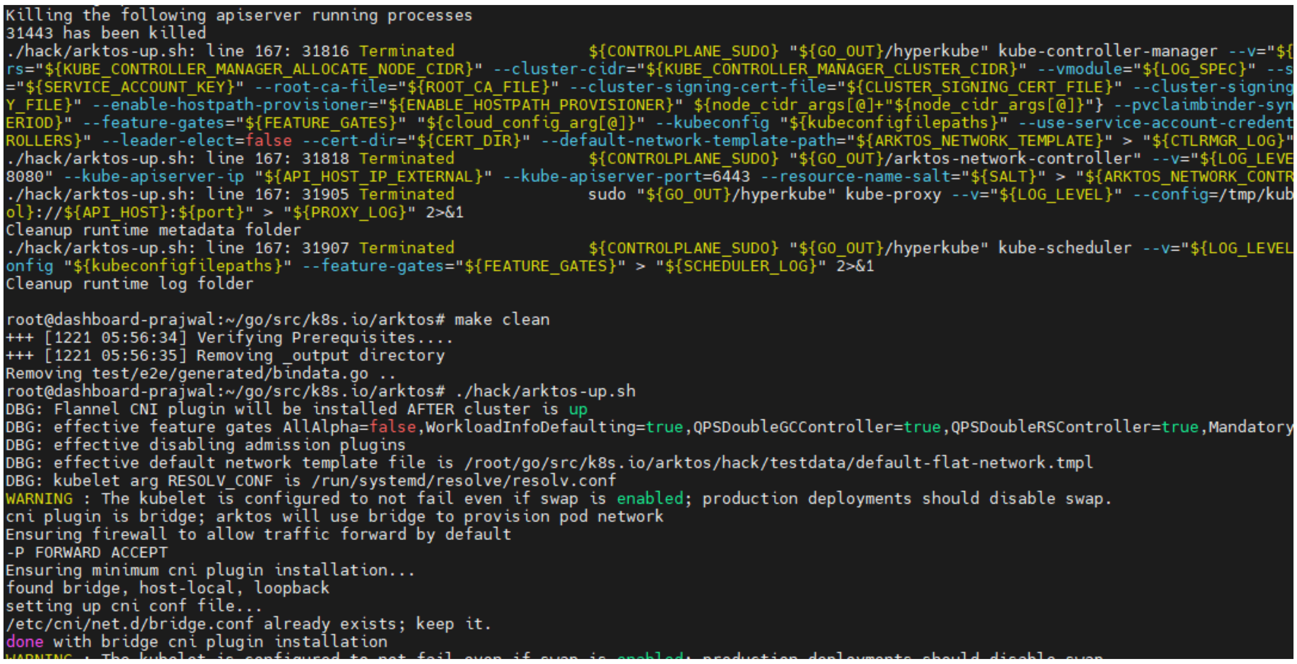


```
APISERVER_LOG=${LOG_DIR}/$apiserverlog
${CONTROLPLANE_SUDO} "${GO_OUT}/hyperkube" kube-apiserver "${authorizer_arg}" "${priv_arg}" ${runtime_config} \
  ${cloud_config_arg} \
  "${advertise_address}" \
  "${node_port_range}" \
  --v="${LOG_LEVEL}" \
  --vmodule="${LOG_SPEC}" \
  --audit-policy-file="${AUDIT_POLICY_FILE}" \
  --audit-log-path="${LOG_DIR}/$apiserverauditlog" \
  --basic-auth-file="/etc/kubernetes/auth/auth.csv" \
  --cert-dir="${CERT_DIR}" \
  --client-ca-file="${CERT_DIR}/client-ca.crt" \
  --kubelet-client-certificate="${CERT_DIR}/client-kube-apiserver.crt" \
  --kubelet-client-key="${CERT_DIR}/client-kube-apiserver.key" \
  --service-account-key-file="${SERVICE_ACCOUNT_KEY}" \
  --service-account-lookup="${SERVICE_ACCOUNT_LOOKUP}" \
  --enable-admission-plugins="${ENABLE_ADMISSION_PLUGINS}" \
  --disable-admission-plugins="${DISABLE_ADMISSION_PLUGINS}" \
  --admission-control-config-file="${ADMISSION_CONTROL_CONFIG_FILE}" \
  --bind-address="${API_BIND_ADDR}" \
  --secure-port=$secureport \
```

Now re-run the arktos script:

make clean

hack/arktos-up.sh



```
Killing the following apiserver running processes
31443 has been killed
./hack/arktos-up.sh: line 167: 31816 Terminated              ${CONTROLPLANE_SUDO} "${GO_OUT}/hyperkube" kube-controller-manager --v="${
rs="${KUBE_CONTROLLER_MANAGER_ALLOCATE_NODE_CIDR}" --cluster-cidr="${KUBE_CONTROLLER_MANAGER_CLUSTER_CIDR}" --vmodule="${LOG_SPEC}" --s
="${SERVICE_ACCOUNT_KEY}" --root-ca-file="${ROOT_CA_FILE}" --cluster-signing-cert-file="${CLUSTER_SIGNING_CERT_FILE}" --cluster-signing
Y_FILE}" --enable-hostpath-provisioner="${ENABLE_HOSTPATH_PROVISIONER}" ${node_cidr_args[@]+"${node_cidr_args[@]}"} --pvclaimbinder-syn
ERIOD}" --feature-gates="${FEATURE_GATES}" "${cloud_config_arg[@]}" --kubeconfig "${kubeconfigfilepaths}" --use-service-account-credent
ROLLERS}" --leader-elect=false --cert-dir="${CERT_DIR}" --default-network-template-path="${ARKTOS_NETWORK_TEMPLATE}" > "${CTLRMGR_LOG}"
./hack/arktos-up.sh: line 167: 31818 Terminated              ${CONTROLPLANE_SUDO} "${GO_OUT}/arktos-network-controller" --v="${LOG_LEVE
8080" --kube-apiserver-ip "${API_HOST_IP_EXTERNAL}" --kube-apiserver-port=6443 --resource-name-salt="${SALT}" > "${ARKTOS_NETWORK_CONTR
./hack/arktos-up.sh: line 167: 31905 Terminated              sudo "${GO_OUT}/hyperkube" kube-proxy --v="${LOG_LEVEL}" --config=/tmp/kub
ol}://${API_HOST}:${port}" > "${PROXY_LOG}" 2>&1
Cleanup runtime metadata folder
./hack/arktos-up.sh: line 167: 31907 Terminated              ${CONTROLPLANE_SUDO} "${GO_OUT}/hyperkube" kube-scheduler --v="${LOG_LEVEL
onfig "${kubeconfigfilepaths}" --feature-gates="${FEATURE_GATES}" > "${SCHEDULER_LOG}" 2>&1
Cleanup runtime log folder

root@dashboard-prajwal:~/go/src/k8s.io/arktos# make clean
+++ [1221 05:56:34] Verifying Prerequisites....
+++ [1221 05:56:35] Removing _output directory
Removing test/e2e/generated/bindata.go ..
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./hack/arktos-up.sh
DBG: Flannel CNI plugin will be installed AFTER cluster is up
DBG: effective feature gates AllAlpha=false,WorkloadInfoDefaulting=true,QPSDoubleGCController=true,QPSDoubleRSController=true,Mandatory
DBG: effective disabling admission plugins
DBG: effective default network template file is /root/go/src/k8s.io/arktos/hack/testdata/default-flat-network.tmpl
DBG: kubelet arg RESOLV_CONF is /run/systemd/resolve/resolv.conf
WARNING : The kubelet is configured to not fail even if swap is enabled; production deployments should disable swap.
cni plugin is bridge; arktos will use bridge to provision pod network
Ensuring firewall to allow traffic forward by default
-P FORWARD ACCEPT
Ensuring minimum cni plugin installation...
found bridge, host-local, loopback
setting up cni conf file...
/etc/cni/net.d/bridge.conf already exists; keep it.
done with bridge cni plugin installation
WARNING : The kubelet is configured to not fail even if swap is enabled; production deployments should disable swap.
```

Input the following commands before deploying the dashboard:

sudo sed -i '0,/RANDFILE/{s/RANDFILE/\#&/}' /etc/ssl/openssl.cnf

openssl genrsa -out dashboard.key 2048
openssl rsa -in dashboard.key -out dashboard.key

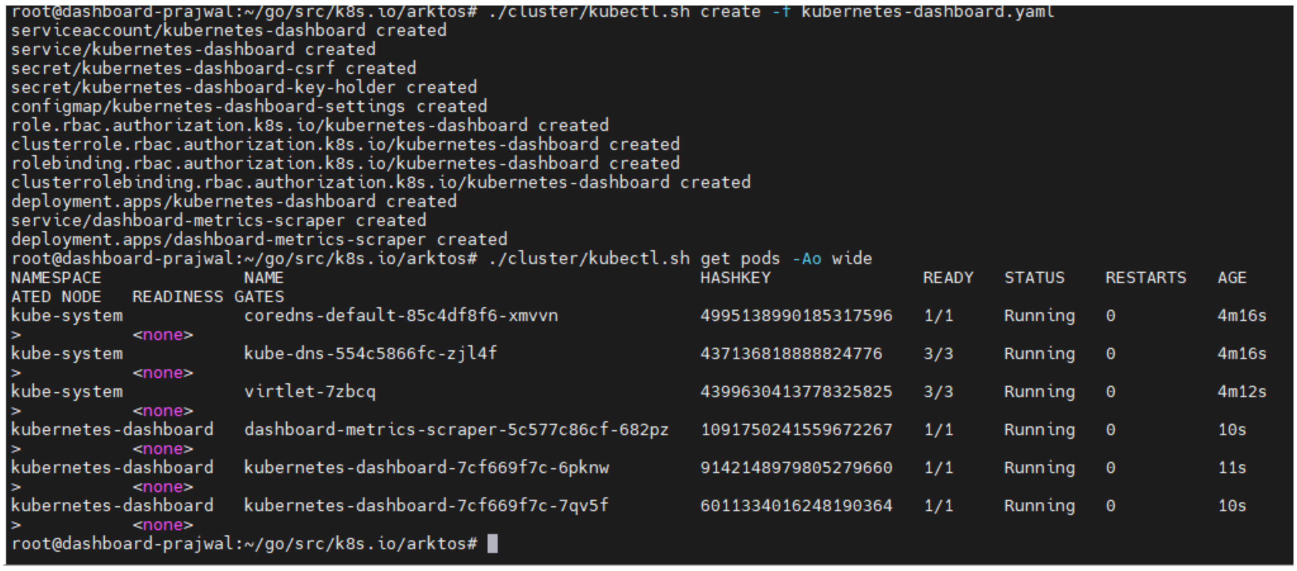
openssl req -sha256 -new -key dashboard.key -out dashboard.csr -subj "/CN=$(hostname -I | awk '{print $1}')"

openssl x509 -req -sha256 -days 365 -in dashboard.csr -signkey dashboard.key -out dashboard.crt

./cluster/kubectl.sh create namespace kubernetes-dashboard

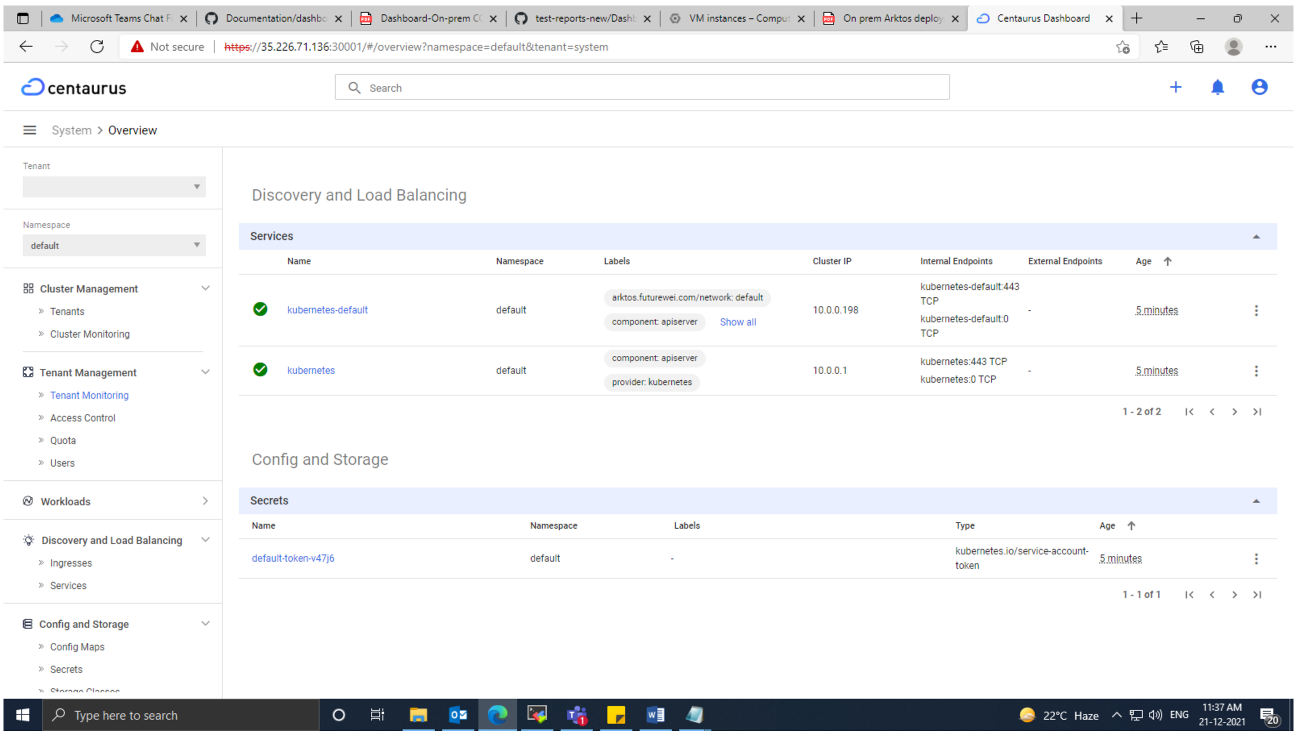
./cluster/kubectl.sh create secret generic kubernetes-dashboard-certs --fromfile=$HOME/dashboard.key --from-file=$HOME/dashboard.crt -n kubernetes-dashboard

./cluster/kubectl.sh create -f kubernetes-dashboard.yaml



```
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh create -f kubernetes-dashboard.yaml
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
root@dashboard-prajwal:~/go/src/k8s.io/arktos# ./cluster/kubectl.sh get pods -Ao wide
NAMESPACE              NAME                                         HASHKEY               READY   STATUS    RESTARTS   AGE
ATED NODE   READINESS GATES
kube-system            coredns-default-85c4df8f6-xmvvn              4995138990185317596   1/1     Running   0          4m16s
>           <none>
kube-system            kube-dns-554c5866fc-zjl4f                    437136818888824776    3/3     Running   0          4m16s
>           <none>
kube-system            virtlet-7zbcq                                4399630413778325825   3/3     Running   0          4m12s
>           <none>
kubernetes-dashboard   dashboard-metrics-scraper-5c577c86cf-682pz   1091750241559672267   1/1     Running   0          10s
>           <none>
kubernetes-dashboard   kubernetes-dashboard-7cf669f7c-6pknw         9142148979805279660   1/1     Running   0          11s
>           <none>
kubernetes-dashboard   kubernetes-dashboard-7cf669f7c-7qv5f         6011334016248190364   1/1     Running   0          10s
>           <none>
root@dashboard-prajwal:~/go/src/k8s.io/arktos# 
```

The Dashboard will be accessible at `https://<host_machine_ip>:30001` and you can log in using `username` & `password` used in `auth.csv`



Discovery and Load Balancing

Services

Name	Namespace	Labels	Cluster IP	Internal Endpoints	External Endpoints	Age
kubernetes-default	default	arktos.futurewei.com/network: default component: apiserver Show all	10.0.0.198	kubernetes-default:443 TCP kubernetes-default:0 TCP	-	5 minutes
kubernetes	default	component: apiserver provider: kubernetes	10.0.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	5 minutes

1 - 2 of 2

Config and Storage

Secrets

Name	Namespace	Labels	Type	Age
default-token-v47j6	default	-	kubernetes.io/service-account-token	5 minutes

1 - 1 of 1