**KUBECTL**

kubectl config set-context --current --namespace=ggckad-s2

kubectl config --kubeconfig=/root/my-kube-config use-context research

kubectl config --kubeconfig=/root/my-kube-config current-context

kubectl run nginx --image=nginx

kubectl run nginx --image=nginx --port=8080

kubectl run nginx --image=nginx --dry-run=client -o yaml

kubectl run --restart=Never --image=busybox static-busybox --dry-run=client -o yaml --command -- sleep 1000

**<DEPLOYMENT>**

kubectl create deployment nginx --image=nginx

kubectl create deployment --image=nginx nginx --dry-run=client -o yaml

kubectl create deployment --image=nginx nginx --dry-run=client -o yaml > nginx-deployment.yaml

kubectl create deployment --image=nginx nginx --replicas=4 --dry-run=client -o yaml > nginx-deployment.yaml

<**SERVICES**>

kubectl expose pod redis --port=6379 --name redis-service --dry-run=client -o yaml

kubectl expose pod nginx --type=NodePort --port=80 --name=nginx-service --dry-run=client -o yaml

(This will automatically use the pod's labels as selectors)

kubectl expose deployment ingress-controller --type=NodePort --port=80 --name=ingress --dry-run=client -o yaml > ingress.yaml

kubectl create service clusterip redis --tcp=6379:6379 --dry-run=client -o yaml

kubectl create service nodeport nginx --tcp=80:80 --node-port=30080 --dry-run=client -o yaml

**OTHERS**

ps -aux | grep kubelet

CLUSTER UPGRADE:

-------------------------

Kubeadm upgrade plan | apt-get upgrade -y kubeadm=1.12.0.-00 | kubeadm upgrade apply v11.12.0

apt-get upgrade -y kubelet=1.12.0.-00 | systemctl restart kubelet | kubectl drain node01 | kubectl cordon node01 | ssh node01 | apt-get upgrade -y kubeadm=1.12.0.-00 | apt-get upgrade -y kubelet=1.12.0.-00 | kubectadm upgrade node config --kubelet-version v1.12.0

systemctl restart kubelet | kubectl uncordon node01

JSONPATH:

---------

kubectl get nodes -o=jsonpath='{range .items[\*]}{.metadata.name}{"\n"}{end}'

kubectl get nodes -o=jsonpath='{.items[\*].status.nodeInfo.osImage}'

kubectl config view --kubeconfig=/root/my-kube-config

kubectl get pv --sort-by=.spec.capacity.storage

kubectl get pv --sort-by=.spec.capacity.storage -o=custom-columns=NAME:.metadata.name,CAPACITY:.spec.capacity.storage

kubectl config view --kubeconfig=my-kube-config -o jsonpath="{.contexts[?(@.context.user=='aws-user')].name}"

**OPENSSL**

**#Generate keys**

openssl genrsa -out ca.key 2048. | openssl genrsa -out admin.key 2048

**#Generate CSR**

openssl req -new -key ca.key -subj "/CN=KUBERNETES-CA" -out ca.csr

openssl req -new -key ca.key -subj "/CN=kube-admin" -out admin.csr

**#Sign certificates**

openssl x509 -req -in ca.csr -signkey ca.key -out ca.crt

openssl x509 -req -in admin.csr -CA ca.crt -CAkey ca.key -out admin.crt

**#To view certificate:**

openssl x509 -in /etc/kubernetes/pki/apiserver.crt -text -noout

**To view certificates:**

Hardway:

cat /etc/systemd/system/kube-apiserver.service

Kubeadm:

cat /etc/kubernetes/manifests/kube-apiserver.yaml

**To view logs:**

Hardway: journalctl -u etcd.service -l && kubeadm: kubectl logs etcd-master

docker ps -a | docker logs container-id

#accessing controlplane and apis

curl http://localhost:6443/apis -k | grep names --> we need to pass certificates to access like this

#instead we can use kubectl proxy command to access kubeapi using certs from kubeconfig (PORT 8001)

kubectl proxy | curl http://localhost:6443/apis -k | grep names

#to view namespaced or non-namespaced resources

kubectl api-resources --namespaced=true | kubectl api-resources --namespaced=false

NETWORKING:

-----------

ip link & ip addr add 192.168.1.10/24 dev eth0 & ip route &

ip route add 192.168.2.0/24 via 192.168.1.1

# transfer packets between interfaces (eth0 and eth1)

cat /proc/sys/net/ipv4/ip\_forward --> 0 (no forward) --> 1(forward)

arp | netstat -nltp | netstat -anp

#DNS Server configuration:

cat /etc/resolve.conf

nameserver 192.168.1.100 --> this is the DNS server IP

ls /opt/cni/bin and ls /etc/cni/net.d/

#weavenet

kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')&env.IPALLOC\_RANGE=10.50.0.0/16"