

set alias for commands below

alias k=kubect1

CLUSTER & EVENTS

k config view	list the current context configuration
k config view -o \jsonpath='{.users[*].name}'	list all users in the cluster
k config view --flatten > ~/.kube/config	compress kube config into one file named 'config'
export KUBECONFIG=config1:config2	use two kube config files simultaneously
k cluster-info	view control plane and etcd information
k get componentstatus	get system health
k api-resources	list available resources
k api-resources --namespaced=true	list namespaced resources
k get --raw \apis/metrics.k8s.io/v1beta1/nodes	Get metrics data on nodes in the k8s cluster (replace nodes with pods to get pod metrics)
k get events -A	list events in the all namespace

k get events -w	watch events in real-time
k get ep kube-scheduler -n kube-system -o yaml	find the elected leader in a highly available cluster (in annotations)
kubeadm version	get the version of kubeadm
kubeadm config print init-defaults	list default values kubeadm uses for cluster init
kubeadm token create --print-join-command	print the join command to add more nodes (sudo)
kubeadm token list	list tokens that haven't expired
kubeadm token generate	generate a new token to authenticate with cluster
kubeadm upgrade plan	plan the upgrade of control plane components
kubeadm upgrade node	upgrade local kubelet
k exec kube-apiserver-pod -n kube-system -- kube-apiserver -h grep enable-admission-plugins	view enabled admission controllers (can also look in /etc/kubernetes/manifests/kube-apiserver.yaml)

NAMESPACE & CONTEXT

k create ns robot-shop	create a namespace named 'robot-shop'
k get ns	list all namespaces in the cluster
k get ns -o yaml	get the yaml config for all namespaces
k get all -A	list all resources in all namespaces (--all-namespaces)
k describe ns	describe the namespace configuration
k edit ns robot-shop	edit the namespace named 'robot-shop'
k delete ns robot-shop	delete the namespace named 'robot-shop'
k config get-contexts	list available contexts from kube config

k config current-context	get the current context for kubect1
k config set-context gkeCluster	switch context to a cluster named 'gkeCluster'
k config set-context --current --namespace webapp	swtich context to the current context in the namespace named 'webapp'
k config set-context gkeCluster --namespace robot-shop	switch context to cluster named 'gkeCluster' and in the 'robot-shop' namespace
k config set-context gkeCluster --user=admin	switch context to cluster 'gkeCluster' as user 'admin'
k config use-context gkeCluster	set the default context to 'gkeCluster'
k config delete-cluster docker-desktop	remove the cluster 'docker-desktop' from kubeconfig

NODES

k get no	list nodes in the default namespace
k get no -o wide	list nodes with IP address & runtime info
k get no -l node-role.kubernetes.io/control-plane	list nodes with the label 'kubernetes.io/control-plane' (k get no --show-labels to show labels)
k describe no	describe all nodes in the cluster
k label no mynode1 disk=ssd	label node 'mynode1' with 'disk=ssd'
k get no --show-labels	shot labels on all nodes in the cluster
k annotate no mynode1 azure=node	add the annotation 'azure=node' to node 'mynode1'
k get nodes -o jsonpath='{items[*].status.addresses[?(@.type=="ExternalIP")].addresses}'	list external IP addresses of all nodes in the default namespace
k get no -o jsonpath='{.items[0].metadata.name}'	list the first node only within the list of nodes in the cluster
k top node mynode1	view cpu and memory metrics for node 'mynode1'

k get no -o json jq '.items[].spec.taints'	view taints on all nodes in the cluster
k get nodes -o jsonpath='{range .items[*]}{.metadata.name}{.spec.taints[?(@.effect=="NoSchedule")].effect}{"\n\n"}}{end}'	view taints on nodes including node names
k taint no mynode1 node-role.kubernetes.io/master:NoSchedule	taint node 'mynode1' to not schedule pods to it
k taint no mynode1 dedicated=special-user:NoSchedule	taint node with key 'dedicated', value 'special-user' and effect 'NoSchedule'
k taint no mynode1 dedicated:NoSchedule-	take away taint which allows pods to be scheduled
k taint no mynode1 dedicated-	take away taint key 'dedicated' from node
k describe no grep Taint	describe nodes and filter out the word 'Taint'
k taint no -l disk=ssd dedicated=mynode1:PreferNoSchedule	taint node with label 'disk=ssd' and key 'dedicated'
k taint no mynode1 bar:NoSchedule	taint node with key 'bar' and no value

set alias for commands below

alias k=kubect1

NODES (cont.)

k drain mynode1 --ignore-daemonsets --force	remove pods from node 'mynode1' ignore daemonsets
k cordon mynode1	set 'mynode1' to unschedulable (cordon)
k uncordon mynode1	set node 'mynode1' to schedulable (uncordon)

k delete no mynode1	delete node 'mynode1' from the cluster
k edit no mynode1	edit the node configuration

PODS

k run nginx --image=nginx	create pod using image 'nginx'
k run busybox --image=busybox -it	create a pod with image 'busybox' and get a shell to it
k run curlpod --image=nicolaka/netshoot --rm -it -- sh	create pod and get a shell to it when exiting the shell will delete the pod
k run dnstools --image infoblox/dnstools --rm -it -- bash	create pod with image 'infoblox/dnstools' and run bash
k run debug-pod --image=busybox -it	describe the namespace configuration
k run nginx --image=nginx --dry-run=client -o yaml > pod.yaml	create the YAML file 'pod.yaml' for a pod with image 'nginx'
k get po	list all pods in the default namespace
k get po -A -w	tail the pods in all namespaces
k wait --for=condition=ready pod -l app=nginx	tail the pods until pod with label 'app=nginx' is ready
k get po --all-namespaces	list pods in all namespaces
k get pods -A -o wide --field-selector spec.nodeName=control-plane	list running pods that reside on the node named 'control-plane'
k get all -A	list all cluster resources in all namespaces
k get po,no,svc -A	list pods, nodes and services in all namespaces
k get po -o wide	list pods with IP addresses and node assignments
k describe po	describe all pods in the default namespace
k label nginx app=prod	add label 'app=prod' on the pod named 'nginx'
k get po --show-labels	show labels on all pods in the default namespace
k get po -l app=nginx	list pods that have the label 'app=nginx'
k exec -it mongodb mongo	run the command 'mongo' on pod named 'mongo'
k exec -it mypod -c cart -- /bin/bash	get a shell to container 'cart' in pod named 'mypod'
k exec -it bux -- sh -c "while true; do echo hello; sleep 2; done"	run command in a running pod named 'bux' to echo 'hello' and sleep for 2 seconds on a loop

k annotate po nginx special=app1	annotate pod with key 'special' and value 'app1'
k get po nginx -o yaml	show the YAML configuration for pod named 'nginx'
k get pod nginx -o yaml --export > podconfig.yaml	export the YAML for a pod to a file named 'podconfig.yaml'
k get po --field-selector status.phase=Running	list pods that are running only
k exec nginx env	list environment variables for pod named 'nginx'
k run curl --image=nicolaka/netshoot --rm -it --restart=Never -- cat /etc/resolv.conf	create a pod and cat out the 'resolv.conf' file and then delete the pod upon exit
k run netshoot --image=nicolaka/netshoot --command sleep --command "3600"	create a pod and run the command 'sleep 3600' inside the container
k exec -t nginx - cat /etc/resolv.conf	cat the 'resolv.conf' file (dns info) on running pod
k logs nginx	get the logs for pod named 'nginx'
k logs -l app=nginx	get the logs for all pods that have label 'app=nginx'
k logs nginx > pod.log	output the logs to file 'pod.log' for pod named 'nginx'
k logs nginx --since=1h	get the logs from pod 'nginx' from the last hour
k logs nginx --tail=20	get the last 20 lines from the logs from pod 'nginx'
k logs -f nginx -c log	stream the logs for container 'named 'log''
k logs nginx -f	stream (follow) the logs from pod 'nginx'
k delete po nginx	delete pod named 'nginx'
k edit po nginx	edit the pod configuration for pod named 'nginx'
k port-forward nginx 8080:80	port forward from 80 on container to 8080 on host
k port-forward elasticsearch-pod 9200:9200 &	port forward from 9200 and run in background (&)
curl --head http://localhost:9200	curl the forwarded port from control plane node

DEPLOYMENTS & REPLICASETS

k create deploy nginx --image nginx	create deployment named 'nginx' using image 'nginx'
k create deploy nginx --image nginx --dry-run=client -o yaml > deploy.yaml	output deployment YAML to a file named 'deploy.yaml'
k create -f deploy.yaml	create deployment from file named 'deploy.yaml'
k create -f deploy.yaml --record	create deployment from file and record the history
k apply -f deploy.yaml	create deployment, even if resource already exists

k replace -f deploy.yaml	create deployment only if resource already exists
k rollout undo deploy nginx	undo deployment rollout named 'nginx'
k rollout undo deploy nginx --to-revision=3	undo deployment rollout to 3rd revision
k rollout pause deploy nginx	pause deployment in the middle of rolling out
k rollout resume deploy nginx	resume a paused deployment rollout
k rollout status deployment/nginx	get the status of a current rollout

set alias for commands below

alias k=kubect1

DEPLOYMENTS & REPLICASETS (cont.)

k rollout restart deployment/nginx	restart pods in a deployment named 'nginx'	k get deploy,po,svc --all-namespaces	list deployments, pods and services in all namespaces
k rollout history deploy nginx	get the rollout history for deployment 'nginx'	k get deploy -o wide	list deployments with image and selector information
k scale deploy nginx --replicas=5	scale a deployment to have a total of 5 pods	k get deploy -o yaml	get the YAML configuration for deployment
k scale deploy nginx --replicas 3 --record	scale a deployment to 3 and record the output	k describe deploy	describe all deployments in the default namespace
k rollout history deploy nginx	get the rollout history for deployment 'nginx'	k delete deploy nginx	delete deployment 'nginx' in default namespace
k set image deployments/nginx nginx:nginx:1.14.2 --v 6	change the image used for deployment 'nginx' to nginx:1.14.2	k get rs	list all replicaset in the default namespace
k edit deploy nginx	edit the deployment configuration for 'nginx'	k get rs -o wide	list replicaset with container image and selector info
k get deploy	list the deployments in the default namespace	k get rs -o yaml	output the YAML manifest for replicaset in default ns
k get deploy --all-namespaces	list deployments in all namespaces in the cluster	k describe rs	describe the replicaset in default namespace
k get all --all-namespaces	list all kubernetes resources in all namespaces		

SERVICES & INGRESS

k create svc nodeport nodeport-svc --tcp=8080:80	create a nodeport type service exposing port 8080 from the container to port 80 on the host	k create ing cool-ing --rule="mycoolwebapp.com/forums=forums-svc:8080,tls=my-cert"	create ingress named 'cool-ing' that directs requests to mycoolwebapp.com/forums to servcie named 'forums-svc' on 8080 with tls secret 'my-cert'
k expose deploy nginx --name=app-service --port=80 --type=NodePort	create service from deployment 'nginx' exposing port 80 and set type to nodeport	k create ing one-ing --rule="/path=myweb-svc:80"	create ingress named 'one-ing' that directs to service 'myweb-svc' on port 80
k expose svc nginx --name nginx-https --port 443 --target-port 8443	clone the service 'nginx', creating a new service named nginx-https exposing port 443 with target 8443	k create ing appgw-ing --rule="azurewebapp.com/shop=web-svc:8080" --annotation kubernetes.io/ingress.class=azure/application-gateway	create ingress 'appgw-ing' that adds an annotation for azure application gateway and directs 'azurewebapp.com/shop' to service 'web-svc' on port 8080
k get svc	list services in the default namespace	k create ing rewire-ing --rule="circuitweb.com/shop=web-svc:8080" --annotation "nginx.ingress.kubernetes.io/rewrite-target= /"	create ingress 'rewire-ing' with annotation to rewrite the path or nginx ingress controllers
k get svc -o wide	list services with selector information	k create ing moo-ing --rule="moo.com/=milk-svc:80" --rule="moo.com/flavors=flavor-svc:8080"	create ingress 'moo-ing' where all requests go to service 'milk-svc' on port 80 but requests 'moo.com/flavors' go to 'flavor-svc' on port 8080
k get svc -o yaml	get YAML output for all services in default namespace		
k describe svc	describe all services in the default namespace		
k get svc --show-labels	view labels applied to all services in default namespace		
k edit svc app-service	edit the service named 'app-service'		
k delete svc app-service	delete servcie named 'app-service'		

ROLES & SERVICE ACCOUNTS

k get roles -n kube-system	list roles in 'kube-system' namespace	k get clusterrolebindings -o json jq -r '.items[] select(.subjects[0].kind=="Group") select(.subjects[0].name=="system:masters")'	list the clusterrolebindings that have a membership in 'system:masters' group
k get roles -n kube-system -o yaml	output YAML for roles in kube-system namespace	k auth can-i get secrets --as chad	see if user 'chad' can list secrets in default namespace
k get clusterroles	list all cluster roles in cluster	k auth can-i delete pods --as chad -n default	see if user 'chad' can delete pods in the default namespace
k create role pod-reader --verb=get --verb=list --verb=watch --resource=pods	create role named 'pod-reader' that can list, get, and watch pods	k get sa	list serviceaccounts in the default namespace
k create clusterrole pod-reader --verb=get,list,watch --resource=pods	create clusterrole that can get, list, and watch pods	k get sa -o yaml	output the YAML for serviceaccounts in default ns
k create rolebinding bob-admin-binding --clusterrole=admin --user=bob --namespace=robot-shop	create rolebinding 'bob-admin-binding' to role 'admin' for user 'bob' in namespace 'robot-shop'	k get sa default -o yaml > sa.yaml	output YAML for the default serviceaccount to sa.yaml
k create clusterrolebinding root-cluster-admin-binding --clusterrole=admin --user=bob	create clusterrolebinding 'root-cluster-admin-binding' to 'admin' clusterrole for user 'bob'	k replace sa default -f sa.yaml	replace 'default' sa with YAML from sa.yaml
		k edit sa default	edit the service account named 'default'
		k delete sa default	delete service account 'default' in default namespace

set alias for commands below

alias k=kubect1

CONFIGMAPS & SECRETS

k get cm	list configmaps in default namespace	k create secret generic vault-tls --from-file=vault.key=key --from-file=vault.crt=vault.example.com.pem --from-file=vault.ca=ca	create a generic secret 'vault-tls' from key file named 'key', client certificate file named 'vault.example.com.pem' and ca file named 'ca'
k get cm --all-namespaces	list configmaps in all namespaces	k get secrets	list secrets in the default namespace
k get cm --all-namespaces -o yaml	get YAML config for all configmaps in all namespaces	k get secrets --all-namespaces	list secrets in all namespaces
k create secret generic db-user-pass --from-file=./username.txt --from-file=./password.txt	create generic secret 'db-user-pass' from file 'username.txt' and 'password.txt'	k get secrets --all-namespaces -o yaml	get YAML for all secrets in all namespaces
k create secret generic db-user-pass --from-file=username=./username.txt --from-file=password=./password.txt	create generic secret from file and give the names of keys as 'username' and 'password'	k get secret db-user-pass -o jsonpath='{.data}'	get data contents out of the secret 'db-user-pass' (in base64 encoded format, decode with base64 -d)
k create secret generic vault-license --from-literal="license=\${secret}"	create generic secret 'vault-license' from an env variable named 'secret'		

DAEMONSETS

k get ds	list all daemonsets in default namespace	k edit ds kube-proxy -n kube-system	edit 'kube-proxy' daemonset in kube-system ns
k get ds --all-namespaces	list all daemonsets in all namespaces	k edit ds kube-proxy -n kube-system	edit 'kube-proxy' daemonset in kube-system ns
k describe ds kube-proxy -n kube-system	describe 'kube-proxy' daemonset in kube-system ns	k delete ds kube-proxy -n kube-system	delete daemonset named 'kube-proxy'
k get ds kube-proxy -n kube-system -o yaml	get YAML output of 'kube-proxy' daemonset		

VOLUMES & STORAGE CLASS

cat <<EOF k apply -f - apiVersion: v1 kind: PersistentVolume metadata: name: pv-volume spec: hostPath: path: "/mnt/data" capacity: storage: 1Gi accessModes: - ReadWriteOnce EOF	create hostpath persistent volume named 'pv-volume' using 1 gigabyte of storage from the host at '/mnt/data' on host	k get pv	list persistent volumes in default namespace
cat <<EOF k apply -f - apiVersion: v1 kind: PersistentVolumeClaim metadata: name: pv-claim spec: accessModes: - ReadWriteOnce resources: requests: storage: 1Gi EOF	create persistent volume claim named 'pv-claim' that requests 1 gigabyte of storage from the first available persistent volume in the cluster	k get pv pv-volume	list only volume named 'pv-volume'
cat <<EOF k apply -f - apiVersion: v1 kind: Pod metadata: name: pv-pod spec: containers: - name: pv-container image: nginx volumeMounts: - mountPath: "/usr/share/nginx/html" name: pv-storage volumes: - name: pv-storage persistentVolumeClaim: claimName: pv-claim EOF	create a pod named 'pv-pod' that uses the persistent volume claim 'pv-claim' and mounts a volume named 'pv-storage' inside of the container at '/usr/share/nginx/html'	k get pv pv-volume -o yaml	show YAML output of persistent volume 'pv-volume'
		k describe pv	describe all persistent volumes in default ns
		k describe pv pv-volume	describe persistent volume 'pv-volume'
		k get pvc	list persistent volume claims in default ns
		k get pvc pv-claim	list persistent volume claim named 'pv-claim'
		k get pvc pv-claim -o yaml	get YAML output of persistent volume claim 'pv-claim'
		k describe pvc	describe all persistent volume claims in default ns
		k describe pvc pv-claim	describe persistent volume claim 'pv-claim'
		k get pv,pvc	list persistent volumes and persistent volume claims
		k delete pv pv-volume	delete persistent volume 'pv-volume'
		k delete pvc pv-claim	delete persistent volume claim 'pv-claim'
		k get sc	list storage classes in the default namespace
		k get sc -o yaml	get YAML output of all storage classes in default ns
		k get volumeattachments	view the volumes attached to nodes (non-namespaced)