kubectl cheatsheet

by **Lino Figueroa** (lino.figueroa@softwareone.com)

Definitions

POD

Minimal application unit. One or several containers.

ConfigMap and Secret

Store configuration. ConfigMap stores in plain text, Secret stores hashed with base64

Namespace

Logical grouping of applications.

ReplicaSet

Stable set of PODs (how many? are they alive? Are they started?)

Deployment

Updates for PODs and Replicasets (deploy new version with minimal downtime, rollback version).

Jobs/ Cronjobs

Components that ensure PODs execution and termination.

Services

Expose PODs to the world (ClusterIP, NodePort, Load Balancer, Ingress).

YAML

Depend on the component you want to create so you have to check the kubernetes documentation.

Introduction to Kubernetes YAML: https://www.mirantis.com/blog/introduction-to-yaml-creating-a-kubernetes-deployment/

Cluster deployment

Deploy components

Deploy a single file:

`kubectl apply -f <filename>`

Deploy all the yaml files in the current path:

`kubectl apply -f .`

Modify a component

Just modify the yaml file with its definition and do a deploy again.

Kubernetes keeps the status of every component in the cluster.

Delete components

Delete components defined in a single file:

`kubectl delete -f <filename>`

Delete the component defined in all the yaml files in the current path:

`kubectl delete -f .`

Delete component by name

`kubectl delete <component_name>`

To get the full component name you have to use `kubectl get all`

Cluster management

You can get information and manipulate any cluster component.

Get namespaces

`kubectl get namespaces`

Use namespaces

"`default`" namespace is used if no one specified in the commands.

The parameter `-n <namespace_name>` allows running commands only over the given namespace and its components.

Get information

`get <component>`

Gets a list of components of type `<component>`. As Kubernetes API is extensible and allow custom components this type can be anything.

The most commons are:

`all` to list all the components

`nodes` list the nodes

`pod` list the pods

`services` list services

`configmaps`, `secrets` list configmaps and secrets

The parameter `-o wide` is used to get a list with extended information.

Inspect/describe components

`kubectl inspect <component_type> <component_name>`

Gets extended information about a given component.

View logs

`kubectl logs -f <pod_name>`

“logs”: recover stdout/stderr

> -f = follow; locks console and recover all messages from the given pod.

Shell

You can access a pod shell using the command “exec”.

`kubectl -it exec <pod_name> sh`

It’s being deprecated

`kubectl -it exec <pod_name> -- sh`

> `-it` means “interactive” to allow the Linux console to interact with the k8s console.

(you can exit the shell using “exit”)