UNDERSTAND THE ENVIRONMENT VARIABLES FOR PODS BEFORE GOING TO SECRETS & CONFIG MAPS.

=====NGINX-WITHOUT-VOLUMES-CONFIGMAPS=================

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: newnginx

name: newnginx

spec:

replicas: 1

selector:

matchLabels:

app: newnginx

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: newnginx

spec:

containers:

- image: nginx:latest

name: nginx

ports:

- containerPort: 80

====================NGINX-WITH-SINGLE-VOLUME-AS-CONFIGMAPS====================

git clone https://github.com/mavrick202/dockertest1.git

cd dockertest1

ku create configmap html1 --from-file=index.html

ku create configmap js1 --from-file=scorekeeper.js

ku create configmap css1 --from-file=style.css

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: newnginx

name: newnginx

spec:

replicas: 1

selector:

matchLabels:

app: newnginx

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: newnginx

spec:

containers:

- image: nginx:latest

name: nginx

ports:

- containerPort: 80

volumeMounts:

- mountPath: /usr/share/nginx/html/

name: nginx-conf

volumes:

- name: nginx-conf

configMap:

name: html1

===================-NGINX-MAP-MULTIPLE-CONFIGMAPS-TO-SINGLE-FOLDER-===========

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: newnginx

name: newnginx

spec:

replicas: 1

selector:

matchLabels:

app: newnginx

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: newnginx

spec:

containers:

- image: nginx:latest

name: nginx

ports:

- containerPort: 80

volumeMounts:

- mountPath: /usr/share/nginx/html/index.html

name: html1

subPath: index.html

- mountPath: /usr/share/nginx/html/scorekeeper.js

name: js1

subPath: index.html

- mountPath: /usr/share/nginx/html/style.css

name: css1

subPath: style.css

volumes:

- name: html1

configMap:

name: html1

- name: js1

configMap:

name: js1

- name: css1

configMap:

name: css1

===============================================================================

ku run hello1 --image=sreeharshav/rollingupdate:v1 --env="Env=Prod" --env="Owner=Sree"

kubectl create configmap testconfig1 --from-literal=sleep-interval=25 --from-literal=owner=sree --from-literal=env=prod

(or)

apiVersion: v1

data:

env: prod

owner: sree

sleep-interval: "25"

kind: ConfigMap

metadata:

name: testconfig2

kubectl create configmap testconfig2 --from-literal=sleep-interval=25 --from-literal=owner=sree --from-literal=env=prod --dry-run -o yaml

ku create configmap test1 --from-file=index.html

ku create configmap test2 --from-file=index.html

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: newnginx

name: newnginx

spec:

replicas: 1

selector:

matchLabels:

app: newnginx

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: newnginx

spec:

containers:

- image: sreeharshav/rollingupdate:v1

name: nginx

ports:

- containerPort: 80

volumeMounts:

- mountPath: /usr/share/nginx/html/index.html

name: html1

subPath: index.html

- mountPath: /usr/share/nginx/html/scorekeeper.js

name: js1

subPath: scorekeeper.js

- mountPath: /usr/share/nginx/html/style.css

name: css1

subPath: style.css

volumes:

- name: html1

configMap:

name: html1

- name: js1

configMap:

name: js1

- name: css1

configMap:

name: css1

=============================================================================

apiVersion: v1

data:

env: prod

owner: sree

sleep-interval: "25"

kind: ConfigMap

metadata:

name: testconfig2

#Assigning selected keys in the ConfigMap

---

apiVersion: v1

kind: Pod

metadata:

name: testpod001

spec:

containers:

- name: testpod001

image: sreeharshav/utils:latest

env:

- name: MUNNA\_BHAI\_MBBS\_1

valueFrom:

configMapKeyRef:

name: testconfig2

key: env

- name: MUNNA\_BHAI\_MBBS\_2

valueFrom:

configMapKeyRef:

name: testconfig2

key: owner

#Assigning all the config info

apiVersion: v1

kind: Pod

metadata:

name: testpod

spec:

containers:

- name: test-container

image: sreeharshav/rollingupdate:v3

envFrom:

- configMapRef:

name: testconfig2

restartPolicy: Never

apiVersion: v1

kind: Pod

metadata:

name: testpod

spec:

containers:

- name: test-container

image: sreeharshav/rollingupdate:v3

envFrom:

- configMapRef:

name: testconfig2

key: env

restartPolicy: Never

ku delete cm $(ku get cm -o json | jq -r '.items[].metadata.name')

apiVersion: v1

data:

env: prod

owner: sree

sleep-interval: "25"

kind: ConfigMap

metadata:

name: testconfig2

#Following will show the config in single line in the env variables.

apiVersion: v1

data:

nginx.conf: |

logdest /tmp

log\_type all

environment production

kind: ConfigMap

metadata:

name: testconfig2

#Following will show the config in multi line in the env variables.

apiVersion: v1

data:

nginx.conf: >

logdest /tmp

log\_type all

environment production

kind: ConfigMap

metadata:

name: testconfig3

===========SECRET-IN-FILE======================

apiVersion: v1

data:

password.file: |

SW5kaWFAMTIzNDU2Cg==

kind: Secret

type: Opaque

metadata:

name: secret1

=====MOUNT-SECRET-AS-VOLUME====

apiVersion: v1

kind: Pod

metadata:

name: mypod

spec:

containers:

- name: mypod

image: redis

volumeMounts:

- name: foo

mountPath: "/etc/foo"

readOnly: true

volumes:

- name: foo

secret:

secretName: mysql-secrets

=====MOUNT-SECRET-AS-ENV-VARIABLE============

<https://www.serverlab.ca/tutorials/containers/kubernetes/how-to-deploy-mysql-server-5-7-to-kubernetes/>

<https://opensource.com/article/19/6/introduction-kubernetes-secrets-and-configmaps>

==================================================

echo -n "my-super-secret-password" | base64

echo 'MWYyZDFlMmU2N2Rm' | base64 --decode

apiVersion: v1

kind: Secret

metadata:

name: mysql-secrets

type: Opaque

data:

ROOT\_PASSWORD: c3VwZXItc2VjcmV0LXBhc3N3b3JkCg==

apiVersion: v1

kind: Secret

metadata:

name: secret1

type: Opaque

data:

ROOT\_PASSWORD: c3VwZXItc2VjcmV0LXBhc3N3b3JkCg==

Secrets run only in the memory of the Nodes where the PODs are running. They are not saved in any folder or disk storage and get wiped out when the secret is created.