Config maps and secrets in Kube

Config Maps:

a Kubernetes API object used to store non-sensitive configuration data in key-value pairs
ConfigMaps enable you to manage configuration settings independently of the container images, allowing you to update configurations without redeploying the container images. store data as key-value pairs, where the keys are strings and the values can be strings or Base64-encoded binary data.

Vi configmap.yml

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: testcm
data:
  db-port: "3306"
```

Kubectl apply -f configmap.yml

Vi Deploy.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp
spec:
  replicas: 2
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
```

```
    spec:
      containers:
        - name: webapp
          image: awsdevops183/birthday
          env:
            - name: db-port
              valueFrom:
                configMapKeyRef:
                  name: testcm
                  key: db-port
```

If we change port number it still shows same port and conflict happens. Instead we use Volume Mounts.

Note : Container doesn't allow changing the environment variable. We need to recreate the container. So Other way is Volume Mounts.

Using Volume mounts instead of env variables we use it as files.

Vi deployment.yml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webapp
spec:
  replicas: 2
  selector:
    matchLabels:
      app: webapp
  template:
    metadata:
      labels:
        app: webapp
    spec:
      containers:
        - name: webapp
          image: awsdevops183/birthday
          volumeMounts:
```

```
        - name: db-connection
          mountPath: /opt
    volumes:
      - name: db-connection
        configMap:
          name: testcm
```

We can change port number in configmap.yml. —> apply

Enter pod : kubectl exec -it webapp-8d88c67d7-48ffj --/bin/bash
cd /opt
cat db-port

SECRET:

kubectl create secret generic testsecret --from-literal=dbport="3306"
kubectl describe secret testsecret
Kubectl edit secret test secret

This will encrypt base-64 data like this.
data:
  dbport: MzMwNg==

To check weather it is 3306 secret or not:

echo MzMwNg== | base64 —decode | more

This will give a random password. To keep our own secrets:

ENCRYPT THE DATA IN ETCD AT REST :

ETCD: It is a key value store will contains all cluster info including configurations, secrets and states.

Step 1:

```
cat <<EOF > encryption-config.yaml
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
      - secrets
    providers:
      - aescbc:
          keys:
            - name: key1
              secret: $(head -c 32 /dev/urandom | base64)
      - identity: {}
EOF
```

Step 2:
Update the API Server Configuration

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

```
spec:
  containers:
    - name: kube-apiserver
      args:
        - --encryption-provider-config=/etc/kubernetes/encryption-
config.yaml
```

Step3: Mount the Encryption Config File: Make sure the encryption config file is mounted into the API server pod

```
volumeMounts:
  - name: encryption-config
    mountPath: /etc/kubernetes/encryption-config.yaml
    readOnly: true

volumes:
  - name: encryption-config
    hostPath:
      path: /etc/kubernetes/encryption-config.yaml
```

type: File

Step 4:
Restart the API Server

Re-encrypt Existing Secrets

kubectl get secrets --all-namespaces -o json | kubectl replace -f -