Kubernetes ConfigMap and Secret explained

ConfigMap

What is ConfigMap and when is it used?
Think of it as a *properties file for your application*. For example, depending on your application environment (dev, int, prod) you will have a different database URL or logging level. So for this kind of thing you can use configMap.

The ***biggest advantage*** is that, with properties file, every time you modify it you have to rebuild and redeploy your application, whereas if you change the configuration in configMap, ***you just need to restart the application pod/container***.

ConfigMap can be used by the application as a set of environmental variable values or as an actual configuration file.

Example ConfigMap with database connection configuration:

```
apiVersion: v1
kind: ConfigMap
metadata:
    name: my-config
data:
  db-host: cluster-mysql.database
  db-port: 3306
  db-name: my-db
```

The values in this configMap can be used in the following way in your app's pod specification:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  containers:
  - name: my-app
    image: my-app-image
    env:
    - name: DB_HOST
      valueFrom:
        configMapKeyRef:
          name: my-config
          key: db-host
    - name: DB_PORT
      valueFrom:
        configMapKeyRef:
          name: my-config
          key: db-port
    - name me: DB_NAME
      valueFrom:
        configMapKeyRef:
          name: my-config
          key: db-name
```

Here is an example ConfigMap which creates a configuration file for the Mosquitto app:

```
apiVersion: v1
kind: ConfigMap
metadata:
    name: mosquitto-config
data:
  mosquitto.conf |
     log_dest stout
     log_type all
     log_timestamp true
     listener 9001
```

In this case, we need to mount the ConfigMap as a volume in Kubernetes:

```
apiVersion: v1
kind: Pod
metadata:
  name: mosquitto
spec:
  containers:
  - name: mosquitto
    image: mosquitto-image
    volumeMounts:
      - name: config-file
        mountPath: /mosquitto/config
  volumes:
  - name: config-file
    configMap:
      name: mosquitto-config
```

This config map will produce a file `mosquitto.conf`, which then can be mounted into the Mosquitto container under `/mosquitto/config` dire ctory.

Secret

Secrets are also used in these 2 ways. Either as a value for env variables or as a secret file with credentials or a certificate etc mounted into a pod.

So for a better comparison, think of secrets as encrypted configMaps.

Example secret with key-value pairs:

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
  type: Opaque
data:
  db-user: dXNlcg==
  db-password: cGFzc3dvcmQ
```

And you can use it the same way as ConfigMap in your application's configuration file:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  containers:
  - name: my-app
    image: my-app-image
    env:
    - name: DB_USER
      valueFrom:
        secretKeyRef:
          name: my-secret
          key: db-user
    - name: DB_PASSWORD
      valueFrom:
        secretKeyRef:
          name: my-secret
          key: db-password
```