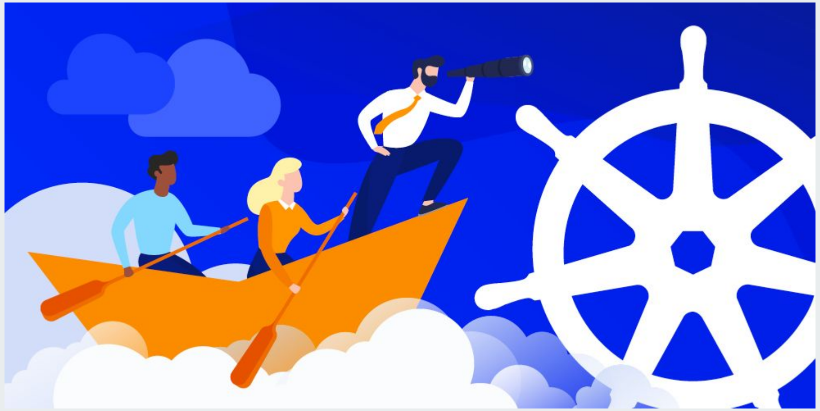
TEL4VN

Configmap and Secret



Content

Configmap

Key value pairs exposed into a Pod used application configuration settings

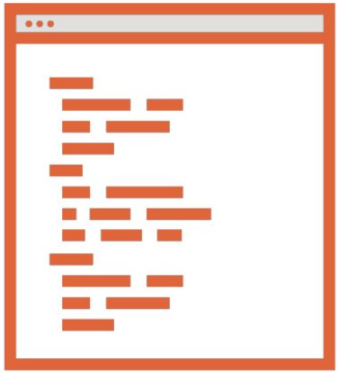
Defining application or environment specific settings

Decouple application and Pod configurations

Maximizing our container image’s portability

Environment Variables or Files

Using Configmap in Pod



Environment variables

`valueFrom` **and** `envFrom`

Volumes and Files

Volume mounted inside a container

Single file or directory

Many files or directories

Volume ConfigMaps can be updated

Define Configmap

```
kubectl create configmap appconfigprod \
 --from-literal=DATABASE_SERVERNAME=sql.example.local \
 --from-literal=BACKEND_SERVERNAME=be.example.local

kubectl create configmap appconfigqa \
 --from-file=appconfigqa

apiVersion: v1
kind: ConfigMap
metadata:
 name: appconfigprod
data:
  BACKEND_SERVERNAME: be.example.local
  DATABASE_SERVERNAME: sql.example.local
```

Using Configmap in Environment Variable

```
containers:
- name: hello-world
  ...
  env:
  - name: DATABASE_SERVERNAME
    valueFrom:
      configMapKeyRef:
        name: appconfigprod
        key: DATABASE_SERVERNAME
  - name: BACKEND_SERVERNAME
    valueFrom:
      configMapKeyRef:
        name: appconfigprod
        key: BACKEND_SERVERNAME
```

```
containers:
- name: hello-world
  ...
  envFrom:
    - configMapRef:
        name: appconfigprod
```

Using Configmap as File

```
spec:
  volumes:
    - name: appconfig
      configMap:
        name: appconfigqa
  containers:
  - name: hello-world
    ...
    volumeMounts:
      - name: appconfig
        mountPath: "/etc/appconfig"
```

Demo 1

- ❑ Create configmap
- ❑ Using configmap in Pod

Secrets

Store sensitive information as Objects

Retrieve for later use

Passwords, API tokens, keys and certificates

Safer and more flexible configurations (Pod Specs and Images)

Property of Secret

base64 encoded

Encryption can be configured

Stored in etcd

Namespaced and can only be referenced by Pods in the same Namespace

Unavailable Secrets will prevent a Pods from starting up

Creating Secrets

```
kubectl create secret generic app1 \
 --from-literal=USERNAME=app1login \
 --from-literal=PASSWORD='S0methingS@Str0ng!'
```

Using Secret in Pod

Environment Variables

Volumes or Files

Referenced Secret must be created and accessible for the Pod to start up

Using Secrets in Environment Variable

```
spec:
  containers:
  - name: hello-world
    ...
    env:
    - name: app1username
      valueFrom:
        secretKeyRef:
          name: app1
          key: USERNAME
    - name: app1password
      valueFrom:
        secretKeyRef:
          name: app1
          key: PASSWORD
```

```
spec:
  containers:
  - name: hello-world
    ...
    envFrom:
    - secretRef:
        name: app1
```

Using Secrets as File

```
spec:
  volumes:
    - name: appconfig
      secret:                                 /etc/appconfig/USERNAME
        secretName: app1                      /etc/appconfig/PASSWORD
  containers:
    ...
    volumeMounts:
      - name: appconfig
        mountPath: "/etc/appconfig"
```

Demo 2

- ❑ Create and access secrets
- ❑ Accessing secret inside a Pod
  - ❑ Environment variable
  - ❑ File

Accessing Private Container Registry

- Secrets for application configuration
- Use Secrets to access a private container registry
- Want to access registries over the Internet
  - Docker Hub
  - Cloud based container registries
- Create a Secret of type `docker-registry`
- Enabling Kubernetes (kubelet) to pull the images from the private registry

Demo 3

- Pulling a image from private registry

Configuration in Pod

Command Line Arguments

Environment Variables

ConfigMaps

Environment Variable

User defined

- Pod Spec for each container
- Defined inside the container image
- Defined in `name/value` or `valueFrom`

System defined

- Names of all Services available at the time the Pod was created

Defined at container startup

Cannot be updated once the Pod is created

Define Environment Variables

```
spec:
  containers:
  - name: hello-world
    image: gcr.io/google-samples/hello-app:1.0
    env:
    - name: DATABASE_SERVERNAME
      value: "sql.example.local"
    - name: BACKEND_SERVERNAME
      value: “be.example.local"
```

Demo 4

- ❏ Passing environment variable

Q
A