Day 4: é…•ç½®ä¸Žå¯†é’¥ç®¡ç•† (ConfigMap, Se

ðŸŽ¯ å-¦ä¹ ç›®æ ‡

- **æŠ€èƒ½ç›®æ ‡**: æŽŒæ•¡åœ¨ Kubernetes ä¸-ç®¡ç•†åº”ç”¨é…•ç½®å’Œæ••æ„Ÿæ•°æ•®çš„æ ¸å¿ƒæ–¹æ³• `ConfigMap` å’Œ `Secret`ã€‚
- **æ ¸å¿ƒæ¦‚å¿µ**: æ·±å…¥ç•†è§£å°†é…•ç½®ä¸Žåº”ç”¨é•œåƒè§£è€¦çš„é‡•è¦•æ€§ï¼Œä»¥å•Šä¸¤ç§•ä¸»è¦•çš„é…•ç½®æ³¨å…¥æ–¹å¼•ï¼šçŽ¯å¢ƒå•˜é‡•å’Œå•·æŒ‚è½½ã€‚
- **å…·ä½“æˆ•æžœ**:èƒ½å¤Ÿç‹¬ç«‹åœ°ä»Žæ–‡ä»¶æˆ–å-—é•¢å€¼åˆ›å»º `ConfigMap` å’Œ `Secret`ã€‚èƒ½å¤Ÿæˆ•åŠŸåœ°å°† `ConfigMap` çš„æ•°æ•®ä½œä¸ºçŽ¯å¢ƒå•˜é‡•æ³¨å…¥åˆ°å®¹å™¨ä¸-ã€‚èƒ½å¤Ÿæˆ•åŠŸåœ°å°† `ConfigMap` å’Œ `Secret` ä½œä¸ºæ–‡ä»¶æŒ‚è½½åˆ° Pod çš„æ–‡ä»¶ç³»ç»Ÿä¸-ã€‚èƒ½å¤Ÿè§£é‡Š `ConfigMap` å’Œ `Secret` çš„ä¸»è¦•åŒºåˆ«å’Œé€‚ç”¨åœºæ™¯ã€‚

ðŸ“š ç•†è®ºåŸºç¡€ (30%)

1. ä¸ºä»€ä¹ˆéœ€è¦•è§£è€¦é…•ç½®ï¼Ÿ

åœ¨è½¯ä»¶å¼€å•‘ä¸-ï¼Œä¸€ä¸ªæœ€ä½³å®žè·µæ˜¯å°†**ä»£ç •**å’Œ**é…•ç½®**åˆ†ç¦»ã€‚å¦‚æžœå°†æ•°æ•®åº“åœ°å•€ã€•API å¯†é’¥ã€•åŠŸèƒ½å¼€å…³ç-‰é…•ç½®ä¿¡æ•¯ç¡¬ç¼–ç •åœ¨åº”ç”¨é•œåƒä¸-ï¼Œä¼šå¸¦æ¥å¾ˆå¤šé—®é¢˜ï¼š

- **ç•µæ´»æ€§å·®**: æ¯•æ¬¡ä¿®æ”¹é…•ç½®éƒ½éœ€è¦•é‡•æ–°æž„å»ºå’Œå•‘å¸ƒé•œåƒã€‚
- **å¤•ç”¨æ€§ä½Ž**: å•Œä¸€ä¸ªåº”ç”¨é•œåƒæ— æ³•ç›´æŽ¥ç”¨äºŽå¼€å•‘ã€•æµ‹è¯•ã€•ç”Ÿäº§ç-‰ä¸•å•ŒçŽ¯å¢ƒï¼Œå› ä¸ºå®ƒä»¬çš„é…•ç½®ä¸•å•Œã€‚
- **å®‰å…¨æ€§é£Žé™©**: å°†æ••æ„Ÿä¿¡æ•¯ï¼ˆå¦‚å¯†ç •ï¼‰æ‰“åŒ…åˆ°é•œåƒä¸-ï¼Œä¼šå¢žï¿½ï¿½ï¿½æ³„éœ²çš„é£Žé™©ã€‚

Kubernetes æ••ä¾›äº†ä¸¤ç§•æ ¸å¿ƒèµ„æº•æ•¥è§£å†³è¿™ä¸ªé—®é¢˜ï¼š`ConfigMap` ç”¨äºŽé•žæ••æ„Ÿé…•ç½®ï¼Œ`Secret` ç”¨äºŽæ••æ„Ÿé…•ç½®ã€‚

2. ConfigMap: ç®¡ç•†æ™®é€šé…•ç½®

- **å®šä¹‰**: `ConfigMap` æ˜¯ä¸€ä¸ªç”¨äºŽå-˜å‚¨é”®å€¼å¯¹å½¢å¼•çš„ã€•é•žæ••æ„Ÿé…•ç½®æ•°æ•®çš„ API å¯¹è±¡ã€‚å®ƒå•¯ä»¥å-˜å‚¨å••ä¸ªé…•ç½®é¡¹ï¼Œä¹Ÿå•¯ä»¥å-˜å‚¨å®Œæ•´çš„é…•ç½®æ–‡ä»¶å†…å®¹ã€‚
- **æ•°æ•®æ•¥æº•**:**å-—é•¢å€¼ (Literal)**: ç›´æŽ¥åœ¨å‘½ä»¤è¡Œæˆ– YAML ä¸-å®šä¹‰é”®å€¼å¯¹ã€‚**æ–‡ä»¶ (File)**: å°†ä¸€ä¸ªæˆ–å¤šä¸ªæ–‡ä»¶çš„å†…å®¹ä½œä¸º `ConfigMap` çš„æ•°æ•®ã€‚æ–‡ä»¶å••ä¼šæˆ•ä¸ºé”® (key)ï¼Œæ–‡ä»¶å†…å®¹ä¼šæˆ•ä¸ºå€¼ (value)ã€‚
- **å¤§å°•é™•åˆ¶**: `ConfigMap`

çš„è®¾è®¡ç›®æ ‡æ˜¯å˜å‚¨å°‘é‡•é…•ç½®æ•°æ•®ï¼Œå…¶æ€»å¤§å°•é€šå¸¸è¢«é™•
åˆ¶åœ¨ 1MiB ä»¥å†…ã€‚ä¸•é€‚å•ˆå˜å‚¨å¤§åž‹æ–‡ä»¶ã€‚

3. Secret: ç®¡ç•†æ••æ„Ÿæ•°æ•®

- **å®šä¹‰**: `Secret`
æ˜¯ä¸€ä¸ªä¸“é—¨ç”¨äºŽå˜å‚¨æ••æ„Ÿæ•°æ•®ï¼ˆå¦‚å¯†ç •ã€•OAuth ä»¤ç‰Œã€•SSH
å¯†é’¥ï¼‰çš„ API å¯¹è±¡ã€‚
- **ä¸Ž ConfigMap çš„åŒºåˆ«**:**è‡ªåŠ¨ç¼–ç •**: `Secret` ä¸çš„æ•°æ•®åœ¨å˜å‚¨åˆ°
`etcd` ä¹‹å‰•ï¼Œä¼šé»˜è®¤è¿›è¡Œ **Base64
ç¼–ç •**ã€‚**æ³¨æ„•ï¼šè¿™å•ªæ˜¯ç¼–ç •ï¼Œä¸•æ˜¯åŠ å¯†ï¼•**
ä»»ä½•æœ‰æ•ƒé™•è®¿é—® `etcd` æˆ– API
çš„äººéƒ½å•¯ä»¥è½»æ•¾è§£ç •ã€‚å®ƒçš„ä¸»è¦•ç›®çš„æ˜¯é˜²æ¢æ•°æ•®ä»¥æ˜Žæ–‡
å½¢å¼•ç›´æŽ¥æš´éœ²åœ¨ YAML æ–‡ä»¶æˆ– API å“•åº”ä¸ã€‚**é¢•å¤–ä¿•æŠ¤**:
Kubernetes ä¼šå¯¹ `Secret`
æ••ä¾›ä¸€äº›é¢•å¤–çš„ä¿•æŠ¤ï¿½ï¿½æ–½ï¼Œä¾‹å¦‚é»˜è®¤æƒ…å†µä¸‹ä¸•å°†
`Secret` æŒ‚è½½åˆ°ä¸´æ—¶å®¹å™¨
(`tmpfs`)ï¼Œä»¥å•Šåœ¨æœªæ•¥çš„ç‰ˆæœ¬ä¸å•¯èƒ½æ••ä¾›é•™æ€•åŠ å¯†
(`Encryption at Rest`)ã€‚**ç‰¹å®šç±»åž‹**: `Secret`
æ”¯æŒ•å¤šç§•ç±»åž‹ï¼Œç”¨äºŽæ»¡è¶³ç‰¹å®šåœºæ™¯ï¼Œä¾‹å¦‚
`kubernetes.io/dockerconfigjson`
ç”¨äºŽå˜å‚¨ç§•æœ‰é•œåƒ•ä»“åº“çš„è®¤è¯•ä¿¡æ•¯ã€‚
- **æ ¸å¿ƒåŽŸå ™**: æ°¸è¿œä¸•è¦•å°† `Secret` çš„ YAML
æ–‡ä»¶æ••äº¤åˆ°å…¬å…±çš„ä»£ç •ä»“åº“ä¸ã€‚

4. æ³¨å…¥æ–¹å¼•ï¼šå¦‚ä½•è®© Pod ä½¿ç”¨å®ƒä»¬ï¼Ÿ

å°† `ConfigMap` æˆ– `Secret`
çš„æ•°æ•®æ••ä¾›ç»™å®¹å™¨ä¸»è¦•æœ‰ä¸¤ç§•æ–¹å¼•ï¼š

a. ä½œä¸ºçŽ¯å¢ƒå˜é‡• (Environment Variables)

- **ä¼˜ç‚¹**:
ç®€å••ç›´æŽ¥ï¼Œå¤§å¤šæ•°åº”ç”¨éƒ½æ”¯æŒ•é€šè¿‡çŽ¯å¢ƒå˜é‡•è¯»å–é…•ç½®ã
€‚
- **ç¼ºç‚¹**: å¦‚æžœæ³¨å…¥çš„çŽ¯å¢ƒå˜é‡•è¿‡å¤šï¼Œ`kubectl describe pod`
çš„è¾“å‡ºä¼šå˜å¾—é•žå¸¸å†—é•¿ã€‚æ›´é‡•è¦•çš„æ˜¯ï¼Œ**å½“ `ConfigMap` æˆ–
`Secret` æ›´æ–°å•Žï¼Œå·²ç»•è¿•è¡Œçš„ Pod
ä¸çš„çŽ¯å¢ƒå˜é‡•ä¸•ä¼šè‡ªåŠ¨æ›´æ–°**ï¼Œå¿…é¡»é‡•å•¯ Pod
æ‰•èƒ½åŠ è½½æ–°å€¼ã€‚
- **æ³¨å…¥æ–¹æ³•**:`env`: é€•ä¸ªåœ°å°† `ConfigMap` æˆ– `Secret`
ä¸çš„æŸ•ä¸ªé”®æ³¨å…¥ä¸ºæŒ‡å®šçš„çŽ¯å¢ƒå˜é‡•ã€‚`envFrom`: å°† `ConfigMap`
æˆ– `Secret`
ä¸çš„æ‰€æœ‰é”®å€¼å¯¹ä¸€æ¬¡æ€§å…¨éƒ¨æ³¨å…¥ä¸ºçŽ¯å¢ƒå˜é‡•ã€‚

b. ä½œä¸ºå•·æŒ‚è½½ (Volume Mount)

- **ä¼˜ç‚¹**:**è‡ªåŠ¨æ›´æ–°**: è¿™æ˜¯æœ€å…³é”®çš„ä¼˜åŠ¿ã€‚å½“æŒ‚è½½çš„
`ConfigMap` æˆ– `Secret` æ›´æ–°å•Žï¼ŒPod
ä¸è¢«æŒ‚è½½çš„æ–‡ä»¶å†…å®¹**ä¼šè‡ªåŠ¨åœ°ã€•è¿‘ä¹Žå®žæ—¶åœ°æ›´æ–°**ï

¼Œæ— éœ€é‡•å•¯
Podã€‚è¿™å¯¹äºŽéœ€è¦•åŠ¨æ€•é‡•è½½é…•ç½®çš„åº”ç”¨é•žå¸¸æœ‰ç”¨ã€‚é€‚å•ˆå-˜å
‚¨å®Œæ•´çš„é…•ç½®æ–‡ä»¶ï¼ˆå¦‚ `nginx.conf`, `application.properties`ï¼‰ã€‚
- **ç¼ºç‚¹**:
åº”ç”¨éœ€è¦•æ”¹é€ ä»¥æ”¯æŒ•ä»Žæ–‡ä»¶ç³»ç»Ÿè¯»å•–é…•ç½®ï¼Œå¹¶åœ¨æ–‡ä»¶
å•˜æ›´æ—¶è‡ªåŠ¨é‡•è½½ã€‚
- **æ³¨å…¥æ–¹æ³•**: åœ¨ Pod çš„ `spec.volumes` ä¸å®šä¹‰ä¸€ä¸ª `configMap` æˆ–
`secret` ç±»åž‹çš„å•·ï¼Œç„¶å•Žåœ¨ `spec.containers.volumeMounts`
ä¸å°†å…¶æŒ‚è½½åˆ°å®¹å™¨çš„æŒ‡å®šè·¯å¾„ã€‚

ðŸ› ï¸• å®žè·µæ“•ä½œ (50%)

1. åˆ›å»º ConfigMap

a. ä»Žå——é•¢å€¼åˆ›å»º

```
kubectl create configmap app-config --from-literal=app.color=blue
--from-literal=app.environment=development
```

b. ä»Žæ–‡ä»¶åˆ›å»º
å…ˆåˆ›å»ºä¸€äº›é…•ç½®æ–‡ä»¶ï¼š

```
echo "user.name=guest" > user.properties
echo "database.url=jdbc:mysql://localhost:3306/mydb" > db.properties
```

åˆ›å»º ConfigMap:

```
kubectl create configmap db-config --from-file=db.properties
--from-file=user.properties
```

c. æŸ¥çœ‹ ConfigMap

```
kubectl get configmap db-config -o yaml
# apiVersion: v1
# data:
#   db.properties: |
#     database.url=jdbc:mysql://localhost:3306/mydb
#   user.properties: |
#     user.name=guest
# kind: ConfigMap
# ...
```

2. å°† ConfigMap æ³¨å…¥ä¸ºçŽ¯å¢ƒå˜é‡•

åˆ›å»ºä¸€ä¸ªæ–‡ä»¶ `pod-env-demo.yaml`:

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-env-demo
spec:
  containers:
  - name: test-container
    image: busybox
    command: [ "/bin/sh", "-c", "env && sleep 3600" ]
    env: # é€•ä¸ªæ³¨å…¥
      - name: APP_COLOR
        valueFrom:
          configMapKeyRef:
            name: app-config # ConfigMap å••ç§°
            key: app.color   # Key å••ç§°
    envFrom: # æ‰¹é‡•æ³¨å…¥
      - configMapRef:
          name: db-config # ConfigMap å••ç§°
  restartPolicy: Never
```

éƒ¨ç½²å¹¶æŸ¥çœ‹æ—¥å¿—ï¼š

```
kubectl apply -f pod-env-demo.yaml
kubectl logs pod-env-demo
# ...
# APP_COLOR=blue
# db.properties=database.url=jdbc:mysql://localhost:3306/mydb
# user.properties=user.name=guest
# ...
```

3. å°† ConfigMap æŒ‚è½½ä¸ºå•·

åˆ›å»ºä¸€ä¸ªæ–‡ä»¶ `pod-volume-demo.yaml`:

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-volume-demo
spec:
  containers:
  - name: test-container
    image: busybox
    command: [ "/bin/sh", "-c", "ls -l /etc/config && sleep 3600" ]
    volumeMounts:
    - name: config-volume # å¯¹åº”ä¸‹é•¢çš„ volume å••ç§°
      mountPath: /etc/config # æŒ‚è½½åˆ°å®¹å™¨çš„è·¯å¾„
  volumes:
  - name: config-volume
    configMap:
      name: db-config # ä½¿ç”¨å“ªä¸ª ConfigMap
```

éƒ¨ç½²å¹¶éªŒè¯ï¼š

```
kubectl apply -f pod-volume-demo.yaml
kubectl logs pod-volume-demo
# total 8
# lrwxrwxrwx ... db.properties -> ..data/db.properties
# lrwxrwxrwx ... user.properties -> ..data/user.properties

# è¿›å…¥ Pod æŸ¥çœ‹æ–‡ä»¶å†…å®¹
kubectl exec -it pod-volume-demo -- cat /etc/config/db.properties
# database.url=jdbc:mysql://localhost:3306/mydb
```

4. åˆ›å»ºå’Œä½¿ç”¨ Secret

a. åˆ›å»º Secret

```
# Base64 ç¼–ç •æ˜¯è‡ªåŠ¨å®Œæˆ•çš„
kubectl create secret generic db-secret --from-literal=username=admin
--from-literal=password='S3cr3tP@ssw0rd'
```

b. æŸ¥çœ‹ Secret

```
# ç›´æŽ¥ get ä¸•ä¼šæ˜¾ç¤ºæ•°æ•®
kubectl get secret db-secret

# ä½¿ç”¨ -o yaml æŸ¥çœ‹ï¼Œæ•°æ•®æ˜¯ Base64 ç¼–ç •çš„
kubectl get secret db-secret -o yaml
# data:
#   password: UzNjcjN0UA==c3cwcmQ=
#   username: YWRtaW4=

# è§£ç •éªŒè¯•
echo 'UzNjcjN0UA==c3cwcmQ=' | base64 --decode
# S3cr3tP@ssw0rd
```

c. æ³¨å…¥ Secret
æ³¨å…¥ `Secret` çš„æ–¹å¼•ä¸Ž `ConfigMap` **å®Œå…¨ç›¸å•Œ**ï¼Œå•ªéœ€å°† `configMapKeyRef` æ›¿æ•¢ä¸º `secretKeyRef`ï¼Œå°† `configMapRef` æ›¿æ•¢ä¸º `secretRef`ï¼Œå°† `volumes` ä¸-çš„ `configMap` æ›¿æ•¢ä¸º `secret` å•³å•¯ã€‚

ðŸ’» Go ç¼–ç¨‹å®žçŽ° (20%)

é¡¹ç›®: k8s-config-creator

ç›®æ ‡: ç¼–å†™ä¸€ä¸ª Go ç¨‹åº•ï¼Œä»¥ç¼–ç¨‹æ–¹å¼•åˆ›å»ºä¸€ä¸ª `ConfigMap`ã€‚

```
package main

import (
 "context"
 "fmt"
 "log"
 "os"
 "path/filepath"

 corev1 "k8s.io/api/core/v1"
 metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
 "k8s.io/client-go/kubernetes"
 "k8s.io/client-go/tools/clientcmd"
)

func main() {
 // --- é…•ç½®å’Œåˆ›å»º clientset ---
 userHomeDir, _ := os.UserHomeDir()
 kubeconfig := filepath.Join(userHomeDir, ".kube", "config")
 config, _ := clientcmd.BuildConfigFromFlags("", kubeconfig)
 clientset, _ := kubernetes.NewForConfig(config)

 namespace := "default"
 cmName := "go-created-cm"

 fmt.Printf("åœ¨å‘½å••ç©ºé—´ '%s' ä¸-åˆ›å»º ConfigMap '%s'...
", namespace, cmName)

 // å®šä¹‰ ConfigMap å¯¹è±¡
 configMap := &corev1.ConfigMap{
  ObjectMeta: metav1.ObjectMeta{
   Name:      cmName,
   Namespace: namespace,
  },
  Data: map[string]string{
   "message": "Hello from Go client!",
   "author":  "Gemini",
  },
 }

 // ä½¿ç”¨ clientset åˆ›å»º ConfigMap
 createdCM, err :=
clientset.CoreV1().ConfigMaps(namespace).Create(context.TODO(), configMap,
metav1.CreateOptions{})
 if err != nil {
  log.Fatalf("åˆ›å»º ConfigMap å¤±è´¥: %v", err)
 }

 fmt.Printf("ConfigMap '%s' åˆ›å»ºæˆ•åŠŸ!
", createdCM.Name)
 fmt.Printf("Data: %v
", createdCM.Data)

 // æ¸…ç•†
 fmt.Println("æŒ‰å›žè½¦é”®åˆ é™¤åˆ›å»ºçš„ ConfigMap...")
 fmt.Scanln()
 clientset.CoreV1().ConfigMaps(namespace).Delete(context.TODO(), cmName,
metav1.DeleteOptions{})
 fmt.Println("æ¸…ç•†å®Œæˆã€‚")
}
```

ðŸ”• æ•…éšœæŽ’æŸ¥ä¸Žä¼˜åŒ–

- **Pod çŠ¶æ€•ä¸º `CreateContainerConfigError`**:`kubectl describe pod <pod-name>` æŸ¥çœ‹äº‹ä»¶ã€‚å¸¸è§•åŽŸå› ï¼šå¼•ç”¨çš„ `ConfigMap` æˆ– `Secret` ä¸•å˜åœ¨ï¼Œæˆ–è€…å¼•ç”¨çš„ `key` åœ¨ `ConfigMap`/`Secret` ä¸ä¸•å˜åœ¨ã€‚
- **è‡ªåŠ¨æ›´æ–°ä¸•ç”Ÿæ•ˆ**:å•ªæœ‰é€šè¿‡**å•·æŒ‚è½½**çš„æ–¹å¼•æ³¨å…¥ï¼Œæ–‡ä»¶å†…å®¹æ‰•ä¼šè‡ªåŠ¨æ›´æ–°ã€‚çŽ¯å¢ƒå•˜é‡•æ–¹å¼•ä¸•ä¼šæ›´æ–°ã€‚æŸ•äº›åº”ç”¨ï¼ˆå¦‚ Java ç¨‹åº•ï¼‰å•¯åŠ¨æ—¶ä¼šå°†é…•ç½®åŠ è½½åˆ°å†…å˜ä¸ï¼Œå³ä½¿æ–‡ä»¶æ›´æ–°äº†ï¼Œåº”ç”¨è‡ªèº«ä¹Ÿéœ€è¦•æœ‰çƒé‡•è½½æœºåˆ¶æ‰•èƒ½ç”Ÿæ•ˆã€‚

ðŸ’• è¿›é˜¶ä½œä¸š

1. **æ¯”è¾ƒæ³¨å…¥æ–¹å¼•**: æ€»ç»“ä¸€ä¸‹ä½¿ç”¨çŽ¯å¢ƒå•˜é‡•å’Œå•·æŒ‚è½½æ³¨å…¥é…•ç½®çš„ä¼˜ç¼ºç‚¹ï¼Œå¹¶åˆ†æžå®ƒä»¬æœ€é€‚åˆçš„åº”ç”¨åœºæ™¯ã€‚
2. **Secret ç±»åž‹**: é˜…è¯»å®˜æ–¹æ–‡æ¡£ï¼Œç ”ç©¶ `Secret` çš„å…¶ä»–ç±»åž‹ï¼Œç‰¹åˆ«æ˜¯ `kubernetes.io/service-account-token` å’Œ `kubernetes.io/tls`ï¼Œäº†è§£å®ƒä»¬çš„ç”¨é€”ã€‚
3. **å•·æŒ‚è½½ç‰¹å®šè·¯å¾„**: å®žè·µä¸€ä¸‹å¦‚ä½•å°† `ConfigMap` ä¸çš„æŸ•ä¸ªç‰¹å®š `key` æŒ‚è½½ä¸ºå•·ä¸çš„ä¸€ä¸ªæ–‡ä»¶å••ï¼Œè€Œä¸•æ˜¯å°†æ‰€æœ‰ `key` éƒ½ä½œä¸ºæ–‡ä»¶å••ã€‚ï¼ˆæ••ç¤º: `volumes.configMap.items`ï¼‰