Secrets in Kubernetes

Robert Rozas Navarro
Customer Engineer
Azure-App Dev Domain

Agenda

1. Overview
2. Q&A
3. Labs

Kubernetes Secrets Overview

Secrets Overview

Secrets Overview(..Continued..)

There are many times when a Kubernetes Pod needs to use sensitive data. Think for examples of:

- SSH keys.
- Database passwords.
- OAuth tokens.
- API keys.
- Image registry keys.

Creating Kubernetes Secrets Objects

There are several ways to create Secrets in Kubernetes. Your choice highly depends on the type of scenario you're in.

- Using The Command Line and Text Files.
- Using The Command Line and Literal Input.
- Using Definition Files.

Creating Kubernetes Secrets Objects(..Continued..)

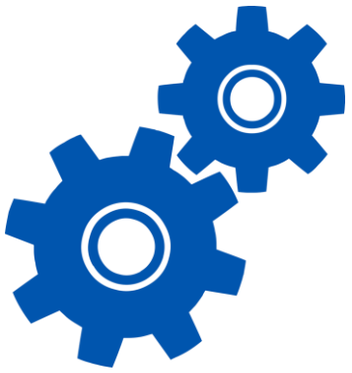
Using The Command Line and Text Files.

```
echo -n 'superuser' > ./username.txt
echo -n '1f2d1e2e67df' > ./password.txt
kubectl create secret generic app-user-cred --from-file=./username.txt --from-file=./password.txt
```

Creating Kubernetes Secrets Objects(..Continued..)

Using The Command Line and Literal Input.

```
kubectl create secret generic app-user-cred --from-literal=username=devuser --from-literal=password=
1f2d1e2e67df
```



Creating Kubernetes Secrets Objects(..Continued..)

Using Definition Files.

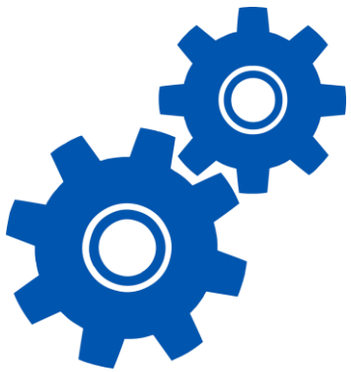
```
$ echo -n 'superuser' | base64
c3VwZXJ1c2Vy
$ echo -n 'Q%FvqS$*F$k^6i' | base64
USVGdnFTJCpGJGteNmk=
```

Now that we have our credentials encoded in base64, let's create the definition file mysecret.yaml:

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  superuser: YWRtaW4=
  password: USVGdnFTJCpGJGteNmk=
```

Apply the file as usual using kubectl as follows:

```
kubectl apply -f mysecret.yaml
```

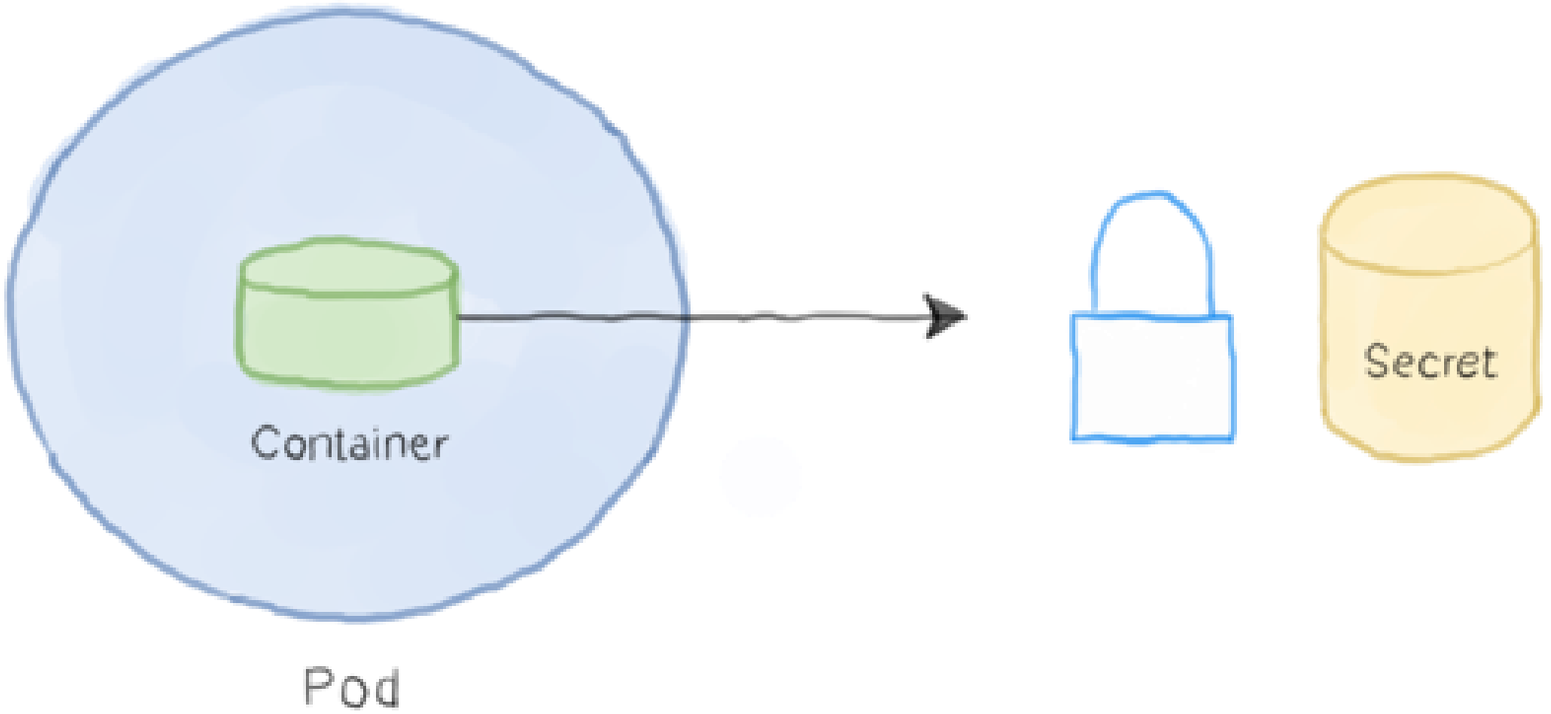


Creating Kubernetes Secrets Objects(..Continued..)

Using Definition Files.

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: configsecret
type: Opaque
stringData:
  configjson: '{"DataBase":{"user":"EPillaoSecret","password":"1234","server":"localhost","
```

Getting The Secret Data Back



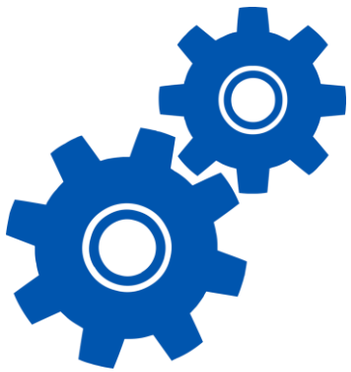
Getting The Secret Data Back (..Continued..)

You can get a Secret from Kubernetes the same way you get other objects using the get subcommand.

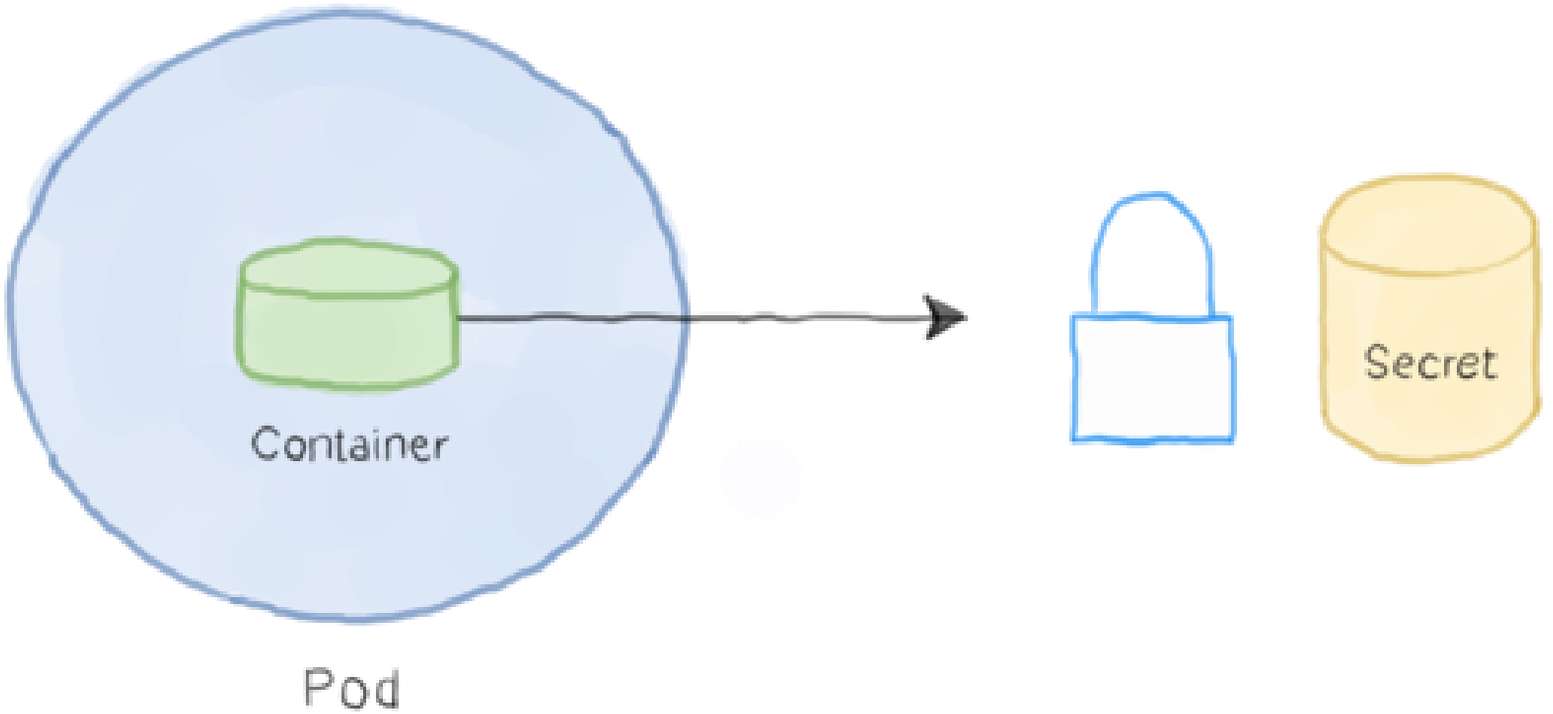
```
$ kubectl get secret mysecret -o yaml
apiVersion: v1
data:
  password: USVGdnFTJCpGJGteNmk=
  superuser: c3VwZXJ1c2Vy
kind: Secret
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
    {"apiVersion":"v1","kind":"Secret","metadata":{"annotations":{},"name":"mysecret","
  creationTimestamp: "2019-07-21T14:14:56Z"
  name: mysecret
  namespace: default
  resourceVersion: "708718"
  selfLink: /api/v1/namespaces/default/secrets/mysecret
  uid: eaddc88c-abc1-11e9-8bff-025000000001
type: Opaque
```

Accessing Secrets Through Environment Variables

```yaml
---
apiVersion: v1
kind: Pod
metadata:
 name: mysqlclient
spec:
 containers:
 - name: mysql
   image: mysql
   env:
   - name: USER
     valueFrom:
       secretKeyRef:
         name: db-creds
         key: user
   - name: PASSWORD
     valueFrom:
       secretKeyRef:
         name: db-creds
         key: password
   - name: HOST
     valueFrom:
       secretKeyRef:
         name: db-creds
         key: host
   command: ["/bin/sh"]
   args: ["-c","mysql -u $USER -p$PASSWORD -h $HOST"]
```



TLDR;



Q&A?