SECRETS

About Secrets:-

secrets and configmap objects are basically used in kubernetes for pushing env file into pods. major difference b/w of these two is, in configmap data will be visible but in secrets data wont not visible, it become encrypted format, so all sensitive data's will be configured in secrets. pods will be separate, whenever required we can include secrets or configmap objects into pod. Secrets are secure objects which store sensitive data, such as passwords, OAuth tokens, and SSH keys, in your clusters. Storing sensitive data in Secrets is more secure than plaintext ConfigMaps or in Pod specifications. Using Secrets gives you control over how sensitive data is used and reduces the risk of exposing the data to unauthorized users. The Secret values are base-64 encoded in Kubernetes and three type are there (generic, docker-registry, TLS).

Actual Readme:-

You create a Secret using the following command:
kubectl create secret [TYPE] [NAME] [DATA]
[TYPE] can be one of the following:
a) generic: Create a Secret from a local file, directory, or literal value.
b) docker-registry: Creates a dockercfg Secret for use with a Docker registry. Used to authenticate against Docker registries.
c) tls: Create a TLS secret from the given public/private key pair. The public/private key pair must exist beforehand. The public key certificate must be .PEM encoded and match the given private key.

[DATA] can be one of the following:
a) a path to a directory containing one or more configuration files, indicated using the --from-file or --from-env-file flags
b) key-value pairs, each specified using --from-literal flags

From files
To create a Secret from one or more files, use --from-file or --from-env-file. The file must be plaintext, but the extension of the file does not matter.
--from-file

When you create the Secret using --from-file, the value of the Secret is the entire contents of the file. If the value of your Secret contains multiple key-value pairs, use --from-env-file instead.

You can pass in a single file or multiple files:
kubectl create secret [TYPE] [NAME] --from-file [/PATH/TO/FILE] --from-file [/PATH/TO/FILE2]

You can also pass in a directory containing multiple files:
kubectl create secret [TYPE] [NAME] --from-file [/PATH/TO/DIRECTORY]
For example, the following command creates a Secret called credentials from two files, username.txt and password.txt, and sets the keys to username.txt and password.txt respectively:
eg: kubectl create secret generic credentials --from-file=username=username.txt --from-file=password=password.txt

From env file

To load multiple key-value pairs into a single Secret, store the key-value pairs in one or more plaintext files and load them using --from-env-file instead of --from-file. You can load multiple files by specifying the flag multiple times. The same limitations as --from-file apply. For example, the following command creates a Secret called credentials from a single file, credentials.txt, which contains multiple key-value pairs:

# Each of these key-value pairs is loaded into the Secret
username=jane
password=d7xnNss7EGCFZusG

eg: kubectl create secret generic credentials --from-env-file ./credentials.txt

From literal values

To create a Secret from literal values, use --from-literal. For example, the following command creates a generic Secret named literal-token with two key-value pairs:

eg: kubectl create secret generic literal-token --from-literal user=admin --from-literal password=1234

You specify --from-literal for each key-value pair.

GENERIC

Create a Secret from a local file, directory, or literal value.

@@ Method1: from the file:-

kubectl get secrets

```
[root@ansikube manifest]# kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      38m
[root@ansikube manifest]#
```

@ Create below two files.

echo -n "user1" >username.txt && echo -n "ebs123" >password.txt

```
[root@ansikube manifest]# ls -lrt
total 8
-rw-r--r--. 1 root root 6 Nov  3 02:50 username.txt
-rw-r--r--. 1 root root 7 Nov  3 02:50 password.txt
[root@ansikube manifest]#
```

kubectl create secret generic user-pass --from-file=username=./username.txt --from-file=password=./password.txt

kubectl get secrets

```
[root@ansikube manifest]# kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      42m
user-pass             Opaque                                2      8s
```

kubectl describe secrets user-pass

```
[root@ansikube manifest]# kubectl describe secrets user-pass
Name:         user-pass
Namespace:    default
Labels:       <none>
Annotations:  <none>

Type:  Opaque

Data
====
password:  7 bytes
username:  6 bytes
[root@ansikube manifest]#
```

Note:- you cannot see the password information on above output, since its encrypted in secrets.

@@ Creating a pod with user-pass secrets.

#cat pod1.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: podsecrets1
spec:
  containers:
    - name: con1
      image: mysql:5.6
      ports:
        - containerPort: 3300
      env:
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: user-pass
              key: password
```

kubectl get pods

```
[root@ansikube manifest]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
podsecrets1   1/1     Running   1          18s
[root@ansikube manifest]#
```

kubectl describe pod podsecrets1

```
Events:
  Type    Reason     Age   From                                          Message
  ----    ------     ----  ----                                          -------
  Normal  Scheduled  86s   default-scheduler                             Successfully assigned default/podsecrets1 to gke-robo-default-pool-df26d11d-
svhh
  Normal  Pulled     85s   kubelet, gke-robo-default-pool-df26d11d-svhh  Container image "mysql:5.6" already present on machine
  Normal  Created    85s   kubelet, gke-robo-default-pool-df26d11d-svhh  Created container
  Normal  Started    85s   kubelet, gke-robo-default-pool-df26d11d-svhh  Started container
```

kubectl exec -it podsecrets1 env

```
[root@ansikube manifest]# kubectl exec -it podsecrets1 env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=podsecrets1
TERM=xterm
MYSQL_ROOT_PASSWORD=ebs123
KUBERNETES_SERVICE_PORT_HTTPS=443
KUBERNETES_PORT=tcp://10.36.0.1:443
KUBERNETES_PORT_443_TCP=tcp://10.36.0.1:443
KUBERNETES_PORT_443_TCP_PROTO=tcp
KUBERNETES_PORT_443_TCP_PORT=443
KUBERNETES_PORT_443_TCP_ADDR=10.36.0.1
KUBERNETES_SERVICE_HOST=10.36.0.1
KUBERNETES_SERVICE_PORT=443
GOSU_VERSION=1.7
MYSQL_MAJOR=5.6
MYSQL_VERSION=5.6.46-1debian9
HOME=/root
[root@ansikube manifest]#
```

Note:- by using secrets objects, password env has been added while creating pod.

@Testing on Pod.

kubectl exec -it podsecrets1 /bin/bash

mysql -u root -p

```
[root@ansikube manifest]# kubectl exec -it podsecrets1 /bin/bash
root@podsecrets1:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.46 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

@@Create secret with literal method.

kubectl create secret generic user-pass1 --from-literal=user1=admin --from-literal=pass1=redhat@123

kubectl get secret

```
[root@ansikube manifest]# kubectl get secret
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      3h18m
user-pass             Opaque                                2      15m
user-pass1            Opaque                                2      8s
[root@ansikube manifest]#
```

kubectl describe secrets user-pass1

@Creating a pod with user-pass1 secrets.

cat pod2.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: podsecrets2
spec:
  containers:
    - name: con2
      image: mysql:5.6
      ports:
        - containerPort: 3306
      env:
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: user-pass1
              key: pass1
```

kubectl apply -f pod2.yml

kubectl get pods

```
[root@ansikube manifest]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
podsecrets1   1/1     Running   0          18m
podsecrets2   1/1     Running   0          19s
[root@ansikube manifest]#
```

kubectl exec -it podsecrets2 env

@For testing purpose, you can login to pod and execute sql command (mysql -u root -p)

#kubectl exec -it podsecrets2 /bin/bash

```
[root@ansikube manifest]# kubectl exec -it podsecrets2 /bin/bash
root@podsecrets2:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.46 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

@@Create secret with literal-token method.

#kubectl create secret generic literal-token --from-literal user=admin --from-literal password=1234

kubectl get secret

```
[root@ansikube manifest]# kubectl get secret
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      4h58m
dockerauth            kubernetes.io/dockerconfigjson        1      81m
literal-token         Opaque                                2      13s
```

kubectl describe secret literal-token

@By using yml file.

cat sec1.yml

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
  namespace: default
type: Opaque
data:
  service_account_key: eyBoZWxsbzogInNkZmFzZCIsCnBhc3N3b3JkOiAid2hhdCIgfQo=
  slack_token: c2ZmcnQ2NHQ3dWs=
  webhook_token: c2RmZGdlcnd3NGRoZ3NmNjQz
[root@ansikube manifest]# kubectl apply -f sec1.yml
secret/my-secret created
```

kubectl get secret

```
[root@ansikube manifest]# kubectl get secret
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      5h5m
dockerauth            kubernetes.io/dockerconfigjson        1      88m
literal-token         Opaque                                2      7m21s
my-secret             Opaque                                3      63s
```

#kubectl describe secret my-secret

@Create secret with env file

cat env

```
username=deepu
password=hello!@#$
```

kubectl create secret generic sec3 --from-env-file=env

kubectl get secrets

```
[root@ansikube manifest]# kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      5h10m
dockerauth            kubernetes.io/dockerconfigjson        1      92m
literal-token         Opaque                                2      11m
my-secret             Opaque                                3      5m29s
sec3                  Opaque                                2      25s
```

kubectl describe secret sec3

Note: - while creating pod we can add above secrets (literal-token, env) method.

DOCKER-REGISTRY

Creates a docker conf Secret for using with a Docker registry. Used to authenticate against Docker registries.

Make sure of docker service running and able to login with your credentials.

docker login

```
[root@ansikube manifest]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: deepu1986
Password:
Login Succeeded
[root@ansikube manifest]#
```

#kubectl create secret docker-registry dockerauth --docker-username=<your-username> --docker-password=<your-password> --docker-server=https://registry-1.docker.io/v2/

Note:- on above "--docker-username=|-docker-password=", you need to use your docker credentials. basically, for private repository we need to have subscription, since free version doesn't have much facilities.

kubectl get secrets

```
[root@ansikube manifest]# kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      3h38m
dockerauth            kubernetes.io/dockerconfigjson        1      73s
user-pass             Opaque                                2      35m
user-pass1            Opaque                                2      20m
[root@ansikube manifest]#
```

kubectl describe secret dockerauth

@@ Create a pod with docker auth secrets.

cat pod3.yml

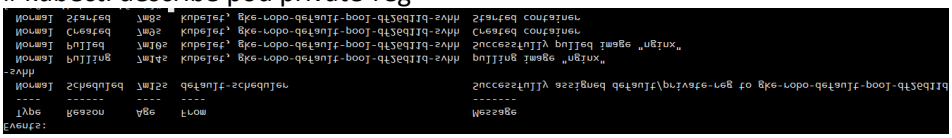
```
apiVersion: v1
kind: Pod
metadata:
  name: private-reg
spec:
  containers:
  - name: con3
    image: nginx
  imagePullSecrets:
  - name: dockerauth
```

kubectl apply -f pod3.yml

kubectl get pods

```
[root@ansikube manifest]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
podsecrets1   1/1     Running   0          49m
podsecrets2   1/1     Running   0          31m
private-reg   1/1     Running   0          107s
```

kubectl describe pod private-reg



Note:- docker registry will store the secret on docker hub.

TLS

Create a TLS secret from the given public/private key pair. The public/private key pair must exist beforehand. The public key certificate must be .PEM encoded and match the given private key.

Example below:-

openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=www.google.com"

```
[root@ansikube manifest]# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=www.google.com"
Generating a 2048 bit RSA private key
.........................+++
.............................................+++
writing new private key to 'tls.key'
-----
[root@ansikube manifest]#
```

kubectl create secret tls test-tls --key="tls.key" --cert="tls.crt"

kubectl get secret

```
[root@ansikube manifest]# kubectl get secret
NAME                  TYPE                                  DATA   AGE
default-token-2fs6w   kubernetes.io/service-account-token   3      4h5m
dockerauth            kubernetes.io/dockerconfigjson        1      28m
test-tls              kubernetes.io/tls                     2      17s
```

@Create tls secret via yml file.

kubectl get secret test-tls -o yaml

Note:- Above command will show you the tls output in yml format, incase if required to create tls method via manifest yml, then we can take reference from existing one.

Take a reference on below link and get the secret created, since TLS yml is having huge content and cannot paste it here.

https://shocksolution.com/2018/12/14/creating-kubernetes-secrets-using-tls-ssl-as-an-example/