VOLUMES AND DATA

Volumes and Data

Secrets

Pods can access local data using volumes, but there is some data you don't want readable to the naked eye. Passwords may be an example. Using the Secret API resource, the same password could be encoded or encrypted.

You can create, get, or delete secrets (see the following commands):

```
$ kubectl get secrets
```

Secrets can be encoded manually or via **`kubectl create secret`**:

```
$ kubectl create secret generic --help
```

```
$ kubectl create secret generic mysql --from-literal=password=root
```

A secret is not encrypted, only base64-encoded, by default. You must create an **`EncryptionConfiguration`** with a key and proper identity. Then, the kube-apiserver needs the **`--encryption-provider-config`** flag set to a previously configured provider, such as **`aescbc`** or **`ksm`**. Once this is enabled, you need to recreate every secret, as they are encrypted upon write.

Multiple keys are possible. Each key for a provider is tried during decryption. The first key of the first provider is used for encryption. To rotate keys, first create a new key, restart (all) kube-apiserver processes, then recreate every secret.

You can see the encoded string inside the secret with **kubectl**. The secret will be decoded and be presented as a string saved to a file. The file can be used as an environmental variable or in a new directory, similar to the presentation of a volume.

A secret can be made manually as well, then inserted into a YAML file (see commands and outputs below):

```
$ echo LFTr@1n | base64
```

```
TEZUckAxbgo=
```

```
$ vim secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: lf-secret
data:
  password: TEZUckAxbgo=
```