**Secrets**

Kubernetes secrets by example

You don’t want sensitive information such as a database password or an API key kept around in clear text. Secrets provide you with a mechanism to use such information in a safe and reliable way with the following properties:

* Secrets are namespaced objects, that is, exist in the context of a namespace
* You can access them via a volume or an environment variable from a container running in a pod
* The secret data on nodes is stored in [tmpfs](https://www.kernel.org/doc/Documentation/filesystems/tmpfs.txt) volumes
* A per-secret size limit of 1MB exists
* The API server stores secrets as plaintext in etcd

Let’s create a secret apikey that holds a (made-up) API key:

$ **echo** -n "A19fh68B001j" > ./apikey.txt

$ kubectl create secret generic apikey --from-file=./apikey.txt

secret "apikey" created

$ kubectl describe secrets/apikey

Name: apikey

Namespace: default

Labels: <none>

Annotations: <none>

Type: Opaque

Data

====

apikey.txt: 12 bytes

Now let’s use the secret in a [pod](https://github.com/openshift-evangelists/kbe/blob/master/specs/secrets/pod.yaml) via a volume:

$ kubectl apply -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/secrets/pod.yaml

If we now exec into the container we see the secret mounted at /tmp/apikey:

$ kubectl exec -it consumesec -c shell -- bash

[root@consumesec /]# mount | grep apikey

tmpfs on /tmp/apikey type tmpfs (ro,relatime)

[root@consumesec /]# cat /tmp/apikey/apikey.txt

A19fh68B001j

Note that for service accounts Kubernetes automatically creates secrets containing credentials for accessing the API and modifies your pods to use this type of secret.

You can remove both the pod and the secret with:

$ kubectl delete pod/consumesec secret/apikey