

ThoughtWorks®

CONFIGMAPS & SECRETS

Rise of the Containers Workshop



*Are we really going to hardcode
configuration in our app?*

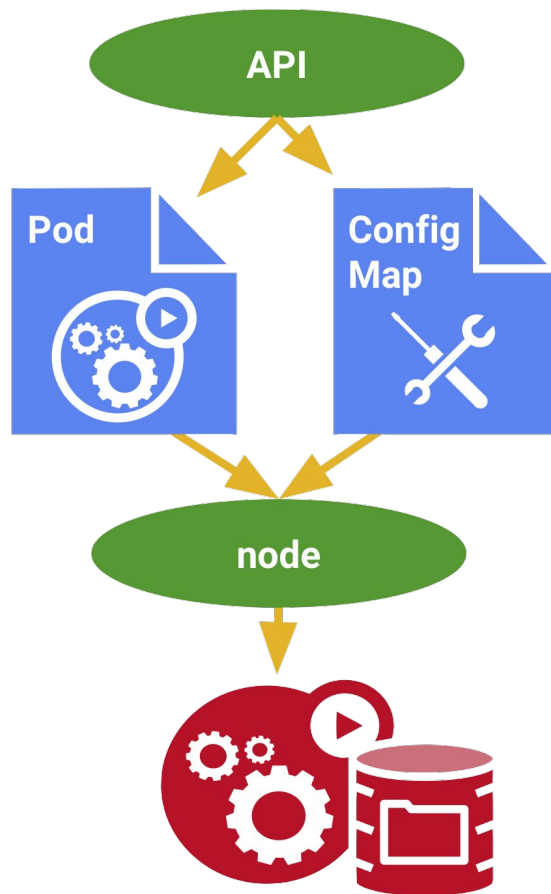
What are our options?

Config Maps

- Represents the configuration of your application
- Stores key-value pairs
- Can be created from directories, files and literals
- Can be injected as environment variables
- Can be mounted as volumes

```
kind: ConfigMap
```

Config Maps



Creating config maps

Config Name

Config Source Type

```
$ kubectl create configmap special-config --from-literal=name=JamesBond
```

```
$ kubectl create configmap special-config --from-file=name=./name.txt
```

Config Value

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: special-config
data:
  name: JamesBond
```

*Let's create a
config map*

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: special-config
data:
  name: JamesBond
```

*Spawn a pod that
uses the config*

```
kind: Pod
apiVersion: v1
metadata:
  name: nameapp
  labels:
    type: nameapp
spec:
  containers:
    - name: nameapp
      image: suvish/helloenv:1.0
      ports:
        - containerPort: 3000
      env:
        - name: NAME
          valueFrom:
            configMapKeyRef:
              name: special-config
              key: name
```

Expose the pod

```
$ kubectl expose pod nameapp \  
--port=3000 \  
--type=NodePort
```


Example:

Mount the config

```
kind: Pod
apiVersion: v1
metadata:
  name: nameapp
  labels:
    type: nameapp
spec:
  volumes:
    - name: config-volume
      configMap:
        name: special-config
  containers:
    - name: nameapp
      image: suvish/helloenv:1.0
      ports:
        - containerPort: 3000
      volumeMounts:
        - name: config-volume
          mountPath: /etc/config
```

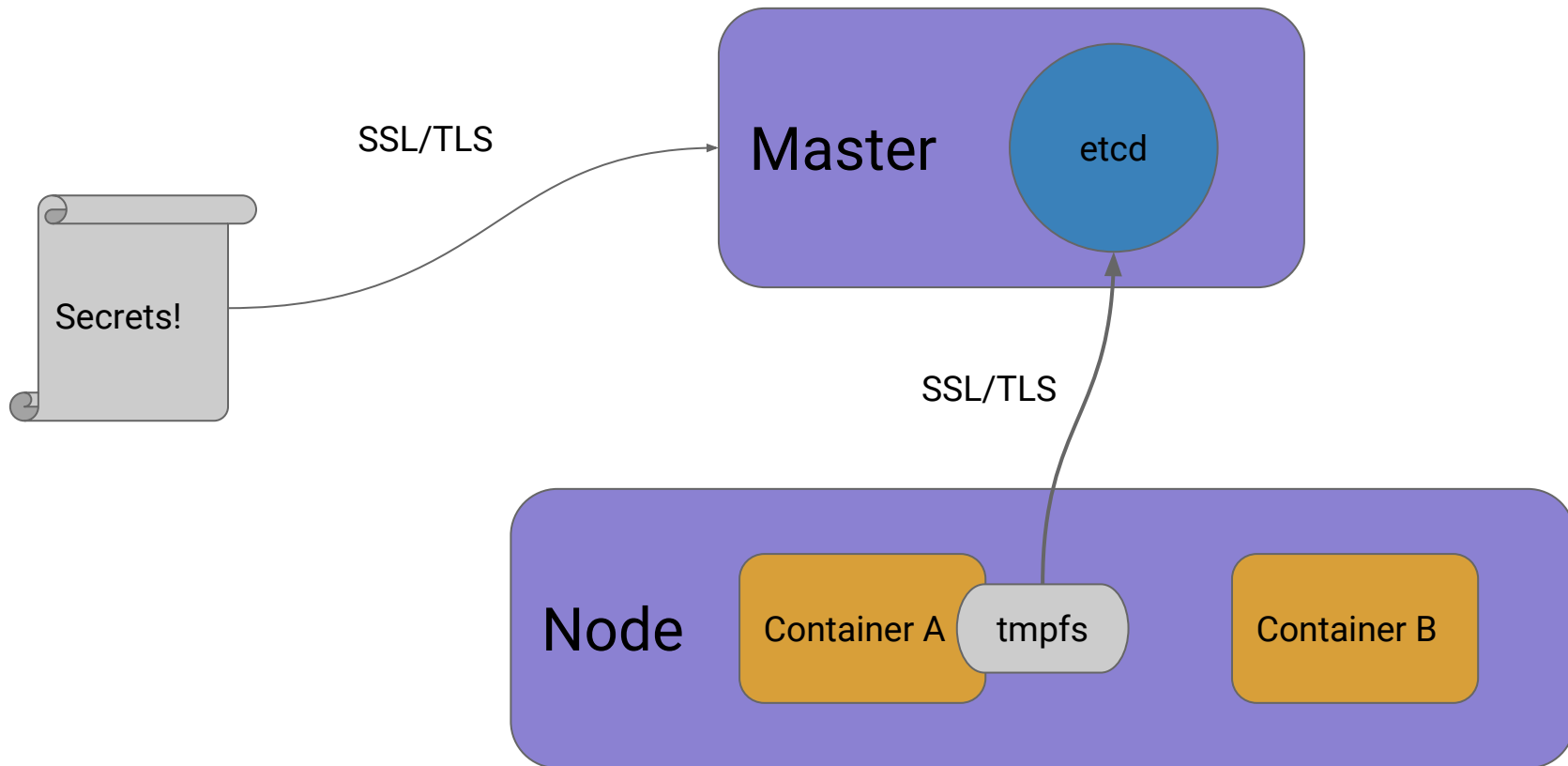
*Is it OK to use passwords and keys in
config maps? If not, how then?*

K8s Secrets

- Store private and sensitive information
- Uses encryption at rest (v1.7 onwards)
- (Very) Similar to config map
- Can be mounted as volume or as env variables

```
kind: Secret
```

How Secrets Work



Creating secrets

Secret Name

Secret Source Type

Base64 encoded

```
$ kubectl create secret generic mysecret --from-literal=password=Ym9uZAo=
```

```
$ kubectl create secret generic mysecret --from-file=password=./password.txt
```

Secret Value

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
Type: Opaque
data:
  password: Ym9uZAo=
```

*Access **Secrets** from a Pod*

```
kind: Pod
apiVersion: v1
metadata:
  name: nameapp
  labels:
    type: nameapp
spec:
  containers:
    - name: nameapp
      image: suvish/helloenv:1.0
      ports:
        - containerPort: 3000
      env:
        - name: NAME
          valueFrom:
            secretKeyRef:
              name: mysecret
              key: name
```

Exercise time!

Use a secret mongo URL in metadata service

P.S. Configure metadata service to use env variable as follows:
`spring.data.mongodb.uri=${MONGO_URL}`

ThoughtWorks®