ConfigMaps vs Secrets

- Creating a ConfigMap

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: example-config
data:
  key1: value1
  key2: value2
```

- Creating a Secret

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
data:
  password: cGFzc3dvcmQ=
  apiKey: YXBpa2V5
```

Key differences

- **Purpose and Usage:**
  - **Secrets:** Designed specifically to store sensitive data such as passwords, OAuth tokens, and SSH keys.
  - **ConfigMaps:** Used to store non-sensitive configuration data, such as configuration files, environment variables, or command-line arguments.
- **Base64 Encoding:**

s is base64 encoded. This is not
naking it slightly obfuscated. This

encoding allows the data to be safely transmitted as part of JSON or YAML files.

- **ConfigMaps:** Data in ConfigMaps is stored as plain text without any encoding.

- **Volatility and Updates:**
  - **Secrets:** Often, the data in Secrets needs to be rotated or updated more frequently due to its sensitive nature.
  - **ConfigMaps:** Configuration data typically changes less frequently compared to sensitive data.
- **Kubernetes Features:**
  - **Secrets:** Kubernetes provides integration with external secret management systems and supports encryption at rest for Secrets when configured properly. Ref https://secrets-store-csi-driver.sigs.k8s.io/concepts.html#provider-for-the-secrets-store-csi-driver
  - **ConfigMaps:** While ConfigMaps are used to inject configuration data into pods, they do not have the same level of support for external management and encryption.