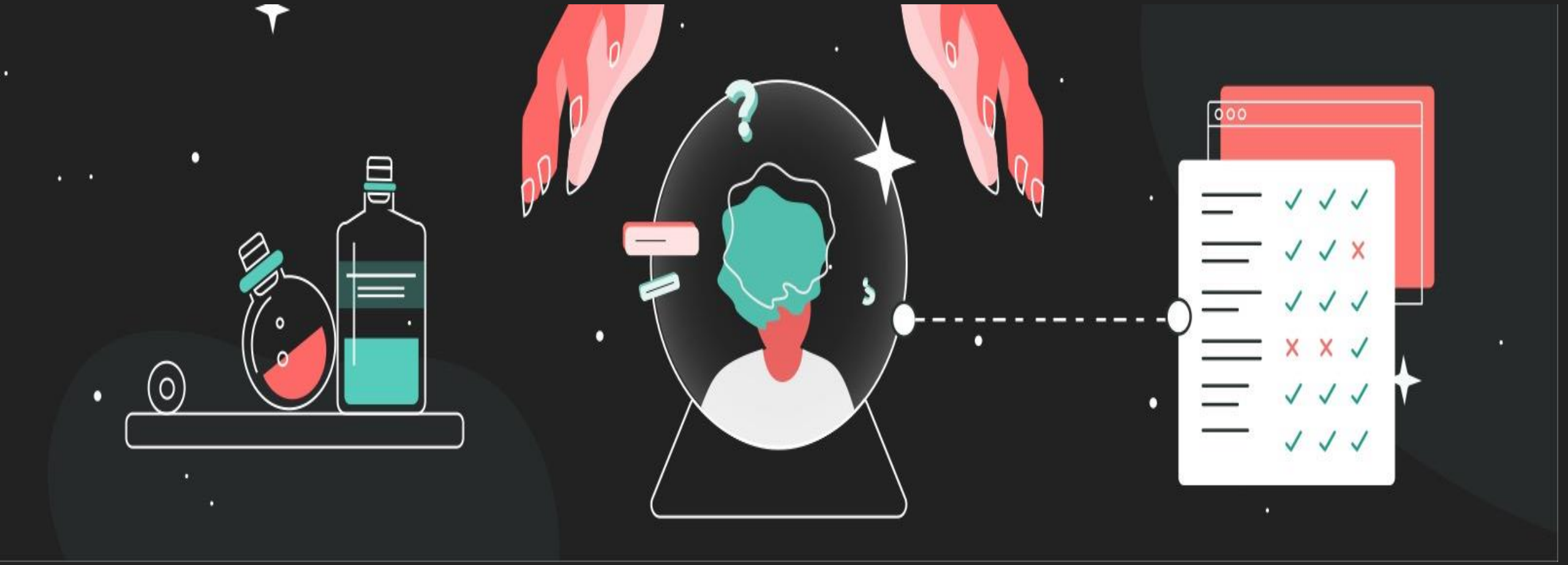
Kubernetes Secrets Management using External Secret Operator (ESO)



Agenda

1. Kubernetes Secrets
2. Challenges of maintaining Kubernetes Secrets
3. External Secret Operator
4. Hands on with ESO on Google Cloud using GKE and Secret manager
5. Best practices of using Kubernetes Secrets

HELLO!

I am Rakesh Saw

I am Lead Engineer with 8+ years of experience in DevOps and Cloud. I am multi cloud and Kubernetes certified .

What is Kubernetes Secret ?

- Kubernetes secret is an object that contains small amount of sensitive data such as token, database credentials or different keys like TLS keys.

- Secrets are created independently of Pods and can se used in different ways inside pods

Uses of Secrets

- Set environment variables for containers
- Provide credentials such as SSH keys or passwords to Pods.
- Allow the kubelet to pull container images from private registries.

Managing Secrets using Configuration File

Create the secret

```
apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
data:
  username: YWRtaW4=
  password: MWYyZDFlMmU2N2Rm
```

Using Secrets as files from a Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  containers:
  - name: mypod
    image: redis
    volumeMounts:
    - name: foo
      mountPath: "/etc/foo"
      readOnly: true
  volumes:
  - name: foo
    secret:
      secretName: mysecret
```

Using secret as environment variable

```
apiVersion: v1
kind: Pod
metadata:
  name: env-single-secret
spec:
  containers:
  - name: envars-test-container
    image: nginx
    env:
    - name: SECRET_USERNAME
      valueFrom:
        secretKeyRef:
          name: mysecret
          key:  username
```

Issue with this approach – Once you push this yaml file in your Github repo then it will be visible to who have access to repo.

Managing Secrets using Kubectl

Create the secret

```
kubectl create secret generic db-user-pass \
   --from-literal=username=admin \
   --from-literal=password='pass123'
```

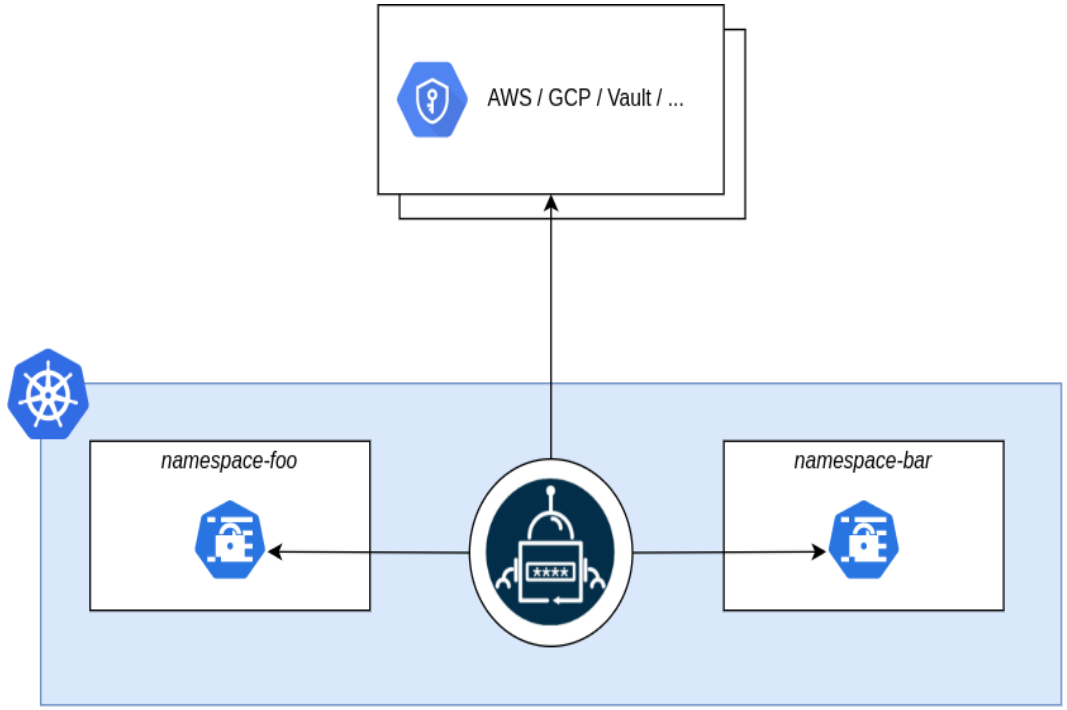
Issue with this approach – Managing secrets manually is quite difficult and it's not best way to store secrets in local or in remote state and update regularly

Challenges of maintaining Kubernetes Secrets

- Manual Management – Managing Secrets manu
- Secret Distributions
- Secret Rotation
- Cross-cluster and Multi-Cloud Challenges
- Auditing and Compliance

External Secret Operator(ESO)

External Secrets Operator is a Kubernetes operator that integrates external secret management systems like **AWS Secrets Manager, HashiCorp Vault, Google Secrets Manager, Azure Key Vault** and many more. The operator reads information from external APIs and automatically injects the values into a Kubernetes Secret.

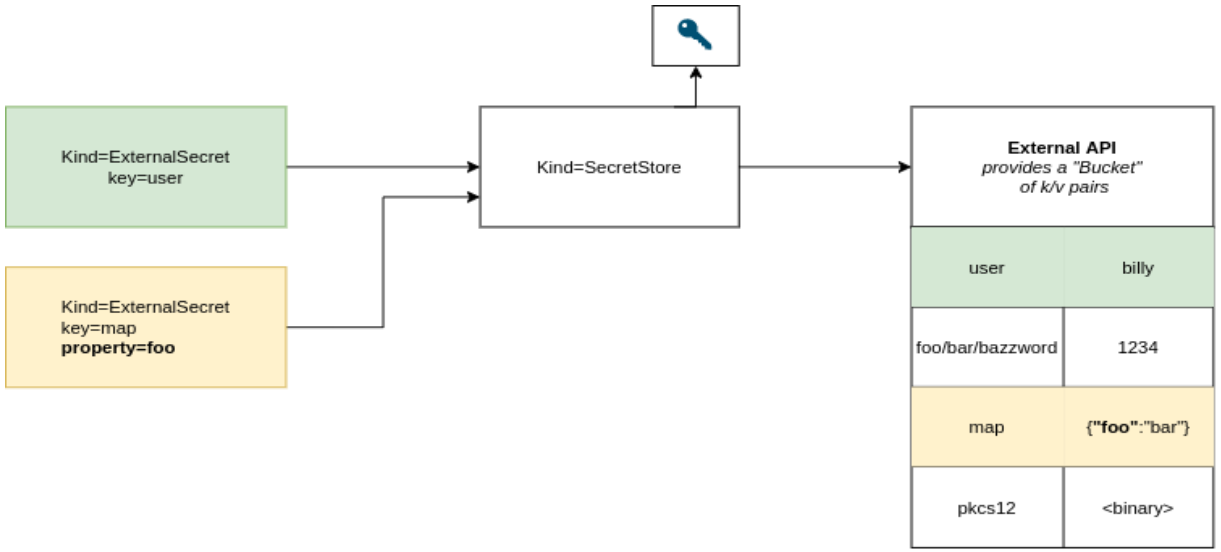


External Secret Operator(ESO)

- ESO is a collections of cutoms APIS resources like External Secret ,SecretStore and ClusterSecretStore that provides a user-friendly abstraction for external APIs that stores and manage the lifecycle of secrets for us.

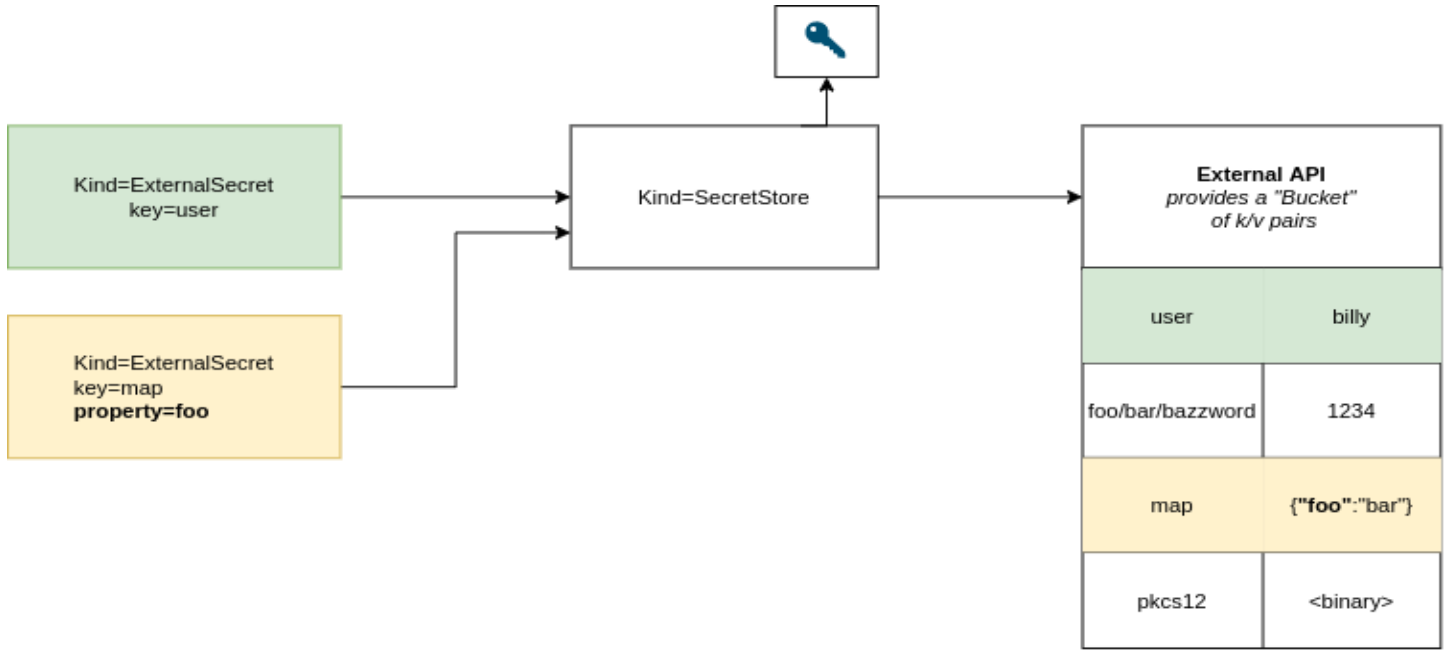
- The External Secrets Operator extends Kubernetes with Custom Resources, which define where secrets live and how to synchronize them

- The controller fetches secrets from an external API and creates Kubernetes secrets. If the secret from the external API changes, the controller will reconcile the state in the cluster and update the secrets accordingly.



Core Resources External Secret Operator

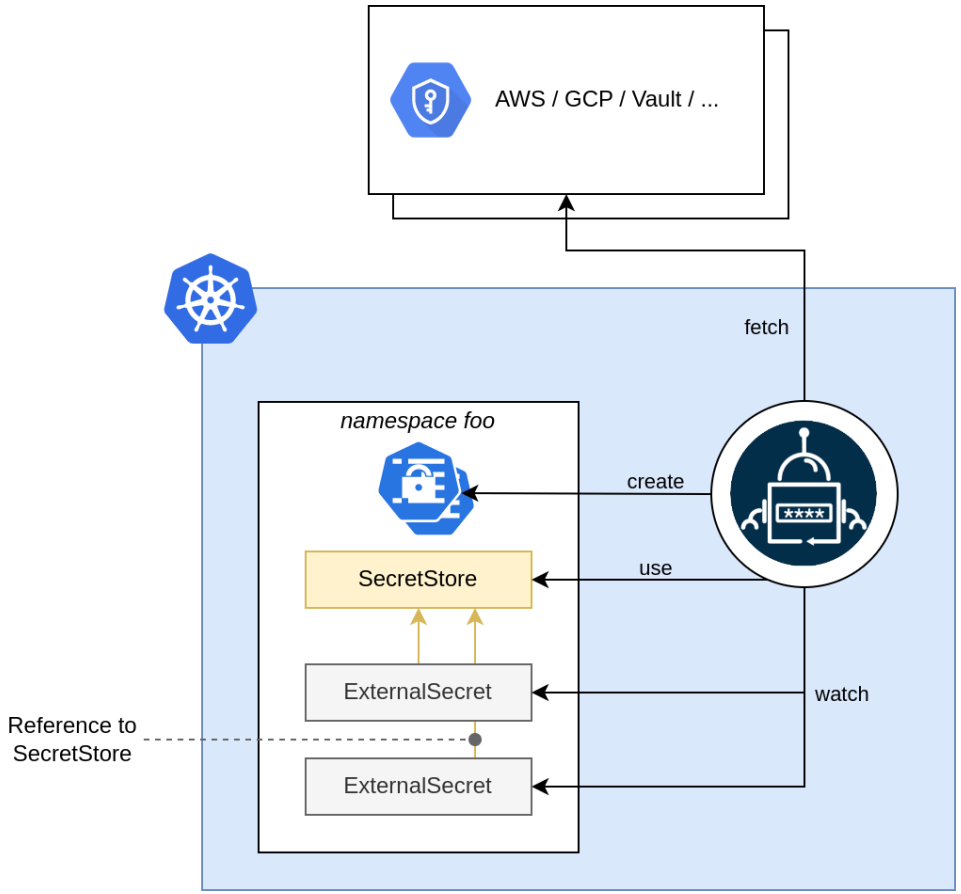
- Provider
- SecretStore
- ClusterSecretStore
- ExternalSecret
- ClusterExternalSecret



SecretStore

SecretStore - It define which provider to use and how to authenticate with provider. It’s a namespaced resource

```yaml
apiVersion: external-secrets.io/v1beta1
kind: SecretStore
metadata:
  name: secretstore-sample   # store name
spec:
  provider:
    aws:                     # provider
      service: SecretsManager
      region: us-east-1
      auth:
        secretRef:
          accessKeyIDSecretRef:
            name: awssm-secret
            key: access-key
          secretAccessKeySecretRef:
            name: awssm-secret
            key: secret-access-key
```



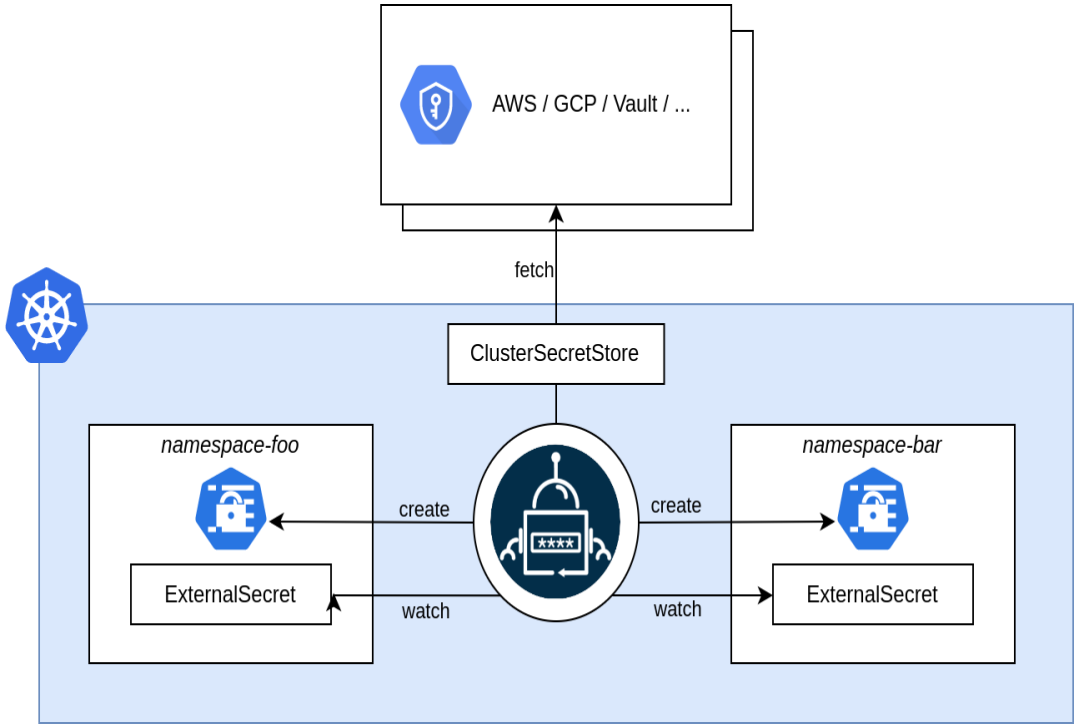
For more details please visit here - https://external-secrets.io/latest/api/secretstore/

ClusterSecretStore

It is as same as SecretStore but it is cluster-scoped .This type of store can be referenced by all ExternalSecrets from different namespace.

```yaml
apiVersion: external-secrets.io/v1beta1
kind: ClusterSecretStore
metadata:
  name: example
spec:
  # Used to select the correct ESO controller (think: ingress.ingressClassName)
  # The ESO controller is instantiated with a specific controller name
  # and filters ES based on this property
  # Optional
  controller: dev

  # provider field contains the configuration to access the provider
  # which contains the secret exactly one provider must be configured.
  provider:
    # (1): AWS Secrets Manager
    # aws configures this store to sync secrets using AWS Secret Manager provider
    aws:
      service: SecretsManager
```



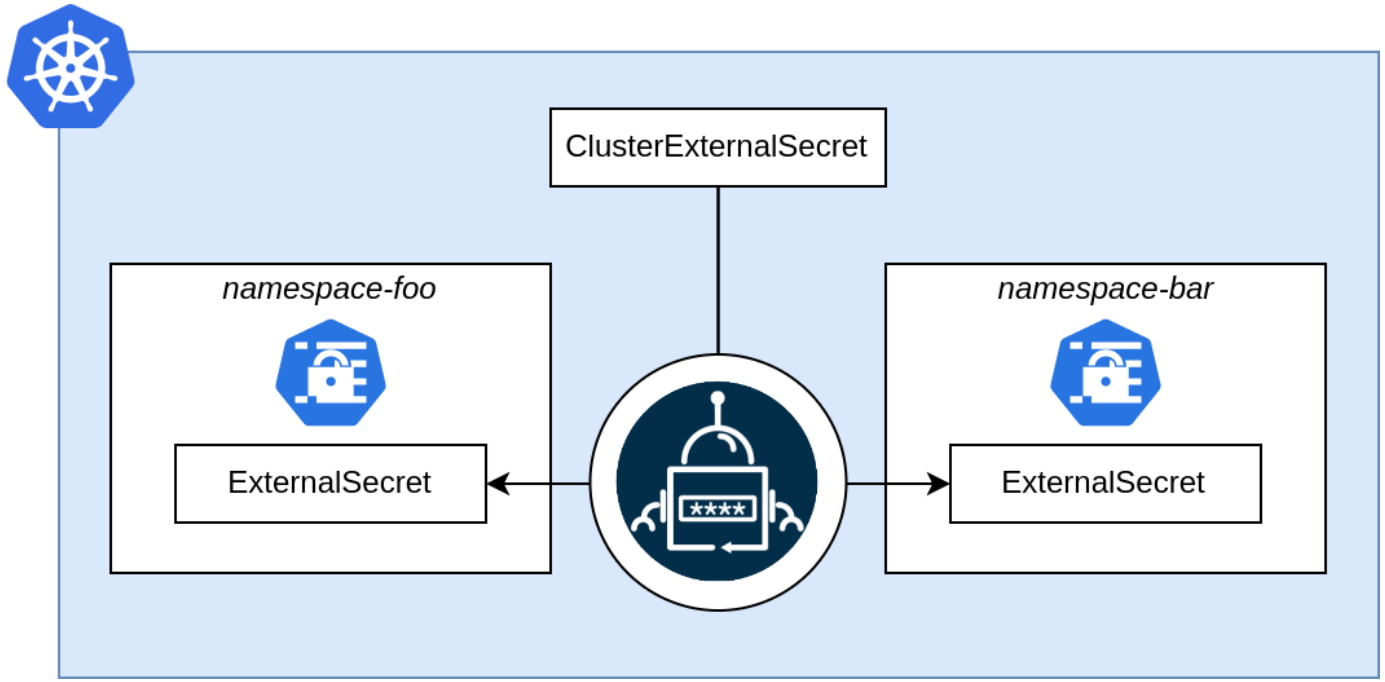
For more details please visit here - https://external-secrets.io/latest/api/pushsecret/

ExternalSecret

The ExternalSecret declares what secret to fetch from external providers. It takes a reference to a SecretStore which knows how to access the provider data.

ClusterExternalSecret

The ClusterExternalSecret is a cluster scoped resource that can be used to manage ExternalSecret resources in specific namespaces.



External Secret Object

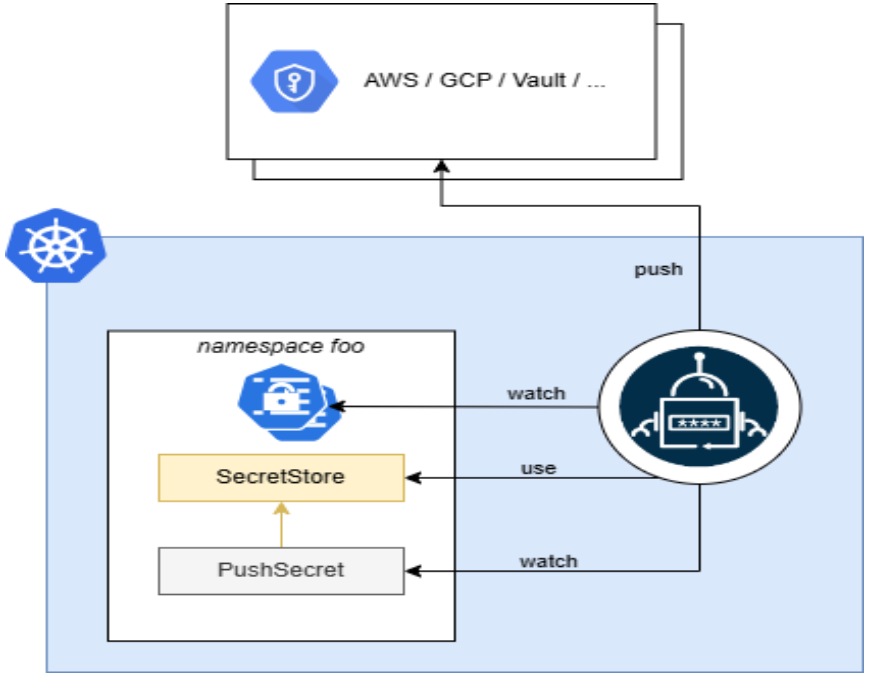
```yaml
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: database-credentials
spec:
  refreshInterval: 1h             # rate SecretManager pulls GCPSM
  secretStoreRef:
    kind: SecretStore
    name: gcp-store               # name of the SecretStore (or kind specified)
  target:
    name: database-credentials    # name of the k8s Secret to be created
    creationPolicy: Owner
  data:
  - secretKey: database_username
    remoteRef:
      key: database_username      # name of the GCPSM secret key
  - secretKey: database_password
    remoteRef:
      key: database_password      # name of the GCPSM secret key
```

For more details please visit here - https://external-secrets.io/latest/api/externalsecret/

PushSecret

The PushSecret is namespaced and it describes what data should be pushed to the secret provider.

```yaml
apiVersion: external-secrets.io/v1alpha1
kind: PushSecret
metadata:
  name: pushsecret-example # Customisable
  namespace: default # Same of the SecretStores
spec:
  deletionPolicy: Delete # the provider' secret will be deleted if the PushSecret is
  refreshInterval: 10s # Refresh interval for which push secret will reconcile
  secretStoreRefs: # A list of secret stores to push secrets to
    - name: aws-parameterstore
      kind: SecretStore
  selector:
    secret:
      name: pokedex-credentials # Source Kubernetes secret to be pushed
  template:
    metadata:
      annotations: { }
      labels: { }
    data:
```



For more details please visit here - https://external-secrets.io/latest/api/pushsecret/

Let’s See ESO in Action

Prerequisites for this hands-on

- GCP Account and one project – You can create free tier account to practice - https://cloud.google.com/docs/get-started
- Understanding of Kubernetes
- Install GCP SDK like gcloud and authenticate gcloud to interact with GCP

Export these variables

```
export GCP_PROJECT_ID=weighty-legend-415316
export GCP_ZONE=us-central1-a
export ESO_GCP_SERVICE_ACCOUNT=secret-accessor       #  Google IAM Service Account
export ESO_K8S_NAMESPACE=external-secrets           #  Kubenetes namespace to deploy
export ESO_K8S_SERVICE_ACCOUNT=external-secrets      #  Kubernetes Service Account
```

create the GKE cluster

```
gcloud container clusters create eso-cluster --zone=$GCP_ZONE --workload-pool=$GCP_PROJECT_ID.svc.id.goog \
--machine-type "e2-medium" --num-nodes "2"  --disk-size "50" \
--project $GCP_PROJECT_ID --scopes="https://www.googleapis.com/auth/cloud-platform"
```

Fetch the credentails to connect to GKE control plane

```
gcloud container clusters get-credentials eso-cluster --zone $GCP_ZONE --project $GCP_PROJECT_ID
```

```
# create a GCP service Account
gcloud iam service-accounts create $ESO_GCP_SERVICE_ACCOUNT \
--project=$GCP_PROJECT_ID

# Assign IAM permission on Service Account to access secret from Google Secret Manager

gcloud projects add-iam-policy-binding $GCP_PROJECT_ID \
--member "serviceAccount:$ESO_GCP_SERVICE_ACCOUNT@$GCP_PROJECT_ID.iam.gserviceaccount.com" \
--role "roles/secretmanager.secretAccessor" --project=$GCP_PROJECT_ID

# IAM binding to allow Kubernetes service account to act as Google service account

gcloud iam service-accounts add-iam-policy-binding $ESO_GCP_SERVICE_ACCOUNT@$GCP_PROJECT_ID.iam.gserviceaccount.com \
  --role roles/iam.workloadIdentityUser \
   --member "serviceAccount:$GCP_PROJECT_ID.svc.id.goog[$ESO_K8S_NAMESPACE/$ESO_K8S_SERVICE_ACCOUNT]" --project=$GCP_PROJECT_ID
```

```
# Now add helm charts
 helm repo add external-secrets https://charts.external-secrets.io

 # update the charts
helm repo update

 # Now install the External secret using the charts . We are annotating the Kubernetes service account to use Google service account

helm upgrade -install external-secrets external-secrets/external-secrets \
   --set 'serviceAccount.annotations.iam\.gke\.io\/gcp-service-account'="$ESO_GCP_SERVICE_ACCOUNT@$GCP_PROJECT_ID.iam.gserviceaccount.com" \
   --namespace external-secrets \
   --create-namespace \
   --debug \
   --wait
# Get All the resources installed

kubectl get all -n external-secrets

# let's create Secret in Google Secret Manager

printf "user1" | gcloud secrets create db-username --data-file=- --project=$GCP_PROJECT_ID
printf "pass1" | gcloud secrets create db-password --data-file=- --project=$GCP_PROJECT_ID
```

```
# create ClusterScret Store

cat <<EOF | kubectl apply -f -
---
apiVersion: external-secrets.io/v1beta1
kind: ClusterSecretStore
metadata:
  name: gcp-store
spec:
  provider:
    gcpsm:
      projectID: $GCP_PROJECT_ID
EOF
```

create app namespace to create secret

Kubectl create namespace app

create ExternalSecret Object

```
cat <<EOF | kubectl apply -f -
---
apiVersion: external-secrets.io/v1beta1
kind: ExternalSecret
metadata:
  name: database-creds
  namespace: app
spec:
  refreshInterval: 10s          # rate SecretManager pulls GCPSM, Low refereshInternval for demo purpose,Set this value based based on apps
  secretStoreRef:
    kind: ClusterSecretStore
    name: gcp-store             # name of the ClusterSecretStore or you can also reference SecretStore
  target:
    name: db-creds              # name of the k8s Secret to be created
    creationPolicy: Owner
  data:
  - secretKey: db-user          # name of secretkey it can be any name
    remoteRef:
      key: db-username          # name of the GCPSM secret key
  - secretKey: db-pass          # name of secretkey it can be any name
    remoteRef:
      key: db-password          # name of the GCPSM secret key
EOF
```

```
# get the secret from Kubernetes

kubectl get secret db-creds -n app  -o jsonpath='{.data.db-user}' | base64 -d
kubectl get secret db-creds -n app  -o jsonpath='{.data.db-pass}' | base64 -d


# update the secret in GSM

printf "user2" | gcloud secrets versions add db-username --data-file=- --project=$GCP_PROJECT_ID
printf "pass2" | gcloud secrets versions add db-password --data-file=- --project=$GCP_PROJECT_ID


# get the values again

kubectl get secret db-creds -n app  -o jsonpath='{.data.db-user}' | base64 -d
kubectl get secret db-creds -n app  -o jsonpath='{.data.db-pass}' | base64 -d
```

Best Practices of using Secrets

- Enable Encryption at Rest for Secrets.
- Enable or configure RBAC rules with least-privilege access to Secrets.
- Restrict Secret access to specific containers.
- Configure access to external Secrets.
- Avoid sharing Secret manifests.

References

- **Official ESO page - https://external-secrets.io/latest/**
- **Providers- https://external-secrets.io/latest/provider/aws-secrets-manager/**
- **Kubernetes Secret - https://kubernetes.io/docs/concepts/configuration/secret/**
- **Using GCP secret Manager - https://external-secrets.io/latest/provider/google-secrets-manager/**
- **Community Contribution - https://external-secrets.io/latest/contributing/process/**

THANKS!

Any questions?

You can find me at:

- rakesh_saw@epam.com