Gérer ses secrets Kube avec l'External Secrets Operator : parce que je le Vault bien

DEEZER

~#whoami : Alexis Fala

~~Clown~~ Infrastructure engineer | SRE chez DEEZER

- → Passionné par l'open-source et le Cloud Native.
- → Coorganisateur de Cloud Native Bordeaux
- → Membre des **SRE du coeur** avec Idriss Neumann et Julien Briault.
- → Auteur à ses heures perdues sur Deezer.io.

LePotiBlagueur

@LePotiBlagueur

https://www.linkedin.com/in/alexis-fala/

Sommaire

I. Les secrets dans Kubernetes
II. Introduction à Vault
III. Qu'est-ce qu'un operator ?
IV. L'External Secrets Operator
V. Petite ouverture

Les secrets dans Kubernetes

DEEZER

Un secret dans le cluster

Un secret certes... Mais pas si secret

- Contient des informations sensibles
- Permet de stocker mot de passe, tokens, certificats, ...
- Encodé en base64 😬

```yaml
apiVersion: v1
kind: Secret
metadata:
 name: mon-super-secret
stringData:
 password: password1234
```

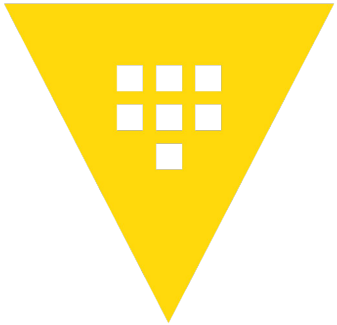
DEEZER

Introduction à Vault

DEEZER

Vault ? Kesako ?

Coffre-fort “API driven” le plus utilisé du marché.

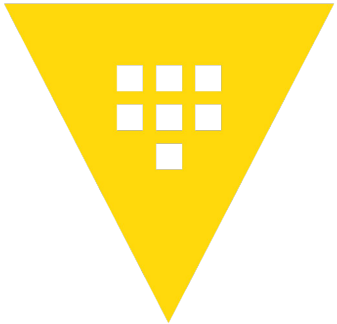


HashiCorp
Vault

Vault ? Kesako ?

Coffre-fort “API driven” le plus utilisé du marché.

- *Secret-engine* : C’est le coffre-fort où sont stockés les secrets.

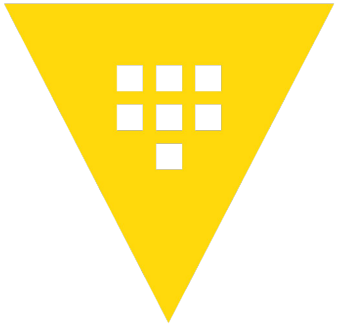


HashiCorp
Vault

Vault ? Kesako ?

Coffre-fort “API driven” le plus utilisé du marché.

- *Secret-engine* : C’est le coffre-fort où sont stockés les secrets.
- *Auth Method* : Les méthodes d’authentifications à Vault (Token, OIDC, Kubernetes, ...).

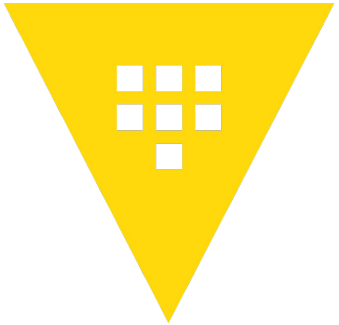


HashiCorp
Vault

Vault ? Kesako ?

Coffre-fort “API driven” le plus utilisé du marché.

- *Secret-engine* : C’est le coffre-fort où sont stockés les secrets.
- *Auth Method* : Les méthodes d’authentifications à Vault (Token, OIDC, Kubernetes, ...).
- *Path* : Chemin d’accès vers une ressource dans Vault (secret, authentification, configuration, ...).

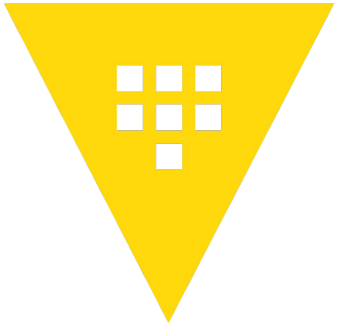


HashiCorp
Vault

Vault ? Kesako ?

Coffre-fort “API driven” le plus utilisé du marché.

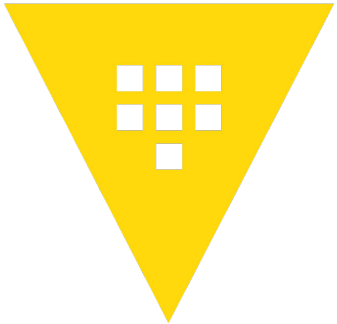
- *Secret-engine* : C’est le coffre-fort où sont stockés les secrets.
- *Auth Method* : Les méthodes d’authentifications à Vault (Token, OIDC, Kubernetes, ...).
- *Path* : Chemin d’accès vers une ressource dans Vault (secret, authentification, configuration, ...).
- *Policies* : Vont ajouter des règles sur l’accès à certains *paths* dans Vault (comme les secrets).



HashiCorp
Vault

Vault ? Kesako ?

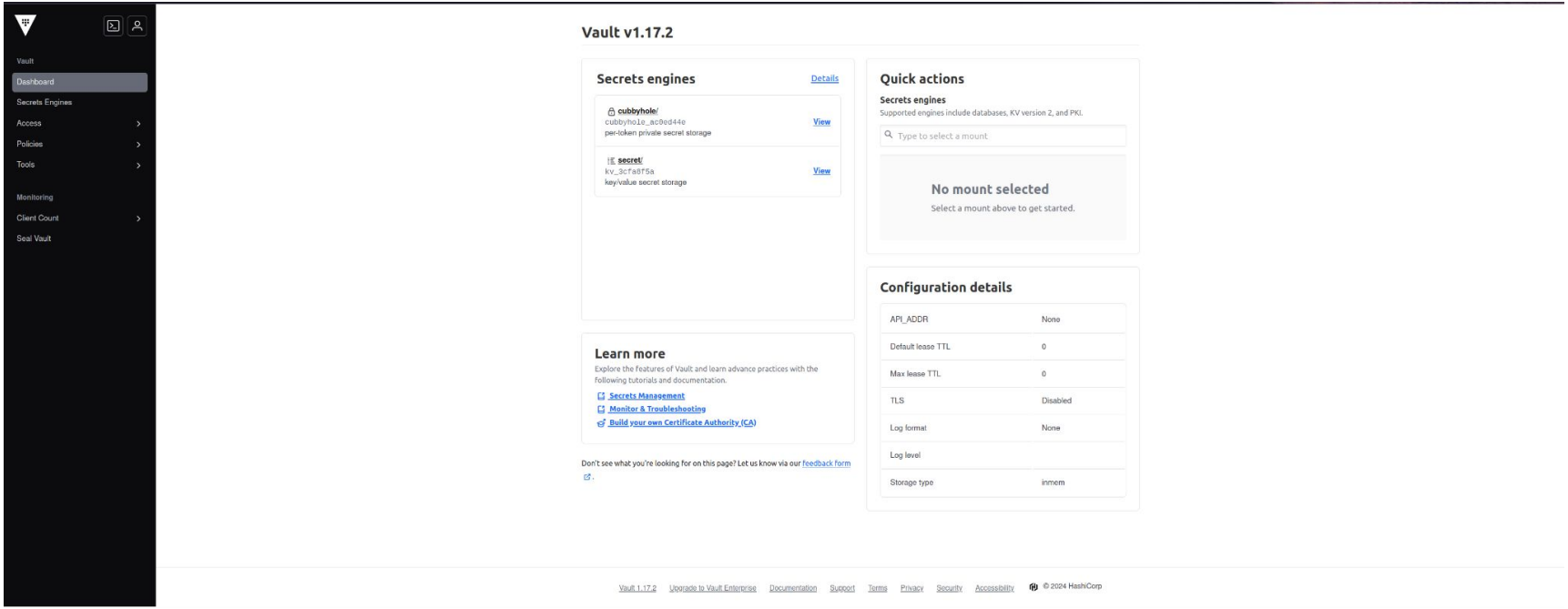
Coffre-fort “API driven” le plus utilisé du marché.



HashiCorp
Vault

- *Secret-engine* : C’est le coffre-fort où sont stockés les secrets.
- *Auth Method* : Les méthodes d’authentifications à Vault (Token, OIDC, Kubernetes, ...).
- *Path* : Chemin d’accès vers une ressource dans Vault (secret, authentification, configuration, ...).
- *Policies* : Vont ajouter des règles sur l’accès à certains *paths* dans Vault (comme les secrets).
- Authentification → Validation → Autorisation → Accès

Vault UI

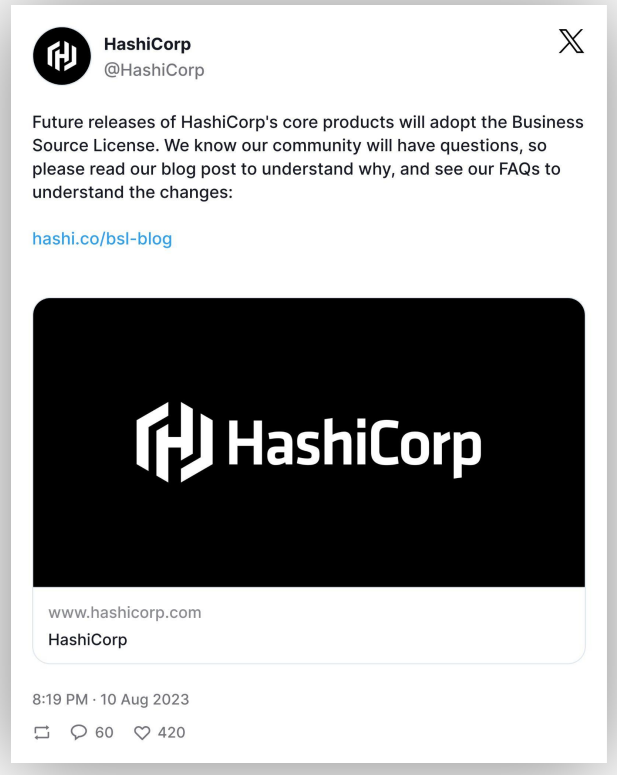


DEEZER

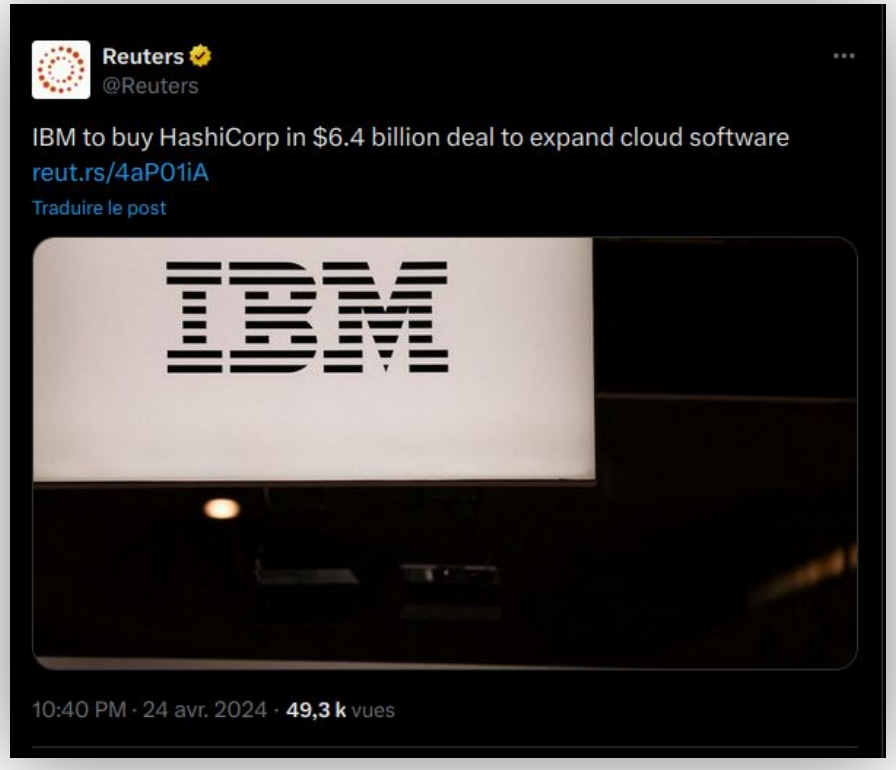
Alerte drama

DEEZER

La Business Source Licence



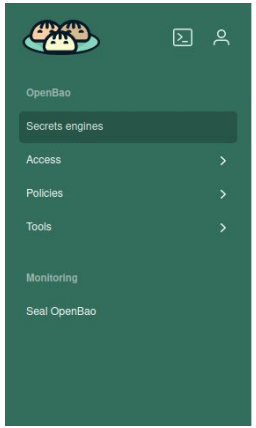
Le rachat par IBM

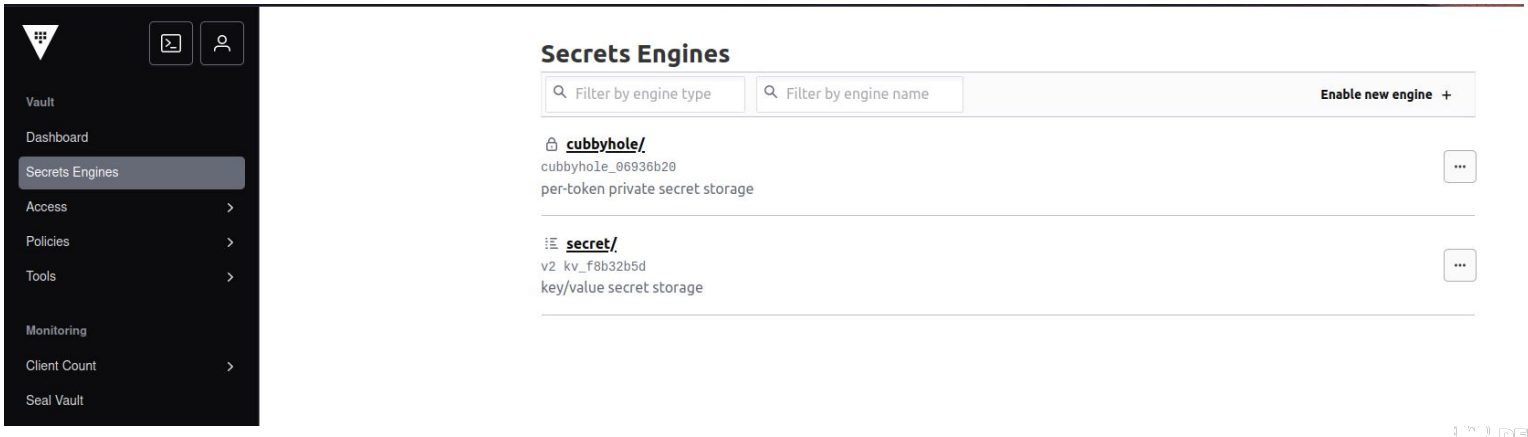


OpenBao


Y a des diffs ?

Le jeu des 7 différences



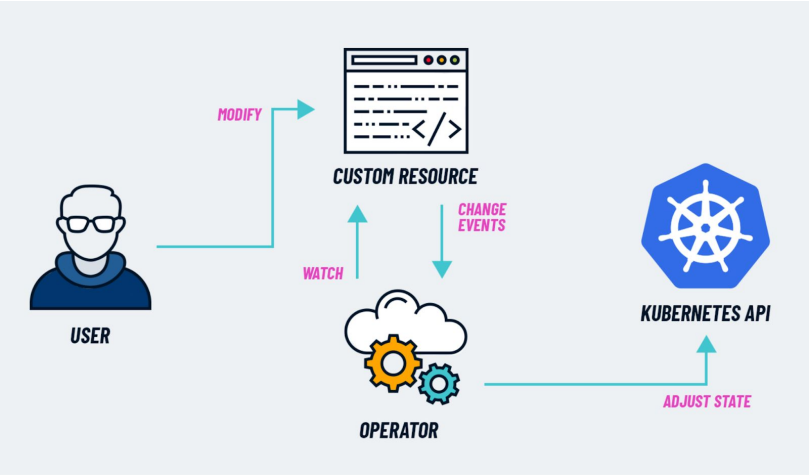


Qu’est-ce qu’un operator ?

Kubernetes Operators

Les opérateurs sont des extensions de l'API/Control plane de Kubernetes.

- Permettent de manager des applications via l'utilisation de CRD en les associant à un controller.
- Se basent sur les principes du control loop.
- Peut gérer des composants internes et externes au cluster.
- Il en existe une grande variété : cert-manager pour gérer les certificats TLS, CNPG pour déployer Postgrès...
- A la recherche d'un Operator ? https://operatorhub.io/



Source cncf.io

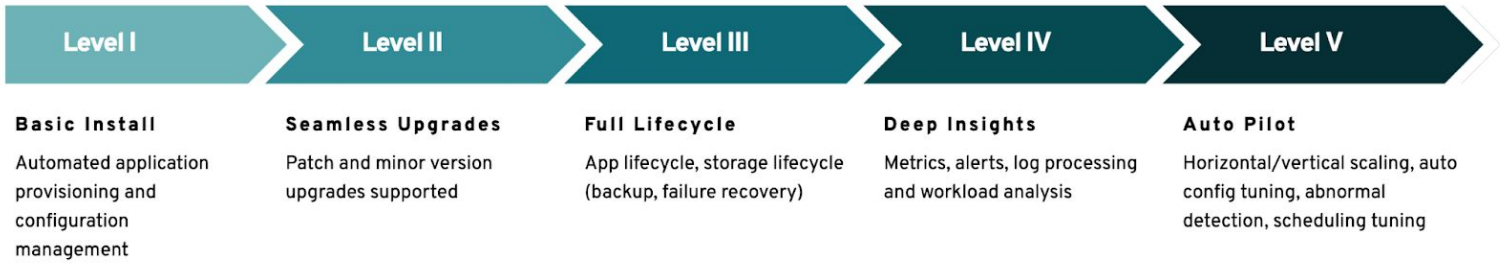
Vous pouvez écrire vos propres opérateurs.

Comment écrire son opérateur ?

S'aider d'un framework

- Operator Framework : https://operatorframework.io/
- Kubebuilder : https://kubebuilder.io/
- Kubernetes Operators Framework : https://kopf.readthedocs.io/en/stable/
- ...

Plusieurs niveaux de capacité



Source operatorframework.io

Si ça ne tourne pas dans Kube, ça peut être géré via Kube.

DEEZER

Le Vault Secrets Operator

DEEZER

~~Le Vault Secrets Operator~~

DEEZER

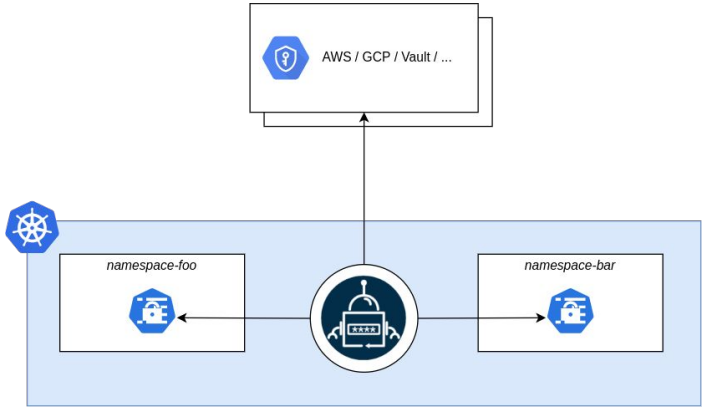
L'External Secrets Operator

DEEZER

Un operator pour les gouverner tous

Spoiler : il ne marche PAS qu'avec Vault

- A pour but de synchroniser les secrets contenus dans les secret engine et les secrets Kubernetes.
- Supporte de nombreuses technos de management de secret comme AWS Secrets Manager, GSM, AKV, IBM, Pulumi...
- Totalement open-source
- Possède une cli *esoctl* et d'autres fonctionnalités apportées par les CRD comme le *Generator* ou encore le *Push Secret*



Source external-secrets.io

Comment le déployer ?

C'est simple !

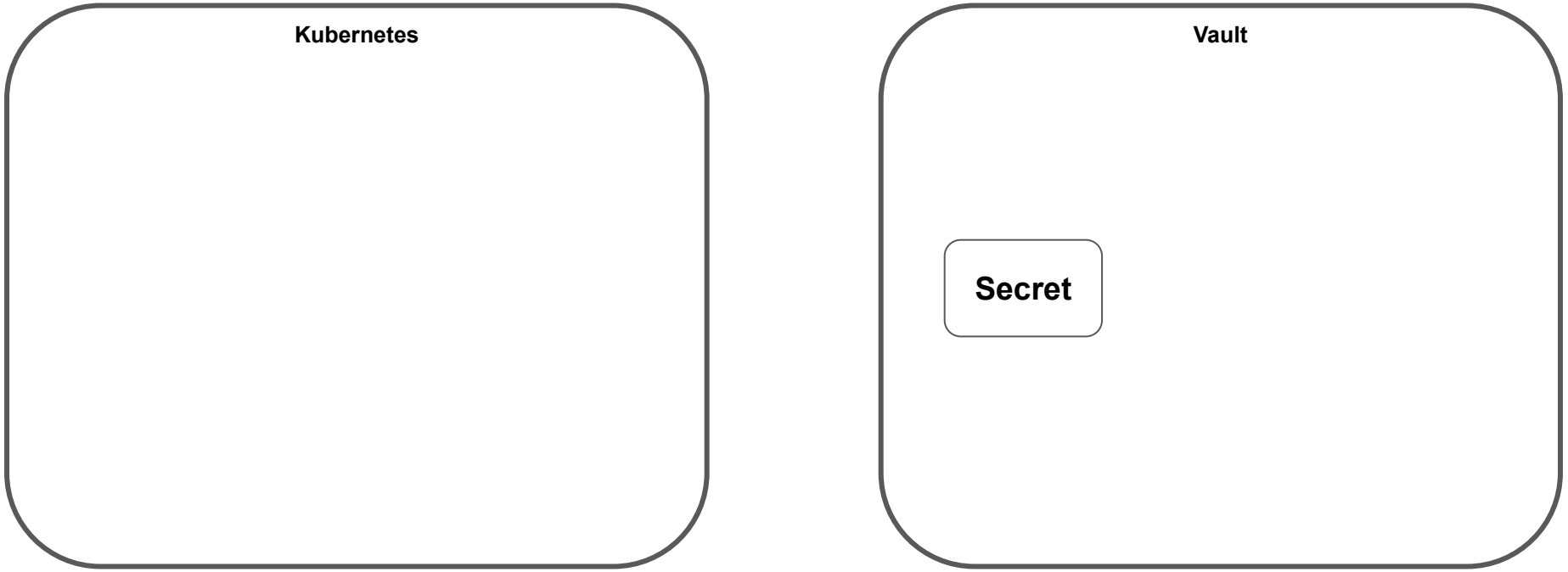
```
helm install external-secrets
external-secrets/external-secrets -n external-secrets
\ --create-namespace
```

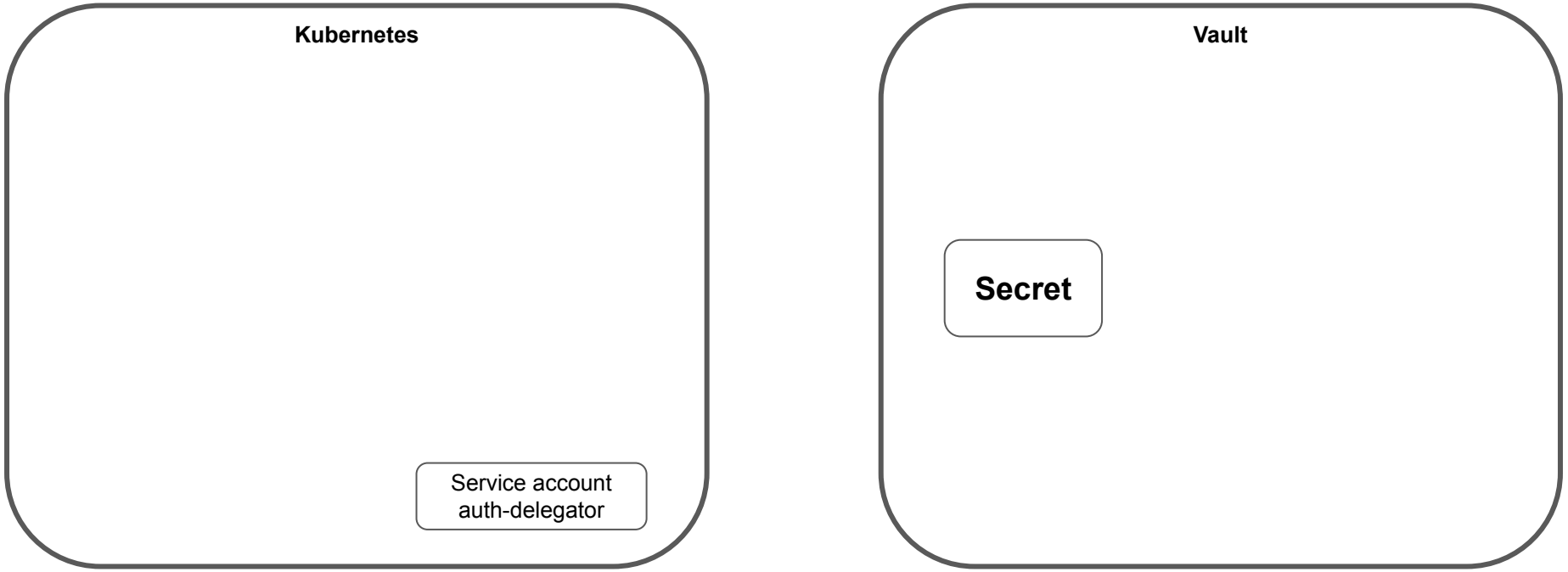
https://artifacthub.io/packages/helm/external-secrets-operator/external-secrets

Les étapes pour faire marcher l'opérateur

Kubernetes

Vault


Kubernetes
Vault
Secret


Kubernetes
Service account
auth-delegator
Vault
Secret

Création d'un service account

Première étape

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
 name: operator-auth
```

Création d'un service account

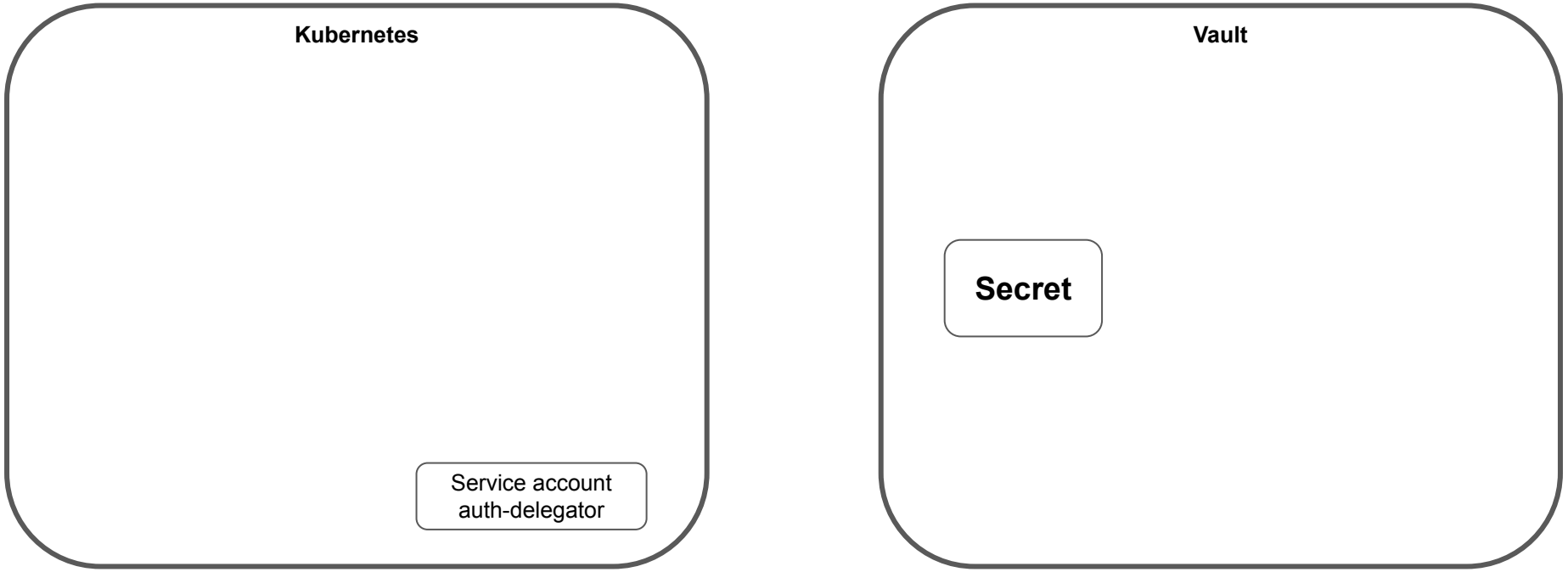
Première étape (bis)

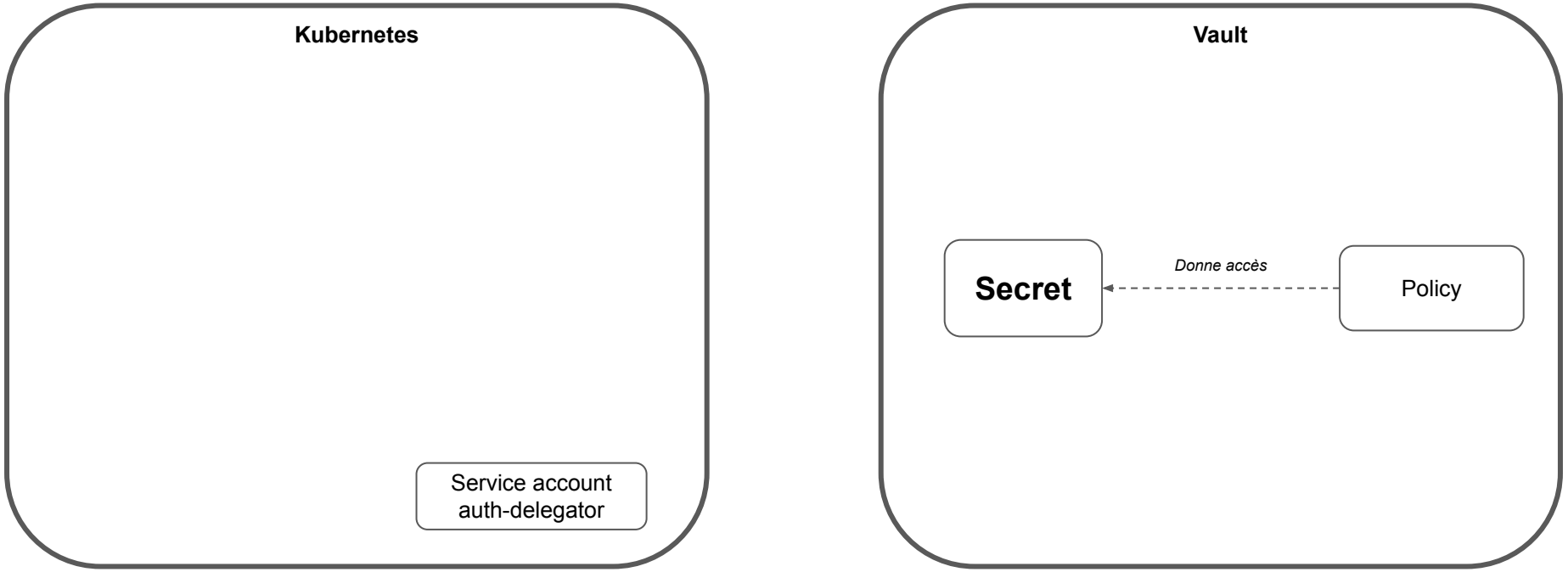
```yaml
apiVersion: v1
kind: Secret
metadata:
 name: operator-auth
 annotations:
   kubernetes.io/service-account.name: operator-auth
type: kubernetes.io/service-account-token
```

Création d'un service account

Première étape (ter)

```
apiVersion:
rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
 name: role-tokenreview-binding
roleRef:
 apiGroup: rbac.authorization.k8s.io
 kind: ClusterRole
 name: system:auth-delegator
subjects:
 - kind: ServiceAccount
   name: operator-auth
   namespace: [namespace du SA]
```


Kubernetes
Service account
auth-delegator
Vault
Secret

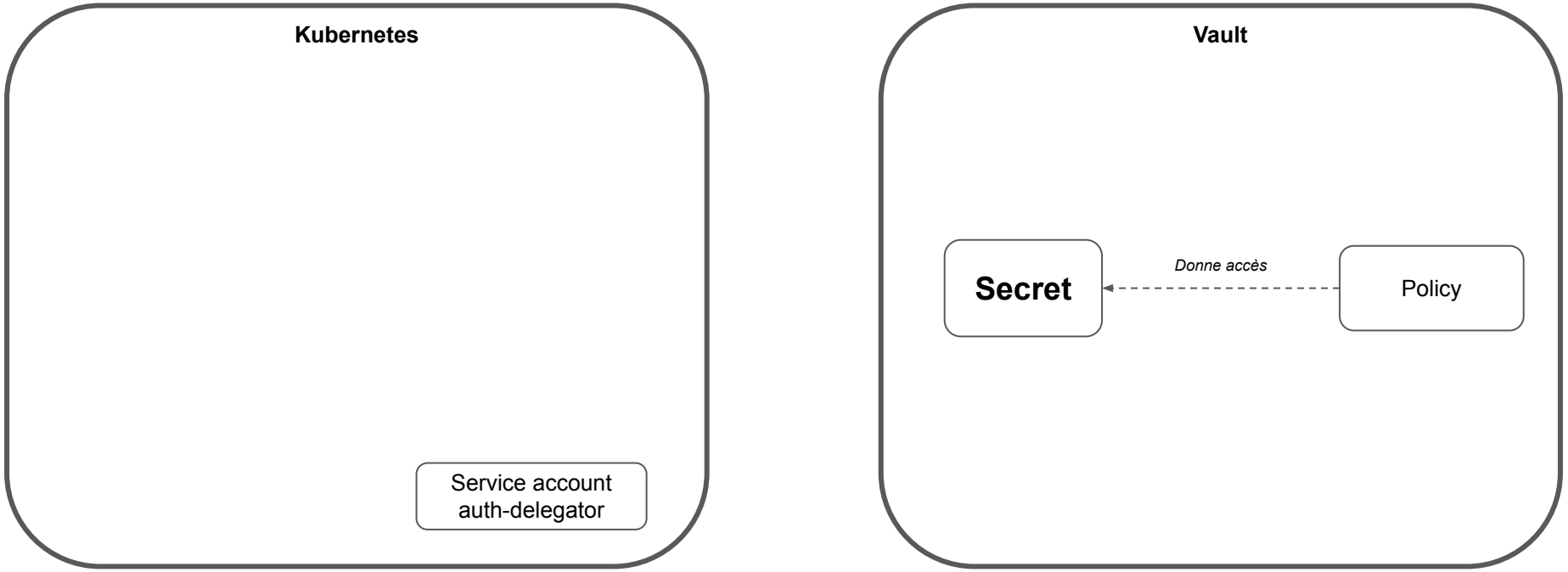

Kubernetes
Service account
auth-delegator
Vault
Secret
Donne accès
Policy

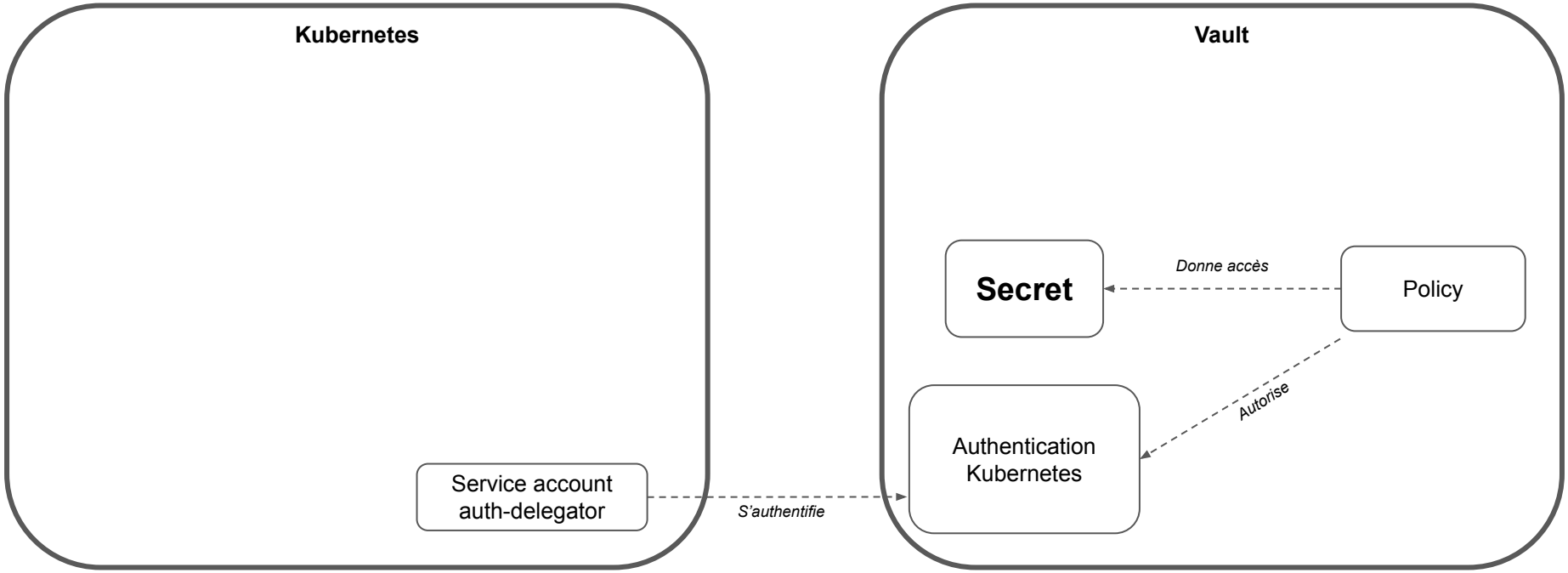
La configuration du Vault

Seconde étape : on écrit une policy

```
vault policy write external-secrets-operator
./policy.hcl
```

```
path "[secret engine]/data/[path des secrets]/*"{
  capabilities = ["create", "read", "update", "patch",
"delete", "list"]
}
path "[secret engine]/metadata/[path des secrets]/*" {
  capabilities = ["create", "read", "update", "patch",
"delete", "list"]
}
```


Kubernetes
Service account
auth-delegator
Vault
Secret
Donne accès
Policy


Kubernetes
Vault
Secret
Donne accès
Policy
Autorise
Authentication
Kubernetes
Service account
auth-delegator
S'authentifie

La configuration du Vault

Seconde étape (bis) : on créé notre authentification Kube

```
vault auth enable -path external-secrets-operator kubernetes
```

La configuration du Vault

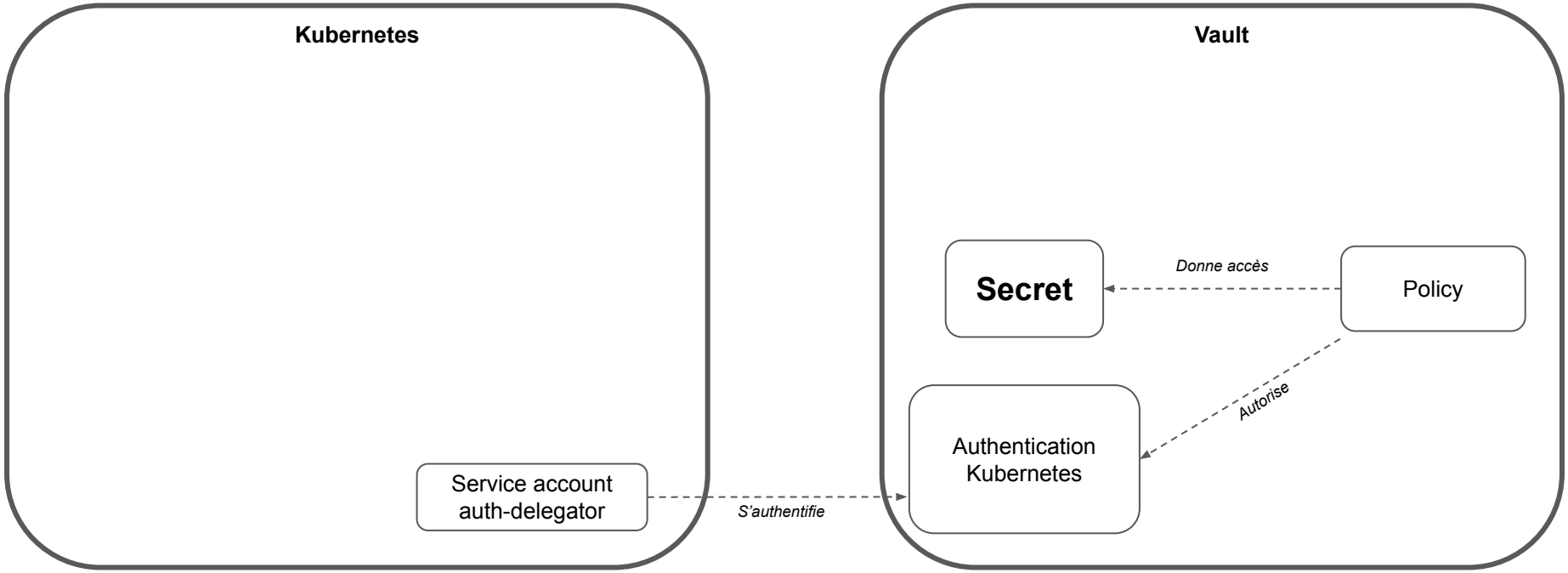
Seconde étape (ter) : on créé la config de notre authentification

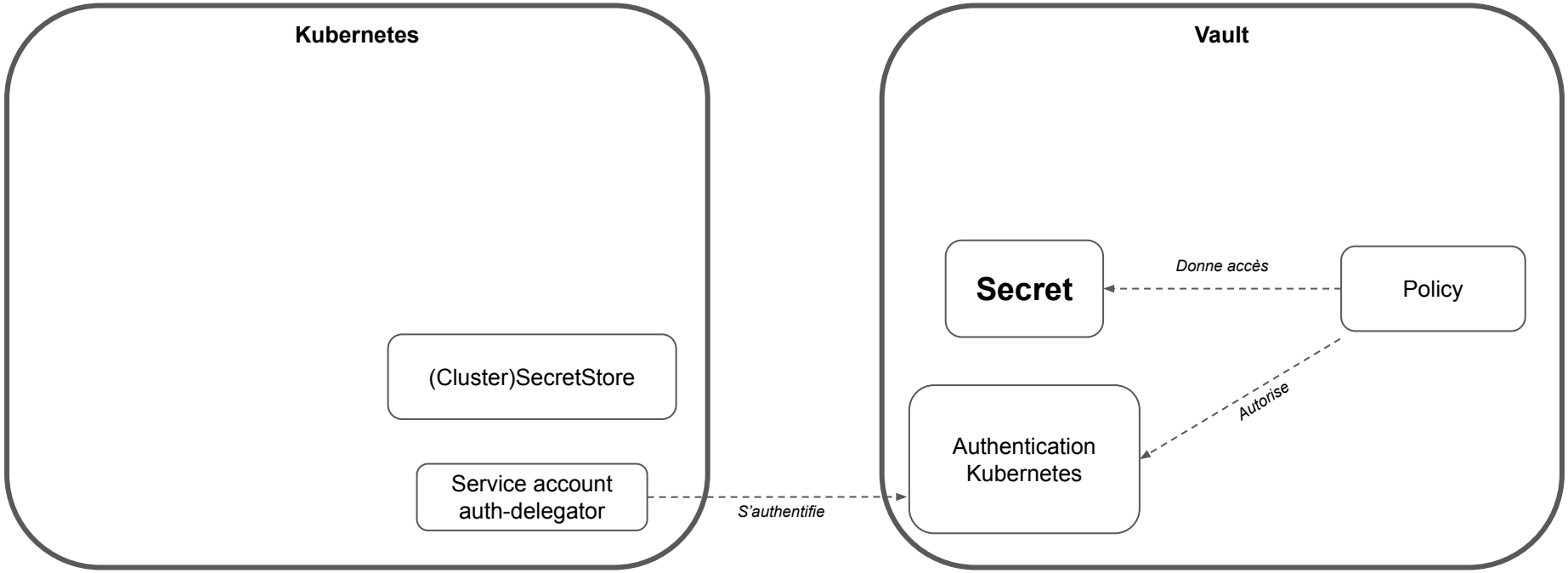
```
vault write auth/external-secrets-operator/config
token reviewer jwt="$TOKEN REVIEW_JWT"
kubernetes host="$KUBE HOST"
kubernetes_ca_cert="$KUBE_CA_CERT"
```

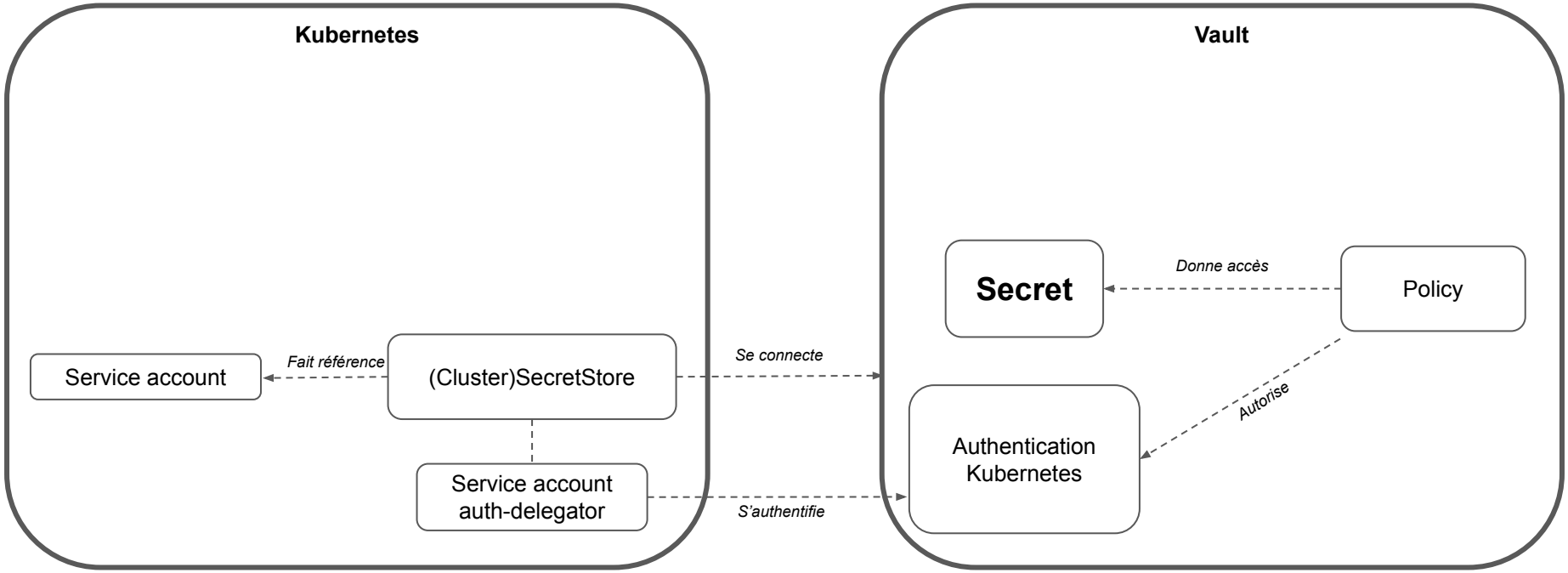
La configuration du Vault

Seconde étape (quater) : on associe un rôle à notre authentification

```
vault write auth/external-secrets-operator/role/default
bound service account names=myapp
bound service account namespaces=”myapp”
policies=external-secrets-operator
```


Kubernetes
Vault
Secret
Donne accès
Policy
Autorise
Authentication
Kubernetes
Service account
auth-delegator
S'authentifie


Kubernetes
Vault
Secret
Donne accès
Policy
(Cluster)SecretStore
Autorise
Authentication
Kubernetes
Service account
auth-delegator
S'authentifie

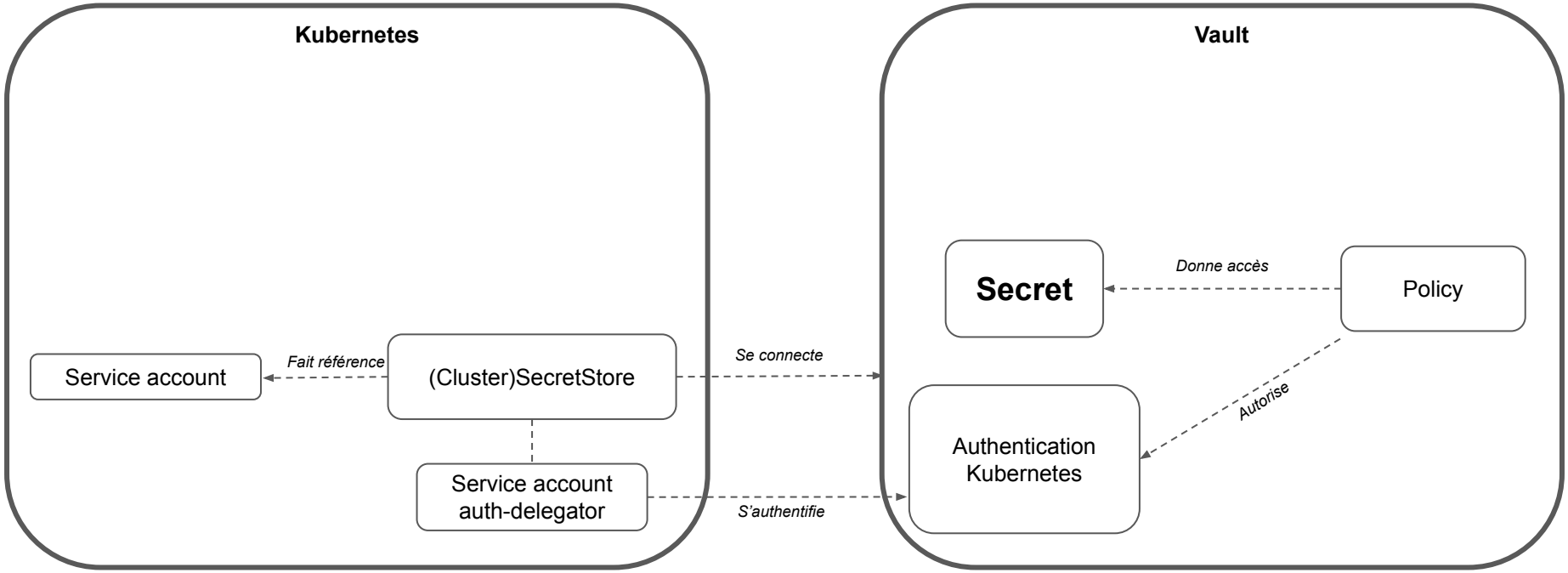

Kubernetes
Vault
Secret
Donne accès
Policy
Service account
Fait référence
(Cluster)SecretStore
Se connecte
Autorise
Authentication
Kubernetes
Service account
auth-delegator
S'authentifie

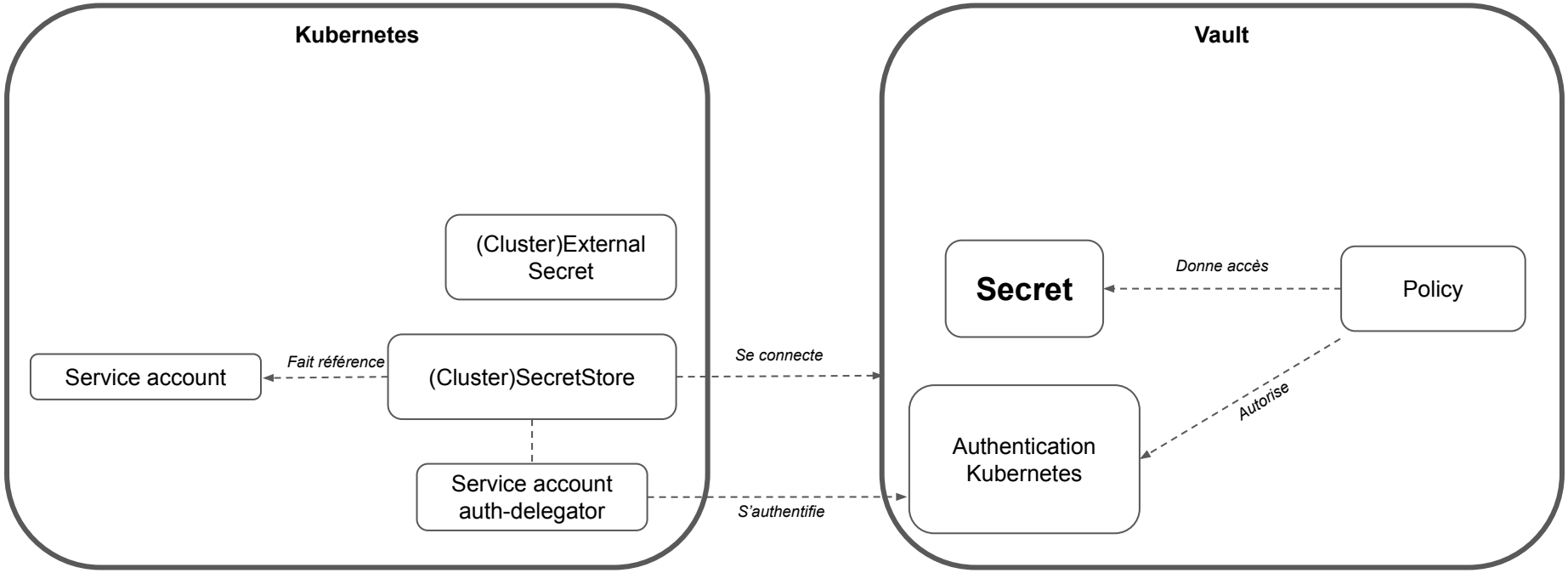
Le SecretStore

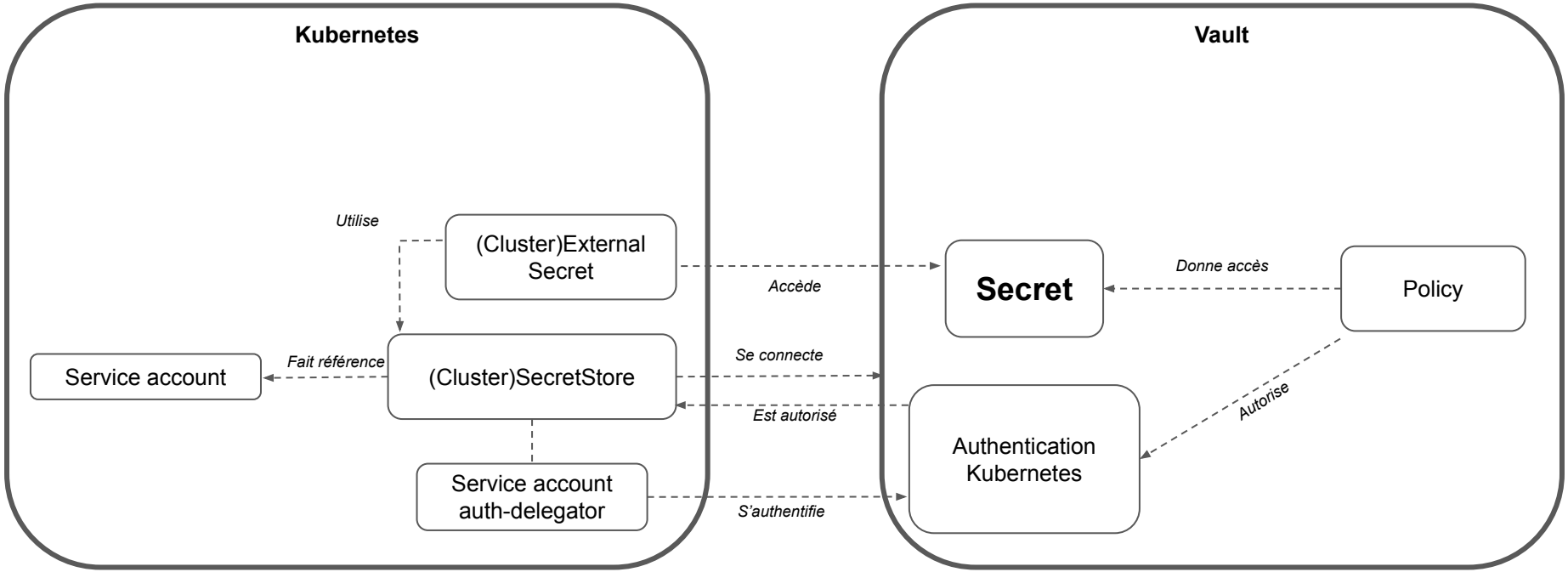
Troisième étape : On se connecte et s'authentifie au Vault

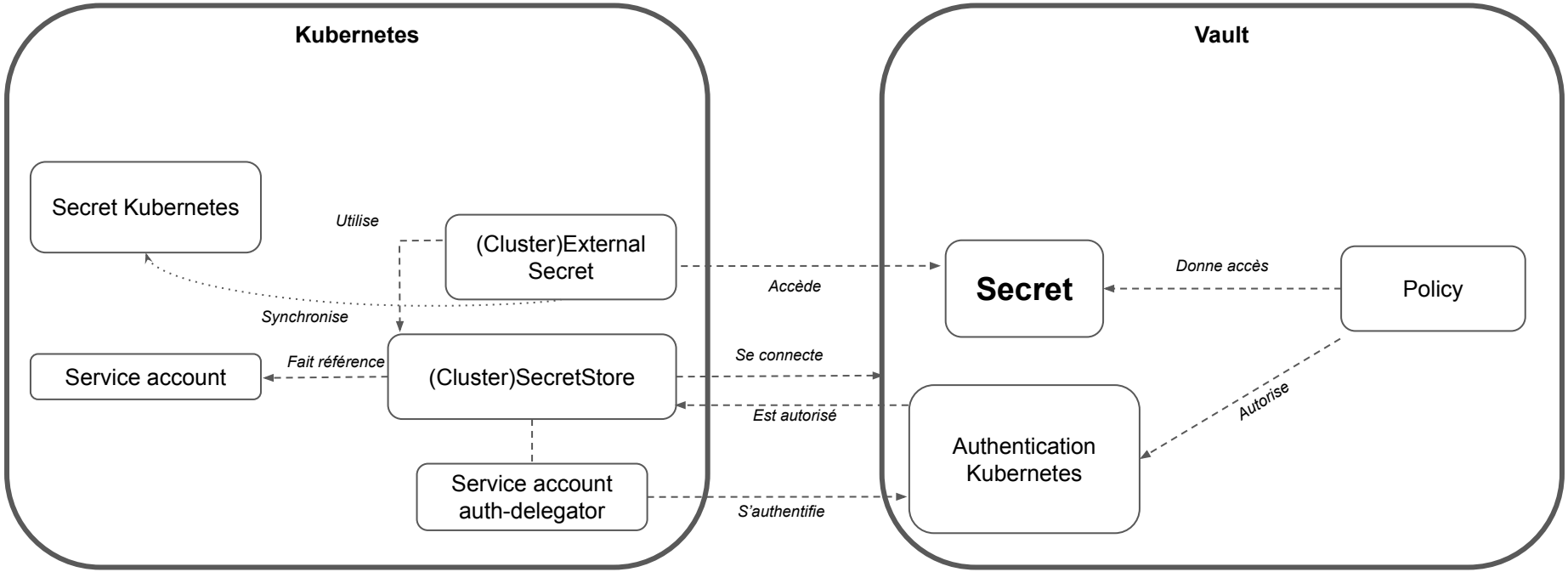
```yaml
apiVersion: external-secrets.io/v1
kind: SecretStore
metadata:
 name: secretstore
spec:
 provider:
   vault:
     server: [Adresse de Vault]
     path: [Nom du secret engine]
     version: "v2"
     auth:
       kubernetes:
         mountPath: [Nom de l'authent Kubernetes]
         role: [Nom du rôle utilisé]
         serviceAccountRef:
           name: [Nom du serviceaccount utilisé]
```

DEEZER


Kubernetes
Vault
Secret
Donne accès
Policy
Service account
Fait référence
(Cluster)SecretStore
Se connecte
Autorise
Authentication
Kubernetes
Service account
auth-delegator
S'authentifie


Kubernetes
(Cluster)External Secret
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Se connecte
S'authentifie
Vault
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes


Kubernetes
Vault
Utilise
(Cluster)External Secret
Accède
Secret
Donne accès
Policy
Service account
Fait référence
(Cluster)SecretStore
Se connecte
Est autorisé
Autorise
Authentication Kubernetes
Service account auth-delegator
S'authentifie

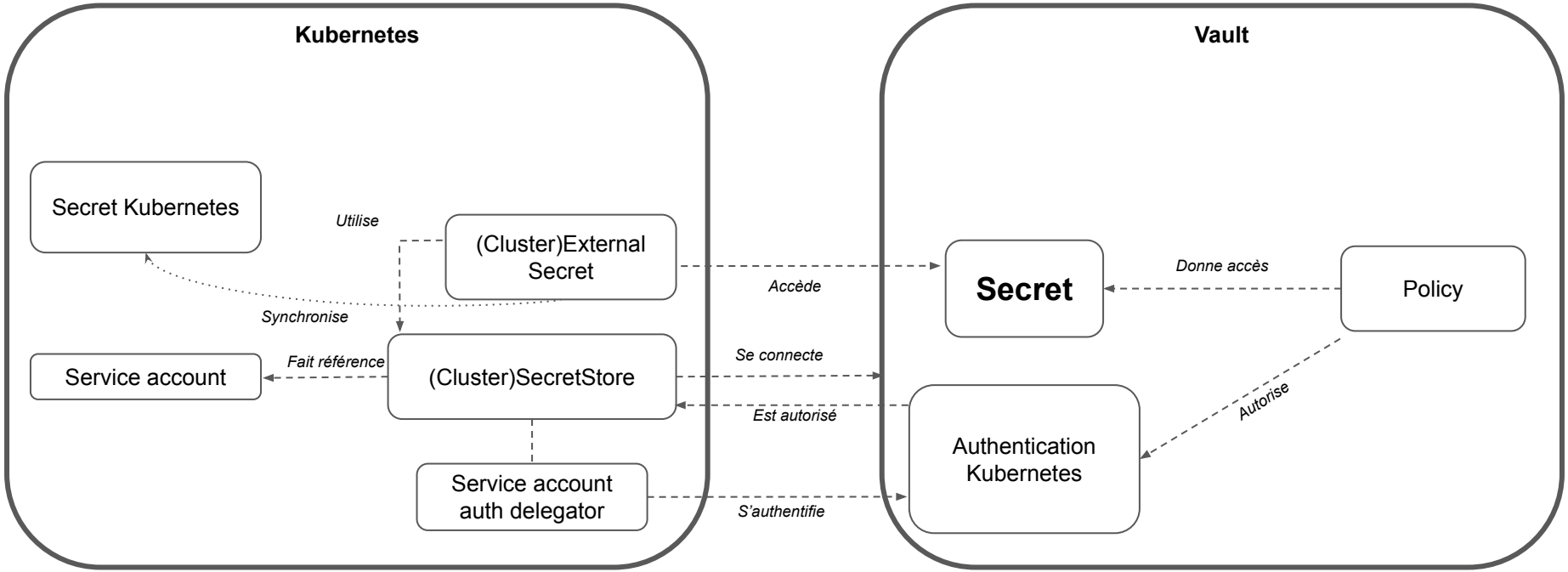

Kubernetes
Vault
Secret Kubernetes
Utilise
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Accède
Se connecte
Est autorisé
S'authentifie
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes

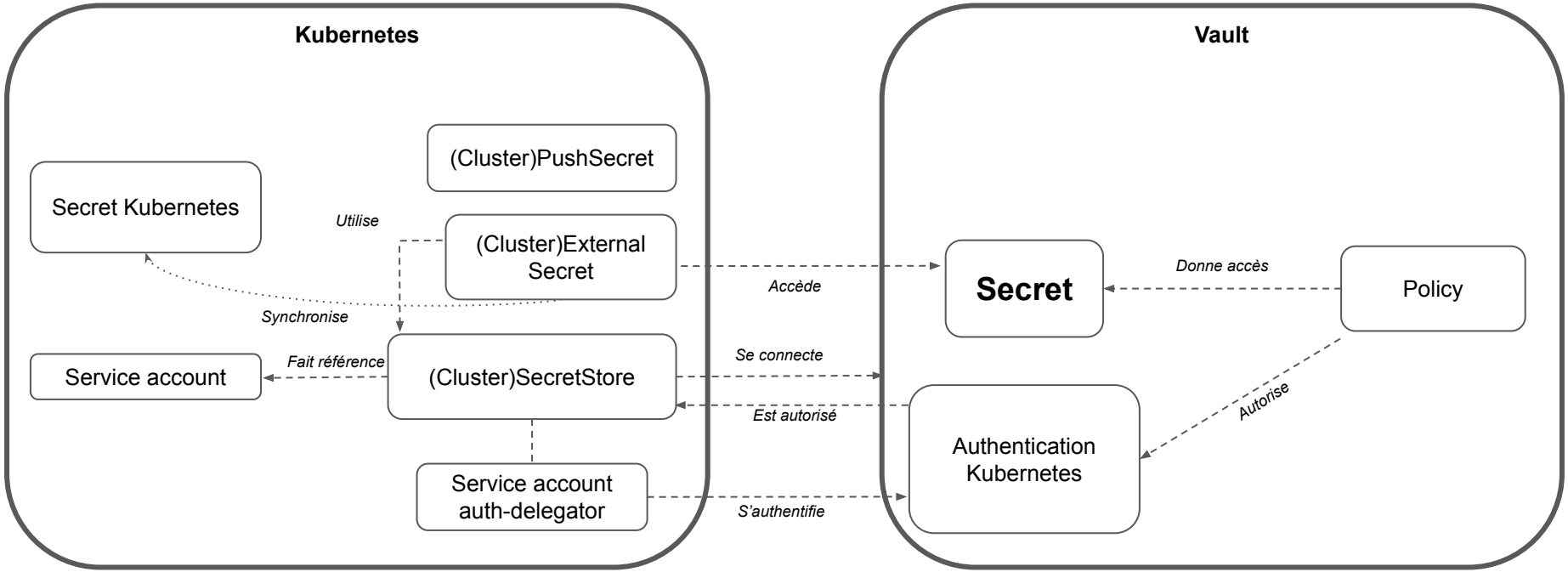
L'ExternalSecret

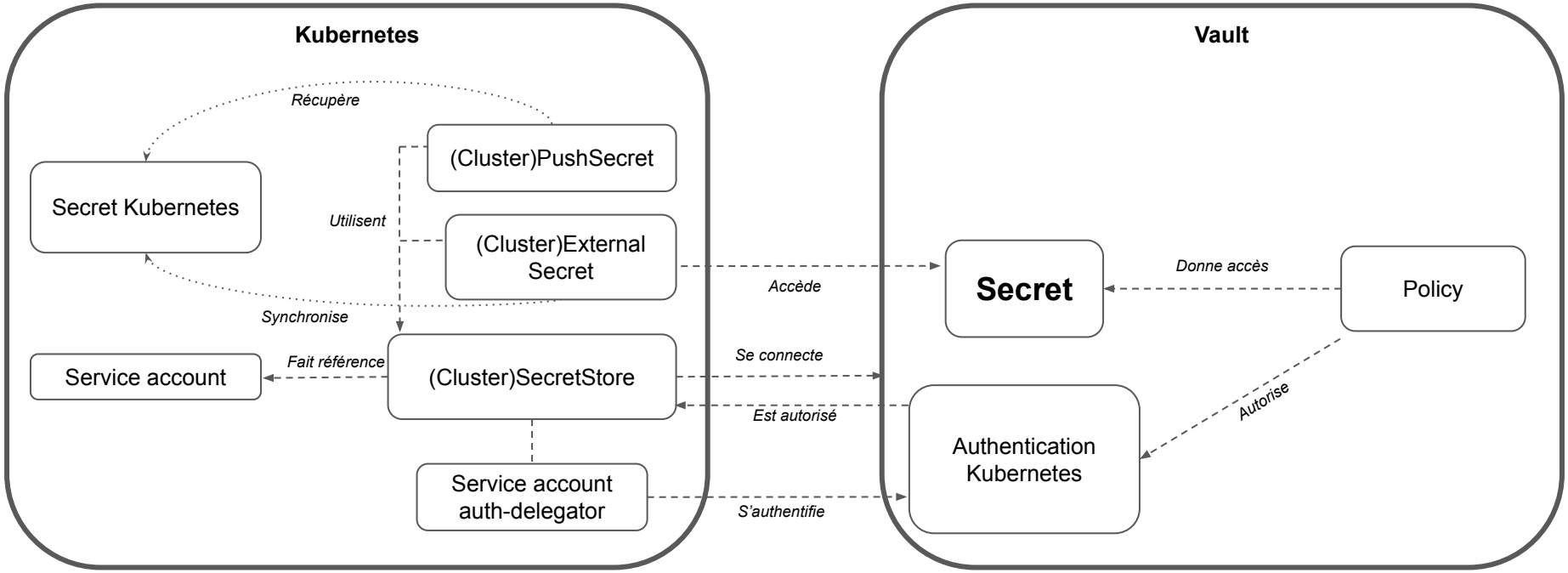
Quatrième étape : on déclare un secret !

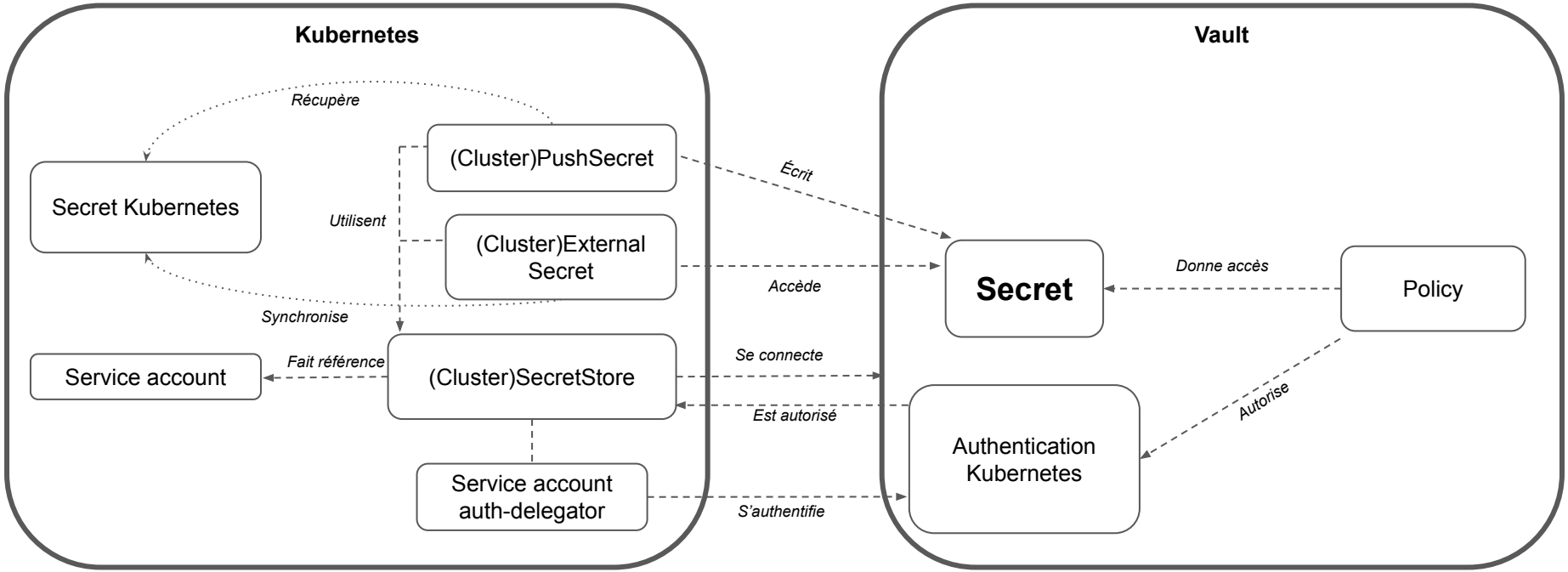
```yaml
apiVersion: external-secrets.io/v1
kind: ExternalSecret
metadata:
 name: external-secret
Spec:
 deletionPolicy: [Delete, Retains, Merge]
 updatePolicy: Replace
 refreshInterval: "15s"
 secretStoreRef:
   name: [Nom du (Cluster)SecretStore]
   kind: [SecretStore,ClusterSecretStore]
 target:
   name: [Nom du secret Kubernetes]
 dataFrom:
 - extract:
     key: [Path du secret à lire]
```

DEEZER


Kubernetes
Vault
Secret Kubernetes
Utilise
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth delegator
Accède
Se connecte
Est autorisé
S'authentifie
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes


Kubernetes
Vault
(Cluster)PushSecret
Secret Kubernetes
Utilise
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Accède
Se connecte
Est autorisé
S'authentifie
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes


Kubernetes
Vault
Récupère
(Cluster)PushSecret
Secret Kubernetes
Utilisent
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Accède
Se connecte
Est autorisé
S'authentifie
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes

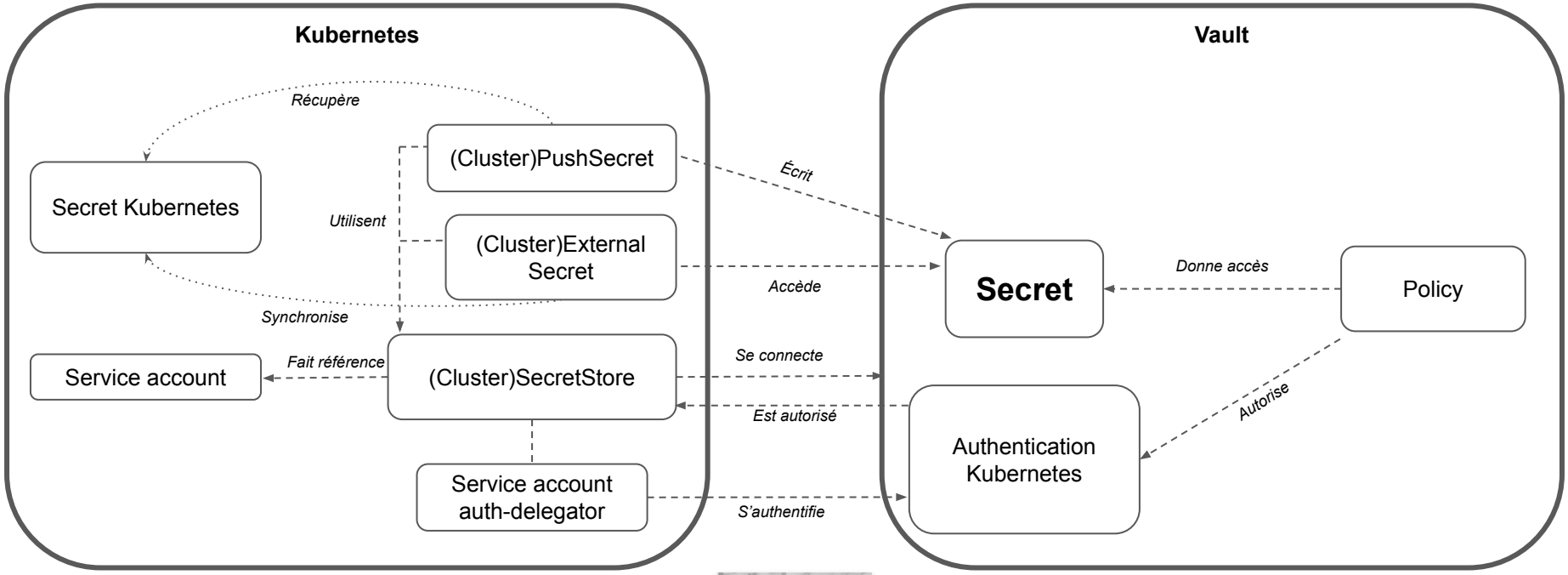

Kubernetes
Vault
Récupère
(Cluster)PushSecret
Secret Kubernetes
Utilisent
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Écrit
Accède
Se connecte
Est autorisé
S'authentifie
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes

Le PushSecret

Cinquième (et facultative) étape : on écrit un secret !

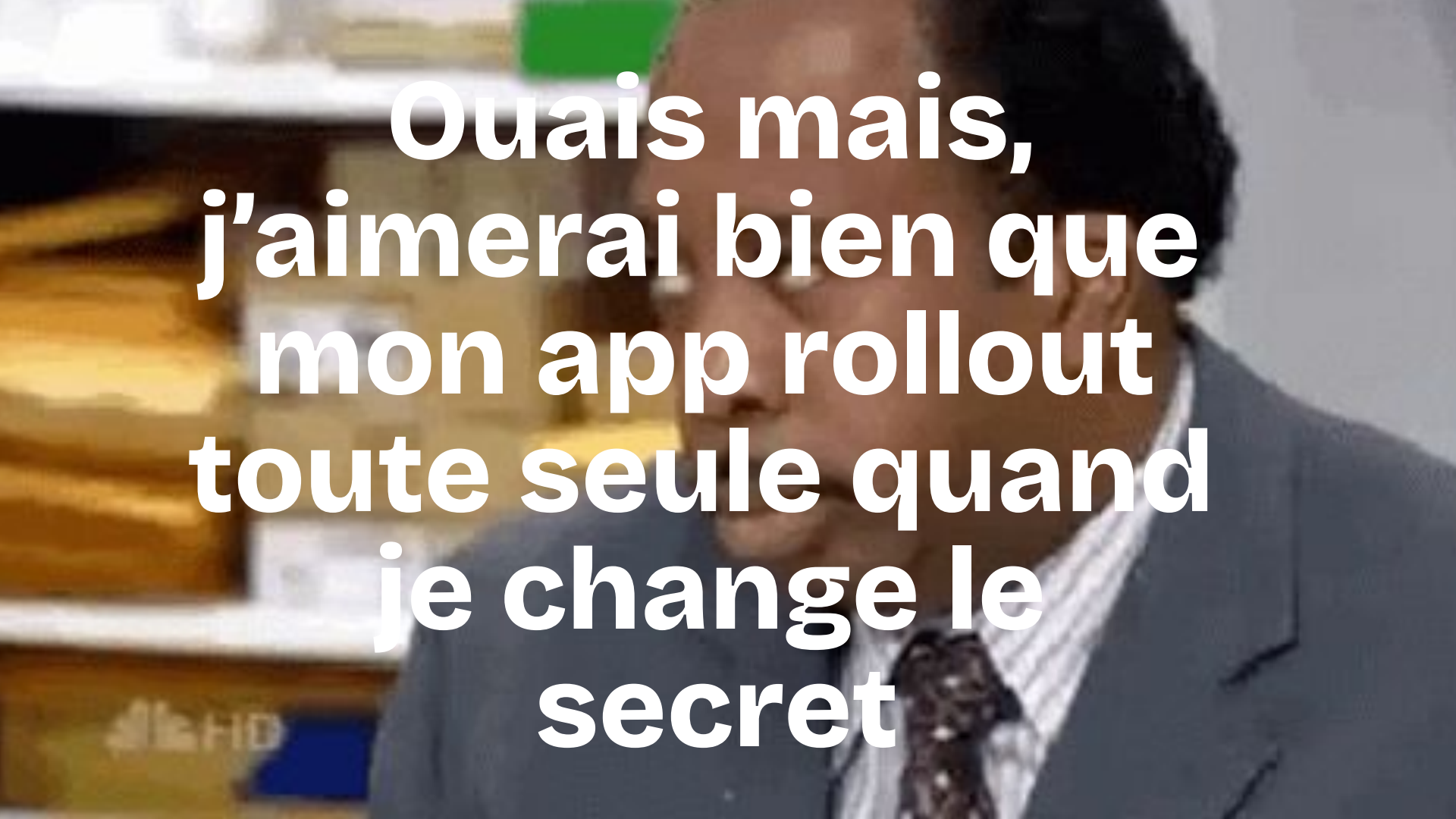
```yaml
apiVersion: external-secrets.io/v1alpha1
kind: PushSecret
metadata:
 name: pushsecret
spec:
 data:
   - conversionStrategy: [None, ReverseUnicode]
     match:
       remoteRef:
         remoteKey: [le path du secret à écrire]
 deletionPolicy: [Delete, Retains, Merge]
 refreshInterval: 10s
 secretStoreRefs:
   - kind: [SecretStore,ClusterSecretStore]
     name: [nom du (Cluster)SecretStore]
 selector:
   secret:
     name: [Le secret à lire]
 updatePolicy: Replace
```

DEEZER


Kubernetes
Récupère
(Cluster)PushSecret
Secret Kubernetes
Utilisent
(Cluster)External Secret
Synchronise
Service account
Fait référence
(Cluster)SecretStore
Service account auth-delegator
Écrit
Accède
Se connecte
Est autorisé
S'authentifie
Vault
Secret
Donne accès
Policy
Autorise
Authentication Kubernetes


ALL PART OF THE PL

DEEZER


Ouais mais,
j'aimerai bien que
mon app rollout
toute seule quand
je change le
secret

Le Reloader

Le rollout automatique de vos déploiements

- Un controller qui redémarre automatiquement vos déploiements lors de changements de secrets et/ou configmap
- Se paramètre directement via des annotations
- Possède un mode automatique
- Fonctionnalités de recherches et plusieurs stratégies de rollout

Petite ouverture...

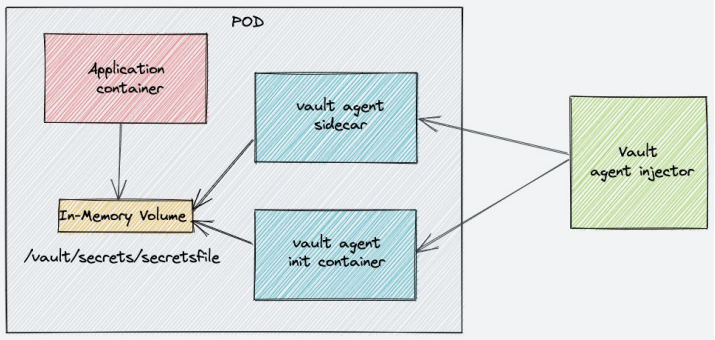
DEEZER


L'opérateur ne renforce pas la sécurité de vos secrets

Le vault/openbao agent injector

La, on est vraiment secure...

- Agent sidecar appelé par des annotations dans vos déploiements/pods
- Injecte directement vos secrets à l'intérieur des pods
- Supporte le *consul-template*
- Forte dépendance à la disponibilité de Vault **: risque de bloquer vos déploiements !**



Source devopscube.com

CEPENDANT

DEEZER


J’ai menti.
LIAR!

Gérer ses secrets Kube avec l'External Secrets Operator : parce que je le vaux bien

DEEZER

Merci !

L'accès aux slides !

Pour me faire un retour !

Si le talk ou le plot twist vous a déçu

LePotiBlagueur

@LePotiBlagueur

https://www.linkedin.com/in/alexis-fala/

DEEZER