Using secrets from Substrate Vault

Document Level Classification

100

Starting from August 2024, it is mandatory that all new Kubernetes clusters use External Secrets Operator (ESO). We strongly recommend using ESO, but the Vault Injector sidecar will still remain available with limited support and bug fixes will be provided if needed to maintain core functionality.

Reference Using External Secrets Operator (ESO) in epic-app to inject secrets for documentation on using ESO.

Introduction

When developing and deploying services on the Cloud Developer Platform in a Substrate account, your service may need to access other backend or downstream systems. In order to access these systems, sensitive information is necessary to authenticate with them. This could be a password, token, API key, etc. In order to use this sensitive information in a service, it must be done securely. This document will cover the different patterns for using secrets in Substrate.

Which Pattern Should I Use for Secrets in Kubernetes?

Using External Secrets Operator (ESO) in epic-app to inject secrets

This is the current preferred, fully supported, and up to date pattern for using secrets in Substrate. It allows you to share secrets with your Substrate applications utilizing native Kubernetes secrets easily without the need of sidecars or other extra tooling. It is now mandatory that all new Kubernetes (EKS) clusters use ESO, thus all new Substrate accounts come provisioned with ESO.

Reference Using External Secrets Operator (ESO) in epic-app to inject secrets for documentation on using ESO.

Using Substrate Vault with vault-injector to Inject Secrets

This is the old method of using secrets in Substrate and no longer the preferred method. Starting from August 2024, the vault injector sidecar will still remain available but with limited support and bug fixes will be provided if needed to maintain core functionality. If you are still utilizing this pattern it is advised that you utilize the ESO pattern instead going forward.

Reference Using Substrate Vault with vault-injector to Inject Secrets or documentation on using vault-injector.

How do I use secrets outside of Kubernetes?

For secret usage outside of Kubernetes, we have account level secret paths available. While old secret paths managed by legacy vault groups are still in place, these are deprecated and it is strongly recommended you move to the new model.

Reference Using Vault Secrets in Non-K8s Workloads for more information

Vault 403 error

Pretty often, when you're using Vault you may face error "403 Permission denied" and may think that something wrong with your permissions...maybe yes, maybe no.

There are two reasons why 403 may appear:

- You're using wrong path in Vault and it's "subset" - you're trying to reach non existing secret
- You're really have no permissions

Because of security reasons Vault does not reply you with "404 - Not found" when path is wrong, because it may lead to partial information

exposure, when its possible to get which secrets are exist and which are not.

403 when you're using wrong path

First of all you have to check if path you're trying to reach is available:

- go to https://vault.substrate.on.epicgames.com
- find proper secret engine name (usually its AWS account name or "secret/")
- then find your secret by path

If you did not found secret in a path, you found a root cause.

If secret exist, check what path you're trying to use, pretty often users forgetting "/data/" in path, if it used by API.
Like your path is `"secret/eos/anti-cheat/use1a/dev/build/uas_secrets"`, but API call have to be done with "/data/" in path, after secret engine name (".../secret/..." or AWS account name), for example "secret/data/eos/anti-cheat/use1a/dev/build/uas_secrets"

403 when you have no permissions

ESO

Pretty often its happening when you trying to reach Vault secret from inside EKS and trying to reach secrets from another namespace:

1. Check that path you're trying to reach really exist
2. Check that your app running in namespace you're trying to reach
   For example your app running in namespace "datarouter-dev" and you're trying to reach secret from "streams-router-dev" - "fdbe-dev/fdbe-dev-analytics/streams-router-dev/streams-router-dev"

TFE

Similar to GHE:

1. Ensure path is exists

2. Check if policy exists in Vault, like [this]

Policy allows to read any code pushed to TFE org `pulse-observability` can read secret from `"secret/+/coreonline/pulse-observability/*"`

```
type: tfe
tfe_org: pulse-observability
policy: |-

# ...

    path "secret/+/coreonline/pulse-observability/*" {
        capabilities = ["read", "list"]
    }
```

GHE

If you're trying to reach some Vault secret from GHE and failing with 403:

1. Ensure path is existing

If it is, check if proper policy exists in Vault, for example.
Following policy file will allow GHE pipelines to grab secret `"secret/+/substrate/..../deployer"` if its running in "online-common" - https://github.ol.epicgames.net/online-common/

```
type: github
github_org: online-common
policy: |-

  path "secret/+/substrate/k8s/use1a/prod/build/eeef/eeef-live-online-i
    capabilities = ["read", "list"]
  }
```

CLI

Sometimes it may occur that you're trying to set value for non-EKS secrets and failing:

```
> vault kv put bbaf-dev/testdom foo=bar
Error writing data to bbaf-dev/data/testdom: Error making API request.

URL: PUT https://vault.substrate.on.epicgames.com/v1/bbaf-dev/data/test
Code: 403. Errors:

* 1 error occurred:
        * permission denied
```

Do not forget "GENERIC", use `bbaf-dev/GENERIC/testdom`

- Using External Secrets Operator (ESO) in epic-app to inject secrets
- Using Vault Secrets in Non-K8s Workloads
- Managing Substrate Vault Secrets with Castle (CLI)
- Using Substrate Vault Secrets with Codefresh
- Using Substrate Vault Secrets with Github Actions
- Using Substrate Vault with vault-injector to Inject Secrets
- Vault Access for K8S Service Accounts (SA)

Page Information:
Page ID: 81068456
Space: Cloud Developer Platform
Downloaded: 2025-07-12 04:07:20