HashiCorp Vault Secrets management

CONCEPTS

The CLI is a thin wrapper over the HTTP API: everything lives at a path, and read/write/list/delete work on any of them.

Auth method userpass, token, oidc...
↓ issues
Token carries policies
↓ checked by
Policy path capabilities
↓ allows
Secrets engine kv, transit, database...

Dev mode is in-memory and insecure. Never use it for real secrets.

CONNECT

`export VAULT_ADDR=<url>`
Set the server address

`export VAULT_TOKEN=<token>`
Authenticate via environment

`vault status`
Show seal and HA state

`-format=json`
Machine-readable output (also yaml)

DEV SERVER

`vault server -dev`
Run in-memory, unsealed

`vault server -dev -dev-root-token-id=<id>`
Dev server with a known root token

`vault server -config=<file>`
Run with a config file

LOGIN AND TOKENS

`vault login`
Log in with a token (prompts)

`vault login -method=<type> <key>=<value>`
Log in with another auth method

LOGIN AND TOKENS (cont.)

`vault token lookup`
Show the current token

`vault token create -policy=<name>`
Create a token with a policy

KV SECRETS

`vault kv put -mount=secret <path> <key>=<value>`
Write a secret

`vault kv get -mount=secret <path>`
Read a secret

`vault kv get -mount=secret -field=<key> <path>`
Print one field only

`vault kv list -mount=secret`
List paths under a mount

RAW API PATHS

`vault read <path>`
GET any path

`vault write <path> <key>=<value>`
POST/PUT data to a path

ENGINES AND AUTH METHODS

`vault secrets enable -path=<path> <type>`
Enable an engine

`vault auth enable <type>`
Enable an auth method

POLICIES

`vault policy write <name> <file>.hcl`
Upload a policy from a file

OPERATE AND SEAL

`vault operator init`
Initialize a new server

`vault operator unseal`
Enter one unseal key (prompts)