# What is Secure Key Vault?

**Key Vault refers to a service that protects encryption keys and secrets like certificates, connection strings, and passwords.**

* **Secrets Management**
* **Key Management**
* **Certificate Management**

# Why Key Vault?

### Centralize application secrets: You can control the distribution of application secrets by storing them centrally in Key Vault. Secrets are even less likely to be mistakenly revealed using Key Vault. Application developers no longer need to store security information in their applications when using Key Vault. The need to make security information part of the code is removed by not having to store security information in applications. An program might, for example, need to connect to a database, you can securely save the connection string in Key Vault instead of storing it in the app's code.

### Securely store secrets and keys: Access to a key vault requires proper authentication and authorization before a caller (user or application) can get access. Authentication establishes the identity of the caller, while authorization determines the operations that they are allowed to perform.

### Monitor access and use: Once you have created a couple of Key Vaults, you will want to monitor how and when your keys and secrets are being accessed. You can monitor activity by enabling logging for your vaults.

# Key Terms

* Secret Engine: A Secret Engine is used to manage the secrets.
* Auth Method: An Auth Method is used to authenticate a Client with Vault.
* Policy: A Policy provides RBAC to Clients in order to retrieve secrets.
* Audit Device: An Audit Device is a component that logs all request and response to the Vault.
* Storage Backend: A Storage Backend provides durable storage for vault.

# Popular Key Vault Services

Azure Key Vault is a widely used key vault service. Useful readings:

* [About](https://docs.microsoft.com/en-us/azure/key-vault/general/overview)
* [Concepts](https://docs.microsoft.com/en-us/azure/key-vault/general/basic-concepts)

Some alternatives of Azure Key Vault:

* [**HashiCorp Vault**](https://www.hashicorp.com/products/vault)
* [Egnyte](https://www.egnyte.com/resources/infographics/secure-file-sharing)
* 1Password
* AWS Certificate Manager

# Pricing

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Pricing** | | **Remarks** |
| Azure Key Vault | 0.03 USD/10k transactions | |  |
| HashiCorp Vault | **Criterion** | **Price** | Self-Hosted and Managed version is free. So I will choose this one. |
| Open Source | Free |
| Cloud | 0.03 USD/ 3 hours |
| Enterprize | Custom |
| Egnyte | 20 USD per employee/Month | |  |
| Amazon Certificate Manager | Complex, Go to [link](https://aws.amazon.com/certificate-manager/pricing/) | |  |

HashiCorp

#### How much does HashiCorp Vault cost?

HashiCorp Vault is a free and open source product with an enterprise offering. The enterprise platform includes disaster recovery, namespaces, and monitoring, as well as features for scale and governance. You can see the full breakdown of features on the [Hashicorp Vault pricing](https://www.hashicorp.com/products/vault/pricing/) page.

#### How do I set up Hashicorp Vault?

Here are the steps to install and configure HashiCorp Vault, as laid out in their [deployment guide](https://learn.hashicorp.com/tutorials/vault/deployment-guide):

1. Download Vault
2. Install Vault
3. Configure systemd
4. Configure Consul
5. Configure Vault
6. Start Vault

Detailed Info of these steps:

* [CLI](https://learn.hashicorp.com/collections/vault/getting-started)
* [Cloud](https://learn.hashicorp.com/collections/vault/cloud)
* [UI](https://learn.hashicorp.com/collections/vault/getting-started-ui)

#### CLI steps

1. Download and Install vault :
   * Download binary 🡺 [Hashicorp dowloads](https://www.vaultproject.io/downloads%20)
   * Install using chocolatey 🡺 Run this command as an administrator :

$ choco install vault

1. Starting the server
   * Dev server 🡺 We will be see the demo using dev server. It can’t be deployed in production as it is pre-configured. To run a non-dev server go to : [deployment guide](https://learn.hashicorp.com/tutorials/vault/deployment-guide)

Run: $ vault server –dev -dev-root-token-id=root

* + Client side 🡺
    - Set Vault Address and store token and unseal key.

$ set VAULT\_ADDR='http://127.0.0.1:8200'

* + - Now set the token

$ set VAULT\_TOKEN=" Stored Token"

* + - To check the server:

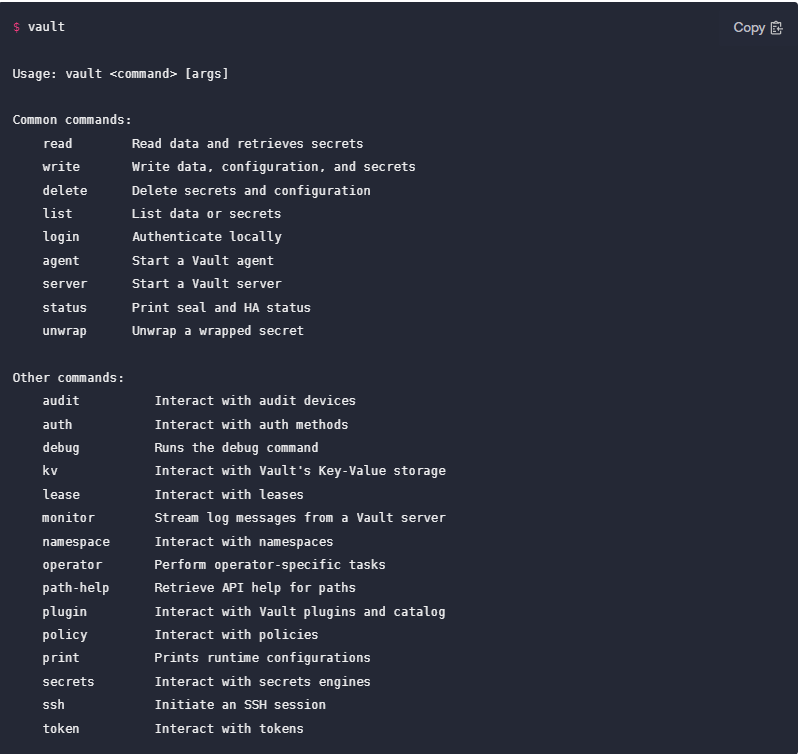
$ vault status

1. Create first secret
   * Create a secret named hello:

$ vault kv put secret/hello foo=world

* + Read the secret named hello

$ vault kv get secret/hello



#### How do I access the Vault UI?

You can access the [Hashicorp Vault web UI](https://learn.hashicorp.com/tutorials/vault/getting-started-ui) by starting the Vault server in dev mode with vault server -dev and navigating to <http://127.0.0.1:8200/ui> in your browser. Check out their [documentation](https://learn.hashicorp.com/tutorials/vault/getting-started-ui) for more tips on getting started.

#### Is HashiCorp Vault secure?

Using HashiCorp Vault for secrets management is certainly more secure than placing plaintext secrets in your configurations. In accordance with industry best practices for data encryption, HashiCorp Vault utilizes both TLS for data in transit and AES 256-bit encryption for data at rest.