



Consuming Secrets from HCP Vault

July 2022

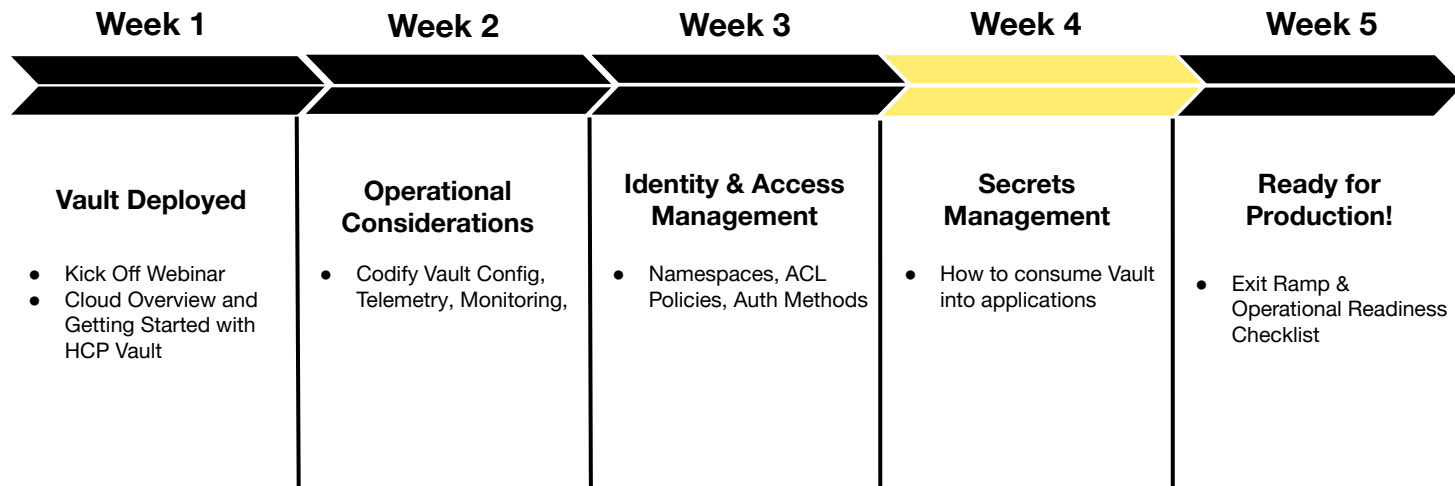
Copyright © 2021 HashiCorp



Agenda

1. Secure Introduction
2. Consuming Secrets
3. Third Party Integrations
4. Kubernetes
5. Vault - Agent injector

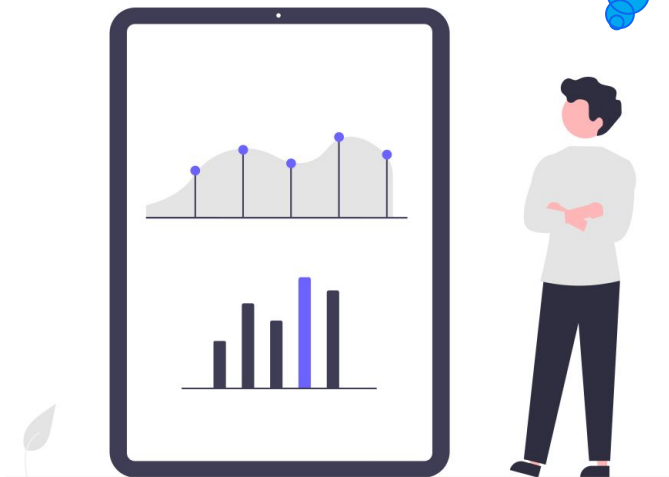
HCP Vault Path to Production



Poll time



This is
different



HCP Vault - Week 4 - Milestone Check In

1. In considering your HCP Vault cluster, please mark the items which are TRUE for your implementation.. (Multiple Choice) *

- ☐ HCP Organization has been created
- ☐ Road map for use case adoption has been created.
- ☐ HCP Vault cluster has been created
- ☐ HCP Vault is live in Production
- ☐ First use case onboarded and consuming secrets stored in Vault.
- ☐ You can successfully connect to HCP Vault cluster,
- ☐ There is at least one automation for Vault management in place.
- ☐ Basic telemetry & monitoring in place.
- ☐ Performance replication in place (Plus Tier customers)

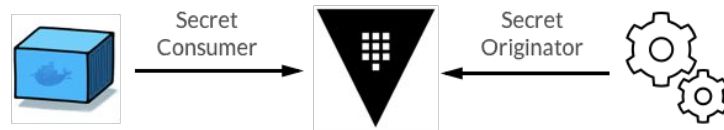
Secure Introduction

Secret Originator and Consumer



If you can securely get the first secret from originator to a consumer, all subsequent secrets transmitted between them can be authenticated with the trust established by the successful distribution of the first secret.

Tokens are the core method for authentication within Vault. Every secret consumer (client) must acquire a valid token



Methods for Secure Introduction



Platform Integration

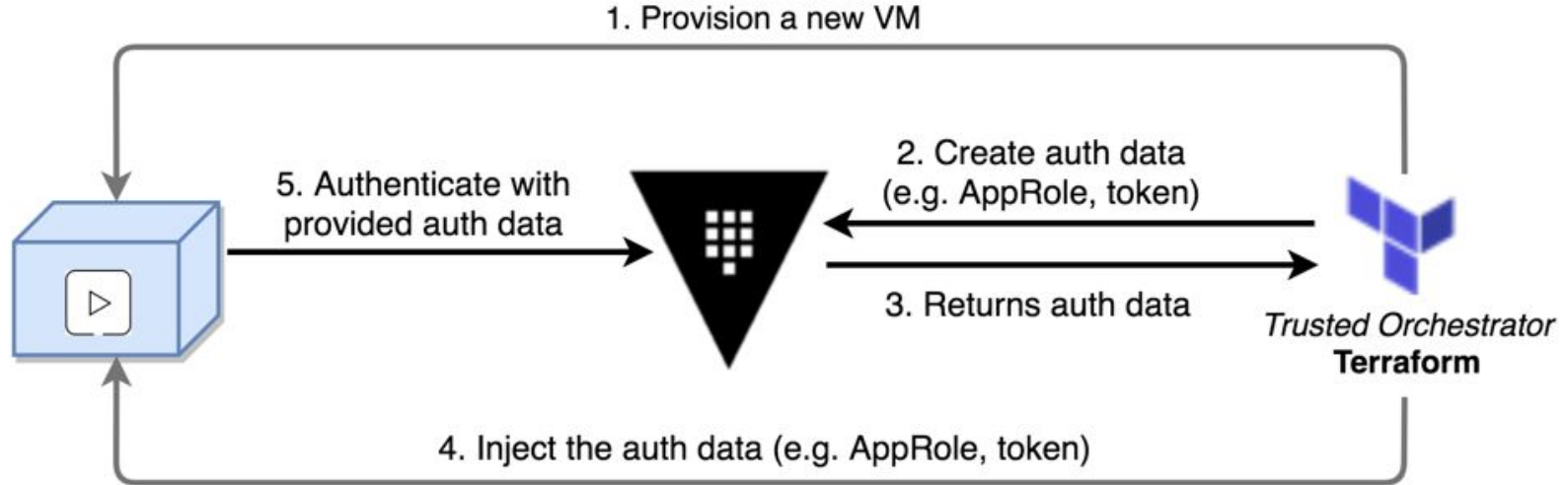
Vault establishes a trust with your trusted platforms (AWS, Azure, GCP) to use the identifier of resources (virtual instances, containers, etc) to authenticate and provide authorization to a Vault token.

Trusted Orchestrator

Your existing trusted orchestrator (Terraform, Kubernetes, Chef) has already been authenticated to Vault with privileged permissions. During deployment of applications, orchestrator injects necessary credentials to authenticate to Vault and retrieve a Vault token.

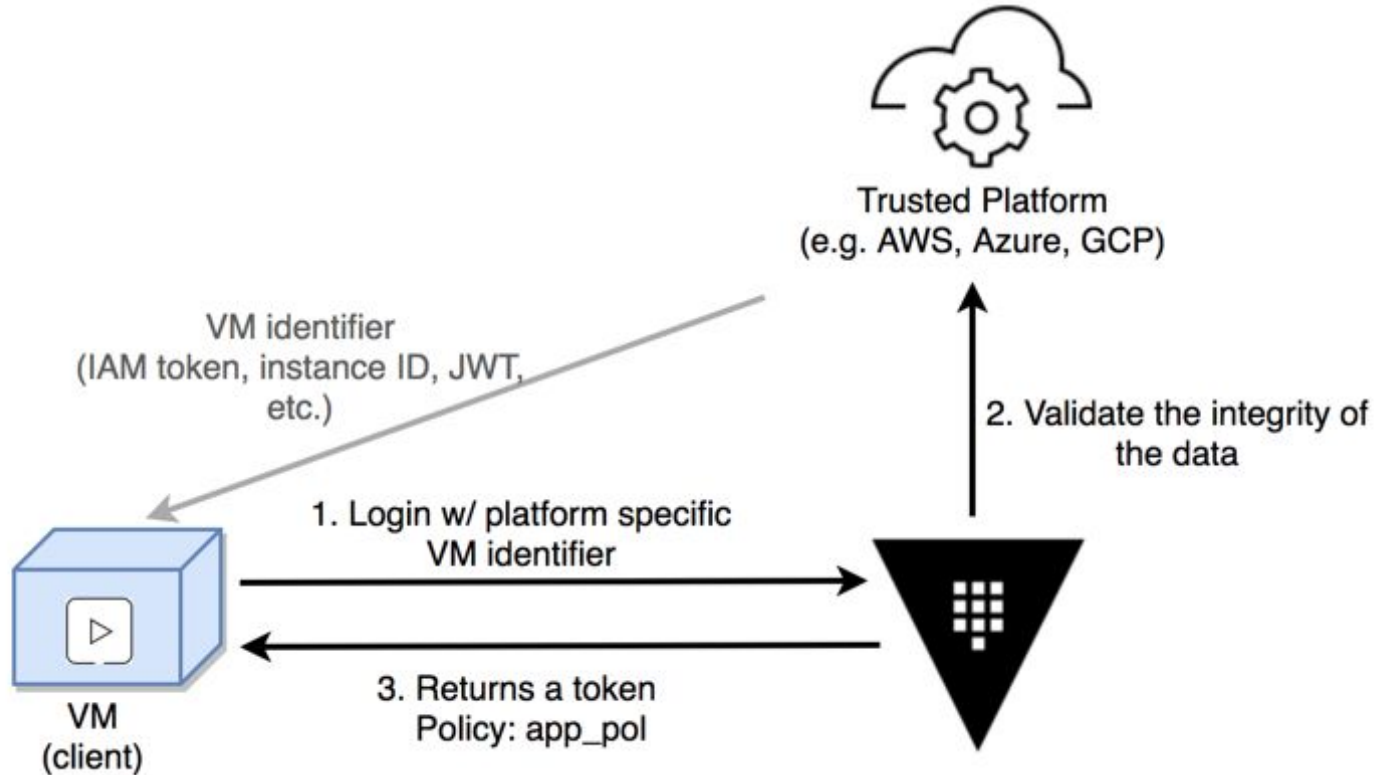
Trusted Orchestrator

Secure introduction in a VM environment



Platform Integration

Secure introduction in a cloud environment



Automating Introduction

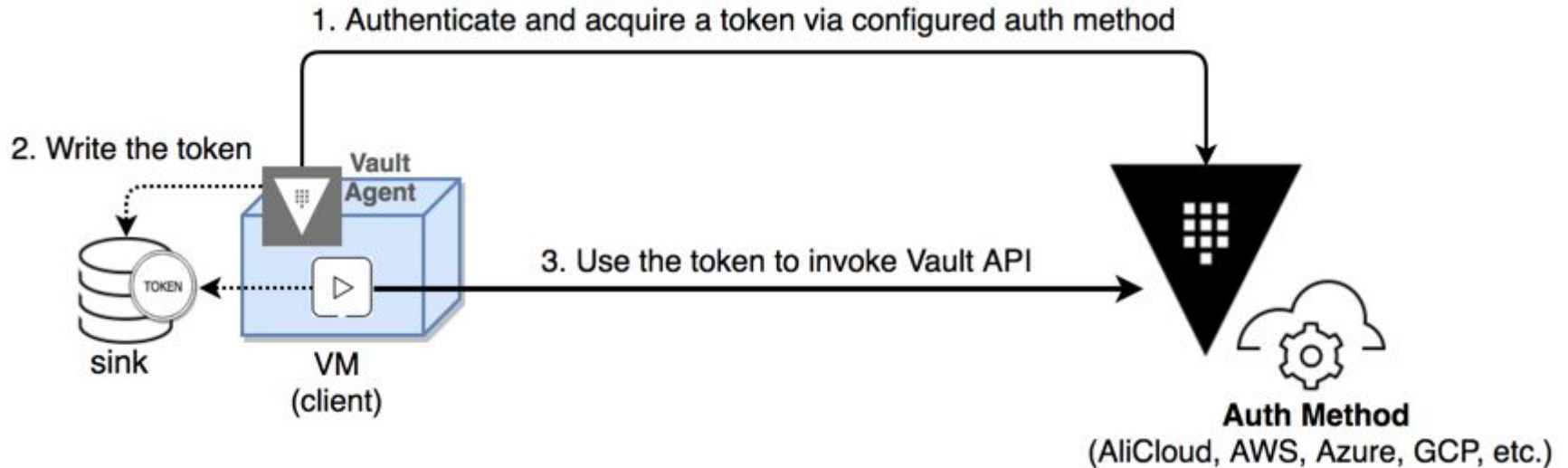


[Vault Agent](#) is a client daemon which automates the client login workflow and will manage the lifecycle for the Vault token. Compatible with both platform integration and trusted orchestrator secure introduction methods.

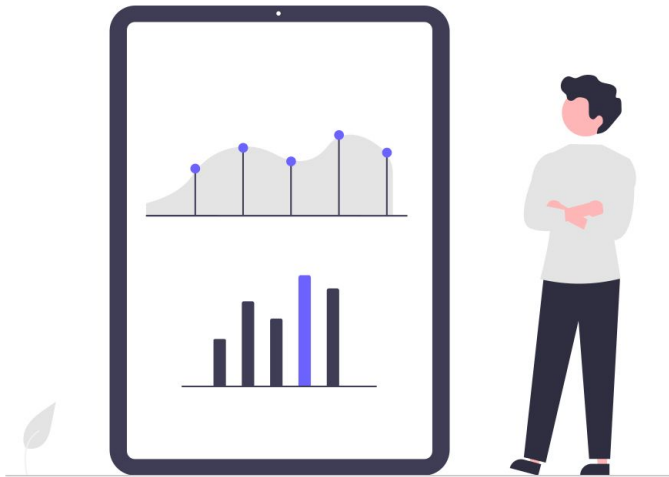
Vault Agent is included as part of the Vault binary and can be run by starting the binary in agent mode `vault agent -config=<config-file>`. Once authentication has completed a Vault token will be written to file sink.

Automate Introduction

Vault Agent



Poll time



Which method do you plan to use to securely introduce your clients to Vault?

Consuming Secrets



Patterns to Consume Secrets

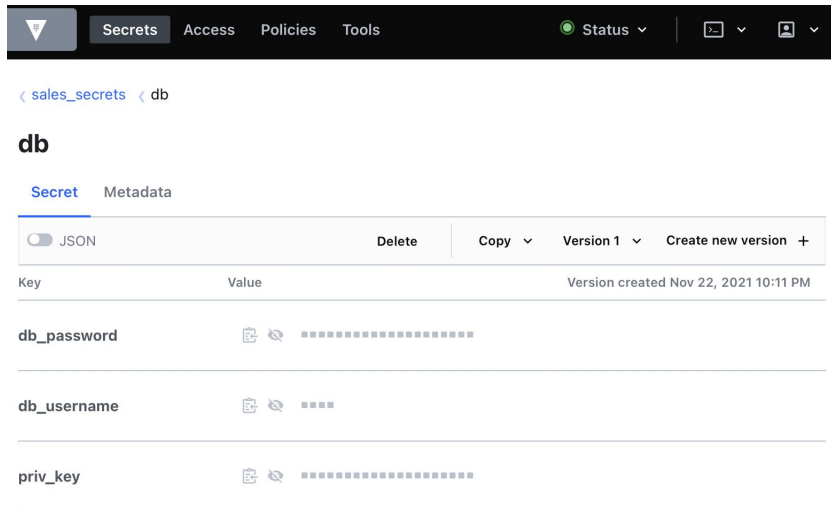
- UI
- CLI
- HTTP API
- Templating
- Environment Variables
- Client Libraries

Web UI



Vault UI enables users to populate and consume secrets from Vault without the need to learn about Vault CLI commands or API.

This approach works well when a user is consuming secrets however it can be limiting when needing to consume a large number of secrets or if secrets are being consumed as part of an application configuration.





CLI

Similar to UI, Vault CLI is most likely being leveraged by users to consume secrets manually

```

TERMINAL
> vault kv get sales_secrets/db

===== Metadata =====
Key                Value
---                -
created_time       2021-11-23T03:11:49.056626Z
custom_metadata    <nil>
deletion_time      n/a
destroyed          false
version            1

===== Data =====
Key                Value
---                -
db_password        jsobdgjubsdjgbsdjiogbnsdjogsbiosdbng
db_username        root
priv_key           djbsdougjbnsdojignsdoigsd

```




HTTP API

The Vault HTTP API gives you full access to Vault via HTTP. Every aspect of Vault can be controlled via this API.

```

> curl -X 'GET' \
  'http://127.0.0.1:8200/v1/sales_secrets/data/db' \
  -H 'accept: */*' \
  -H 'X-Vault-Token: s.US1nHb9AUa06r4QXA5XHnVPZ'

{
  "request_id": "ebc2ed73-e7d6-de3b-88bb-a52ee11143cd",
  "lease_id": "",
  "renewable": false,
  "lease_duration": 0,
  "data": {
    "data": {
      "db_password": "jsobdgjubsdjgbsdjiogbnsdjogsbiosdbng",
      "db_username": "root",
      "priv_key": "djbsdougjbnsdojignsdoigsd"
    },
    "metadata": {
      "created_time": "2021-11-23T03:11:49.056626Z",
      "custom_metadata": null,
      "deletion_time": "",
      "destroyed": false,

```



HTTP API Explorer

<VAULT_ADDR>/ui/vault/api-explorer

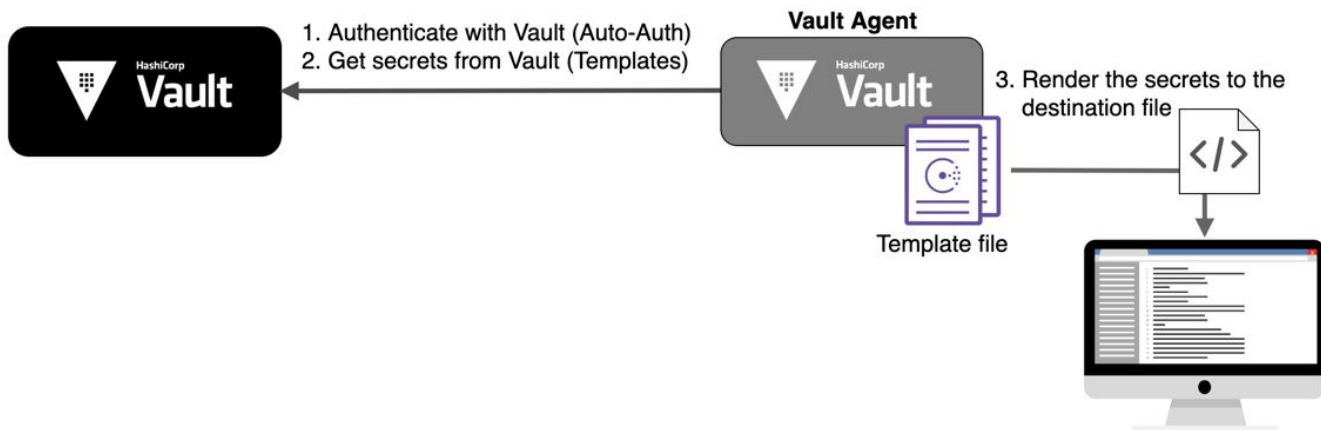
The screenshot displays the Vault HTTP API Explorer interface. At the top, there is a navigation bar with tabs for 'Secrets', 'Access', 'Policies', and 'Tools'. To the right of these tabs is a 'Status' indicator with a green dot and a dropdown arrow, followed by two user profile icons. Below the navigation bar, the interface lists several API endpoints, each with a colored header indicating the HTTP method: GET (blue), POST (green), and DELETE (red). The endpoints are as follows:

- DELETE** (pink background): Deletes the secret at the specified location.
- GET** (blue background): `/sales_secrets/config`. Read the backend level settings.
- POST** (green background): `/sales_secrets/config`. Configure backend level settings that are applied to every key in the key-value store.
- GET** (blue background): `/sales_secrets/data/{path}`. Write, Patch, Read, and Delete data in the Key-Value Store.
- POST** (green background): `/sales_secrets/data/{path}`. Write, Patch, Read, and Delete data in the Key-Value Store.
- DELETE** (pink background): `/sales_secrets/data/{path}`. Write, Patch, Read, and Delete data in the Key-Value Store.

Vault Agent Templating



Vault Agent can fully automate the last mile and securely authenticate and retrieve secrets from Vault. When configured with auto-auth, templating can be configured to retrieve a secret for which the resource has authorization to and template that file to a sink. Template files are written using the Consul Template markup language.





Vault Agent Templating

Example Template

```
> cat customer.tpl
```

```
{{ with secret "secret/data/customers/acme" }}  
Organization: {{ .Data.data.organization }}  
ID: {{ .Data.data.customer_id }}  
Contact: {{ .Data.data.contact_email }}  
{{ end }}
```

```
> cat customer.txt
```

```
Organization: ACME Inc.  
ID: ABXX2398YZPIE7391  
Contact: james@acme.com
```



envconsul

A subprocess which dynamically populates environment variables from secrets read from Vault. Your applications then read those environment variables.

```
#!/usr/bin/env bash
```

```
cat <<EOT
```

```
My connection info is:
```

```
username: "${DATABASE_CREDS_READONLY_USERNAME}"
```

```
password: "${DATABASE_CREDS_READONLY_PASSWORD}"
```

```
database: "my-app"
```

```
EOT
```

```
$ VAULT_TOKEN=<token> envconsul -uppercase -secret
```

```
database/creds/readonly ./app.sh
```

```
My connection info is:
```

```
username: "v-token-readonly-ww1tq33s7z5uprpaxy68-1527631219"
```

```
password: "A1a-u54wut0v605qwz95"
```

```
database: "my-app"
```



Go Client Library

[Reference Documentation](#)

```
CODE EDITOR

// get secret
secret, err := client.Logical().Read("kv-v2/data/creds")
if err != nil {
    return "", fmt.Errorf("unable to read secret:%w", err)
}

data, ok := secret.Data["data"].(map[string]interface{})
if !ok {
    return "", fmt.Errorf("data type assertion failed:%T
%#v", secret.Data["data"], secret.Data["data"])
}

// data map can contain more than one key-value pair,
// in this case we're just grabbing one of them
key := "password"
value, ok := data[key].(string)
if !ok {
    return "", fmt.Errorf("value type assertion failed:%T
%#v", data[key], data[key])
}
```

Poll time



Which are the top patterns
for consuming secrets in
your organization?

Third-Party Integrations

Ecosystem



A broad ecosystem of frameworks and tooling have been created to help support integrations between third party tools and services. These frameworks and tooling can ease the burden on your end users to integrate and consume secrets from Vault.

Considerations



Support

HashiCorp is unable to provide technical support for third party frameworks and tooling. We can support you from the Vault side however any issues with the framework or tooling will need to be raised with the creator of those frameworks or tooling.

Enterprise Capabilities

We have established partnerships with a number of partners who have created tooling and framework that support enterprise capabilities (ex. namespaces). If the tooling or framework that you are attempting to use does not support enterprise capabilities, please have them reach out to us if they are interested in supporting enterprise capabilities.



Java Applications

Spring Cloud Vault client
libraries

[Spring Cloud Vault](#)
[Java Application Demo](#)

```
class Person < ActiveRecord::Base
  include Vault::EncryptedModel
  vault_attribute :ssn
end

class AddEncryptedSSNToPerson < ActiveRecord::Migration
  add_column :persons, :ssn_encrypted, :string
end

person = Person.new
person.ssn = "123-45-6789"
person.save #=> true
person.ssn_encrypted #=> "vault:v0:EE3EV8P5hyo9h..."
```



Vault C# Client

Integrate with your .Net
Applications

[Using HashiCorp Vault C#
Client with .NET Core](#)

```
CODE EDITOR

public VaultConfigurationProvider(VaultOptions config)
{
    _config = config;
    var vaultClientSettings = new VaultClientSettings(
        _config.Address,
        new AppRoleAuthMethodInfo(_config.Role,
                                   _config.Secret)
    );
    _client = new VaultClient(vaultClientSettings);
}

public class VaultOptions
{
    public string Address { get; set; }
    public string Role { get; set; }
    public string Secret { get; set; }
    public string MountPath { get; set; }
    public string SecretType { get; set; }
}
```



Pipeline Integration

Github Actions

[Vault Secrets · Actions ·
GitHub Marketplace ·
GitHub](#)

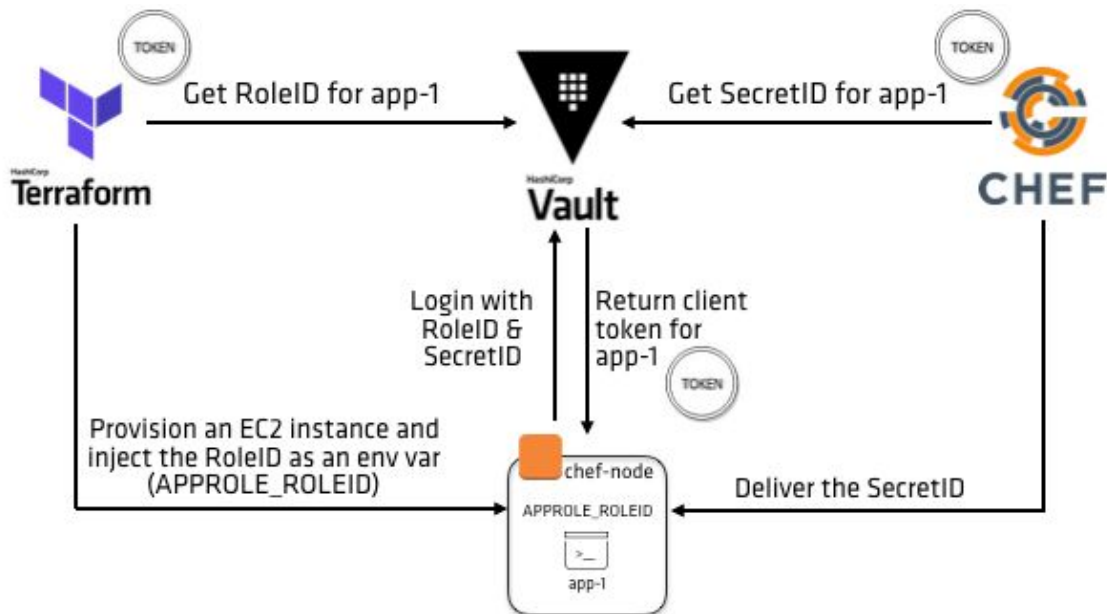
```
CODE EDITOR

jobs:
  build:
    # ...
    steps:
      # ...
      - name: Import Secrets
        uses: hashicorp/vault-action@v2.3.1
        with:
          url: https://vault.mycompany.com:8200
          token: ${ secrets.VaultToken }
          caCertificate: ${ secrets.VAULTCA }
          secrets: |
            secret/data/ci/aws accessKey |
AWS_ACCESS_KEY_ID ;
            secret/data/ci/aws secretKey |
AWS_SECRET_ACCESS_KEY ;
            secret/data/ci npm_token
```



Pipeline Integration

Chef



[AppRole With Terraform & Chef | Vault](#)

Poll time



Which third party
integrations will you be
using?

(short answer)

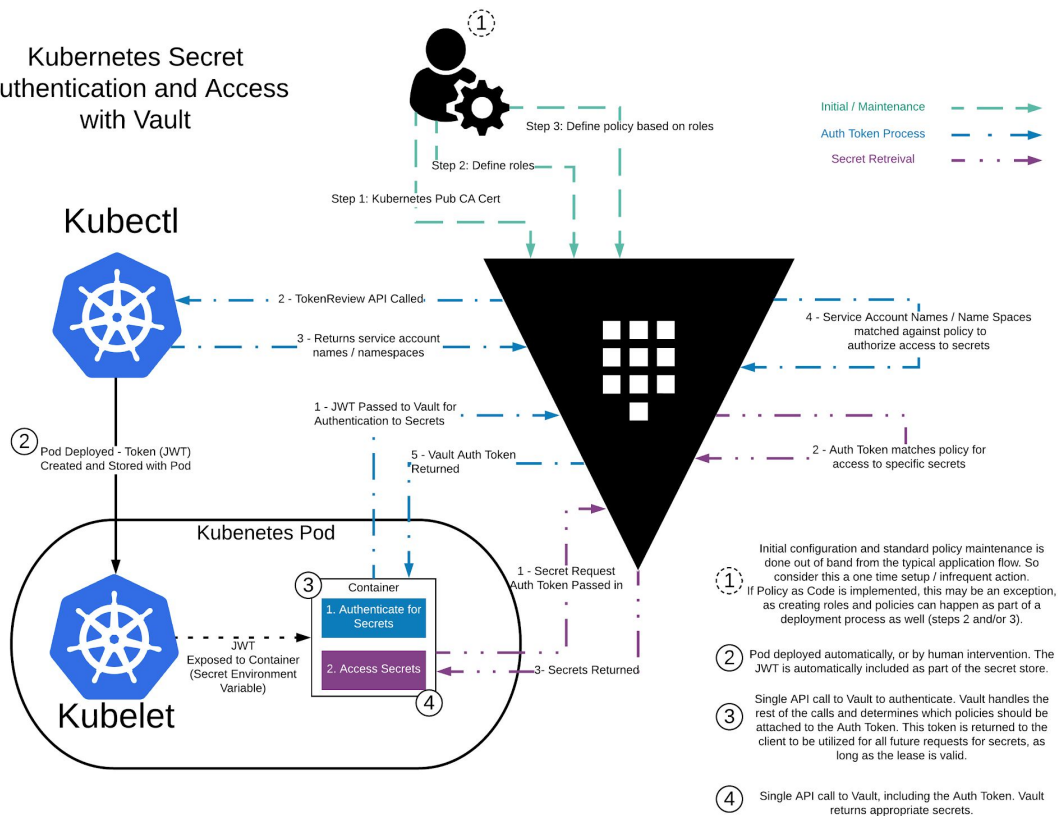
Kubernetes

Pod Secret Access



Kubernetes Auth Flow

Kubernetes Secret Authentication and Access with Vault





Application Pod Definition

```
apiVersion: v1
kind: Pod
...
spec:
  serviceAccountName: k8s-service-acct
  containers:
    - name: app
      image: burtlo/exampleapp-ruby:k8s
      env:
        - name: VAULT_ADDR
          value:
            "http://vault.default.svc.cluster.local:8200"
        - name: VAULT_ROLE
          value: "internal-app"
```



Example App Code Changes

```
CODE EDITOR

response =
  HTTP.put("#{vault_url}/v1/auth/kubernetes/login") do
    |req|
      req.headers['Content-Type'] = 'application/json'
      req.body = { "role" => vault_role, "jwt" => jwt
    }.to_json
  end

vault_token =
  JSON.parse(response.body)["auth"]["client_token"]

logger.info "Received Vault Token: [#{vault_token}]"
```

Vault Agent Injector

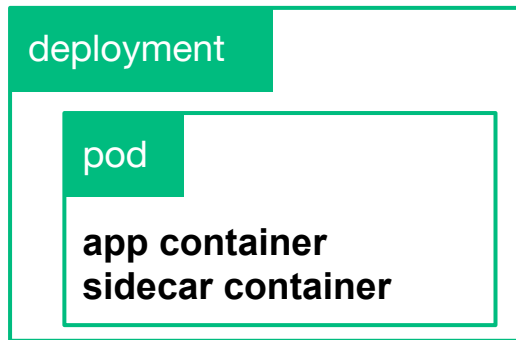


Sidecar Pattern

Vault unaware pods would offload the authentication and secret retrieval to a dedicated container appended to every deployment/pod.

Sidecar container needs:

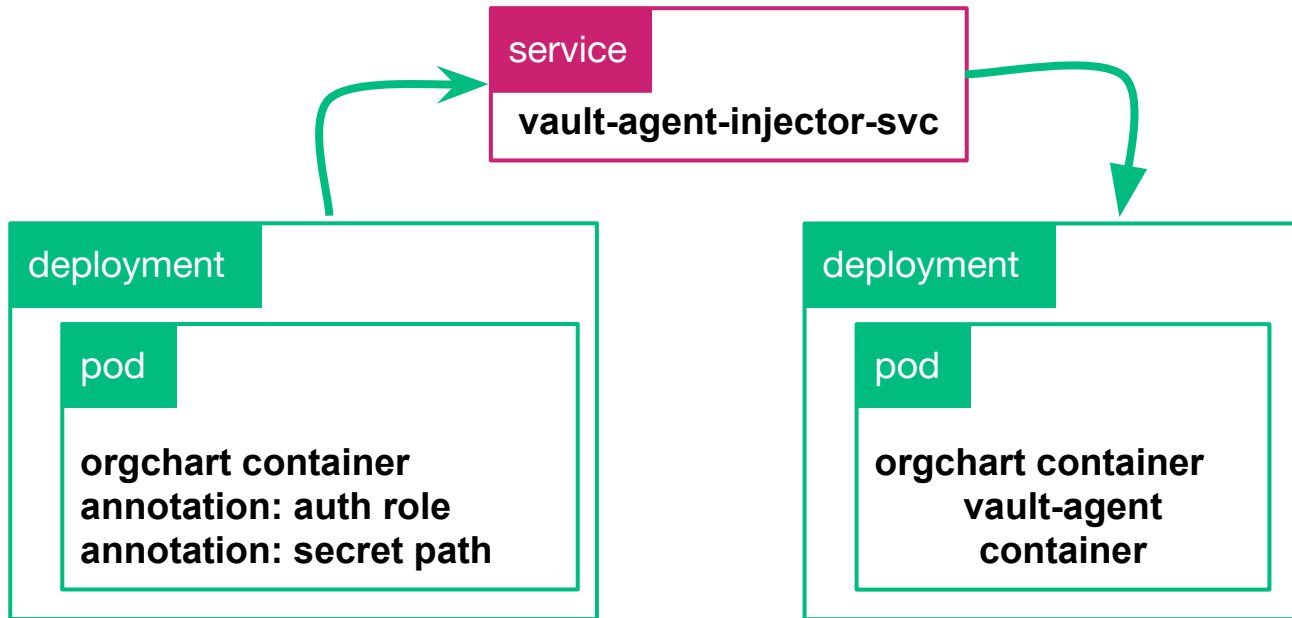
- Vault address
- Vault authentication role
- Vault secret path





Sidecar Pattern

Registers a Mutating Webhook Configuration that takes action when pod/deployment annotations are defined.





Install Agent Injector



TERMINAL

```
> helm repo add hashicorp
https://helm.releases.hashicorp.com
"hashicorp" has been added to your repositories

> helm search repo hashicorp/vault
NAME                CHART VERSION    APP VERSION DESCRIPTION
hashicorp/vault 0.18.0          1.9.0           Official
HashiCorp Vault Chart

> helm install vault hashicorp/vault \
--set="injector.enabled=true"
```


Agent Annotations



```
spec:
  template:
    metadata:
      annotations:
        vault.hashicorp.com/agent-inject: "true"
        vault.hashicorp.com/role: "internal-app"
        vault.hashicorp.com/agent-inject-secret-database-config.txt:
"internal/data/database/config"
```



View the Secret

```
> kubectl exec orgchart --container orgchart \
  -- cat /vault/secrets/database-config.txt
```

```
data: map[password:db-secret-password
username:db-readonly-user]
metadata:
map[created_time:2019-12-20T18:17:50.930264759Z
deletion_time: destroyed:false version:2]
```

Container Storage Interface

Overview



Secrets Store CSI driver for Kubernetes secrets - Integrates secrets stores with Kubernetes via a Container Storage Interface (CSI) volume.

The Secrets Store CSI driver `secrets-store.csi.k8s.io` allows Kubernetes to mount multiple secrets, keys, and certs stored in enterprise-grade external secrets stores into their pods as a volume. Once the Volume is attached, the data in it is mounted into the container's file system.

Secrets Store CSI Driver

kubernetes-sigs/secrets-store-csi-driver



The screenshot shows the GitHub repository page for `kubernetes-sigs/secrets-store-csi-driver`. The repository is public and has 17 watchers, 569 stars, and 136 forks. It has 48 issues, 4 pull requests, 1 project, and 1 wiki page. The repository is currently on the `main` branch, which has 6 branches and 28 tags. The repository description is: "Secrets Store CSI driver for Kubernetes secrets - Integrates secrets stores with Kubernetes via a CSI volume." The repository includes a README, Apache-2.0 License, and Code of conduct. The repository has 20 releases. The repository is maintained by `k8s-ci-robot`, who merged pull request #814 from `spiffxp/use-k8s-infra-for-...` 2 days ago. The repository also includes a list of recent commits and a table of files.

File	Commit	Time
<code>.github</code>	ci: add markdown-link-check workflow	22 days ago
<code>.local</code>	chore: remove deprecated <code>--filtered-watch-secret</code> flag	23 days ago
<code>apis</code>	feat: add SecretProviderClass and SecretProviderClassPodStatu...	2 months ago
<code>charts</code>	release: update manifest and helm charts for v1.0.0	2 months ago
<code>cmd/secrets-store-csi-driver</code>	chore: remove deprecated <code>--filtered-watch-secret</code> flag	23 days ago
<code>config</code>	feat: add SecretProviderClass and SecretProviderClassPodStatu...	2 months ago
<code>controllers</code>	Issue#636 - add last lint check items	2 months ago
<code>deploy</code>	release: update manifest and helm charts for v1.0.0	2 months ago
<code>docker</code>	images: use k8s-staging-test-infra/gcb-docker-gcloud	2 days ago
<code>docs</code>	docs: fix dead links based on errors	22 days ago



Install Container Storage Interface

```

TERMINAL

> helm repo add hashicorp
https://helm.releases.hashicorp.com
"hashicorp" has been added to your repositories

> helm search repo hashicorp/vault
NAME                CHART VERSION   APP VERSION DESCRIPTION
hashicorp/vault     0.18.0          1.9.0          Official
HashiCorp Vault Chart

> helm install vault hashicorp/vault \
--set "injector.enabled=false" \
--set "csi.enabled=true" \
--set "injector.externalVaultAddr=http://addr:8200"
```



Install Secrets Store CSI Driver

```

> helm repo add secrets-store-csi-driver \
https://raw.githubusercontent.com/kubernetes-sigs/secrets-store-csi-driver/master/charts
...

> helm install csi
secrets-store-csi-driver/secrets-store-csi-driver
...
```

Define SecretProviderClass



```
apiVersion: secrets-store.csi.x-k8s.io/v1alpha1
kind: SecretProviderClass
metadata:
  name: vault-database
spec:
  provider: vault
  parameters:
    vaultAddress: "http://vault.default.svc.cluster.local:8200"
    roleName: "internal-app"
    objects: |
      - objectName: "db-password"
        secretPath: "internal/data/database/config"
        secretKey: "password"
```

CODE EDITOR

Define a Pod with a Volume



```
spec:
  containers:
    - image: nginx
      name: webapp
      volumeMounts:
        - name: secrets-store-inline
          mountPath: "/mnt/secrets-store"
          readOnly: true
  volumes:
    - name: secrets-store-inline
      csi:
        driver: secrets-store.csi.k8s.io
        readOnly: true
        volumeAttributes:
          secretProviderClass: "vault-database"
```

CODE EDITOR

Next Steps

Next Steps



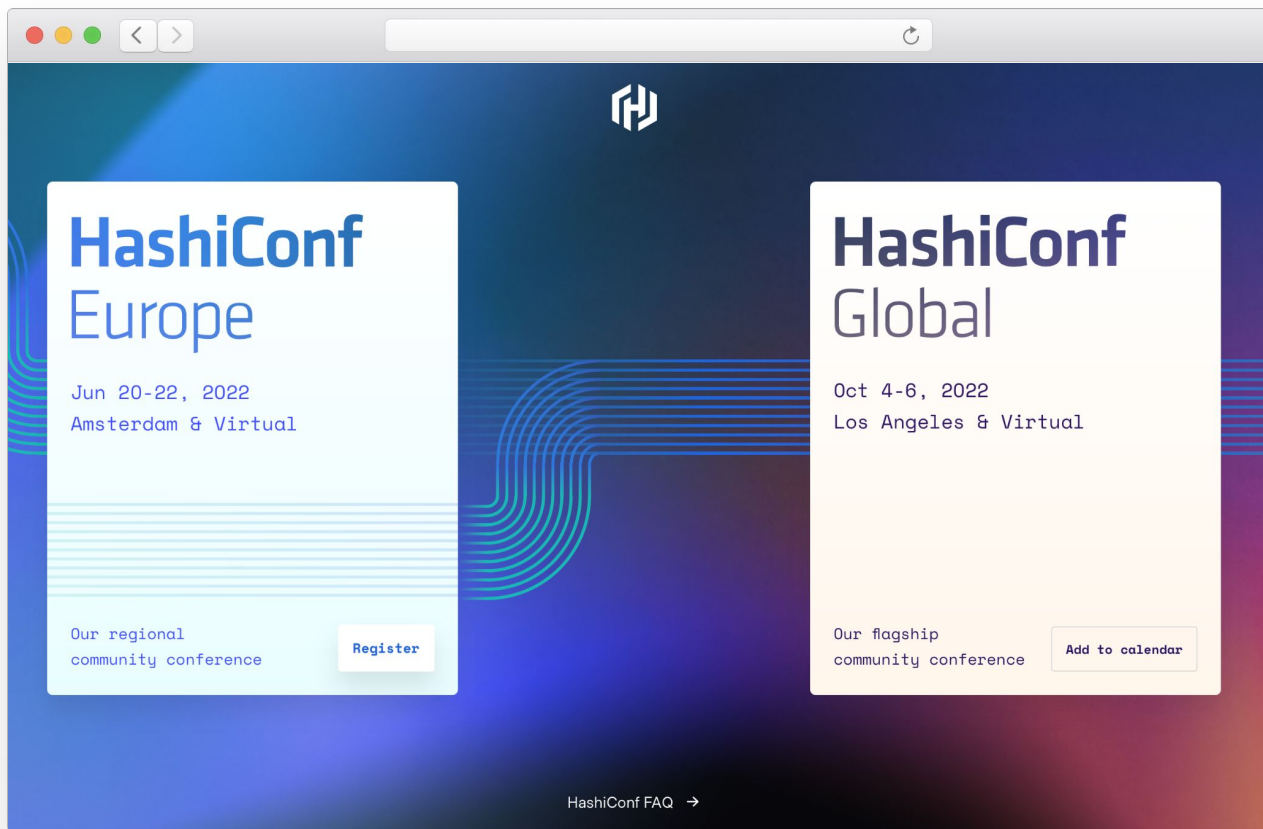
- Upcoming Schedule:

 Week 5 - HCP Vault Closing Session



HashiConf

<https://hashiconf.com>



<


>

HashiCorp | Discuss

Sign in





Q

≡



Information on HashiCorp Cloud Platform (HCP) use cases, Q&A, and best practices discussions. Please note that offerings may be in various release lifecycles. Categorize your question or comment under [HCP Consul](#) or [HCP Vault](#) subcategories. If your question or comment relates to networking, access control, or billing, post in this category. For support requests, please [open a ticket](#).

HashiCorp Cloud Platform (HCP) ▾ All ▾ Tags ▾ | Latest Top

Topic	Replies	Views	Activity
About the HashiCorp Cloud Platform (HCP) category			
<div>HashiCorp Cloud Platform (HCP)</div> <div>Information on HashiCorp Cloud Platform (HCP) use cases, Q&A, and best practices discussions. Please note that offerings may be in various release lifecycles. Categorize your question or comment under HCP Consul or HCP V... read more</div>	 1	387	May 24
<div>HCP Vault “per-client” pricing</div> <div>HCP Vault</div>	 0	39	9d
<div>Failing to use HCP Consul as my terraform backend</div> <div>HashiCorp Cloud Platform (HCP)</div>	 1	72	12d
<div>Does HCP support Automation APIs in AWS</div> <div>HashiCorp Cloud Platform (HCP) vault</div>	 0	73	27d



Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers.

discuss.hashicorp.com

Learn

Step-by-step guides to implement features in HCP and HCP Vault



HashiCorp Learn [Browse tutorials](#)

Search

[Sign in](#)

HCP Vault Week 1

- Week 1 - Welcome to HCP Vault
- HCP Vault Introduction
- Create a Vault Cluster on HCP
- Multi-tenancy with Namespaces
- Your First Secret
- Create Vault Policies
- Manage Authentication Methods
- Vault Operation Tasks
- Week 1 Wrap-up

HCP Vault Onboarding Week 1

1 HR 9 MIN 9 TUTORIALS

Get off to a strong start with HCP Vault in week 1 of your onboarding journey.

15 MIN

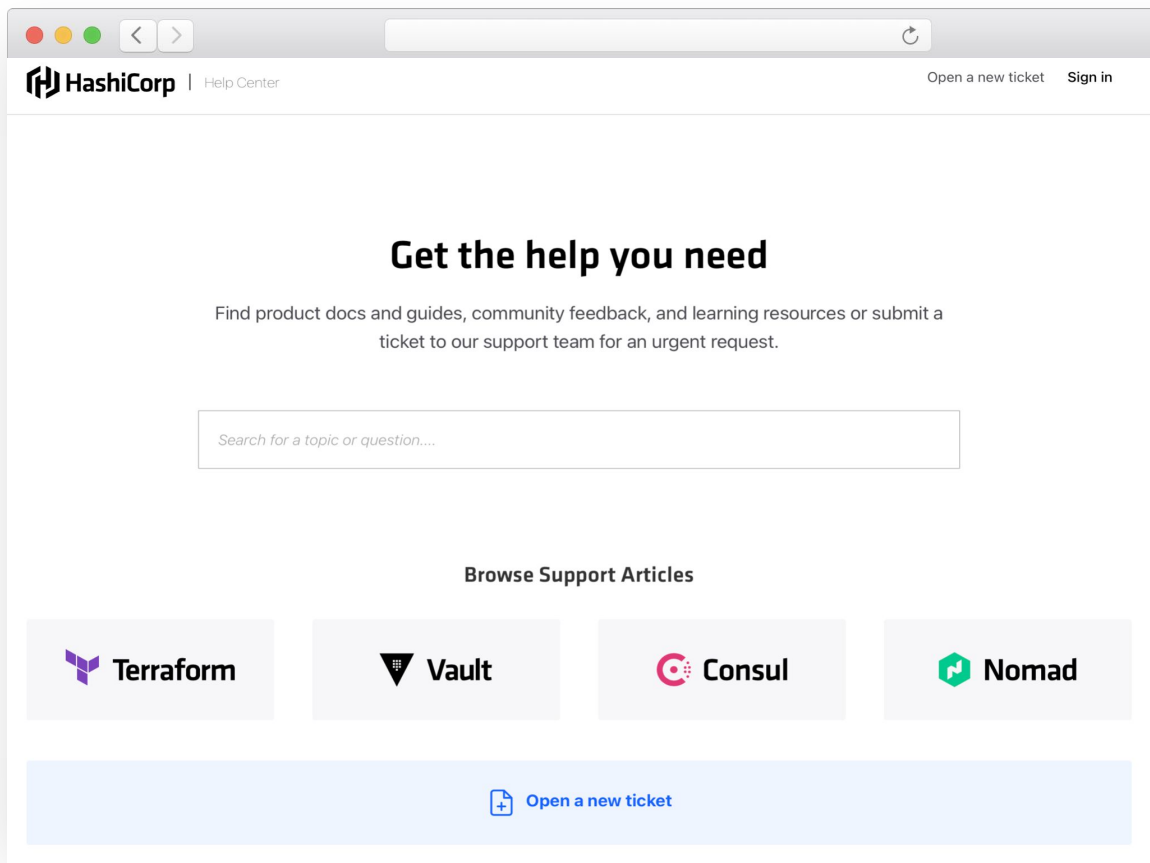
Week 1 - Welcome to HCP Vault

• Welcome to week 1 with HashiCorp Cloud Platform (HCP) Vault. Watch a quickstart demo and learn more about the internals of Vault.

5 MIN

HCP Vault Introduction

• HashiCorp Cloud Platform (HCP) Vault enables you to deploy a managed Vault service on AWS.



Support

<https://support.hashicorp.com>

Need Additional Help?



Customer Success

Contact our Customer Success

Management team with any questions.

We will help coordinate the right resources for you to get your questions answered.

customer.success@hashicorp.com

Technical Support

Something not working quite right?

Engage with HashiCorp Technical

Support by opening a new ticket for your issue at [Hashicorp Support](#).



Thank You

customer.success@hashicorp.com

www.hashicorp.com