



Consuming Secrets from HCP Vault



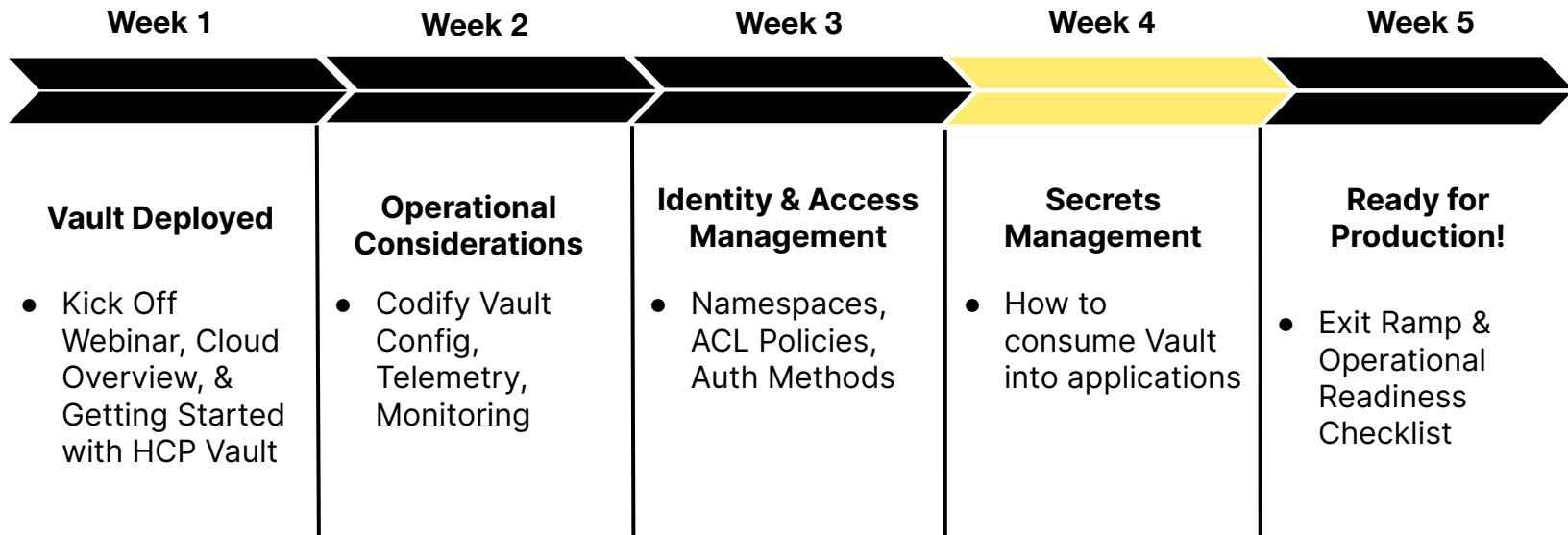
Agenda

- | | |
|-----------------------------|----|
| Secure Introduction | 01 |
| Consuming Secrets | 02 |
| Third Party Integrations | 03 |
| Kubernetes | 04 |
| Vault - Agent injector | 05 |
| Container Storage Interface | 06 |
-
-
-
-
-
-

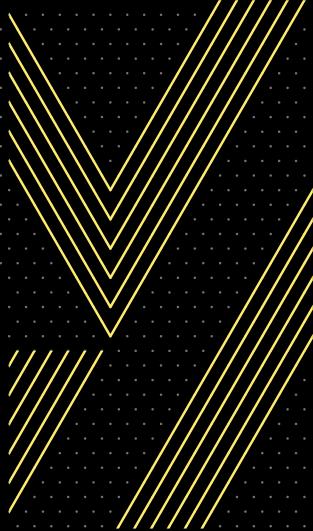
HCP Vault Onboarding Program

A 5 week guided community environment

Assisting customers with onboarding and adoption



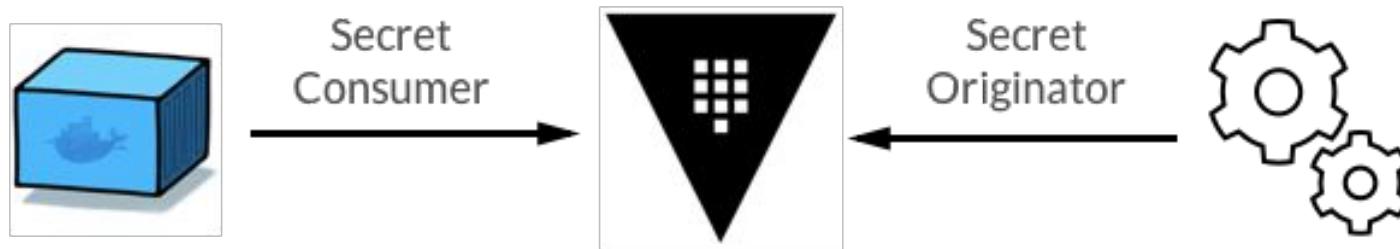
01



Secure Introduction

Secret Originator and Consumer

Successful secure distribution of a secret from an originator to a consumer, allows all subsequent secrets transmitted between them to be authenticated by the trust established by that initial successful transaction



- 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 trust relationship with a trusted platforms (AWS, Azure, GCP)
- The platforms identifier of resources (VMs, containers, etc) is used to authenticate and provide authorization to a Vault token

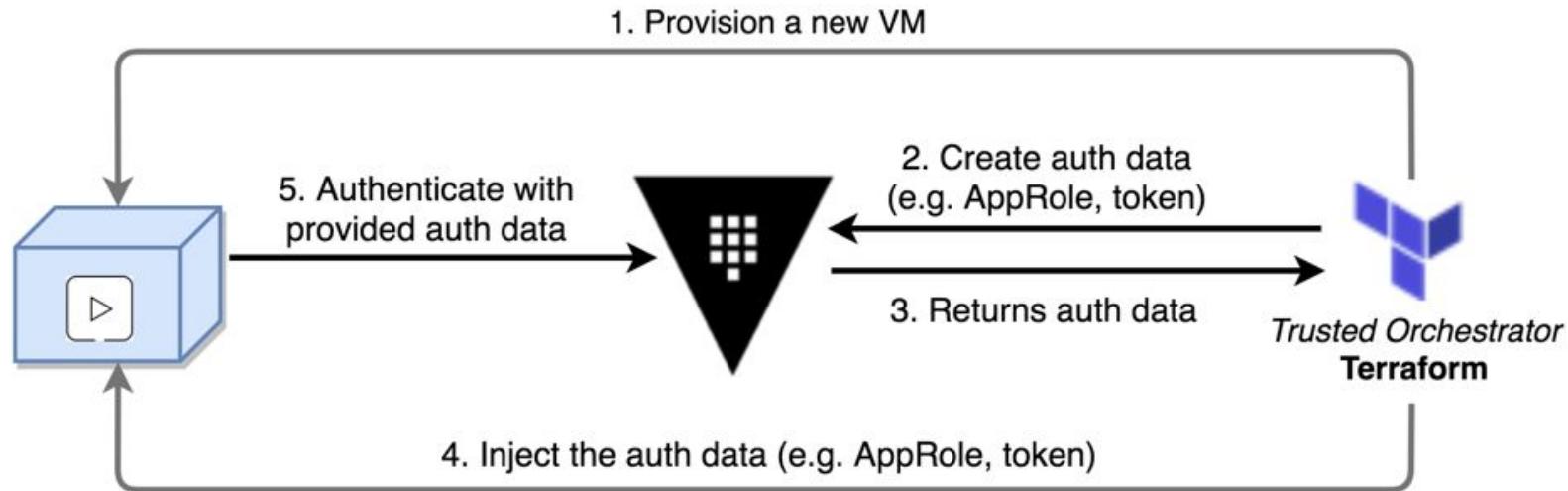
Trusted Orchestrator

- Existing trusted orchestrator (Terraform, Kubernetes, Chef) has already authenticated to Vault with privileged permissions
- At application deployment orchestrator injects necessary credentials to authenticate to Vault and retrieve a Vault token



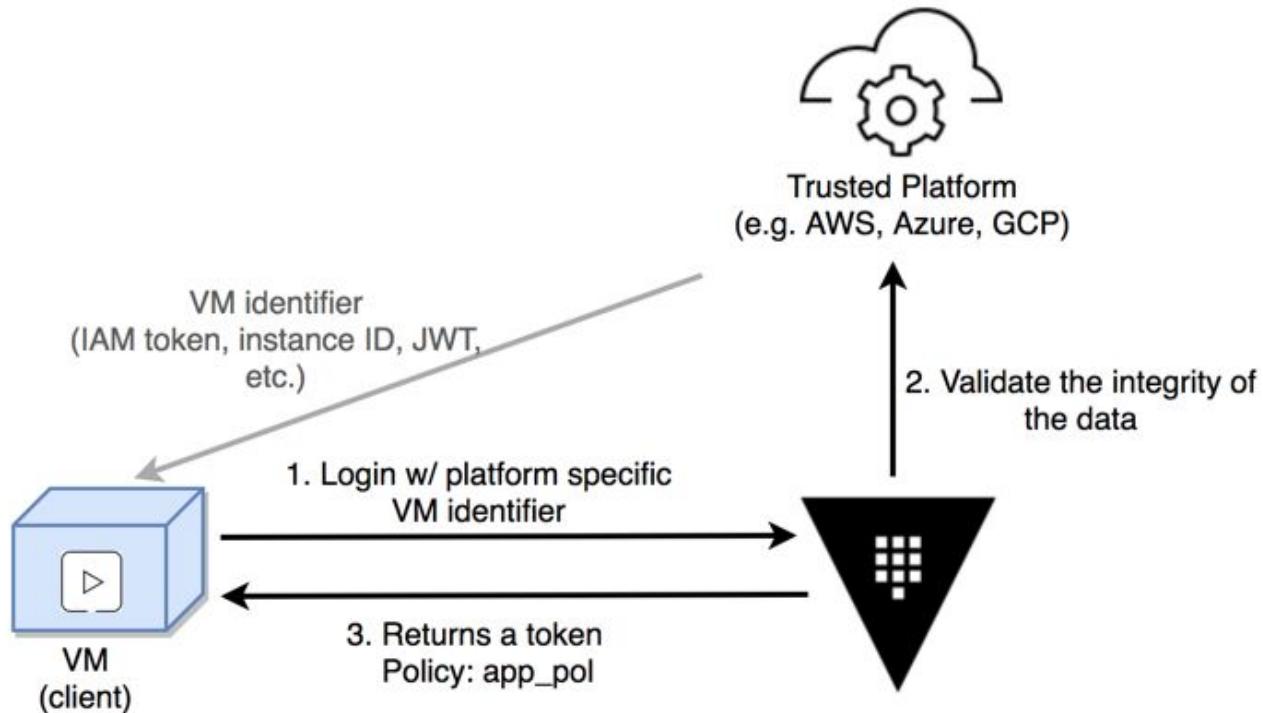
Trusted Orchestrator

Secure introduction in a VM environment



Platform Integration

Secure introduction in a cloud environment



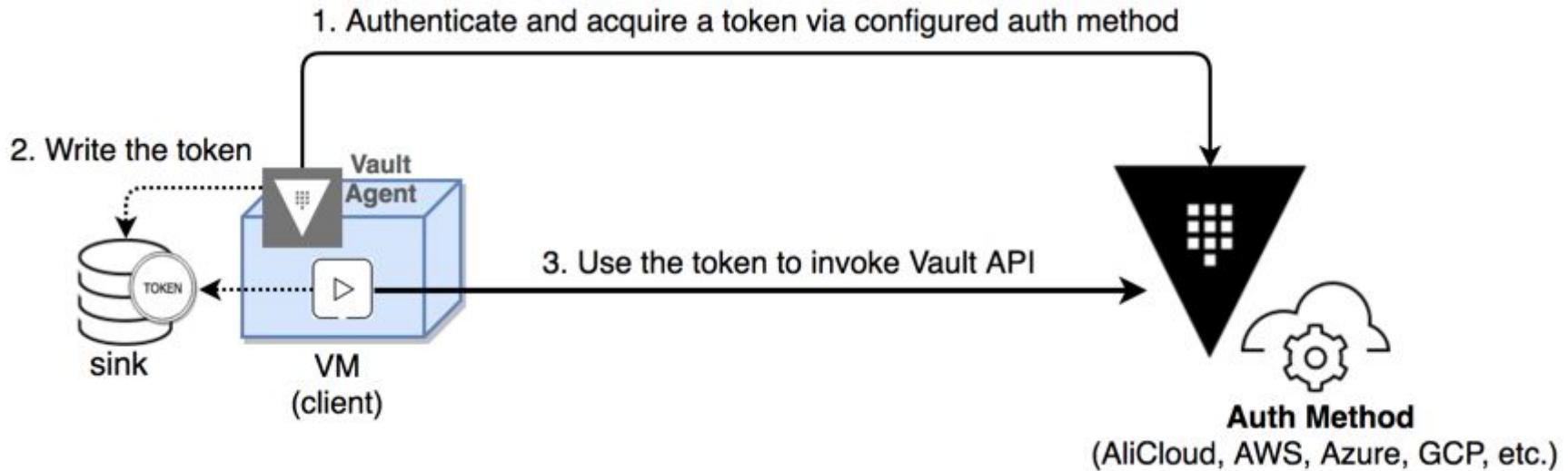
Automating Introduction

[Vault Agent](#) is a client daemon which automates the client login workflow and the lifecycle for Vault tokens

- Compatible with both platform integration and trusted orchestrator secure introduction methods
- Included as part of the Vault binary and can be run by starting the binary in agent mode - “***vault agent -config=<config-file>***”
- After authentication completes a Vault token is written to file sink

Automate Introduction

Vault Agent



Vault Agent Metrics

| Metric | Description | Type |
|---|---|---------|
| <code>vault.agent.auth.failure</code> | Number of authentication failures | Counter |
| <code>vault.agent.auth.success</code> | Number of authentication successes | Counter |
| <code>vault.agent.proxy.success</code> | Number of requests successfully proxied | Counter |
| <code>vault.agent.proxy.client_error</code> | Number of request for which Vault returned an error | Counter |
| <code>vault.agent.proxy.error</code> | Number of requests the agent failed to proxy | Counter |
| <code>vault.agent.cache.hit</code> | Number of cache hits | Counter |
| <code>vault.agent.cache.miss</code> | Number of cache misses | Counter |

02



Consuming Secrets

Patterns to Consume Secrets

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

Web UI

- Users can populate and consume secrets without learning CLI or API commands
- Works well for users to consuming secrets
- Can be limiting when secrets need to be consumed at scale or as part of an application configuration

The screenshot shows the HashiCorp Vault Web UI interface. At the top, there's a navigation bar with tabs for 'Secrets', 'Access', 'Policies', and 'Tools'. On the right side of the bar are 'Status' (green), and dropdown menus for 'Copy', 'Version 1', and 'Create new version'. Below the bar, the path 'sales_secrets < db' is displayed. The main area is titled 'db' and contains two tabs: 'Secret' (which is selected) and 'Metadata'. Under the 'Secret' tab, there's a JSON toggle switch, a 'Delete' button, and a 'Copy' dropdown menu. A timestamp 'Version created Nov 22, 2021 10:11 PM' is shown. The table lists three secrets:

| Key | Value | Actions |
|-------------|-------|--------------|
| db_password | ***** | Copy, Delete |
| db_username | **** | Copy, Delete |
| priv_key | ***** | Copy, Delete |

CLI

Typically used by users
for manual secret
consumption

```
...  
eye $ 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     djbsdougjbnndoijignsdoigsd
```

HTTP API

Feature rich API provides full access to Vault and every aspect of Vault can be controlled via this method

```
...
$ curl --header "X-Vault-Token: $VAULT_TOKEN" \
--header "X-Vault-Namespace: $VAULT_NAMESPACE" \
$VAULT_ADDR/v1/secret/data/customer/acme | jq -r ".data"

===== Output =====

{
  "data": {
    "contact_email": "john.smith@acme.com",
    "customer_name": "ACME Inc."
  },
  "metadata": {
    "created_time": "2021-10-29T02:09:32.112647Z",
    "custom_metadata": null,
    "deletion_time": "",
    "destroyed": false,
    "version": 2
  }
}
```

HTTP API Explorer

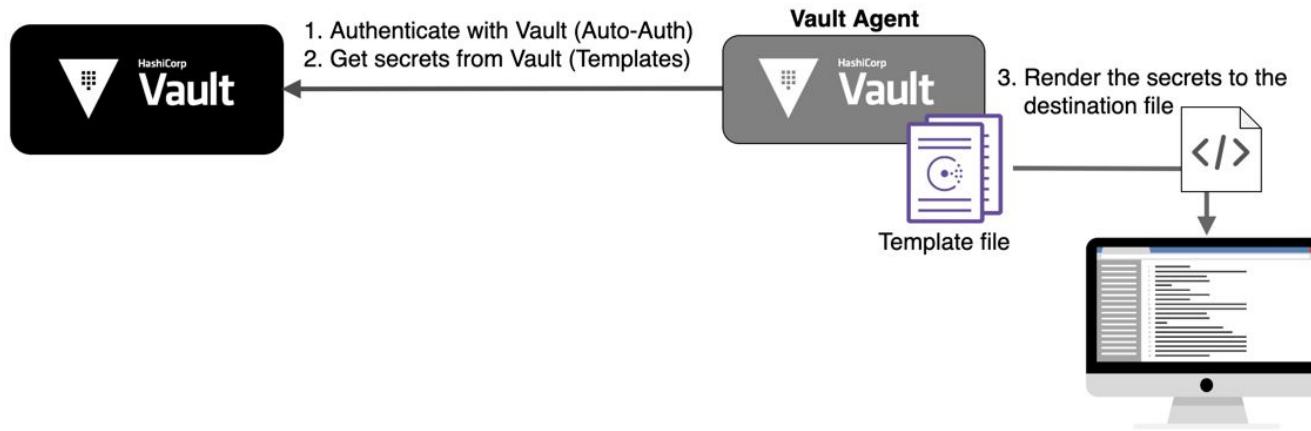
<VAULT_ADDR>/ui/vault/api-explorer

The screenshot shows the HashiCorp Vault API Explorer interface. At the top, there is a navigation bar with tabs for Secrets, Access, Policies, Tools, and Status. Below the navigation bar, there are several API endpoint cards:

- Deletes the secret at the specified location.**
- GET /sales_secrets/config**
Read the backend level settings.
- POST /sales_secrets/config**
Configure backend level settings that are applied to every key in the key-value store.
- GET /sales_secrets/data/{path}**
Write, Patch, Read, and Delete data in the Key-Value Store.
- POST /sales_secrets/data/{path}**
Write, Patch, Read, and Delete data in the Key-Value Store.
- DELETE /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 with secrets read from Vault making them available to applications

```
...
#!/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 -upcase -secret
database/creds/readonly ./app.sh

My connection info is:
username:
"v-token-readonly-ww1tq33s7z5uprpxxy68-1527631219"
password: "A1a-u54wut0v605qwz95"
database: "my-app"
```

Go Client Library

[Reference Documentation](#)

```
...
// 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]))
}
```

03



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
- HashiCorp Support Engineering can support teams from a Vault perspective, any issues with the framework or tooling will need to be raised with the creator of those frameworks or tooling

Enterprise Capabilities

- HashiCorp has 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 is being used does not support enterprise capabilities, please have the creators reach out to HashiCorp to assist with supporting enterprise capabilities



Java Applications

Spring Cloud Vault
client libraries

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

```
...
@Configuration
@RestController
public class Application {

    @Value("${config.name}")
    String name = "World";

    @RequestMapping("/")
    public String home() {
        return "Hello " + name;
    }

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Vault C# Client

Integrate with your .Net Applications

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

```
...
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; }
}
```

Ruby Plugin

Integrate with Ruby on
Rails Applications

[Vault Rails](#)

```
...
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..."
```

Pipeline Integration

Github Actions

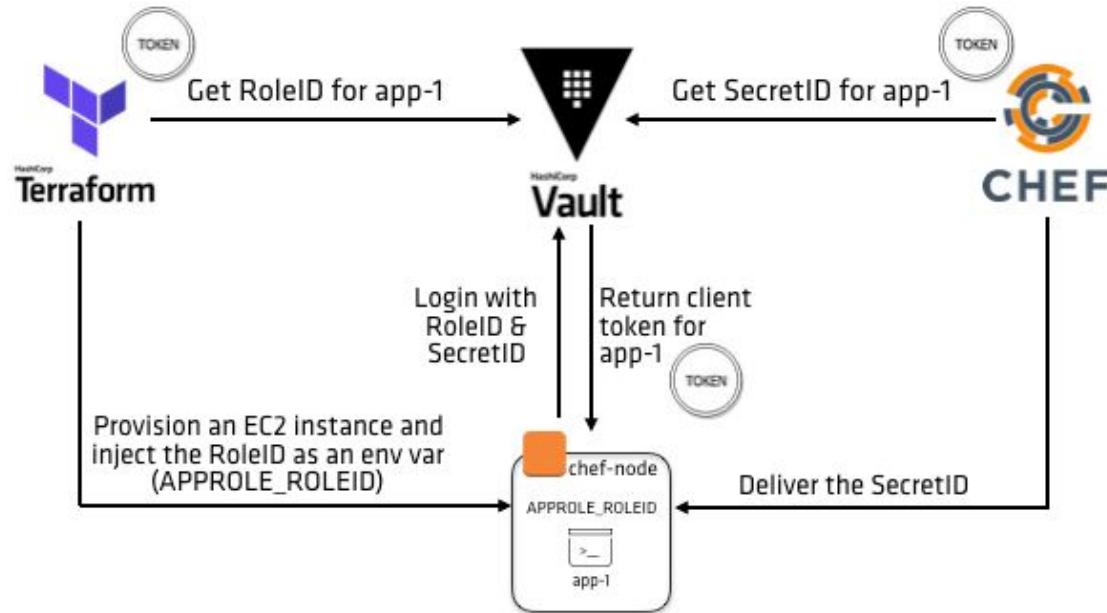
[Github Actions : Vault Secrets](#)

```
...
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](#)



04

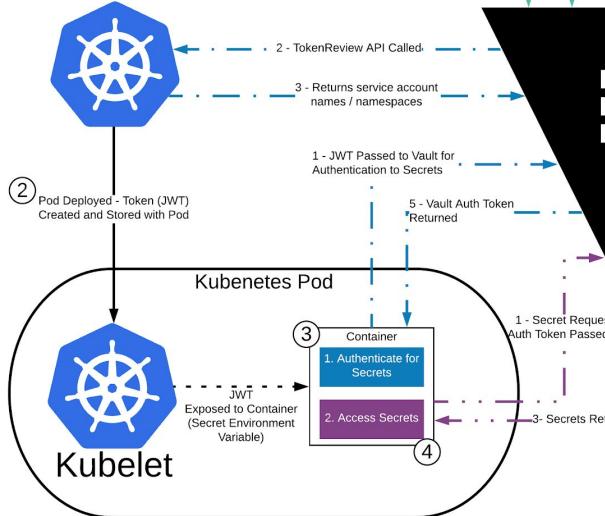


Kubernetes

Kubernetes Auth Flow

Kubernetes Secret Authentication and Access with Vault

Kubectl



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

```
...  
    response =  
      HTTP.put("#{vault_url}/v1/auth/kubernetes/login")  
      do |req|  
        req.headers['Content-Type'] = 'application/json'  
        req.body = { "role" => vault_role, "jwt" => jwt }  
      end  
  
    vault_token =  
      JSON.parse(response.body)["auth"]["client_token"]  
  
    logger.info "Received Vault Token:  
    [#{vault_token}]"
```

05



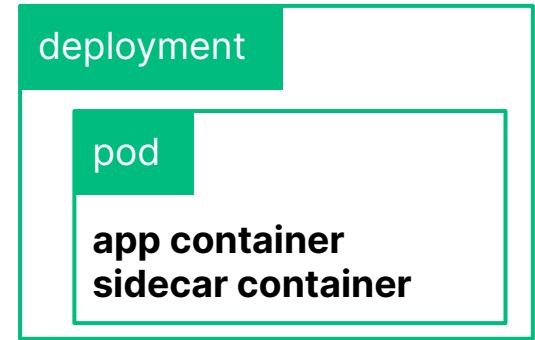
Vault Agent Injector

Sidecar Pattern

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

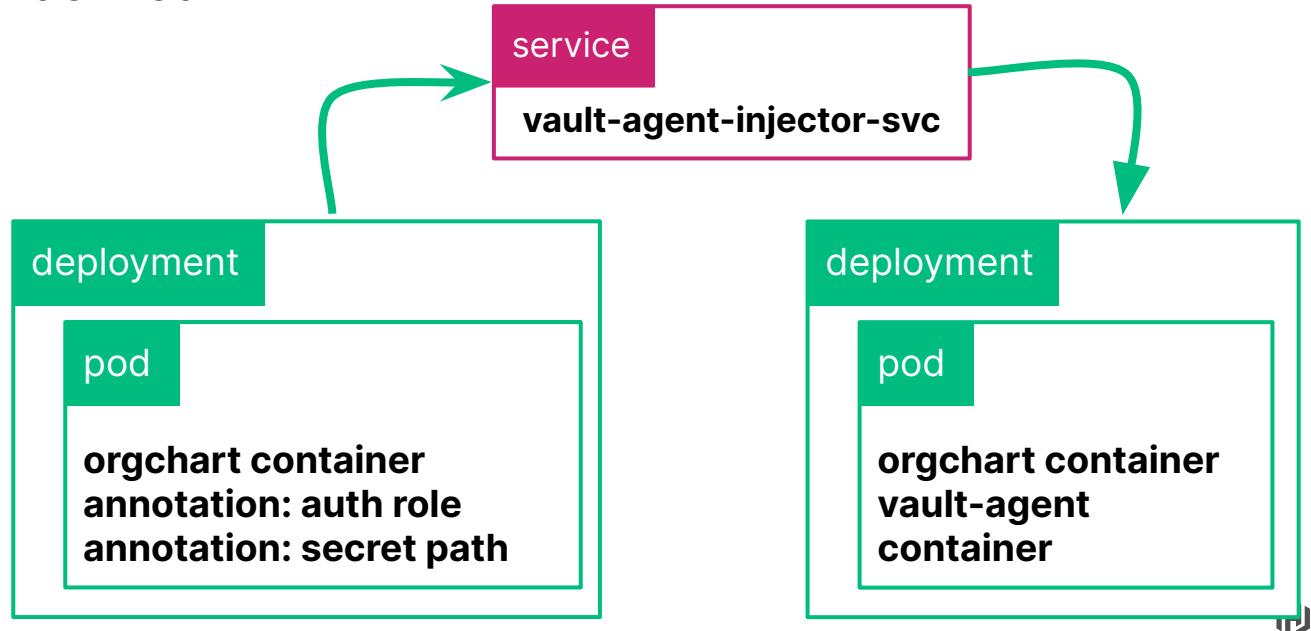
Sidecar container needs:

- 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

```
...
> 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]
```

06



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 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 is mounted into the container's file system

Secrets Store CSI Driver

[CSI Driver](#)

The screenshot shows the GitHub repository page for `kubernetes-sigs/secrets-store-csi-driver`. The repository is public, has 17 stars, 569 forks, and 136 issues. It features 6 branches and 28 tags. The main branch is highlighted. The repository description is: "Secrets Store CSI driver for Kubernetes secrets - Integrates secrets stores with Kubernetes via a CSI volume." Key contributors include `k8s-ci-robot` and `spiffxp`. The repository is tagged with `kubernetes`, `hashicorp-vault`, `csi`, `azure-keyvault`, `aws-secrets-manager`, `k8s-sig-auth`, `gcp-secret-manager`, `csi-secrets-store`, and `mount-multiple-secrets`. It includes links to `Readme`, `Apache-2.0 License`, and `Code of conduct`. The repository has 20 releases.

kubernetes-sigs / secrets-store-csi-driver Public

Watch 17 ⌂ Star 569 ⌂ Fork 136

< Code Issues 48 Pull requests 4 Actions Projects 1 Wiki Security Insights

main 6 branches 28 tags Go to file Add file ▾ Code ▾

About

Secrets Store CSI driver for Kubernetes secrets - Integrates secrets stores with Kubernetes via a CSI volume.

Commits

`k8s-ci-robot` Merge pull request #814 from spiffxp/use-k8s-infra-for-... 0f4ff7d 2 days ago 859 commits

`.github` ci: add markdown-link-check workflow 22 days ago

`.local` chore: remove deprecated `--filtered-watch-secret` flag Nov 10, 2021, 1:44 PM EST 23 days ago

`apis` feat: add SecretProviderClass and SecretProviderClassPodStatus 2 months ago

`charts` release: update manifest and helm charts for v1.0.0 2 months ago

`cmd/secrets-store-csi-driver` chore: remove deprecated `--filtered-watch-secret` flag 23 days ago

`config` feat: add SecretProviderClass and SecretProviderClassPodStatus 2 months ago

`controllers` Issue#636 - add last lint check items 2 months ago

`deploy` release: update manifest and helm charts for v1.0.0 2 months ago

`docker` images: use k8s-staging-test-infra/gcb-docker-gcloud 2 days ago

`docs` docs: fix dead links based on errors 22 days ago

Tags

`kubernetes` `hashicorp-vault` `csi` `azure-keyvault` `aws-secrets-manager` `k8s-sig-auth` `gcp-secret-manager` `csi-secrets-store` `mount-multiple-secrets`

Links

`Readme` `Apache-2.0 License` `Code of conduct`

Releases 20



Install Container Storage Interface

```
...
> 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"
```

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"
```



Pattern Comparison

[Kubernetes Vault](#)
[Integration via Sidecar](#)
[Agent Injector vs. CSI Provider](#)

| | Agent Sidecar | CSI |
|----------------------------|--|---|
| Secret projection | Shared Memory Volume Environment Variable | Ephemeral Disk Environment Variables Kubernetes Secrets |
| Secret scope | Global | Global |
| Secret types | All Secret Engines (Static & Dynamic) | All Secret Engines (Static & Dynamic) |
| Secret templating | Yes | No |
| Secret size limit | No Limit (both storage types) | No Limit (both storage types) |
| Secret definitions | CLI / API / UI | CLI / API / UI |
| Encryption | Yes (at rest & in-transit) | Yes (at rest & in-transit) |
| Secret rotation | Yes | No |
| Secret caching | Yes | No |
| Auditability | Yes | Yes |
| Deployment method | 1 Shared K8s Cluster Service + 1 Sidecar Container Per Application Pod | Daemonset |
| Vault agent support | Yes | No |
| Helm support | Yes | Yes |

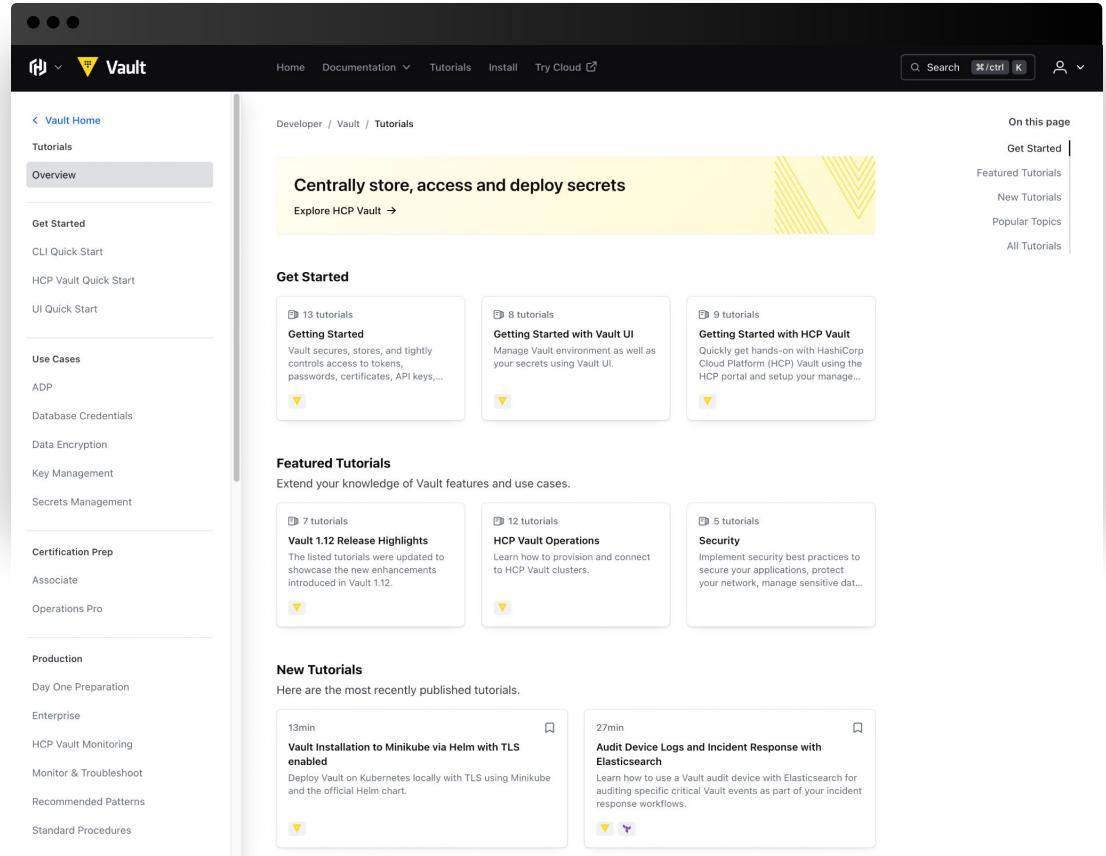


Next Steps



Tutorials

Step-by-step guides to accelerate deployment of Vault



The screenshot shows the HashiCorp Vault Tutorials page. At the top, there's a navigation bar with links for Home, Documentation, Tutorials, Install, Try Cloud, and a search bar. On the left, a sidebar lists various tutorial categories: Overview, Get Started, CLI Quick Start, HCP Vault Quick Start, UI Quick Start, Use Cases (ADP, Database Credentials, Data Encryption, Key Management, Secrets Management), Certification Prep (Associate, Operations Pro), Production (Day One Preparation, Enterprise, HCP Vault Monitoring, Monitor & Troubleshoot, Recommended Patterns, Standard Procedures). The main content area features a yellow banner with the text "Centrally store, access and deploy secrets" and a link to "Explore HCP Vault". Below this, there are three sections: "Get Started" (with 13 tutorials), "Featured Tutorials" (with 7, 12, and 5 tutorials for Vault 1.12 Release Highlights, HCP Vault Operations, and Security), and "New Tutorials" (with two examples: "Vault Installation to Minikube via Helm with TLS enabled" and "Audit Device Logs and Incident Response with Elasticsearch"). A sidebar on the right titled "On this page" includes links for Get Started, Featured Tutorials, New Tutorials, Popular Topics, and All Tutorials.

<https://developer.hashicorp.com/vault/tutorials>

Resources

- [Vault API Explorer](#)
- [Vault Agent](#)
- [Vault Agent Templates](#)
- [Vault Agent Metrics](#)
- [Consul Template & Envconsul with Vault](#)
- [Secure Introduction of Vault Clients](#)
- [Vault AWS Lambda Extension](#)
- [Collection of sample code using Vault client libraries \(C#, Go, Ruby, Python, Java\)](#)



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 ticket for your issue at
support.hashicorp.com

Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers

discuss.hashicorp.com



Upcoming Webinar

Program Closing

Asynchronous content sent to your inbox that includes some useful resources to validate production readiness and ensure operational best practices for your HCP Vault clusters



HCP Vault

Action Items

- Share to customer.success@hashicorp.com
 - Authorized technical contacts for support
 - Stakeholders contact information (name and email addresses)
- Assess how teams or applications will access Vault
- Plan how your organization will internally share patterns and best practices for utilizing secrets from Vault



Q&A





Thank you

customer.success@hashicorp.com

www.hashicorp.com/customer-success