HashiCorp Vault

Consuming Vault

Agenda

Vault Onboarding Program

A 7 week guided community environment
Assisting customers with onboarding and adoption

Week 1-2

Vault Architecture Deep dive

- Kick Off & Architectural Quickstart
- Architectural Deep-dive
- Migrating from Vault OSS*

Week 3-4

Identity and Access Management

- Namespaces, ACL Policies, Auth Methods
- Community Office Hours
- Using Vault with Kubernetes*

Week 5

Operationalize

- Telemetry & Monitoring, DR Operations, Runbooks

Week 6

Secrets Management

- How to consume Vault into applications
- Vault Dynamic Secrets*

Week 7

Ready for Production!

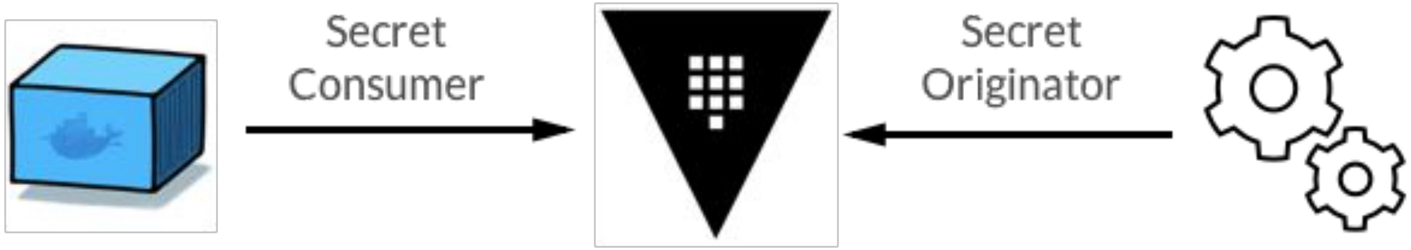
- Vault Governance
- Exit Ramp & Operational Readiness Checklist (asynchronous)

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

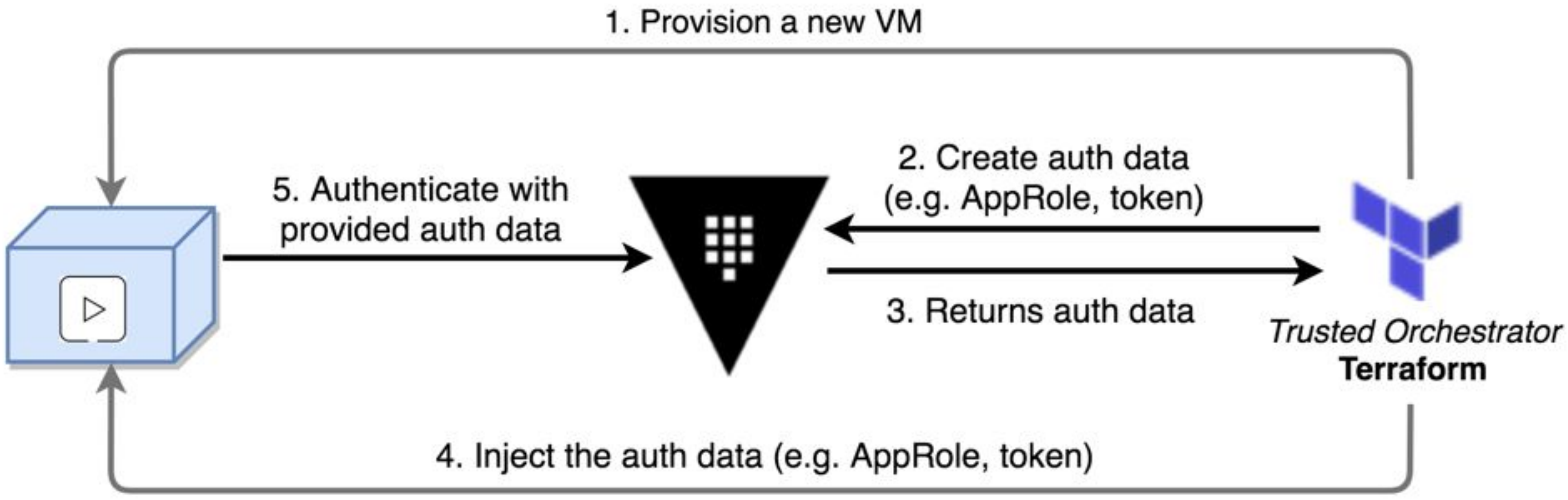
Vault establishes a trust with your trusted platforms (AWS, Azure, GCP) to use the identifier of resources (virtual instances, containers, etc) to authenticate and provide authorization to a Vault token

Trusted Orchestrator

Existing trusted orchestrator (Terraform, Kubernetes, Chef) has already been authenticated to Vault with privileged permissions; during deployment of applications the orchestrator injects necessary credentials to authenticate to Vault and retrieve a Vault token

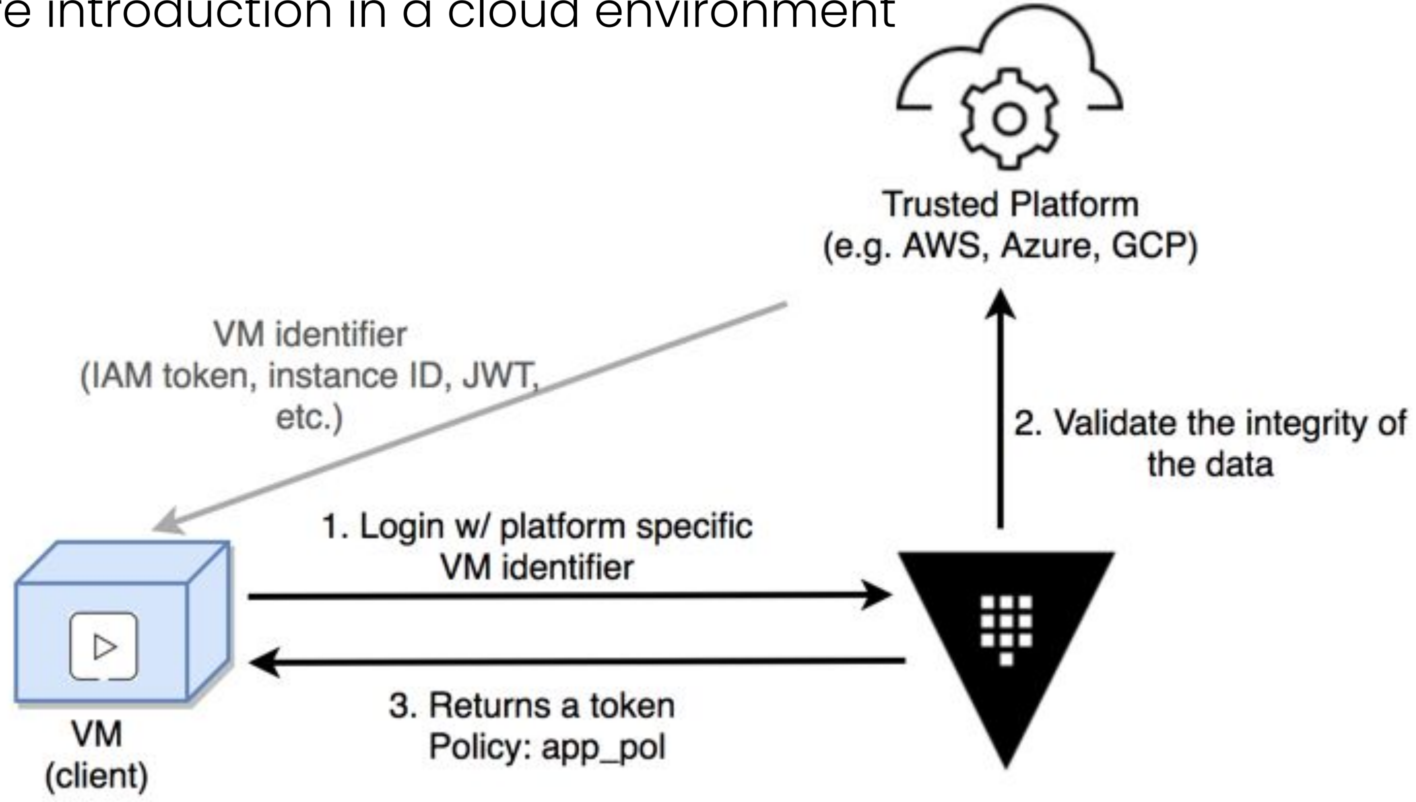
Trusted Orchestrator

Secure introduction in a VM environment


1. Provision a new VM
5. Authenticate with provided auth data
2. Create auth data (e.g. AppRole, token)
3. Returns auth data
Trusted Orchestrator
Terraform
4. Inject the auth data (e.g. AppRole, token)

Platform Integration

Secure introduction in a cloud environment


Trusted Platform
(e.g. AWS, Azure, GCP)
VM identifier
(IAM token, instance ID, JWT, etc.)
2. Validate the integrity of the data
1. Login w/ platform specific VM identifier
3. Returns a token
Policy: app_pol
VM
(client)

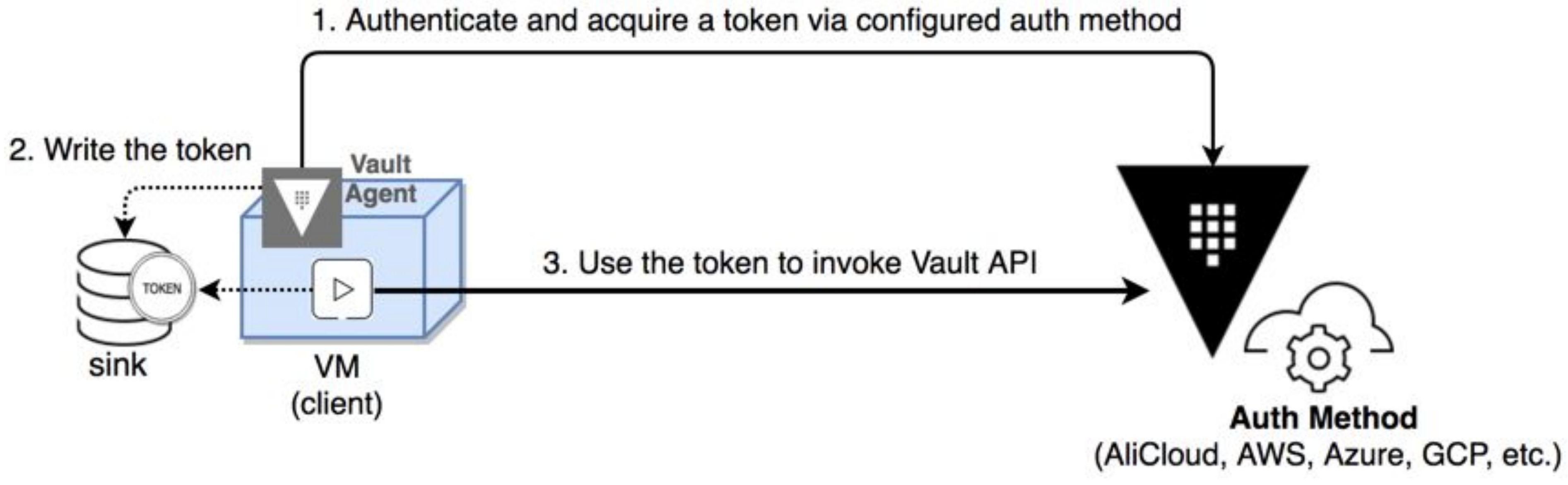
Automating Introduction

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


1. Authenticate and acquire a token via configured auth method
2. Write the token
Vault Agent
TOKEN
sink
VM
(client)
3. Use the token to invoke Vault API
Auth Method
(AliCloud, AWS, Azure, GCP, etc.)

Vault Agent Metrics

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

02

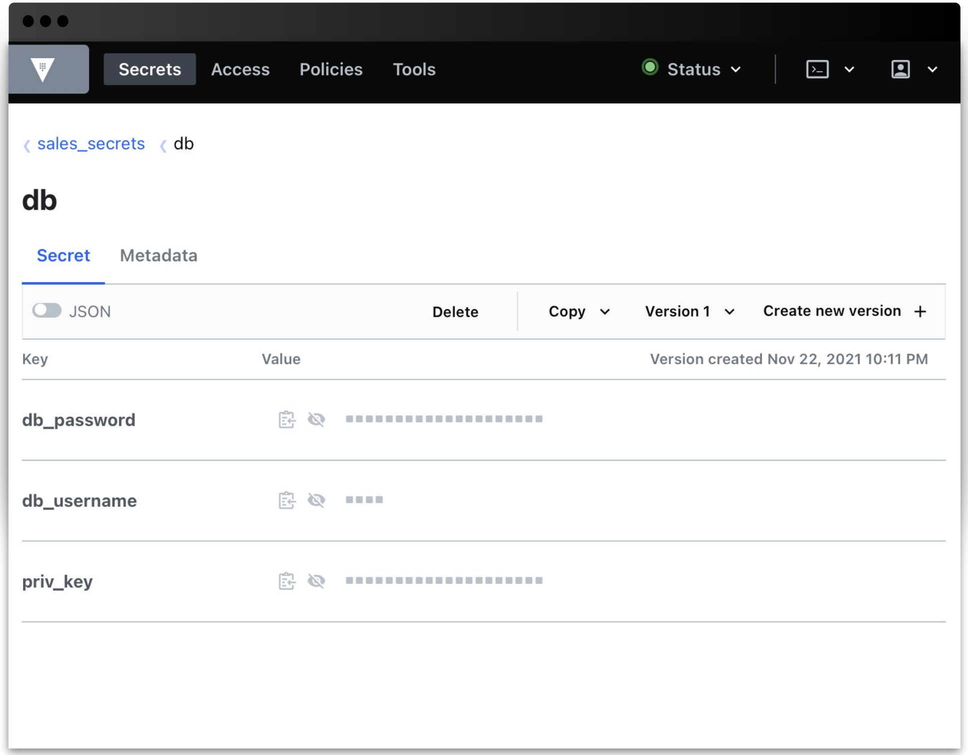
Consuming Secrets

Patterns to Consume Secrets

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

Web UI

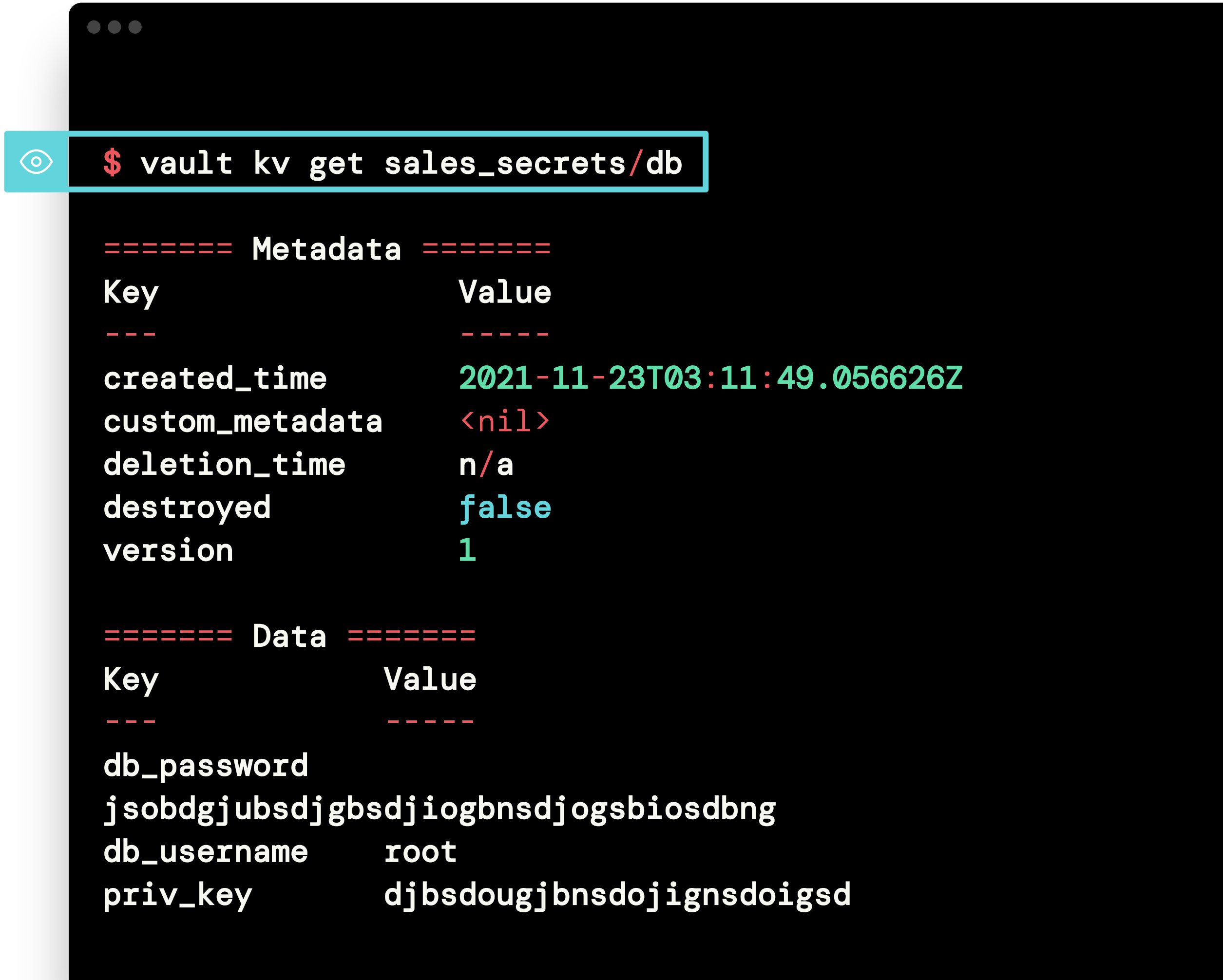
- Users can populate and consume secrets without learning CLI or API commands
- Works well for users to consuming secrets
- Can be limiting when secrets need to be consumed at scale or as part of an application configuration



Key	Value	Version created Nov 22, 2021 10:11 PM
db_password	••••••••••••••••••••	
db_username	••••	
priv_key	••••••••••••••••••••	

CLI

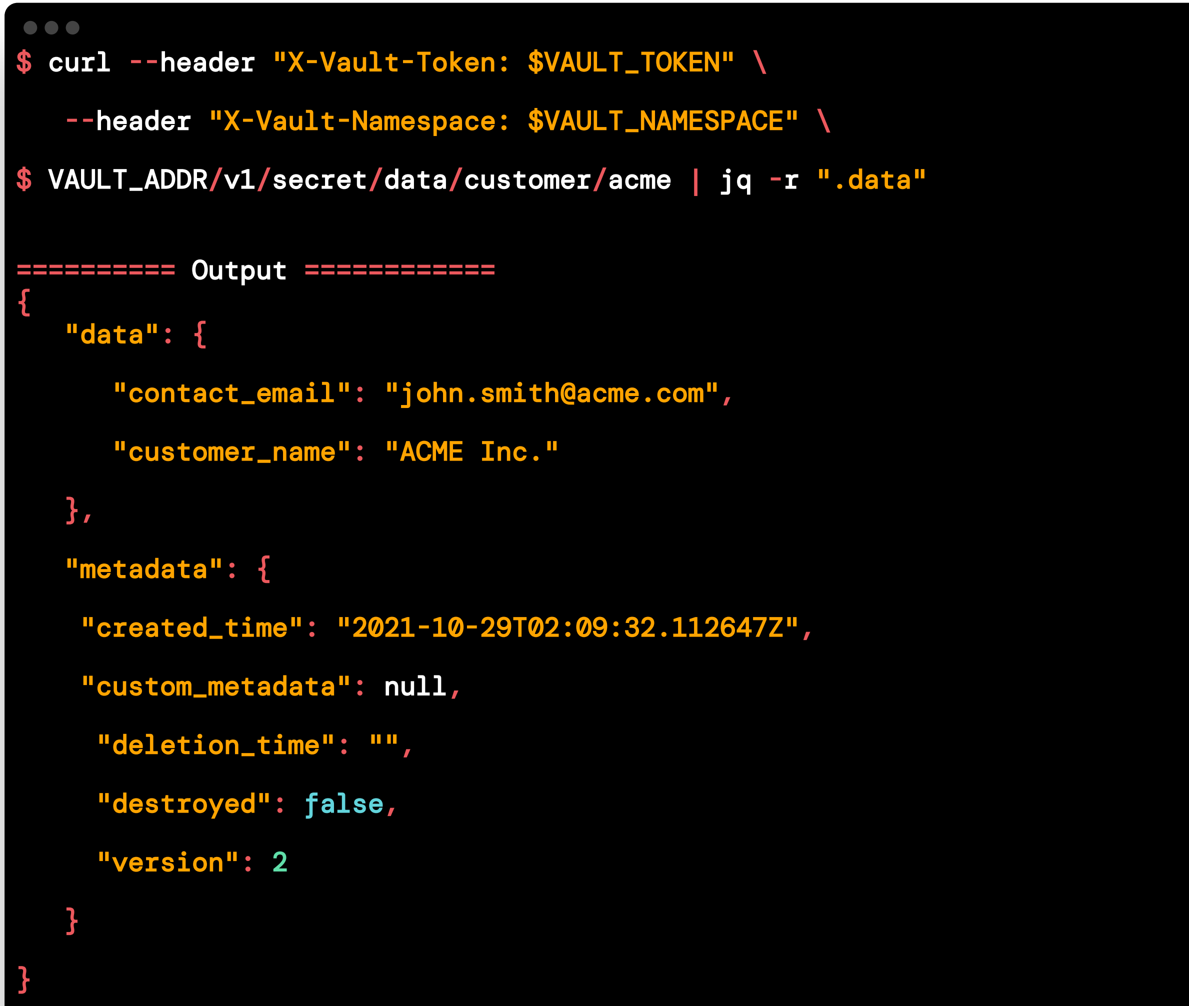
Typically used by users for manual secret consumption



```
$ 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

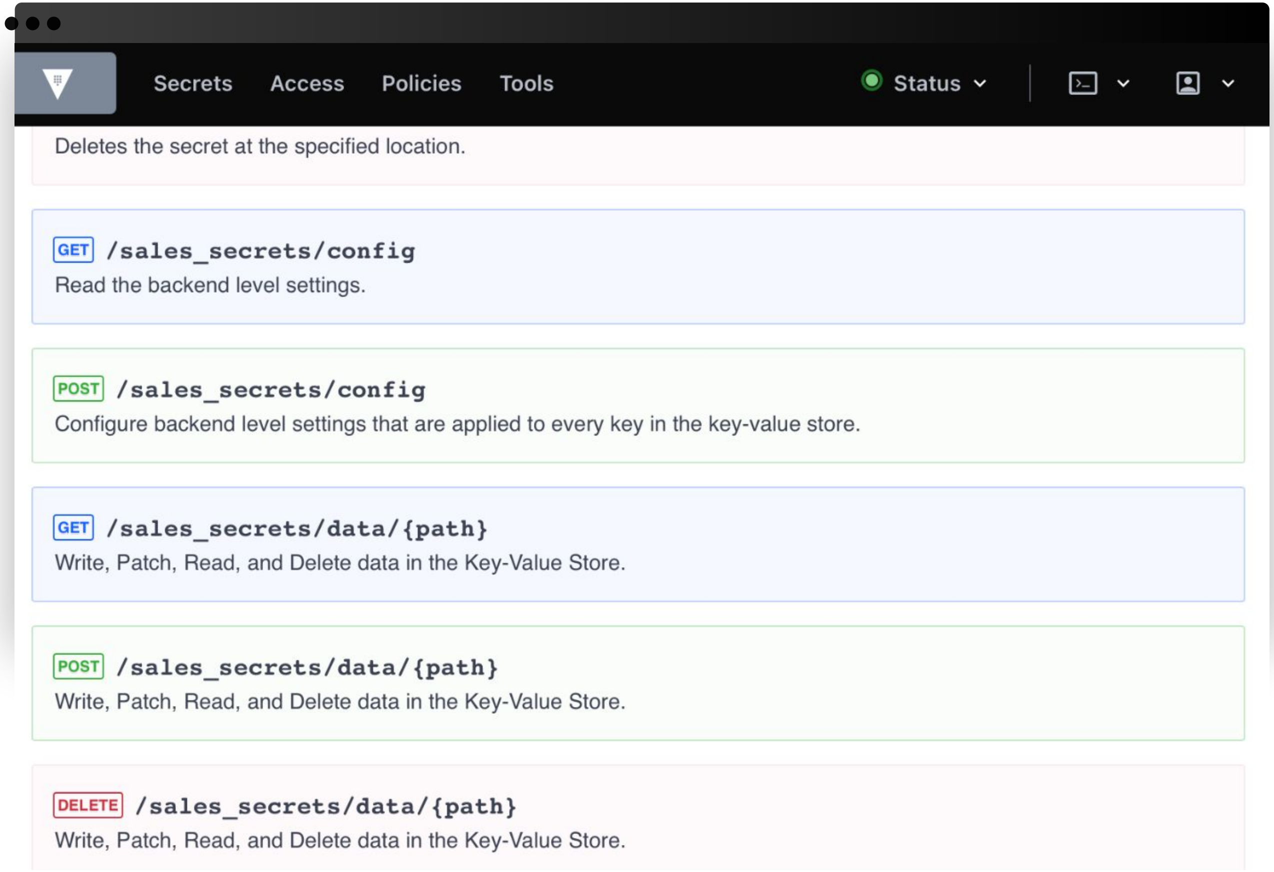
Feature rich API provides full access to Vault and every aspect of Vault can be controlled via this method



```
$ 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



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.

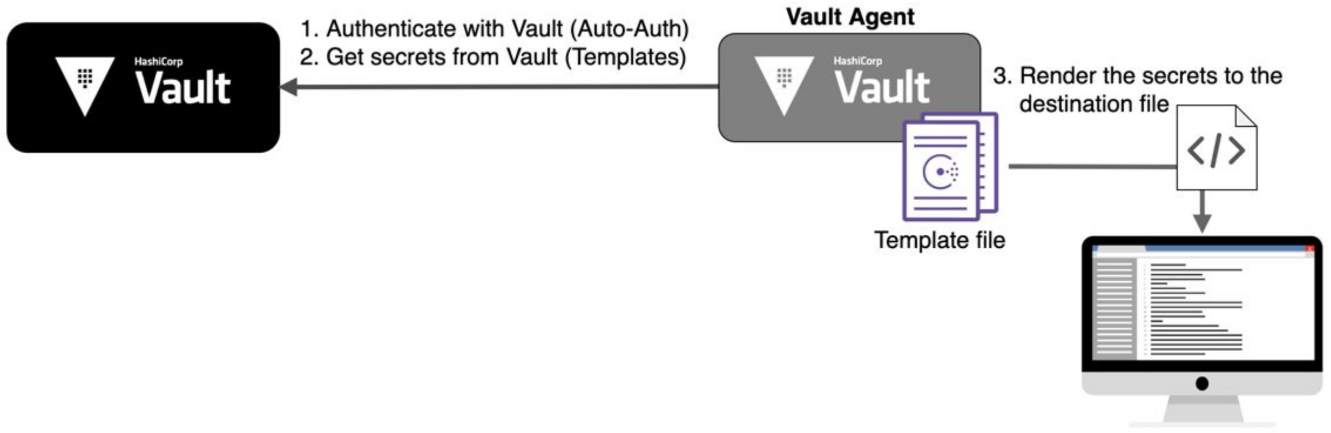
GET /sales_secrets/data/{path}
Write, Patch, Read, and Delete data in the Key-Value Store.

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.

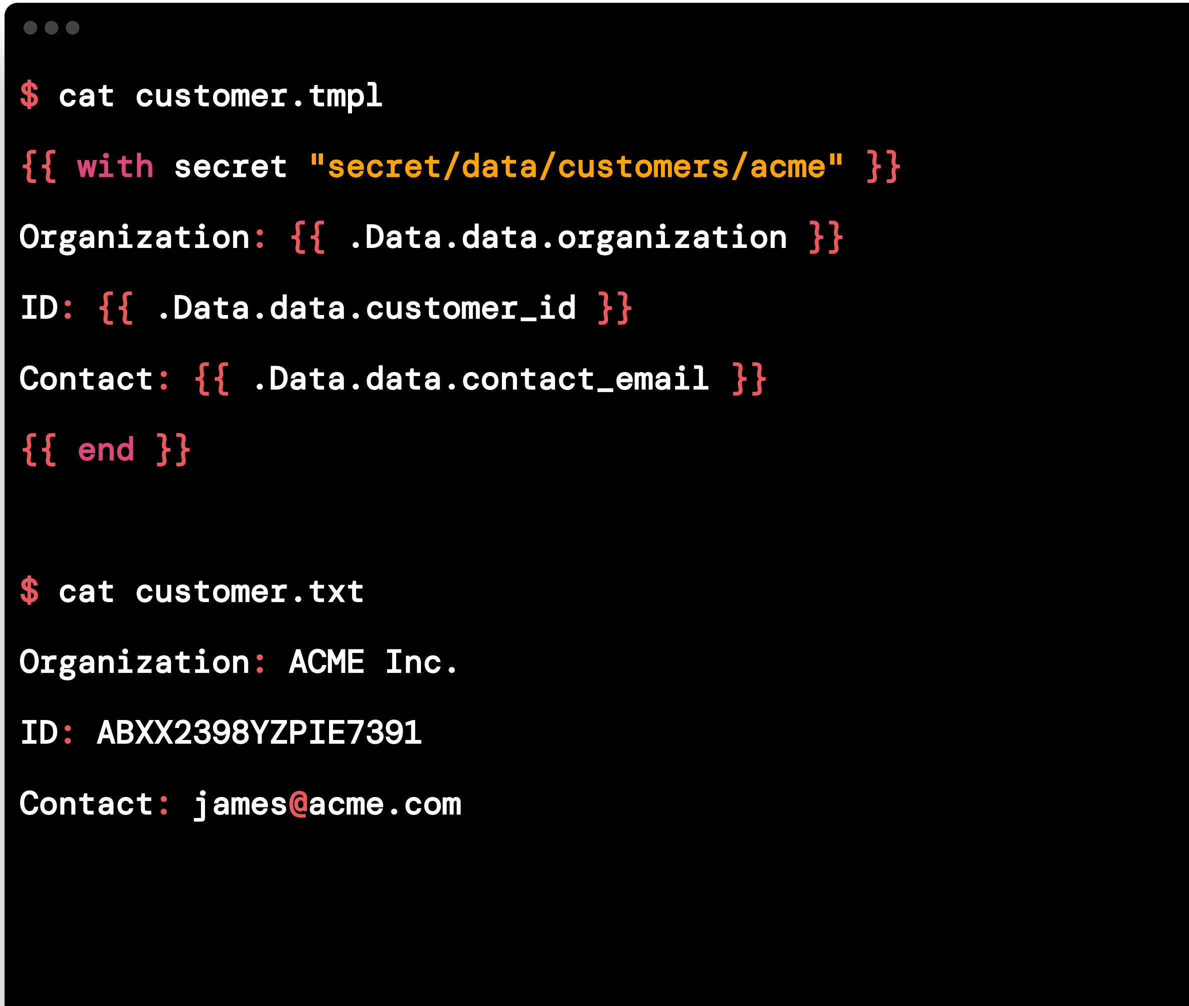
Vault Agent Templating

- 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


HashiCorp Vault
1. Authenticate with Vault (Auto-Auth)
2. Get secrets from Vault (Templates)
Vault Agent
HashiCorp Vault
3. Render the secrets to the destination file
Template file

Vault Agent Templating

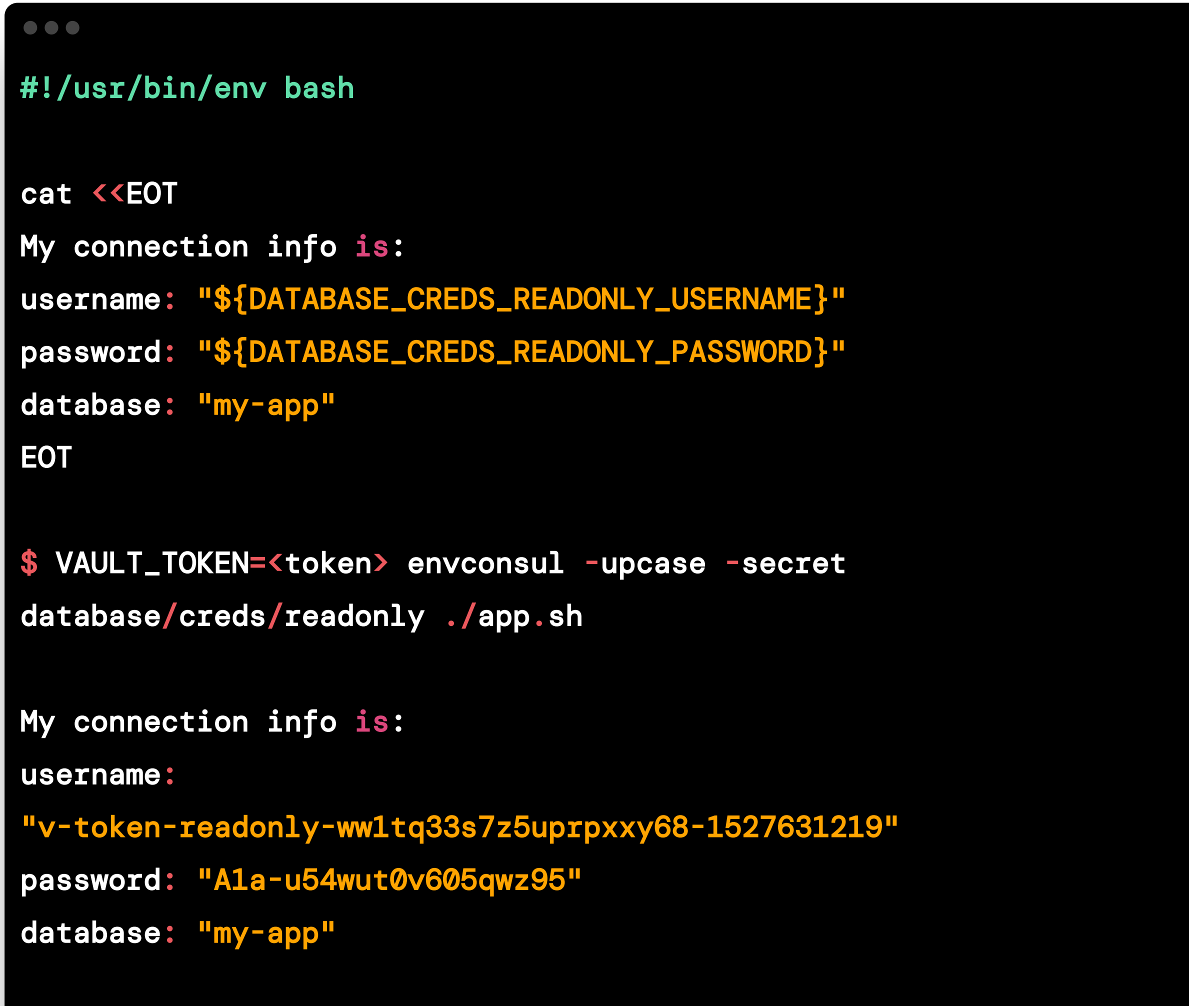
Example Template



```
$ cat customer.tmpl
{{ 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

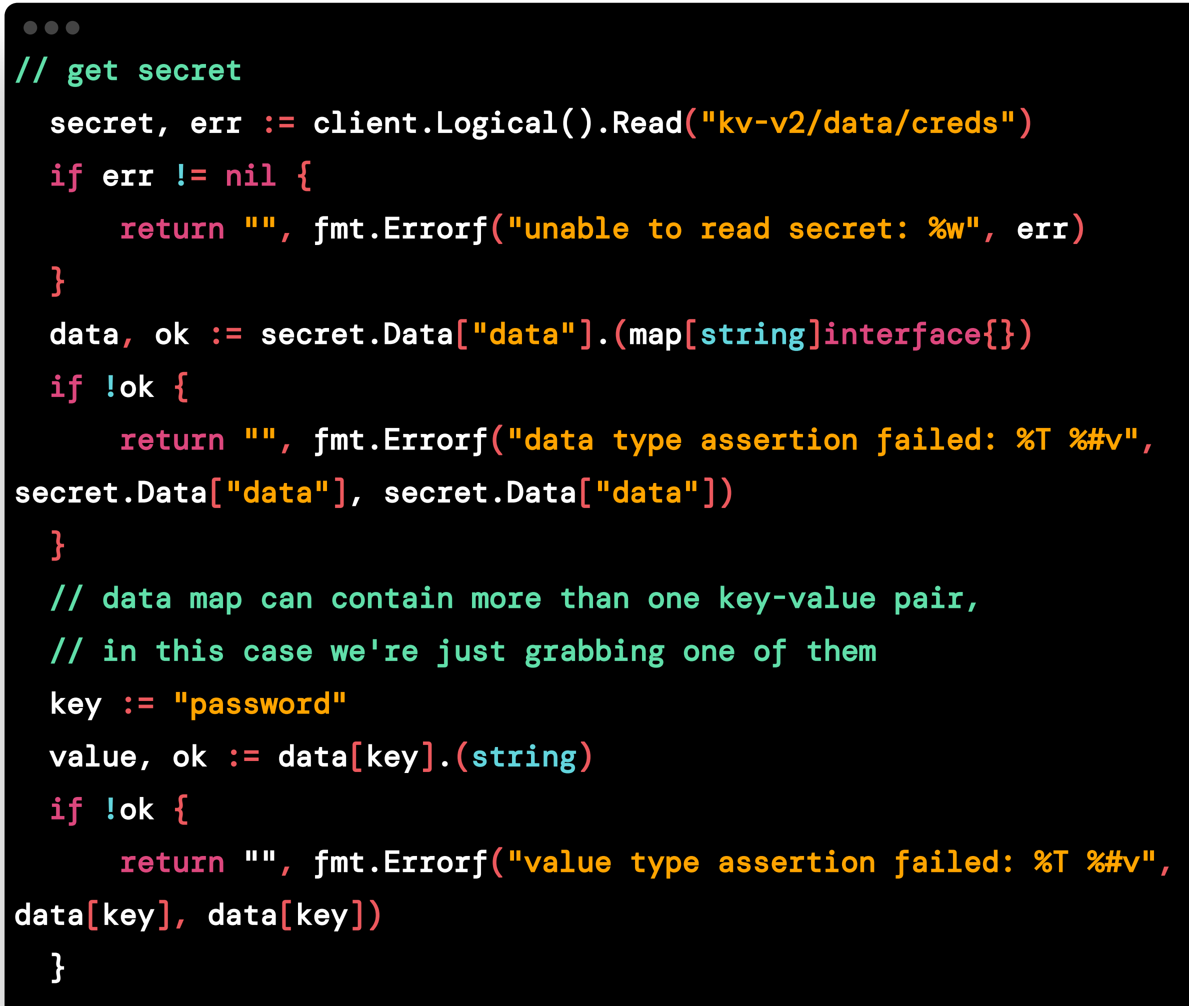
A subprocess which dynamically populates environment variables with secrets read from Vault making them available to applications



```
#!/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]



```go
// 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

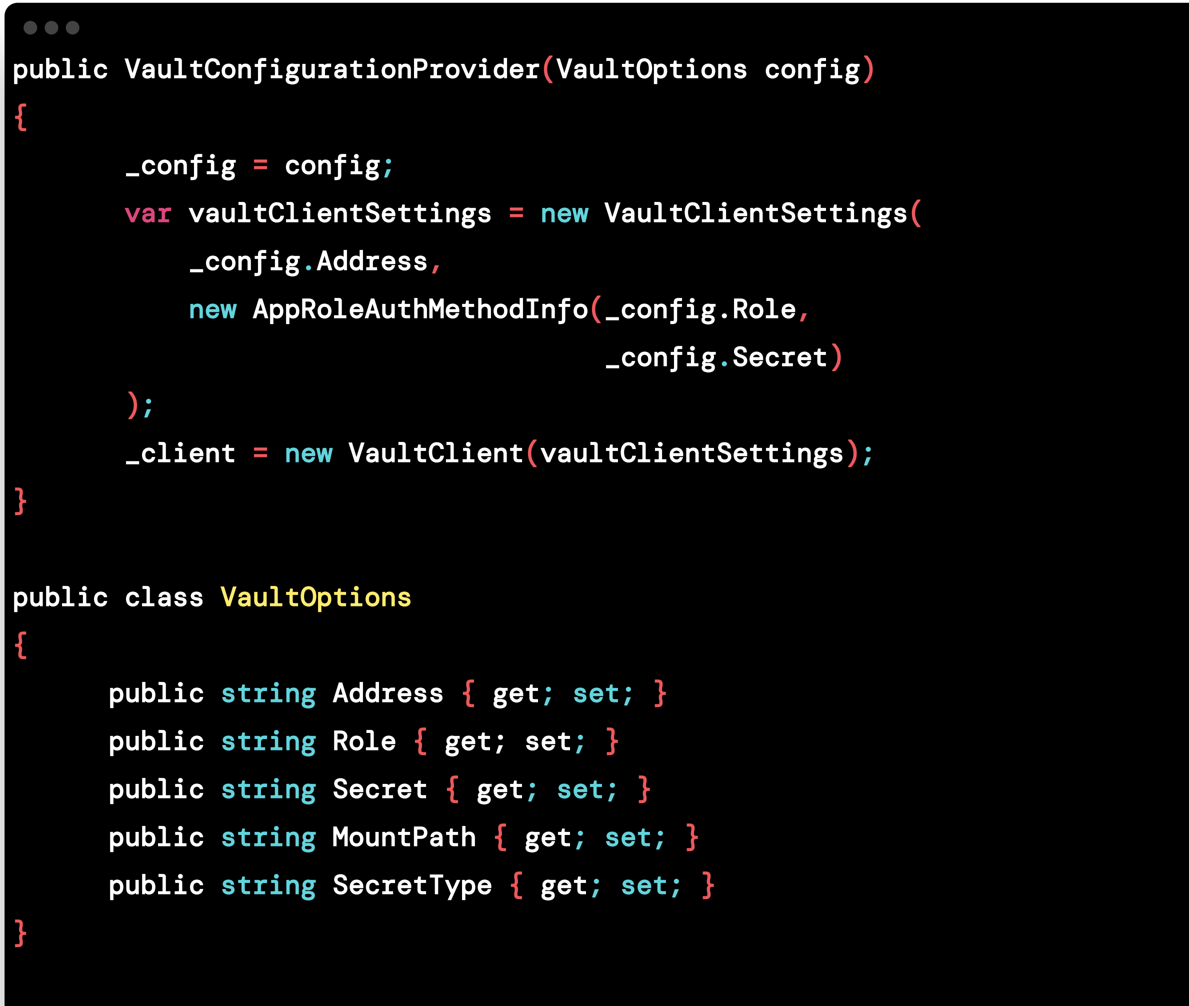
[Spring Cloud Vault](#)
[Java Application Demo](#)

```
@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

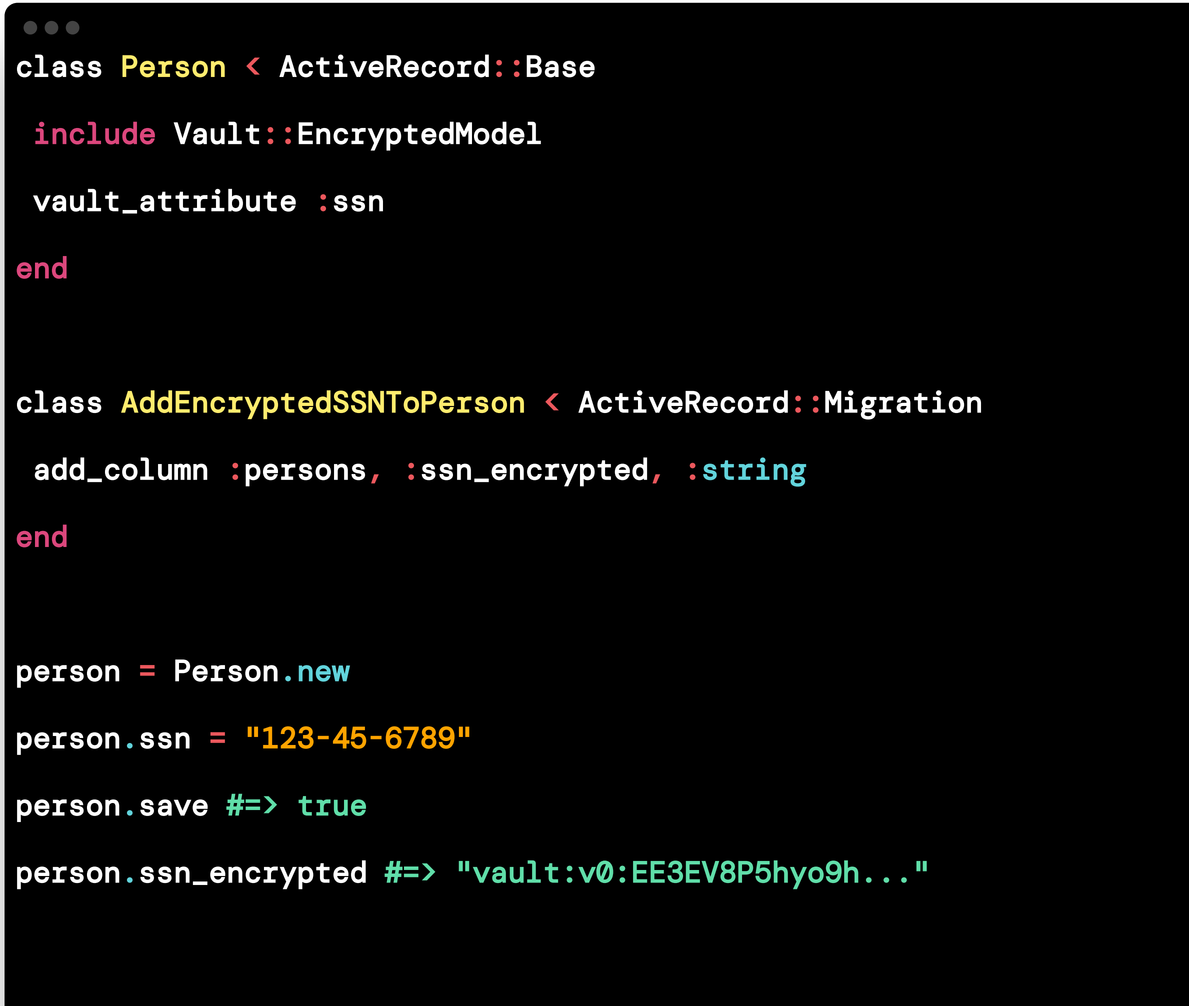


```
public VaultConfigurationProvider(VaultOptions config)
{
        _config = config;
        var vaultClientSettings = new VaultClientSettings(
            _config.Address,
            new AppRoleAuthMethodInfo(_config.Role,
                                      _config.Secret)
        );
        _client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

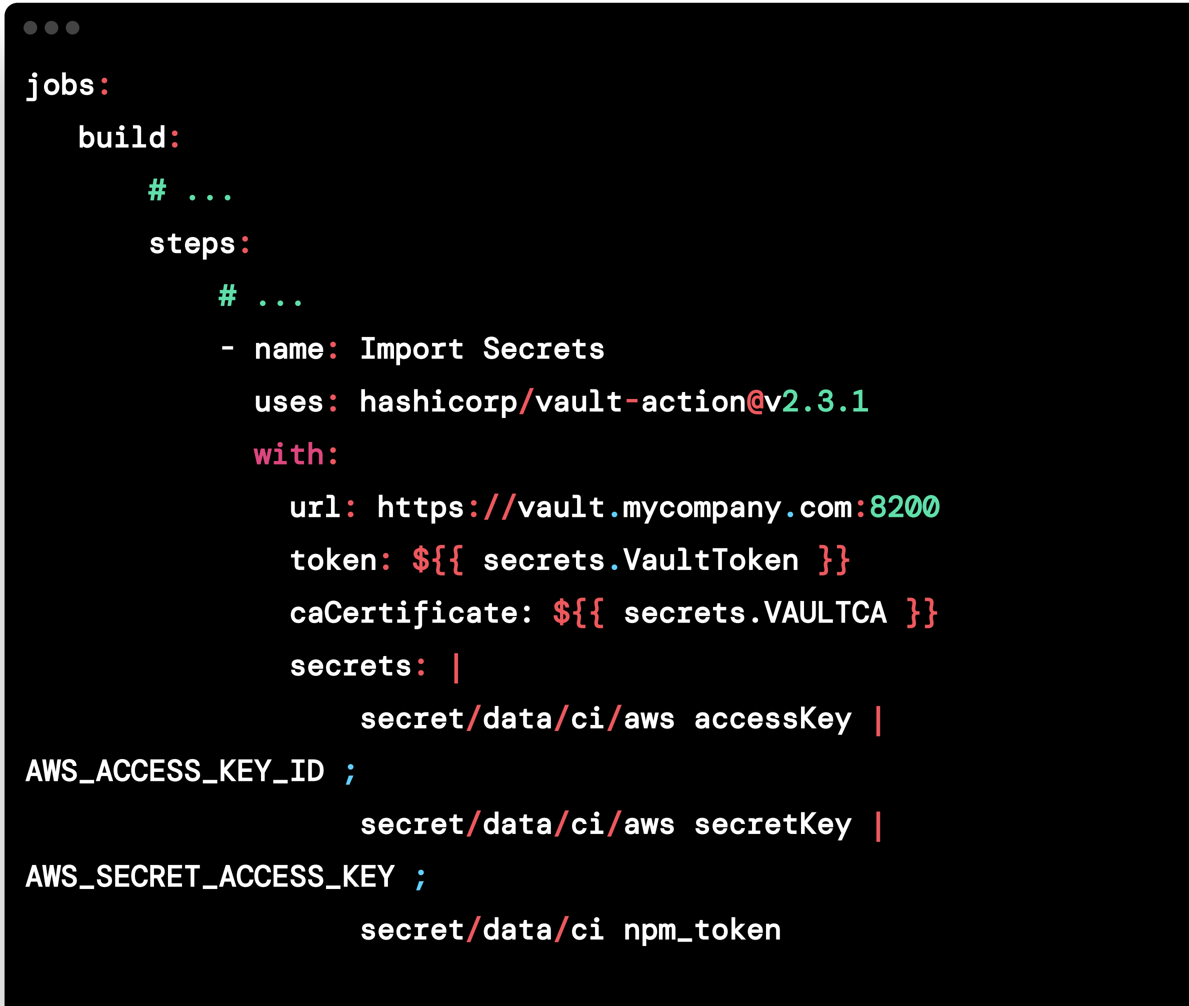


```
class Person < ActiveRecord::Base
 include Vault::EncryptedModel
 vault_attribute :ssn
end


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

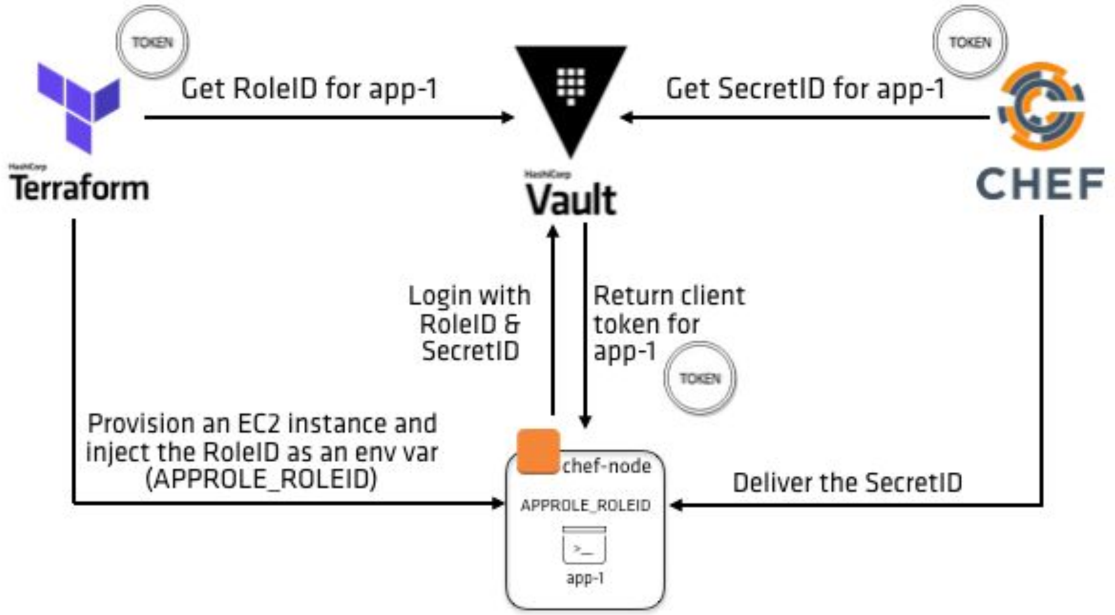


```
jobs:
   build:
       # ...
       steps:
           # ...
           - name: Import Secrets
             uses: hashicorp/vault-action@v2.3.1
             with:
               url: https://vault.mycompany.com:8200
               token: ${{ secrets.VaultToken }}
               caCertificate: ${{ 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

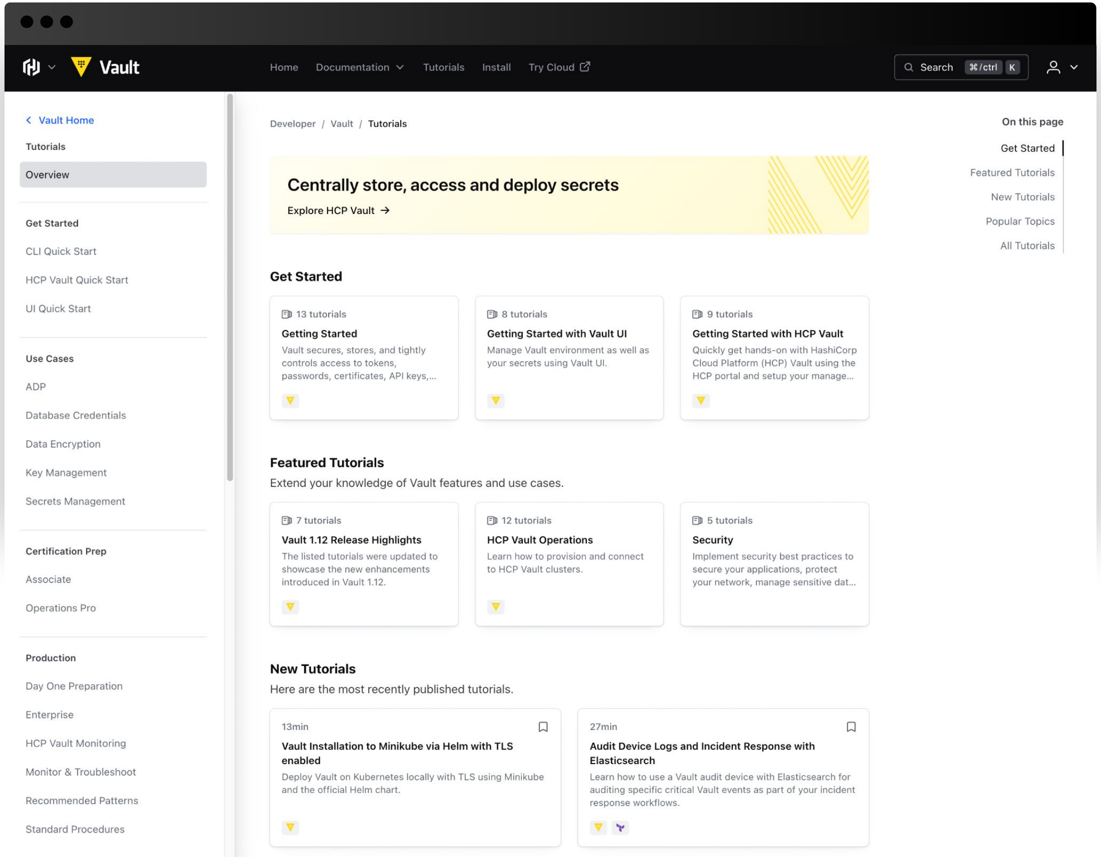
[AppRole With Terraform & Chef | Vault]


TOKEN
Terraform
Get RoleID for app-1
Vault
Get SecretID for app-1
TOKEN
CHEF
Login with RoleID & SecretID
Return client token for app-1
TOKEN
Provision an EC2 instance and inject the RoleID as an env var (APPROLE_ROLEID)
chef-node
APPROLE_ROLEID
app-1
Deliver the SecretID

Next Steps

Tutorials

Step-by-step guides to accelerate deployment of Vault



Vault Home

Tutorials

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

Developer / Vault / Tutorials

Centrally store, access and deploy secrets

Explore HCP Vault →

Get Started

13 tutorials
Getting Started
Vault secures, stores, and tightly controls access to tokens, passwords, certificates, API keys,...

8 tutorials
Getting Started with Vault UI
Manage Vault environment as well as your secrets using Vault UI.

9 tutorials
Getting Started with HCP Vault
Quickly get hands-on with HashiCorp Cloud Platform (HCP) Vault using the HCP portal and setup your manage...

Featured Tutorials

Extend your knowledge of Vault features and use cases.

7 tutorials
Vault 1.12 Release Highlights
The listed tutorials were updated to showcase the new enhancements introduced in Vault 1.12.

12 tutorials
HCP Vault Operations
Learn how to provision and connect to HCP Vault clusters.

5 tutorials
Security
Implement security best practices to secure your applications, protect your network, manage sensitive dat...

New Tutorials

Here are the most recently published tutorials.

13min
Vault Installation to Minikube via Helm with TLS enabled
Deploy Vault on Kubernetes locally with TLS using Minikube and the official Helm chart.

27min
Audit Device Logs and Incident Response with Elasticsearch
Learn how to use a Vault audit device with Elasticsearch for auditing specific critical Vault events as part of your incident response workflows.

On this page

Get Started

Featured Tutorials

New Tutorials

Popular Topics

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 Webinars

Vault Dynamic Secrets

This Lunch & Learn (separate link) covers the best practices for leveraging the power of Vault dynamic secrets engines

Vault Governance

Learn how to implement governance best practices for Vault Enterprise using policy & Sentinel

Program Closing

Asynchronous content that will be delivered to your Inbox

Action Items

- Share to customer.success@hashicorp.com
  - Authorized technical contacts for support
  - Stakeholders contact information (name and email addresses)
- Assess how teams & 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