Module 1: Dynamic Secrets

Dynamic secrets allow you to generate credentials on-demand dynamically and are automatically revoked after a certain duration.

Not all the secrets engine supports dynamic credentials.

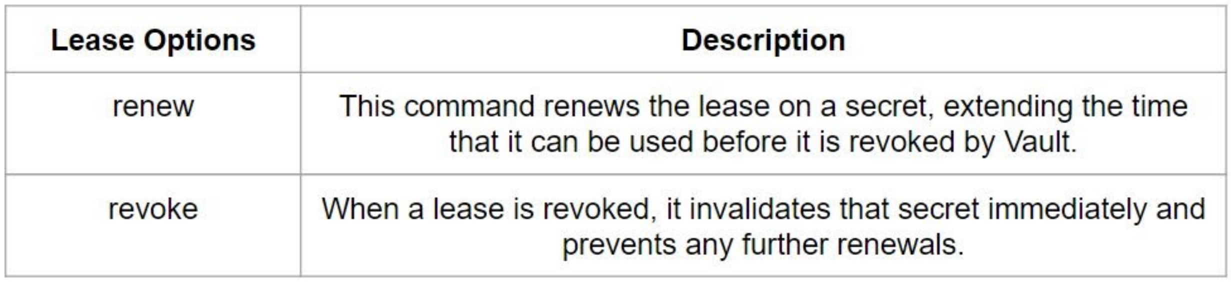
Common Supported Engines: AWS, Database, Google Cloud, Azure.

Dynamic Secret does not provide any stronger cryptographic key generation.

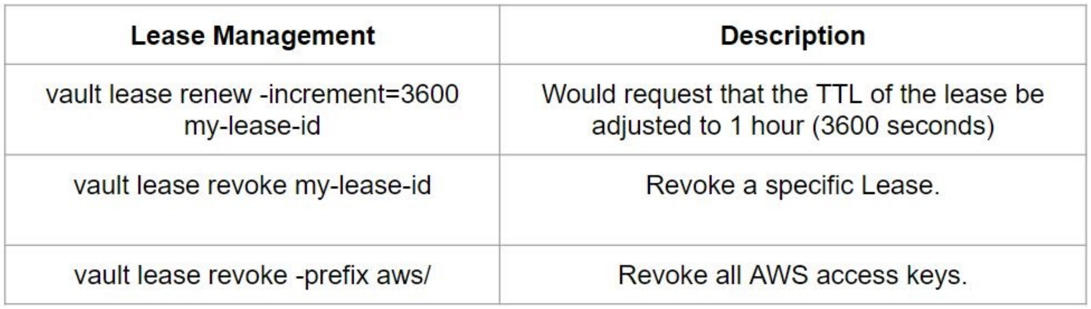
Module 2: Lease Management

With every dynamic secret and service type authentication token, Vault creates a lease: metadata containing information such as a time duration, renewability, and more.

Once the lease is expired, Vault can automatically revoke the data, and the consumer of the secret can no longer be certain that it is valid.



Lease Options	Description
renew	This command renews the lease on a secret, extending the time that it can be used before it is revoked by Vault.
revoke	When a lease is revoked, it invalidates that secret immediately and prevents any further renewals.



Lease Management	Description
vault lease renew -increment=3600 my-lease-id	Would request that the TTL of the lease be adjusted to 1 hour (3600 seconds)
vault lease revoke my-lease-id	Revoke a specific Lease.
vault lease revoke -prefix aws/	Revoke all AWS access keys.

Module 3: Transit Secrets Engine

The transit secrets engine handles cryptographic functions on data-in-transit.

All plaintext data must be base64-encoded.

The reason for this requirement is that Vault does not require that the plaintext is "text". It could be a binary file such as a PDF or image.

We can rotate encryption keys at regular intervals to ensure that not all data is encrypted with just 1 encryption key.

Module 4: Transit Secrets Engine - Key Version

The Transit engine supports the versioning of keys.

Key versions that are earlier than a key's specified min_decryption_version get archived, and the rest of the key versions belong to the working set.

This leads to both performance benefits as well as security benefits.

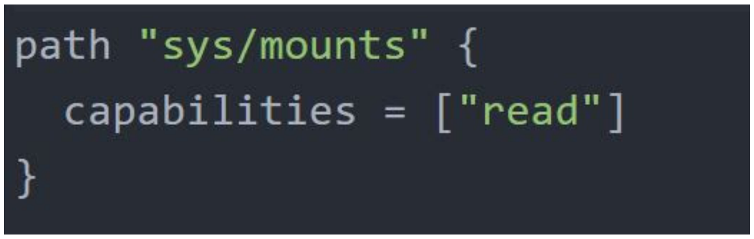
By disallowing decryption of old versions of keys, found ciphertext to obsolete (but sensitive) data can not be decrypted, but in an emergency, the min_decryption_version can be moved back to allow for legitimate decryption.

Module 5: Vault Policies

Vault Policies are used to govern the access of users and roles (authorization)

When you first initialize Vault, the root and default policy gets created by default.

Policies are denied by default, so an empty policy grants no permission in the system.



```
path "sys/mounts" {
  capabilities = ["read"]
}
```

Module 6: Default Policy

The default policy is a built-in Vault policy that cannot be removed.

By default, it is attached to all tokens, but can be explicitly excluded at token creation time by supporting authentication methods.

The policy contains basic functionality such as the ability for the token to look up data about itself and to use its cubbyhole data.

However, Vault is not prescriptive about its contents. It can be modified to suit your needs;

Module 7: ROOT Policy

The root policy is a built-in Vault policy that can not be modified or removed.

A root user can do anything within the Vault. As such, it is highly recommended that you revoke any root tokens before running Vault in production.

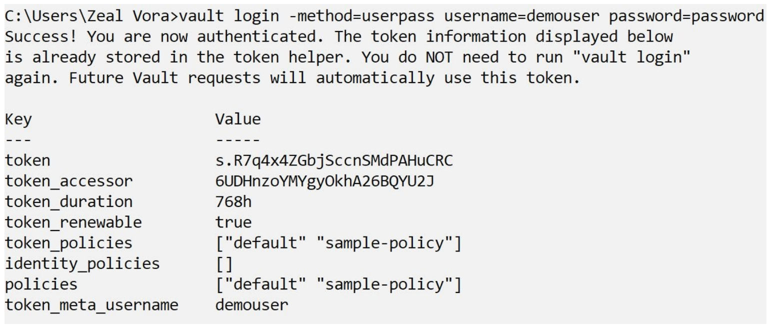
When a Vault server is first initialized, there always exists one root user. This user is used to do the initial configuration and setup of Vault.

After configured, the initial root token should be revoked, and more strictly controlled users and authentication should be used.

Module 8: Token Accessor

This accessor is a value that acts as a reference to a token and can only be used to perform limited actions:

- Lookup a token's properties (not including the actual token ID)
- Lookup a token's capabilities on a path
- Renew the token
- Revoke the token



```
C:\Users\Zeal Vora>vault login -method=userpass username=demouser password=password
Success! You are now authenticated. The token information displayed below
is already stored in the token helper. You do NOT need to run "vault login"
again. Future Vault requests will automatically use this token.

Key                    Value
---                    -----
token                  s.R7q4x4ZGbjSccnSMdPAHuCRC
token_accessor         6UDHnzoYMYgyOkhA26BQYU2J
token_duration         768h
token_renewable        true
token_policies         ["default" "sample-policy"]
identity_policies      []
policies               ["default" "sample-policy"]
token_meta_username    demouser
```

Module 9: Policy Association

During the token creation time, the policy is associated with the token.

At the later stage, if you attach a new policy to the user or role, it will not affect the existing tokens.

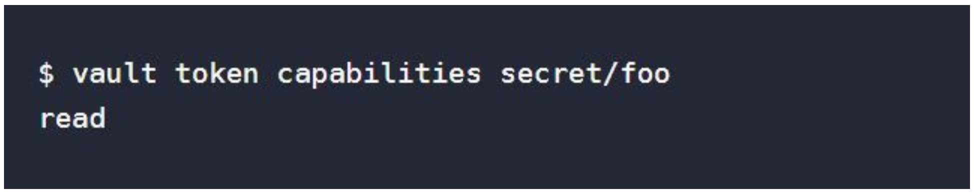
For such cases, a new token must be created.

A policy which is updated and is already attached to the token, the rules will be reflected accordingly as part of the token’s permission.

Module 10: Token Capabilities

The token capabilities command fetches the capabilities of a token for a given path.

Token Accessor can also be used to check the capabilities of a specific token.



```
$ vault token capabilities secret/foo
read
```

Module 11: Authentication Methods

There are multiple authentication methods available in the Vault.

After successful authentication, a token is generated which can be used for interaction with Vault.

Remember that the GitHub auth method is a user-oriented method and is easiest to use for developers’ machines.

For Servers, AppRole method is the recommended choice.

```
$ vault auth enable -path=my-login userpass
```

Module 12: Disabling Auth Method

A specific authentication method can be disabled with the following command:

vault auth disable <method-name>

When an auth method is disabled, all users authenticated via that method are automatically logged out.

Module 13: Storage Backend

Vault has multiple storage backends, each for different use-cases.

Storage backends represent the location for the durable storage of Vault's information.

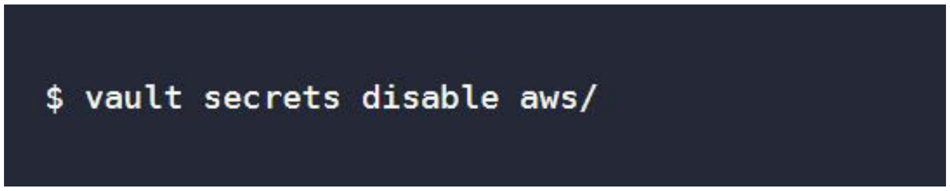
Remember that not all the storage backends support high-availability.

```
storage "dynamodb" {
  ha_enabled =  "true"
  region     =  "us-west-2"
  table      =  "vault-data"
}
```

Module 14: Disabling Secret Engine

The secrets disable command disables a secrets engine at a given PATH.

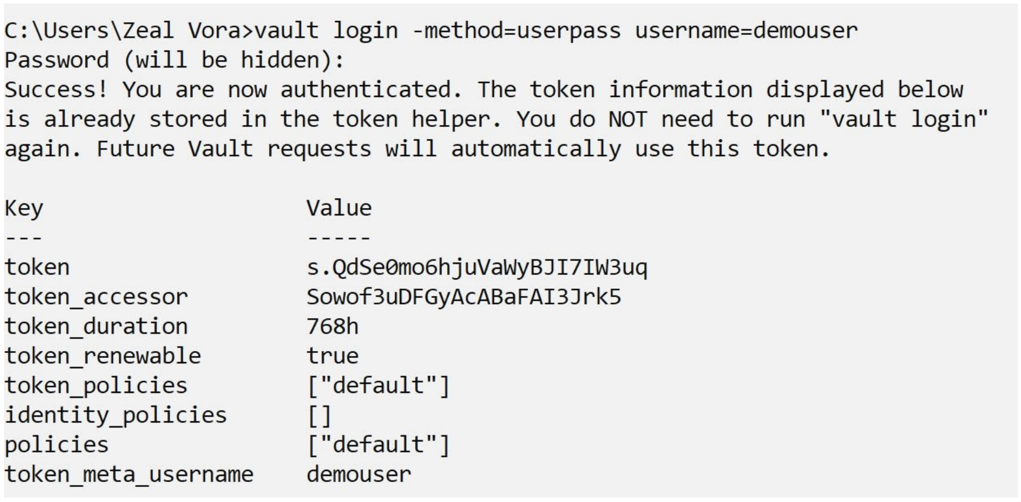
Once a secrets engine is disabled, all secrets generated via the secrets engine are immediately revoked.



```
$ vault secrets disable aws/
```

Module 15: Login based on UserPass Auth Method

While logging in via userpass auth method, it is important to not define the password directly within the CLI command.



```
C:\Users\Zeal Vora>vault login -method=userpass username=demouser
Password (will be hidden):
Success! You are now authenticated. The token information displayed below
is already stored in the token helper. You do NOT need to run "vault login"
again. Future Vault requests will automatically use this token.

Key                    Value
---                    -----
token                  s.QdSe0mo6hjuVaWyBJI7IW3uq
token_accessor         Sowof3uDFGyAcABaFAI3Jrk5
token_duration         768h
token_renewable        true
token_policies         ["default"]
identity_policies      []
policies               ["default"]
token_meta_username    demouser
```

Module 15: Unseal Vault

When a Vault server is started, it starts in a sealed state.

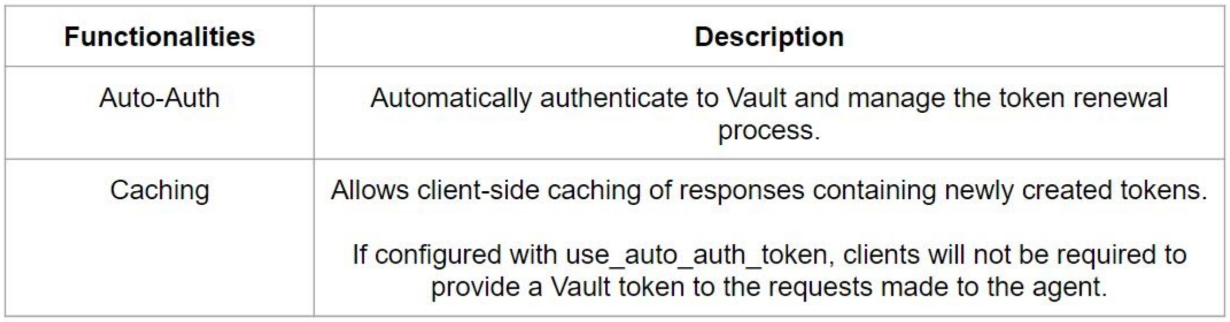
Unsealing is the process of constructing the master key necessary to read the decryption key to decrypt the data, allowing access to the Vault.

Unsealing is the process of reconstructing this master key.

Module 16: Vault Agent

The Agent does not persist anything to storage. Everything lives in memory.

There are two primary functionalities related to Vault Agent:

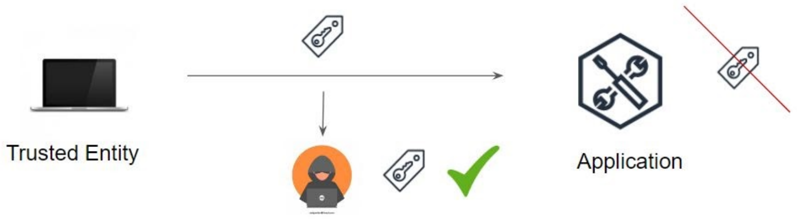


Functionalities	Description
Auto-Auth	Automatically authenticate to Vault and manage the token renewal process.
Caching	Allows client-side caching of responses containing newly created tokens. If configured with use_auto_auth_token, clients will not be required to provide a Vault token to the requests made to the agent.

Module 17: Response Wrapping Token

When response wrapping is requested, Vault creates a temporary single-use token (wrapping token) and insert the response into the token's cubbyhole with a short TTL

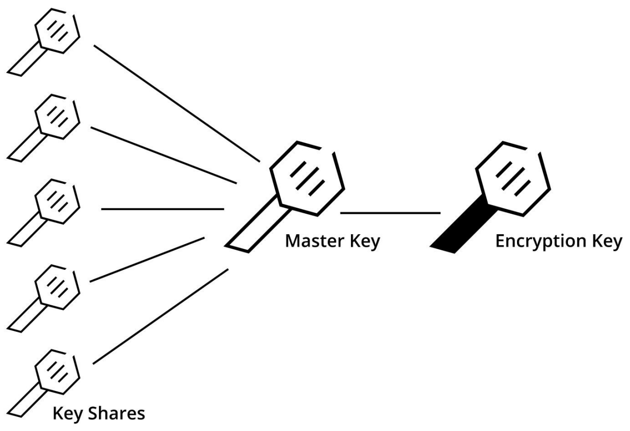
If the wrapping token is compromised and attacker unwraps the secret, the application will not be able to unwrap again and this can sound an alarm and you can revoke things accordingly.



Module 18: Shamir Secret Sharing for Unsealing Vault

The storage backend is considered to be untrusted and Vault uses an encryption key which is used to protect all the data. That key is protected by a master key.

Vault uses a technique known as Shamir's secret sharing algorithm to split the master key into 5 shares, any 3 of which are required to reconstruct the master key.

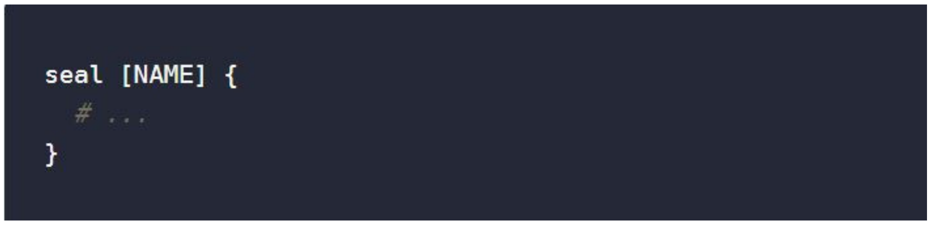


The best practice states that the key shares that are entered to get the master key should be done from a different workstation by different users having an individual key.

Module 19: Seal Stanza

The seal stanza configures the seal type to use for additional data protection, such as using HSM or Cloud KMS solutions to encrypt and decrypt the master key.

This stanza is optional, and in the case of the master key, Vault will use the Shamir algorithm to cryptographically split the master key if this is not configured.



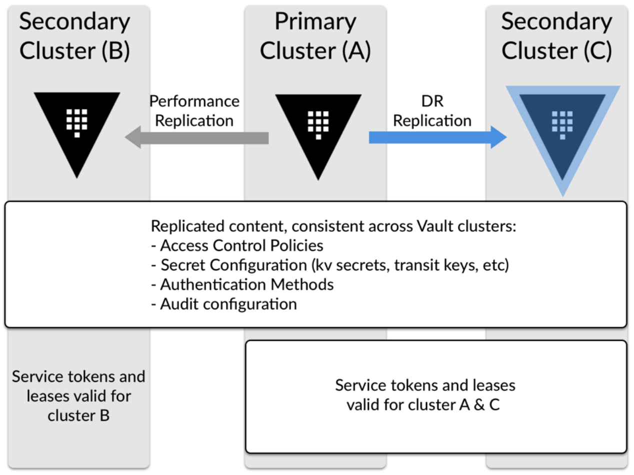
```
seal [NAME] {
  # ...
}
```

Module 20: Vault Replication

For performance replication, secondary clusters will service reads locally.

Some data is also stored locally and not replicated to the primary cluster.

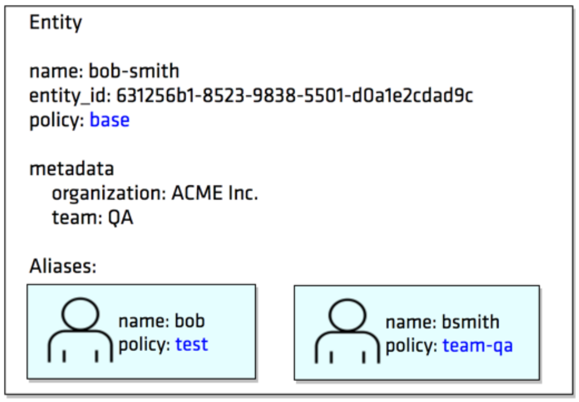
In DR, all the data is replicated but the secondary cluster cannot accept the client requests.


Secondary Cluster (B)
Primary Cluster (A)
Secondary Cluster (C)
Performance Replication
DR Replication
Replicated content, consistent across Vault clusters:
- Access Control Policies
- Secret Configuration (kv secrets, transit keys, etc)
- Authentication Methods
- Audit configuration
Service tokens and leases valid for cluster B
Service tokens and leases valid for cluster A & C

Module 21: Entities and Aliases

Each client is internally termed as an Entity. An entity can have multiple Aliases.

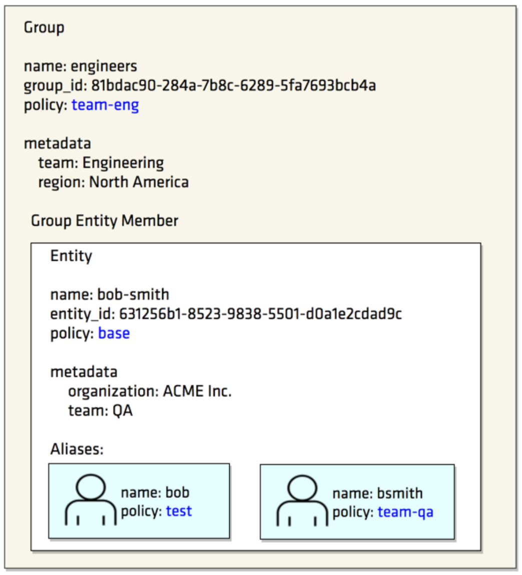
The policy defined at Entity level are associated with the aliases.



Module 22: Identity Groups

A group can contain multiple entities as its members.

Policies set on the group is granted to all members of the group.



Group

name: engineers
group_id: 81bdac90-284a-7b8c-6289-5fa7693bcb4a
policy: team-eng

metadata
team: Engineering
region: North America

Group Entity Member

Entity

name: bob-smith
entity_id: 631256b1-8523-9838-5501-d0a1e2cdad9c
policy: base

metadata
organization: ACME Inc.
team: QA

Aliases:

name: bob
policy: test

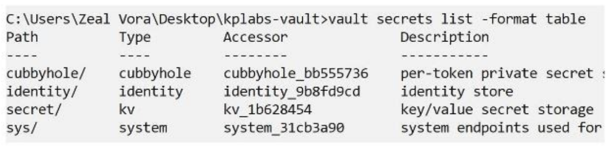
name: bsmith
policy: team-qa

Module 23: Vault Output

We can have Vault output in the following mode: Table, JSON, YAML

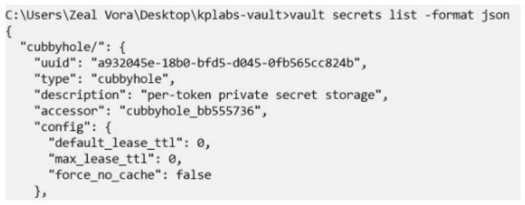
The default is the Table.

Table View



```
C:\Users\Zeal Vora\Desktop\kplabs-vault>vault secrets list -format table
Path          Type         Accessor              Description
----          ----         --------              -----------
cubbyhole/    cubbyhole    cubbyhole_bb555736    per-token private secret
identity/     identity     identity_9b8fd9cd     identity store
secret/       kv           kv_1b628454           key/value secret storage
sys/          system       system_31cb3a90       system endpoints used for
```

JSON View



```
C:\Users\Zeal Vora\Desktop\kplabs-vault>vault secrets list -format json
{
  "cubbyhole/": {
    "uuid": "a932045e-18b0-bfd5-d045-0fb565cc824b",
    "type": "cubbyhole",
    "description": "per-token private secret storage",
    "accessor": "cubbyhole_bb555736",
    "config": {
      "default_lease_ttl": 0,
      "max_lease_ttl": 0,
      "force_no_cache": false
    },
```

Module 24: Multiple Encryption Keys

For Transit Engine, it is considered as a best practice to regularly rotate the encryption key.

This limits the number of data encrypted via a single key.

All the data should not be encrypted with the single encryption key.

The above increases the risk.

Module 25: Reading Output From KV Path

You want to be able to read a specific secret at secret/demosecret. Which capability needs to be used?

There are two primary capabilities:

- LIST
- READ

List allows listing values at the given path.
Read allows reading the data at the given path.

Module 26: Audit Devices

Audit devices are the components in Vault that keep a detailed log of all requests and responses to Vault.

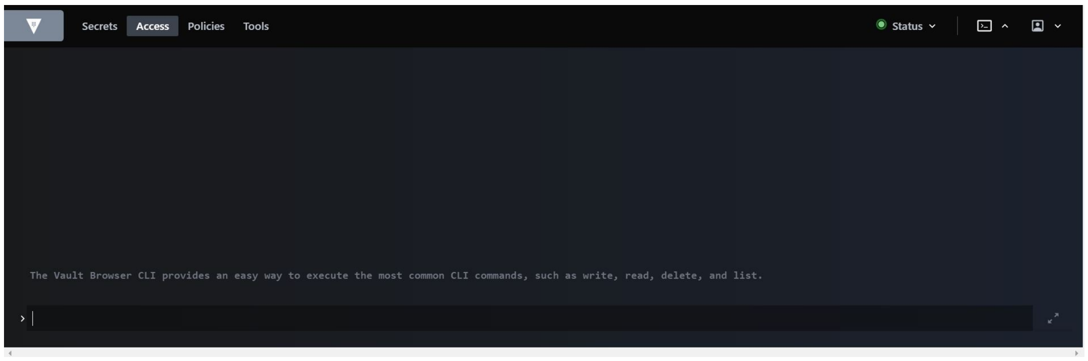
When a Vault server is first initialized, no auditing is enabled.

Audit devices must be enabled by a root user using vault audit enable.

Module 27: Vault Browser CLI

Allows running of basic CLI commands like write, read, delete, and list.

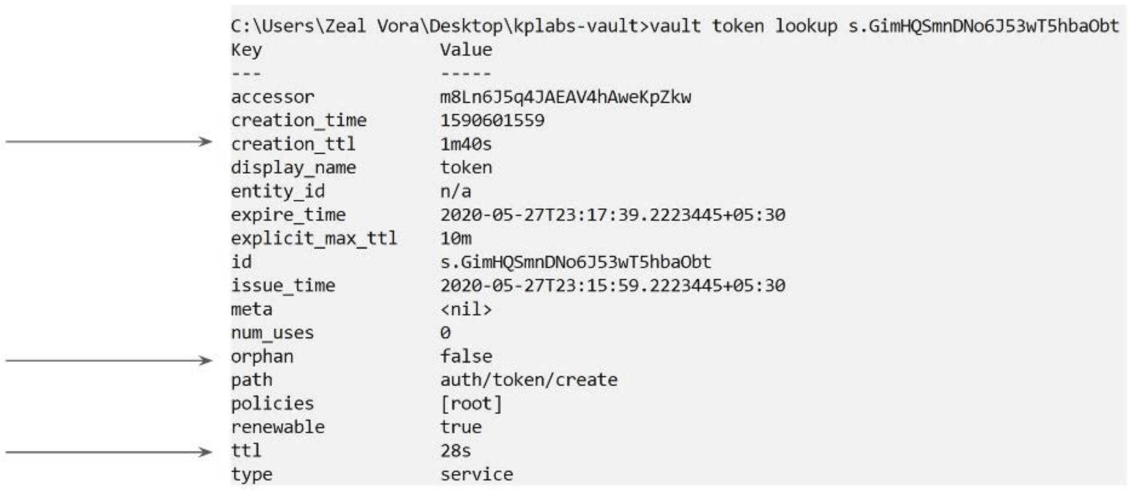
You will not be able to perform various operations like creating new authentication methods and others.



The Vault Browser CLI provides an easy way to execute the most common CLI commands, such as write, read, delete, and list.

Module 28: Vault Token Lookup

Be familiar with the output of the vault token lookup.



```
C:\Users\Zeal Vora\Desktop\kplabs-vault>vault token lookup s.GimHQSmnDNo6J53wT5hbaObt
Key                 Value
---                 -----
accessor            m8Ln6J5q4JAEAV4hAweKpZkw
creation_time       1590601559
creation_ttl        1m40s
display_name        token
entity_id           n/a
expire_time         2020-05-27T23:17:39.2223445+05:30
explicit_max_ttl    10m
id                  s.GimHQSmnDNo6J53wT5hbaObt
issue_time          2020-05-27T23:15:59.2223445+05:30
meta                <nil>
num_uses            0
orphan              false
path                auth/token/create
policies            [root]
renewable           true
ttl                 28s
type                service
```

Module 29: Orphan Tokens

Orphan tokens are not children of their parents; therefore, orphan tokens do not expire when their parent does.

They are the root of their own token tree.

Orphan tokens still expire when their own max TTL is reached.

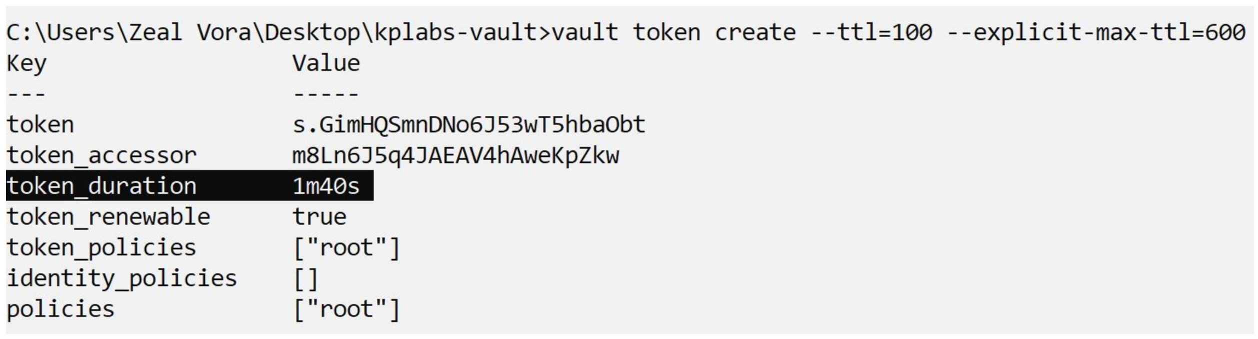
Token 1
(Parent Token)
Orphaned Token 1

Module 30: Create a Token with Explicit TTL

TTL = Initial TTL to associate with the token.

Explicit Max TTL = Maximum lifetime for the token.
Hard limits and cannot be exceeded.

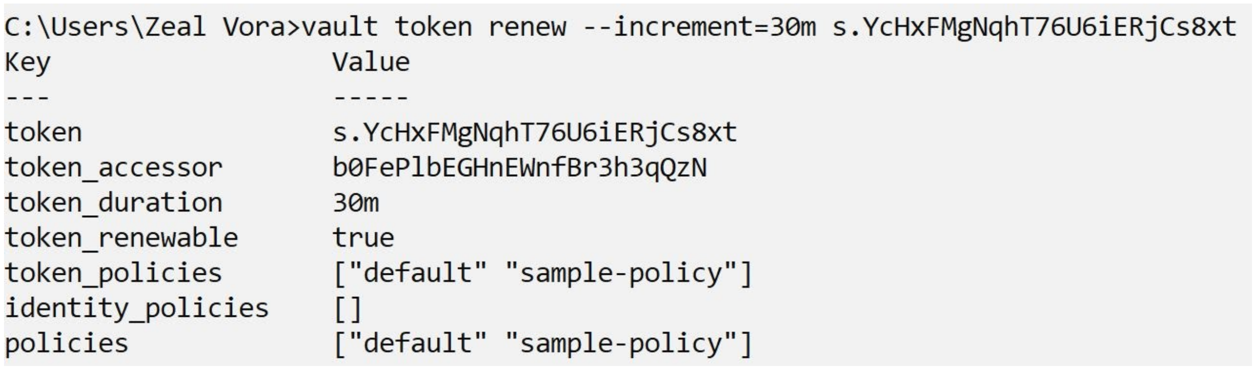
The system max TTL, which is 32 days but can be changed in the Vault's configuration file.



```
C:\Users\Zeal Vora\Desktop\kplabs-vault>vault token create --ttl=100 --explicit-max-ttl=600
Key                  Value
---                  -----
token                s.GimHQSmnDNo6J53wT5hbaObt
token_accessor       m8Ln6J5q4JAEAV4hAweKpZkw
token_duration       1m40s
token_renewable      true
token_policies       ["root"]
identity_policies    []
policies             ["root"]
```

Module 31: Renewing a Token

vault token renew command can be used to extend the validity of the renewable tokens.



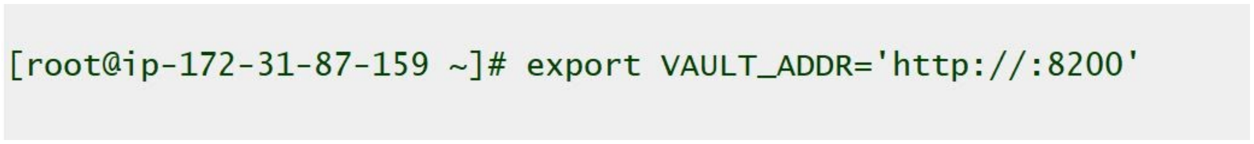
```
C:\Users\Zeal Vora>vault token renew --increment=30m s.YcHxFMgNqhT76U6iERjCs8xt
Key                     Value
---                     -----
token                   s.YcHxFMgNqhT76U6iERjCs8xt
token_accessor          b0FePlbEGHnEWnfBr3h3qQzN
token_duration          30m
token_renewable         true
token_policies          ["default" "sample-policy"]
identity_policies       []
policies                ["default" "sample-policy"]
```

Module 32: Basic Environment Variables

VAULT_ADDR

Address of the Vault server expressed as a URL and port, for example, https://127.0.0.1:8200/.

http://127.0.0.1:8200 can also be replaced with http://8200

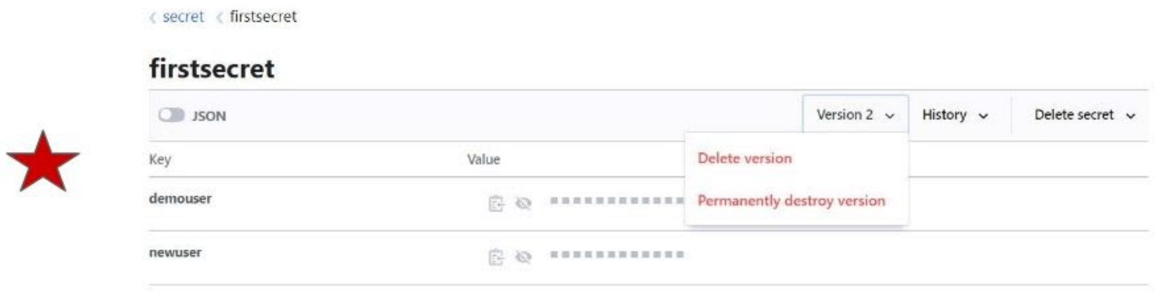


```
[root@ip-172-31-87-159 ~]# export VAULT_ADDR='http://:8200'
```

Module 33: GUI Related Questions

Be aware of how you can do an X step within the GUI.

1. Delete Version 2 of the secret.



Module 34: Secrets Engine

Multiple Secrets Engine of the same type can be enabled at a given time.

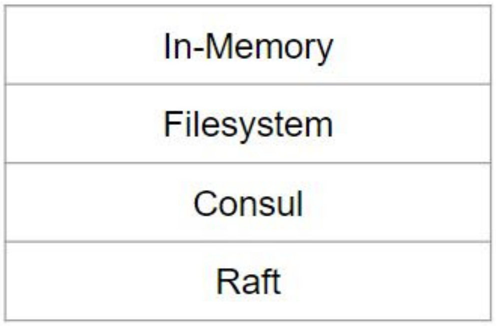
We can distinguish them uniquely by separating them by the path.

k/v secret engine mounted on /secret
kv secret engine mounted on /kv

Module 35: Supported Backends - HashiCorp Support

Following BackEnds are officially supported by HashiCorp.

You will receive technical support for these backends.



In-Memory
Filesystem
Consul
Raft

Module 36: Vault Enterprise Features

Vault Enterprise includes a number of features that may be useful in specific workflows.

It is important for us to understand the unique features that are part of Vault Enterprise.

- Disaster Recovery
- Namespaces
- Monitoring
- Multi-Factor Authentication
- Auto-unseal with HSM

.

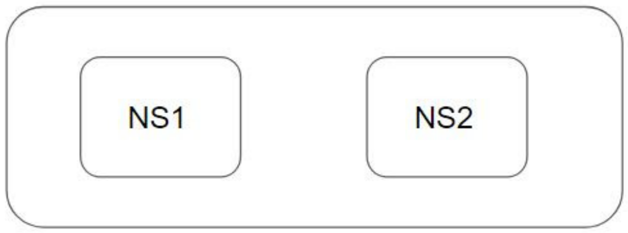
Module 37: Vault Namespaces

Namespaces are isolated environments that functionally exist as "Vaults within a Vault."

They have separate login paths and support creating and managing data isolated to their namespace

Each namespace can have its own:

- Policies
- Auth Methods
- Secrets Engines
- Tokens
- Identity entities and groups



Vault within Vault

Module 38: Vault Replication

Whenever you have replication enabled, all of the secondary clusters existing data will be destroyed. This is irrevocable.

Vault does not support an automatic failover/promotion of a DR secondary cluster

Vault Replication is an Enterprise only feature.

DR replicated cluster will replicate all data from the primary cluster, including tokens. A performance replicated cluster, however, will not replicate the tokens from the primary.

Module 39: Auto-Completion Feature

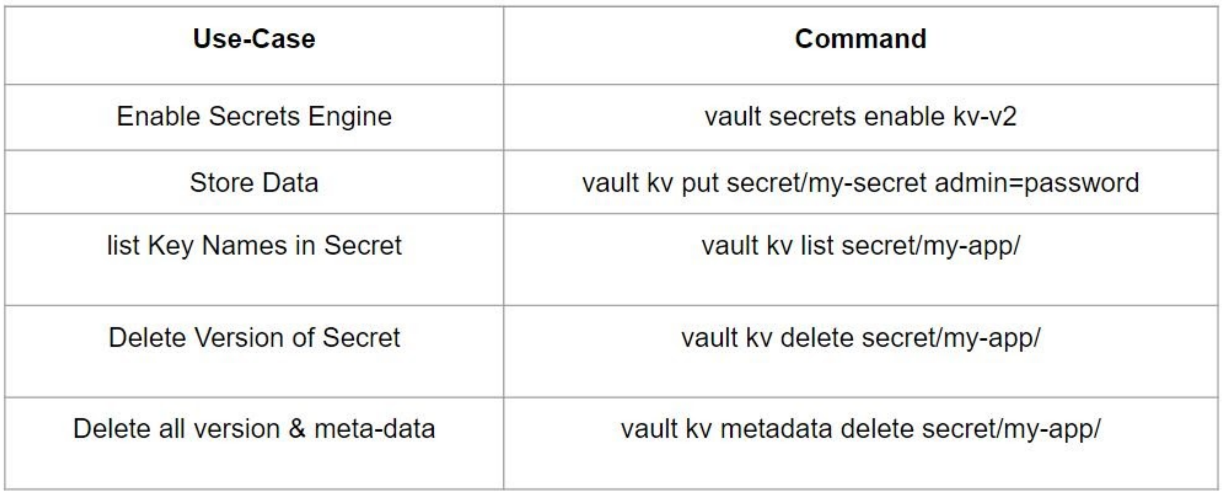
Vault Auto-Complete feature allows automatic completion for flags, subcommands, and arguments.

Be sure to restart your shell after installing autocompletion!

Installation Command:

vault -autocomplete-install

Module 40: Vault CLI Commands



Use-Case	Command
Enable Secrets Engine	vault secrets enable kv-v2
Store Data	vault kv put secret/my-secret admin=password
list Key Names in Secret	vault kv list secret/my-app/
Delete Version of Secret	vault kv delete secret/my-app/
Delete all version & meta-data	vault kv metadata delete secret/my-app/

Module 41: Auto Unseal

Auto Unseal delegates the responsibility of securing the unseal key from users to a trusted device or service

The following are some of the supported services:

- AWS KMS
- Transit Secret Engine
- Azure Key Vault
- HSM
- GCP Cloud KMS

For private connectivity, VPC Endpoints can be used.

Module 42: Identify Output of Transit Engine

```
vault write encryption/encrypt/demo plaintext=$(base64 <<< "Sample data")
```

Output:

```
vault:v3:+tkVyRMMSK+z9Ooa4H0Oa+v2rrMDJL2eouZuVyvInkeIQ+YH0g==
```

- V3 indicates key version 3 was used to encrypt the plain text (3 version of keys exists)
- The name of the keyring is demo
- Transit Secret engine is mounted at /encryption path.

Module 43: PKI Secrets Engine

PKI Secrets Engine generates dynamic X.509 certificates

Benefits of PKI Secrets Engine:

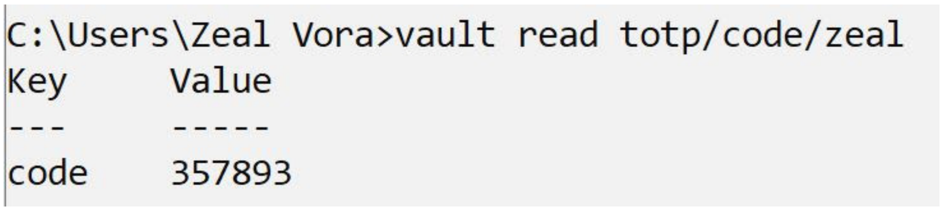
- Vault can act as an Intermediate CA
- Reducing, or eliminating certificate revocations
- Reduces time to get a certificate by eliminating the need to generate a private key and CSR

Module 44: TOTP Secrets Engine

TOTP stands for Time-based one-time passwords

These are temporary passcodes and they typically expire after 30, 60, 120, or 240 seconds.

The TOTP secrets engine can act as both a generator (like Google Authenticator) and a provider (like the Google.com sign-in service).

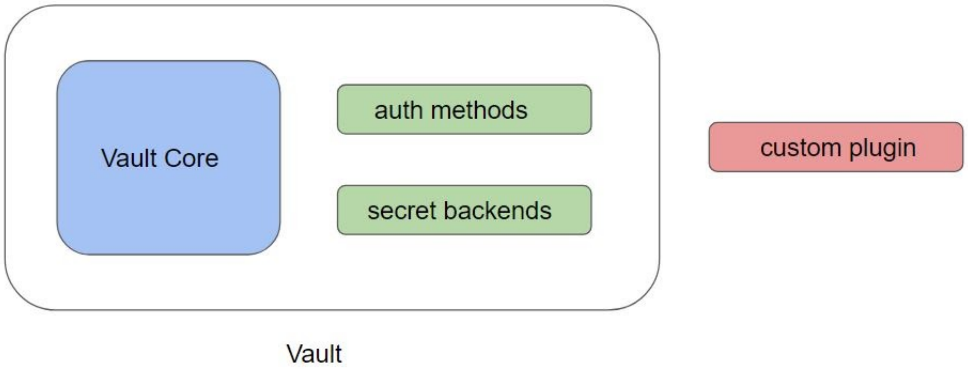


```
C:\Users\Zeal Vora>vault read totp/code/zeal
Key      Value
---      -----
code     357893
```

Module 45: Vault Plugins

All Vault auth and secret backends are considered plugins.

This simple concept allows both built-in and external plugins to be treated like Legos.

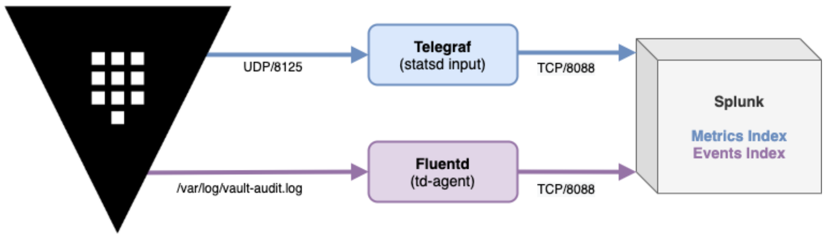


Module 46: Telemetry in Vault

Telemetry related metrics are available at /sys/metrics endpoint.

Important metrics & logs to be analyzed:

- Metrics
- Vault Audit Logs



Module 47: Security Best Practices - Root Tokens

It is generally considered a best practice to not persist root tokens.

 Instead, a root token should be generated using Vault's operator generate-root command only when absolutely necessary

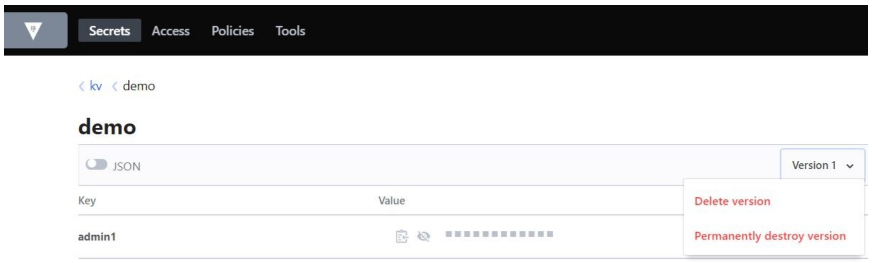
For day-to-day operations, the root token should be deleted after configuring other auth methods.

Module 48: Enabling Versioning in KV - Version 1

When you enable K/V Version 1, the versioning feature is not enabled by default.

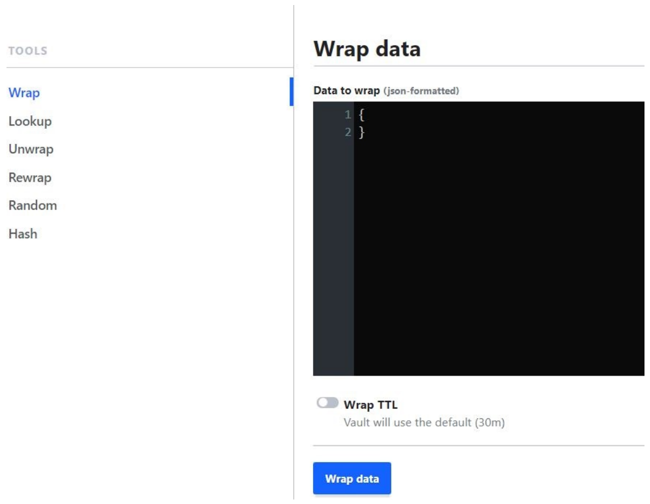
The **kv enable-versioning** command turns on versioning for an existing non-versioned key/value secrets engine (K/V Version 1) at its path.

Note: This command also upgrades the KV from v1 to v2



Module 49: Response Wrapping from UI

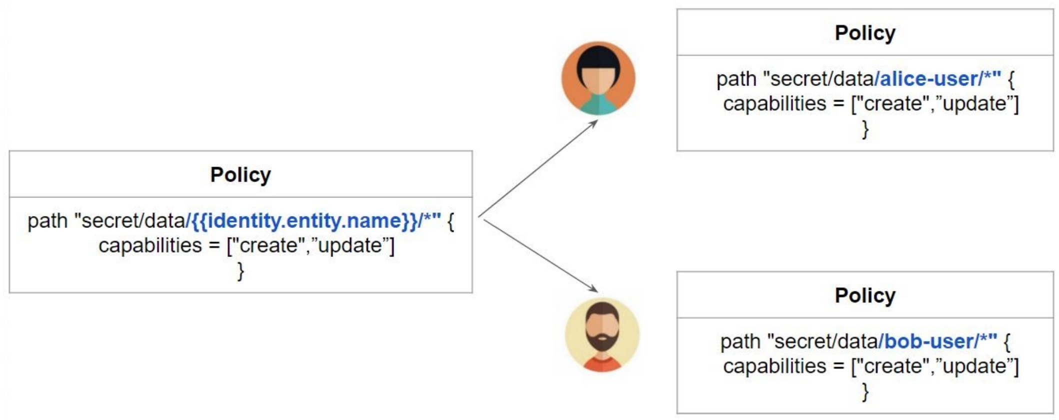
This feature under Vault Tools allows you to achieve response wrapping functionality.



```
{
}
```

Module 50: Path Templating

Path Templating allows variable replacement based on information of the entity.

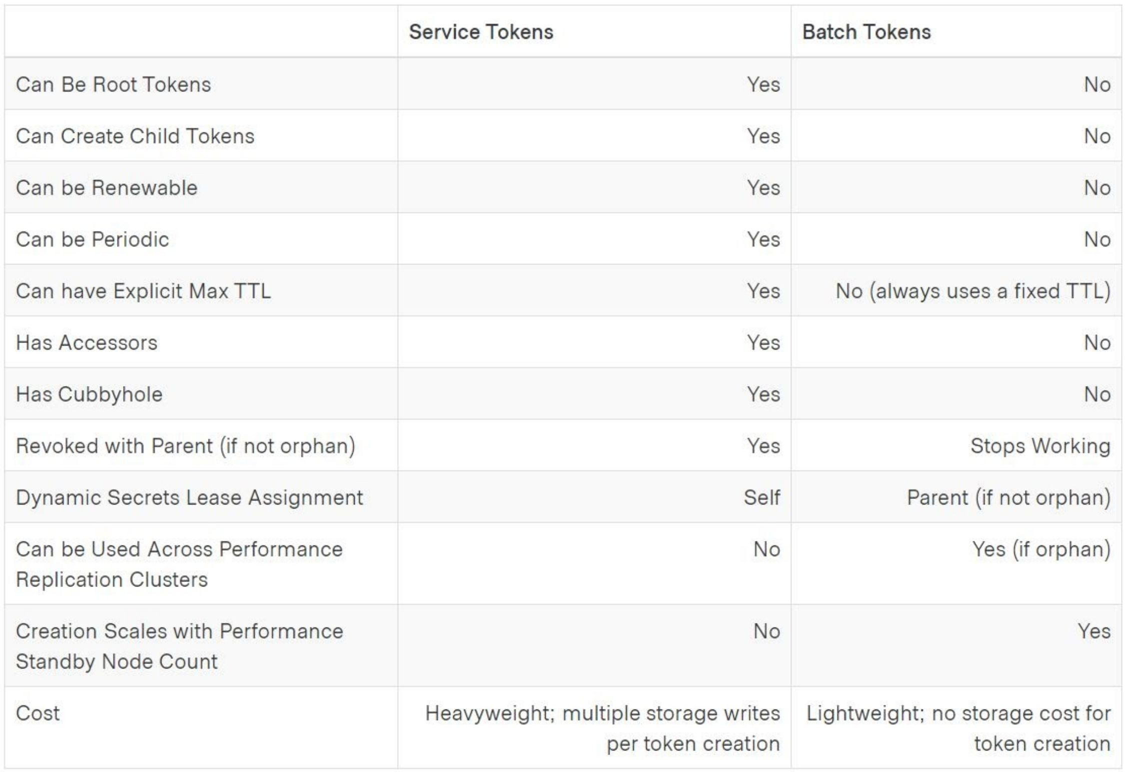


```
path "secret/data/{{identity.entity.name}}/*" {
capabilities = ["create","update"]
}
```

```
path "secret/data/alice-user/*" {
capabilities = ["create","update"]
}
```

```
path "secret/data/bob-user/*" {
capabilities = ["create","update"]
}
```

Module 51: Service Tokens vs Batch Tokens

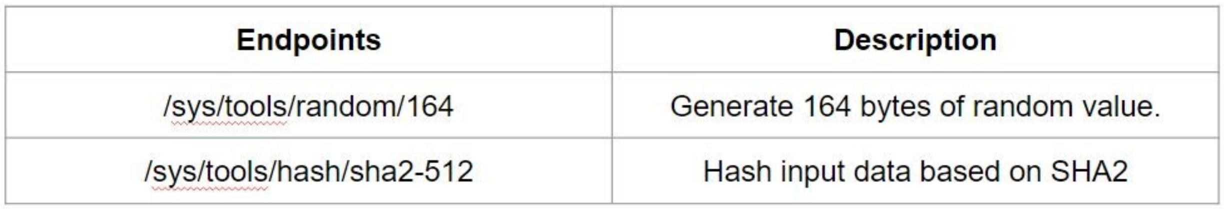


	Service Tokens	Batch Tokens
Can Be Root Tokens	Yes	No
Can Create Child Tokens	Yes	No
Can be Renewable	Yes	No
Can be Periodic	Yes	No
Can have Explicit Max TTL	Yes	No (always uses a fixed TTL)
Has Accessors	Yes	No
Has Cubbyhole	Yes	No
Revoked with Parent (if not orphan)	Yes	Stops Working
Dynamic Secrets Lease Assignment	Self	Parent (if not orphan)
Can be Used Across Performance Replication Clusters	No	Yes (if orphan)
Creation Scales with Performance Standby Node Count	No	Yes
Cost	Heavyweight; multiple storage writes per token creation	Lightweight; no storage cost for token creation

Module 52: Vault Tools

Vault contains a certain set of tools that will allow us to achieve a specific function.

These are also available in the /sys/tools endpoint



Endpoints	Description
/sys/tools/random/164	Generate 164 bytes of random value.
/sys/tools/hash/sha2-512	Hash input data based on SHA2

Module 53: Miscellaneous Pointers - 1

root and default are the two default policies in Vault.

When a lease is revoked, it invalidates that secret immediately and prevents any further renewals.

To remove all secrets at a specific path:

vault lease revoke -prefix <path>

userpass auth method cannot read usernames and passwords from an external source.

Module 54: Miscellaneous Pointers - 2

The **/sys/leader** endpoint is used to check the high availability status and current leader of Vault.

vault operator init command initializes a Vault server

Vault configuration file can be used to configure various settings like cluster name, storage backends, seal settings, and so on. Other things like namespaces, auth methods are directly configured within the vault itself.

The identity secrets engine is mounted by default in Vault.

Storage backends are not trusted by Vault

Module 55: Miscellaneous Pointers - 3

Important Port Numbers:

- 8200 = Vault API and UI
- 8201 - Cluster to Cluster communication

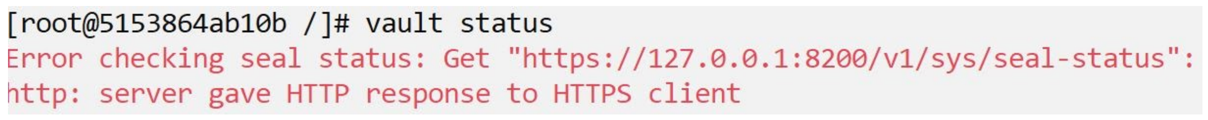
Root tokens in the vault are not associated with TTLs and therefore they do not expire. Non-root tokens are associated with TTLs.

Default Max TTL of tokens is 32 days but it can be modified from the configuration file.

Module 56 Miscellaneous Pointers - 4

Transit Secret Engine does not store any data.

If you receive the following error, make sure to set the VAULT_ADDR variable to HTTP

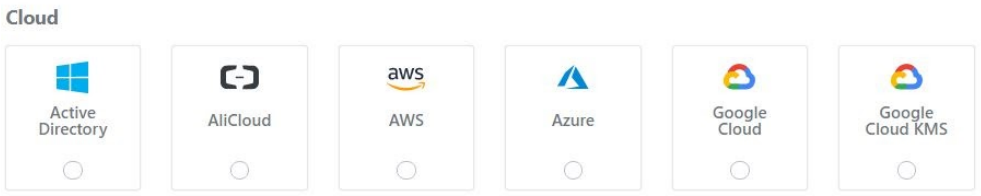


```
[root@5153864ab10b /]# vault status
Error checking seal status: Get "https://127.0.0.1:8200/v1/sys/seal-status":
http: server gave HTTP response to HTTPS client
```

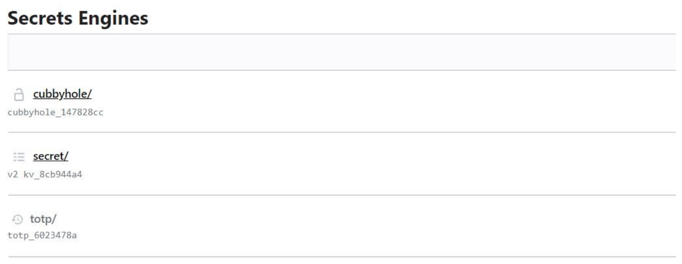
After authentication, the CLI and GUI automatically assume the token for subsequent requests. API on the other hand will require to copy the token and use it for all subsequent requests.

Module 57 Miscellaneous Pointers - 5

Vault secrets engine supports various cloud providers like AWS, Azure, Alibaba Cloud & GCP.



Cubbyhole is a default secrets engine that is enabled for all the users.


Secrets Engines

cubbyhole/
cubbyhole_147828cc

secret/
v2 kv_8cb944a4

totp/
totp_6023478a

Module 58 Miscellaneous Pointers - 6

While generating dynamic secrets, vault returns the lease_id which can in turn be used with commands such as vault lease renew, revoke, and others.

vault login command is used for authentication in CLI.

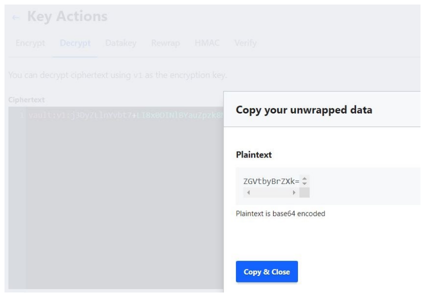
After initializing, Vault provides the root token to the user, this is the only way to log in to Vault to configure additional auth methods.

and * are the two characters that are used for denoting wildcard paths in policies.

The Vault configuration file supports either JSON or HCL

Module 59 Miscellaneous Pointers -7

When data is decrypted via transit engine, the output is in base64 format. To fetch the original plain-text data, you will have to decode the data with base64 -decode



You can decrypt ciphertext using v1 as the encryption key.

```
1 vault:v1:j3DyZLlnYvbt7+LI8x0DINlBYauZpzk8
```

Copy your unwrapped data

Plaintext

```
ZGVtbyBrZXk=
```

Plaintext is base64 encoded

Vault UI needs to be enabled from the configuration file and not the CLI.

In order to renew a token, a user can issue a vault token renew command to extend the TTL

Module 60 - Miscellaneous Pointers - 8

When Vault is sealed, the only two options available are, viewing the vault status and unsealing the Vault. All the other actions performed after the Vault is unsealed and the user is authenticated

If Vault CLI is able to access a specific path (/secret/app/creds) but the user is not able to view it via GUI then the issue is related to missing LIST permission so that the user can browse the path that leads to a specific key-value path.

If the secret has been manually removed, if you try to do a vault lease revoke, it will result in an error stating “credentials could not be found”. In such a case, you need to use the -force flag.

Module 61 - Miscellaneous Pointers -9

HashiCorp regularly updates the exams. This document contains a list of new areas that are covered in the new exams that are not part of this course yet.

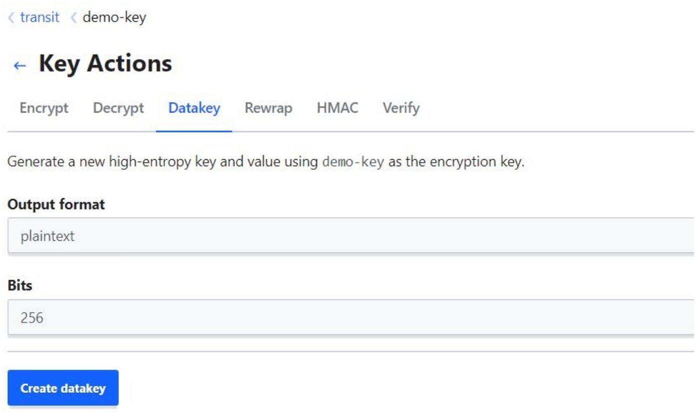
Storage backend and HTTP API are outside of the security barrier hence can't be protected.

WAL stands for Write-Ahead Logging. The changes are first recorded in the log, which must be written to stable storage before the changes are written to the datastore.

Module 62 - Dealing with Larger Data Sizes - Transit Engine

When the data size is large (say 10 GB), we do not want to send it over the network to Vault and get the encrypted data back. It will increase the latency and slow things down.

Instead, you can generate a data key and encrypt it locally and use the same data key to decrypt it locally when needed.



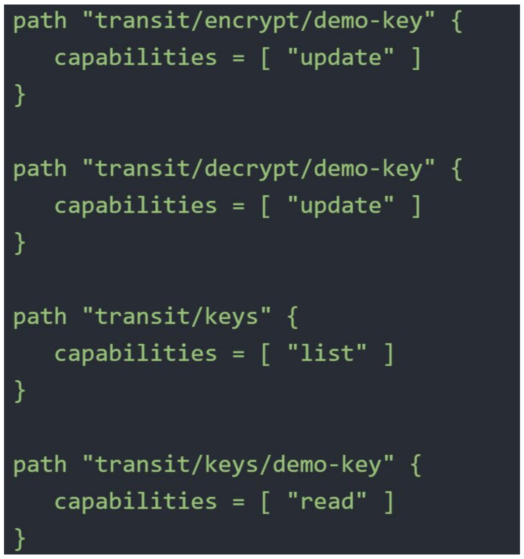
Generate a new high-entropy key and value using demo-key as the encryption key.

Output format: plaintext

Bits: 256

Module 63 - Vault Policy Rules - Transit Engine

Be aware of the important policy rules and capabilities related to transit engine.



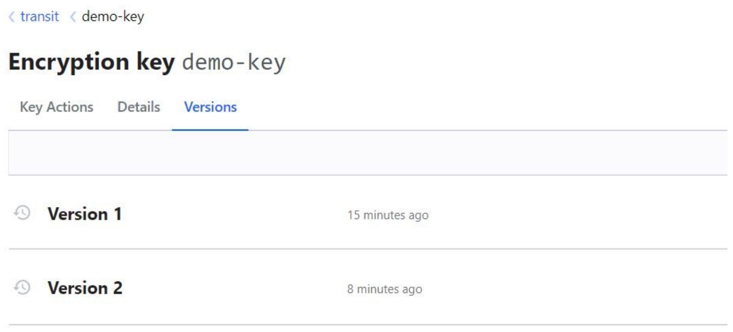
```
path "transit/encrypt/demo-key" {
   capabilities = [ "update" ]
}

path "transit/decrypt/demo-key" {
   capabilities = [ "update" ]
}

path "transit/keys" {
   capabilities = [ "list" ]
}

path "transit/keys/demo-key" {
   capabilities = [ "read" ]
}
```

Module 64 - Key Rotation in Vault

It is not recommended to encrypt all of your data with one encryption key.

Transit Engine allows customers to perform the encryption key rotation.

Vault maintains the versioned keyring and the operator can decide the minimum version allowed for decryption operations.



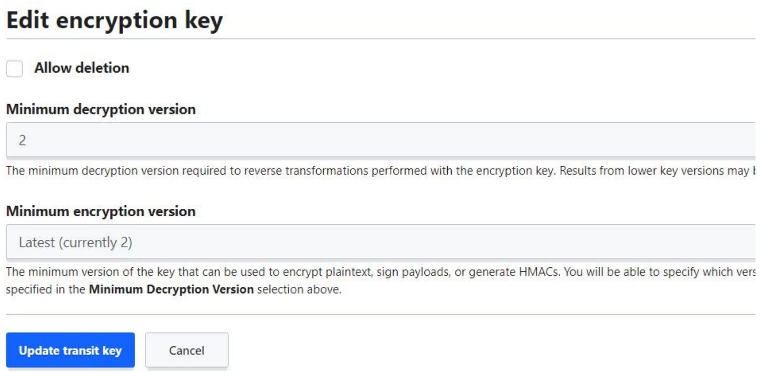
Version 1 15 minutes ago

Version 2 8 minutes ago

Module 65 - Min Decrypt Version

With multiple version of keys, with the help of min_decryption_version , we can plan on which data can get decrypted.

By disallowing decryption of old versions of keys, found ciphertext to obsolete (but sensitive) data can not be decrypted, but in an emergency, the min_decryption_version can be moved back to allow for legitimate decryption.



Edit encryption key

☐ **Allow deletion**

Minimum decryption version

2

The minimum decryption version required to reverse transformations performed with the encryption key. Results from lower key versions may

Minimum encryption version

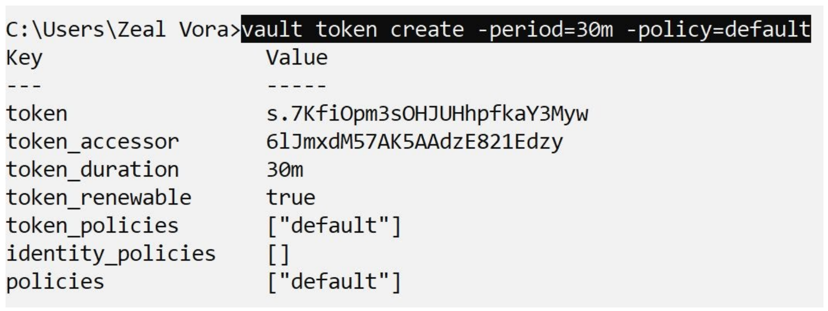
Latest (currently 2)

The minimum version of the key that can be used to encrypt plaintext, sign payloads, or generate HMACs. You will be able to specify which vers
specified in the **Minimum Decryption Version** selection above.

Module 66 - Periodic Tokens

Periodic Tokens never expire provided that they are renewed.

Outside of root tokens, it is currently the only way for a token in Vault to have an unlimited lifetime.



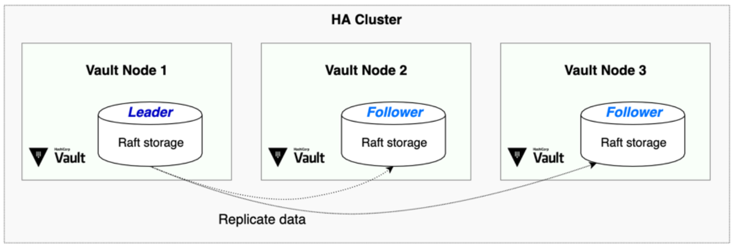
```
C:\Users\Zeal Vora>vault token create -period=30m -policy=default
Key                  Value
---                  -----
token                s.7KfiOpm3sOHJUHhpfkaY3Myw
token_accessor       6lJmxdM57AK5AAdzE821Edzy
token_duration       30m
token_renewable      true
token_policies       ["default"]
identity_policies    []
policies             ["default"]
```

Module 67 - Vault High-Availability

Vault supports a multi-server mode for high availability. This mode protects against outages by running multiple Vault servers

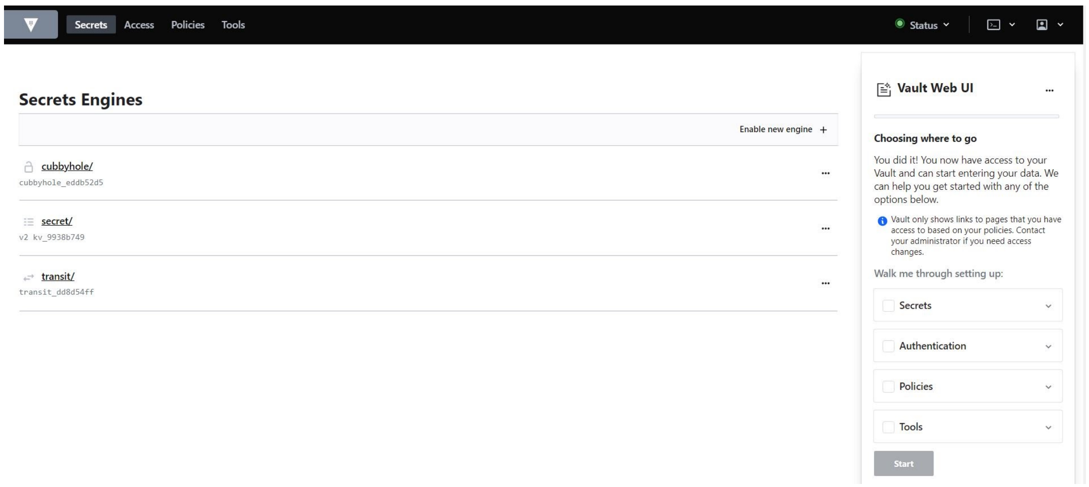
The operator step-down forces the Vault server at the given address to step down from active duty

Consul, and Integrated Storage (Raft) are some of the recommended storage backends.



Module 68 - Guide in Vault GUI

Vault GUI also provides the guide that gives information about various features



Secrets Engines

cubbyhole/
cubbyhole_eddb52d5

secret/
v2 kv_9938b749

transit/
transit_dd8d54ff

Vault Web UI

Choosing where to go

You did it! You now have access to your Vault and can start entering your data. We can help you get started with any of the options below.

Vault only shows links to pages that you have access to based on your policies. Contact your administrator if you need access changes.

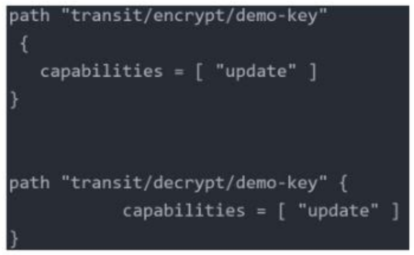
Walk me through setting up:

- Secrets
- Authentication
- Policies
- Tools

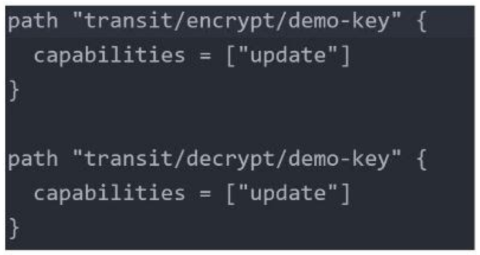
Module 69 - Vault Policy Format

The policy fmt formats a local policy file to the policy specification.

vault policy fmt file-name.hcl



```
path "transit/encrypt/demo-key"
 {
   capabilities = [ "update" ]
}


path "transit/decrypt/demo-key" {
           capabilities = [ "update" ]
}
```



```
path "transit/encrypt/demo-key" {
  capabilities = ["update"]
}

path "transit/decrypt/demo-key" {
  capabilities = ["update"]
}
```

Module 70 - Miscellaneous Pointers

Kubernetes Authentication is required for Kubernetes based workload.

Note that for Kubernetes Authentication, passing JWT token is not required for all the API requests.

/sys/seal endpoint is used to seal the vault. It requires a token with root policy or sudo capability on the path.

For EC2 based workloads, you can make use of AWS Authentication Method

Note that it is not mandatory to use the AWS auth method. You can always use other methods like AppRole.

Best of Luck for Exams, Rockstar!