Vault

* Run Dev Mode:

# vault server -dev

# export VAULT\_ADDR='http://127.0.0.1:8200'

* current secret engines:

# vault secrets list

* run vault in prod server

# vault server -config=<file>

* logs:

# journalctl -u vault

* vault conf file path:

/etc/vault.d

<https://github.com/btkrausen/hashicorp/blob/master/vault/config_files/vault.hcl>

* service file path:

/lib/systemd/system

* Since we aren't using TLS, set the VAULT\_ADDR to use HTTP instead of HTTPS and save the environment value in the bob's .bashrc file.

…>

Run the command: echo 'export VAULT\_ADDR=http://127.0.0.1:8200' >> /home/bob/.bashrc  
and reload the changes by running the source command: - source /home/bob/.bashrc

* **Integrated Storage Raft**

The Integrated Storage backend is used to maintain Vault's data. Unlike other storage backends, Integrated Storage does not operate from a single source of data. Instead, all the nodes in a Vault cluster will have a replicated copy of Vault's data. Data gets replicated across all the nodes via the[Raft Consensus Algorithm](https://raft.github.io/).

<https://raft.github.io/>

* Join standby node to the cluster

# vault operator raft join https://<leader\_node>:8200

* List cluster members

# vault operator raft list-peers

* **Seal and unsealing**

<https://developer.hashicorp.com/vault/docs/concepts/seal>

* **Login to vault by root token: if not login you will receive an error**

# vault login

Then insert root token

* **To initialize vault: after this you will get the unseal keys and the initial root token**

#Vault operator init

* **Step leader node down so another node will be the leader**

# vault operator step-down

* **initialize our Vault node but reduce the number of key shares from 5 to 3 and the key thresholds from 3 to 2.**

# vault operator init -key-shares=3 -key-threshold=2

* **enable a new KV V2 secrets engine at the path of secrets/**

# vault secrets enable -path=secrets kv-v2

* if we are using key shards and we need to migrate to autosealing we should update the conf file with the new sealing then run below command and provide it with 2 or 3 key depending on the threshold number.

# vault operator unseal -migrate

* HCL = hashicorp configuration language
* to validate the configuration file before using it by the vault service. Use the following command:

# vault operator diagnose -config=/etc/vault.d/vault.hcl

**Auth method**

* enable auth method:

# vault auth enable <type-of-authmethod>

* enable for specific path

#vault auth enable <methodname> -path=<path-name>

* disable auth method

# vault auth disable <path>

* list auth method

# vault auth list

* modify one of auth method

# vault auth tune <something-to-modify> <path>

# vault auth tune –default-lease-ttl=24h ahmed/

* to write

# vault write auth/ahmed/users/dos password=vault policies=desoki

Here we created a user

To list users

# vault list auth/ahmed/users

* to read additional info about user

# vault read auth/ahmed/users/dos

* login to vault CLI using certain method

# vault login -method=userpass username=ahmed

* print the accessors for all the tokens. The number of accessors is equal to the number of tokens created

# vault list auth/token/accessors

* create vault root token

# vault token create

* You can now use the accessor to view the properties of the token. For example, to view the properties of the root token with the accesor E6whCkegkXfetfKY1WAPwj2F , run the command:

# vault token lookup -accessor E6whCkegkXfetfKY1WAPwj2F

* To revoke token

# vault token revoke <insert root token here>

Or

# vault token revoke -accessor <insert accessor here>

* set env variable for the token

# export VAULT\_TOKEN=<token>

* auth method provides authentication, auth policy provides authorization

**Policies**

* list policies

# vault policy list

* read policy

# vault policy read <policy-name>

* create policy

# vault policy write <policy-name> <location-of-the-pre-written-polixy>

# vault policy write webapp /tmp/wepap\_policy.hcl

* create token with policy

# vault token create -policy=”webapp”

* update existing policy

# vault policy write hr-policy /etc/vault.d/hr-policy.hcl

Here we modified the policy file

**Token**

* create token

# vault token create -policy=<policy-name> -ttl=24h

* get info about token

# vault token lookup <token>

* create periodic token

# vault token create -policy=<policy-name> -period=24h

* create use limit token

# vault token create -policy=<policy> -use-limit=2

* create orphan token

# vault token create -policy=<policy-name> -orphan

Another way to create token, by adding it when creating a role for ex

* generate batch token

# vault auth enable approle

# vault write auth/approle/role/training policies=”training” token\_type=”batch” token\_ttl=”60s”

* generate periodic token

# vault write auth/approle/role/Jenkins policies=”training”

* to revoke a token

# vault token revoke <token>

* print the accessors for all the tokens. The number of accessors is equal to the number of tokens created

# vault list auth/token/accessors

**Secrets**

* enable kv-v2 secret engine on specific path

# vault secrets enable path=ahmed kv-v2

If I didn’t specify the path it will go to the name of the engine path

**KV**

* enable KV

# vault secrets enable kv

* enable at specific path

# vault secrets enable -path=dev kv

* list secrets in details

# vault secrets list -detailed