**HA VAULT WITH RAFT STORAGE ON EKS CLUSTER**

**Step -1 : Installation steps for Helm**

apt-get update

apt-get updatecurl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 > get\_helm.sh

chmod 700 get\_helm.sh

./get\_helm.sh

Now, run helm version to see if it is installed properly.

**helm version**

**output:** version.BuildInfo{Version:"v3.3.4", GitCommit:"a61ce5633af99708171414353ed49547cf05013d", GitTreeState:"clean", GoVersion:"go1.14.9"}

Run helm help to see the commands for helm

**helm help**

**Step-2 : HELM CHART**

The [Vault Helm chart](https://github.com/hashicorp/vault-helm) is the recommended way to install and configure Vault on Kubernetes. In addition to running Vault itself, the Helm chart is the primary method for installing and configuring Vault to integrate with other services such as Consul for High Availability (HA) deployments.

To use the Helm chart, add the Hashicorp helm repository and check that you have access to the chart.

**$** helm repo add hashicorp https://helm.releases.hashicorp.com

"hashicorp" has been added to your repositories

**Step-3 : Installing vault with raft storage enabled**

helm install vault hashicorp/vault \

--set='server.ha.enabled=true' \

--set='server.ha.raft.enabled=true'

Now , see the pods which gets deployed using **kubectl get pods** command

NAME READY STATUS RESTARTS AGE

vault-0 1/1 Running 0 8m12s

vault-1 1/1 Running 0 8m12s

vault-2 1/1 Running 0 8m12s

vault-agent-injector-bdbf7b844-6q7rs 1/1 Running 0 8m12s

**Step-4 : init all your vaults and unseal**

Run the following command to init your vault :

kubectl exec -ti vault-0 -- vault operator init

**output:**

Unseal Key 1: bodIIs/nXU+sUDrW6xjzcBWtJ4Z3CpuRMWJk9iB5Oc6U

Unseal Key 2: CUGlU6xMCYvQMTbvVBCaYcWCw4suIyIanv6mMNSmxLCI

Unseal Key 3: k0lr2VFPrqLAgta9zCgdAXT7LixCy6h/bqXjwF0nKR8K

Unseal Key 4: fnt31V8OZsdODGfB5KNM122p5ksv8yzd480+H2S88Jdg

Unseal Key 5: CPeR2J0IJRVTiFINXBPdrOOYlALfgOAmcfZJSZmMco4F

Initial Root Token: s.2Mw9rmUkuZt0fTxxJMJhV7RE

Vault initialized with 5 key shares and a key threshold of 3. Please securely

distribute the key shares printed above. When the Vault is re-sealed,

restarted, or stopped, you must supply at least 3 of these keys to unseal it

before it can start servicing requests.

Vault does not store the generated master key. Without at least 3 key to

reconstruct the master key, Vault will remain permanently sealed!

It is possible to generate new unseal keys, provided you have a quorum of

existing unseal keys shares. See "vault operator rekey" for more information

Run the following command to unseal “vault-0”,

kubectl exec -ti vault-0 -- vault operator unseal

give unseal keys from the above output

**FOR VAULT-1 :**

join the vault servers to cluster using following command,

kubectl exec -ti vault-1 -- vault operator raft join <http://vault-0.vault-internal:8200>

**output :**

Key Value

--- -----

Joined true

Unseal vault-1 using following command, give unseal keys from the output of vault init for “vault - 0 “

kubectl exec -ti vault-1 -- vault operator unseal

**FOR VAULT-2 :**

join the vault servers to cluster using following command,

kubectl exec -ti vault-2 -- vault operator raft join <http://vault-0.vault-internal:8200>

**output :**

Key Value

--- -----

Joined true

Unseal vault-1 using following command, give unseal keys from the output of vault init for “vault - 0 “

kubectl exec -ti vault-2 -- vault operator unseal

Now, your vaults are unsealed and joined and are ready to be used.

**Verification:**

Get inside your first vault,

kubectl exec -it vault-0 bin/sh

**output :**

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.2Mw9rmUkuZt0fTxxJMJhV7RE

token\_accessor ztiTEPUqB7SwFo4P1rqF7GjO

token\_duration ∞

token\_renewable false

token\_policies ["root"]

identity\_policies []

policies ["root"]

Now, type the following commands to see the peers that are joined to the cluster,

vault operator raft list-peers

**output :**

Node Address State Voter

---- ------- ----- -----

223592f0-74b9-eb1a-e8e5-c5ee869e596e vault-0.vault-internal:8201 leader true

b3824dbb-6a66-cc47-53f6-97cdff4b5f33 vault-1.vault-internal:8201 follower true

fb845e92-658e-17c0-24ea-8558ad7cadfa vault-2.vault-internal:8201 follower true

**Enable secrets engine and check if the secret is replicated in other vaults**

vault secrets enable -version=1 kv

vault kv put kv/my-secret my-value=secret

vault kv get kv/my-secret

**output :**

====== Data ======

Key Value

--- -----

my-value secret

Now, get inside other vaults and then login using the root key and then type the following command, you should see the following secret in every vault .

vault kv get kv/my-secret

**output :**

====== Data ======

Key Value

--- -----

my-value secret

**Links** :

<https://www.vaultproject.io/docs/platform/k8s/helm/examples/ha-with-raft>