Workshop
Úvod do Hashicorp Vault

Osnova

1. WEB & Dokumentácia
2. Learning Platform
3. Inštalácia
4. Konfigurácia Auth
5. Konfigurácia KV
6. Autentifikácia
7. Vytvorenie secretu
8. Získanie secretu
9. Restart, Unseal

Predpríprava

Pomocou gitu si naclonujeme repicko s workshopom

```
git clone https://github.com/rvojcik/vault-workshop-beginer.git ./
```

Alebo ak nemame git pouzijeme download metódu

```
wget https://github.com/rvojcik/vault-workshop-beginer/archive/master.zip
unzip master.zip
```

Následne už len

```
cd vault-workshop-beginer
```

WEB & Dokumentace

Hashicorp: https://www.hashicorp.com/

Vault Project: https://www.vaultproject.io/

Vault Doc: https://www.vaultproject.io/docs/

API Doc: https://www.vaultproject.io/api/

Learning Platform

https://learn.hashicorp.com/vault

Inštalácia

Download binárky

```
wget https://releases.hashicorp.com/vault/1.2.3/vault_1.2.3_linux_amd64.zip -O vault.zip
unzip vault.zip
./vault version
```

```
Vault v1.2.3
```

Príprava datadiru a configu

```
mkdir data
cat server.hcl
```

```
storage "file" {
  path = "./data"
}
listener "tcp" {
  address = "0.0.0.0:8200"
  tls_disable = true
}
api_addr = "http://127.0.0.1:8200"
ui = true
disable_mlock = true
```

Spustenie Vault Serveru

```
./vault server -config server.hcl
```

```
==> Vault server configuration:
                Api Address: http://127.0.0.1:8200
                        Cgo: disabled
          Cluster Address: https://127.0.0.1:8201
                Listener 1: tcp (addr: "0.0.0.0:8200", cluster address: "0.0.0.0:8201",
max_request_duration: "1m30s", max_request_size: "33554432", tls: "disabled")
                Log Level: info
                    Mlock: supported: true, enabled: false
                  Storage: file
                  Version: Vault v1.2.3

==> Vault server started! Log data will stream in below:
```

Náš Vault server ešte nieje nainicializovaný. Môžeme to spraviť pomocou WEB UI, ktoré sme si zapli v server.hcl, alebo pomocou CLI utility vault.

Inicializácia

CLI

Inicializácia environmentu. Musíme najprv cli utilite vysvetliť, kde má hľadať vault server.

Nasledne nastavíme 1 key share a limit 1 pre unseal.

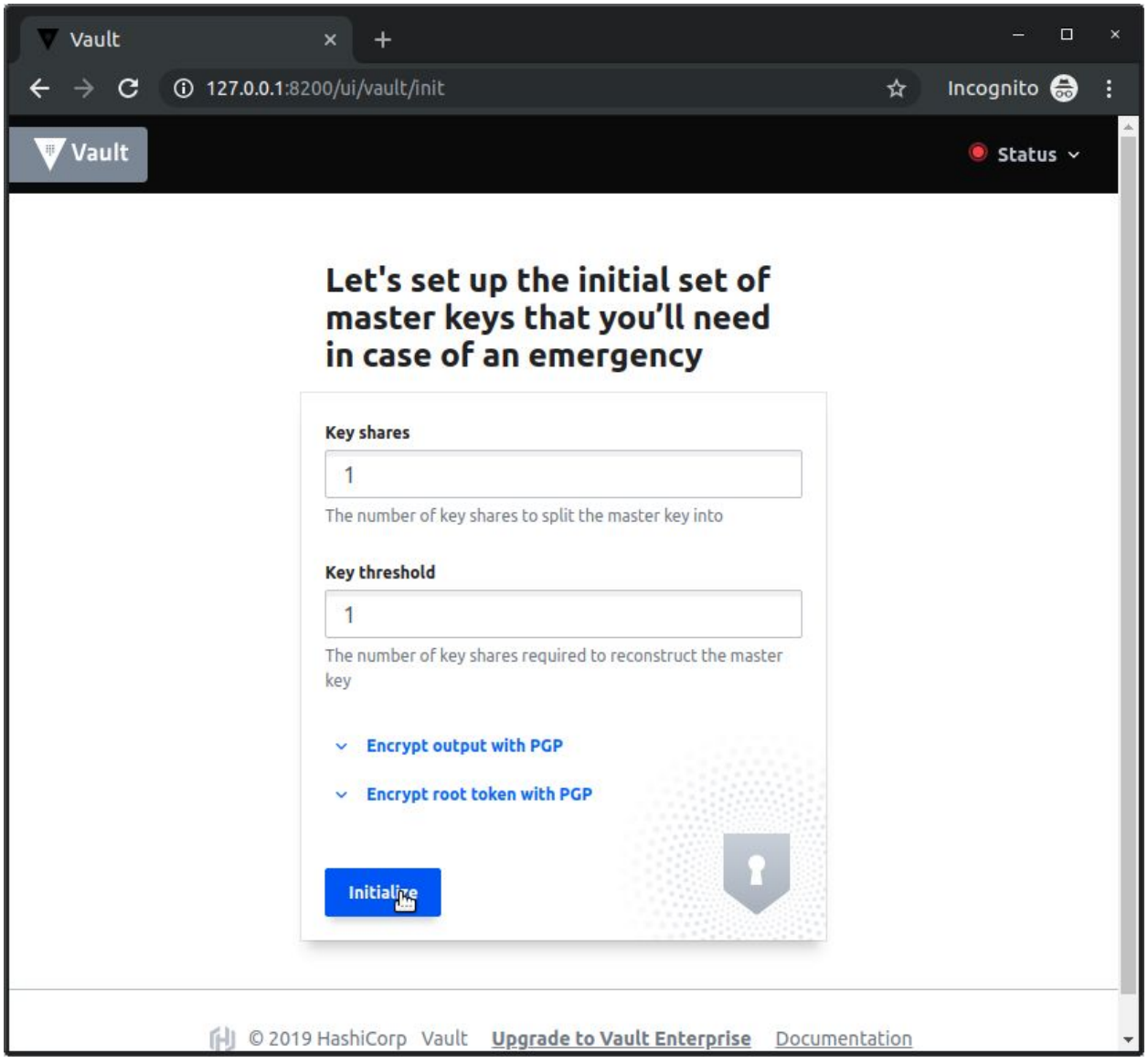
```
export VAULT_ADDR=http://127.0.0.1:8200

./vault operator init -key-shares=1 -key-threshold=1
```

```
Unseal Key 1: kKPzRyCMcAZAw2KmT2VH3/+BE3YGeUye+p/a24GBMOc=

Initial Root Token: s.rhKaReo3u4LUC9WcH87wDJ1s

Vault initialized with 1 key shares and a key threshold of 1. Please securely
distribute the key shares printed above. When the Vault is re-sealed,
restarted, or stopped, you must supply at least 1 of these keys to unseal it
before it can start servicing requests.
```

Unseal Key a Root Token si uložíme do súboru. Budeme ich ešte potrebovať.

UI

Otvoríme si prehliadač a inicializujeme náš Vault Server.



Klikneme na “Initialize” a skopírujeme si niekam **Root Token** a **Unseal Key**, alebo klikneme na “Download keys” a stiahneme si JSON.

Unseal

Po inicializacií musíme Vault unsealnut, aby mohli prebiehať potrebné operácie, teda dešifrovať barrier.

```
./vault operator unseal
Unseal Key (will be hidden): <nas key>
```

```
Key             Value
---             -----
Seal Type       shamir
Initialized     true
Sealed          false
Total Shares    1
Threshold       1
Version         1.2.3
Cluster Name    vault-cluster-9bb44e33
Cluster ID      4437e161-aa15-c64f-bf13-2eb7c3c3ff18
HA Enabled      false
```

Admin Login

Keďže okrem inicializácie nemáme pripravené nič viac, použijeme Admin Root Login aby sme mohli s Vaultom pracovať.

```
./vault login
Token (will be hidden): <root token>
```

```
Success!
Key                  Value
---                  -----
token                s.2dRoYbxRlQcoihESHkyvp2wC
token_accessor       SWojy4C8Y2AwVLFGHMHx8wkP
token_duration       ∞
token_renewable      false
token_policies       ["root"]
identity_policies    []
policies             ["root"]
```

Konfigurácia KV

Vault umožňuje použiť veľké množstvo rôznych Secrets Storage riešení. Zoznam aktuálne podporovaných nájdete na https://www.vaultproject.io/docs/secrets/index.html

My pre jednoduchosť použijeme najjednoduchší KV, teda Key Value storage Version 1.

Vytvoríme si dva KV storage pre APP1 a APP2

```
./vault secrets enable -version=1 -path=app1 kv
./vault secrets enable -version=1 -path=app2 kv
./vault secrets list
```

```
Path          Type         Accessor              Description
----          ----         --------              -----------
app1/         kv           kv_16f9cdee           n/a
app2/         kv           kv_ac3d2665           n/a
cubbyhole/    cubbyhole    cubbyhole_99861b77    per-token private secret storage
identity/     identity     identity_71a3d72c     identity store
sys/          system       system_8ffd4386       system endpoints used for control, policy
and debugging
```

Konfigurácia Auth

Pripravíme si AppRole autentifikáciu, dôvod je ten, že je to odporúčaná auth pre scripty, appky a podobne. To je to čo si chceme na workshope hlavne vyskúšať. Čo sa týka autentifikácie ľudi, v Livesporte pouzivame LDAP. Možnosti ako autentifikovat ľudi je ale pomerne dosť, takže odporúčam si prejst dokumentáciu a vybrat si to, čo najviac bude sedieť vám do infraštruktúry: https://www.vaultproject.io/docs/auth/

AppRole

Pri konfigurácií budeme používať hlavne rozhranie CLI. Dovodom je, že v UI sú niektoré časti pomerne metúce a zo skúsenosti viac preferujem prave CLI pre jeho jednoznačnosť.

```
./vault enable approle
```

```
Success! Enabled approle auth method at: approle/
```

Týmto máme v mountpointe approle/ pripravený endpoint našeho autentifikačnáho pluginu.

Aby sme mohli pokračovať , musíme si definovať policy, ktoré budú určovať práva autentifikovanej aplikácie.

V naclonovanom repozitari workshopu v adresari policy mame pripravene 3 subory.

- app1.hcl
- app2.hcl
- app3.hcl

Sú to 3 policy pre 3 rôzne aplikácie z ktorých každá má iné možnosti toho, ako môže s Vaultom pracovať. Záleží len na nás koľko a akých policy si vytvoríme. Dobrý postup je ale nepridelovať vyššie práva než je bezpodmienečne nutné. Môžeme tým výrazne obmedziť tzv. **Attack Surface** v prípade bezpečnostného incidentu.

Príklad z praxe:

V Livesportu pre systém okolo certifikátov máme napríklad rozdelené policy takto

- cert-deploy
    - umožňuje prečítať určitý certifikát pre potreby jeho nasadenia
    - neumožňuje vylistovanie certifikátov
    - neumožňuje jeho update a delete
- cert-upload
    - umoznuje vytvorit alebo updatnut certifikat
    - neumoznuje vylistovanie certifikatov
    - neumoznuje delete
- cert-report
    - umožňuje prečítať certifikáty pre potreby ich kontroly a monitoringu
    - neumožňuje ich update a delete

Pripravíme si naše 3 policy a nakoniec si vypíšeme zoznam

```
./vault policy write app1 policy/app1.hcl
./vault policy write app2 policy/app2.hcl
./vault policy write app3 policy/app3.hcl
./vault policy list
```

```
app1
app2
app3
default
root
```

Viac informacií o možnostiach policy nájdete v dokumentácií:
https://www.vaultproject.io/docs/concepts/policies.html

V tomto bode máme aktivovaný AppRole auth v ceste approle/, a máme pripravené 3 policy app1-3.

Vytvoríme si teda v approle/ 3 role a priradíme im policy.

```
./vault write auth/approle/role/app1 policies=app1,default period=30m secret_id_num_uses=0
secret_id_ttl=0 token_max_ttl=1h token_num_uses=0 token_ttl=30m

./vault write auth/approle/role/app2 policies=app2,default period=30m secret_id_num_uses=0
secret_id_ttl=0 token_max_ttl=1h token_num_uses=0 token_ttl=30m

./vault write auth/approle/role/app3 policies=app3,default period=30m secret_id_num_uses=0
secret_id_ttl=0 token_max_ttl=1h token_num_uses=0 token_ttl=30m

./vault list auth/approle/role/
```

```
Keys
----
app1
app2
app3
```

Môžeme si pre app1 overiť parametre aké sme použili a aké ďalšie majú svoju default hodnostu.

```
./vault read auth/approle/role/app1
```

```
Key                        Value
---                        -----
bind_secret_id             true
local_secret_ids           false
period                     30m
policies                   [app1 default]
secret_id_bound_cidrs      <nil>
secret_id_num_uses         0
secret_id_ttl              0s
token_bound_cidrs          []
token_explicit_max_ttl     0s
token_max_ttl              1h
token_no_default_policy    false
token_num_uses             0
token_period               30m
token_policies             [app1 default]
token_ttl                  30m
token_type                 default
```

Role máme teda pripravené. Musíme ešte získať role-id a secret-id pre každú approle aby sme mohli prejsť k samotnej autentifikacií.

```
./vault read auth/approle/role/app1/role-id > app1.auth
./vault write -f auth/approle/role/app1/secret-id >> app1.auth

./vault read auth/approle/role/app2/role-id > app2.auth
./vault write -f auth/approle/role/app2/secret-id >> app2.auth

./vault read auth/approle/role/app3/role-id > app3.auth
./vault write -f auth/approle/role/app3/secret-id >> app3.auth

cat app1.auth
```

```
Key         Value
---         -----
role_id     a116cc97-264a-350d-c351-c408c93ef3fc

Key                   Value
---                   -----
secret_id             a6a47c9d-225e-b895-5de8-132317831a83
secret_id_accessor    8075163c-f17d-82ca-ab41-02dac4412aea
```

Autentifikácia

Jeden druh autentifikácie pomocou vault binarky sme si uz skúsili, keď sme použíli root token.

Teraz je čas si vyskúšať autentifikáciu priamo cez API s použitím našej AppRole.

Autentifikácia prebieha nasledovne

- odoslanie role-id a secret-id
- príjem tokenu
- podľa potreby renew tokenu behom činnosti
- token vyprší, alebo ho sami invalidujeme na konci činnosti

Získanie tokenu

Skúsme si len za pomoci dokumentácie, Curl-u a JQ utility (utilita pre parsovanie json v cli) vyskúšať autentifikovať sa pomocou approle ako app1 a získať token.

CHEAT1

Skusme sa takto autentifikovat aj pomocou APP2 a APP3, tokeny si uložíme do premenných TOKEN1, TOKEN2 a TOKEN3

Informácie o tokenu

O tomto tokene môžeme získať ďalšie informácie pomocou lookup API:
https://www.vaultproject.io/api/auth/token/index.html#lookup-a-token-self-

CHEAT2

Skúsme si pomocou dokumentácie spraviť lookup na naše tokeny.

Renew tokenu

https://www.vaultproject.io/api/auth/token/index.html#renew-a-token-self-

Renew tokenu nám umožňuje predĺžiť platnosť tokenu v prípade potreby. Táto funkcionalita poskytuje možnosť predlžovať token v pripade, ze je aktivne pouzivany ale len do jeho MAX TTL.

Je možné stanoviť pri nastavení ROLE v AppRole parametre, ktoré určitým spôsobom upravujú chovanie a možnosti renew.

V našom prípade to bolo:

- period=30m — Perioda v rámci ktorej, môže byť token obnovený(renew)
- token_max_ttl=1h — Maximálna dĺžka života tokenu, ktorá nejde prekročiť
- token_num_uses=0 — Maximálny počet použití / requestov
- token_ttl=30m — Počiatočné TTL tokenu

Vytvorenie secretu

https://www.vaultproject.io/api/secret/kv/kv-v1.html

```
curl -s -X PUT -H "X-Vault-Token: $TOKEN3" -d '{"heslo": "Password1" }'
http://127.0.0.1:8200/v1/app1/tajne_heslo
```

Úlohy

- Skúste si secret vytvoriť pomocou tokenov 1 a 2, vault by vám podľa policy mal vrátiť chybu
- Vytvorte si secret aj v ceste app2

Získanie secretu

Vylistovanie secretov v ceste APP1

```
curl -s -X LIST -H "X-Vault-Token: $TOKEN1" http://127.0.0.1:8200/v1/app1/ | jq .
```

Ziskanie naseho secretu

```
curl -s -X GET -H "X-Vault-Token: $TOKEN1" http://127.0.0.1:8200/v1/app1/tajne_heslo | jq .
```

Úlohy

- Získajte pomocou TOKEN1 secret z app2, chová sa to správne podľa policy ?
- Získajte si pomocou TOKEN2 secret z app1, chová sa to správne podľa policy ?

Rozširujúce informácie

Inštalácia na server v produkcií

Binarka, config, data a linky

Tu je viac moznych rieseni a kazde z nich moze mat svoj zmysel. My pouzivame Puppet ako CM, takze nasledujuce akcie su sucastou naseho Vault modulu.

Typicky deploy ale je:

- vytvorime si adresar v **/opt/vault-<version>**
- vytvorime si link **/opt/vault-current -> /opt/vault-<version>**
- vytvorime link **/usr/local/bin/vault -> /opt/vault-current/vault**
- vytvorime uzivatela **vault** a pridame ho do grupy **ssl-cert**
- vytvorime adresar pre data, **/data/vault**
    - chown -R vault.vault /data/vault
    - chmod 0700 /data/vault
- vytvorime konfiguracny adresar **/etc/vault**
    - touch /etc/vault/config.json
    - chown vault:vault /etc/vault/config.json
    - chmod 0640 /etc/vault/config.json

SystemD

```
[Unit]
Description=HashiCorp's Vault Server
After=network.target
ConditionPathExists=/etc/vault/config.json

[Service]
Type=simple
User=vault
Group=vault
ExecStart=/usr/local/bin/vault server \
  -config /etc/vault/config.json
ExecReload=/bin/kill -HUP $MAINPID
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Nastavenia default lease ttl v configu

Mnohé auth pri vytváraní tokenov používajú default TTL (v prípade, že sa nešpecifikuje), ktoré je pomerne veľké - 768h.

Je vhodné v configu Vault serveru nastaviť túto hodnotu na priateľne nízku.

Jedna sa konkrtne o default_lease_ttl - https://www.vaultproject.io/docs/configuration/#default_lease_ttl

Cheaty

CHEAT1

```
curl -s -X POST \
 --data \
'{"role_id": "<role-id>",
"secret_id": "<secret-id"}' \
http://127.0.0.1:8200/v1/auth/approle/login \
| jq -r .auth.client_token

# Nacteni do premennych
TOKEN1=$(command)
```

Response Example:


```
{
  "request_id": "73140665-955b-f996-aa85-4b6f3d183829",
  "lease_id": "",
  "renewable": false,
  "lease_duration": 0,
  "data": null,
  "wrap_info": null,
  "warnings": null,
  "auth": {
    "client_token": "s.MKIFlPhb9qCNAkNmUDUiBZGN",
    "accessor": "dB9RJ0y5k2peQ7CUOqUwbcdH",
    "policies": [
      "app1",
      "default"
    ],
    "token_policies": [
      "app1",
      "default"
    ],
    "metadata": {
      "role_name": "app1"
    },
    "lease_duration": 1800,
    "renewable": true,
    "entity_id": "06b25558-4039-1c64-7224-443863677b6f",
    "token_type": "service",
    "orphan": true
  }
}
```

CHEAT2

```
curl -s -XGET -H "X-Vault-Token: $TOKEN1" \
 http://127.0.0.1:8200/v1/auth/token/lookup-self | jq .
```


```
{
  "request_id": "53b7a6ef-8b30-5cab-de5d-0ec088ee426c",
  "lease_id": "",
  "renewable": false,
  "lease_duration": 0,
  "data": {
    "accessor": "7W7nLCtdWCEgDK9DCKOpHfhi",
    "creation_time": 1569397490,
    "creation_ttl": 1800,
    "display_name": "approle",
    "entity_id": "06b25558-4039-1c64-7224-443863677b6f",
    "expire_time": "2019-09-25T10:14:50.316800164+02:00",
    "explicit_max_ttl": 0,
    "id": "s.H8g3huTz84YNXwDEDndo3W2O",
    "issue_time": "2019-09-25T09:44:50.316799959+02:00",
    "meta": {
      "role_name": "app1"
    },
    "num_uses": 0,
    "orphan": true,
    "path": "auth/approle/login",
    "policies": [
      "app1",
      "default"
    ],
    "renewable": true,
    "ttl": 825,
    "type": "service"
  },
  "wrap_info": null,
  "warnings": null,
  "auth": null
}
```