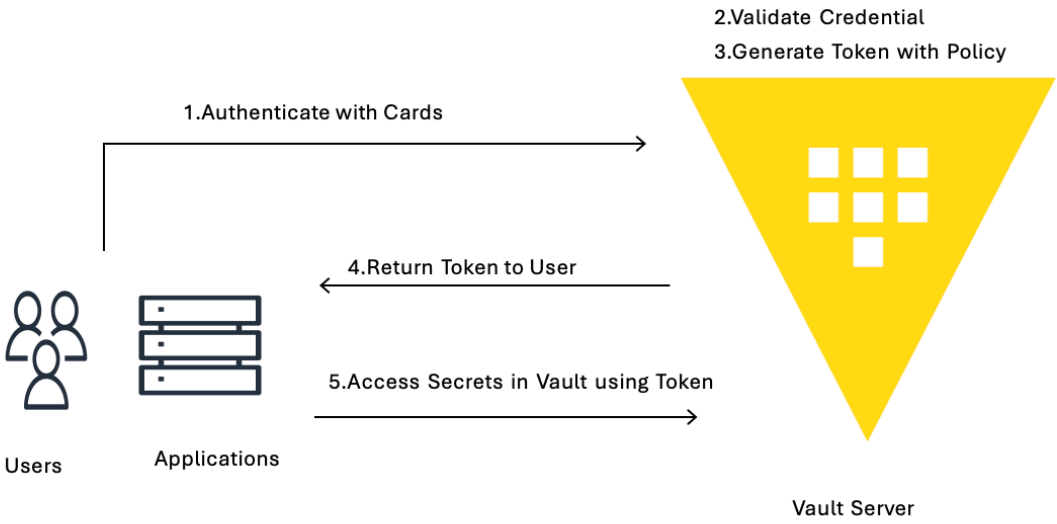
Step-by-Step Guide to Configuring HashiCorp Vault Auth Methods Using CLI and UI



By

Mahendran Selvakumar

https://devopstronaut.com/

https://www.linkedin.com/in/mahendran-selvakumar-36444a77/

Step1: Install Hashicorp Vault on Linux instance and running as Dev server

Login to Vault Server

```
mahendranselvakumar@Mahendrans-MBP Downloads % ssh -i "kubernetes.pem" ec2-user@ec2-54-75-119-120.eu-west-1.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Oct  9 16:25:25 2024 from 176.248.232.84
[ec2-user@ip-172-31-33-10 ~]$ sudo su -
```

Install **yum-utils** and **shadow-utils** on the Linux Instance

- yum-utils: This is a package that provides utilities and tools for managing YUM repositories, installing, removing, and updating packages, and more.
- shadow-utils: This is a package that provides essential utilities for managing user accounts and passwords, including useradd, usermod, passwd, and others.

```
[ec2-user@ip-172-31-33-10 ~]$ sudo su -
Last login: Wed Oct  9 16:25:29 UTC 2024 on pts/3
[root@ip-172-31-33-10 ~]# sudo yum install -y yum-utils shadow-utils
Last metadata expiration check: 0:16:01 ago on Wed Oct  9 16:14:41 2024.
Package dnf-utils-4.3.0-13.amzn2023.0.4.noarch is already installed.
Package shadow-utils-2:4.9-12.amzn2023.0.4.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-33-10 ~]#
```

Add hashicorp repo to the Yum Package manager

```
[root@ip-172-31-33-10 ~]# sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
Adding repo from: https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
[root@ip-172-31-33-10 ~]#
```

Install Vault using Yum package

```
[root@ip-172-31-33-10 ~]# sudo yum -y install vault
Hashicorp Stable - x86_64                                              16 MB/s | 1.4 MB     00:00
Dependencies resolved.
=====================================================================================================
 Package            Architecture         Version               Repository                      Size
=====================================================================================================
Installing:
 vault              x86_64               1.18.0-1              hashicorp                      154 M

Transaction Summary
=====================================================================================================
Install  1 Package

Total download size: 154 M
Installed size: 436 M
Downloading Packages:
vault-1.18.0-1.x86_64.rpm                                              82 MB/s | 154 MB     00:01
-----------------------------------------------------------------------------------------------------
Total                                                                  82 MB/s | 154 MB     00:01
Hashicorp Stable - x86_64                                             115 kB/s | 3.9 kB     00:00
Importing GPG key 0xA621E701:
 Userid     : "HashiCorp Security (HashiCorp Package Signing) <security+packaging@hashicorp.com>"
 Fingerprint: 798A EC65 4E5C 1542 8C8E 42EE AA16 FCBC A621 E701
 From       : https://rpm.releases.hashicorp.com/gpg
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                            1/1
  Running scriptlet: vault-1.18.0-1.x86_64                                                      1/1
  Installing       : vault-1.18.0-1.x86_64                                                      1/1
  Running scriptlet: vault-1.18.0-1.x86_64                                                      1/1
Generating Vault TLS key and self-signed certificate...
[illegible]
-----
Vault TLS key and self-signed certificate have been generated in '/opt/vault/tls'.

  Verifying        : vault-1.18.0-1.x86_64                                                      1/1

Installed:
  vault-1.18.0-1.x86_64

Complete!
[root@ip-172-31-33-10 ~]#
```

Verify the Vault installation

```
[root@ip-172-31-33-10 ~]# vault --version
Vault v1.18.0 (77f26ba561a4b6b1ccd5071b8624cefef7a72e84), built 2024-10-08T09:12:52Z
[root@ip-172-31-33-10 ~]#
```

Step2: Start Vault with Dev Server Mode

We can vault instance as dev mode using -dev

```
[root@ip-172-31-33-10 ~]# vault server -dev
==> Vault server configuration:

Administrative Namespace:
             Api Address: http://127.0.0.1:8200
                     Cgo: disabled
         Cluster Address: https://127.0.0.1:8201
   Environment Variables: BASH_FUNC_which%%, HISTCONTROL, HISTSIZE, HOME, HOSTNAME, LANG, LESSOPEN, LOGNAME, LS_COLORS, MAIL, PATH, PWD, SHELL, SHLVL, SYSTEMD_COLORS,
 S_COLORS, TERM, USER, _, which_declare
              Go Version: go1.22.7
              Listener 1: tcp (addr: "127.0.0.1:8200", cluster address: "127.0.0.1:8201", disable_request_limiter: "false", max_request_duration: "1m30s", max_request
_size: "33554432", tls: "disabled")
               Log Level:
                   Mlock: supported: true, enabled: false
           Recovery Mode: false
                 Storage: inmem
                 Version: Vault v1.18.0, built 2024-10-08T09:12:52Z
             Version Sha: 77f26ba561a4b6b1ccd5071b8624cefef7a72e84

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

```
2024-10-09T16:42:42.183Z [INFO]  rollback: starting rollback manager
WARNING! dev mode is enabled! In this mode, Vault runs entirely in-memory
and starts unsealed with a single unseal key. The root token is already
authenticated to the CLI, so you can immediately begin using Vault.

You may need to set the following environment variables:

    $ export VAULT_ADDR='http://127.0.0.1:8200'

The unseal key and root token are displayed below in case you want to
seal/unseal the Vault or re-authenticate.

Unseal Key: nsIUrMzCPSRlw2uw7yAFzTHMlvL1CDMcoN9qDxpaDfY=
Root Token: hvs.aTdW0lH1d6jGwKFs0OBrkrrh

Development mode should NOT be used in production installations!
```

Vault server started as dev mode and it's running with in-memory if we close the terminal we can't able to access the vault.

Set Environment Variable

Open new terminal of Vault server and set environment variable

```
mahendranselvakumar@Mahendrans-MBP Downloads % ssh -i "kubernetes.pem" ec2-user@ec2-54-75-119-120.eu-west-1.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Oct  9 16:29:37 2024 from 176.248.232.84
[ec2-user@ip-172-31-33-10 ~]$ sudo su -
Last login: Wed Oct  9 16:30:00 UTC 2024 on pts/3
[root@ip-172-31-33-10 ~]# export VAULT_ADDR='http://127.0.0.1:8200'
[root@ip-172-31-33-10 ~]#
```

Verify Vault Setup and Access Secrets

```
[root@ip-172-31-33-10 ~]# vault status
Key             Value
---             -----
Seal Type       shamir
Initialized     true
Sealed          false
Total Shares    1
Threshold       1
Version         1.18.0
Build Date      2024-10-08T09:12:52Z
Storage Type    inmem
Cluster Name    vault-cluster-73e03a04
Cluster ID      a4a02085-d6b3-7bd8-f422-6c92eec6f194
HA Enabled      false
[root@ip-172-31-33-10 ~]#
```

Now we can see Vault server seal type as Shamir and it's initialized, unsealed automatically. Vault storage type also in memory so we can't use in production.

Step3: Configure Hashicorp Vault authentication methods using the CLI

Enable the userpass auth method using the command below

```
[root@ip-172-31-33-10 ~]# vault auth enable -path=dev -description="Development environment credentials" userpass
Success! Enabled userpass auth method at: dev/
[root@ip-172-31-33-10 ~]#
```

Verify the enabled Vault auth method using the **vault auth list** command

```
[root@ip-172-31-33-10 ~]# vault auth list
Path      Type        Accessor                 Description                         Version
----      ----        --------                 -----------                         -------
dev/      userpass    auth_userpass_a4be7502   Development environment credentials  n/a
token/    token       auth_token_57c944dc      token based credentials              n/a
[root@ip-172-31-33-10 ~]#
```

Configure the lease for the enabled auth method using the vault auth tune command

```
[root@ip-172-31-33-10 ~]# vault auth tune -default-lease-ttl=24h dev/
Success! Tuned the auth method at: dev/
[root@ip-172-31-33-10 ~]#
```

Create a username and password under the userpass auth method and assign a policy to the user

```
[root@ip-172-31-33-10 ~]# vault write auth/dev/users/mahi password=Welcome@123 policies=devpolicy
Success! Data written to: auth/dev/users/mahi
[root@ip-172-31-33-10 ~]#
```

The username is **mahi**, and the assigned policy is **devpolicy** under the **userpass** auth method

Verify the created user

```
[[root@ip-172-31-33-10 ~]# vault list auth/dev/users
Keys
----
mahi
[root@ip-172-31-33-10 ~]#
```

Now we can see that the user mahi has been created

Using the **vault read** command, we can see the details of the created user, including their assigned policy

```
[[root@ip-172-31-33-10 ~]# vault read auth/dev/users/mahi
Key                        Value
---                        -----
policies                   [devpolicy]
token_bound_cidrs          []
token_explicit_max_ttl     0s
token_max_ttl              0s
token_no_default_policy    false
token_num_uses             0
token_period               0s
token_policies             [devpolicy]
token_ttl                  0s
token_type                 default
[root@ip-172-31-33-10 ~]#
```

Log in to Vault using the newly created user to check whether access is granted

```
[root@ip-172-31-33-10 ~]# vault login -method=userpass username=mahi password=Welcome@123
Error authenticating: Error making API request.

URL: PUT http://127.0.0.1:8200/v1/auth/userpass/login/mahi
Code: 403. Errors:

* permission denied
[root@ip-172-31-33-10 ~]#
```

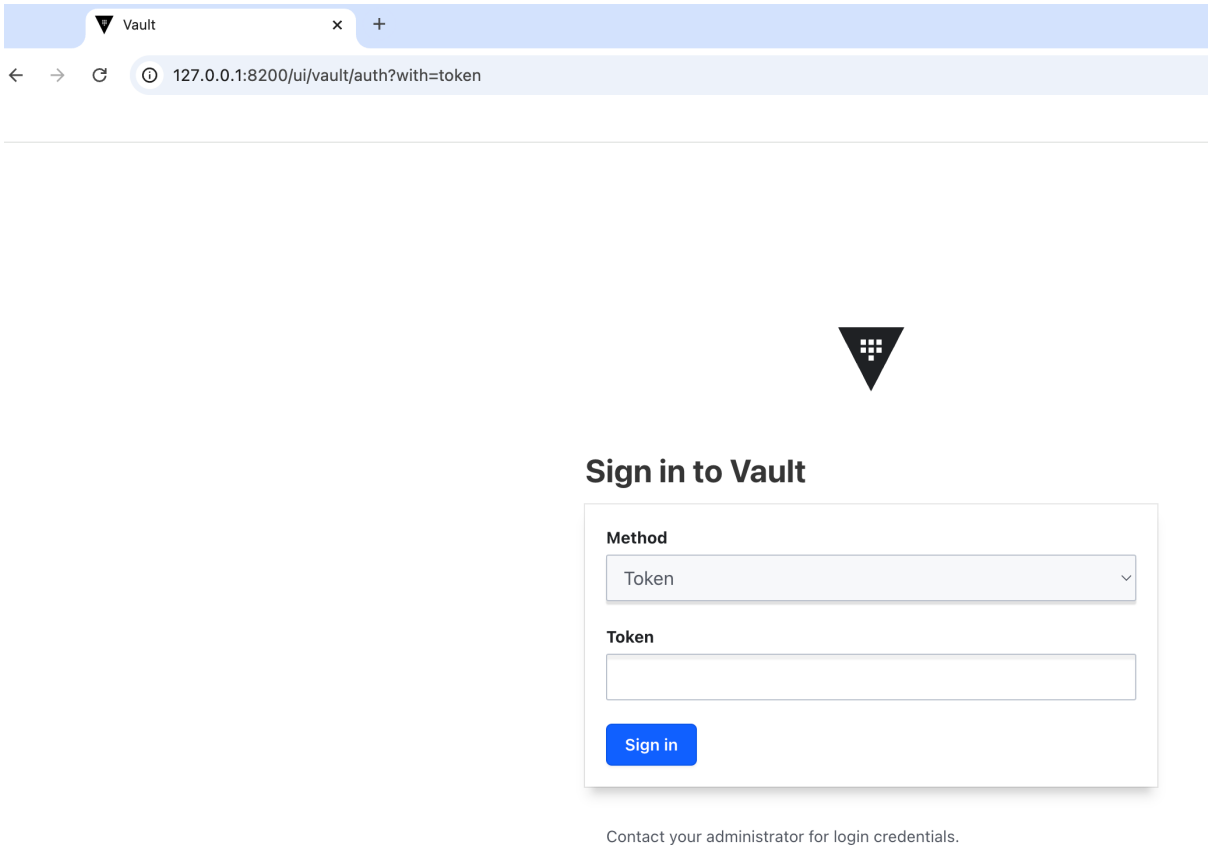
Log in using the newly created username and password, but I am getting an error because Vault is trying to log in with the default path, while we have used the path **dev**

Use Vault with the userpass method at a custom mount path (dev). If you are using the default mount path, you don't need to specify it while logging in

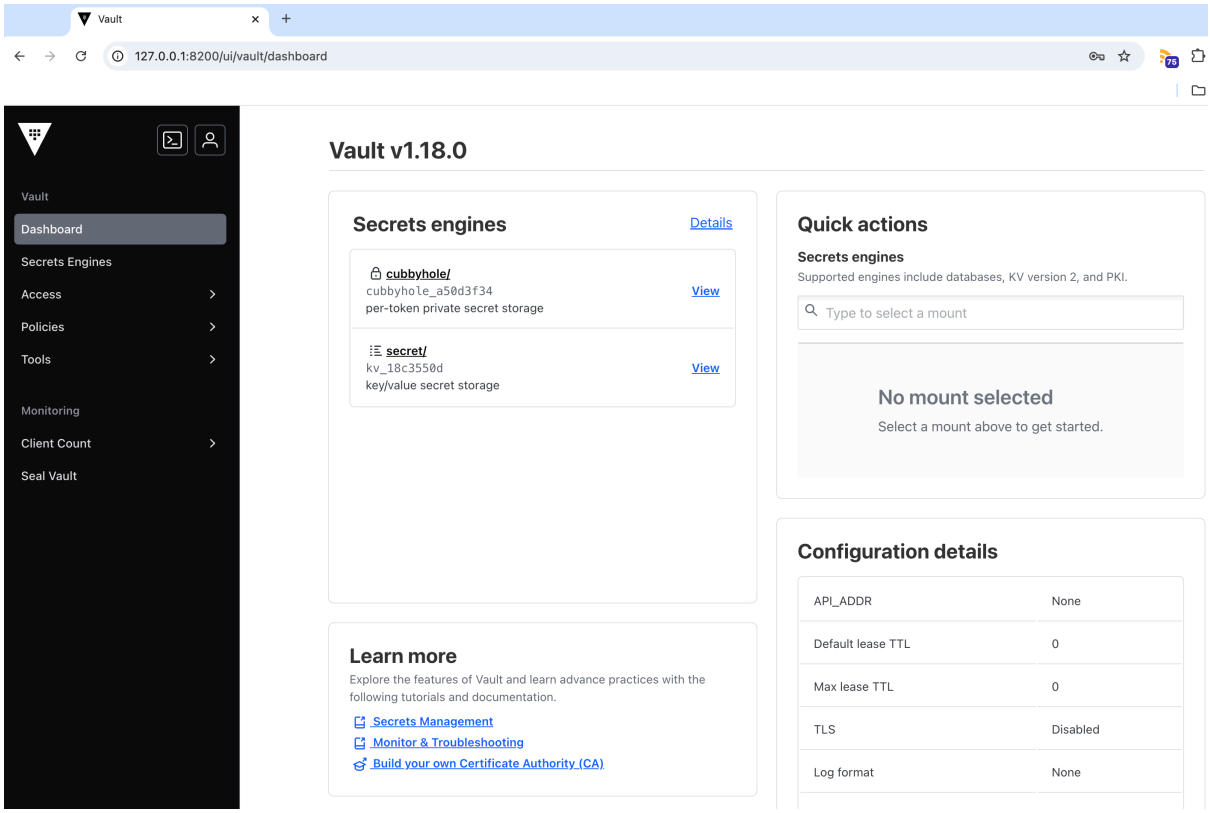
```
[root@ip-172-31-33-10 ~]# vault login -method=userpass username=mahi password=Welcome@123 mount=dev
Success! You are now authenticated. The token information displayed below
is already stored in the token helper. You do NOT need to run "vault login"
again. Future Vault requests will automatically use this token.

Key                    Value
---                    -----
token                  hvs.CAESIHCkLN0Dx_OlxDlpRLsU0vGyJN2orh9pOSSRfYJE6RZRGh4KHGh2cy5USUZuZk44eUdoOWhXeFBVaEpOWWswWU8
token_accessor         UHhJVy3rP5t1gxbjlx8eTR7T
token_duration         24h
token_renewable        true
token_policies         ["default" "devpolicy"]
identity_policies      []
policies               ["default" "devpolicy"]
token_meta_username    mahi
[root@ip-172-31-33-10 ~]#
```

Step4: Configure Hashicorp Vault authentication methods using UI

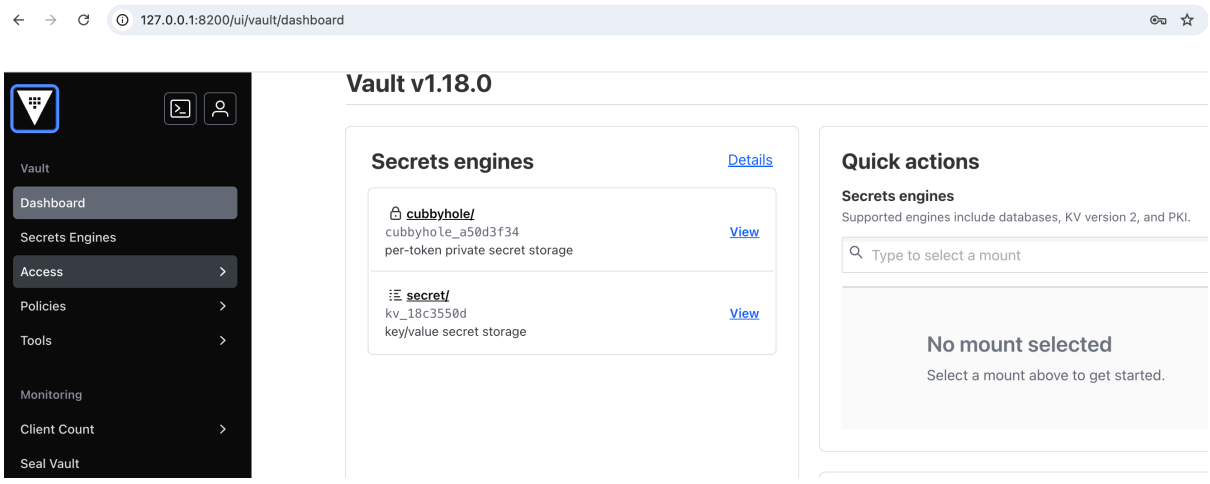
Log in to the Vault server at http://127.0.0.1:8200 using the root token. Here, I am using a Vault dev server, so I log in locally. In a production environment, log in with the domain



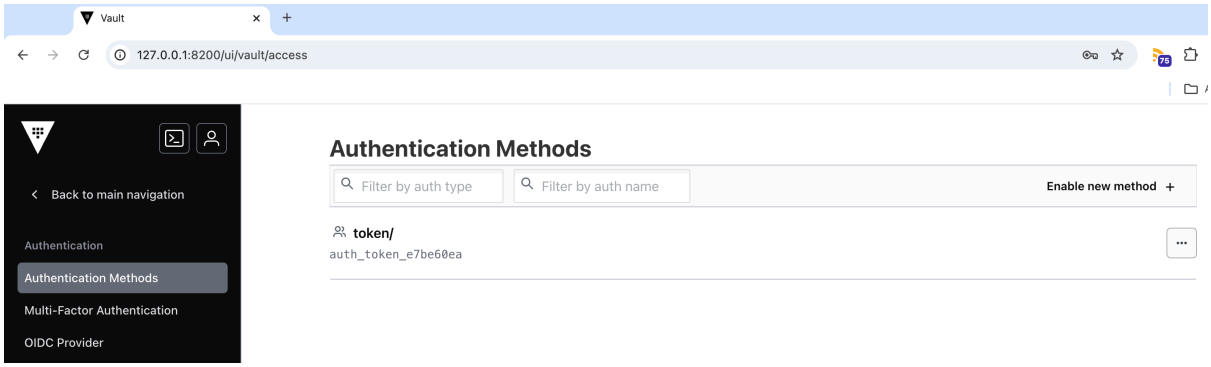
Once signed in, we will be able to see the screen below



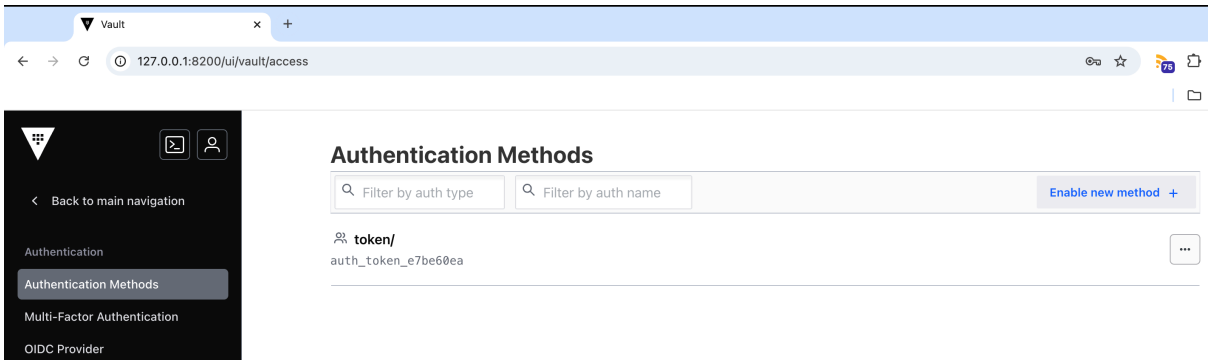
Go to **Access**



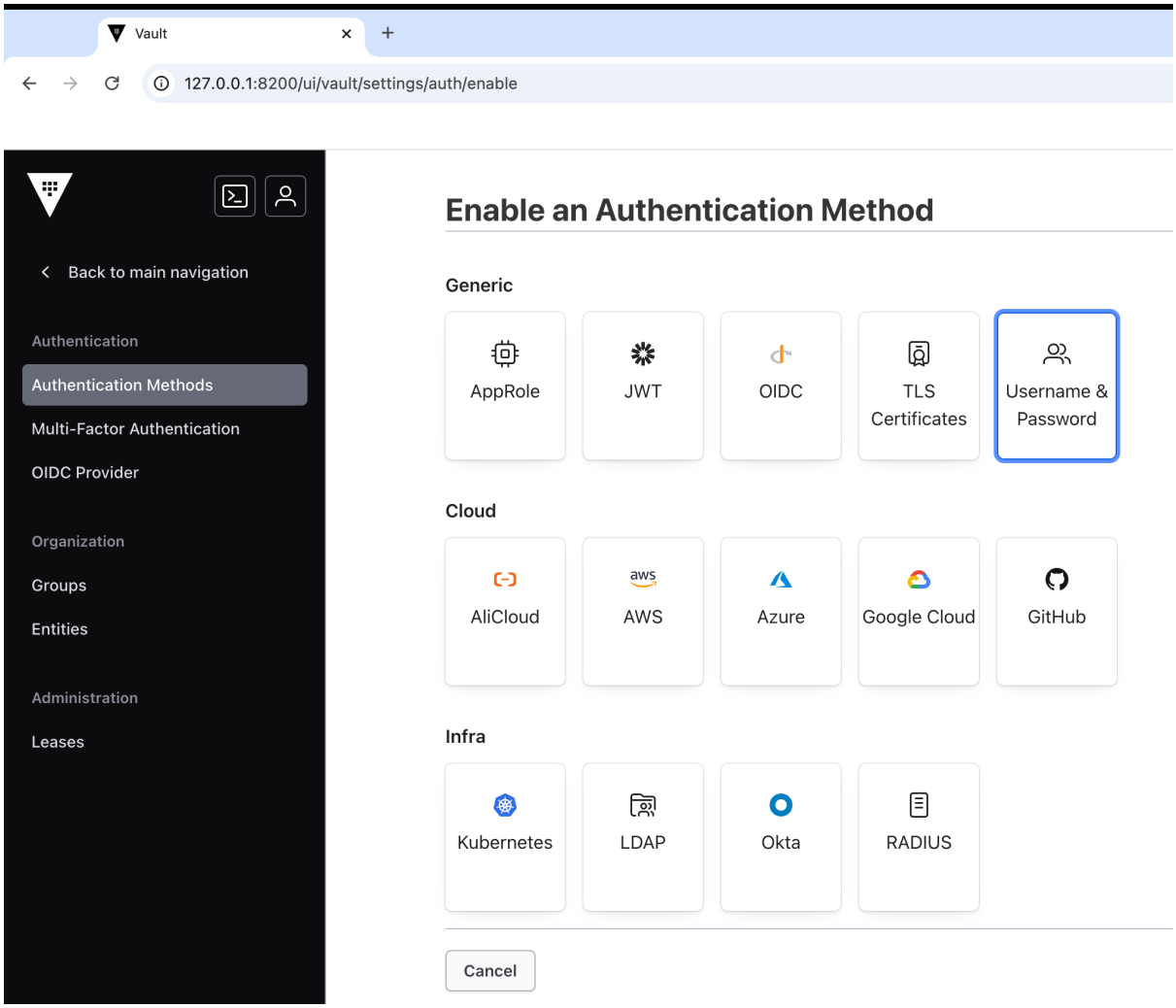
Select Authentication Methods



Click “Enable new Method”



Select “Username and Password”



In the **Path** section, we can change our own path and configure other options, such as the seal method, default lease TTL, and maximum lease TTL. Click **Enable Method**

Enable an Authentication Method

Path

userpass

Hide Method Options

Description

List method when unauthenticated

Local

Seal wrap

Default Lease TTL
Vault will use the default lease duration.

Max Lease TTL
Vault will use the default lease duration.

Token type

Select one

Request keys excluded from HMACing in audit

Add one item per row.

Add

Response keys excluded from HMACing in audit

Add one item per row.

Add

Allowed passthrough request headers

Add one item per row.

Add

Allowed response headers

Add one item per row.

Add

Plugin version

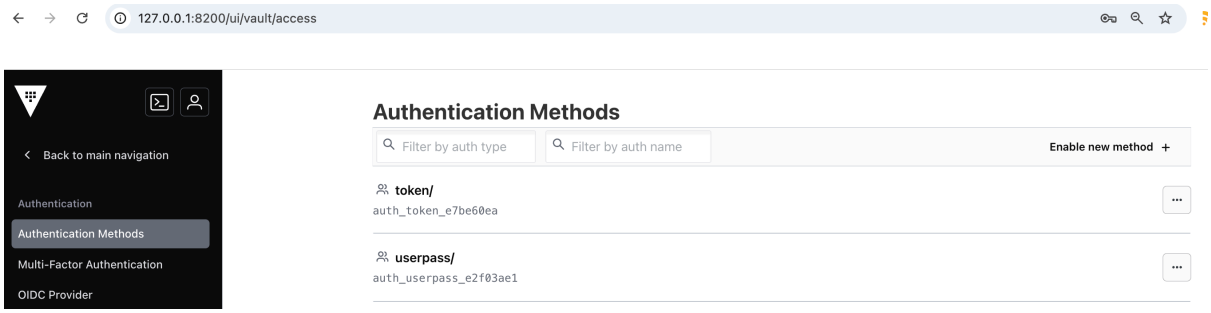
Specifies the semantic version of the plugin to use, e.g. "v1.0.0". If unspecified, the server will select any matching un-versioned plugin that may have been registered, the latest versioned plugin registered, or a built-in plugin in that order of precedence.

Enable method

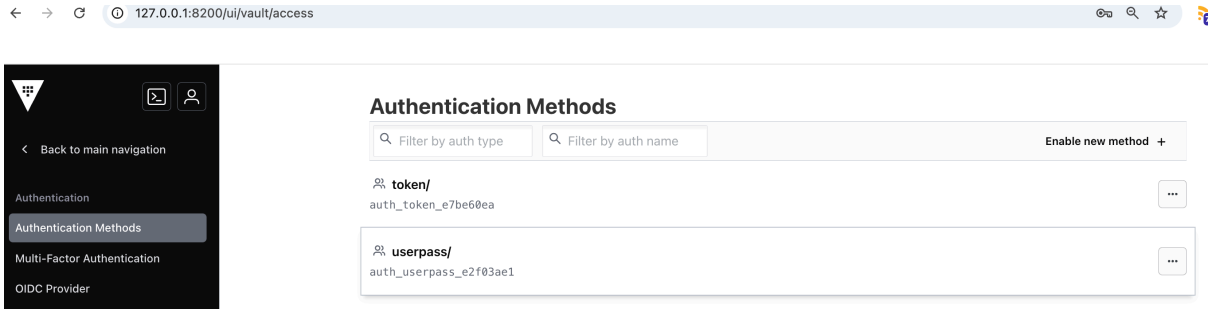
Back

https://www.linkedin.com/in/mahendran-selvakumar-36444a77/

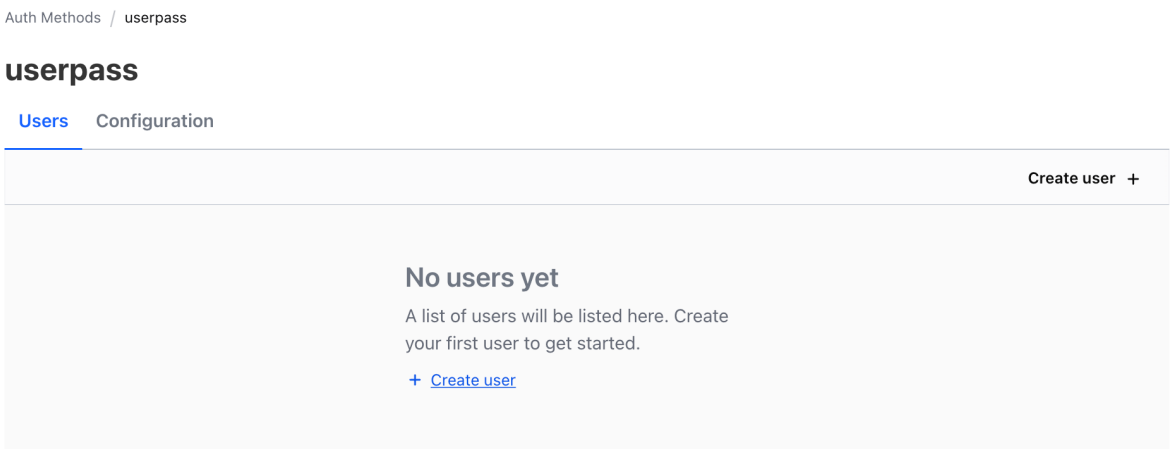
Now we can see the authentication method 'Userpass’



Click “Userpass” Authentication method

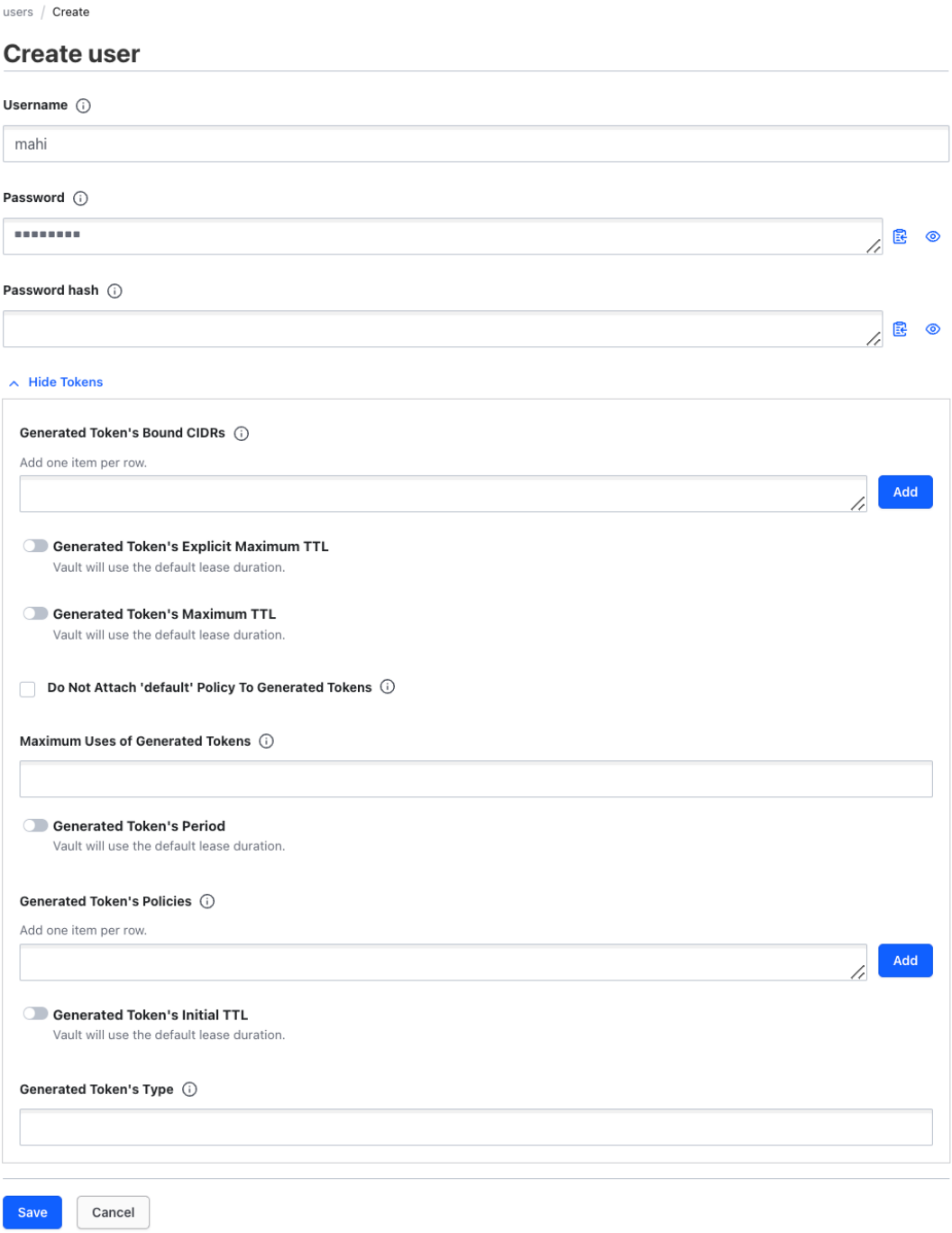


Click “Create User”

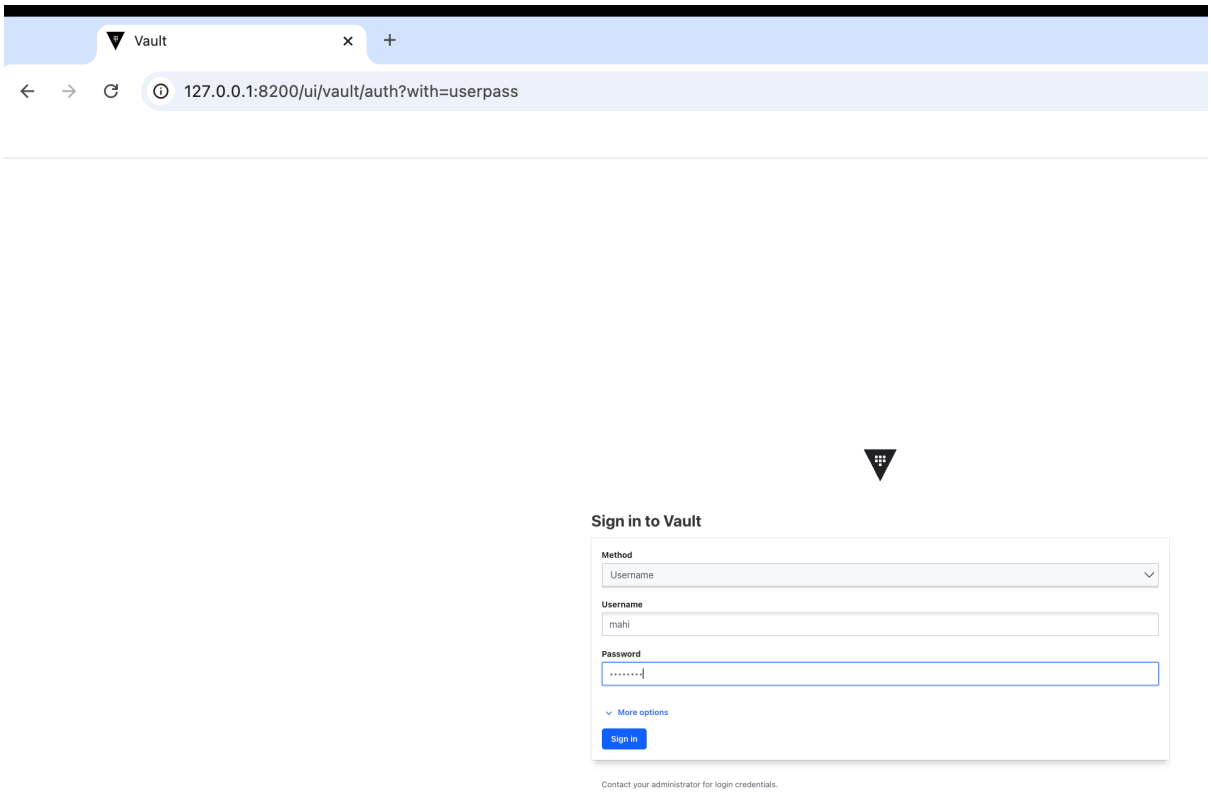


https://www.linkedin.com/in/mahendran-selvakumar-36444a77/

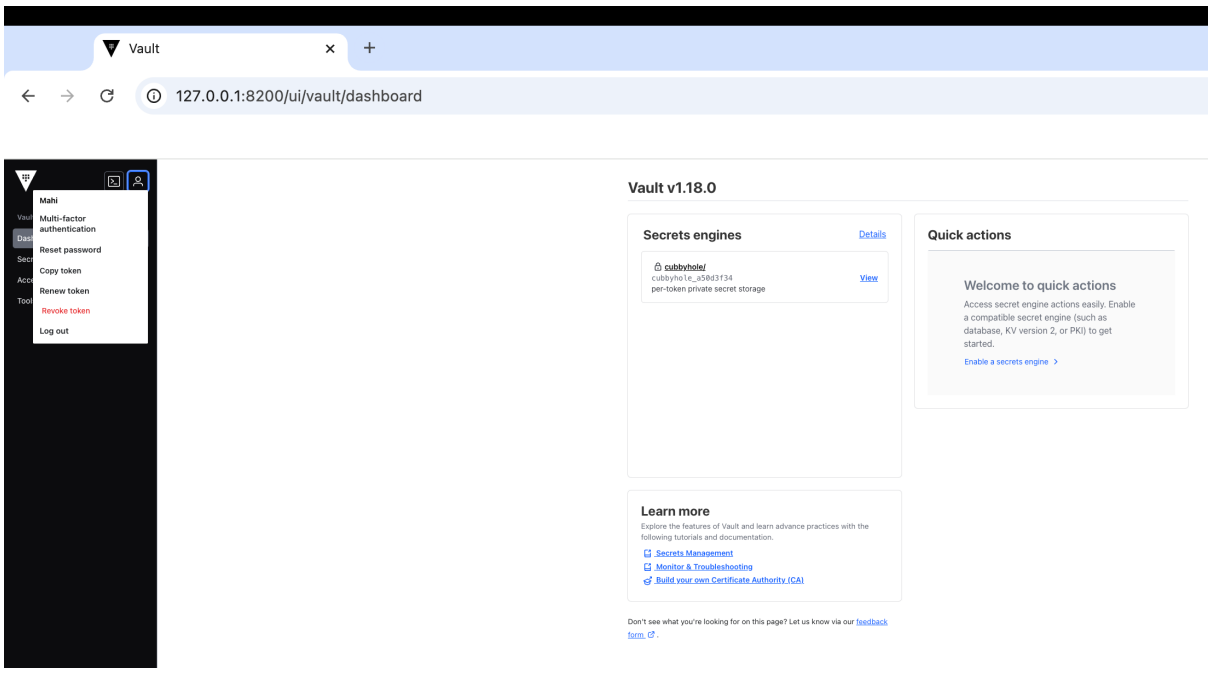
Provide the username and password, then click **Sign In**



Go to the Vault login page, change the method to **Username**, provide the created username and password, and then click **Sign In**



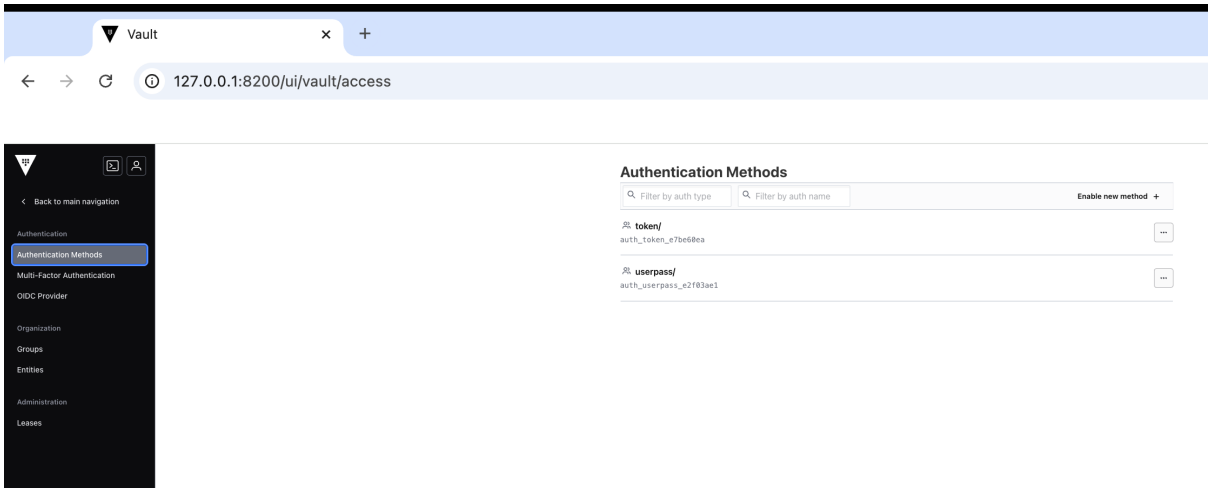
Now we can log in and see the Vault dashboard



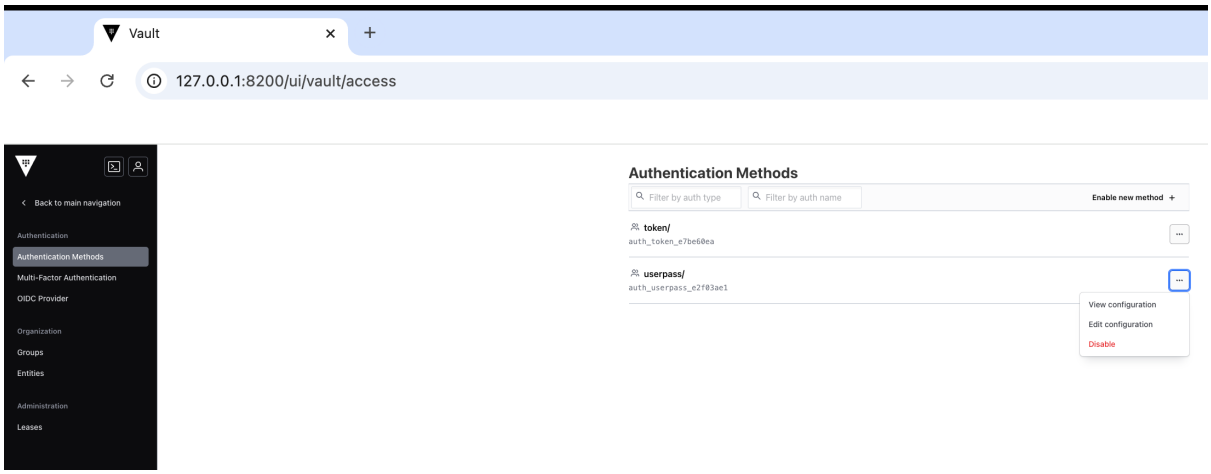
https://www.linkedin.com/in/mahendran-selvakumar-36444a77/

Step4: Disable Vault Authentication Method

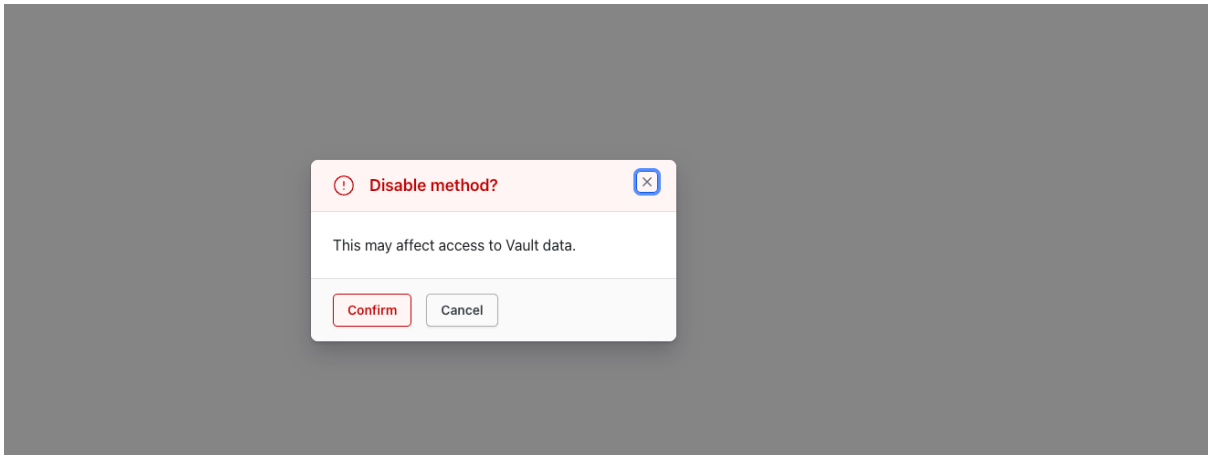
Log in with the root token, navigate to Access, and click on Authentication Methods



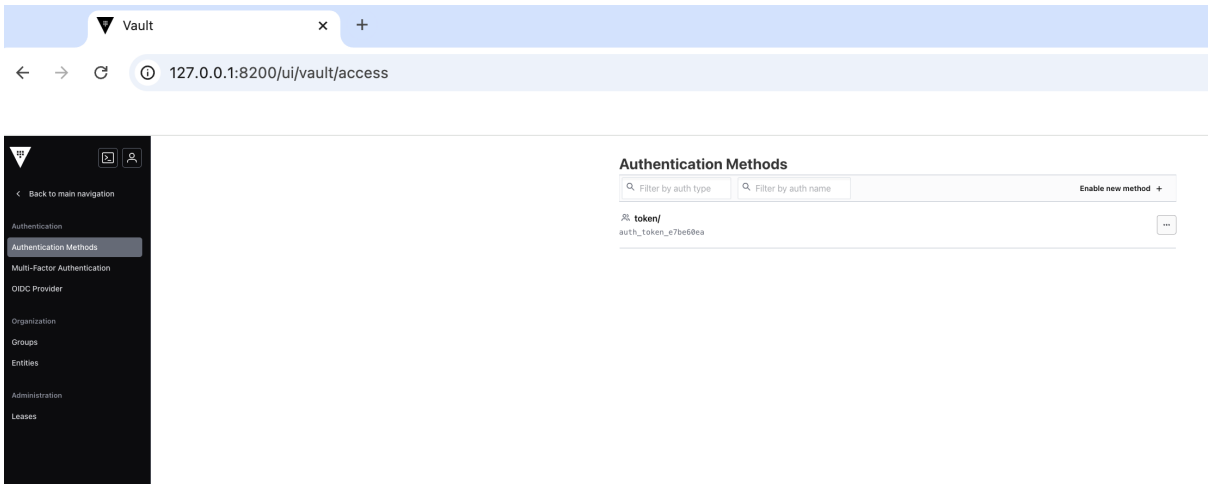
Click the (...) in the userpass authentication method and then click **Disable**



Click “Confirm” to delete



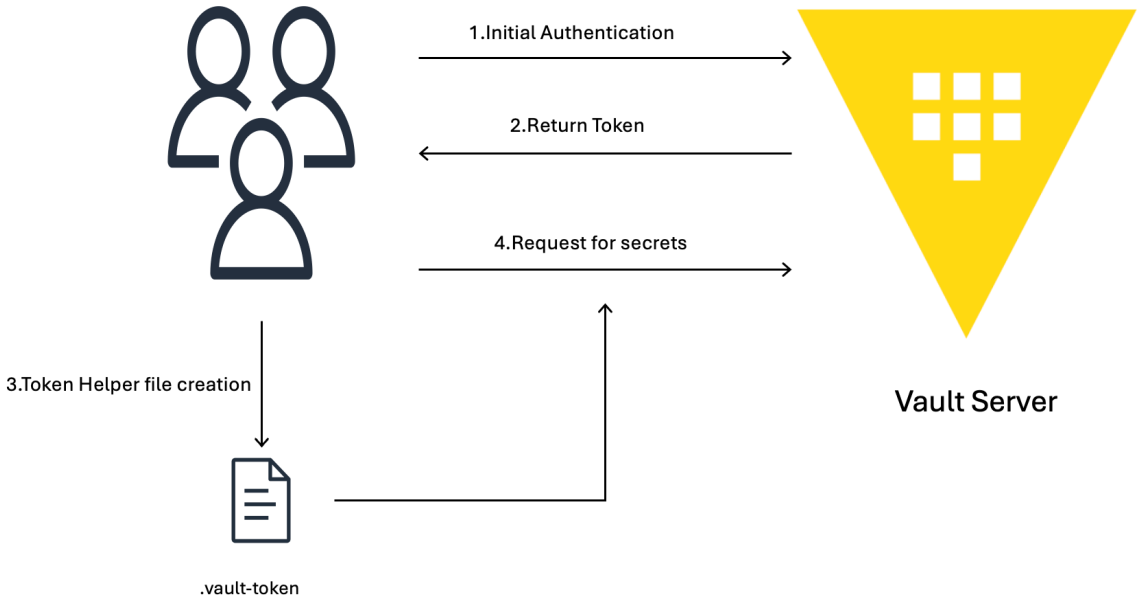
Now the userpass auth method has been disabled, and you can see the default auth methods



What is Vault Token Helper?

Token Helper caches the token after authentication and stores the token in a local file(. vault-token). It can be referenced for subsequent requests

Token Helper Workflow:



Validate the token in. vault-token file after authentication

```
[root@ip-172-31-33-10 ~]# cat .vault-token
hvs.CAESIFvc3ayMgLHkqtp9kUP_R702uWRgEQBBE0AGjzsN2aTpGh4KHGh2cy5SNDl0YnpsellGVjBWUzZsQmZHdnpXbUs
```

Keep Learning, Keep Securing!!

Feel free to reach out to me, if you have any other queries or suggestions

Stay connected on LinkedIn: Mahendran Selvakumar

Stay connected on Medium https://devopstronaut.com/