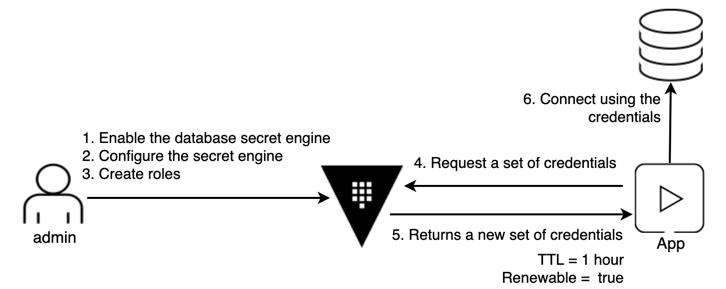
**CONFIGURING HASHICROP VAULT TO GENERATE DYNAMIC DATABASE CREDENTIALS**

In general practice, create a static set of database credentials for an application and either feed them in the source code (very insecure) or export them as environment variables and have the application look them up. Neither of these methods provide the privacy around the credentials that they really should have. Implementing either of the above methods can cause a secrets sprawl.

**SOLUTION:**

So, we can overcome this using Hashicrop’s vault secret engine using which we can generate secrets dynamically which lasts for a specific period of time.

Applications ask Vault for database credentials rather than setting them as environment variables. The administrator specifies the TTL of the database credentials to enforce its validity so that they are automatically revoked when they are no longer used.



Each app instance can get unique credentials that they don't have to share. By making those credentials short-lived, you reduce the chance that they might be compromised. If an app was compromised, the credentials used by the app can be revoked rather than changing more global sets of credentials.

**PREREQUISITES:**

* Vault environment
* Docker for running postgresql/mysql image

**Docker Installation:**

$apt-get update

$apt install docker.io

**DYNAMIC POSTGRESQL CREDENTIALS**

Pull a Postgres server image with docker.

**$** docker pull postgres:latest

Create a Postgres database with a root user named root with the password rootpassword.

**$** docker run \

--name postgres \

--env POSTGRES\_USER=root \

--env POSTGRES\_PASSWORD=rootpassword \

--detach \

--publish 5432:5432 \

postgres

Connect to the Postgres database via the CLI within the postgres container.

**$** docker exec -it postgres psql

Once we are connected we will have a new prompt *root#*

Create a role named ro.

**$** CREATE ROLE ro NOINHERIT;

CREATE ROLE

Grant the ability to read all tables to the role named ro.

**$** GRANT SELECT ON ALL TABLES IN SCHEMA public TO "ro";

GRANT

The role is created and assigned the appropriate permissions.

Disconnect from the Postgres database.

**$** \q

Now, login to your vault server with the root token available,

$vault login

**ENABLE THE DATABASE SECRETS ENGINE**

Enable the database secrets engine at the database/ path.

**$** vault secrets enable database

The database secrets engine is enabled.

**CONFIGURE POSTGRESQL SECRETS ENGINE**

The database secrets engine supports many databases through a plugin interface. To use a Postgres database with the secrets engine requires further configuration with the postgresql-database-plugin plugin and connection information.

Configure the database secrets engine with the connection credentials for the Postgres database.

**$** vault write database/config/postgresql \

plugin\_name=postgresql-database-plugin \

connection\_url="postgresql://{{username}}:{{password}}@54.255.248.129:54 32/postgres?sslmode=disable" \

allowed\_roles=readonly \

username="root" \

password="rootpassword"

## CREATE A ROLE

## First way :

## vault write database/roles/readonly \

## db\_name=postgresql \

## creation\_statements="CREATE ROLE \"{{name}}\" WITH LOGIN PASSWORD '{{password}}' VALID UNTIL '{{expiration}}'; \

## GRANT SELECT ON ALL TABLES IN SCHEMA public TO \"{{name}}\";" \

## default\_ttl="1h" \

## max\_ttl="2h"

## Second way :

In [Step 2](https://learn.hashicorp.com/tutorials/vault/database-secrets#step-2-configure-postgresql-secrets-engine), we configured the PostgreSQL secrets engine with the allowed role named readonly. A role is a logical name within Vault that maps to database credentials. These credentials are expressed as SQL statements and assigned to the Vault role.

Define the SQL used to create credentials.

**$** tee readonly.sql <<EOF

CREATE ROLE "{{name}}" WITH LOGIN PASSWORD '{{password}}' VALID UNTIL '{{expiration}}' INHERIT;

GRANT ro TO "{{name}}";

EOF

The SQL contains the templatized fields {{name}}, {{password}}, and {{expiration}}. These values are provided by Vault when the credentials are created. This creates a new role and then grants that role the permissions defined in the Postgres role named ro. This Postgres role was created when [Postgres was started](https://learn.hashicorp.com/tutorials/vault/database-secrets#start-postgres).

Create the role named readonly that creates credentials with the readonly.sql.

**$** vault write database/roles/readonly \

db\_name=postgresql \

creation\_statements=@readonly.sql \

default\_ttl=1h \

max\_ttl=24h

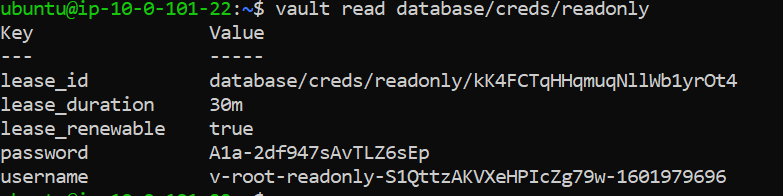
The role generates database credentials with a default TTL of 1 hour and max TTL of 24 hours.

**REQUEST POSTGRESQL CREDENTIALS**

The applications that require the database credentials read them from the secret engine's *readonly* role.

Read credentials from the readonly database role.

$vault read database/creds/readonly



For Generating secrets dynamically using Curl command:

$curl \

    --header "X-Vault-Token:s.IZeqoi10Pez65z1hEqHvbiWb " \

    --request GET \

<http://52.90.88.15:8200/v1/database/creds/readonly>

**VALIDATION**

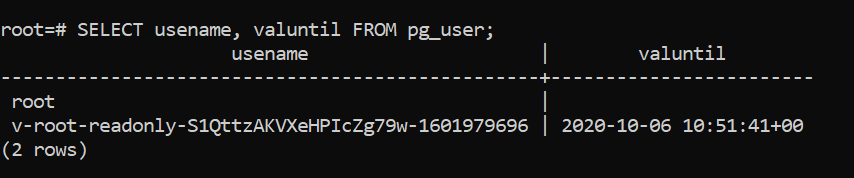
Connect to the Postgres database via the CLI within the postgres container.

**$** docker exec -it postgres psql

Your system prompt is replaced with a new prompt root=#. Commands issued at this prompt are executed against the Postgres database running within the container.

List all the database users.

$SELECT usename, valuntil FROM pg\_user;



The output displays a table of all the database credentials generated. The credentials that were recently generated appear in this list.

Disconnect from the Postgres database.

**$** \q

**DYNAMIC MYSQL CREDENTIALS**

**DOCKER COMMANDS TO RUN MYSQL CONTAINER**

docker pull mysql

docker run --name db -d -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=Welcome123 mysql

**VERIFYING THE USERS**

docker exec -it c737e4e2914e bin/bash

mysql -uroot –pWelcome123

check the users using following command,

select user from mysql.user;

**output :**

user |

+------------------+

| root |

| mysql.infoschema |

| mysql.session |

| mysql.sys |

| root |

+------------------+

**CONFIGURE THE MYSQL DATABASE ENGINE FOR VAULT ON ONE OF THE VAULT VMS**

The following command will create new user by replacing the {{name }} & {{password}} with the username and password which is created dynamically during **vault read**

vault write database/roles/my-role \

db\_name=my-mysql-database \

creation\_statements="CREATE USER '{{name}}'@'%' IDENTIFIED BY '{{password}}';GRANT SELECT ON \*.\* TO '{{name}}'@'%';" \

default\_ttl="1h" \

max\_ttl="1h"

The following command will install the required plugin for mysql and then configure the root user credentials using which a dynamic user credentials are generated.

vault write database/config/my-mysql-database \

plugin\_name=mysql-database-plugin \

connection\_url="{{username}}:{{password}}@tcp(54.255.248.129:3306)/" \

allowed\_roles="my-sql-role" \

username="root" \

password="Welcome123"

Now, perform vault read operation to generate dynamic mysql credentials

vault read database/creds/my-sql-role

**output:**

Key Value

--- -----

lease\_id database/creds/my-role/aGYp7UIjCqrc3rdRBWbUvwdP

lease\_duration 1h

lease\_renewable true

password A1a-kaJNJ0FiTd2BcahZ

username v-root-my-role-XARNzWzG7o6bfev2t

To check the created temporary users in sql :

docker exec -it c737e4e2914e bin/bash

mysql -uroot –pWelcome123

instead of root you can try using following username and password :

mysql -uv-root-my-role-XARNzWzG7o6bfev2t –pA1a-kaJNJ0FiTd2BcahZ

See all the users using following command ,

select user from mysql.user

**output:**

+----------------------------------+

| user |

+----------------------------------+

| root |

| v-root-my-role-2XZ1FFi4NYTGFdKVe |

| v-root-my-role-XARNzWzG7o6bfev2t |

| mysql.infoschema |

| mysql.session |

| mysql.sys |

| root |

+----------------------------------+

**Curl command to generate mysql secrets :**

curl \

--header "X-Vault-Token:s.IZeqoi10Pez65z1hEqHvbiWb " \

--request GET \

http://52.90.88.15:8200/v1/database/creds/my-role

**output:**

{"request\_id":"bbb61326-8a25-2861-e9c6-9d04c33fa39f","lease\_id":"database/creds/my-role/Vv8gD6ly9IXqYGoOhyNXOIXN","renewable":true,"lease\_duration":3600,"data":{"password":"A1a-O9SYxBREGFTmQvvZ","username":"v-root-my-role-r0By9OWlgdizdFjm7"},"wrap\_info":null,"warnings":null,"auth":null}

**Reference links :**

<https://www.vaultproject.io/docs/secrets/databases/mysql-maria>

<https://www.vaultproject.io/api/secret/databases/mysql-maria>

<https://www.vaultproject.io/docs/secrets/databases/postgresql>