How to Manage Secrets

https://blog.cryptomove.com/secrets-management-guide-approaches-open-source-tools-commercial-products-challenges-db560fd0584d (https://blog.cryptomove.com/secrets-management-guide-approaches-open-source-tools-commercial-products-challenges-db560fd0584d)

Secrets aka. digital authentication credentials

What

Type of secrets

- AWS access keys
- SSH Keys
- Database credentials (host, port, username, password)
- APIs and API tokens
- Pem, Cert files
- Accounts: url, username, password, email, secure Q&A

Why

DevOps processes & microservices based architecture

leads to secrets proliferation. Teams undergoing DevOps transformations move fast and manage many different infrastructure environments and services for development, testing, integration, and deployment. Secrets management for DevOps environments is vital as part of the secure software development lifecycle.

Best Practice

7 Best practices by Beyond Trust (https://www.beyondtrust.com/blog/entry/secrets-management-overview-7-best-practices)

- Use **strong passwords** and don't share

- **Remove hardcoded** / embedded secrets in DevOps tool configurations, build scripts, code files, test builds, production builds, applications
- Discover secrets and bring them under **centralized control**
- **Log, audit, and monitor** all privileged sessions (for accounts, users, scripts, automation tools, etc.) to improve oversight and accountability.
- **Threat detection/prevention** – continuously analyze secrets usage to detect anomalies and potential threats
- Adopt **DevSecOps** - build security into each step of DevOps (inception, design, build, test, release, support, maintenance)
- Extend secrets management to **third-parties**

Who

Providers

https://stackshare.io/secrets-management (https://stackshare.io/secrets-management)

- Vault (https://learn.hashicorp.com/vault/) (Hashicorp)
- AWS Key Management Service (KMS) (https://aws.amazon.com/kms/)
- Knox (https://github.com/pinterest/knox) (Pinterest)
- Confidant (https://github.com/lyft/confidant) (Lyft)
- Docker secrets (https://blog.docker.com/2017/02/docker-secrets-management/)
- Keywhiz (https://square.github.io/keywhiz/) (Square)

How

Tutorial - Vault by Hashicorp

- Review (https://thenewstack.io/using-vault-to-manage-your-apps-secrets/)
- Learn Vault (https://learn.hashicorp.com/vault/)
- HashiCorp Vault on AWS (https://aws.amazon.com/quickstart/architecture/vault/)
- setup-hashicorp-vault-beginners-guide (https://devopscube.com/setup-hashicorp-vault-beginners-guide/)
- Installing Vault On AWS Linux (https://gist.github.com/cludden/12ef62dad35aff69e5bb)
- Taking Your Hashicorp Vault To The Next Level (https://www.prodops.io/blog/taking-your-hashicorp-vault-to-the-next-level)

Install

- download one binary
- add to PATH, set 2 env vars

```
$ export VAULT_ADDR="http://127.0.0.1:8200"
$ export VAULT_DEV_ROOT_TOKEN_ID="s.4sSqAnf111111xxxxxxxxxx"
```

- ready to go

```
$ vault
$ vault server -dev    # start dev server
$ vault status
```

Get started - CLI

write a secret

```
$ vault kv put secret/hello foo=world
$ vault kv put secret/test cloud_provider=aws today=2019-02-09
```

get secret

```
$ vault kv get secret/hello
$ vault kv get -field=today secret/hello
$ vault kv get -format=json secret/hello
$ vault kv get -format=json secret/hello | jq -r .data.data.today
```

delete

```
$ vault kv delete secret/test
```

enable another secrets engine besides default secret

```
$ vault secrets enable kv
$ vault write kv/my-secret value="s3c(eT"
$ vault write kv/hello target=world
$ vault write kv/airplane type=boeing class=787
$ vault list kv
```

Python API (https://github.com/hvac/hvac)

```
$ pip install hvac
```

In [1]:
```
import os, hvac
```

In [2]:
```
client = hvac.Client(url=os.environ['VAULT_ADDR'], token=os.environ['VAULT_DEV_ROOT_TOKEN_ID'])
```

In [3]:
```
print(client.read('kv/hello'))
```

```
{'request_id': '4ec50a5f-e3c3-fc5c-1d41-5a373bc1da99', 'lease_id': '', 'renewable': False, 'lease_du
ration': 2764800, 'data': {'target': 'world'}, 'wrap_info': None, 'warnings': None, 'auth': None}
```

In [4]:
```
client.write('kv/postgresql_dev', hostname='metadb.cjnq5am.us-west-1.rds.amazonaws.com', \
             port='5432', user='user1', pwd='PostGreSq123', db_name='iotdb')
```

In [5]:
```
db_secrets = client.read('kv/postgresql_dev')
print(db_secrets)
```

```
{'request_id': '36255d2e-aa72-bdbd-4e0b-895a517de66e', 'lease_id': '', 'renewable': False, 'lease_du
ration': 2764800, 'data': {'db_name': 'iotdb', 'hostname': 'metadb.cjnq5am.us-west-1.rds.amazonaws.c
om', 'port': '5432', 'pwd': 'PostGreSq123', 'user': 'user1'}, 'wrap_info': None, 'warnings': None,
'auth': None}
```

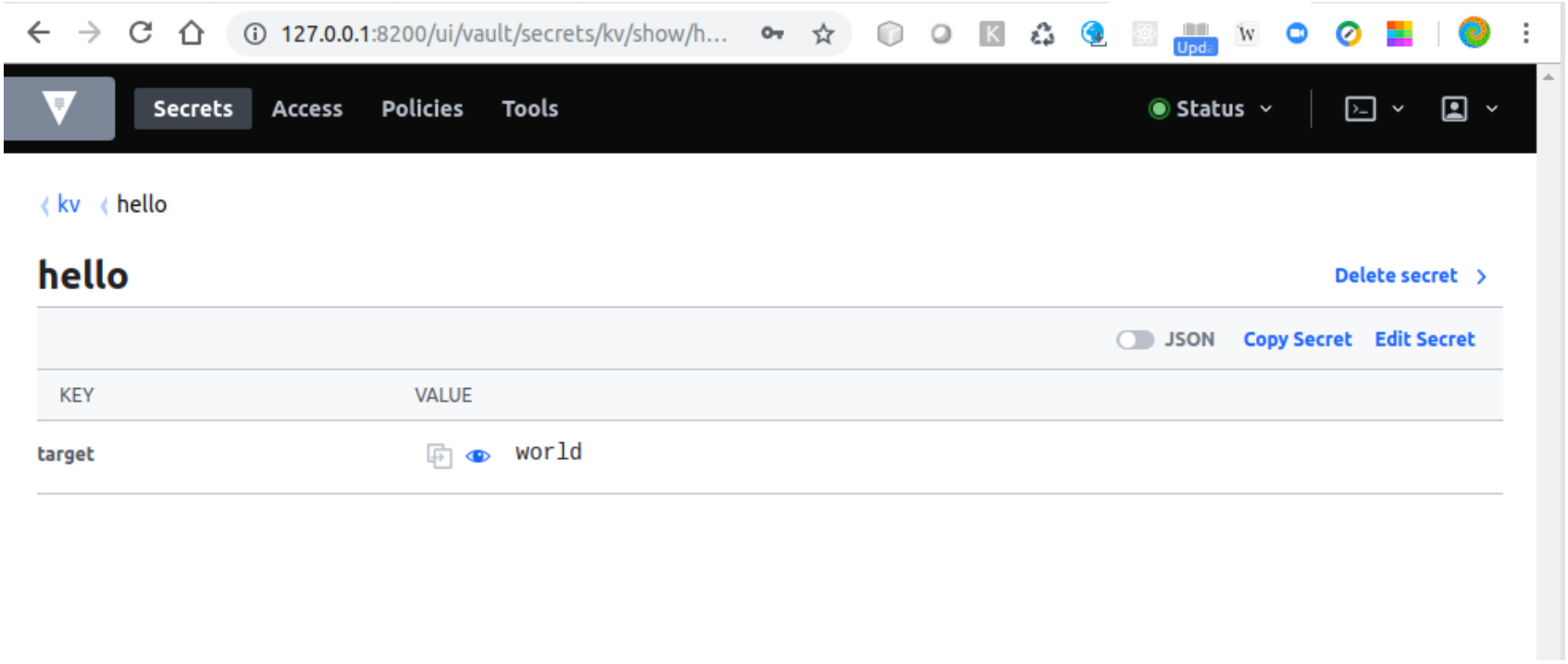
In [6]:
```
db_host,db_port,table_name,db_user,db_pwd = \
    db_secrets['data']['hostname'], \
    db_secrets['data']['port'], \
    db_secrets['data']['db_name'], \
    db_secrets['data']['user'], \
    db_secrets['data']['pwd']
```

In [7]:
```
print([db_host,db_port,table_name,db_user,db_pwd])
```

```
['metadb.cjnq5am.us-west-1.rds.amazonaws.com', '5432', 'iotdb', 'user1', 'PostGreSq123']
```

Web UI

- http://127.0.0.1:8200/ui (http://127.0.0.1:8200/ui)


127.0.0.1:8200/ui/vault/secrets/kv/show/h...
Secrets
Access
Policies
Tools
Status
kv
hello
hello
Delete secret
JSON
Copy Secret
Edit Secret
KEY
VALUE
target
world

In [ ]: