GEHEIMEN MANAGER > AAN DE SLAG

Snel aan de slag voor ontwikkelaars

Snel aan de slag voor ontwikkelaars

Met Bitwarden Secrets Manager kunnen ontwikkelaars, DevOps- en cyberbeveiligingsteams geheimen centraal opslaan, beheren en op schaal inzetten. De Secrets Manager CLI is het primaire middel om secrets in je applicaties en infrastructuur te injecteren via een geauthenticeerde service account.

In dit artikel demonstreren we het gebruik van de Secrets Manager CLI door te kijken naar een paar manieren om databasegegevens op te halen die zijn opgeslagen in je kluis en die worden geïnjecteerd tijdens containerruntime voor een Bitwarden Unified Docker-image.

> **Tip**
>
> Als je op zoek bent naar SDK-informatie en taalwrappers voor de functionaliteit van Secrets Manager, raadpleeg dan dit artikel.

Als je het artikel Secrets Manager Quick Start nog niet hebt gelezen, raden we je aan om dat te doen voordat je verder leest.

Basishandleiding

In dit meest eenvoudige voorbeeld haal je databasegegevens op die zijn opgeslagen in je kluis en sla je ze op als tijdelijke omgevingsvariabelen op een Linux systeem. Eenmaal opgeslagen, injecteer je ze tijdens runtime in een `docker run` commando. Hiervoor moet je het volgende geïnstalleerd hebben:

- Bitwarden Geheimen Manager CLI
- Docker
- Een opdrachtregel JSON-verwerker zoals jq

Authenticeren

Er kan ingelogd worden op de Secrets Manager CLI met een toegangstoken dat gegenereerd is voor een bepaalde serviceaccount. Dit betekent dat **alleen geheimen en projecten waartoe de service account toegang heeft** , benaderd mogen worden met de CLI (leer meer over service account permissies). Er zijn een aantal manieren om een CLI-sessie te authenticeren, maar de eenvoudigste manier is door bijvoorbeeld een omgevingsvariabele `BWS_ACCESS_TOKEN` op te slaan met de waarde van je toegangstoken:

```bash
export BWS_ACCESS_TOKEN=0.48c78342-1635-48a6-accd-afbe01336365.C0tMmQqHnAp1h0gL8bngprlPOYutt0:B3h5D+YgLvFiQhWkIq6Bow==
```

Het geheim ophalen

Gebruik vervolgens het volgende commando om uw databasegebruikersnaam op te halen en deze op te slaan als een tijdelijke omgevingsvariabele. In dit voorbeeld staat `fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff` voor de specifieke identifier voor de database gebruikersnaam geheim:

```bash
export SECRET_1=$(bws secret get fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff | jq '.value')
```

Dit commando slaat de `waarde` van je geheim op in een tijdelijke omgevingsvariabele, die wordt gewist als het systeem opnieuw opstart, de gebruiker uitlogt of in een nieuwe shell. Voer nu hetzelfde commando uit voor het databasewachtwoord:

```bash
export SECRET_2=$(bws secret get 80b55c29-5cc8-42eb-a898-acfd01232bbb | jq '.value')
```

Injecteer het geheim

Nu je databasegegevens zijn opgeslagen als tijdelijke omgevingsvariabelen, kunnen ze worden geïnjecteerd in een `docker run` commando. In dit voorbeeld hebben we veel variabelen weggelaten die Bitwarden Unified nodig heeft om de nadruk te leggen op de geïnjecteerde geheimen:

```bash
docker run -d --name bitwarden .... -env BW_DB_USERNAME=$SECRET_1 BW_BD_PASSWORD=$SECRET_2 .... bit
warden/self-host:beta
```

Wanneer dit commando wordt uitgevoerd, zal je Docker-container opstarten en je databasegegevens injecteren vanuit de tijdelijk opgeslagen omgevingsvariabelen, zodat je Bitwarden Unified veilig kunt opstarten zonder gevoelige waarden als platte tekst door te geven.

Tips voor gevorderden

In dit voorbeeld gebruik je de Secrets Manager CLI in je Docker image om database referenties opgeslagen in je vault tijdens runtime te injecteren. Je zult dit bereiken door je Dockerfile te manipuleren om de CLI op de image te installeren, in plaats van op de host, en om de databasegegevens op te halen tijdens de runtime van de container. Vervolgens maak je je omgevingsvariabelenbestand klaar voor injectie en rijg je alles aan elkaar door een container uit te voeren.

Dockerbestand instellen

Om de Secrets Manager CLI in je Docker image te installeren, moet je het volgende toevoegen aan je Dockerfile:

```bash
RUN curl -O https://github.com/bitwarden/sdk/releases/download/bws-v1.0.0/bws-x86_64-unknown-linux-
gnu-1.0.0.zip && unzip bws-x86_64-unknown-linux-gnu-1.0.0.zip && export PATH=/this/directory:$PATH
```

Vervolgens moet je `RUN` statements maken om elke credential op te halen en beschikbaar te maken voor injectie. Deze verklaringen bevatten inline authenticatie, maar dit is niet de enige methode die je kunt implementeren:

```bash
RUN SECRET_1=$(bws secret get fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff --access-token $BWS_ACCESS_TOKEN
| jq '.value')
```

```
Bash

RUN SECRET_2=$(bws secret get 80b55c29-5cc8-42eb-a898-acfd01232bbb --access-token $BWS_ACCESS_TOKEN
| jq '.value')
```

Deze `RUN-instructies` vragen je Dockerfile om de aangegeven secrets op te halen, waarbij `fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff` de specifieke identifier van de secret is.

Bereid uw env-bestand voor

Nu je databasegegevens beschikbaar zijn voor injectie, moet je het bestand `settings.env` zo instellen dat het deze waarden kan ontvangen. Om dit te doen, vervang je relevante hardcoded waarden in het bestand door de aangewezen variabelenamen (in dit geval `SECRET_1` en `SECRET_2`):

```
Bash

# Database
# Available providers are sqlserver, postgresql, mysql/mariadb, or sqlite
BW_DB_PROVIDER=mysql
BW_DB_SERVER=db
BW_DB_DATABASE=bitwarden_vault
BW_DB_USERNAME=$SECRET_1
BW_DB_PASSWORD=$SECRET_2
```

De container uitvoeren

Nu je databasereferenties klaar zijn voor injectie, start je je container op en specificeer je het toegangstoken om te gebruiken met `bws login` als omgevingsvariabele:

```
Bash

docker run --rm -it -e BWS_ACCESS_TOKEN=<your-access-token> image-name
```

Wanneer dit commando wordt uitgevoerd, zal je Docker-container opstarten en je databasegegevens injecteren uit de waarden die door de container zijn opgehaald, zodat je Bitwarden Unified veilig kunt opstarten zonder gevoelige waarden als platte tekst door te geven.