SECRETS MANAGER > COMMENCEZ

Démarrage Rapide du Développeur

Démarrage Rapide du Développeur

Bitwarden Secrets Manager permet aux développeurs, DevOps et aux équipes de cybersécurité de stocker, gérer et déployer des secrets à grande échelle de manière centralisée. Le CLI de Secrets Manager est votre principal moyen d'injecter des secrets dans vos applications et votre infrastructure via un compte de service authentifié.

Dans cet article, nous démontrerons l'utilisation du CLI de Secrets Manager en examinant quelques façons de récupérer les identifiants de base de données stockés dans votre coffre pour être injectés au moment de l'exécution du conteneur pour une image Docker Bitwarden Unified.

> **Tip**
>
> Si vous recherchez des informations sur le SDK et des enveloppes de langage pour la fonctionnalité Secrets Manager, reportez-vous à cet article.

Si vous n'avez pas déjà consulté l'article Démarrage rapide de Secrets Manager, nous vous recommandons de le faire avant de continuer à lire.

Tutoriel de base

Dans cet exemple le plus simple, vous récupérerez les identifiants de la base de données stockés dans votre coffre et les stockerez en tant que variables d'environnement temporaires sur un système Linux. Une fois stockés, vous les injecterez au moment de l'exécution à l'intérieur d'une commande `docker run`. Pour faire cela, vous aurez besoin d'avoir installé :

- Bitwarden Secrets Manager CLI
- Docker
- Un processeur JSON en ligne de commande comme jq

Authentifiez

Le CLI de Secrets Manager peut se connecter en utilisant un jeton d'accès généré pour un compte de service particulier. Cela signifie que **seuls les secrets et les projets auxquels le compte de service a accès** peuvent être manipulés à l'aide du CLI (en savoir plus sur les autorisations de compte de service). Il existe un certain nombre de façons d'authentifier une session CLI, mais pour l'option la plus simple, faites-le en enregistrant une variable d'environnement `BWS_ACCESS_TOKEN` avec la valeur de votre jeton d'accès, par exemple :

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

Récupère le secret

Ensuite, utilisez la commande suivante pour récupérer votre nom d'utilisateur de base de données et le stocker en tant que variable d'environnement temporaire. Dans cet exemple, `fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff` représente l'identifiant spécifique pour le secret du nom d'utilisateur de la base de données:

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

Cette commande va enregistrer la `valeur` de votre secret dans une variable d'environnement temporaire, qui sera effacée lors du redémarrage du système, de la déconnexion de l'utilisateur, ou dans tout nouveau shell. Maintenant, exécutez la même commande pour le mot de passe de la base de données :

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

Injecte le secret

Maintenant que vos identifiants de base de données sont enregistrés en tant que variables d'environnement temporaires, ils peuvent être injectés à l'intérieur d'une commande `docker run`. Dans cet exemple, nous avons omis de nombreuses variables requises par Bitwarden Unified pour mettre l'accent sur les secrets injectés :

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

Lorsque cette commande est exécutée, votre conteneur Docker démarre et injecte vos identifiants de base de données à partir des variables d'environnement stockées temporairement, vous permettant de démarrer Bitwarden Unified en toute sécurité sans transmettre de valeurs sensibles en texte brut.

Tutoriel avancé

Dans cet exemple, vous utiliserez le CLI de Secrets Manager dans votre image Docker pour injecter les identifiants de la base de données stockés dans votre coffre au moment de l'exécution. Vous accomplirez cela en manipulant votre Dockerfile pour installer le CLI sur l'image, au lieu de sur l'hôte, et pour récupérer les identifiants de la base de données au moment de l'exécution du conteneur. Vous préparerez ensuite votre fichier de variables d'environnement pour l'injection et vous lierez le tout en exécutant un conteneur.

Configurez votre Dockerfile

Pour installer le CLI de Secrets Manager dans votre image Docker, vous devrez ajouter ce qui suit à votre 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
```

Ensuite, vous devrez construire des instructions `RUN` pour récupérer chaque identifiant afin de les rendre disponibles pour l'injection. Ces déclarations incluront une authentification intégrée, cependant ce n'est pas la seule méthode que vous pourriez mettre en œuvre :

```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')
```

Ces instructions `RUN` inciteront votre Dockerfile à récupérer les secrets indiqués, où `fc3a93f4-2a16-445b-b0c4-aeaf0102f0ff` représente l'identifiant spécifique du secret.

Préparez votre fichier env

Maintenant que vos identifiants de base de données seront disponibles pour injection, conditionnez votre fichier `settings.env` pour être capable de recevoir ces valeurs. Pour ce faire, remplacez les valeurs codées en dur pertinentes dans le fichier par les noms de variables désignés (dans ce cas, `SECRET_1` et `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
```

Exécutez le conteneur

Maintenant que vos identifiants de base de données sont prêts et prêts pour l'injection, démarrez votre conteneur et spécifiez le jeton d'accès à utiliser avec `bws identifiant` comme variable d'environnement :

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

Lorsque cette commande est exécutée, votre conteneur Docker démarre et injecte vos identifiants de base de données à partir des valeurs récupérées par le conteneur, vous permettant de démarrer Bitwarden Unified en toute sécurité sans transmettre des valeurs sensibles en texte brut.