bitwarden | Article du Centre d'Aide

PASSWORD MANAGER > OUTILS DE DÉVELOPPEMENT

Gestionnaire de mots de passe CLI

Afficher dans le centre d'aide:
https://bitwarden.com/help/cli/

Gestionnaire de mots de passe CLI

L'interface en ligne de commande (CLI) de Bitwarden est un outil puissant et entièrement fonctionnel pour accéder à votre coffre et le gérer. La plupart des fonctionnalités que vous trouvez dans d'autres applications client Bitwarden (bureau, extension de navigateur, etc.) sont disponibles depuis le CLI.

Bitwarden CLI

Le CLI de Bitwarden est auto-documenté. Depuis la ligne de commande, apprenez les commandes disponibles en utilisant :

```bash
bw --help
```

Ou, passez `--help` comme une option sur n'importe quelle commande `bw` pour voir les options disponibles et des exemples :

```Bash
bw list --help

bw move --help
```

La plupart des informations dont vous aurez besoin peuvent être accessibles en utilisant `--help`, cependant cet article reproduit toutes ces informations et approfondit certains sujets.

Téléchargez et installez

Le CLI peut être utilisé sur plusieurs plateformes telles que Windows, macOS et les distributions Linux. Pour télécharger et installer le Bitwarden CLI :

> **Note**
> Pour les appareils arm64, installez le CLI en utilisant `npm`.

⇒Exécutable Natif

Des versions nativement emballées du CLI sont disponibles pour chaque plateforme et n'ont aucune dépendance. Téléchargez en utilisant l'un de ces liens:

- Windows x64
- macOS x64
- Linux x64

Notez que, lorsque vous utilisez l'exécutable natif téléchargé, vous devrez ajouter l'exécutable à votre PATH ou exécuter les commandes à partir du répertoire où le fichier a été téléchargé.

> **Tip**
> Dans les systèmes Linux et UNIX, vous pourriez recevoir un message `Autorisation refusée`. Si vous le faites, accordez l'autorisation en exécutant :
>
> ```Bash
> chmod +x </path/to/executable>
> ```

⇒NPM

Si vous avez Node.js installé sur votre système, vous pouvez installer le CLI en utilisant NPM. L'installation avec NPM est la manière la plus simple de maintenir votre installation à jour et devrait être la **méthode préférée pour ceux qui sont déjà à l'aise avec NPM** :

```bash
npm install -g @bitwarden/cli
```

Afficher le paquet sur npmjs.org.

> **Note**
> L'installation du CLI Bitwarden sur les systèmes Linux en utilisant `npm` peut nécessiter l'installation préalable de la dépendance build-essential (ou équivalent de distribution). Par exemple:
>
> ```plain text
> apt install build-essential
> ```

⇒Chocolaté

Pour installer avec Chocolatey :

```bash
choco install bitwarden-cli
```

Afficher le paquet sur community.chocolatey.org.

⇒Clic

Pour installer avec snap :

```bash
sudo snap install bw
```

Afficher le paquet sur snapcraft.io.

Se connecter

Avant de vous connecter, assurez-vous que votre CLI est connecté au bon serveur (par exemple, cloud EU ou auto-hébergé) en utilisant la commande de configuration (en savoir plus). Il existe trois méthodes pour se connecter à la CLI Bitwarden en utilisant la commande `identifiant`, chacune étant adaptée à différentes situations. Veuillez examiner les options suivantes pour déterminer quelle méthode utiliser :

- Utilisation du courriel et du mot de passe principal
- Utilisation d'une clé API
- Utilisation de SSO

Peu importe l'option que vous utilisez, assurez-vous toujours d'utiliser les commandes `bw verrouiller` ou `bw déconnexion` lorsque vous avez terminé.

> **Tip**
>
> Se connecter en utilisant le courriel et le mot de passe principal utilise votre mot de passe principal et peut donc enchaîner les commandes d'`identifiant` et de `déverrouillage` pour authentifier votre identité et déchiffrer votre coffre en tandem. L'utilisation d'une clé API ou d'un SSO vous obligera à suivre la commande d'`identifiant` avec un `bw déverrouiller` explicite si vous allez travailler directement avec les données du coffre.
>
> C'est parce que votre mot de passe principal est la source de la clé nécessaire pour déchiffrer les données du coffre. Il existe cependant quelques commandes qui ne nécessitent pas que votre coffre soit déchiffré, y compris `config`, `encode`, `générer`, `mettre à jour`, et `status`.

En utilisant le courriel et le mot de passe

Se connecter avec courriel et mot de passe est **recommandé pour les sessions interactives**. Pour se connecter avec courriel et mot de passe:

```Bash
bw login
```

Cela déclenchera une invite pour votre **Adresse de Courriel**, **Mot de Passe Principal**, et (si activé) à **Code d'Identifiant en Deux Étapes**. Le CLI prend actuellement en charge l'identifiant en deux étapes via l'authentificateur, le courriel, ou YubiKey.

Vous *pouvez* assembler ces facteurs en une seule commande comme dans l'exemple suivant, cependant cela n'est pas recommandé pour des raisons de sécurité :

```Bash
bw login [email] [password] --method <method> --code <code>
```

Voir Enums pour les valeurs d'identifiant en deux étapes .

> **Tip**
>
> Êtes-vous invité à fournir une authentification supplémentaire ou recevez-vous une erreur `Votre demande d'authentification semble provenir d'un bot.` ? Utilisez votre clé API `client_secret` pour répondre au défi d'authentification. En savoir plus.

Utilisation d'une clé API

Se connecter avec la clé API personnelle est recommandé pour les flux de travail automatisés, pour fournir un accès à une application externe, ou si votre compte utilise une méthode 2FA non prise en charge par le CLI (FIDO2 ou Duo). Pour se connecter avec la clé API :

```bash
bw login --apikey
```

Cela déclenchera une invite pour votre `client_id` et `client_secret` personnels. Une fois que votre session est authentifiée en utilisant ces valeurs, vous pouvez utiliser la commande `déverrouiller`. En savoir plus.

> **Tip**
> Si votre organisation nécessite SSO, vous pouvez toujours utiliser `--apikey` pour vous connecter au CLI.

Utilisation de variables d'environnement de clé API

Dans les scénarios où un travail automatisé est effectué avec le CLI Bitwarden, vous pouvez enregistrer des variables d'environnement pour éviter la nécessité d'une intervention manuelle lors de l'authentification.

Nom de la variable d'environnement	Valeur requise
ID_CLIENTBW	`identifiant du client`
BW_SECRETCLIENT	`secret_client`

Utilisation de SSO

Se connecter avec SSO est recommandé si une organisation nécessite une authentification SSO. Pour se connecter avec SSO :

```bash
bw login --sso
```

Cela initiera le flux d'authentification SSO dans votre navigateur web. Une fois votre session authentifiée, vous pouvez utiliser la commande `déverrouiller`. En savoir plus.

> **Tip**
> Si votre organisation nécessite SSO, vous pouvez alternativement utiliser `--apikey` pour vous connecter au CLI.

Connectez-vous à plusieurs comptes

Comme l'utilisation de la commutation de compte sur d'autres applications Bitwarden, le CLI a la capacité de se connecter à plusieurs comptes simultanément en utilisant la variable d'environnement `BITWARDENCLI_APPDATA_DIR` pointant vers l'emplacement d'un fichier de configuration `bw`, généralement nommé `data.json`. Vous pouvez, par exemple, définir des alias dans un profil `.bashrc` pour deux paramètres distincts :

```Bash
alias bw-personal="BITWARDENCLI_APPDATA_DIR=~/.config/Bitwarden\ CLI\ Personal /path/to/bw $@"
alias bw-work="BITWARDENCLI_APPDATA_DIR=~/.config/Bitwarden\ CLI\ Work /path/to/bw $@"
```

En utilisant cet exemple, vous pourriez alors utiliser l'identifiant pour deux comptes en exécutant d'abord `source /path/to/.bashrc`, suivi de `bw-personal identifiant` et `bw-work identifiant`.

Déverrouiller

L'utilisation d'une clé API ou SSO pour se connecter nécessitera que vous suiviez la commande d'`identifiant` avec un `bw déverrouiller` explicite si vous allez travailler directement avec les données du coffre.

Le déverrouillage de votre coffre génère une **clé de session** qui agit comme une clé de déchiffrement utilisée pour interagir avec les données dans votre coffre. La clé de session doit être utilisée pour effectuer toute commande qui touche aux données du coffre (par exemple, `liste`, `obtenir`, `éditer`). Les clés de session sont valides jusqu'à ce qu'elles soient invalidées en utilisant `bw verrouiller` ou `bw déconnexion`, cependant elles ne persisteront pas si vous ouvrez une nouvelle fenêtre de terminal. Générez une nouvelle clé de session à tout moment en utilisant :

```Bash
bw unlock
```

Lorsque vous avez terminé, terminez toujours votre session en utilisant la commande `bw verrouiller`.

Options de déverrouillage

Vous pouvez utiliser les options `--passwordenv` ou `--passwordfile` avec `bw déverrouiller` pour récupérer votre mot de passe principal plutôt que de le saisir manuellement, par exemple :

1. La suite recherchera une variable d'environnement `BW_PASSWORD`. Si `BW_PASSWORD` n'est pas vide et a des valeurs correctes, le CLI réussira à déverrouiller et renverra une clé de session :

   ```Bash
   bw unlock --passwordenv BW_PASSWORD
   ```

2. Ce qui suit recherchera le fichier `~Users/Me/Documents/mp.txt` (qui doit avoir votre mot de passe principal comme première ligne). Si le fichier n'est pas vide et a une valeur correcte, le CLI réussira à déverrouiller et renvoyer une clé de session :

   ```Bash
   bw unlock --passwordfile ~/Users/Me/Documents/mp.txt
   ```

> ⚠ **Warning**
>
> Si vous utilisez l'option `--passwordfile`, protégez votre fichier de mot de passe en verrouillant l'accès uniquement à l'utilisateur qui doit exécuter `bw déverrouiller` et en ne fournissant qu'un accès en lecture seule à cet utilisateur.

En utilisant une clé de session

Lorsque vous déverrouillez votre coffre en utilisant `bw identifiant` avec courriel et mot de passe ou `bw déverrouiller`, le CLI renverra à la fois une commande `exporter BW_SESSION` (Bash) et `env:BW_SESSION` (PowerShell), incluant votre clé de session. Copiez et collez l'entrée pertinente pour enregistrer la variable d'environnement requise.

Avec la variable d'environnement BW_SESSION définie, les commandes `bw` feront référence à cette variable et pourront être exécutées proprement, par exemple :

```
Bash

export BW_SESSION="5PBYGU+5yt3RHcCjoeJKx/wByU34vokGRZjXpSH7Ylo8w=="

bw list items
```

Alternativement, si vous ne définissez pas la variable d'environnement, vous pouvez passer la clé de session en option avec chaque commande `bw` :

```
Bash

bw list items --session "5PBYGU+5yt3RHcCjoeJKx/wByU34vokGRZjXpSH7Ylo8w=="
```

> 💡 **Tip**
>
> Lorsque vous avez terminé, terminez toujours votre session en utilisant les commandes `bw verrouiller` ou `bw déconnexion`. Cela invalidera la clé de session active.

Commandes Principales

créer

La commande `créer` crée un nouvel objet (`élément`, `pièce jointe`, et plus) dans votre coffre :

```
Bash

bw create (item|attachment|folder|org-collection) <encodedJson> [options]
```

La commande `créer` prend du JSON encodé. Un flux de travail typique pour créer un objet pourrait ressembler à quelque chose comme :

1. Utilisez la commande `obtenir le modèle` (voir obtenir les commandes principales pour plus de détails) pour afficher le modèle JSON approprié pour le type d'objet à saisir.

2. Utilisez un processeur JSON en ligne de commande comme jq pour manipuler le modèle sortant comme nécessaire.

3. Utilisez la commande `encode` (voir détails) pour encoder le JSON manipulé.

4. Utilisez la commande `créer` pour créer un objet à partir du JSON encodé.

Par exemple:

```bash
bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder
```

ou

```bash
bw get template item | jq ".name=\"My Login Item\" | .login=$(bw get template item.login | jq '.use
rname="jdoe" | .password="myp@ssword123"')" | bw encode | bw create item
```

Après sa création réussie, l'objet nouvellement créé sera renvoyé sous forme de JSON.

créer d'autres types d'éléments

La commande de création est par défaut pour créer un élément d'identifiant, mais vous pouvez utiliser un processeur JSON en ligne de commande comme jq pour modifier un attribut `.saisir=` afin de créer d'autres types d'éléments :

Nom	Valeur
Identifiant	`.saisir=1`
Note sécurisée	`.saisir=2`
Carte de paiement	`.saisir=3`
Identité	`.saisir=4`

Par exemple, la commande suivante créera une Note sécurisée :

```Bash
bw get template item | jq '.type = 2 | .secureNote.type = 0 | .notes = "Contents of my Secure Note." | .name = "My Secure Note"' | bw encode | bw create item
```

> **Note**
> Remarquez dans l'exemple ci-dessus que les Notes sécurisées nécessitent un sous-modèle (`.secureNote.type`). Vous pouvez afficher le type d'élément des sous-modèles en utilisant `bw get template` (voir ici pour plus de détails).

créer une pièce jointe

La commande `créer un fichier joint` attache un fichier à un **élément existant**.

Contrairement à d'autres opérations de `création`, vous n'avez pas besoin d'utiliser un processeur JSON ou de `coder` pour créer une pièce jointe. Au lieu de cela, utilisez l'option `--file` pour spécifier le fichier à joindre et l'option `--itemid` pour spécifier l'élément auquel le joindre. Par exemple:

```Bash
bw create attachment --file ./path/to/file --itemid 16b15b89-65b3-4639-ad2a-95052a6d8f66
```

> **Tip**
> Si vous ne connaissez pas l'`id de l'élément` exact que vous souhaitez utiliser, utilisez `bw get item` pour renvoyer l'élément (voir détails), y compris son `id`.

obtenir

La commande `get` récupère un seul objet (`élément`, `nom d'utilisateur`, `mot de passe`, et plus) de votre coffre :

```Bash
bw get (item|username|password|uri|totp|exposed|attachment|folder|collection|organization|org-collection|template|fingerprint) <id> [options]
```

La commande `get` prend un élément `id` ou une chaîne pour son argument. Si vous utilisez une chaîne (par exemple, autre chose qu'un `id` exact), `get` va rechercher dans vos objets de coffre celui qui a une valeur correspondante. Par exemple, la commande suivante renverrait un mot de passe Github:

```Bash
bw get password Github
```

> **Note**
> La commande `get` peut **retourner uniquement un résultat**, donc vous devriez utiliser des termes de recherche spécifiques. Si plusieurs résultats sont trouvés, le CLI renverra une erreur.

obtenir la pièce jointe

La commande `obtenir un fichier joint` télécharge un fichier joint :

```Bash
bw get attachment <filename> --itemid <id>
```

La commande `obtenir pièce jointe` prend un `nom de fichier` et un **id** `exact`. Par défaut, `obtenir la pièce jointe` téléchargera la pièce jointe dans le répertoire de travail actuel. Vous pouvez utiliser l'option `--output` pour spécifier un répertoire de sortie différent, par exemple :

```Bash
bw get attachment photo.png --itemid 99ee88d2-6046-4ea7-92c2-acac464b1412 --output /Users/myaccoun
t/Pictures/
```

> **Note**
> Lors de l'utilisation de `--output`, le chemin **doit** se terminer par une barre oblique (`/`) pour spécifier un répertoire ou un nom de fichier (`/Users/moncompte/Pictures/photo.png`).

obtenir des notes

La commande `obtenir des notes` récupère la note pour tout élément de coffre :

```Bash
bw get notes <id>
```

La commande `obtenir des notes` prend un élément `id` ou une chaîne exacte. Si vous utilisez une chaîne (par exemple, autre chose qu'un `id` exact), `obtenir des notes` va rechercher dans vos objets de coffre celui qui a une valeur correspondante. Par exemple, la commande suivante renverrait une note Github:

```Bash
bw get notes Github
```

obtenir le modèle

La commande `obtenir le modèle` renvoie le formatage JSON attendu pour un objet (`élément`, `élément.champ`, `élément.connexion`, et plus):

```
Bash

bw get template (item|item.field|item.login|item.login.uri|item.card|item.identity|item.securenote|folder|collection|item-collections|org-collection)
```

Bien que vous puissiez utiliser `get template` pour afficher le format sur votre écran, l'utilisation la plus courante consiste à diriger la sortie vers une opération `bw create`, en utilisant un processeur JSON en ligne de commande comme jq et `bw encode` pour manipuler les valeurs récupérées à partir du modèle, par exemple :

```
Bash

bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder
```

ⓘ Note

Tout modèle `élément.xxx` doit être utilisé comme un sous-objet à un modèle `élément`, par exemple:

```
Bash

bw get template item | jq ".name=\"My Login Item\" | .login=$(bw get template item.login | jq '.username="jdoe" | .password="myp@ssword123"')" | bw encode | bw create item
```

éditer

La commande `éditer` modifie un objet (`élément`, `collections d'éléments`, etc.) dans votre coffre :

```
Bash

bw edit (item|item-collections|folder|org-collection) <id> [encodedJson] [options]
```

La commande `éditer` prend un **exact** `id` (l'objet à éditer) et du JSON codé (les modifications à effectuer). Un flux de travail typique pourrait ressembler à quelque chose comme :

1. Utilisez la commande `get` (voir détails) pour afficher l'objet à éditer.
2. Utilisez un processeur JSON en ligne de commande comme jq pour manipuler l'objet sorti comme requis.
3. Utilisez la commande `encode` (voir détails) pour encoder le JSON manipulé.

4. Utilisez la commande `éditer` (y compris l'objet `id`) pour éditer l'objet.

Par exemple, pour éditer le mot de passe d'un élément d'identifiant :

```Bash
bw get item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328 | jq '.login.password="newp@ssw0rd"' | bw encode | bw edit item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328
```

Ou, pour éditer la collection dans laquelle un élément se trouve :

```Bash
echo '["5c926f4f-de9c-449b-8d5f-aec1011c48f6"]' | bw encode | bw edit item-collections 28399a57-73a0-45a3-80f8-aec1011c48f6 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

Ou, pour éditer une collection:

```Bash
bw get org-collection ee9f9dc2-ec29-4b7f-9afb-aac8010631a1 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5 | jq '.name="My Collection"' | bw encode | bw edit org-collection ee9f9dc2-ec29-4b7f-9afb-aac8010631a1 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

La commande `éditer` effectuera une opération de **remplacement** sur l'objet. Une fois terminé, l'objet mis à jour sera renvoyé sous forme de JSON.

liste

La commande `liste` récupère un tableau d'objets (`éléments`, `dossiers`, `collections`, et plus) de votre coffre :

```Bash
bw list (items|folders|collections|organizations|org-collections|org-members) [options]
```

Les options pour la commande `list` sont des **filtres** utilisés pour dicter ce qui sera renvoyé, y compris `--url` , `--folderid` , `--collectionid` , `--organizationid` et `--trash`. Tout filtre acceptera `null` ou `notnull`. Combiner plusieurs filtres en une seule commande effectuera une opération OU, par exemple :

```Bash
bw list items --folderid null --collectionid null
```

Cette commande renverra des éléments qui ne sont pas dans un dossier ou une collection.

De plus, vous pouvez **rechercher** des objets spécifiques en utilisant `--recherche` . Combiner le filtre et la recherche en une seule commande effectuera une opération ET, par exemple :

```Bash
bw list items --search github --folderid 9742101e-68b8-4a07-b5b1-9578b5f88e6f
```

Cette commande va rechercher des éléments avec la chaîne `github` dans le dossier spécifié.

supprimer

La commande `supprimer` supprime un objet de votre coffre. `supprimer` prend **seulement un exact** `id` pour son argument.

```Bash
bw delete (item|attachment|folder|org-collection) <id> [options]
```

Par défaut, `supprimer` va envoyer un élément à la Corbeille, où il restera pendant 30 jours. Vous pouvez supprimer définitivement un élément en utilisant l'option `-p, --permanent`.

```Bash
bw delete item 7063feab-4b10-472e-b64c-785e2b870b92 --permanent
```

Pour supprimer une `collection-org`, vous devrez également spécifier `--organizationid` . Voir IDs d'organisation.

> ⚠ **Warning**
>
> Bien que les éléments qui sont supprimés en utilisant `supprimer` peuvent être récupérés en utilisant la commande `restaurer` pendant jusqu'à 30 jours (voir détails), les éléments qui sont supprimés en utilisant `supprimer --permanent` **sont complètement supprimés et irrécupérables.**

restaurer

La commande `restaurer` restaure un objet supprimé de votre corbeille. `restaurer` prend **seulement un** `id` exact pour son argument.

```Bash
bw restore (item) <id> [options]
```

Par exemple:

```Bash
bw restore item 7063feab-4b10-472e-b64c-785e2b870b92
```

envoyer

La commande `envoyer` crée un objet Bitwarden Send pour le partage éphémère. Cette section détaillera des opérations simples d'`envoyer`, cependant envoyer est un outil très flexible et nous recommandons de se référer à l'article dédié sur Send from CLI.

Pour créer un simple texte Send :

```Bash
bw send -n "My First Send" -d 7 --hidden "The contents of my first text Send."
```

Pour créer un simple fichier Send :

```Bash
bw send -n "A Sensitive File" -d 14 -f /Users/my_account/Documents/sensitive_file.pdf
```

recevoir

La commande `recevoir` accède à un objet Bitwarden Send. Pour recevoir un objet Send :

```Bash
bw receive --password passwordforaccess https://vault.bitwarden.com/#/send/yawoill8rk6VM6zCATXv2A/9WN8wD-hzsDJjfnXLeNc2Q
```

Commandes de l'organisation

Identifiants de l'organisation

L'accès à une organisation depuis le CLI nécessite la connaissance d'un ID pour votre organisation, ainsi que des ID pour les membres individuels et les collections.

Récupérez ces informations directement depuis le CLI en utilisant `bw list`, par exemple :

```Bash
bw list organizations
bw list org-members --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
bw list org-collections --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

> **Tip**
> Vous pouvez `bw list` à la fois les `collections` et les `org-collections`. La commande `bw list collections` listera toutes les collections, indépendamment de l'organisation à laquelle elles appartiennent. `bw list org-collections` listera uniquement les collections qui appartiennent à l'organisation spécifiée en utilisant `--organizationid`.

bouger

> **Note**
> **Août 2021** : La commande `partager` a été changée en `déplacer`. En savoir plus .

La commande `déplacer` transfère un élément de coffre à une organisation:

```Bash
bw move <itemid> <organizationid> [encodedJson]
```

La commande `déplacer` vous oblige à `encoder` un ID de collection, et prend un **id** `exact` (l'objet à partager) et un **organizationid** `exact` (l'organisation à laquelle partager l'objet). Par exemple:

```Bash
echo '["bq209461-4129-4b8d-b760-acd401474va2"]' | bw encode | bw move ed42f44c-f81f-48de-a123-ad010
13132ca dfghbc921-04eb-43a7-84b1-ac74013bqb2e
```

Une fois terminé, l'élément mis à jour sera renvoyé.

confirmer

La commande `confirmer` confirme les membres invités à votre organisation qui ont accepté leur invitation:

```Bash
bw confirm org-member <id> --organizationid <orgid>
```

La commande `confirmer` prend un **exact** identifiant de membre `id` et un **exact** `organizationID`, par exemple:

```Bash
bw confirm org-member 7063feab-4b10-472e-b64c-785e2b870b92 --organizationid 310d5ffd-e9a2-4451-af87
-ea054dce0f78
```

Autres commandes

configuration

La commande `config` spécifie les paramètres que le Bitwarden CLI doit utiliser :

```
Bash

bw config server <setting> [value]
```

Une utilisation principale de `bw config` est de connecter votre CLI à un serveur Bitwarden auto-hébergé :

```
Bash

bw config server https://your.bw.domain.com
```

Tip

Connectez-vous au serveur EU de Bitwarden en exécutant la commande suivante:

```
Bash

bw config server https://vault.bitwarden.eu
```

Passez `bw config server` sans valeur pour lire le serveur auquel vous êtes connecté.

Les utilisateurs avec des configurations uniques peuvent choisir de spécifier l'URL de chaque service indépendamment. Notez que toute utilisation ultérieure de la commande de configuration écrasera toutes les spécifications précédentes, donc cela doit être exécuté comme une seule commande à chaque fois que vous effectuez un changement :

```
Bash

bw config server --web-vault <url> \
  --api <url> \
  --identity <url> \
  --icons <url> \
  --notifications <url> \
  --events <url> \
  --key-connector <url>
```

> **Note**
> La commande `bw config server --key-connector` est nécessaire si votre organisation utilise Key Connector et que vous utilisez l'option `--apikey` pour vous connecter après avoir supprimé votre mot de passe principal.
>
> Contactez le propriétaire d'une organisation pour obtenir l'URL requise.

synchroniser

La commande `synchroniser` télécharge votre coffre crypté depuis le serveur Bitwarden. Cette commande est la plus utile lorsque vous avez modifié quelque chose dans votre coffre Bitwarden sur une autre application client (par exemple le coffre web, l'extension de navigateur, l'application mobile) depuis que vous vous êtes connecté sur le CLI.

```Bash
bw sync
```

Vous pouvez passer l'option `--last` pour ne renvoyer que l'horodatage (ISO 8601) de la dernière fois qu'une synchronisation a été effectuée.

> **Tip**
> Il est important de savoir que `synchroniser` **n'effectue qu'une extraction** depuis le serveur. Les données sont automatiquement envoyées au serveur chaque fois que vous apportez un changement à votre coffre (par exemple, `créer`, `éditer`, `supprimer`).

coder

La commande `encode` encode en Base 64 stdin. Cette commande est généralement utilisée en combinaison avec un processeur JSON en ligne de commande comme jq lors de l'exécution des opérations de `création` et d'`édition`, par exemple :

```Bash
bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder

bw get item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328 | jq '.login.password="newp@ssw0rd"' | bw encode |
bw edit item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328
```

importer

La commande `importer` importe des données à partir d'une exportation Bitwarden ou d'une autre application de gestion de mots de passe prise en charge. La commande doit être dirigée vers un fichier et inclure les arguments suivants :

```Bash
bw import <format> <path>
```

Par exemple:

```Bash
bw import lastpasscsv /Users/myaccount/Documents/mydata.csv
```

> **Tip**
>
> Bitwarden prend en charge de nombreux formats pour l'importer, trop nombreux pour être énumérés ici! Utilisez `bw importer --formats` pour renvoyer la liste dans votre CLI, ou voir ici.
>
> Si vous êtes en train d'importer un fichier .json crypté que vous avez créé avec un mot de passe, on vous demandera d'entrer le mot de passe avant que l'importation ne soit terminée.

exporter

La commande `exporter` exporte les données du coffre en tant que fichier `.json` ou `.csv`, ou fichier .json crypté :

```Bash
bw export [--output <filePath>] [--format <format>] [--password <password>] [--organizationid <orgi
d>]
```

Par défaut, la commande `exporter` va générer un `.csv` (équivalent à spécifier `--format csv`) dans le répertoire de travail actuel, cependant vous pouvez spécifier :

- `--format json` pour exporter un fichier `.json`
- `--format encrypted_json` pour exporter un fichier .json crypté
  - `--mot de passe` pour spécifier un mot de passe à utiliser pour chiffrer les `exportations_json_chiffrées` au lieu de votre clé de chiffrement de compte
- `--output` pour exporter vers un emplacement spécifique
- `--raw` pour renvoyer l'exportation vers la sortie standard plutôt que vers un fichier

exporter depuis un coffre d'organisation

En utilisant la commande `exporter` avec l'option `--organizationid`, vous pouvez exporter un coffre d'organisation :

```Bash
bw export --organizationid 7063feab-4b10-472e-b64c-785e2b870b92 --format json --output /Users/myacc
ount/Downloads/
```

générer

La commande `générer` génère un mot de passe fort ou une phrase secrète :

```
Bash

bw generate [--lowercase --uppercase --number --special --length <length> --passphrase --separator <separator> --words <words>]
```

Par défaut, la commande `générer` va générer un mot de passe de 14 caractères avec des caractères majuscules, des caractères minuscules, et des nombres. C'est l'équivalent de réussir :

```
Bash

bw generate -uln --length 14
```

Vous pouvez générer des mots de passe plus complexes en utilisant les options disponibles pour la commande, y compris :

- `--majuscule`, `-m` (inclure les majuscules)
- `--minuscule`, `-l` (inclure minuscule)
- `--nombre`, `-n` (inclure les nombres)
- `--spécial`, `-s` (inclure des caractères spéciaux)
- `--longueur` (longueur du mot de passe, minimum de 5)

générer une phrase secrète

En utilisant la commande `générer` avec l'option `--passphrase`, vous pouvez générer une phrase secrète au lieu d'un mot de passe :

```
Bash

bw generate --passphrase --words <words> --separator <separator>
```

Par défaut, `bw générer --passphrase` générera une phrase secrète de trois mots séparés par un tiret (`-`). C'est l'équivalent de réussir :

```
Bash

bw generate --passphrase --words 3 --separator -
```

Vous pouvez générer une phrase secrète complexe en utilisant les options disponibles pour la commande, y compris :

- `--mots` (nombre de mots)
- `--séparateur` (caractère de séparation)
- `--majuscule`, `-c` (inclure pour mettre la phrase secrète en majuscule)

- `--includeNumber` (inclure des nombres dans la phrase secrète)

mettre à jour

La commande `mettre à jour` vérifie si votre Bitwarden CLI exécute la version la plus récente. `mettre à jour` **ne met pas automatiquement à jour le CLI pour vous.**

```Bash
bw update
```

Si une nouvelle version est détectée, vous devrez télécharger la nouvelle version du CLI en utilisant l'URL imprimée pour l'exécutable, ou en utilisant les outils disponibles pour le gestionnaire de paquets que vous avez utilisé pour télécharger le CLI (par exemple, `npm install -g @bitwarden/cli`).

statut

La commande `status` renvoie des informations sur le statut du CLI Bitwarden, y compris l'URL du serveur configuré, l'horodatage de la dernière synchronisation (ISO 8601), le courriel et l'ID de l'utilisateur, et le statut du coffre.

```Bash
bw status
```

Le statut renverra des informations sous forme d'objet JSON, par exemple :

```Bash
{
  "serverUrl": "https://bitwarden.example.com",
  "lastSync": "2020-06-16T06:33:51.419Z",
  "userEmail": "user@example.com",
  "userId": "00000000-0000-0000-0000-000000000000",
  "status": "unlocked"
}
```

`statut` peut être l'un des suivants:

- `"unlocked"` , indiquant que vous êtes connecté et que votre coffre-fort est déverrouillé (une variable d'environnement de clé `BW_SESSION` est enregistrée avec une clé de session active)
- `"verrouillé"` , indiquant que vous êtes connecté mais que votre coffre-fort est verrouillé (**aucune** variable d'environnement de clé `BW_SESSION` n'est enregistrée avec une clé de session active)
- `"non authentifié"` , indiquant que vous n'êtes pas connecté

> **Tip**
> Quand `"status": "non authentifié"`, `dernièreSynchronisation`, `emailUtilisateur`, et `IDutilisateur` retourneront toujours `null`.

servir

La commande `serve` démarre un serveur web express local qui peut être utilisé pour effectuer toutes les actions accessibles depuis le CLI sous forme d'appels API RESTful depuis une interface HTTP.

```Bash
bw serve --port <port> --hostname <hostname>
```

Par défaut, `serve` démarrera le serveur web sur le port 8087, cependant, vous pouvez spécifier un port alternatif avec l'option `--port`.

Par défaut, `serve` liera votre serveur web API à `localhost` cependant, vous pouvez spécifier un nom d'hôte alternatif avec l'option `--hostname`. Les demandes d'API ne peuvent être faites que depuis le nom d'hôte lié.

Par défaut, `serve` bloquera toute demande avec un en-tête `Origin`. Vous pouvez contourner cette protection en utilisant l'option `--disable-origin-protection`, cependant **ce n'est pas recommandé**.

> **⚠ Warning**
> Vous pouvez spécifier `--hostname all` pour aucune liaison de nom d'hôte, cependant cela permettra à n'importe quelle machine sur le réseau de faire des demandes d'API.

Afficher les spécifications de l'API pour obtenir de l'aide pour passer des appels avec `serve`.

Annexes

Options globales

Les options suivantes sont disponibles à l'échelle mondiale :

Option	Description
`--joli`	Format de sortie. JSON est ongleté avec deux espaces.
`--brut`	Renvoyer une sortie brute au lieu d'un message descriptif.
`--réponse`	Retournez une version de la réponse de sortie formatée en JSON.

Option	Description
`--silencieux`	Ne renvoyez rien à stdout. Vous pourriez utiliser cette option, par exemple, lorsque vous acheminez une valeur d'identification vers un fichier ou une application.
`--nointeraction`	Ne pas demander d'entrée utilisateur interactive.
`--session`	Passez la clé de session au lieu de la lire à partir d'une variable d'environnement.
`-v, --version`	Affichez le nombre de version du CLI Bitwarden.
`-h, --aide`	Affichez le texte d'aide pour la commande.

Complétion de shell ZSH

Le CLI de Bitwarden comprend le support pour l'achèvement de la coquille ZSH. Pour configurer l'achèvement de la coquille, utilisez l'une des méthodes suivantes :

1. **Vanilla ZSH:** Ajoutez la ligne suivante à votre fichier `.zshrc` :

```bash
eval "$(bw completion --shell zsh); compdef _bw bw;"
```

2. **Vanilla (complétions du vendeur) :** Exécutez la commande suivante :

```bash
bw completion --shell zsh | sudo tee /usr/share/zsh/vendor-completions/_bw
```

3. zinit: Exécutez les commandes suivantes:

```bash
bw completion --shell zsh > ~/.local/share/zsh/completions/_bw
zinit creinstall ~/.local/share/zsh/completions
```

Utilisation de certificats auto-signés

Si votre serveur Bitwarden auto-hébergé expose un certificat TLS auto-signé, spécifiez la variable d'environnement Node.js NODE_EXTRA_CA_CERTS :

Bash

```bash
export NODE_EXTRA_CA_CERTS="absolute/path/to/your/certificates.pem"
```

PowerShell

```bash
$env:NODE_EXTRA_CA_CERTS="absolute/path/to/your/certificates.pem"
```

Énumérations

Les tableaux suivants énumèrent les valeurs requises dans les scénarios documentés :

Méthodes d'identification en deux étapes

Utilisé pour spécifier quelle méthode d'identifiant en deux étapes utiliser lors de la connexion :

Nom	Valeur
Authentificateur	0
Courriel	1
YubiKey	3

> **Note**
> FIDO2 et Duo ne sont pas pris en charge par le CLI.

Types d'éléments

Utilisé avec la commande `créer` pour spécifier un type d'élément de coffre :

Nom	Valeur
Identifiant	1
Note sécurisée	2
Carte de paiement	3
Identité	4

Types de correspondance d'identifiant URI

Utilisé avec la commande `créer` et `éditer` pour spécifier le comportement de détection de correspondance URI pour un élément d'identifiant :

Nom	Valeur
Domaine	0
Hôte	1
Commence Avec	2
Exact	3

Nom	Valeur
Expression régulière	4
Jamais	5

Types de champs

Utilisé avec les commandes `créer` et `éditer` pour configurer des champs personnalisés :

Nom	Valeur
Texte	0
Masqué	1
Booléen	2

Types d'utilisateurs de l'organisation

Indique le type de l'utilisateur :

Nom	Valeur
Propriétaire	0
Administrateur	1
Utilisateur	2
Gestionnaire	3

Nom	Valeur
Personnalisé	4

Statuts des utilisateurs de l'organisation

Indique le statut de l'utilisateur au sein de l'organisation :

Nom	Valeur
Invité	0
Accepté	1
Confirmé	2
Révoqué	-1