キーコネクターをデプロイする

キーコネクターをデプロイする

この記事では、既存の自己ホスト型環境でキーコネクターを有効にし、設定する手順を説明します。**続行する前に**、キーコネクタに関する記事をよく読んで、キー コネクタとは何か、その仕組み、および実装の影響を完全に理解してください。

Bitwardenは、自己ホスト型インスタンスのための1つの組織による1つのキーコネクターのデプロイメントをサポートします。

要件

> ⚠ **Warning**
>
> Management of cryptographic keys is incredibly sensitive and is **only recommended for enterprises with a team and infrastructure** that can securely support deploying and managing a key server.

キーコネクターを使用するためには、以下の手順を守る必要があります:

- エンタープライズ組織を持っている。
- 自己ホスト型の Bitwarden サーバーを用意します。
- アクティブな SSO 実装がある。
- [単一組織] ポリシーと [シングル サインオンを要求する] ポリシーをアクティブ化します。

あなたの組織がこれらの要件を満たすか、または満たすことができ、
キーサーバーの管理をサポートできるチームとインフラストラクチャを含む場合、私たちに連絡してください。そうすれば、
私たちはキーコネクターを有効にします。

キーコネクターの設定とデプロイ

キーコネクターについて私たちに連絡を取ったら、キーコネクターの議論を始めるために私たちは連絡を取ります。この記事に続く手順は、Bitwardenのカスタマーサクセス＆実装スペシャリストと協力して完了させる必要があります。

新しいライセンスファイルを取得する

キーコネクターについて私たちに連絡を取った後、カスタマーサクセス＆実装チームのメンバーが、
あなたの組織のためのキーコネクター対応ライセンスファイルを生成します。あなたのBitwardenの共同作業者が準備ができたと指示したら、
新しいライセンスを取得するための次の手順を完了してください：

1. Bitwardenクラウドウェブアプリを開き、管理者コンソール内のあなたの組織の**請求書** → **サブスクリプション**画面に移動します。
2. 下にスクロールして、**ライセンスをダウンロード**ボタンを選択してください。
3. 指示に従って、自己ホスト型サーバーのインストールに使用したインストールIDを入力し、**送信**を選択してください。
   あなたがインストールIDをすぐに思い出せない場合は、`./bwdata/env/global.override.env`から取得することができます。

ライセンス ファイルはすぐには必要ありませんが、後の手順でセルフホスト サーバーにアップロードする必要があります。

キーコネクターを初期化します

あなたのBitwardenサーバーをキーコネクターのために準備するには：

1. 少なくとも.bwdata/mssqlのバックアップを保存してください。キーコネクターが使用中の場合、問題が発生した場合に備えて、
   キーコネクター使用前のバックアップイメージにアクセスできることをお勧めします。

> **Note**
> あなたが外部のMSSQLデータベースを使用している場合、
> あなたの実装に最も適した方法でデータベースのバックアップを取ってください。

2. 最新の変更を取得するために、自己ホスト型のBitwardenインストールを更新してください。

```
Bash

./bitwarden.sh update
```

3. `.bwdata/config.yml` ファイルを編集し、`enable_key_connector` を `true` に切り替えてキーコネクターを有効にしてください。

```
Bash

nano bwdata/config.yml
```

4. あなたの自己ホスト型Bitwardenインストールを再構築します：

```
Bash

./bitwarden.sh rebuild
```

5. 変更を適用するために、自己ホスト型のBitwardenインストールを再度更新してください。

```
Bash

./bitwarden.sh update
```

キーコネクターを設定する

キーコネクターを設定するには：

1. `.bwdata/env/key-connector.override.env` ファイルを編集してください。このファイルは `./bitwarden.sh 更新` でダウンロードされるはずです。

```
Bash

nano bwdata/env/key-connector.override.env
```

> **⚠ Warning**
> This file will be pre-populated with default values that will spin up a functional local Key Connector setup, however the **default values are not recommended for production environments**.

2. key-connector.overrideで、次の項目に値を指定する必要があります：

- エンドポイント: キーコネクターが通信できるBitwardenのエンドポイント。
- データベース：キーコネクターがユーザーキーを保存し、取得する場所。
- RSAキーペア: キーコネクターがユーザーキーを保護するためにRSAキーペアにどのようにアクセスするか。

エンドポイント

自動設定は、インストール設定に基づいてエンドポイントの値を自動的に設定しますが、key-connector.override.env の以下の値があなたの設定に適していることを確認することをお勧めします:

```bash
keyConnectorSettings__webVaultUri=https://your.bitwarden.domain.com
keyConnectorSettings__identityServerUri=http://identity:5000
```

データベース

キーコネクターは、組織のメンバーのための暗号化されたユーザーキーを保存するデータベースにアクセスする必要があります。暗号化されたユーザーキーを保存するための安全なデータベースを作成し、デフォルトのkeyConnectorSettings__database__の値をkey-connector.override.envで指定された値に置き換えてください。選択したデータベースの**必要な値**列に記載されています。

> ⚠ **Warning**
> Migration from one database to another is **not supported** at this time. Regardless of which provider you choose, **implement a frequent automated backup schedule** for the database.

データベース	必要な値
ローカルJSON（**デフォルト**）	**テスト以外では推奨されません。** keyConnectorSettings__database__provider=json keyConnectorSettings__database__jsonFilePath={File_Path}
Microsoft SQLサーバー	keyConnectorSettings__database__provider=sqlserver keyConnectorSettings__database__sqlServerConnectionString={Connection_String} MSSQL接続文字列のフォーマット方法を学びましょう

データベース	必要な値

PostgreSQL	`keyConnectorSettings__database__provider=postgresql`
	`keyConnectorSettings__database__postgreSqlConnectionString={Connection_String}`
	PostgreSQL接続文字列のフォーマット方法を学びましょう

MySQL/MariaDB	`keyConnectorSettings__database__provider=mysql`
	`keyConnectorSettings__database__mySqlConnectionString={Connection_String}`
	MySQL接続文字列のフォーマット方法を学びましょう

MongoDB	`keyConnectorSettings__database__provider=mongo`
	`keyConnectorSettings__database__mongoConnectionString={Connection_String}`
	`keyConnectorSettings__database__mongoDatabaseName={DatabaseName}`
	MongoDB接続文字列のフォーマット方法を学びましょう

RSAキーペア

キーコネクターはRSAキーペアを使用して、ユーザーキーを保護します。キーペアを作成し、デフォルトの`keyConnectorSettings__rsaKey__`と`keyConnectorSettings__certificate__`の値を`key-connector.override.env`で選択した実装に必要な値に置き換えてください。

> **Tip**
> The RSA key pair must be **at a minimum** 2048 bits in length.

一般的に、選択肢としては、キーペアを含むX509 **証明書**へのキーコネクターへのアクセスを許可するか、直接**キーペア**へのキーコネクターへのアクセスを許可するかがあります。

⇒証明書

RSAキーペアを含むX509証明書を使用するには、証明書が保存されている場所に応じて必要な値を指定します（**ファイルシステム**、**OS証明書ストア**などを参照してください）：

Tip

The certificate **must** be made available as a PKCS12 `.pfx` file, for example:

```
Bash

openssl req -x509 -newkey rsa:4096 -sha256 -nodes -keyout bwkc.key -out bwkc.crt -subj "/CN=Bi
twarden Key Connector" -days 36500

openssl pkcs12 -export -out ./bwkc.pfx -inkey bwkc.key -in bwkc.crt -passout pass:{Password}
```

In all certificate implementations, you'll need the `CN` value shown in this example.

ファイルシステム（デフォルト）

証明書がキーコネクターを実行しているマシンのファイルシステムに保存されている場合、次の値を指定してください：

Note

By default, Key Connector will be configured to create a `.pfx` file located at `etc/bitwarden/key-connector/bwkc.pfx` with a generated password. **It is not recommended** for enterprise implementations to use these defaults.

```
Bash

keyConnectorSettings__rsaKey__provider=certificate
keyConnectorSettings__certificate__provider=filesystem
keyConnectorSettings__certificate__filesystemPath={Certificate_Path}
keyConnectorSettings__certificate__filesystemPassword={Certificate_Password}
```

アジュール ブロブ ストレージ

証明書がAzure Blob Storageにアップロードされる場合、次の値を指定してください：

```
Bash

keyConnectorSettings__rsaKey__provider=certificate
keyConnectorSettings__certificate__provider=azurestorage
keyConnectorSettings__certificate__azureStorageConnectionString={Connection_String}
keyConnectorSettings__certificate__azureStorageContainer={Container_Name}
keyConnectorSettings__certificate__azureStorageFileName={File_Name}
keyConnectorSettings__certificate__azureStorageFilePassword={File_Password}
```

`azureStorageConnectionString`を、Azureポータルで**共有アクセス署名**(SAS)のページからあなたのストレージアカウントで生成できる**接続文字列**に設定します。SASは以下を持つ必要があります：

- 許可されるサービス：BlobとFile
- 許可されたリソースタイプ：サービス、コンテナ、オブジェクト
- 許可された権限：読み取り、書き込み、リスト
- 許可されたblobインデックスの権限：読み取り/書き込みとフィルター

Azure キー保管庫

証明書がAzure Key Vaultに保存されている場合、以下の値を指定してください：

> **Note**
>
> To use Azure Key Vault to store your `.pfx` certificate, you'll need to create an Active Directory **App Registration**. This App Registration must:
>
> - Give delegated API permissions to access Azure Key Vault
> - Have a client secret generated to allow access by Key Connector

```bash
keyConnectorSettings__certificate__provider=azurekv
keyConnectorSettings__certificate__azureKeyvaultUri={Vault_URI}
keyConnectorSettings__certificate__azureKeyvaultCertificateName={Certificate_Name}
keyConnectorSettings__certificate__azureKeyvaultAdTenantId={ActiveDirectory_TenantId}
keyConnectorSettings__certificate__azureKeyvaultAdAppId={AppRegistration_ApplicationId}
keyConnectorSettings__certificate__azureKeyvaultAdSecret={AppRegistration_ClientSecretValue}
```

Hashicorp 保管庫

証明書がHashicorp保管庫に保存されている場合、次の値を指定してください：

> **Note**
>
> Key Connector integrates with the Hashicorp Vault KV2 Storage Engine. As per the top of this tab, the certificate file should be in PKCS12 format and stored base64-encoded as the value to a named key in your Vault. If following a Vault tutorial for the KV2 Storage Engine, the key name may be `file` unless otherwise specified.

```
Bash

keyConnectorSettings__rsaKey__provider=certificate
keyConnectorSettings__certificate__provider=vault
keyConnectorSettings__certificate__vaultServerUri={Server_URI}
keyConnectorSettings__certificate__vaultToken={Token}
keyConnectorSettings__certificate__vaultSecretMountPoint={Secret_MountPoint}
keyConnectorSettings__certificate__vaultSecretPath={Secret_Path}
keyConnectorSettings__certificate__vaultSecretDataKey={Secret_DataKey}
keyConnectorSettings__certificate__vaultSecretFilePassword={Secret_FilePassword}
```

⇒キーペア

RSA 2048キーペアを保存するためにクラウドプロバイダーまたは物理的なデバイスを使用するには、
選択した実装に応じて必要な値を指定します（**Azure Key 保管庫**、**Google Cloud Key 管理**などを参照してください）：

Azure キー保管庫

RSA 2048キーペアを保存するためにAzure Key Vaultを使用している場合、次の値を指定してください：

> **Note**
>
> To use Azure Key Vault to store your RSA 2048 key, you'll need to create an Active Directory **App Registration**. This App Registration must:
>
> - Give delegated API permissions to access Azure Key Vault
> - Have a client secret generated to allow access by Key Connector

```
Bash

keyConnectorSettings__rsaKey__provider=azurekv
keyConnectorSettings__rsaKey__azureKeyvaultUri={Vault_URI}
keyConnectorSettings__rsaKey__azureKeyvaultKeyName={Key_Name}
keyConnectorSettings__rsaKey__azureKeyvaultAdTenantId={ActiveDirectory_TenantId}
keyConnectorSettings__rsaKey__azureKeyvaultAdAppId={AppRegistration_ApplicationId}
keyConnectorSettings__rsaKey__azureKeyvaultAdSecret={AppRegistration_ClientSecretValue}
```

Azure Key Vaultを使用してキーペアを作成する方法を学びましょう

Google Cloudキー管理

あなたがGoogle Cloud Key Managementを使用してRSA 2048キーペアを保存している場合、次の値を指定してください：

```Bash
keyConnectorSettings__rsaKey__provider=gcpkms
keyConnectorSettings__rsaKey__googleCloudProjectId={Project_Id}
keyConnectorSettings__rsaKey__googleCloudLocationId={Location_Id}
keyConnectorSettings__rsaKey__googleCloudKeyringId={Keyring_Id}
keyConnectorSettings__rsaKey__googleCloudKeyId={Key_Id}
keyConnectorSettings__rsaKey__googleCloudKeyVersionId={KeyVersionId}
```

Google Cloud Key Management Serviceを使用してキーリングと非対称キーを作成する方法を学びましょう

AWSキーマネージメントサービス

あなたがAWSキー管理サービス（KMS）を使用してRSA 2048キーペアを保存する場合、次の値を指定してください：

```Bash
keyConnectorSettings__rsaKey__provider=awskms
keyConnectorSettings__rsaKey__awsAccessKeyId={AccessKey_Id}
keyConnectorSettings__rsaKey__awsAccessKeySecret={AccessKey_Secret}
keyConnectorSettings__rsaKey__awsRegion={Region_Name}
keyConnectorSettings__rsaKey__awsKeyId={Key_Id}
```

AWS KMSを使用して非対称キーを作成する方法を学びましょう

PKCS11 物理的な HSM

PKCS11プロバイダーと物理的なHSMデバイスを使用している場合、次の値を指定してください:

```
Bash

keyConnectorSettings__rsaKey__provider=pkcs11
keyConnectorSettings__rsaKey__pkcs11Provider={Provider}
keyConnectorSettings__rsaKey__pkcs11SlotTokenSerialNumber={Token_SerialNumber}
keyConnectorSettings__rsaKey__pkcs11LoginUserType={Login_UserType}
keyConnectorSettings__rsaKey__pkcs11LoginPin={Login_PIN}

ONE OF THE FOLLOWING TWO:
keyConnectorSettings__rsaKey__pkcs11PrivateKeyLabel={PrivateKeyLabel}
keyConnectorSettings__rsaKey__pkcs11PrivateKeyId={PrivateKeyId}

OPTIONALLY:
keyConnectorSettings__rsaKey__pkcsLibraryPath={path/to/library/file}
```

どこで

- {Provider}はyubihsmまたはopenscになることができます。
- {Login_UserType}はユーザー、だから、またはコンテキストに依存することができます

> **Note**
> If you are using the PKCS11 provider to store your private key on an HSM device, the associated public key must be made available and configured as a certificate using any of the options found in the **Certificates** tab.

キーコネクターを有効にする

キーコネクターが完全に設定され、キーコネクター対応ライセンスを取得したので、次の手順を完了してください:

1. 設定の変更を適用するために、自己ホスト型のBitwardenインストールを再起動してください。

   ```
   Bash

   ./bitwarden.sh restart
   ```

2. セルフホスト型 Bitwarden に組織**所有者**としてログインし、管理コンソールの**[請求]** → **[サブスクリプション]**画面に移動します。
3. **ライセンスを更新**ボタンを選択し、キーコネクター対応のライセンスをアップロードします。これは前のステップで取得したものです。
4. まだ行っていない場合は、**設定→ポリシー**画面に移動し、単一組織と単一サインオン認証が必要のポリシーを有効にしてください。**両方ともキーコネクターを使用する必要があります。**
5. **設定→ シングルサインオン**画面に移動します。

> **Tip**
>
> The next few steps assume that you already have an active login with SSO implementation using SAML 2.0 or OIDC. **If you don't**, please implement and test login with SSO before proceeding.

6. **メンバー復号化オプション**セクションで、**キーコネクター**を選択します。

7. **キーコネクターURL**の入力欄に、キーコネクターが動作しているアドレス（デフォルトでは、`https://your.domain/key-connector`）を入力し、**テスト**ボタンを選択してキーコネクターにアクセスできることを確認してください。

8. 画面の下までスクロールし、**保存**を選択してください。