One Identity Safeguard Secrets Vault 6.12

User Guide

Contents

Introduction to the Secrets Broker Vault

The Secrets Broker Vault add-on converts the open source Secrets Broker product into a Secrets Broker Vault which includes a built-in vault and is capable of storing and forwarding credentials using all of the Hashicorp Vault commandline tools. It will also support all of the Hashicorp REST APIs. For information on the open source Secrets Broker product plus initial installation and configuration information required for Secrets Broker Vault, see Secrets Broker open source project site.

Prerequisites

While deploying the Secrets Broker Service does not require any prior configuration to the Safeguard for Privileged Passwords Appliance, deploying the Secrets Broker Vault Add-on requires some preconfiguration. The following outlines the configuration that must be on the Secrets Broker Service as well as the corresponding Safeguard for Privileged Passwords appliance.

The following are required prior to installing the Secrets Broker Vault:

1. Download and install the Safeguard Secrets Broker for DevOps 6.12 or higher from the Secrets Broker open source project site.

2. Follow all of the Secrets Broker installation and configuration instructions on the Secrets Broker open source project site.

3. Configure the Secrets Broker Service using the web interface:

   - Specify the Safeguard for Privileged Passwords appliance that will be connected to the Secrets Broker Service.

   - Create a new client certificate from a CSR or upload a new client certificate.

   - Import the trusted certificates from the connected Safeguard for Privileged Passwords appliance.

4. Purchase and install the Secrets Broker Vault add-on license. Since the Secrets Broker Vault Add-on is an add-on product, it requires an additional license that must be added to the Safeguard for Privileged Passwords appliance that the Secrets Broker Vault has been connected to.

Deploying the Secrets Broker Vault

Once the Prerequisites have been completed, the Secrets Broker Vault Add-on is ready to be installed on the Secrets Broker Service. By uploading the Secrets Broker Vault Add-on to the Secrets Broker Service, the add-on will automatically convert the Secrets Broker Service into a Secrets Broker Vault.

Deploying the add-on module does not disable any of the existing Secrets Broker Service functionality. The Secrets Broker Vault Add-on is completely additive. Deploying the add-on takes all of the current Secrets Broker functionality and adds an embedded vault that is capable of storing credentials that have been pushed from the Safeguard for Privileged Passwords appliance. These credentials can then be accessed using the existing Hashicorp Vault command line tools. The embedded vault can also be configured with additional functionality in the same way that any other Hashicorp Vault can be modified, such as new secrets engines and authentication methods.

To deploy the Secrets Broker Vault add-on

1. As stated in the Prerequisites, follow all installation and configuration instructions on the Secrets Broker open source project site.

2. Once the open source Secrets Broker service has been installed and configured, use the **Upload** button in the **Add-ons** section to upload the `.sbao` file you received from One Identity upon purchasing the add-on. Secrets Broker Vault will validate the license and validate that the add-on `.sbao` file is valid. If a valid Secrets Broker Vault Add-on license has not been installed on the connected Safeguard for Privileged Passwords, the upload button will not be available

   The Secrets Broker Vault add-on `.sbao` file will automatically convert the open source Secrets Broker service into a Secrets Broker Vault which includes a built-in vault and is capable of storing and forwarding credentials using all of the Hashicorp Vault commandline tools. It will also support all of the Hashicorp REST APIs.

3. Once fully deployed, the Secrets Broker Vault requires that the user open the Secrets Broker Vault settings page and finish the configuration.

   a. Click the **Secrets Broker Vault** button in the **Add-ons** section.

   b. In the **Add-on Settings** dialog, click the **Configuring Add-on** button to complete the configuration.

      Once that is done, the settings page should show that the Secrets Broker Vault add-on is healthy and that it is a valid One Identity add-on.

c. Enter the Secrets Broker Vault Plugin setup page by clicking on the **Manage Accounts** button on the plugin tile.

d. Click on the **Test Configuration** button to test the configuration of the plugin.

At this point the Secrets Broker instance should have been converted to a Secrets Broker Vault with the following in place:

- Open Source Secrets Broker service running.

- Embedded vault running alongside the Secrets Broker service.

    - Configured with a One Identity policy.

    - Configured with a One Identity Key/Value secrets engine.

- Embedded web proxy listening to port 443 and forwarding requests to the Secrets Broker service or the embedded vault.

- Secrets Broker Vault plugin configured to push credentials to the embedded vault.

- The connected Safeguard for Privileged Passwords appliance should have been updated with the following:

    - A new Other type asset that corresponds to the Secrets Broker instance.

    - 5 vault accounts with associated credentials. 1 root account and 4 unseal shards.

    - A new dynamic account group that corresponds to the Secrets Broker instance and contains all of the vault accounts.

Post-deployment configuration and information

Once you have completed Deploying the Secrets Broker Vault, the following configuration is available:

- Installing and using the Vault CLI tool on Windows or Linux clients
- Getting the vault root token from the connected Safeguard for Privileged Passwords appliance
- Enabling and configuring the Username/Password authentication method
- Enabling a new secrets engine in the embedded Secrets Broker Vault
- Pulling credentials from the OneIdentity K/V secrets engine

Installing and using the Vault CLI tool on Windows or Linux clients

The Secrets Broker Vault is fully compatible with the Hashicorp vault CLI and can be further configured and accessed using this CLI. By default, the embedded vault has been configured with the K/V secrets engine and a policy which allows the Secrets Broker service to push account credentials into a OneIdentity secrets store. For additional information about how to install and use the Hashicorp vault CLI, see Vault Commands (CLI).

Connecting to the embedded vault

1. Set the following environment variable:

   VAULT_ADDR=https://<SecretsBroker Address>

2. Only one of the following variables can be selected:

   a. If selected, the following variable can be set to specify the SSL certificate using the method outlined in Vault Commands (CLI):

   VAULT_CACERT=<PEM encoded certificate file>

        b. If selected, the following variable can be set to specify the SSL certificate using the method outlined in Vault Commands (CLI):

            `VAULT_CAPATH=<Directory containing PEM encoded certificates>`

        c. If selected, the following variable will bypass SSL certificate validation entirely:

            `VAULT_SKIP_VERIFY=true`

3. Log in to the embedded Secrets Broker Vault. For more information, see Login.

4. The CLI will prompt for the root token. To connect to the embedded vault using the Hashicorp vault CLI, the root token needs to be fetched from Safeguard for Privileged Passwords. This token was stored in Safeguard for Privileged Passwords during the deployment of the Secrets Broker Vault Add-on. For more information, see Getting the vault root token from the connected Safeguard for Privileged Passwords appliance.

Getting the vault root token from the connected Safeguard for Privileged Passwords appliance

During the deployment of the Secrets Broker Vault Add-on, the add-on automatically deploys and configures an embedded vault which is capable of storing specific account information and credentials that are pushed from the connected Safeguard for Privileged Passwords appliance. During the configuration of the embedded vault, the root token and unseal shards are automatically added to the Safeguard for Privileged Passwords appliance as new accounts and an account group is created that contains these accounts. The account group can be used to create an access policy and subsequent access request to retrieve the root token so that the embedded vault can be further configured by an administrator.

NOTE: For more information on the Safeguard for Privileged Passwords settings and pages mentioned in the following instructions, see the *Safeguard for Privileged Passwords Administration Guide*.

1. Retrieve the embedded vault root token:

        a. Open the Safeguard for Privileged Passwords appliance web interface by going to `https://<spp-address>`.

        b. Navigate to **Security Policy Management** | **Entitlements**.

        c. Create a new entitlement for accessing the Secrets Broker Vault accounts.

        d. On the **Access Request Policies** tab within the new Secrets Broker Vault entitlement, add a new Access Request Policy that includes the following settings:

i. On the **General** tab, ensure the policy type is set to **Credential** and the credential type is set to **Password**.

ii. On the **Security** tab, ensure **Change Password After Check-in** is set to **False**.

> ⚠ **CAUTION: If Change Password After Check-in is not set to False, then the tokens will be lost since they cannot be changed.**

iii. On the **Scope** tab, add the new Secrets Broker account group that corresponds to the Secrets Broker instance where the Secrets Broker Vault Add-on was deployed.

e. On the **Users** tab of the new Secrets Broker Vault entitlement, add the Safeguard for Privileged Passwords user(s) that should have access to the Secrets Broker Vault root token.

2. Navigate back to the main Safeguard for Privileged Passwords page and select **Access Requests** | **My Requests**.

3. Create a new access request to fetch the root token that corresponds to the Secrets Broker Vault instance.

4. Fetch the root token and copy it to the clipboard.

Enabling and configuring the Username/Password authentication method

By default, the deployment of the Secrets Broker Vault Add-on automatically configures the vault by enabling the Key/Value secrets engine, but it doesn't enable any additional authentication methods. When the Secrets Broker Vault plugin is configured with Safeguard for Privileged Passwords accounts, the corresponding credentials are pushed to the vault K/V secrets engine. The Hashicorp vault CLI can be used to pull the secrets from the K/V secrets engine by simply logging in using the root token, or the vault can be configured with the Username/Password authentication method which will allow users to be added to access and manage the embedded vault.

To enable the Username/Password authentication method

1. Log into the vault using the root token that is stored in Safeguard for Privileged Passwords. For more information, see Getting the vault root token from the connected Safeguard for Privileged Passwords appliance.

2. Use the following command to enable the Username/Password authentication method. For more information, see Userpass Auth Method: Configuration.

```
vault auth enable userpass
```

3. Create a new admin policy. This is done using the following command which adds the policy that is contained in the `.hcl` file to the vault policies. For more information, see Policy.

```
vault policy write adminpolicy "<path to file>/admin-policy.hcl
```

Example `.hcl` file

```
# Read system health check
path "sys/health"
{
capabilities = ["read", "sudo"]
}
# Create and manage ACL policies broadly across Vault
# List existing policies
path "sys/policies/acl"
{
capabilities = ["list"]
}
# Create and manage ACL policies
path "sys/policies/acl/*"
{
capabilities = ["create", "read", "update", "delete", "list", "sudo"]
}
# Enable and manage authentication methods broadly across Vault
# Manage auth methods broadly across Vault
path "auth/*"
{
capabilities = ["create", "read", "update", "delete", "list", "sudo"]
}
# Create, update, and delete auth methods
path "sys/auth/*"
{
capabilities = ["create", "update", "delete", "sudo"]
```

```
}
# List auth methods
path "sys/auth"
{
capabilities = ["read"]
}
# Enable and manage the key/value secrets engine at `secret/` path
# List, create, update, and delete key/value secrets
path "secret/*"
{
capabilities = ["create", "read", "update", "delete", "list", "sudo"]
}
# Manage secrets engines
path "sys/mounts/*"
{
capabilities = ["create", "read", "update", "delete", "list", "sudo"]
}
# List existing secrets engines.
path "sys/mounts"
{
capabilities = ["read"]
}
```

4. Create a new user/password and assign policies. This is done using the following CLI command which adds a new user to the user database that is enabled with the policies that correspond to adminpolicy and oneidentity policy. For more information, see Auth.

    ```
    vault write auth/userpass/users/<username> password=<password>
    policies=adminpolicy,oneidentity
    ```

Enabling a new secrets engine in the embedded Secrets Broker Vault

The embedded vault that is deployed by the Secrets Broker Vault Add-on deploys a vault that is completely compatible with the Hashicorp vault CLI. The embedded vault can be configured with additional secrets engines. One of the secrets engines that can be configured is the database engine. For more information about how to configure a database secrets engine or any of the other secrets engines, see Secrets Engines.

1. To enable the database secrets engine, use the following command. For more information, see PostgreSQL Database Secrets Engine: Setup.

   ```
   vault secrets enable database
   ```

2. To configure the database secrets engine with the PostgreSQL plugin, use the following command. For more information, see PostgreSQL Database Secrets Engine: Setup.

   ```
   vault write database/config/postgresql-database plugin_name=postresql-database-plugin allowed_roles=postgresql-role connection_url="postgresql://{{username}}:{{password}}@localhost:5432" username="postgresqluser" password="userpass"
   ```

3. To generate a new credential, use the following command. For more information, see PostgreSQL Database Secrets Engine: Usage.

   ```
   vault read database/creds/postgresql-role
   ```

Pulling credentials from the OneIdentity K/V secrets engine

By default, the Secrets Broker Vault Add-on enables the Key/Value secrets engine in the embedded vault and configures a OneIdentity policy for storing the credentials that are pushed from Safeguard for Privileged Passwords. Accessing the credentials can be done using the Hashicorp vault CLI or the Hashicorp REST API. The credentials can then be used in other parts of a devops environment as needed. For more information, see KV Secrets Engine - Version 2: Writing/Reading arbitrary data.

1. Get a list of all the accounts whose credentials have been pushed from Safeguard for Privileged Passwords and are available from the embedded vault, use the following command:

   ```
   vault kv list oneidentity
   ```

2. Get the metadata and credential for a specified account, use the following command:

   ```
   vault kv get oneidentity/<account/key name>
   ```

Removing the Secrets Broker Vault

To remove the Secrets Broker Vault

1. Open the Secrets Broker Vault settings page.

2. Click the **Secrets Broker Vault** button in the **Add-ons** section.

3. On the **Add-on Settings** dialog, click the **Delete Add-on** button to remove the Secrets Broker Vault.

4. On the Delete Add-on dialog, click **Delete Add-on**.

   NOTE: Although not required, it is suggested that you leave the **Restart Safeguard Secrets Broker for DevOps Service** option selected. You can also use the **Restart Secrets Broker** option (accessed using the ⚙ button) to manually perform the restart.

One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product