Integrate Terraform with Password Safe

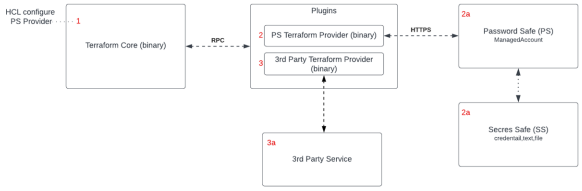
Overview

Terraform is an Infrastructure as Code (IaC) tool, used primarily to automate various infrastructure tasks. The Password Safe Terraform Provider enables the use of Password Safe's credential management solution with Terraform, providing a streamlined way for users to secure their Terraform associated secrets.

> ***Note:*** *The Password Safe Terraform Provider is available in the Terraform Registry at https://registry.terraform.io/providers/BeyondTrust/passwordsafe/latest.*

The integration works as follows:

1. The end user writes a Terraform configuration that sets Password Safe variables for the Password Safe provider. All configuration is run on Terraform plan/apply.
2. Terraform plan/apply invokes the Password Safe provider to retrieve the secret.
3. Lastly, Terraform plan/apply also invokes the third-party provider, using the Password Safe secret for authentication to the third-party service.



> ***Note:*** *This integration requires Password Safeversion 23.1 or higher for Secrets Safe secrets.*
>
> *This integration uses Transport Layer Security (TLS) 1.2 protocol.*

> *For more information, please see The Core Terraform Workflow at https://developer.hashicorp.com/terraform/intro/core-workflow.*

Configure Password Safe to Allow Terraform Files to Retrieve Secrets

The following sections outline how to set up the required authorization in Password Safe to allow Terraform HCL files to retrieve secrets.

> ***Note:*** *The following instructions are meant to be a quick start guide to help with Password Safe setup. For more information, please see the Password Safe Administration Guide at https://www.beyondtrust.com/docs/beyondinsight-password-safe/ps/admin/index.htm.*

Set Up API Key and User Account

1. Create an API registration in BeyondInsight (does not require a user password).
2. Create or use an existing Secrets Safe group.
3. Create or use an existing BeyondInsight user.

4. Add the API registration to the group.
5. Add the user to the group.
6. Add the Secrets Safe feature to the group.

Managed Accounts Setup

1. Create or use an existing access policy that has the **View Password Auto Approve** option set.
2. Add the **All Managed Accounts Smart Group** to the BeyondInsight group.
3. Add the access policy to the **All Managed Accounts Smart Group** role, and ensure that both requestor and approver are set.
4. Create or use an existing managed system.
5. Create or use an existing managed account associated with the managed system.
6. Configure the managed account with the **API Enabled** and **Max Concurrent Requests Unlimited** options selected.

Configure Terraform

The provider must be configured with an API key and a user account to access target secrets in Password Safe.

IMPORTANT!

Terraform state files and plan files contain sensitive information. Ensure best practices are followed for securing files.

Plugin Directory Setup

The provider is installed automatically when configured to use *beyondtrust/passwordsafe* in your Terraform configuration.

Note: *For Linux and Mac OS, set the executable bit on the provider. Example:* ***chmod +x****.*

Configure Provider

Configure the provider in your HCL files.

Example: *Provider Configuration Example:*

```
terraform {
   required_providers {
      passwordsafe = {
        source = "beyondtrust/passwordsafe"
        version = "1.0.1"
      }
    }
  }
```

```
# configure the Password Safe provider
provider "passwordsafe" {
   api_key = "${var.api_key}"
   url = "${var.url}"
   api_account_name = "${var.api_account_name}"
   verify_ca = true
   client_certificates_folder_path = "${var.client_certificates_folder_path}"
   client_certificate_name = "${var.client_certificate_name}"
   client_certificate_password = "${var.client_certificate_password}"
}
```

Configure Secrets Retrieval

Configure data sources to retrieve secrets from Password Safe. The first data block in the example below is *passwordsafe_managed_account*, used to retrieve managed account secrets. The second type of data source is *passwordsafe_secret*, which is used to retrieve credentials, text, and file secrets.

Example:

```
# retrieve a managed account secret
data "passwordsafe_managed_account" "manage_account" {
   system_name = "ServerStandard"
   account_name = "serveruser1"
}

output "manage_account" {
   value = "${data.passwordsafe_managed_account.manage_account.value}"
}


# retrieve a secrets safe credential
data "passwordsafe_secret" "secret_credential" {
   path = "folder1/folder2/folder3/folder4"
   title = "credLevel6"
}

output "secret_credential" {
   value = "${data.passwordsafe_secret.secret_credential.value}"
}

# retrieve a secrets safe file
data "passwordsafe_secret" "secret_file" {
   path = "folder1"
   title = "RootFile"
}

output "secret_file" {
   value = "${data.passwordsafe_secret.secret_file.value}"
```

```
}
```

Provider Arguments

- **url**: The URL for the Password Safe instance from which to request a secret.
- **api_account_name**: The username for the API request to the Password Safe instance. For use when authenticating with an API key.
- **api_key**: The API key for making requests to the Password Safe instance. For use when authenticating to Password Safe.
- **client_certificates_folder_path** (optional): The path to the client certificate associated with the Password Safe instance. For use when authenticating with an API key using a client certificate.
- **client_certificate_password** (optional): The password associated with the client certificate. For use when authenticating with an API key using a client certificate.
- **client_certificate_name** (optional): The name of the client certificate for use when authenticating with an API key using a client certificate.
- **verify_ca** (optional): Indicates whether to verify the certificate authority on the Password Safe instance. For use when authenticating to Password Safe.