Integrate Password Safe with Ansible

Ansible is an open-source software provisioning, configuration management, and application-deployment tool that enables IaC (infrastructure as code). Password Safe's integration with Ansible provides an easy to use method for users to secure and manage their Ansible related secrets for automated workflows. Secrets Safe users can leverage this integration for Continuous Integration and Continuous Deployment (CI/CD) and DevOps workflows.

The Secrets Safe Lookup plugin can be configured to retrieve ASCII secrets from Secrets Safe.

Prerequisites

The Secrets Safe Lookup plugin supports the following software, features, and configurations:

- Supports retrieval of secrets from BeyondInsight and Password Safe version 23.1 or later releases.
- Supports OAuth client secret for authentication across applications and services.
- Supports managed account password retrieval.
- Is compatible with Ansible core v2.14 and later releases.
- Is compatible with Python v3.11 and later releases.
- The Secrets Safe instance must be preconfigured with the secret in question and an account must be authorized to read it in order for this plugin to retrieve a secret.

IMPORTANT!

Warnings:

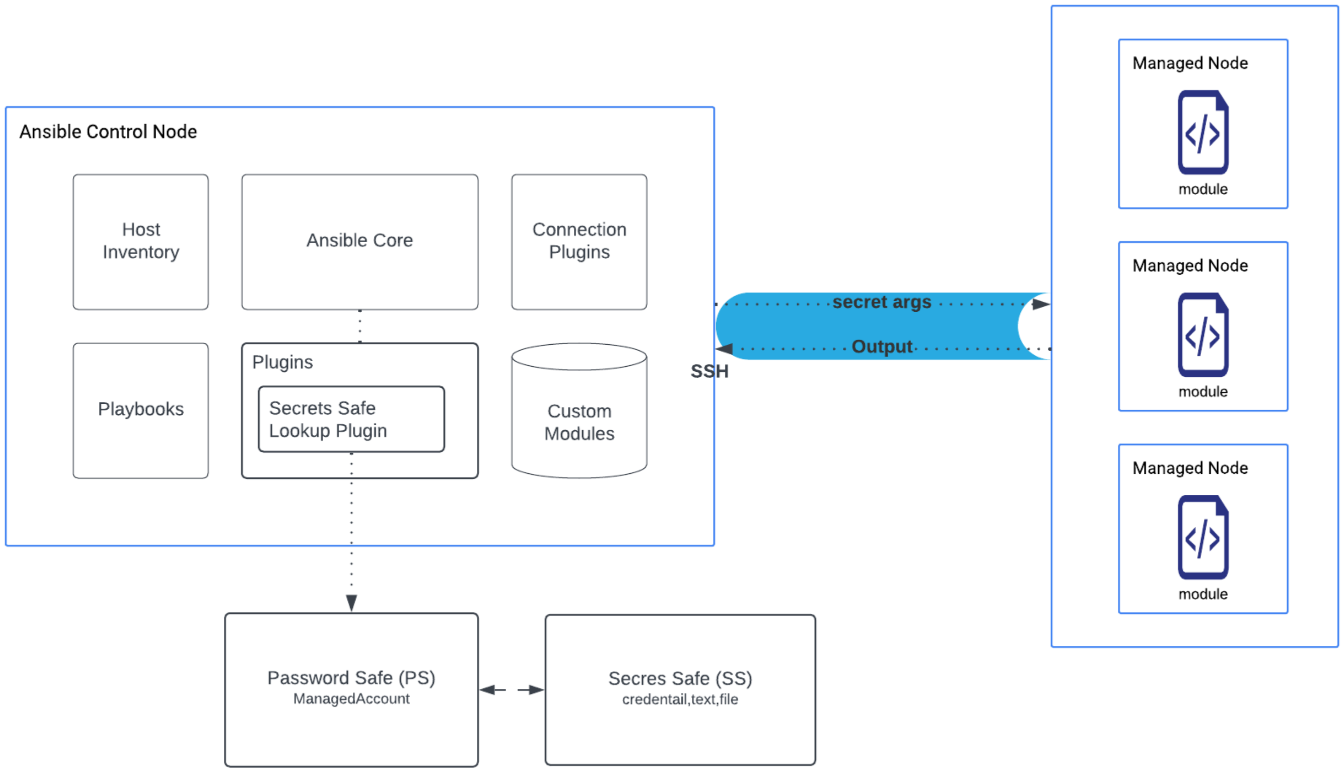
- *Do not log the secrets to stdout. It is important that security-minded engineers review playbooks composition before changes are run with access to secrets.*
- *Ansible writes lookup plugin arguments to stdout. Do not put secret text directly in the lookup plugin arguments. Use variables as an alternative.*

Overview

The Secrets Safe Lookup plugin is named **secrets_safe_lookup**. It is designed to be run on an Ansible Control node (Linux). The lookup plugin is available from the Ansible Galaxy Registry.

For more information, please see the Ansible Galaxy Registry at https://galaxy.ansible.com/beyondtrust/secrets_safe.

Architecture Workflow Diagram



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.*

General Setup

- Create an API registration in BeyondInsight. Ensure the **User password required** option is not enabled.
- Create or use an existing Secrets Safe group.
- Create or use an existing BeyondInsight user.
- Add API registration to the group.
- Add the user to the group.
- Add the Secrets Safe feature to the group. Assign read-only or full access based on your use case.

Managed Accounts Setup

1. Create or use an existing access policy that has the **View Password** option enabled with the **Auto Approve** option selected.
2. Add the **All Managed Accounts** Smart Group to the BeyondInsight group.
3. Add the access policy to Password Safe role assigned to the **All Managed Accounts** Smart Group. Ensure that both **Requestor** and **Approver** are selected in the assigned role.
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.

Use the Secrets Safe Lookup Plugin

Usage

- **api_url**
  - **description**: BeyondTrust Password Safe API URL.
  - **type**: string
  - **required**: True
- **retrieval_type**
  - **description**: Type of secret to retrieve (use MANAGED_ACCOUNT or SECRET)
  - **type**: string
  - **required**: True
- **client_id**
  - **description**: API OAuth Client ID.
  - **type**: string
  - **required**: True
- **client_secret**
  - **description**: API OAuth Client Secret.
  - **type**: string
  - **required**: True
- **secret_list**
  - **description**: List of secrets (path/title,path/title) or managed accounts (ms/ma,ms/ma) to be retrieved, separated by a comma.
  - **type**: string
  - **required**: True
- **certificate_path**
  - description: Password Safe API pfx Certificate Path. For use when authenticating using a Client Certificate.
  - type: string

  - required: False
- **certificate_password**
  - **description**: Password Safe API pfx Certificate Password. For use when authenticating using a Client Certificate.
  - **type**: string
  - **required**: False
- **verify_ca**
  - **description**: Indicates whether to verify the certificate authority on the Secrets Safe instance.
  - **type**: boolean
  - **default**: True
  - **required**: False

Usage Example:

Example:

```
"{{lookup('beyondtrust.secrets_safe.secrets_safe_lookup', api_url=apiURL, retrieval_
type='SECRET', client_id=clientIdFromEnvVar, client_secret=secretFromEnvVar, secret_
list=secretList, certificate_path=certificatePath, certificate_
password=certificatePasswordFromEnVar, wantlist=False, verify_ca=True)}}"
```

Output

- **description**: list of retrieved secret(s) in the requested order.
- **type**: list
- **elements**: str

Example Playbook

Example:

```
export PASSWORD_SAFE_CLIENT_ID=*******************
export PASSWORD_SAFE_CLIENT_SECRET=*******************
export CERTIFICATE_PASSWORD=*******************
export PASSWORD_SAFE_API_URL=https://example.com:443/BeyondTrust/api/public/v3

---
- name: book
  hosts: localhost
  connection: local
  vars:
      apiURL: "{{ lookup('ansible.builtin.env', 'PASSWORD_SAFE_API_URL') }}"

      clientIdFromEnvVar: "{{ lookup('ansible.builtin.env', 'PASSWORD_SAFE_CLIENT_ID')
        }}"
      secretFromEnvVar: "{{ lookup('ansible.builtin.env', 'PASSWORD_SAFE_CLIENT_SECRET')
        }}"

      certificatePasswordFromEnVar:  "{{ lookup('ansible.builtin.env',
        'CERTIFICATE_PASSWORD') }}"
      certificatePath: "<path>/ClientCertificate.pfx"

      secretManagedAccounts: "fake_system/fake_ managed_account,fake_system/
        fake_managed_account01"
      gotManagedAccount: "{{lookup('beyondtrust.secrets_safe.secrets_safe_lookup',
        api_url=apiURL, retrieval_ type='MANAGED_ACCOUNT', client_id=clientIdFromEnvVar,
        client_secret=secretFromEnvVar, secret_list=secretManagedAccounts,
        certificate_path=certificatePath,
        certificate_password=certificatePasswordFromEnVar, wantlist=False)}}"

      secretList: "fake_grp/credential,fake_grp/file"
      gotSecrets: "{{lookup('beyondtrust.secrets_safe.secrets_safe_lookup',
        api_url=apiURL, retrieval_type='SECRET', client_id=clientIdFromEnvVar,
        client_secret=secretFromEnvVar, secret_list=secretList,
        certificate_path=certificatePath,
        certificate_password=certificatePasswordFromEnVar, wantlist=False,
        verify_ca=True)}}"


      secretList: "fake_grp/credential,fake_grp/file"
      gotSecrets: "{{lookup('beyondtrust.secrets_safe.secrets_safe_lookup',
        api_url=apiURL, retrieval_type='SECRET', client_id=clientIdFromEnvVar,
        client_secret=secretFromEnvVar, secret_list=secretList,
        certificate_path=certificatePath,
        certificate_password=certificatePasswordFromEnVar, wantlist=False,
        verify_ca=True)}}"

  tasks:
- name: Display Retrieved Managed accounts
      ansible.builtin.debug:
      msg: "{{ gotManagedAccount }}"
- name: Display Retrieved Secrets
      ansible.builtin.debug:
      ansible.builtin.debug:
```