Red Hat Enterprise Linux Automation with Ansible

FEEDBACK TRANSLATIONS CERTIFICATE OF ATTENDANCE

Guided Exercise: Managing Secrets ☆

In this exercise, you will encrypt sensitive variables with Ansible Vault to protect them, and then run a playbook that uses those variables.

Outcomes

You should be able to:

- Execute a playbook using variables defined in an encrypted file.

Log in to `workstation` as `student` using `student` as the password.

On `workstation`, run the `lab data-secret start` command. This script ensures that Ansible is installed on `workstation` and creates a working directory for this exercise. This directory includes an inventory file that points to `servera.lab.example.com` as a managed host, which is part of the `devservers` group.

```
[student@workstation ~]$ lab data-secret start
```

Procedure 3.2. Instructions

1. On `workstation`, as the `student` user, change to the `/home/student/data-secret` working directory.

   ```
   [student@workstation ~]$ cd ~/data-secret
   [student@workstation data-secret]$
   ```

2. Edit the contents of the provided encrypted file, `secret.yml`. The file can be decrypted using `redhat` as the password. Uncomment the `username` and `pwhash` variable entries.

   2.1. Edit the encrypted file `/home/student/data-secret/secret.yml`. Provide a password of `redhat` for the vault when prompted. The encrypted file opens in the default editor, `vim`.

   ```
   [student@workstation data-secret]$ ansible-vault edit secret.yml
   Vault password: redhat
   ```

   2.2. Uncomment the two variable entries, then save the file and exit the editor. They should appear as follows:

   ```
   username: ansibleuser1
   pwhash: $6$jf...uxhP1
   ```

3. Create a playbook named `/home/student/data-secret/create_users.yml` that uses the variables defined in the `/home/student/data-secret/secret.yml` encrypted file.

Configure the playbook to use the `devservers` host group. Run this playbook as the `devops` user on the remote managed host. Configure the playbook to create the `ansibleuser1` user defined by the `username` variable. Set the user's password using the password hash stored in the `pwhash` variable.

```
---
- name: create user accounts for all our servers
  hosts: devservers
  become: True
  remote_user: devops
  vars_files:
    - secret.yml
  tasks:
    - name: Creating user from secret.yml
      user:
        name: "{{ username }}"
        password: "{{ pwhash }}"
```

4. Use the `ansible-playbook --syntax-check` command to verify the syntax of the `create_users.yml` playbook. Use the `--ask-vault-pass` option to prompt for the vault password, which decrypts `secret.yml`. Resolve any syntax errors before you continue.

```
[student@workstation data-secret]$ ansible-playbook --syntax-check \
> --ask-vault-pass create_users.yml
Vault password (default): redhat

playbook: create_users.yml
```

> **NOTE**
>
> Instead of using `--ask-vault-pass`, you can use the newer `--vault-id @prompt` option to do the same thing.

5. Create a password file named `vault-pass` to use for the playbook execution instead of asking for a password. The file must contain the plain text `redhat` as the vault password. Change the permissions of the file to `0600`.

```
[student@workstation data-secret]$ echo 'redhat' > vault-pass
[student@workstation data-secret]$ chmod 0600 vault-pass
```

6. Execute the Ansible Playbook using the `vault-pass` file, to create the `ansibleuser1` user on a remote system using the passwords stored as variables in the `secret.yml` Ansible Vault encrypted file.

```
[student@workstation data-secret]$ ansible-playbook \
> --vault-password-file=vault-pass create_users.yml

PLAY [create user accounts for all our servers] ******************************

TASK [Gathering Facts] *******************************************************
ok: [servera.lab.example.com]

TASK [Creating users from secret.yml] ****************************************
changed: [servera.lab.example.com]

PLAY RECAP *******************************************************************
servera.lab.example.com    : ok=2    changed=1    unreachable=0    failed=0
```

7. Verify that the playbook ran correctly. The user `ansibleuser1` should exist and have the correct password on `servera.lab.example.com`. Test this by using `ssh` to log in as that user on `servera.lab.example.com`. The password for `ansibleuser1` is `redhat`. To make sure that SSH only tries to authenticate by password and not by an SSH key, use the `-o PreferredAuthentications=password` option when you log in.

   Log off from `servera` when you have successfully logged in.

```
[student@workstation data-secret]$ ssh -o PreferredAuthentications=password \
> ansibleuser1@servera.lab.example.com
ansibleuser1@servera.lab.example.com's password: redhat
Activate the web console with: systemctl enable --now cockpit.socket

[ansibleuser1@servera ~]$ exit
logout
Connection to servera.lab.example.com closed.
```

Finish

On `workstation`, run the `lab data-secret finish` script to clean up this exercise.

```
[student@workstation ~]$ lab data-secret finish
```

This concludes the guided exercise.

← PREVIOUS (/ROL/APP/COURSES/RH294-8.4/PAGES/CH03S03) → NEXT (/ROL/APP/COURSES/RH294-8.4/PAGES/CH03S05)