Ansible-Day-6

13 December 2023 14:58

==

Ansible vault

==

- This feature allows us to encrypt the sensitive information like passwords & secret key's rather than using it as a plain text in ansible playbooks or roles.

- **Create a file with secret key in plain text**
  ```
  echo "API_KEY: ThisIsSuperSecretDontDisplay" > vars/api_key.yml
  ```

- **Create a sample playbook to display value of the variable API_KEY**
  ```
  ---
  - hosts: all
    become: yes
    vars_file:
          - vars/api_key.yml
    tasks:
          - name: "Display the API_KEY value"
            debug:
                  msg: "{{ API_KEY }}"
  ```

  ```
  ansible-playbook -v test.yml
  ```
  o/p: ThisIsSuperSecretDontDisplay

- **Let's encrypt the vars/api_key.yml file**

  - **Encrypt the file**
    ```
    ansible-vault encrypt vars/api_key.yml
    cat vars/api_key.yml
    ```

    The file whatever present vars/api_key.yml is encrypted now.
    ```
    cat vars/api_key.yml
    ```

  - So we can push this file to GitHub along with our playbook/role.

  - Execute ansible playbook
    ```
    ansible-playbook -v test.yml --ask-vault-pass
    ```

  - Execute ansible playbook with password stored in a file
    ```
    ansible-playbook -v test.yml --vault-password-file
    ```

  - Having a separate script file for getting the passwords is also possible.
    you need to make sure the script file is executable and the password is printed to standard output for it to work without annoying errors.
    ```
    ansible-playbook launch.yml --vault-password-file ~/ .vault_pass.py
    ```

- **Decrypt the file**
  ```
  ansible-vault decrypt vars/api_key.yml
  ```

- View an Encrypted File in Ansible
  ```
  ansible-vault view vars/api_key.yml
  ```

- Edit an Encrypted File in Ansible
  ```
  ansible-vault edit vars/api_key.yml
  ```

- Change the password for the encrypted file
  ```
  ansible-vault rekey vars/api_key.yml
  ```

==

What is ansible role?

==

1. Normally when we write playbook for particular requirement we mention everything in a single file
   like
   - tasks
   - variables
   - handlers
   - metadata

2. Ansible roles allow you to develop reusable automation components by grouping related automation things,
   - configuration files
   - templates
   - tasks
   - handlers

3. Roles will have structured layout.

4. Ansible roles consists of multiple folders & each folder by default contains main.yml file.

==

How to create a role

==

1. Create ansible role for apache-setup

```
ansible-galaxy init apache-setup
```

2. The role structure of the apache-setup will be like

```
apache-setup
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

3. Now let's understand every folder purpose,

a. defaults/main.yml ==> In this configuration file we declare default values for the variables.
Example: httpd is the default package name for installing apache on Linux machine,
so we will declare it as variable here.

b. files ==> We copy the static files like .tar.gz/.zip files into this directory.
During ansible role execution files that are present over here will be transfer to target machine.

c. handlers/main.yml ==> In this section we will declare handlers like restart apache.

d. meta/main.yml ==> It's metadata file, which contains the information like
   - author
   - license
   - dependency management

From this we can another dependency role for current apache-setup role.
Example: Let's assume we have two roles for maven-setup & java-setup
When we create maven-setup role, we can make java-setup role as dependecy in meta/main.yml

e. README.md ==> In this file we write Description about the role & its usage.

f. tasks ==> In this folder we will create the list of the .yml files related to our activities
Example:
Installing apache {apache.yml}
Configure apache {configure.yml}

g. templates ==> This folder contains the template files(jinja2) used by the role to create the actual configuration files.
Example: After installation there might be need to update the configuration files as well,
like including usernames/os-details in-order to do this we will use templates.

h. vars ==> Used to declare the variables that are needed for ansible playbooks

Here we can see variables we can declare at defaults/main.yml & vars/main.yml.

The variables which are declared in vars/main.yml have high priority.

4. Create a ansible for apache-setup & push it GitHub