# Ansible File

Ansible file module is used to creating and deleting the file or multiple files in the remote server. You can also create and delete the directories and change the permissions of the data.

You can also create and delete the soft links (symlinks) as well as hard links. With the help of the Ansible file module, you can set the permission of the files.

## Creating a File in Remote Server

In the Ansible file module, we have different parameters. We are using **path** and **state** parameters that are must in every file module. In the file parameter, we will mention the path of the file in the remote server. On this path, only the file will be created.

**At path:** It mentions the path of the file in the remote server.

**At state:** It mentions touch, and touch will create file exact like Linux command.

Then, it will create a new empty file with the name devops.txt. So mention filename in the path. So in the state: we will mention touch to create the file.

```
- name: create the file in a remote server
  file:
   path: /path/to/file/in/remote/server/devops.txt
   state: touch
```

## Deleting a File in Remote Server

If you want to delete any command in the remote server. So at path parameter, mention the path of the file which you want to delete.

**At path:** Mention the path of the file in the remote server.

**At state:** Mention absent to delete the file.

So in the state: we will use touch to create the file, absent to delete the e file.

```
- name:  delete the file in a remote server
```

```
 file:
  path: /etc/abcd.conf
  state: absent
```

## Creating a File with Permissions

We can also create the file with permission by using the file module.

At the mode parameter: we have 4 digits. Always mention zero at the starting, and remaining digits will be your file permissions.

At owner parameter: mention the owner of the file.

```
tasks:
  - name: Ansible file module to create a new file with permissions.
    file:
    path: /path/to/cretae/file/devops.txt
    state: touch
    mode: 0421
    owner: devops
```

This permission will be set to that newly created file.

```
file:
 path: /path/to/cretae/file/devops.txt
 state: touch
 mode: "u=rw,g=w,o=e"
 owner: devops
```

Both the codes work the same, but in the other code, we are using the symbolic mode, which is equivalent to 0421.

## Creating Multiple Files

A path parameter: we can create a loop to create multiple files by using "{{item}}".

At with_items parameter: mention file names which you want to create.

By using "{{item}}" and with_items parameter, we can create loop or multiple files.

```
tasks:
- name: Ansible file module to create multiple files
  file:
   path: "{{ item }}"
   state: touch
   mode: 0421
  with_items:
  - devops1.txt
  - devops2.txt
  - devops3.txt
```

## Deleting Multiple Files

The code will be the same to create multiple files and to delete files but a small change in the state parameter.

State parameter: Touch the create files and absent to delete files.

```
- name: Ansible file module to delete multiple files
  file:
   path: "{{ item }}"
   state: absent
  with_items:
  - devops1.txt
  - devops2.txt
  - devops3.txt
```

# Ansible Vault

Ansible Vault is a feature which allows user to encrypt values and data structures within Ansible projects. This provides the ability to secure any secrets or sensitive data that is necessary to run Ansible plays successfully but should not be publicly visible, such as private keys or passwords. Ansible automatically decrypts the vault-encrypted content at runtime when the key is provided.

To integrate these secrets with regular Ansible data, both the Ansible and Ansible-playbook commands, for executing ad hoc tasks and structured playbook respectively, have support for decrypting vault-encrypted content at runtime.

Ansible Vault is implemented with file-level granularity; it means files are either entirely encrypted or unencrypted. It uses the AES256 algorithm to provide symmetric encryption keyed to a user-supplied password.

This means the same password is used to encrypt and decrypt the content, which is helpful from a usability standpoint. Ansible can identify and decrypt any vault-encrypted files it finds while executing a task or playbook.

Though there is a proposal to change this, at the time of writing this, users can only pass in a single password to Ansible. It means that each of the encrypted files involved must share a password.

## Using Ansible Vault

The simple use of the Ansible vault is to encrypt variables files. It can encrypt any YAML file, but the most common files to encrypt are:

- A role's defaults/ main.yml file
- A role's vars/main.yml file
- Files within the group_vars directory
- Any other file used to store variables

## Encrypting an Existing File

You can encrypt a regular plaintext variable file by using the ansible vault and define the password that needed later to decrypt it.

```
#encrypt a role's defaults/main.yml file
ansible-vault encrypt defaults/main.yml
>New vault password:
>Confirm new vault password:
>Encryption successful
```

The ansible-vault command will prompt you a password twice. After that, the file will be encrypted.

## Creating an Encrypted File

To create an encrypted data file, use the ansible-vault to create command, and pass the filename.

```
$ansible-vault create <file name>
```

You will be prompted to create a password and then confirm it by re-typing it.

Once your password is confirmed, a new file will be created and will open an editing a window. By default, the editor for Ansible vault is VI. You can add data, save it, and exit from it.

## Editing Encrypted Files

If you want to edit the encrypted file, you can edit it using ansible-vault edit command. This command will decrypt the file to a temporary file and allow you to edit the file.

```
$ansible-vault edit <file name>
```

You will be prompted to insert the vault password. The decrypted file will open in a VI editor, and then you can make the required changes. Save the changes and removing the temporary file.

## Rekeying Encrypted Files

If you want to change your password on a vault on a vault-encrypted file, you can do it by using the rekey command.

```
$ansible-vault rekey <file1> <file2> <file3>
```

The above command can rekey multiple data files at once and ask for the original password and the new password.

## Encrypting Unencrypted Files

If you have existing files which you want to encrypt, use the ansible-vault encrypt command. This command can operate on multiple files at once.

```
$ansible-vault encrypt <file1> <file2> <file3>
```

## Decrypting Encrypted Files

If you have existing files that you no longer want to keep encrypted, you can decrypt them permanently by running the ansible-vault decrypt command. This command will save them unencrypted to the disk.

```
$ansible-vault decrypt <file1> <file2> <file3>
```

## Viewing Encrypted Files

If you want to view the contents of an encrypted file without editing it, then you can use the ansible-vault view command.

```
$ansible-vault view <file1> <file2> <file3>
```