Lab: Managing Secrets Using Ansible Vault

Introduction:

Secrets are meant to stay secret. Whether they are login credentials to a cloud service or passwords to database resources, they are secret for a reason. Should they fall into the wrong hands, they can be used to discover trade secrets, customers' private data, create infrastructure for nefarious purposes, or worse. All of which could cost you or your organization a lot of time, money, and headache!

Objectives:

- **Creating new encrypted files**
- **Encrypting existing unencrypted files**
- **Decrypting encrypted files**
- **Changing the encryption password on files**
- **Running ansible-playbook referencing encrypted files**

1 Creating new encrypted files

1.1 Type 1: **PASSWORD PROMPT**

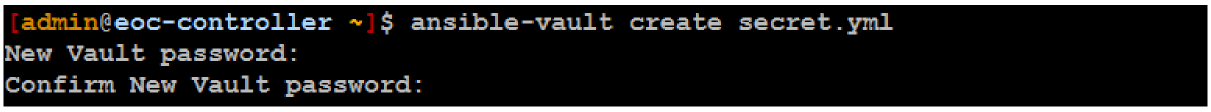
1.2 Let's create an encrypted File.

Note: It prompts for password

Password: **centos**

```
# ansible-vault create secret.yml
```

Output:

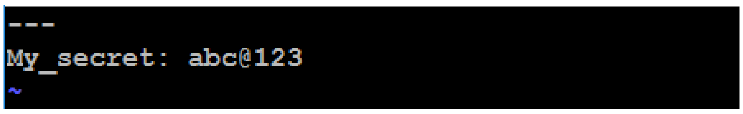


```
[admin@eoc-controller ~]$ ansible-vault create secret.yml
New Vault password:
Confirm New Vault password:
```

1.3 Let's add some data inside the **secret.yml.**

```
---
My_secret: abc@123
```

Output:



```
---
My_secret: abc@123
~
```

Note: Type **i** to switch into insert mode so that you can start editing the file.

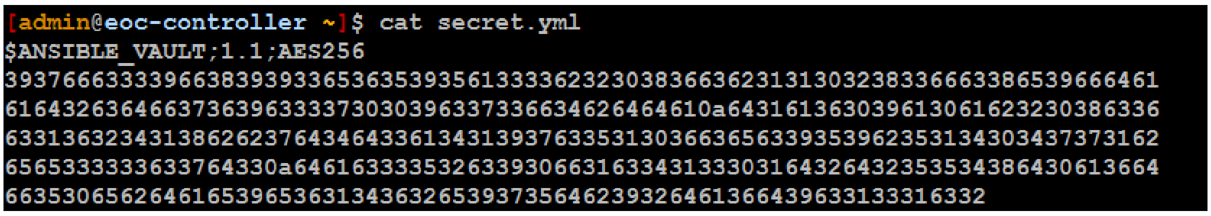
1.4 Let's save the data and exit from the vim editor by executing the below command.

```
# :wq
```

1.5 Let's verify the data inside the **secret.yml**.

```
# cat secret.yml
```

Output:



```
[admin@eoc-controller ~]$ cat secret.yml
$ANSIBLE_VAULT;1.1;AES256
39376663333966383939336536353935613333623230383663623131303238336663386539666461
61643263646637363963333730303963373366346264646610a6431613630396130616232303863336
63313632343138626237643464336134313937633531303663656339353962353134303437373162
6565333333633764330a6461633335326339306631633431333031643264323535343864306136664
66353065626461653965363134363265393735646239326461366439633133316332
```

1.6 Type 2: PASSWORD FILE

1.7 Let's create the file and pass that file as a password.

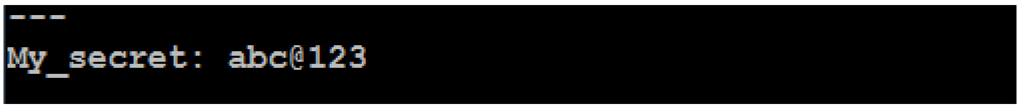
```
# echo "my long password" > password_file
```

```
# ansible-vault create --vault-password-file password_file
more_secret.yml
```

1.8 Let's add some data inside the **more_secret.yml.**

```
---
My_secret: abc@123
```

Output:



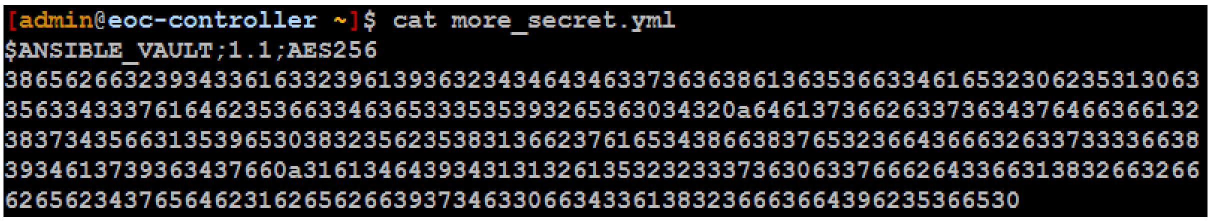
```
---
My_secret: abc@123
```

Note: save and exit by typing**: wq**

1.9 Let's verify the data inside the **more.secret.yml**.

```
# cat more_secret.yml
```

Output:



```
[admin@eoc-controller ~]$ cat more_secret.yml
$ANSIBLE_VAULT;1.1;AES256
38656266323934336163323961393632343464346337363638613635366334616532306235313063
35633433376164623536633463653335353932653630343200a6461373662633736343764636366132
38373435663135396530383235623538313662376165343866383765323664366632633733336638
3934613739363437660a316134643934313132613532323337363063376662643366313832663266
62656234376564623162656266393734633066343361383236663664396235366530
```

1.10 Type 3: Password Script

1.11 Let's create script and pass that script as a password.

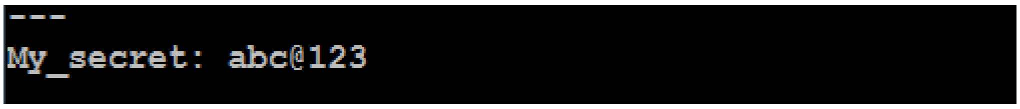
```
# cat > password.sh << EOF
!/bin/bash
echo "a long password"
EOF
```

```
# ansible-vault create --vault-password-file password.sh
password-as-script.yml
```

1.12 Let's add some data inside the **password-as-script.yml**.

```
---
My_secret: abc@123
```

Output:

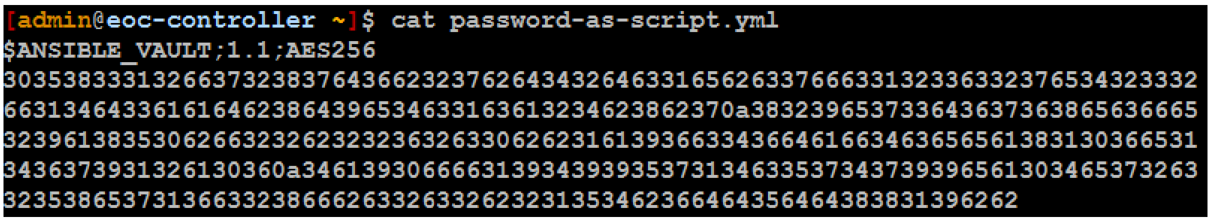


```
---
My_secret: abc@123
```

1.13 Let's verify the data inside the **password-as-script.yml**.

```
# cat password-as-script.yml
```

Output:



```
[admin@eoc-controller ~]$ cat password-as-script.yml
$ANSIBLE_VAULT;1.1;AES256
30353833313266373238376436623237626434326463316562633766633132336332376534323332
6631346433616164623864396534633163613234623862370a383239653733643637363865636665
32396138353062663232623232363263306262316139366334366461663463656561383130366531
3436373931326130360a346139306666313934393935373134633537343739396561303465373263
32353865373136633238666263326332623231353462366464356464383831396262
```

2 Encrypting existing unencrypted files

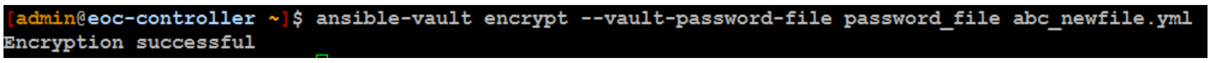
2.1 Let's create a file with some secret.

```
# cat > abc_newfile.yml<< EOF
---
Something: "better than nothing"
EOF
```

2.2 Encrypting the created file **abc_newfile.yml**.

```
# ansible-vault encrypt --vault-password-file password_file
abc_newfile.yml
```

Output:

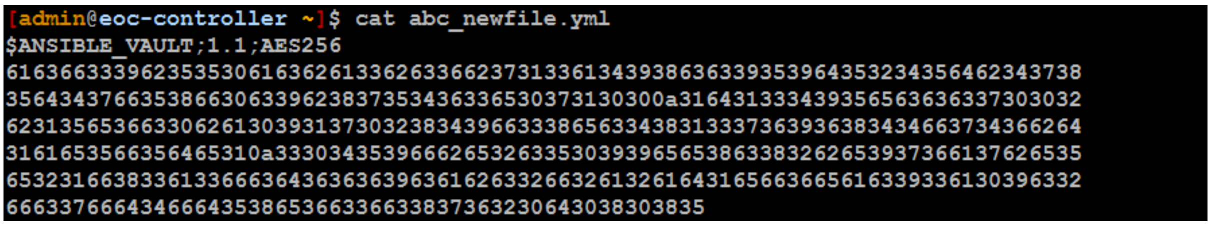


```
[admin@eoc-controller ~]$ ansible-vault encrypt --vault-password-file password_file abc_newfile.yml
Encryption successful
```

2.3 Let's check the contents of the **abc_newfile.yml** file.

```
# cat abc_newfile.yml
```

Output:



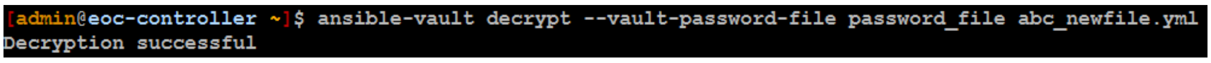
```
[admin@eoc-controller ~]$ cat abc_newfile.yml
$ANSIBLE_VAULT;1.1;AES256
61636633396235353061636261336263366237313361343938636339353964353234356462343738
3564343766353866306339623837353436336530373130300a316431333439356563636337303032
62313565366330626130393137303238343966333865633438313337363936383434663734366264
3161653566356465310a333034353966626532633530393965653863383262653937366137626535
65323166383361336663643636363963616263326632613261643165663665616339336130396332
6663376664346664353865366336633837363230643038303835
```

3 Decrypting encrypted files

3.1 Using decrypt option.

```
# ansible-vault decrypt --vault-password-file password_file
abc_newfile.yml
```

output:

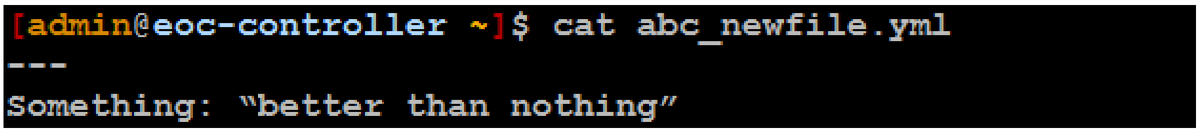


```
[admin@eoc-controller ~]$ ansible-vault decrypt --vault-password-file password_file abc_newfile.yml
Decryption successful
```

3.2 Let's check the contents of the **abc_newfile.yml** file.

```
# cat abc_newfile.yml
```

Output:



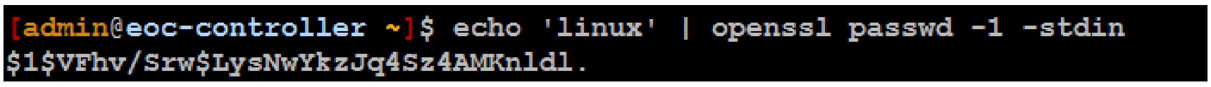
```
[admin@eoc-controller ~]$ cat abc_newfile.yml
---
Something: "better than nothing"
```

4 Changing the encryption password on files

4.1 Let's generate a hashed password.

```
# echo 'linux' | openssl passwd -1 -stdin
```

Output:

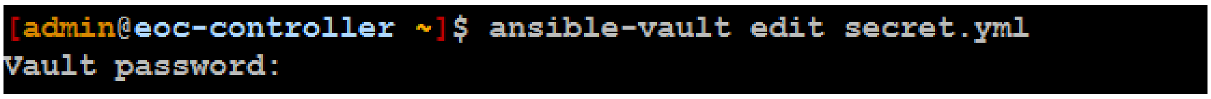


```
[admin@eoc-controller ~]$ echo 'linux' | openssl passwd -1 -stdin
$1$VFhv/Srw$LysNwYkzJq4Sz4AMKnldl.
```

4.2 Let's edit the **secret.yml** file using ansible vault.

```
# ansible-vault edit secret.yml
```

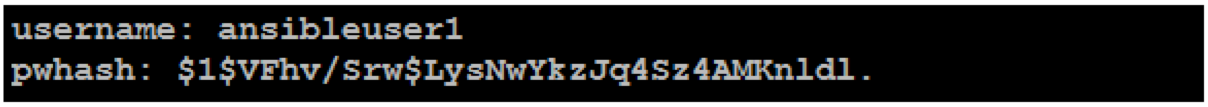
Output:



```
[admin@eoc-controller ~]$ ansible-vault edit secret.yml
Vault password:
```

Password: **centos**

4.3 Append the content inside the **secret.yml** file with the user details.



```
username: ansibleuser1
pwhash: $1$VFhv/Srw$LysNwYkzJq4Sz4AMKnldl.
```

Note: Replace the pwhash value with the hashed value generated in the last step.

5 Running ansible-playbook referencing encrypted files

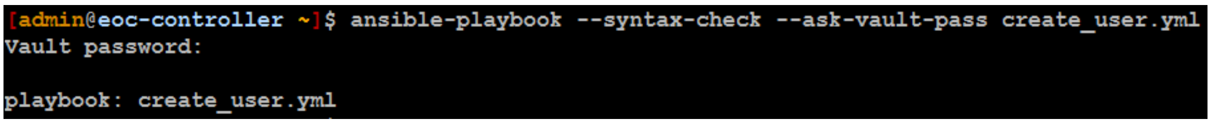
5.1 Let's create a file manifest named **create_user.yml.**

```
# cat > create_user.yml << EOF
---
- name: create user accounts for all our servers
  hosts: eoc-node1
  become: yes
  vars_files:
  - secret.yml
  tasks:
    - name: Creating user from secret.yml
      user:
        name: "{{ username }}"
        password: "{{ pwhash }}"
EOF
```

5.2 Let's dry run the manifest file to check the syntax.

```
# ansible-playbook --syntax-check --ask-vault-pass
create_user.yml
```

Output:



```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check --ask-vault-pass create_user.yml
Vault password:

playbook: create_user.yml
```

Password: centos

5.3 Let's create a password file named **vault-pass** to use for the playbook execution instead of asking for a password.

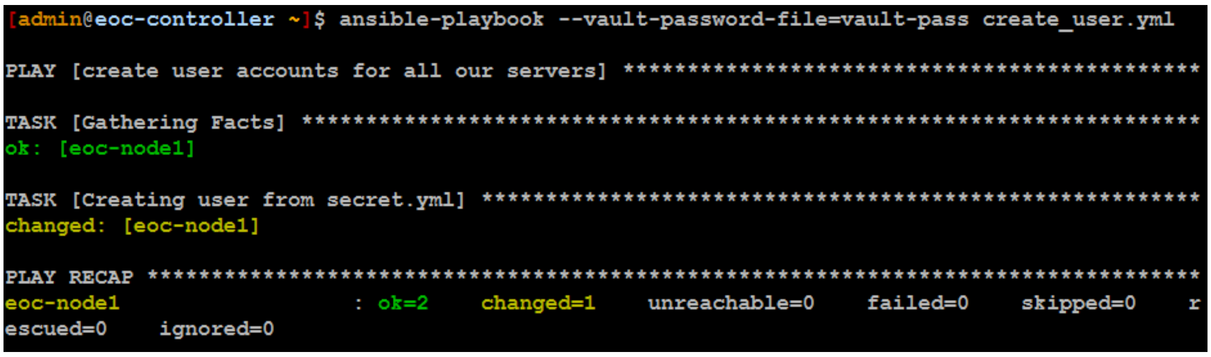
5.4 The file must contain the plain text centos as the vault password. Change the permissions of the file to 0600.

```
# echo 'centos' > vault-pass
# chmod 0600 vault-pass
```

5.5 Let's 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.

```
# ansible-playbook --vault-password-file=vault-pass
create_user.yml
```

Output:



```
[admin@eoc-controller ~]$ ansible-playbook --vault-password-file=vault-pass create_user.yml

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

TASK [Gathering Facts] **********************************************************************
ok: [eoc-node1]

TASK [Creating user from secret.yml] ********************************************************
changed: [eoc-node1]

PLAY RECAP **********************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    r
escued=0    ignored=0
```