



Ansible pour professionnels

Linux / Unix

Le support du cours «Ansible pour professionnel Linux/Unix » est non contractuel ; il ne doit pas être redistribué et/ou reproduit en partie ou en totalité sans permission explicite et écrite de la société Adlere.

Red Hat, le logo Red Hat, OpenShift et Ansible sont des marques déposées ou commerciales de Red Hat, Inc ou ses filiales aux États-Unis et dans d'autre pays. Linux® est une marque déposée de Linus Torvalds aux États-Unis et dans d'autre pays.

UNIX ® est une marque déposée par « The Open Group » aux Etats-Unis et dans d'autres pays.
Windows® est une marque déposée de Microsoft Corporation aux États-Unis et dans d'autre pays.

Les autres marques citées sont déposées par leurs propriétaires respectifs.



+adlere
DIGITAL EXPERTISE

ansible-vault

- fonctionnalité pour chiffrer des données sensibles (AES256, clef symétrique, AES128 pour les versions plus anciennes)
- les données chiffrées peuvent être contenues dans le playbook ou dans un fichier externe
- un fichier chiffré peut être désigné sur la ligne de commande ou dans le playbook
- commande 'ansible-vault' pour mettre en œuvre la fonctionnalité

ansible-vault <option>


create	crée un nouveau fichier chiffré
decrypt	déchiffre un fichier
edit	édition d'un fichier chiffré
view	visualiser le contenu du fichier chiffré
encrypt	chiffre un fichier
rekey	change le chiffrement d'un fichier (ré-écriture complète)
encrypt_string	chiffrement d'une chaîne

--ask-vault-pass, --vault-password-file --vault-id




Chiffrer un fichier entier

5




```
 Lorem ipsum dolor sit amet,
consectetur adipiscing
elit. Nullam commodo, nisl
sit amet fringilla
dignissim, augue dui
ultrices ipsum, et dapibus
massa lorem quis nulla.
Nunc ac mi eget elit
commodo interdum et eget
neque.
```




```
$ ansible-vault decrypt lorem.txt
Vault password:
Decryption successful
```

```
$ ansible-vault encrypt lorem.txt
New Vault password:
Confirm New Vault password:
Encryption successful
```



```
$ANSIBLE_VAULT;1.1;AES256
3762346639393365333263616636306533653
6646136636361343931646465393066313139
6263363134393030386264623338613766383
438303662613066350a663466396635313766
3737333537626361643562373532323265303
6636366313834666237356265616436396238
663036623732646138
[...]
```





Cas #1 : variable dans un playbook

```
$ ansible-vault encrypt_string 'My_Passw0rd' --name 'db_passwd'

New Vault password: <password>
Confirm New Vault password: <password>
Encryption successful
db_passwd: !vault |
          $ANSIBLE_VAULT;1.1;AES256
          33653831326366656138613535366634623434616461363363616165633862643934346230663539
          3132653166333863656334663336613665643666386366620a663233343537656439383433323334
          61393764343165353264646566363638623330616333313139363635336531396232613563346131
          6462303030333237370a363763666530636161363661316462353066373334373632353462366631
          6238
```

- copier le résultat précédent à partir du nom de la variable dans un playbook
- inclure toute la sortie depuis le nom de la variable (ici db_passwd) jusqu'à la fin de la donnée chiffrée



Exemple #1 : variable dans un playbook

```
---
- name: Test variable vault
  hosts: localhost
  connection: local
  gather_facts: no
  vars:
    db_passwd: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      33653831326366656138613535366634623434616461363363616165633862643934346230663539
      3132653166333863656334663336613665643666386366620a663233343537656439383433323334
      61393764343165353264646566363638623330616333313139363635336531396232613563346131
      6462303030333237370a363763666530636161363661316462353066373334373632353462366631
      6238

  tasks:
    - ansible.builtin.debug:
        msg: "{{ db_passwd }}"
```

```
$ ansible-playbook -l localhost --ask-vault-password ./vault_demo.yml
Vault password:

PLAY [Vault usage demo] *****
TASK [Display secret value2] *****
ok: [localhost] => {
    "msg": "My_Passw0rd"
}

PLAY RECAP *****
localhost                : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



Example #2: variable dans un fichier dédié

8

1. fichier `datas.yml` :
2. chiffrer avec : `$ ansible-vault encrypt datas.yml`
3. `datas.yml` devient un fichier chiffré

```
---  
db_passwd: 'password'
```

```
---  
- hosts: localhost  
  connection: local  
  gather_facts: no  
  
  tasks:  
    - name: Affiche donnée chiffrée  
      ansible.builtin.debug:  
        var: db_passwd
```

```
---  
- hosts: localhost  
  connection: local  
  gather_facts: no  
  vars_files:  
    - datas.yml  
  
  tasks:  
    - name: Affiche donnée chiffrée  
      ansible.builtin.debug:  
        var: db_passwd
```

```
---  
- hosts: localhost  
  connection: local  
  gather_facts: no  
  
  tasks:  
    - name: Charge un fichier de données  
      ansible.builtin.include_vars: datas.yml  
  
    - name: Affiche donnée chiffrée  
      ansible.builtin.debug:  
        var: db_passwd
```

```
ansible-playbook vault.yml \  
--ask-vault-password -e@datax.yml
```

```
ansible-playbook -l localhost vault.yml --ask-vault-password
```




- Afficher le prompt sur la ligne de commande :

```
ansible-playbook [...] --ask-vault-pass ou --ask-vault-password  
export ANSIBLE_ASK_VAULT_PASS=False|True disponible aussi
```

- En mettant la clef dans un fichier :

```
ansible-playbook [...] --vault-password-file=./mot_de_passe.txt  
ajouter quand même le fichier dans un éventuel .gitignore  
une variable d'environnement ANSIBLE_VAULT_PASSWORD_FILE disponible
```





Prévenir l'apparition d'un secret dans un log

- `no_log: true` pour que le secret ne soit ni loggué, ni affiché sur la console en cas de plantage d'une tâche

```
---  
- name: Utilisation no_log  
  hosts: all  
  gather_facts: no  
  
  tasks:  
    - name: No display  
      ansible.builtin.debug:  
        msg: "No hello world !"  
        no_log: true
```

- avant Ansible 2.4, une clef par playbook
- chaque fichier/variable devait utiliser la même clef
- vault IDs : identifiant de mot de passe ou fichier de mot de passe spécifique
 - doivent être définis dans ansible.cfg

```
[defaults]  
vault_identity_list = dev@./password_file1 , prod@./password_file2
```

```
ansible-vault encrypt --encrypt-vault-id dev file_to_encrypt
```

```
ansible-vault encrypt_string 'secret_data' --name 'db_passwd' --encrypt-vault-id dev
```

- peuvent être renseignés à la volée au déchiffrement

```
ansible-playbook playbook.yml --vault-id dev@prompt --vault-id prod@./password_file2
```

- dans le même fichier, on peut avoir différentes variables chiffrées avec des clefs différentes



Variables chiffrées et vault-ids dans un playbook

```
--
- name: Vault usage demo
  hosts: all
  gather_facts: false
  vars:
    - db_passwd: !vault |
      $ANSIBLE_VAULT;1.2;AES256;dev
      37303463336139323362623634303737316534326665353037326237656432316164633636356234
      6335633166386662633162303936613661643766343762330a316437336430303934623665303164
      62333961353930656165656638313536383561363135343338323937666664613339313266353032
      6430613966623963320a326636313663336332303837376339356236323232373335323633306562
      61623163623133643634383035633231613366623837656130343936616430323537303332636636
      6433336562656230373863653232616562306338323634346265
    - prod_password: !vault |
      $ANSIBLE_VAULT;1.2;AES256;prod
      66353934323061316130363130366663303130633338343132393466303433306161613234306330
      3533396331646164396630663239623062386666643430370a666531323162626464306165396166
      31633235656265636164326232613365613830363665643234386164646261356639313363366264
      3561333330613231310a303336313935366564303736376536643439356535323438633266316261
      3636

  tasks:
    - name: Display secret value
      debug:
        msg: "{{ db_passwd }}"
        when: db_passwd is defined

    - name: Display secret value 2
      debug:
        msg: "{{ prod_password }}"
        when: prod_password is defined
```



Merci

