Tips and tricks

Probably the best way to do this - assuming that you can't use the NOPASSWD solution provided by scottod is to use Mircea Vutcovici's solution in combination with Ansible vault.

For example, you might have a playbook something like this:

```
- hosts: all

  vars_files:
    - secret

  tasks:
    - name: Do something as sudo
      service: name=nginx state=restarted
      sudo: yes
```

Here we are including a file called secret which will contain our sudo password.

We will use ansible-vault to create an encrypted version of this file:

`ansible-vault create secret`

This will ask you for a password, then open your default editor to edit the file. You can put your ansible_sudo_pass in here.

e.g.: secret:

ansible_sudo_pass: mysudopassword

Save and exit, now you have an encrypted secret file which Ansible is able to decrypt when you run your playbook. Note: you can edit the file with ansible-vault edit secret (and enter the password that you used when creating the file)

The final piece of the puzzle is to provide Ansible with a --vault-password-file which it will use to decrypt your secret file.

Create a file called vault.txt and in that put the password that you used when creating your secret file. The password should be a string stored as a single line in the file.

From the Ansible Docs:

`.. ensure permissions on the file are such that no one else can access your key and do not add your key to source control`

Finally: you can now run your playbook with something like

```
ansible-playbook playbook.yml -u someuser -i hosts --sudo --vault-password-file=vault.txt
```

The above is assuming the following directory layout:

```
.
|_ playbook.yml
|_ secret
|_ hosts
|_ vault.txt
```

You can read more about Ansible Vault here: https://docs.ansible.com/playbooks_vault.html

---

Revision #3
Created 2026-04-01 17:14:19 CEST by Philip
Updated 2026-04-13 19:24:56 CEST by Philip