PROJET ANSIBLE

DÉPLOIEMENT D'UN HÔTE ANSIBLE

Pablo Pardo

2024-03-04

CONFIGURER UNE MACHINE CENTOS 9 AVEC ANSIBLE (HÔTE)

INSTALLATION DE VISUAL STUDIO

Il est possible d'utiliser nano pour éditer les fichiers de configuration de Ansible mais Visual Code permet une meilleure visualisation et hiérarchisation. Ceci facilite la configuration des fichiers Ansible. Pour plus d'informations sur l'installation, vous pouvez consulter la page suivante :

https://code.visualstudio.com/docs/setup/linux

Pour commencer nous devons créer un fichier à la racine de notre ***HOME*** ou un chemin dans lequel le dossier va contenir nos fichiers (ex. : répertoire homeserver).

mkdir homeserver



Ensuite, nous importons le package de Visual Code :

sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc

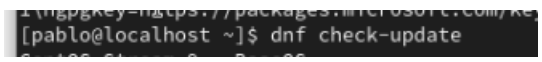


sudo sh -c 'echo -e "[code]\nname=Visual Studio Code\nbaseurl=https://packages.microsoft.com/yumrepos/vscode\nenabled=1\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" > /etc/yum.repos.d/vscode.repo'

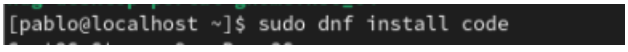


Faire un **check-update** avant l'installation pour s'assurer que tout est correcte avec la version et vérifier s'il ne manque pas des fichiers.

dnf check-update



sudo dnf install code



INSTALLATION DE ANSIBLE

Voici les commandes pour procéder à l'installation de Ansible dans CentOs9:

dnf -y install ansible-core # installation du programme Ansible

```
[pablo@localhost ~]$ dnf -y install ansible-core
```

ansible –version # vérification de la version de Ansible

```
[pablo@localhost ~]$ ansible -version
```

CRÉATION D'UNE CLÉ SSH

Nous devons créer une clé SSH. Cette clé nous permettra des manipulations avec Ansible sans la demande fastidieuse de rentrer un mot de passe. De plus, sans la clé, Ansible ne pas faire de demande aux machines cibles.

Ansible ne pousse pas le code s'il y a une demande de mot de passe par la cible.

La clé permettra de créer la connexion nécessaire pour pouvoir pousser les configurations sur le client CentOS. Pour commencer nous allons créer la clé SSH et la placer dans un dossier pour pouvoir avoir accéder à celle-ci le moment venu.

ssh-keygen -o -a 100 -t ed25519 -f /home/pablo/.ssh/homeserver -C pablodiablo@gmail.com

```
[pablo@localhost ~]$ ssh-keygen -o -a 100 -t ed25519 -f /home/pablo/.ssh/homeserver -C pablodiablo@gmail.com
Generating public/private ed25519 key pair.
Created directory '/home/pablo/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/pablo/.ssh/homeserver
Your public key has been saved in /home/pablo/.ssh/homeserver.pub
The key fingerprint is:
SHA256:z1RMw+oUHMAopmQqvlwmZbyO68vXuGlFPsr7hGfudvI pablodiablo@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|       o.o.oo    |
|  o o . . oo..   |
| +.o .     oo    |
|o .+ .    o.     |
|o o +   So.      |
| o +.+   +.      |
|. O.=+.   o      |
|.+ B** .         |
|.==+++E          |
+----[SHA256]-----+
```

Ensuite nous allons envoyer la clé vers le serveur client (adresse IP du client cible) dans laquelle nous voulons pousser les configurations avec Ansible. Cette commande s'exécute sur le client hôte et non sur la cible.

ssh-copy-id -i /home/pablo/.ssh/homeserver pablo@10.10.10.144

```
+----[SHA256]-----+
[pablo@localhost ~]$ ssh-copy-id -i /home/pablo/.ssh/homeserver pablo@10.10.10.144
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/pablo/.ssh/homeserver.pub"
The authenticity of host '10.10.10.144 (10.10.10.144)' can't be established.
ED25519 key fingerprint is SHA256:yuR5ISqx0Li4YPLD9Po91rdymrVD75KudESrkHyOkYg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
pablo@10.10.10.144's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'pablo@10.10.10.144'"
and check to make sure that only the key(s) you wanted were added.
```

Nous allons tester le SSH et nous assurer lors de cette connexion que le mot de passe n'est pas demandé.

ssh pablo@10.10.10.144

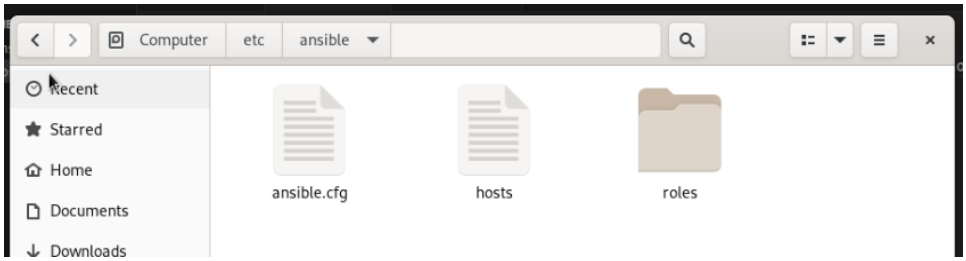
```
[pablo@localhost ~]$ ssh pablo@10.10.10.144
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sun Mar  3 09:50:04 2024
[pablo@localhost ~]$ systemctl enable --now cockpit.socket
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-unit-files ====
Authentication is required to manage system service or unit files.
Authenticating as: pablo
```

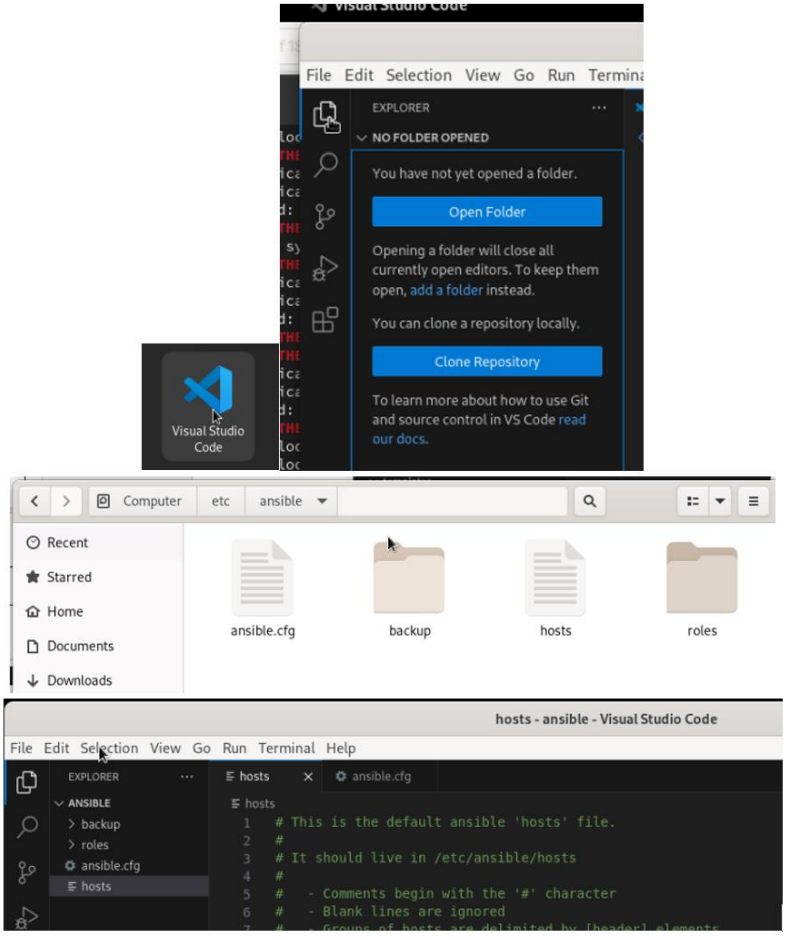
Si vous vous connectez sans erreur, alors la clé est bien configurée. Dans le cas contraire, recommencer l'étape de création et de l'exportation de la clé SSH.

CRÉATION DE LA CONNEXION ENTRE ANSIBLE ET LE CLIENT

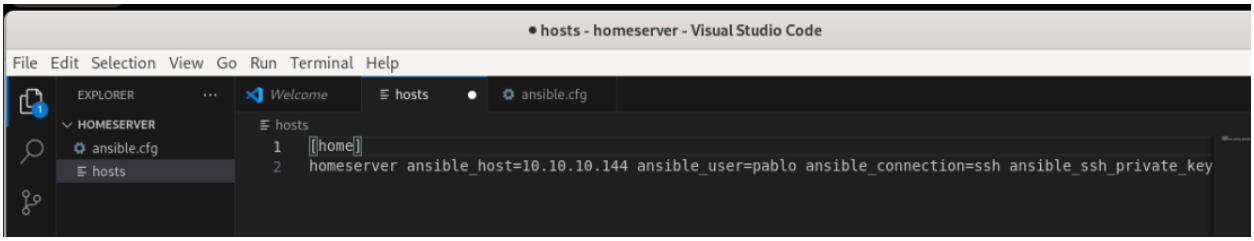
Par défaut, les fichiers de configuration de Ansible (***ansible.cfg*** et ***hosts***) se retrouvent sur le ***/etc/ansible***



Pour pousser les configurations pour les différents services nous devons configurer ces fichiers. Pour ce faire, nous démarrons Visual Studio et nous nous positions sur le chemin ***/home/pablo/homeserver*** (assurez-vous de faire une copie de sauvegarde avant de modifier les fichiers).



Nous allons créer le fichier ***hosts*** dans lequel nous allons configurer les différentes cibles (serveurs) qui recevrons les configurations de Ansible pour les différents services.

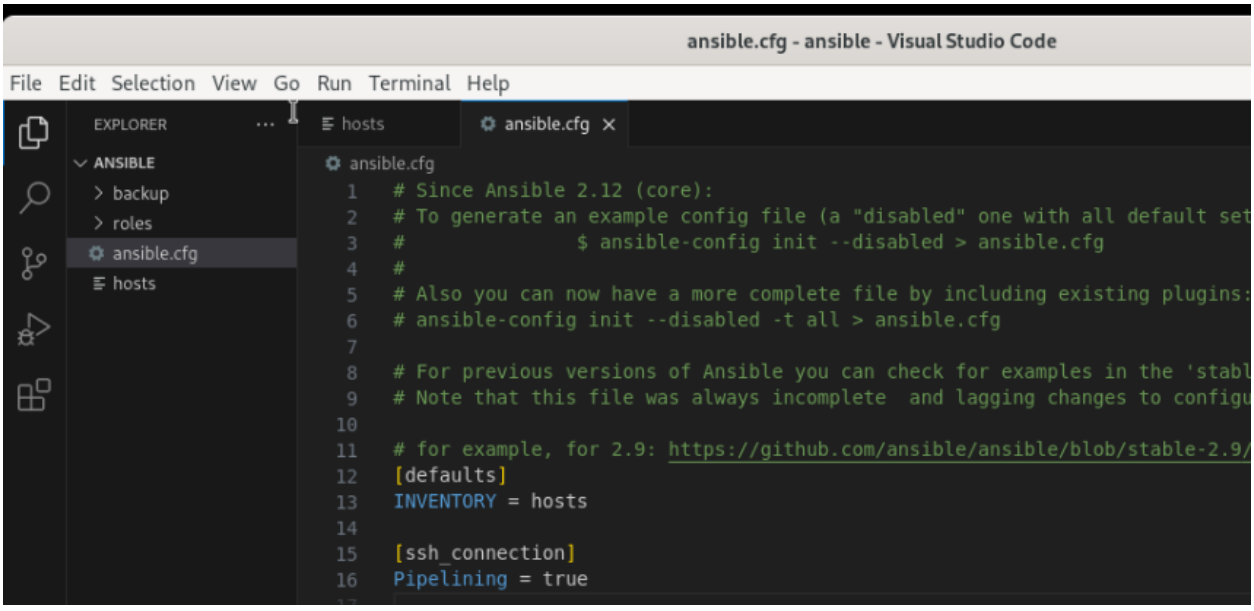


Dans le fichier nous allons écrire :

[home]
homeserver ansible_host=10.10.10.144 ansible_user=pablo ansible_connection=ssh ansible_ssh_private_key_file=/home/pablo/.ssh/homeserver

ansible_host= IP du client que nous voulons nous connecter
ansible_user= nom utilisateur que nous voulons utiliser pour nous connecter au client.
ansible_connection= Le type de connections avec lequel on se connecte au client
ansible_ssh_private_key_file= Le chemin de notre clé priver créer précédemment

Pour continuer, nous allons créer le fichier **ansible.cfg** pour placer l'inventaire de nos hosts et activer le pipeline de SSH.



Nous allons marquer dans le fichier **ansible.cfg** :

[defaults]
INVENTORY = hosts

[ssh_connection]
Pipelining = true

Une fois l'étape terminée nous pouvons tester la connexion de Ansible sur le client cible avec la commande :

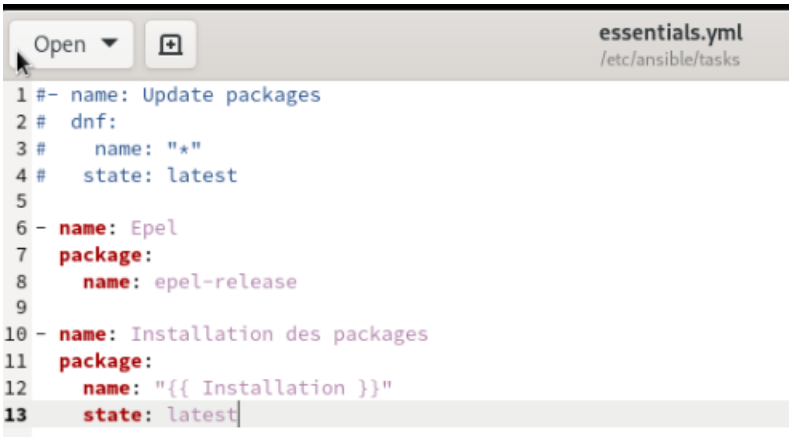
ansible all -m ping

Voici le résultat si nous avons réussi la connexion :

```
[pablo@localhost ansible]$ ansible all -m ping
homeserver | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

INSTALLER LES UPDATES, SERVICES ET EPEL-RELEASE SUR LE CLIENT

Nous allons créer un groupe de tâche (group task) et créer le fichier **essentials.yml** dans le répertoire **/home/pablo/homeserver/tasks** qui va être responsable de pousser du code pour mettre à jour tous les packages, installer ***epel-release*** et installer les services avec notre variable "***Installation***" que nous allons configurer.



Voici les paramètres du fichier :

Name = nom donné à la configuration que nous voulons créer

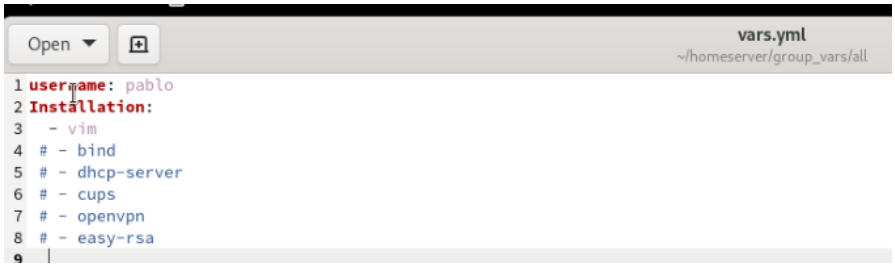
dnf/package = la commande utiliser pour installer des packages ou services.

Name = le nom du service à mettre à jour (nous avons marqué **" * "** pour mettre à jour tous les services)

State = la version du service ou du package à installer (nous avons marqué **latest** pour mettre la dernière mise à jour)

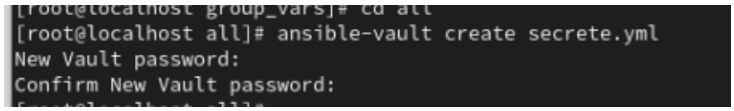
Nous avons mis comme **nom '' {{ Installation }} "**. Ceci est une variable qui possède tous les services que nous voulons installer.

Pour créer la variable **nom '' {{ Installation }} "**nous allons créer le dossier **"group_vars/all"** dans le dossier **/home/pablo/homeserver.** Le **/all** est là pour que les variables soient applicables à tous les hosts.

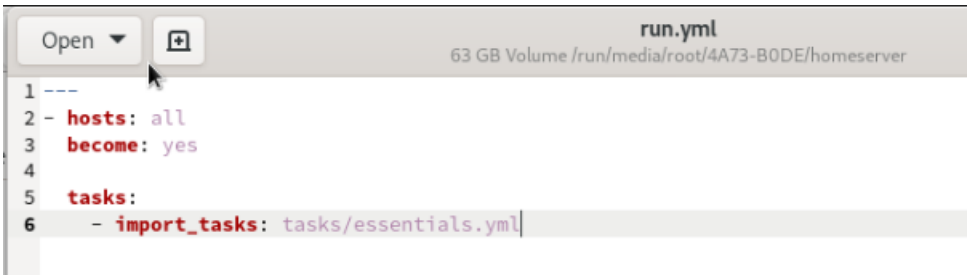


Nous allons configurer la variable **username** avec le nom avec lequel on veut faire notre connexion SSH. Ici, nous avons utilisé **pablo** comme **username.** Dans la variable **'' {{ Installation }} ''** nous avons mis les services vi et les autres services en commentaires (ex. : bind, dhcp-server,…)

Pour continuer, nous allons utiliser la commande **ansible-vault create secrete.yml** dans le ficher **group_vars/all** et configurer un **mot de passe.** Ce mot de passe sera demandé par Ansible pour sécuriser le push vers les machines pour les services que nous avons mis dans le **/all**

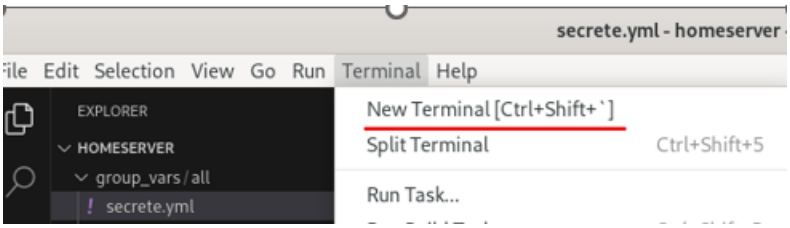


Finalement, nous allons créer le fichier **run.yml** que nous allons utiliser pour pousser les configurations que nous venons de faire.



Hosts = les machines que nous voulons pousser le code.
Become : yes = c'est comme le sudo mais pour ansible
Import_tasks = le chemin du code que nous volons pousser sur le client

Pour pouvoir exécuter les commandes (pousser le code) nous devons ouvrir un **terminal** à l'intérieur de Visual code comme la figure ci-dessous.



Le terminal est le même que celui par défaut dans Linux. Il est donc possible d'ouvrir une fenêtre ***Shell*** est de taper la commande :

ansible-playbook run.yml -K --ask-vault-pass

Le mot de passe demandé est le mot de passe crée avec la commande **ansible-vault create secrete.yml**

```
[root@localhost ansible]# ansible-playbook run.yml -K --ask-vault-pass
BECOME password:
Vault password:

PLAY [all] *********************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************
The authenticity of host '10.10.10.144 (10.10.10.144)' can't be established.
ED25519 key fingerprint is SHA256:yuR5ISqx0Li4YPLD9Po91rdymrVD75KudESrkHyOkYg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
ok: [homeserver]

TASK [Epel] ********************************************************************************************************
changed: [homeserver]

TASK [Installation des packages] ***********************************************************************************
ok: [homeserver]

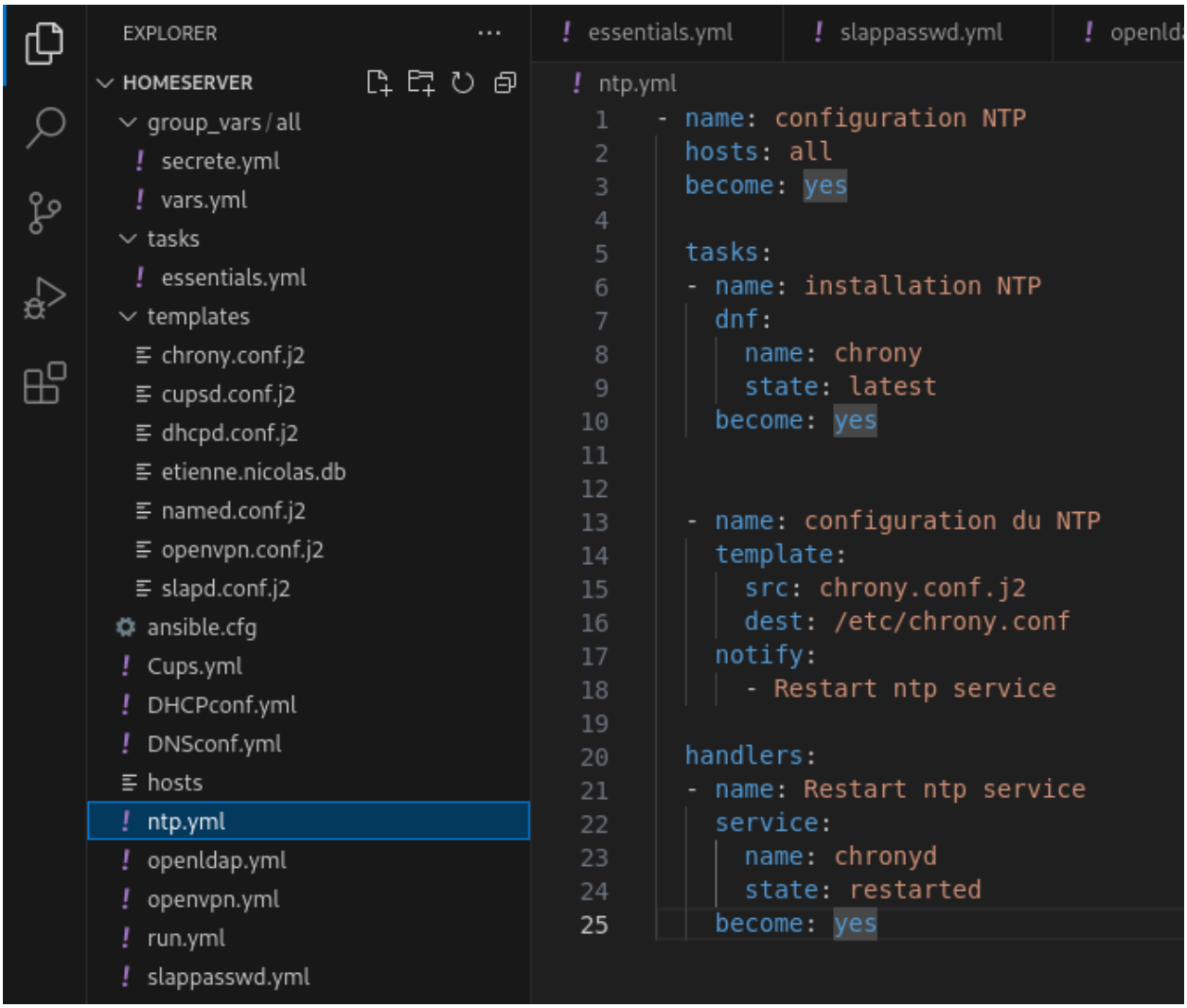
PLAY RECAP *********************************************************************************************************
homeserver                 : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@localhost ansible]#
```

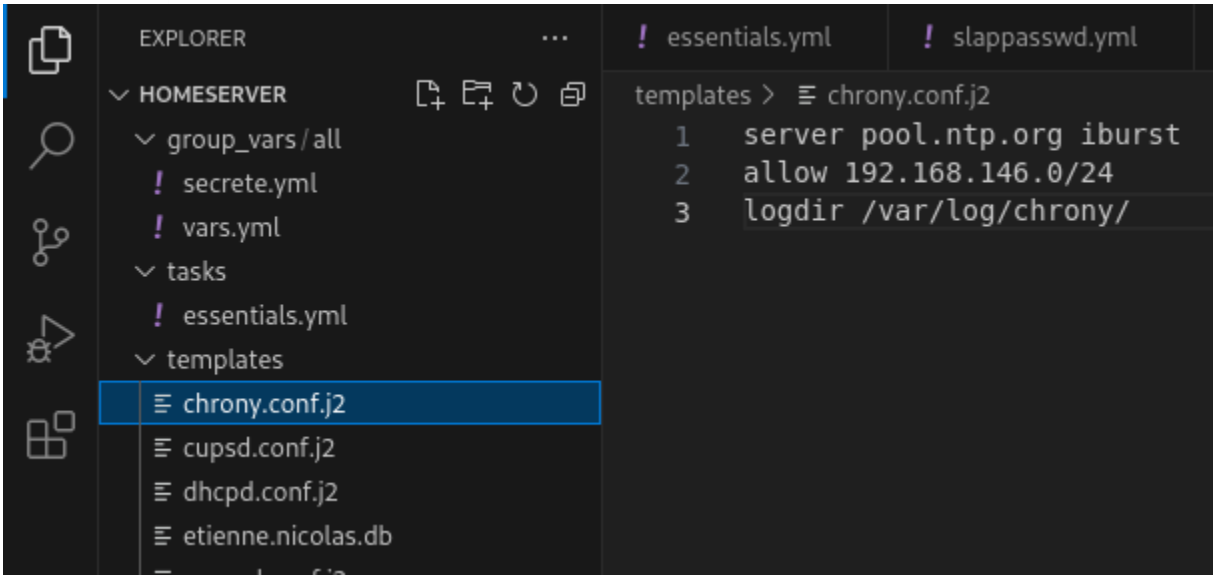
On peut voir sur la capture d'écran les tâches qui ont été complétés. Ansible renvoi **Ok** si une tâche a été accomplie et **changed** si le fichier que nous voulons **modifier** a été modifié.

LE SERVICE NTP

Pour le dernier service voici les configurations pour pousser le **service NTP** :



Voici les configurations pour le fichier **chrony.conf.j2** **(attention à l'adressage IP)**



Après avoir terminé la configuration du NTP nous allons la pousser avec la commande :

ansible-playbook ntp.yml -K --ask-vault-pass