# Ansible Helec

## 3. Installation et utilisation d'Ansible

sous la vm packer debian accessible sous moodle, on recupere le script d'ansible.

```
git pull https://github.com/pushou/tp3automatisation.git/
```

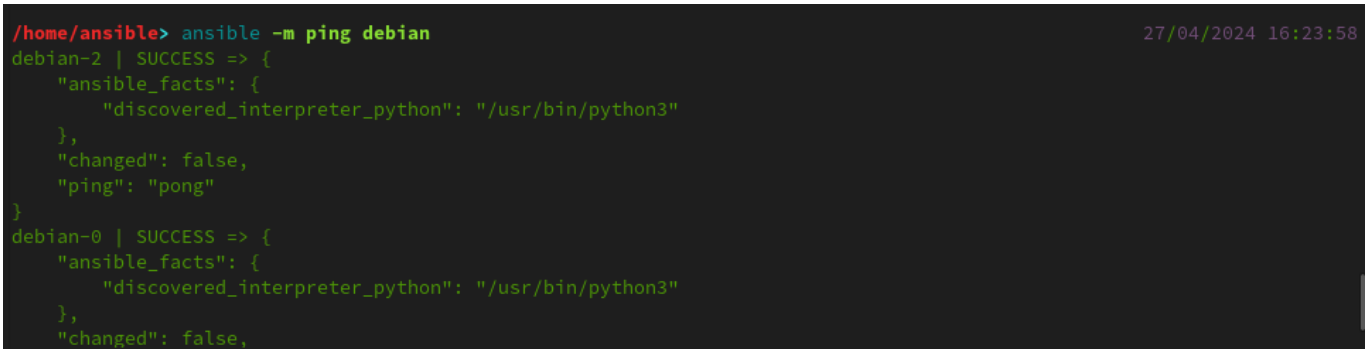
Puis on lance le script create-cont.sh qui va nous generer un espace de travail complet pour ansible.

```
./create-cont.sh
```

## 4 Prise en main d'Ansible

on va verifier avec la commande ansible et le module ping que les cibles sont vivantes

```
ansible -m ping debian
```




Les collection ansible-galaxy :

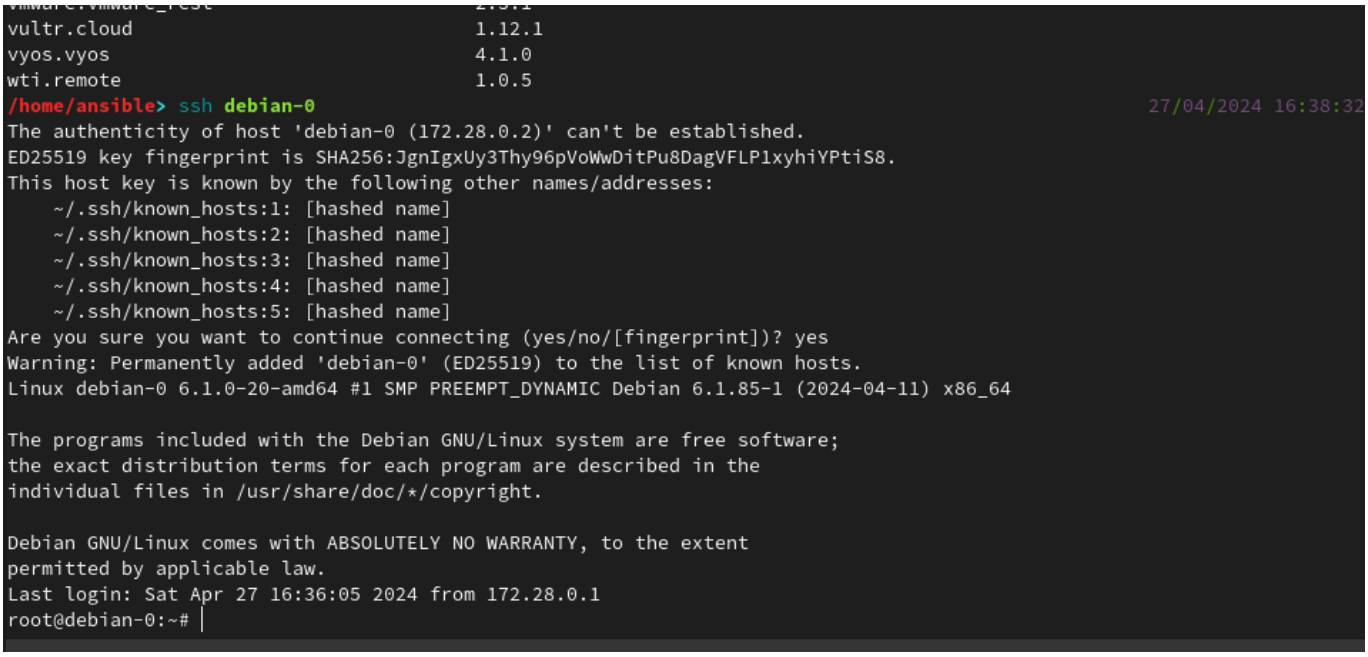
```
ansible-galaxy collection list
```

```
/home/ansible> ansible-galaxy collection list                                    27/04/2024 16:38:18

# /root/.ansible/collections/ansible_collections
Collection                              Version
--------------------------------------- -------
nokia.srlinux                           0.4.0

# /usr/local/lib/python3.11/dist-packages/ansible_collections
Collection                              Version
--------------------------------------- -------
amazon.aws                              7.4.0
ansible.netcommon                       5.3.0
ansible.posix                           1.5.4
ansible.utils                           2.12.0
ansible.windows                         2.3.0
arista.eos                              6.2.2
awx.awx                                 23.9.0
azure.azcollection                      1.19.0
check_point.mgmt                        5.2.3
chocolatey.chocolatey                   1.5.1
cisco.aci                               2.8.0
cisco.asa                               4.0.3
cisco.dnac                              6.13.1
cisco.intersight                        2.0.7
```

```
debian-2 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if59        UP             172.30.0.2/16 fe80::42:acff:fe1e:2/64
debian-3 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if61        UP             172.31.0.2/16 fe80::42:acff:fe1f:2/64
debian-4 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if63        UP             192.168.16.2/20 fe80::42:c0ff:fea8:1002/64
rocky-1 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if47        UP             172.19.0.2/16 fe80::42:acff:fe13:2/64
rocky-0 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if45        UP             172.18.0.2/16 fe80::42:acff:fe12:2/64
rocky-2 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if49        UP             172.20.0.2/16 fe80::42:acff:fe14:2/64
rocky-3 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if51        UP             172.21.0.2/16 fe80::42:acff:fe15:2/64
rocky-4 | CHANGED | rc=0 >>
lo               UNKNOWN        127.0.0.1/8 ::1/128
eth0@if53        UP             172.22.0.2/16 fe80::42:acff:fe16:2/64
```

ansible utilise le protocole ssh on peut donc se connecter aux containeurs avec la commande ssh

```
ssh debian-0 # de 0 a 4
ssh rocky-0
```




1. on va utilisez l'option host_key_checking=false du fichier ansible.cfg pour eviter les warnings désagréable
2. on utilise ensuite ansible-console pour lancer la commande :

```
ansible-console

ip -br a
```

3. Le proctole réseau utilisé par ansible et le protocole ssh
4. Analyser le fonctionnement de la commande ansible avec l'option -vvv. Que pouvez vous en déduire du fonctionnement d'Ansible ? expliquer comment Ansible peut être "agentless"?

-vvv permet d'afficher le multi debug

Création d'un groupe container qui regroupe tout les containeurs actuelle :

```
ansible-inventory --list all
ansible-navigator
```

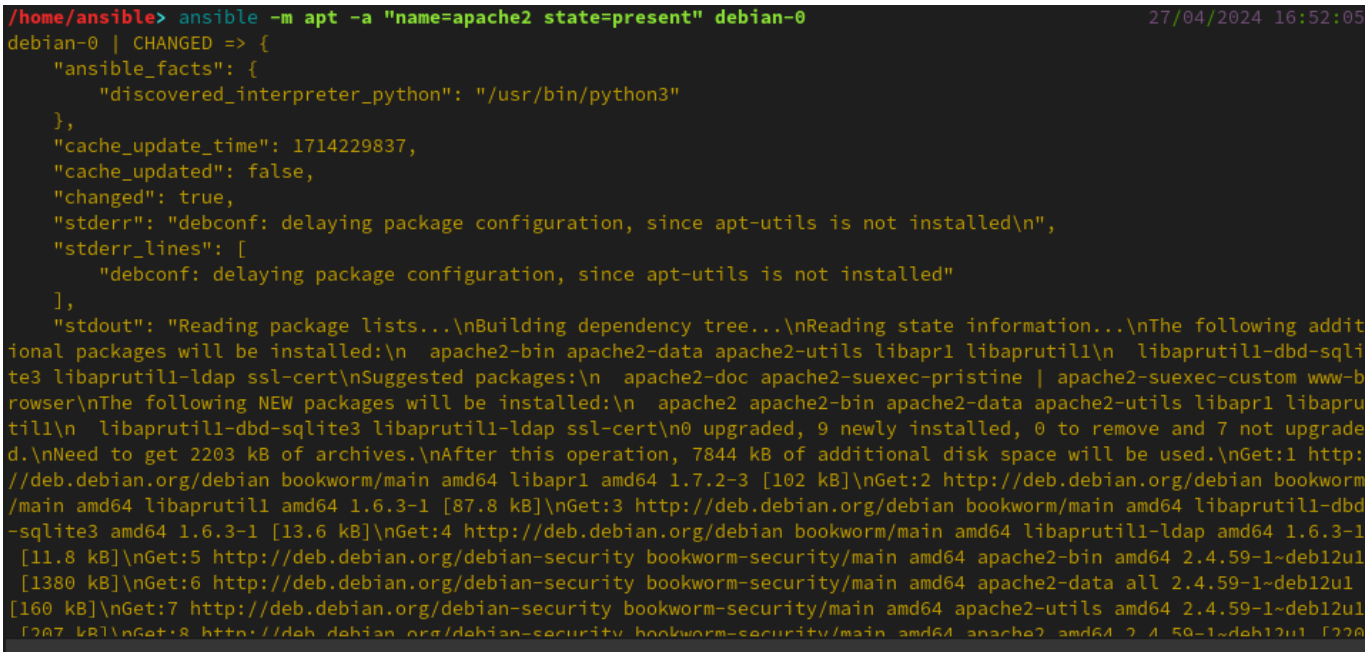
```
            "ansible_port": 3223,
            "ansible_ssh_private_key_file": "~/.ssh/id_ed25519"
        },
        "rocky-4": {
            "ansible_host": "127.0.0.1",
            "ansible_port": 3224,
            "ansible_ssh_private_key_file": "~/.ssh/id_ed25519"
        }
    }
},
"all": {
    "children": [
        "ungrouped",
        "linux",
        "eos"
    ]
},
"debian": {
    "hosts": [
        "debian-0",
        "debian-1",
        "debian-2",
        "debian-3",
        "debian-4"
    ]
```

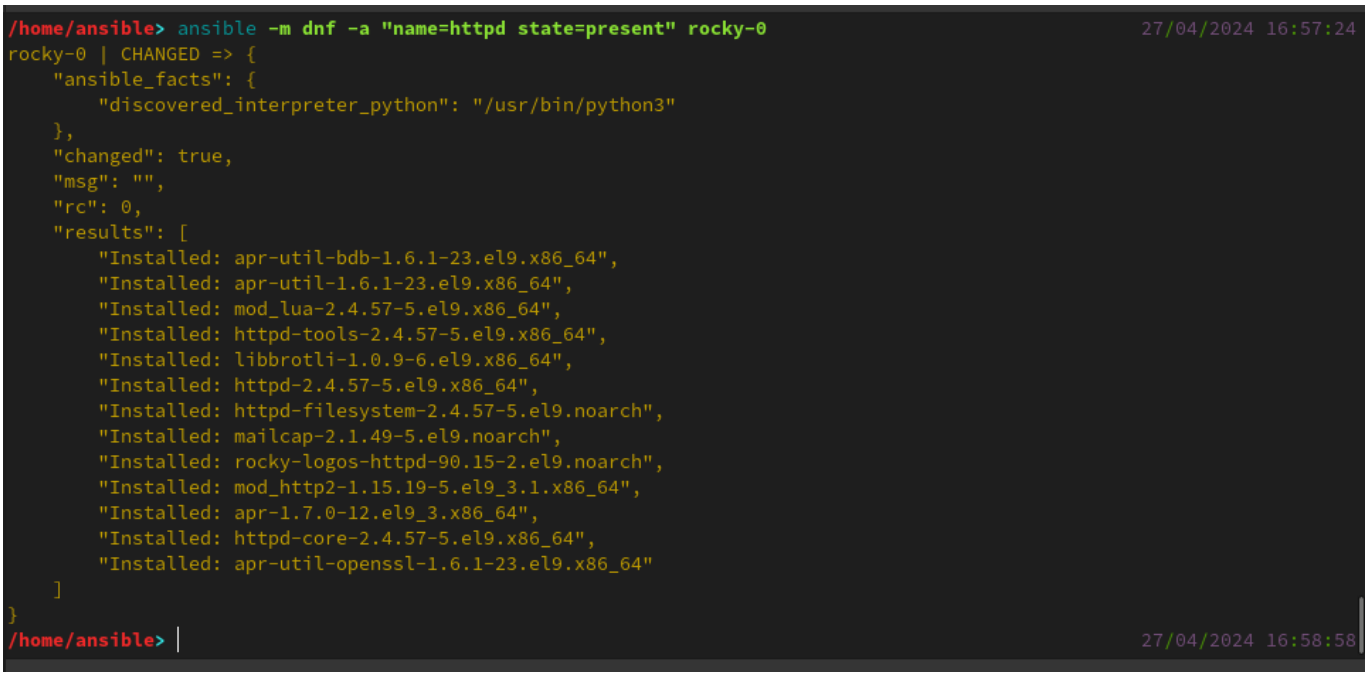

le groupe ungrouped regroupe tout les noeuds sans groupe.

## 4.2 Installation d'apache via les modules dnf et apt ansible core :

1. Sur 2 des containeurs créer installez un serveur web apache2 et Centos a l'aide de la commande ansible et des modules Ansible de gestion des paquets apt et dnf. Le package s'appelle httpd sous Centos.

```
ansible -m apt -a "name=apache2 state=present" debian-0
ansible -m dnf -a "name=httpd state=present" rocky-0
```




2. Le playbook suivant installera Apache et PHP, un fichier info.php et qui démarrera le serveur web Apache sur vos containers Debian.

```
---
- hosts: debian-0
    tasks:
    - name: Installer Apache
    ansible.builtin.apt:
        name: apache2
        state: present
        update_cache: true


    - name: Installer Php7
    ansible.builtin.apt:
        name: libapache2-mod-php8.2
```

```
        state: present

    - name: Démarrer le service Apache
    ansible.builtin.service:
        name: apache2
        state: started
        enabled: true

    - name: Copier le fichier phpinfo
    ansible.builtin.copy:
        src: info.php
        dest: /var/www/html/index.php
        owner: www-data
        group: www-data
        mode: 0664
```

Pour l'integrer il faut ecrire

```
ansible-playbook apache.yml
```

3. Avant de le lancer avec la commande ansible-playbook, vérifiez le bon enchainement des taches avec les options --check et --diff (pas d'éxécution réelle), --list-hosts, --list-tasks de la commande ansible-playbook. Testez aussi la conformité de votre playbook avec le "linteur" ansible-lint

```
ansible-playbook apache.yml --check --diff
ansible-playbook apache.yml --list-hosts
ansible-playbook apache.yml --list-tasks
ansible-lint apache.yml
```

```
        "Installed: httpd-2.4.57-5.el9.x86_64",
        "Installed: httpd-filesystem-2.4.57-5.el9.noarch",
        "Installed: mailcap-2.1.49-5.el9.noarch",
        "Installed: rocky-logos-httpd-90.15-2.el9.noarch",
        "Installed: mod_http2-1.15.19-5.el9_3.1.x86_64",
        "Installed: apr-1.7.0-12.el9_3.x86_64",
        "Installed: httpd-core-2.4.57-5.el9.x86_64",
        "Installed: apr-util-openssl-1.6.1-23.el9.x86_64"
    ]
}
/home/ansible> ansible-playbook apache.yml --check --diff                    27/04/2024 16:58:58
ERROR! the playbook: apache.yml could not be found
/home/ansible> ansible-playbook apache.yml --list-hosts                  1 27/04/2024 17:07:03
ERROR! the playbook: apache.yml could not be found
/home/ansible> touch apache.yml                                          1 27/04/2024 17:07:12
/home/ansible> ansible-playbook apache.yml --check --diff                  27/04/2024 17:07:22
ERROR! Empty playbook, nothing to do: /home/ansible/apache.yml
/home/ansible> ansible-playbook apache.yml --list-hosts                  4 27/04/2024 17:07:24
ERROR! Empty playbook, nothing to do: /home/ansible/apache.yml
/home/ansible> ansible-playbook apache.yml --list-tasks                  4 27/04/2024 17:07:28
ERROR! Empty playbook, nothing to do: /home/ansible/apache.yml
/home/ansible> ansible-lint apache.yml                                   4 27/04/2024 17:07:37

Passed: 0 failure(s), 0 warning(s) on 1 files. Last profile that met the validation criteria was 'production'.
/home/ansible>                                                             27/04/2024 17:07:50
```

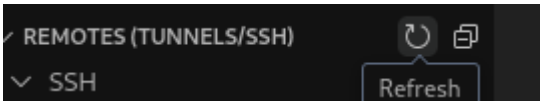
```
/home/ansible> ansible-playbook apache.yml --list-hosts                  2 27/04/2024 18:10:17

playbook: apache.yml

  play #1 (debian-0): debian-0  TAGS: []
    pattern: ['debian-0']
    hosts (1):
      debian-0
/home/ansible> ansible-playbook apache.yml --list-tasks                    27/04/2024 18:20:18

playbook: apache.yml

  play #1 (debian-0): debian-0  TAGS: []
    tasks:
      Installer Apache  TAGS: []
      Installer Php7    TAGS: []
      Démarrer le service Apache        TAGS: []
      Copier le fichier phpinfo TAGS: []
      Reload Apache     TAGS: []
/home/ansible>                                                             27/04/2024 18:20:24
```

```
/home/ansible> ansible-lint apache.yml                                     27/04/2024 18:20:2
WARNING  Listing 8 violation(s) that are fatal
name[play]: All plays should be named.
apache.yml:2

yaml[indentation]: Wrong indentation: expected 6 but found 7
apache.yml:6

yaml[indentation]: Wrong indentation: expected 6 but found 7
apache.yml:12

yaml[indentation]: Wrong indentation: expected 6 but found 7
apache.yml:17

yaml[indentation]: Wrong indentation: expected 6 but found 7
apache.yml:23

yaml[octal-values]: Forbidden implicit octal value "0664"
apache.yml:27

args[module]: Unsupported parameters for (basic.py) module: tags. Supported parameters include: arguments, enabled, nam
e, pattern, runlevel, sleep, state (args). (warning)
apache.yml:29 Task/Handler: Reload Apache

yaml[indentation]: Wrong indentation: expected 6 but found 7
```

1. Créer une tache nommée "reload" qui "reload" le server web 10. Tagué cette tache "relance" puis utilisez le tag pour n’exécuter que cette tache dans le playbook.

```
---
- hosts: debian
    tasks:
    - name: Reload Apache
    ansible.builtin.service:
        name: apache2
        state: reloaded
        tags: relance
```

```
ansible-playbook apache.yml --tags relance
```




5. A quoi servent les tags "never" et "always" (testez) ?

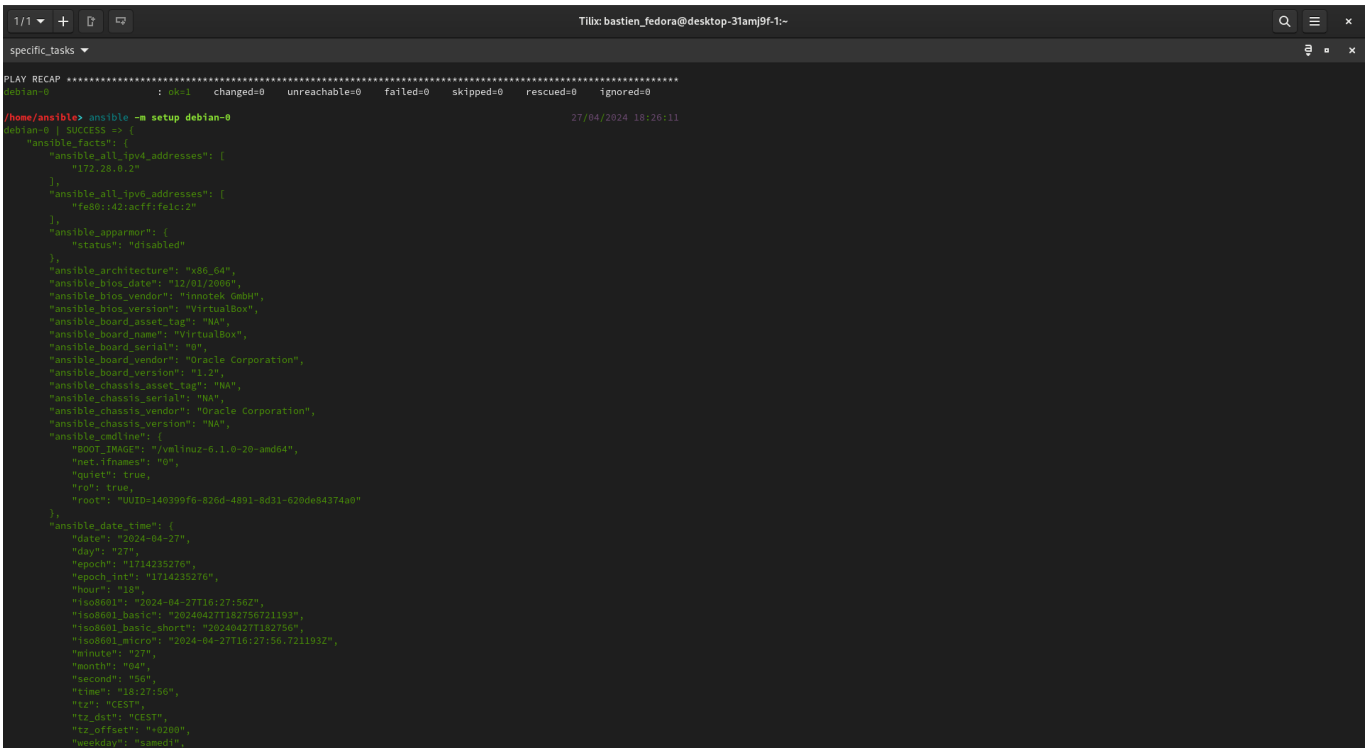
never et always sont des tags qui permettent de ne jamais executer une tache ou de toujours l'executer.

```
ansible-playbook apache.yml --tags never
ansible-playbook apache.yml --tags always
```

```
/home/ansible> ansible-playbook apache.yml --tags never                    27/04/2024 18:25:0

PLAY [debian-0] ***************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [debian-0]

PLAY RECAP ********************************************************************************************************
debian-0                   : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

/home/ansible> ansible-playbook apache.yml --tags always                   27/04/2024 18:25:1

PLAY [debian-0] ***************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [debian-0]

PLAY RECAP ********************************************************************************************************
debian-0                   : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

/home/ansible>                                                             27/04/2024 18:26:1
```

6. Lister les facts à l'aide du module setup. A quoi servent ces facts ?

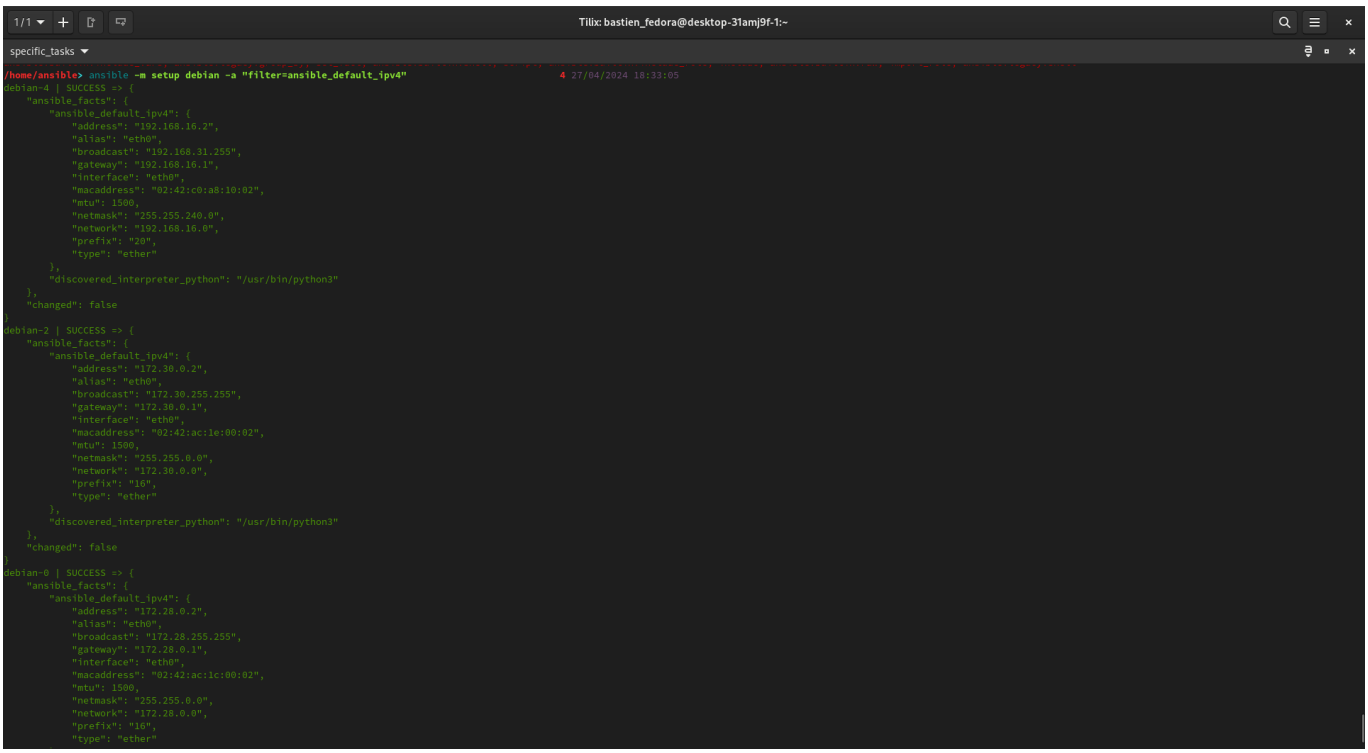
```
ansible -m setup debian-0
```




C'est facts permettent de connaitre les informations sur la machine cible.

7. Récuperez les facts contenant les adresses IP de tous les containers via un filtre.

```
ansible -m setup debian -a "filter=ansible_default_ipv4"
```




8. Utilisez les facts et les templates jinja2 afin de modifier la directive fichier ports.conf sous Debian pour que lors de l'installation du module apache2 de Debian n'écoute que sur l'IP interne du container. Vous utiliserez les mots clefs notify et handler pour relancer le service apache dans le playbook.

```
---
- hosts: debian
    tasks:
    - name: Modifier le fichier ports.conf
    ansible.builtin.template:
        src: ports.conf.j2
        dest: /etc/apache2/ports.conf
        owner: root
        group: root
        mode: 0644
    notify: Reload Apache

    handlers:
    - name: Reload Apache
    ansible.builtin.service:
        name: apache2
        state: reloaded
```

```
Listen {{ ansible_default_ipv4.address }}
```

```
ansible-playbook ports.yml
```