

Peertube

PLAYBOOK

Service Web (contient la conf de NGINX et PEERTEUBE):

Service BDD PostgreSQL:

pg_hba.conf

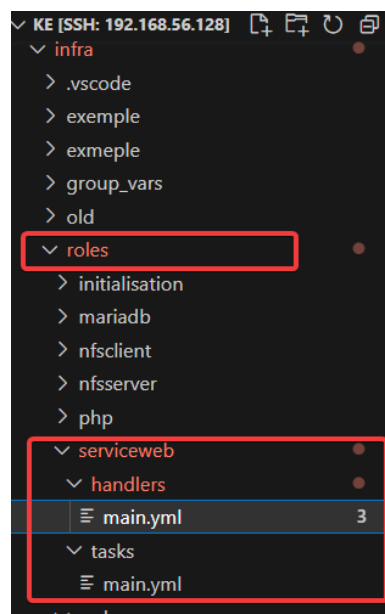
postgresql.conf

Playbook pour la création de la base

PLAYBOOK

Service Web (contient la conf de NGINX et PEERTEUBE):

J'ai commencé à utiliser les handlers, je vais appeler mon serveur : serviceweb
Voici capture d'écran :



le contenu de main.yml dans tasks :

```

- name: Installer Nginx
  package:
    name: nginx
    state: present

- name: Téléchargement de l'archive de PeerTube
  unarchive:
    src: "https://github.com/Chocobozzz/PeerTube/releases/download/v{{ peertube_version }}/peertube-v{{ peertube_version }}.zip"
    dest: "{{ chemin_peertube | realpath | dirname }}"
    remote_src: yes
    creates: "{{ chemin_peertube | realpath }}"

- name: Envoi de la configuration nginx pour PeerTube
  template:
    src: exemple/nginx-peertube.conf.j2
    dest: /etc/nginx/sites-available/peertube
  notify:
    - Reload Nginx

- name: Désactiver le site par défaut de Nginx
  file:
    path: /etc/nginx/sites-enabled/default
    state: absent
  notify:
    - Reload Nginx

- name: Activer le site PeerTube (lien symbolique)
  file:
    src: /etc/nginx/sites-available/peertube
    dest: /etc/nginx/sites-enabled/peertube
    state: link
  notify:
    - Reload Nginx

- name: Changement des permissions pour PeerTube
  file:

```

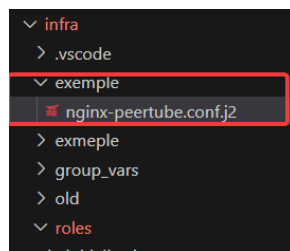
```
path: "{{ chemin_peertube | realpath }}"
recurse: true
owner: www-data
group: www-data
```

Le contenu de main.yml dans handlers :

```
---

- name: Reload Nginx
  service:
    name: nginx
    state: reloaded
```

On voit que dans task cela appelle un fichier de conf (dans mon cas ça va chercher vers exemple/nginx-peertube.conf.j2) , le voici :



```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name {{ peertube_nomduserveur }};

    location / {
        proxy_pass http://{{ ansible_host }}:{{ peertube_port }};
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_f
```

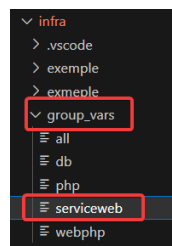
```

orwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}

access_log /var/log/nginx/peertube_access.log;
error_log /var/log/nginx/peertube_error.log;
}

```

Les variables utilisées sont donc les suivantes :



```

peertube_version: 6.0.3
peertube_nomduserveur: peerteube
peertube_port: 9000
chemin_peertube: /var/www/peertube

```

Service BDD PostgreSQL:

Pour commencer il faut préparer 2 fichiers à envoyer avec le playbook, c'est les fichiers **pg_hba.conf** et **postgresql.conf** situé à l'emplacement :
/etc/postgresql/{{ version }}/main/

pg_hba.conf

```

local    all                all
trust
local    all                all
trust

```

host	all	all	127.0.0.1/32
trust			
host	all	all	:::1/128
trust			

⚠ C'est une configuration non sécurisée mais je n'ai pas trouvé mieux, ces lignes signifient qu'on utilise TOUS UTILISATEURS distant à se connecter à la base de données
Notamment la 1ère ligne qui correspond à l'utilisateur SUPERUSER

postgresql.conf

⚠ Je n'ai pas encore réussi à faire en sorte que des variables s'appliquent pour ces 2 fichiers de conf, je comprends pas pourquoi

```
data_directory = '/var/lib/postgresql/15/main'
hba_file = '/etc/postgresql/15/main/pg_hba.conf'
ident_file = '/etc/postgresql/15/main/pg_ident.conf'
external_pid_file = '/var/run/postgresql/15-main.pid'
listen_addresses = '*'
port = 5432
max_connections = 100
unix_socket_directories = '/var/run/postgresql'
ssl = on
ssl_cert_file = '/etc/ssl/certs/ssl-cert-snakeoil.pem'
ssl_key_file = '/etc/ssl/private/ssl-cert-snakeoil.key'
shared_buffers = 128MB
dynamic_shared_memory_type = posix
max_wal_size = 1GB
min_wal_size = 80MB
log_line_prefix = '%m [%p] %q%u@%d '
log_timezone = 'Europe/Paris'
cluster_name = '15/main'
datestyle = 'iso, dmy'
timezone = 'Europe/Paris'
```

```
lc_messages = 'fr_FR.UTF-8'
lc_monetary = 'fr_FR.UTF-8'
lc_numeric = 'fr_FR.UTF-8'
lc_time = 'fr_FR.UTF-8'
default_text_search_config = 'pg_catalog.french'
include_dir = 'conf.d'
```

Voici le main.yml pour le role du playbook :

```
---

- name: Installer les packages PostgreSQL
  package:
    name:
      - postgresql
    state: present

- name: Installer psycopg2
  package:
    name:
      - python3-psycopg2
    state: present

- name: Démarrer le service PostgreSQL
  service:
    name: postgresql
    state: started
    enabled: yes

- name: Remplacer le fichier pg_hba.conf par le fichier p
ersonnalisé
  copy:
    src: exemple/confpostgr
    dest: /etc/postgresql/15/main/pg_hba.conf
  notify:
    - Reboot PostgreSQL
```

```

- name: Remplacer le fichier postgresql.conf par le fichier personnalisé
  copy:
    src: exemple/postgrlisten
    dest: /etc/postgresql/15/main/postgresql.conf
  notify:
    - Reboot PostgreSQL

```

Et voici le handler qui va avec :

```


---

- name: Reboot PostgreSQL
  service:
    name: postgresql
    state: restarted

```

Playbook pour la création de la base

Et voici le playbook pour créer la BDD peertube

 On ne peut pas exécuter les 2 playbooks ensemble car il faut absolument que le service postgresql redemarre avant de pouvoir créer la base

```

---

- name: Créer la base de données pour PeerTube
  postgresql_db:
    name: "{{ peertube_db_name }}"
    state: present
  notify: Reboot PostgreSQL

- name: Créer l'utilisateur PostgreSQL pour PeerTube
  postgresql_user:

```

```
db: "{{ peertube_db_name }}"
name: "{{ peertube_db_user }}"
password: "{{ peertube_db_password }}"
role_attr_flags: CREATEDB
notify: Reboot PostgreSQL
```

- name: Donner les privilèges à l'utilisateur PostgreSQL pour PeerTube

```
postgresql_privs:
  db: "{{ peertube_db_name }}"
  privs: ALL
  type: database
  login_user: "{{ peertube_db_user }}"
  roles: "{{ peertube_db_user }}"
  notify: Reboot PostgreSQL
```

et voici son handler :

```
---
```

```
- name: Reboot PostgreSQL
  service:
    name: postgresql
    state: restarted
```