

# Système de gestion centralisée des utilisateurs avec FreeIPA et authentification réseau via FreeRADIUS

## Introduction :

Dans le cadre de ce TP, nous avons mis en place un système de gestion centralisée des utilisateurs en utilisant **FreeIPA**, une solution open source qui combine les services LDAP, Kerberos, DNS et de gestion des certificats pour assurer l'authentification, l'autorisation et l'administration des utilisateurs. Pour compléter cette architecture, nous avons intégré **FreeRADIUS** afin de permettre une authentification réseau, notamment pour les connexions **Wi-Fi** utilisant les protocoles **WPA2-Enterprise avec PEAP ou TTLS**. Ce système assure une meilleure sécurité et une gestion unifiée des accès pour les utilisateurs du réseau.

## I. Mettre en place un système de gestion centralisée des utilisateurs avec FreeIPA :

### Étape 1 : Installation des outils de base et configuration du nom d'hôte

```
[aziz@localhost ~]$ sudo dnf install -y epel-release
```

Cette commande permet d'installer plusieurs outils essentiels :

- **wget** : pour télécharger des fichiers via HTTP/HTTPS,
- **vim** : un éditeur de texte pour modifier des fichiers de configuration,
- **net-tools** : pour avoir accès à des outils réseau classiques (ifconfig, etc.),
- **firewalld** : pour la gestion du pare-feu.

Ensuite, on configure le nom d'hôte de la machine :

```
[aziz@localhost ~]$ hostnamectl hostname ipa.tp.local
```

Cette commande définit le nom d'hôte permanent de la machine comme **ipa.tp.local**. C'est une étape cruciale, car FreeIPA repose sur une configuration DNS cohérente et un nom d'hôte entièrement qualifié (FQDN).

### Étape 2 : Activation du pare-feu

```
[aziz@localhost ~]$ sudo systemctl enable --now firewalld
```

Ici, on active le service **firewalld** et on le démarre immédiatement. Il est nécessaire d'avoir un pare-feu fonctionnel pour autoriser uniquement les ports nécessaires à FreeIPA.

### Étape 3 : Installation du serveur FreeIPA

```
[aziz@localhost ~]$ sudo systemctl enable --now firewalld  
[aziz@localhost ~]$ sudo dnf install -y ipa-server
```

Cette commande installe le paquet principal **FreeIPA Server**, qui regroupe plusieurs services :

- un serveur **LDAP** (389 Directory Server),
- un service d'authentification **Kerberos**,
- un serveur **DNS** (optionnel),
- un système de gestion des utilisateurs, groupes et politiques d'accès.

#### Étape 4 : Configuration du fichier /etc/hosts

```
[aziz@localhost ~]$ sudo nano /etc/hosts
```

Dans cette étape, nous avons modifié le fichier /etc/hosts afin d'assurer une résolution correcte du nom d'hôte local. Ce fichier permet d'associer manuellement des adresses IP à des noms de domaine.

Voici un extrait du fichier après modification :

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.163.15.254 ipa.tp.local ipa
```

#### Étape 5 : Configuration du serveur FreeIPA

```
[aziz@localhost ~]$ sudo ipa-server-install
```

Cette commande lance le script d'installation interactif du serveur FreeIPA. Ce script configure tous les composants nécessaires au bon fonctionnement du système : LDAP, Kerberos, DNS (optionnel), certificat SSL, etc.

#### Étape 6 : Validation des paramètres de configuration

Lors de l'exécution de la commande `sudo ipa-server-install`, un récapitulatif des paramètres saisis est affiché avant que l'installation ne commence réellement :

```
The IPA Master Server will be configured with:
Hostname:      ipa.tp.local
IP address(es): 10.163.15.254
Domain name:   tp.local
Realm name:    TP.LOCAL

The CA will be configured with:
Subject DN:    CN=Certificate Authority,O=TP.LOCAL
Subject base:  O=TP.LOCAL
Chaining:      self-signed

Continue to configure the system with these values? [no]: yes
```

#### Explication :

- **Hostname** : le nom d'hôte FQDN de la machine (défini avec `hostnamectl`) est reconnu correctement.
- **IP address** : l'adresse IP statique de la machine, nécessaire pour les services réseau.

- **Domain name** : le domaine DNS utilisé pour les utilisateurs, services, et le serveur lui-même.
- **Realm name** : nom du domaine Kerberos, utilisé pour l'authentification centralisée (toujours en majuscules).
- **CA (Certificate Authority)** : FreeIPA installe une autorité de certification interne, ici **auto-signée** (chaining: self-signed), qui génère les certificats SSL utilisés par les services.

## Étape 7 : Fin de l'installation

Une fois l'installation terminée, le message suivant s'affiche :

```
The ipa-server-install command was successful
```

Ce message confirme que l'installation du serveur FreeIPA s'est déroulée sans erreur. Tous les services nécessaires (LDAP, Kerberos, HTTP, Certificate Authority, etc.) sont maintenant installés, configurés et démarrés.

## Étape 8 : Configuration du pare-feu pour FreeIPA

### Explication :

Cette commande permet d'autoriser de façon permanente les services suivants via le pare-feu :

```
[aziz@localhost ~]$ sudo firewall-cmd --permanent --add-service={freeipa-ldap,freeipa-ldaps,dns}
[sudo] password for aziz:
success
[aziz@localhost ~]$ sudo firewall-cmd --reload
success
```

- freeipa-ldap : pour permettre les communications LDAP non chiffrées (port 389),
- freeipa-ldaps : pour LDAP sécurisé via SSL (port 636),
- dns : si le serveur FreeIPA gère aussi un service DNS (port 53).
- Ensuite, le rechargement du pare-feu applique ces nouvelles règles
- Résultat : success

## Étape 9 : Vérification de la configuration de FreeIPA

Après l'installation, on vérifie que le serveur FreeIPA fonctionne correctement à l'aide des commandes suivantes :

1. Authentification Kerberos

```
kinit admin
```

Cette commande permet d'obtenir un ticket Kerberos pour l'utilisateur admin, qui est le compte administrateur de FreeIPA.

2. Recherche d'utilisateurs avec IPA

```
ipa user-find
```

Résultat obtenu :

```
[aziz@localhost ~]$ kinit admin
Password for admin@TP.LOCAL:
[aziz@localhost ~]$ ip user-find
Object "user-find" is unknown, try "ip help".
[aziz@localhost ~]$ ipa user-find
-----
1 user matched
-----
  User login: admin
  Last name: Administrator
  Home directory: /home/admin
  Login shell: /bin/bash
  Principal alias: admin@TP.LOCAL, root@TP.LOCAL
  UID: 1901800000
  GID: 1901800000
  Account disabled: False
-----
Number of entries returned 1
```

### Explication :

Cette commande interroge le serveur FreeIPA pour afficher la liste des utilisateurs existants. Ici, on retrouve l'utilisateur admin créé automatiquement lors de l'installation du serveur.

Ce résultat confirme que :

- Le serveur LDAP de FreeIPA fonctionne,
- La base de données des utilisateurs est accessible,
- L'environnement est prêt pour l'ajout d'autres utilisateurs ou l'intégration avec d'autres services (comme FreeRADIUS).

### Etape 10 : Vérifier l'interface Web

Ouvre : <https://192.168.248.128>

Identifiants : admin / passer123

## II. Ajout d'utilisateurs et intégration de FreeRADIUS

Après avoir mis en place et vérifié le bon fonctionnement du serveur FreeIPA, la prochaine étape consiste à :

1. Ajouter de nouveaux utilisateurs dans la base LDAP de FreeIPA.
2. Intégrer le serveur FreeIPA à FreeRADIUS, afin de permettre l'**authentification réseau centralisée** (notamment pour un accès Wi-Fi via WPA2-Enterprise avec PEAP ou TTLS).

**LDAP** (Lightweight Directory Access Protocol) est un **protocole de communication** utilisé pour interroger et modifier des services d'annuaire, comme FreeIPA, Active Directory ou OpenLDAP.

### Étape 1 : Installation des paquets FreeRADIUS et LDAP

```
[root@ipa ~]# dnf install -y freeradius freeradius-ldap openldap-clients
```

## Explication :

- freeradius : installe le serveur FreeRADIUS, qui fournit un service d'authentification, d'autorisation et de comptabilité (AAA) pour les connexions réseau.
- freeradius-ldap : plugin LDAP permettant à FreeRADIUS d'interroger un annuaire LDAP (comme FreeIPA) pour authentifier les utilisateurs.
- openldap-clients : outils en ligne de commande pour interagir manuellement avec un annuaire LDAP (par exemple pour tester une recherche d'utilisateur via ldapsearch).

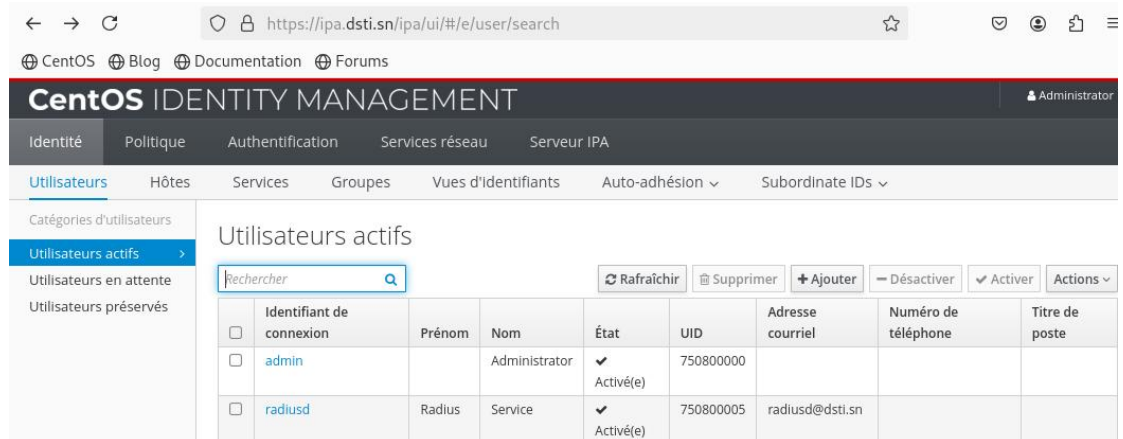
## Etape 2 : Créer un compte de liaison LDAP pour RADIUS (dans FreeIPA)

Tout d'abors on se connection dans la page Web puis on vas dans "Identité" → "Utilisateurs" → "Ajouter"

On remplis :

- **Username** : radiusd
- **First Name** : Radius
- **Last Name** : Service
- **Password** : Radius123! (ou autre mot de passe fort)

Après on sauvegarde l'utilisateur



Identifiant de connexion	Prénom	Nom	État	UID	Adresse courriel	Numéro de téléphone	Titre de poste
<input type="checkbox"/> admin		Administrator	✓ Activé(e)	750800000			
<input type="checkbox"/> radiusd	Radius	Service	✓ Activé(e)	750800005	radiusd@dsti.sn		

## Explication :

L'utilisateur radiusd est un **compte de service**, c'est-à-dire un utilisateur technique utilisé par FreeRADIUS pour **s'authentifier auprès de l'annuaire LDAP de FreeIPA**.

Il doit donc exister dans la base FreeIPA, avec un mot de passe fort, et avoir les droits nécessaires pour interroger les entrées utilisateur.

## Étape 3 : Configuration du module LDAP dans FreeRADIUS

```
[root@ipa ~]# nano /etc/raddb/mods-available/ldap
```

Cette commande ouvre le fichier de configuration du module LDAP de FreeRADIUS. C'est dans ce fichier que l'on précise **comment** FreeRADIUS doit se connecter à l'annuaire LDAP (ici, FreeIPA).

```
GNU nano 5.6.1 /etc/raddb/mods-available/ldap
ldap {
    server = "ipa.dsti.sn"
    identity = 'uid=radiusd,cn=users,cn=accounts,dc=dsti,dc=sn'
    password = 'Radius123!'
    base_dn = 'cn=accounts,dc=dsti,dc=sn'

    user {
        base_dn = "${..base_dn}"
        filter = "(uid=%{%{Stripped-User-Name}:-%{User-Name}})"
        scope = "sub"
    }

    group {
        base_dn = "${..base_dn}"
        filter = "(objectClass=posixGroup)"
        scope = "sub"
        name_attribute = cn
        membership_attribute = member
    }

    timeout = 10
    timelimit = 10
    net_timeout = 1
    start_tls = no
}
```

### Explication :

on indique

- l'adresse du serveur FreeIPA (ou LDAP). Ce nom doit correspondre au nom DNS ou IP du serveur.
- Identifiants utilisés par FreeRADIUS pour se connecter à l'annuaire LDAP.
- Il s'agit d'un **compte de service** (radiusd) défini dans FreeIPA.
- DN (Distinguished Name) racine utilisé pour rechercher les utilisateurs et groupes dans FreeIPA.
- Détermine comment FreeRADIUS recherche un utilisateur dans LDAP.
- Il utilise le champ uid et extrait le nom d'utilisateur reçu lors de la connexion.
- Indique que la recherche s'applique à tous les sous-niveaux du base\_dn.
- Recherche tous les groupes de type posixGroup.
- Champ utilisé pour déterminer si un utilisateur est membre d'un groupe.
- Délais en secondes pour les requêtes LDAP.

- Indique que la connexion ne se fait **pas via TLS**. Il faudrait mettre yes pour sécuriser la communication LDAP.

#### Étape 4 : Activation du module LDAP dans FreeRADIUS

```
[magui@ipa ~]$ sudo ln -s /etc/raddb/mods-available/ldap /etc/raddb/mods-enabled/
```

Cette commande crée un **lien symbolique** vers le fichier de configuration LDAP.

#### Étape 5 : Configuration du fichier default

```
[magui@ipa ~]$ sudo nano /etc/raddb/sites-enabled/default
```

Ce fichier contient plusieurs sections importantes. Voici ce qu'il faut modifier pour intégrer LDAP (FreeIPA) à l'authentification.

##### 1. Section authorize { ... }

Dans cette section, **décommente** ou **ajoute**

##### **Ldap**

Cela permet à FreeRADIUS d'interroger le serveur LDAP (FreeIPA) lors de la phase d'autorisation (quand un utilisateur tente de se connecter).

##### 2. Section authenticate { ... }

Dans cette section, **décommente** ou **ajoute** aussi :

```
Auth-Type LDAP {
    ldap
}
```

Cela indique que si FreeRADIUS détecte que l'authentification doit se faire avec LDAP, il utilise le module ldap activé précédemment.

#### Étape 6 : Déclaration des clients dans clients.conf

```
[magui@ipa ~]$ sudo nano /etc/raddb/clients.conf
```

Ce fichier permet de définir **les équipements ou services autorisés à interroger le serveur FreeRADIUS**, comme un **point d'accès Wi-Fi**, un **commutateur réseau**, ou tout autre **client RADIUS**.

```
client localhost_ipv6 {
    ipv6addr      = ::1
    secret        = testing123
}
```

#### Explication :

- client localhost\_ipv6 : nom symbolique du client (ici un point d'accès Wi-Fi).
- Ipv6addr : adresse IP du client autorisé à interroger le serveur FreeRADIUS.
- secret : **clé partagée** entre le client (ex: hostapd) et le serveur RADIUS. Elle doit être identique des deux côtés



## Étape 7 : Ajout d'un utilisateur dans FreeIPA via la ligne de commande

```
[root@ipa ~]# kinit admin
Password for admin@DSTI.SN:
[root@ipa ~]# ipa user-add testuser --first=Test --last=User -password
```

Détails :

- `ipa user-add` : commande pour créer un nouvel utilisateur dans FreeIPA.
- `testuser` : nom de connexion (login) de l'utilisateur.
- `--first=Test` : prénom de l'utilisateur.
- `--last=User` : nom de famille de l'utilisateur.
- `--password` : invite l'administrateur à saisir un mot de passe sécurisé pour ce compte.

```
Utilisateur « testuser » ajouté
-----
Identifiant de connexion: testuser
Prénom: Test
Nom: User
Nom complet: Test User
Nom affiché: Test User
Initiales: TU
Répertoire personnel: /home/testuser
GECOS: Test User
Interpréteur de commande: /bin/sh
Nom principal: testuser@DSTI.SN
Principal alias: testuser@DSTI.SN
User password expiration: 20250519193730Z
Adresse courriel: testuser@dsti.sn
UID: 750800006
GID: 750800006
Mot de passe: True
Membre des groupes: ipausers
Clés Kerberos disponibles: True
```

L'utilisateur sera alors **immédiatement disponible pour l'authentification via FreeRADIUS.**

## Utilisateurs actifs

Rechercher

Rafraîchir

Supprimer

+ Ajouter

— Désactiver

✓ Activer

Actions

<input type="checkbox"/>	Identifiant de connexion	Prénom	Nom	État	UID	Adresse courriel	Numéro de téléphone	Titre de poste
<input type="checkbox"/>	admin		Administrator	✓ Activé(e)	750800000			
<input type="checkbox"/>	radiusd	Radius	Service	✓ Activé(e)	750800005	radiusd@dsti.sn		
<input type="checkbox"/>	testuser	Test	User	✓ Activé(e)	750800006	testuser@dsti.sn		

Affichage des entrées 1 à 3 sur 3.



## Étape 8 : Lancement de FreeRADIUS en mode debug

```
[root@ipa ~]# radiusd -X
```

Ce mode affiche **en temps réel** tout ce que le serveur fait : démarrage, chargement des modules, réception des requêtes, erreurs éventuelles, etc. C'est **le meilleur moyen de tester et diagnostiquer** si FreeRADIUS fonctionne correctement et si l'intégration LDAP (FreeIPA) est bien prise en compte.

## Étape 9 : Test d'authentification avec radtest

Décomposition de la commande :

- testuser : nom d'utilisateur que tu as créé dans FreeIPA.
- motdepasse : le mot de passe associé à testuser.
- 127.0.0.1 : adresse IP du serveur FreeRADIUS (ici en local).
- 0 : ID du NAS (Network Access Server), souvent 0 pour les tests.
- testing123 : **clé secrète partagée** entre le client (ici radtest) et FreeRADIUS (elle doit être identique à celle définie dans /etc/raddb/clients.conf).
- On obtient une réponse de type **Access-Accept** ☒ qui montres que les identifiants sont corrects et que LDAP (FreeIPA) répond bien

```
[root@ipa ~]# radtest testuser testpass localhost 0 testing123
Sent Access-Request Id 152 from 0.0.0.0:42354 to 127.0.0.1:1812 length 78
  User-Name = "testuser"
  User-Password = "testpass"
  NAS-IP-Address = 192.168.248.128
  NAS-Port = 0
  Cleartext-Password = "testpass"
Received Access-Accept Id 152 from 127.0.0.1:1812 to 127.0.0.1:42354 length 38
  Message-Authenticator = 0x4bc59ea64842018fc81de7fdd5a5b6c7
```

## Simulation d'un Connection a un reseau via freeradius :

### 1.Partie Serveur :

**Etape1** : installation des éléments nécessaire.

```
server@server-VMware:~$ sudo apt install hostapd iw wpasupplicant iproute2
```

### ETAPE 2 : Creation du WIFI virtuelle :

```
server@server-VMware:~$ sudo modprobe mac80211_hwsim radios=2
```

La commande **sudo modprobe mac80211 hwsim radios=2** est utilisée dans un environnement Linux pour charger un module noyau spécifique qui simule des interfaces Wi-Fi virtuelles.

**modprobe** : C'est un outil Linux qui permet d'ajouter ou de supprimer des modules noyau. Dans ce cas, il est utilisé pour charger un module.

**mac80211** : C'est le nom du module noyau qui fournit un framework pour les pilotes Wi-Fi sous Linux. Il est essentiel pour la gestion des interfaces sans fil.

**hwsim** : Ce paramètre active le sous-module `mac80211_hwsim`, qui est un simulateur matériel Wi-Fi. Il crée des interfaces Wi-Fi virtuelles pour des tests et des simulations.

**radios=2** : Ce paramètre spécifie que le simulateur doit créer deux interfaces Wi-Fi virtuelles. Cela permet de simuler un environnement avec deux appareils sans fil, ce qui est utile pour des tests de connectivité ou des expériences réseau.

Elle nous a permis de créer **wlan1** et **wlan0** pour notre simulation.

```
server@server-VMware:~$ iw dev
phy#1
    Interface wlan1
        ifindex 5
        wdev 0x100000001
        addr 02:00:00:00:01:00
        type managed
        txpower 20.00 dBm
        multicast TXQ:
            qsz-byt qsz-pkt flows drops marks overlmt hashcolt
x-bytes tx-packets
0
phy#0
    Unnamed/non-netdev interface
        wdev 0x2
        addr 42:00:00:00:00:00
        type P2P-device
        txpower 20.00 dBm
    Interface wlan0
        ifindex 4
```

### **Etape 3 :** Création de notre point d'accès.

Le fichier `/etc/hostapd/hostapd.conf` est un fichier de configuration pour **hostapd**, un outil Linux permettant de transformer une interface Wi-Fi (comme `wlan0`) en point d'accès (AP) ou en serveur d'authentification.

```
GNU nano 7.2 /etc/hostapd/hostapd.conf
interface=wlan0
ssid=Test-RADIUS
hw_mode=g
channel=6
auth_algs=1
wpa=2
wpa_key_mgmt=WPA-EAP
rsn_pairwise=CCMP

# Ajout des paramètres FreeRADIUS
ieee8021x=1
auth_server_addr=192.168.50.10      # ← Adresse IP de ton serveur FreeRADIUS
auth_server_port=1812
auth_server_shared_secret=testing123

[ Read 15 lines ]
```

### Explication de quelques paramètres :

**wpa=2**

Active **WPA2** (Wi-Fi Protected Access 2), un protocole de sécurité robuste.

**wpa\_key\_mgmt=WPA-EAP**

Configure la méthode d'authentification **WPA-Enterprise (EAP)**, qui repose sur un serveur RADIUS (comme FreeRADIUS) plutôt que sur une clé pré-partagée (PSK).

**auth\_server\_addr=192.168.50.10**

Adresse IP du serveur FreeRADIUS qui validera les identifiants des clients.

**auth\_server\_port=1812**

Port utilisé pour communiquer avec le serveur RADIUS (1812 est le port par défaut).

**auth\_server\_shared\_secret=testing123**

Clé secrète partagée entre **hostapd** et FreeRADIUS pour sécuriser les échanges.

### ETAPE 4 : Activation du point d'accès

```
server@server-VMware:~$ sudo hostapd /etc/hostapd/hostapd.conf
wlan0: RADIUS Authentication server 192.168.50.10:1812
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

**On voit que c'est écrit AP-ENABLED**

### 2.Partie 2 : Partie Client.

**wpa\_supplicant** est un outil essentiel pour gérer les connexions Wi-Fi sécurisées (WPA, WPA2, WPA3).

Il permet aux machines clientes de s'authentifier sur des réseaux Wi-Fi protégés (entreprises, hotspots).

```
ubuntu@ubuntu-VMware:~$ sudo dpkg -i wpasupplicant_2.10-21build4_amd64.deb
```

### Création d'un repertoire de configuration :

```
ubuntu@ubuntu-VMware:~$ sudo mkdir -p /etc/wpa_supplicant
```

Le fichier `/etc/wpa_supplicant/wpa_supplicant.conf` est utilisé pour configurer **wpa\_supplicant**, un outil qui gère les connexions Wi-Fi sécurisées sur Linux (notamment pour les réseaux **WPA-Enterprise**).

```
GNU nano 7.2 /etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
eapol_version=1
ap_scan=1
network={
    ssid="Test-RADIUS"
    key_mgmt=WPA-EAP
    eap=PEAP
    identity="usertest"
    password="passer123"
    phase2="auth=MSCHAPV2"
    ca_cert="/etc/ssl/certs/ca-certificates.crt"
}
```

[ Lecture de 12 lignes ]

Voici une explication détaillée de chaque section :

Le bloc `network={ ... }` contient les paramètres pour se connecter au réseau "**Test-RADIUS**" avec une authentification **PEAP/MSCHAPv2** (typique dans les entreprises/écoles) :

- **ssid="Test-RADIUS"**  
Nom du réseau Wi-Fi cible.

- **key\_mgmt=WPA-EAP**  
Méthode de gestion des clés : **WPA-Enterprise** (EAP).
- **eap=PEAP**  
Protocole EAP utilisé : **PEAP** (Protected EAP), une méthode sécurisée encapsulant EAP dans TLS.
- **identity="usertest"**  
Identifiant de l'utilisateur pour l'authentification.
- **password="passer123"**  
Mot de passe associé à l'identité.
- **phase2="auth=MSCHAPV2"**  
Méthode d'authentification interne à PEAP : **MSCHAPv2** (utilisé pour valider les identifiants).
- **ca\_cert="/etc/ssl/certs/ca-certificates.crt"**  
Chemin vers le certificat racine (CA) pour vérifier le serveur RADIUS.

```
ubuntu@ubuntu-VMware:~$ sudo wpa_supplicant -i wlan1 -c /etc/wpa_supplicant/wpa_supplicant.conf -d
wpa_supplicant v2.10
random: getrandom() support available
Successfully initialized wpa_supplicant
Initializing interface 'wlan1' conf '/etc/wpa_supplicant/wpa_supplicant.conf' driver 'default' ctrl_interface 'N/A' bridge 'N/A'
Configuration file '/etc/wpa_supplicant/wpa_supplicant.conf' -> '/etc/wpa_supplicant/wpa_supplicant.conf'
```