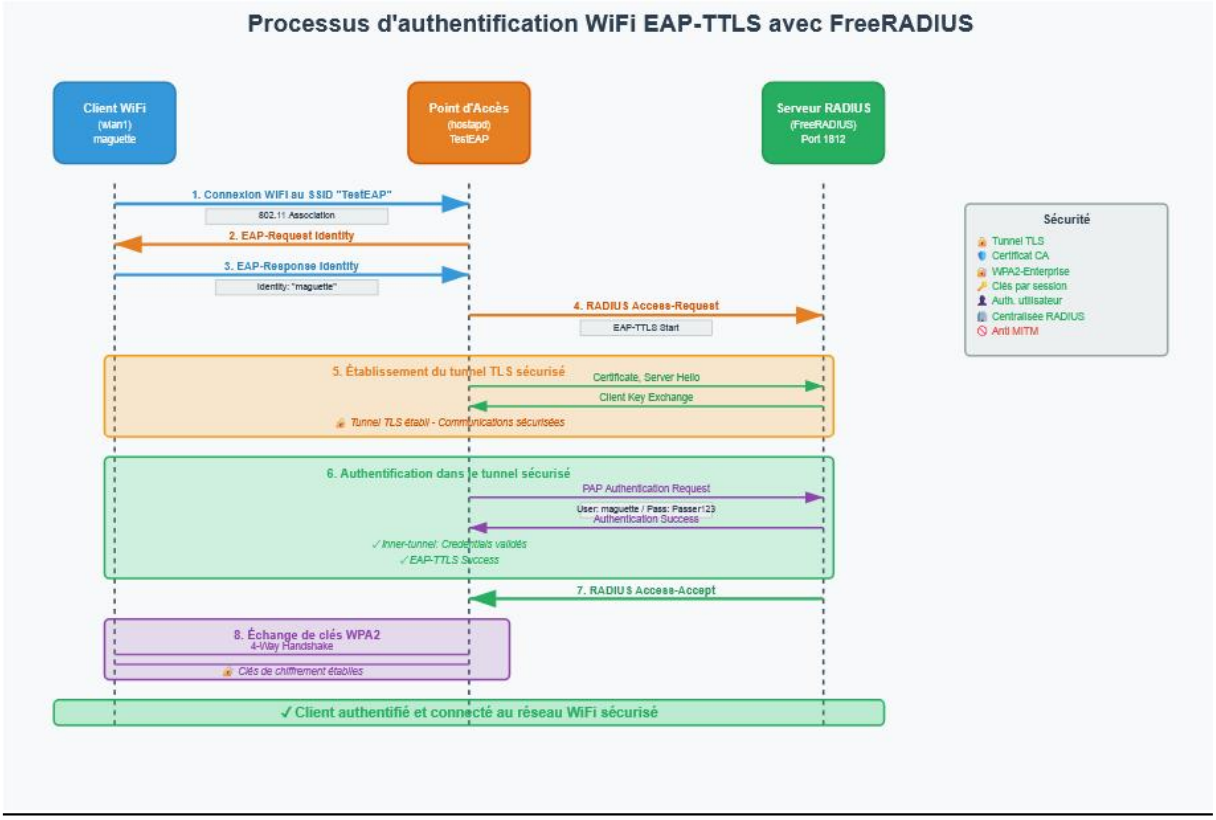
Implémentation de l’authentification centralisée Wi-Fi avec FreeRADIUS et LDAP

Introduction :

Dans le cadre de ce projet, nous avons mis en place une solution d’authentification centralisée pour un réseau Wi-Fi d’entreprise. L’objectif est de sécuriser l’accès au réseau sans fil en s’appuyant sur le protocole 802.1X, avec FreeRADIUS comme serveur d’authentification, et un annuaire LDAP comme source centralisée des identités utilisateurs. Ce système permet de gérer les accès de manière cohérente et sécurisée, tout en facilitant l’administration grâce à une base d’utilisateurs unique.

Schéma Explicatif :



---

1. Installation des outils réseau :

```
phone1@phone:~$ sudo apt install -y hostapd iw iproute2
```

Explication : Cette commande installe les packages essentiels pour créer et gérer un point d'accès **WiFi** :

- **hostapd** : daemon qui transforme une interface **WiFi** en **point d'accès (Access Point)**
- **iw** : outil moderne pour configurer les interfaces **WiFi** (remplace **iwconfig**)
- **iproute2** : suite d'outils pour la configuration réseau avancée **(ip, ss, etc.**

2. Suppression du pilote WiFi existant :

```
phone1@phone:~$ sudo modprobe -r mac80211_hwsim
```

Explication : Cette commande décharge le module **kernel mac80211_hwsim** de la mémoire. Ce module est un simulateur d'interface **WiFi** souvent utilisé pour les tests. Sa suppression permet de "**nettoyer**" l'environnement avant de reconfigurer les interfaces **WiFi**.

3. Rechargement du pilote avec configuration spécifique :

```
phone1@phone:~$ sudo modprobe mac80211_hwsim radios=2
phone1@phone:~$
```

Explication : Recharge le module **mac80211_hwsim** avec le paramètre **radios=2**, créant ainsi **deux interfaces WiFi virtuelles**. Cela permet de simuler :

- Une interface pour le point d'accès (AP)
- Une interface pour le client WiFi Cette configuration est idéale pour tester l'authentification WiFi en local.

Ces commandes préparent l'infrastructure de base nécessaire avant la configuration de FreeRADIUS et l'authentification WiFi.

4. Vérification des interfaces réseau créées :

```
phone1@phone:~$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
 group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mod
e DEFAULT group default qlen 1000
    link/ether 00:0c:29:a1:8f:d2 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
9: wlan0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN mode
DEFAULT group default qlen 1000
    link/ether ea:a2:e0:4e:8f:c6 brd ff:ff:ff:ff:ff:ff permaddr 02:00:00:00:00:0
0
10: wlan1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN mode
 DEFAULT group default qlen 1000
    link/ether 52:1e:69:27:66:3a brd ff:ff:ff:ff:ff:ff permaddr 02:00:00:00:01:0
0
11: hwsim0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT gr
oup default qlen 1000
    link/ieee802.11/radiotap 12:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
```

Explication : Cette commande affiche toutes les interfaces réseau disponibles. On peut observer :

- **lo** : interface de loopback (127.0.0.1)
- **ens33** : interface Ethernet physique (état UP)
- **wlan0** et **wlan1** : les deux interfaces WiFi virtuelles créées par **mac80211_hwsim**
  - **wlan0** : sera utilisée comme point d'accès (AP)
  - **wlan1** : sera utilisée pour simuler un client
- **hwsim0** : interface de contrôle du simulateur WiFi

5. Configuration du point d'accès WiFi (hostapd.conf) :

```
 GNU nano 7.2                  /etc/hostapd/hostapd.conf
# /etc/hostapd/hostapd.conf
interface=wlan0
driver=nl80211
ssid=TestEAP
hw_mode=g
channel=6

# WPA2-Enterprise / 802.1X
auth_algs=1
wpa=2
wpa_key_mgmt=WPA-EAP
rsn_pairwise=CCMP

# RADIUS backend
ieee8021x=1
auth_server_addr=192.168.200.10
auth_server_port=1812
auth_server_shared_secret=monsecret
```

Le fichier `/etc/hostapd/hostapd.conf` configure le comportement du point d'accès :

Configuration de base :

- **interface=wlan0** : utilise l'interface wlan0 comme AP
- **driver=nl80211** : pilote moderne pour les cartes WiFi compatibles mac80211
- **ssid=TestEAP** : nom du réseau WiFi visible
- **hw_mode=g** : utilise la bande 2.4GHz (802.11g)
- **channel=6** : canal WiFi utilisé

Configuration de sécurité WPA2-Enterprise :

- **auth_algs=1** : active l'authentification ouverte (nécessaire pour EAP)
- **wpa=2** : utilise WPA2
- **wpa_key_mgmt=WPA-EAP** : gestion des clés via EAP (pas de PSK)
- **rsn_pairwise=CCMP** : chiffrement AES-CCMP pour WPA2

Configuration RADIUS :

- **ieee8021x=1** : active 802.1X pour l'authentification d'entreprise
- **auth_server_addr**=192.168.200.10 : adresse IP du serveur RADIUS (FreeRADIUS)
- **auth_server_port**=1812 : port standard pour l'authentification RADIUS
- **auth_server_shared_secret=monsecret** : secret partagé entre l'AP et le serveur RADIUS

Cette configuration établit un pont entre le point d'accès WiFi et le serveur FreeRADIUS pour l'authentification des utilisateurs.

6. Lancement du point d'accès WiFi :

```
phone1@phone:~$ sudo hostapd /etc/hostapd/hostapd.conf
wlan0: RADIUS Authentication server 192.168.200.10:1812
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
```

Explication : Cette commande démarre le daemon **hostapd** avec la configuration précédente. Les messages de sortie confirment le bon fonctionnement :

- **RADIUS Authentication server 192.168.200.10:1812** : connexion établie avec le serveur RADIUS
- **interface state UNINITIALIZED->ENABLED** : l'interface wlan0 passe en mode actif
- **AP-ENABLED** : le point d'accès est maintenant opérationnel et diffuse le SSID **"TestEAP"**

Le point d'accès est maintenant prêt à recevoir des connexions et à rediriger les demandes d'authentification vers **FreeRADIUS**.

7. Configuration des clients RADIUS autorisés :

Le fichier **/etc/freeradius/3.0/clients.conf** définit quels équipements peuvent utiliser le serveur RADIUS :

```
  GNU nano 7.2                /etc/freeradius/3.0/clients.conf
client localhost {
    ipaddr = 127.0.0.1
    secret = testing123
    require_message_authenticator = no
}

client borne-wifi {
    ipaddr = 192.168.20.10
    secret = monsecret
    require_message_authenticator = no
}
```

Cette configuration établit la confiance mutuelle entre le point d'accès WiFi et le serveur FreeRADIUS en spécifiant à FreeRadius qu'il y'a un point d'accès wifi à l'adresse 192.168.20.10.

Génération et déploiement des certificats SSL/TLS

8. Préparation de l'environnement des certificats :

```
┌─[root@parrot]─[/etc/freeradius/3.0/certs]
└──╼ #cd /etc/freeradius/3.0/certs
┌─[root@parrot]─[/etc/freeradius/3.0/certs]
└──╼ #sudo make clean
┌─[root@parrot]─[/etc/freeradius/3.0/certs]
└──╼ #sudo make
openssl req -new -x509 -keyout ca.key -out ca.pem \
        -days '60' -config ./ca.cnf \
        -passin pass:'whatever' -passout pass:'whatever'
```

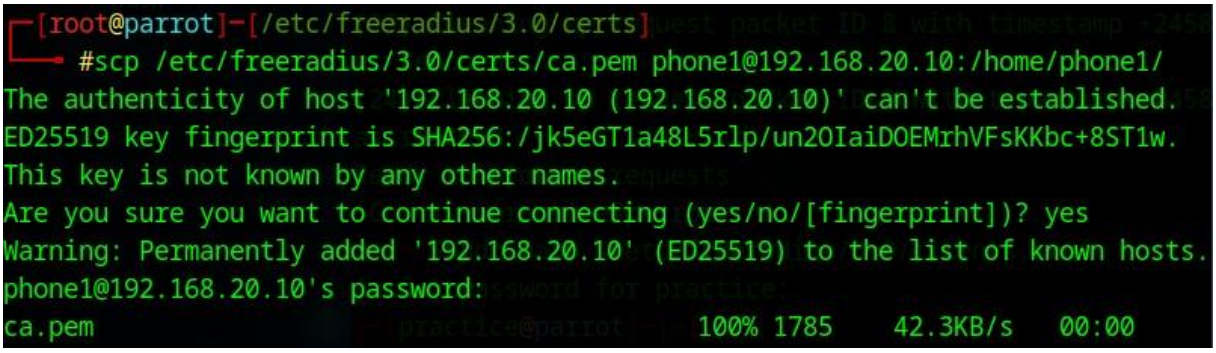
Explication :

- Navigation vers le répertoire des certificats FreeRADIUS
- **make clean** : supprime les anciens certificats pour repartir sur une base propre
- **make** : génère automatiquement les certificats nécessaires

openssl req -new -x509 -keyout ca.key -out ca.pem -days '60' -config ./ca.cnf -passin pass:'whatever' -passout pass:'whatever'

Cette commande crée un certificat d'autorité de certification (CA) auto-signé.

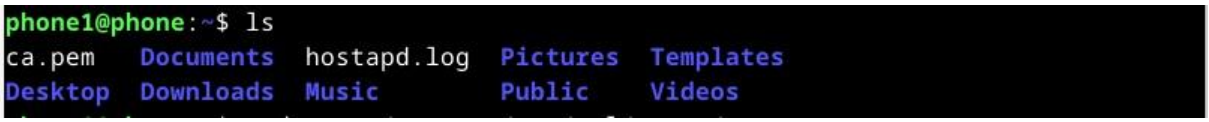
9. Transfert du certificat CA vers le serveur distant :



Explication : Cette commande copie le certificat de l'autorité de certification vers le serveur hébergeant le point d'accès WiFi (192.168.20.10) en utilisant **SCP** . Ce certificat est nécessaire pour :

- Établir la confiance TLS entre le client et le serveur RADIUS
- Valider l'identité du serveur RADIUS lors des authentifications EAP-TLS/PEAP
- Sécuriser les échanges d'authentification

10. Vérification du transfert :



Le fichier **ca.pem** est maintenant présent dans le répertoire home, confirmant le transfert réussi.

Importance sécuritaire : Ce certificat CA permettra aux clients WiFi de vérifier l'authenticité du serveur RADIUS, empêchant les attaques de type "man-in-the-middle" durant l'authentification EAP.

Installation du certificat et configuration du client EAP

11. Installation du certificat CA dans le système :

Explication : Cette commande copie le certificat de l'autorité de certification dans le répertoire système **/etc/ssl/certs/.** Cette localisation permet au système d'exploitation et aux applications (comme **wpa_supplicant**) de faire confiance à ce certificat pour valider l'identité du serveur RADIUS.

12. Configuration du client WiFi EAP

```
phone1@phone:~$ sudo nano /etc/wpa_supplicant/test-eap.conf
```

```
  GNU nano 7.2              /etc/wpa_supplicant/test-eap.conf *
ctrl_interface=/var/run/wpa_supplicant/test_eap
ctrl_interface_group=0
network={
    ssid="TestEAP"
    key_mgmt=WPA-EAP
    eap=TTLS
    identity="maguette"
    password="Passer123"
    ca_cert="/etc/ssl/certs/ca.pem"
    phase2="auth=PAP"
}
```

Le fichier de configuration contient les paramètres essentiels pour l'authentification EAP :

Configuration générale :

- **p2p_disabled=1** : désactive WiFi Direct (P2P) pour éviter les conflits
- **ctrl_interface=/var/run/wpa_supplicant/test_eap** : socket de contrôle pour **wpa_supplicant**
- **ctrl_interface_group=0** : groupe autorisé à utiliser l'interface de contrôle

Configuration réseau WiFi :

- **ssid="TestEAP"** : nom du réseau WiFi (correspond à hostapd.conf)
- **key_mgmt=WPA-EAP** : utilise WPA2-Enterprise avec EAP
- **eap=TTL**S: utilise EAP-TTLS
    - EAP-TTLS est une méthode d'authentification qui utilise un tunnel TLS pour sécuriser l'échange d'informations d'authentification dans les réseaux sans fil et filaires.
- **identity="maguette"** : nom d'utilisateur pour l'authentification
- **password="Passer123"** : mot de passe utilisateur
- **ca_cert="/etc/ssl/certs/ca.pem"** : certificat CA pour valider le serveur RADIUS
- **phase2="auth=MSCHAPV2"** : méthode d'authentification dans le tunnel PEAP (MS-CHAPv2)

13. Préparation de l'interface client :

```
phone1@phone:~$ sudo mkdir -p /var/run/wpa_supplicant/test_eap
phone1@phone:~$ sudo chmod 0755 /var/run/wpa_supplicant/test_eap
phone1@phone:~$ sudo ip link set wlan1 up
```

Explication :

- Création du répertoire pour l'interface de contrôle de wpa_supplicant
- Attribution des permissions appropriées (lecture/écriture/exécution pour root)
- Activation de l'interface réseau wlan1 qui servira de client WiFi

Cette configuration prépare un client WiFi capable de s'authentifier via PEAP/MS-CHAPv2 contre le serveur FreeRADIUS.

Configuration du tunnel d'authentification interne (inner-tunnel) :

14. Modification du fichier inner-tunnel :

```
┌─[practice@parrot]─[~]
└──╼ $sudo nano /etc/freeradius/3.0/sites-enabled/inner-tunnel
```

```
  GNU nano 7.2     /etc/freeradius/3.0/sites-enabled/inner-tunnel     Modified
#  The order of the realm modules will determine the order that
#  we try to find a matching realm.
#
#  Make *sure* that 'preprocess' comes before any realm if you
#  need to setup hints for the remote radius server
authorize {
        ldap
    if ((ok || updated) && &User-Password) {
        update control {
            Auth-Type := ldap
        }
    }
```

Explication :

- **ldap** : active le module LDAP pour l'autorisation
- La condition vérifie si LDAP a réussi (ok || updated) ET si un mot de passe utilisateur existe
- Si ces conditions sont remplies, définit le type d'authentification sur LDAP
- Cette logique permet d'utiliser LDAP comme backend d'authentification principal

```
  GNU nano 7.2     /etc/freeradius/3.0/sites-enabled/inner-tunnel     Modified
#  others will not.
#
#  The common reasons to set the Auth-Type attribute by hand
#  is to either forcibly reject the user, or forcibly accept him.
#
authenticate {
        Auth-Type LDAP {
            ldap
        }
```

Explication :

- Configure le module **LDAP** pour gérer l'authentification quand **Auth-Type** est défini sur **LDAP**
- Le module **LDAP** vérifiera les **credentials** de l'utilisateur contre l'annuaire LDAP

Contexte PEAP : Dans PEAP, après l'établissement du tunnel TLS sécurisé, **l'authentification réelle (phase 2)** se fait via ce tunnel interne. Cette configuration permet d'utiliser un serveur LDAP (comme Active Directory) pour valider les identifiants utilisateur de manière sécurisée.

Avantage : Cette architecture permet une authentification centralisée via **LDAP** tout en bénéficiant de la sécurité du **tunnel TLS de PEAP**.

Redemarrage du service :

```
^C┌─[✗]─[practice@parrot]─[~]
└──╼ $sudo systemctl restart  freeradius
┌─[practice@parrot]─[~]
└──╼ $sudo systemctl stop freeradius
┌─[practice@parrot]─[~]
└──╼ $sudo freeradius -X
FreeRADIUS Version 3.2.1
```

Nous activons FreeRadius en mode debug/verbose

```
Listening on auth address * port 1812 bound to server default
Listening on acct address * port 1813 bound to server default
Listening on auth address :: port 1812 bound to server default
Listening on acct address :: port 1813 bound to server default
Listening on auth address 127.0.0.1 port 18120 bound to server inner-tunnel
Listening on proxy address * port 60537
Listening on proxy address :: port 54315
Ready to process requests
```

le serveur FreeRADIUS est opérationnel et prêt à traiter les demandes d'authentification

17. Connexion du client WiFi:

```
phone1@phone:~$ sudo wpa_supplicant -i wlan1 -c /etc/wpa_supplicant/test-eap.con
f -D nl80211 -d
```

Explication des paramètres :

- -i wlan1 : utilise l'interface wlan1 comme client WiFi
- -c /etc/wpa_supplicant/test-eap.conf : fichier de configuration EAP
- -D nl80211 : pilote pour les cartes WiFi modernes
- -d : mode debug pour voir les détails de la connexion

Processus : Cette commande initie la connexion **WiFi** avec authentification **EAP**. Le client va :

1. Se connecter au SSID **"TestEAP"**
2. Établir un tunnel TLS avec le serveur RADIUS
3. Authentifier l'utilisateur **"maguette"** via PEAP/MS-CHAPv2

Le mode **debug** permettra de voir tout le processus d'authentification en temps réel.

Processus d'authentification WiFi EAP complet :

18. Logs du point d'accès WiFi (hostapd) :

```
wlan0: RADIUS Authentication server 192.168.200.10:1812
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
wlan0: STA be:eb:17:e9:2b:f0 IEEE 802.11: authenticated
wlan0: STA be:eb:17:e9:2b:f0 IEEE 802.11: associated (aid 1)
wlan0: CTRL-EVENT-EAP-STARTED be:eb:17:e9:2b:f0
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1
wlan0: CTRL-EVENT-EAP-SUCCESS2 be:eb:17:e9:2b:f0
wlan0: STA be:eb:17:e9:2b:f0 WPA: pairwise key handshake completed (RSN)
wlan0: EAPOL-4WAY-HS-COMPLETED be:eb:17:e9:2b:f0
wlan0: AP-STA-CONNECTED be:eb:17:e9:2b:f0
wlan0: STA be:eb:17:e9:2b:f0 RADIUS: starting accounting session 9C7BCD1E9F87D8C
7
wlan0: STA be:eb:17:e9:2b:f0 IEEE 802.1X: authenticated - EAP type: 21 (TTLS)
```

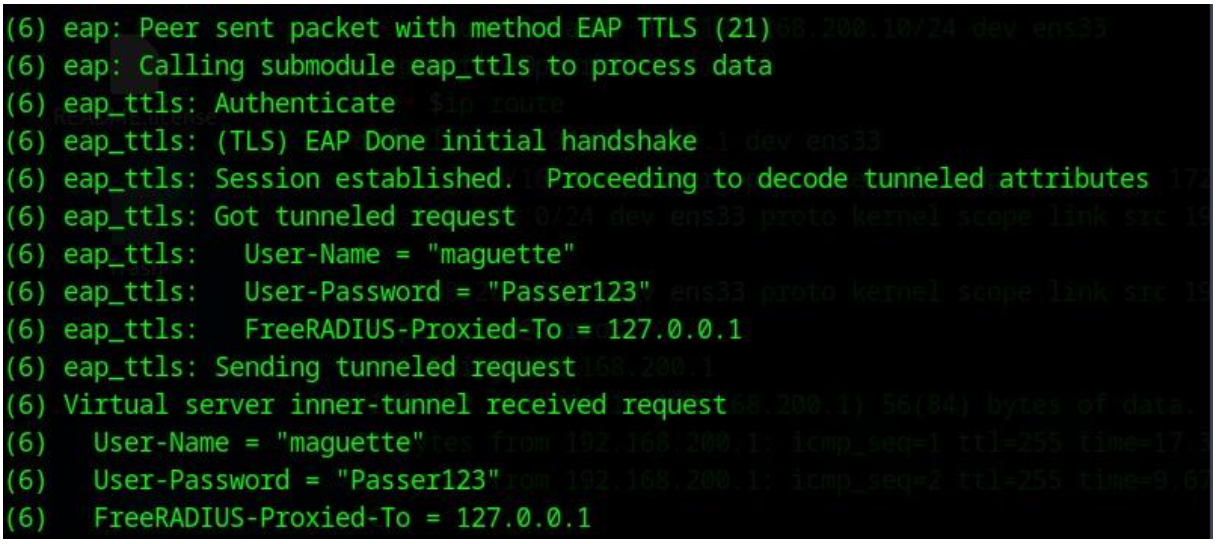
Établissement de la connexion :

- **RADIUS Authentication server 192.168.200.10:1812** : connexion au serveur RADIUS confirmée
- **interface state UNINITIALIZED->ENABLED** : activation de l'interface WiFi
- **AP-ENABLED** : point d'accès opérationnel

Processus d'authentification du client (MAC: be:eb:17:e9:2b:f0) :

- **IEEE 802.11: authenticated** : authentification 802.11 réussie
- **IEEE 802.11: associated (aid 1)** : association WiFi établie avec ID 1
- **CTRL-EVENT-EAP-STARTED** : début du processus EAP
- **CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1** : méthode EAP proposée
- **CTRL-EVENT-EAP-SUCCESS2** : authentification EAP réussie
- **WPA: pairwise key handshake completed (RSN)** : échange de clés WPA2 terminé
- **EAPOL-4WAY-HS-COMPLETED** : handshake EAPOL 4-way complet
- **AP-STA-CONNECTED** : station connectée avec succès
- **IEEE 802.1X: authenticated - EAP type: 21 (TTLS)** : authentification **EAP-TTLS** confirmée.

19. Logs du serveur FreeRADIUS :



Attributs RADIUS reçus :

- **User-Name** = "**maquette**" : identifiant utilisateur
- **Called-Station-Id** = **"8E-5E-AB-2B-3A-EA:TestEAP"** : MAC du point d'accès + SSID
- **NAS-Port-Type** = **Wireless-802.11** : type de connexion sans fil
- **Service-Type = Framed-User** : type de service utilisateur
- **Calling-Station-Id = "BE-EB-17-E9-2B-F0"** : MAC du client WiFi
- **Connect-Info = "CONNECT 54Mbps 802.11g"** : informations de connexion
- **Attributs de chiffrement WPA2 (WLAN-Pairwise-Cipher, WLAN-Group-Cipher, etc.)**

Processus EAP-TTLS :

- **eap**: Peer sent packet with method EAP TTLS (21) : méthode EAP-TTLS utilisée
- **eap_ttls: (TLS) EAP Done initial handshake** : établissement du tunnel TLS
- **eap_ttls: Session established. Proceeding to decode tunneled attributes** : tunnel sécurisé établi

- **eap_ttls: Got tunneled request** : réception des credentials dans le tunnel
- **User-Name = "maquette" et User-Password = "Passer123"** : identifiants déchiffrés
- **Virtual server inner-tunnel received request** : transfert vers le tunnel interne

Résultat final :

```
(6) Sent Access-Accept Id 6 from 192.168.200.10:1812 to 192.168.20.10:43396 leng
th 176
```

Synthèse du processus d'authentification :

1. **Connexion WiFi initiale** : Le client se connecte au SSID "TestEAP"
2. **Négociation EAP** : Établissement du protocole EAP-TTLS
3. **Tunnel TLS** : Création d'un tunnel chiffré entre client et serveur RADIUS
4. **Authentification interne** : Validation des credentials "maquette/Passer123" via le tunnel sécurisé
5. **Access-Accept** : Serveur RADIUS autorise la connexion
6. **Échange de clés WPA2** : Établissement des clés de chiffrement pour la session WiFi
7. **Connexion établie** : Client authentifié et connecté au réseau WiFi sécurisé

Sécurité : Cette authentification combine la sécurité du tunnel TLS (EAP-TTLS) avec le chiffrement WPA2, offrant une protection robuste contre l'interception et les attaques de type man-in-the-middle.

Conclusion :

L'implémentation de l'authentification centralisée Wi-Fi avec FreeRADIUS et LDAP permet de renforcer la sécurité du réseau tout en simplifiant la gestion des utilisateurs. Grâce au protocole EAP-TTLS, les identifiants sont transmis de manière chiffrée via un tunnel TLS, garantissant leur confidentialité. L'intégration avec l'annuaire LDAP assure un contrôle d'accès unifié, évolutif et centralisé. Cette architecture constitue une solution robuste, adaptée aux environnements professionnels ou pédagogiques souhaitant concilier sécurité, fiabilité et gestion centralisée des accès Wi-Fi.