Man-in-the-Middle attack on WPA2-Enterprise networks.

Project for the Télécom SudParis course NET4104.

Nicolas Rocq, Benoit Marzelleau, Baptiste Sauvé,
Baptiste Legros, Tom Burellier

2024-05-02

Contents

List of Figures

1 Introduction

Ce projet vise à exploiter une faiblesse de configuration rencontrée dans la majorité des réseaux WPA2-Entreprise. En effet, il est très courant que les clients n'aient pas mis en oeuvre la validation du certificat CA, ce qui permet à un tier d'usurper l'identité d'un point d'accès Wi-Fi utilisant le mécanisme de sécurité WPA2-Entreprise.

En particulier, on s'intéressera aux réseaux utilisant le protocole d'authentification PEAP-MSCHAPv2 pour sa popularité. En l'espèce, on montrera dans quel mesure la faiblesse de configuration susmentionnée permet d'obtenir l'empreinte MD4 du mot de passe de l'utilisateur.

1.1 Contributeurs

Encadrant :

- Rémy Grünblatt[1]

Étudiants :

- Nicolas Rocq[2]
- Benoit Marzelleau[3]
- Baptiste Sauvé[4]
- Baptiste Legros[5]
- Tom Burellier[6]

[1]https://github.com/rgrunbla
[2]https://github.com/Nishogi
[3]https://github.com/xanode
[4]https://github.com/Nepthales
[5]https://github.com/Direshaw
[6]https://github.com/Balmine

2 EAP

Le protocole EAP (*Extensible Authentification Protocol*) est un protocole de communication réseau qui est utilisé pour authentifier un partenaire.

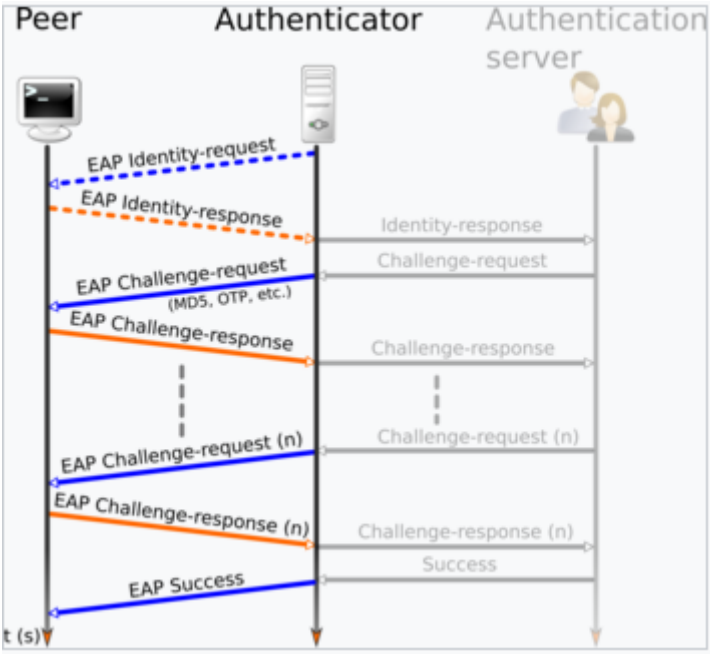


Figure 1: Séquence MSCHAPv2

Il y a 3 grandes étapes dans la communication entre un partenaire et l'authentificateur (*respectivement Peer et Authenticator ci-dessus*) : - Une phase d'identification, l'authenticator envoie une requête au peer et recoit une réponse. - Une phase de challenge du peer, l'authenticator va envoyer de un à plusieurs challenges contenant une méthode d'authentification et le peer va répondre à chaque challenge avant d'en recevoir un autre. - Une phase de validation, en fonction des réponses du peer, l'authenticator va envoyer un code au peer signifiant le succes (et donc l'établissement de la connexion) ou l'echec.

Ce protocole est considéré comme extensible, car il y a plusieurs de méthodes d'authentification possibles (MD5, OTP, SIM, GTC…) mais il n'est pas nécessaire de refaire un protocole si on souhaite implémenter une autre méthode. Dans notre étude, nous utiliserons comme méthode d'authentification MSCHAPv2 qui est adopté par le standard WPA et WPA2.

Le PEAP (*Protected Extensible Authenticator Protocol*) est la version "protégée" concu par Microsoft et Cisco et inspiré par le EAP-TTLS (utilisant un tunnel TLS). Le PEAP se déroule en deux phases, analogue au EAP: - une phase d'identification du serveur via utilisation de clés publique (PKI), une fois identifié, un tunnel sécurisé chiffre la phase suivante. - une phase d'identification du client, qui se déroule à l'intérieur du tunnel de la phase 1.

L'utilisation du PEAP nécessite d'avoir un Certificat d'Authentification (*CA*) SSL ou TLS du coté serveur, cependant pour le client, le certificat n'est pas requis.

3 MSCHAPv2

Le protocole MSCHAPv2 est un protocole d'authentification de type CHAP (*Challenge Handshake Authentication Protocol*). L'objectif de cette classe de protocoles est de permettre à un client de s'authentifier en respectant les critères suivants : - pas d'échange en clair du mot de passe ou d'une empreinte de celui-ci ; - rejouabilité des échanges de nul effet pour un tier qui l'aurait intercepté.

Pour y parvenir, les protocoles de type CHAP réclament une preuve d'identité du client en lui demandant de répondre à un défi qui ne peut être relevé que par une entité connaissant le mot de passe. Par exemple, il pourrait être demandé au client de chiffrer un nombre aléatoire (fourni par l'authentificateur) avec le mot de passe (chose qui ne peut être réalisée qu'en connaissant le mot de passe).

Le protocole MSCHAPv2 fonctionne de la manière suivante :

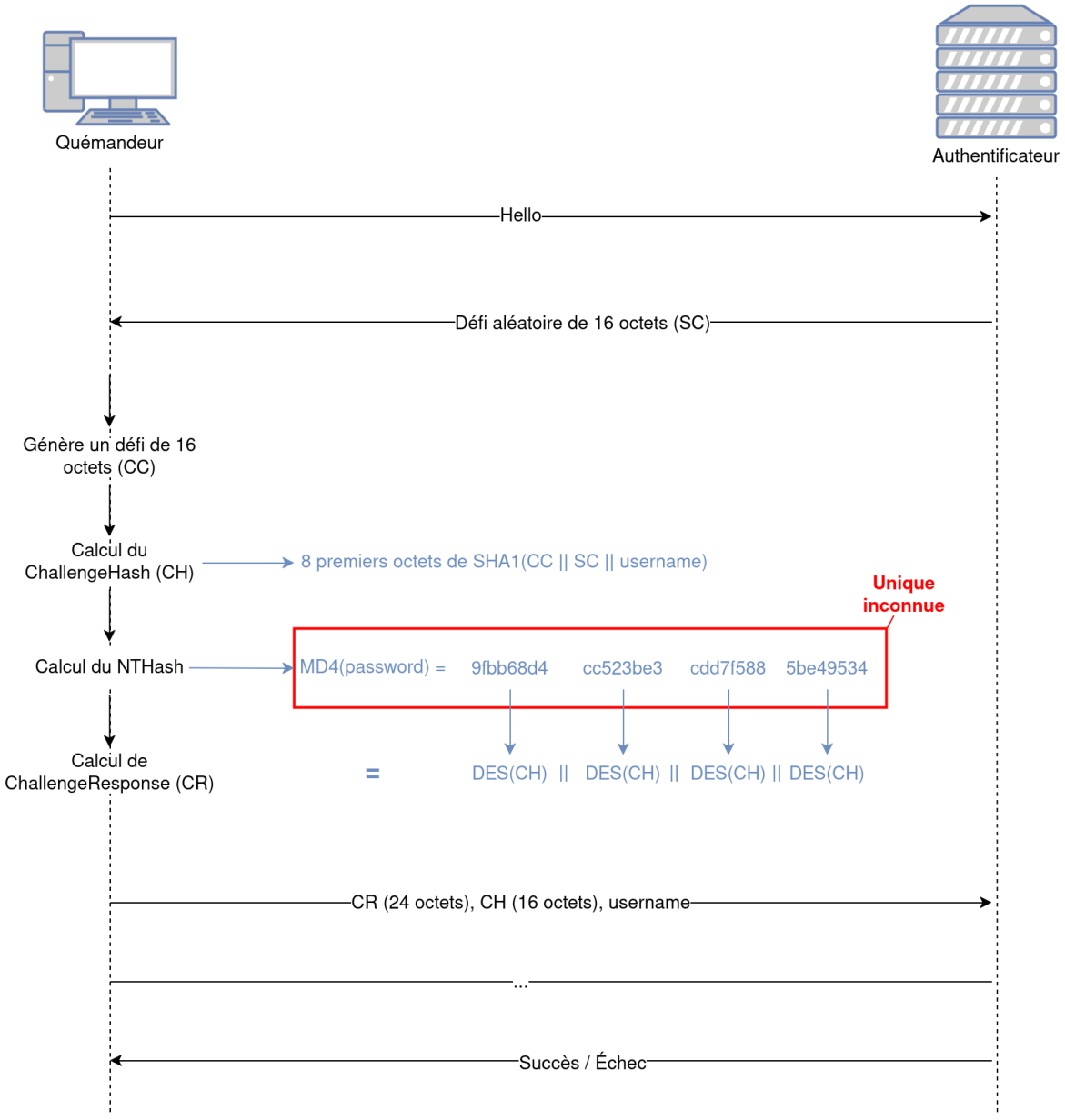


Figure 2: Séquence MSCHAPv2

1. Le serveur d'authentification envoie 16 octets aléatoires au client (le défi, *SC*) ;
2. Le client, pour prouver son identité, réalise :
    - Génération d'un défi de 16 octets (*CC*) ;
    - Calcul du `ChallengeHash` (*CH*) : `SHA1(CC || SC || username)` ;
    - Calcul de l'empreinte MD4 du mot de passe ;
    - Calcul de la réponse au défi à partir de l'empreinte du défi ;
3. Le client envoie au serveur le nom d'utilisateur et la réponse au défi ;
4. Le serveur vérifie la réponse au défi en vérifiant que la réponse du client est correcte, et accepte ou non l'authentification.

Ainsi la seule inconnue pour un attaquant qui intercepte les échanges est l'empreinte MD4 du mot de passe de l'utilisateur, qui est utilisé pour construire les trois clés DES utilisées pour calculer la réponse au défi. Tout autre élément du protocole est soit envoyé en clair, soit facilement déductible des échanges.

4 L'attaque

4.1 Evil twin

4.1.1 Principe

Une attaque de type "evil twin" exploite la façon dont les clients WiFi reconnaissent les réseaux, en se basant principalement sur le nom du réseau (ESSID) sans exiger de la station de base (point d'accès) qu'elle s'authentifie auprès du client. Il s'agit d'une faiblesse de configuration rencontrée dans la majorité des réseaux WPA2-Entreprise, lesquels n'imposent pas aux utilisateurs de contrôler le certificat présenté par le point d'accès pour des raisons de simplicité.

Les points clés sont les suivants de l'attaque sont les suivants :

- **Différenciation difficile** : Il est difficile de différencier un point d'accès légitime d'un point d'accès malveillant lorsque leurs ESSID sont confondus et qu'ils partagent le même mécanisme de sécurité (WPA2-Enterprise dans notre cas). D'autant plus que les réseaux WPA2-Enterprise que l'ont retrouve dans les établissements utilisent souvent plusieurs point d'accès avec le même ESSID pour étendre la courverture de manière transparente pour les utilisateurs finaux.
- **Itinérance des clients et manipulation des connexions** : Le protocole 802.11 permet aux appareils de passer d'un point d'accès à l'autre au sein d'un même ESS. Il est possible d'exploiter cette possibilité en incitant un appareil à se déconnecter de son point d'accès actuel et à se connecter à un point d'accès malveillant. Il est possible d'y parvenir en offrant un signal plus fort ou en perturbant la connexion au point d'accès légitime en envoyant des paquets de désauthentification ou en le brouillant.

4.1.2 Mise en œuvre

Nous allons configurer le point d'accès malveillant à l'aide de `hostapd-wpe` qui est un correctif de `hostapd` qui permet de réaliser l'attaque "evil twin" et surtout d'obtenir les informations d'identification du client (dont le sujet est traité ci-après) échangés lors de l'authentification (et normalement inaccessibles avec `hostapd` seul).

```
# Installation de hostapd-wpe
sudo apt install hostapd-wpe
```

Nous configurons `hostapd-wpe` de la sorte pour qu'il diffuse un point d'accès avec le même ESSID que le réseau cible.

```
interface=wlan0
ssid=eduroam
channel=1
ignore_broadcast_ssid=0
eap_user_file=mi-net4104/hostapd-wpe.eap_user
ca_cert=mi-net4104/attack/ca.pem
server_cert=mi-net4104/attack/server.pem
private_key=mi-net4104/attack/server.pem
private_key_passwd=password
dh_file=mi-net4104/attack/dh
eap_fast_a_id=101112131415161718191a1b1c1d1e1f
eap_server=1
```

```
eap_fast_a_id_info=hostapd-wpe
eap_fast_prov=3
ieee8021x=1
pac_key_lifetime=604800
pac_key_refresh_time=86400
pac_opaque_encr_key=000102030405060708090a0b0c0d0e0f
wpa=2
wpa_key_mgmt=WPA-EAP
wpa_pairwise=TKIP CCMP
```

```
# Lance le point d'accès malveillant
sudo hostapd-wpe hostapd-wpe.conf -s
```

4.2 MSCHAPv2

Supposons que nous souhaitons nous authentifier auprès d'un réseau Wi-Fi utilisant le protocole PEAP-MSCHAPv2. Pour obtenir les informations d'authentification de d'un utilisateur du réseau cible, diffuser un point d'accès Wi-Fi avec le même SSID que le réseau cible suffit, à condition que les clients tentent de s'y connecter en ne vérifiant pas le certificat CA du serveur d'authentification.

Dès lors, nous pouvons nous concentrer auprès de la méthode d'authentification (MSCHAPv2), et remarquer que l'empreinte MD4 du mot de passe de l'utilisateur est suffisante pour s'authentifier en ce qu'elle agit comme le mot de passe lui-même.

4.2.1 Attaque par dictionnaire ?

Une première approche pour obtenir le mot de passe de l'utilisateur pourrait être de réaliser une attaque par dictionnaire sur l'empreinte MD4. Par exemple, on pourrait simplement calculer l'empreinte MD4 d'un grand nombre de mots de passe possibles, s'en servir pour calculer la réponse à un défi et comparer avec la réponse fournie par le client.

Le problème de cette approche est que la réussite de cette attaque n'est pas garantie, car le mot de passe de l'utilisateur peut être complexe et ne pas figurer dans le dictionnaire.

4.2.2 Attaque par force brute ?

Dans la mesure où le mot de passe de l'utilisateur est susceptible d'avoir une longueur arbitraire et d'être composé de caractères d'un large ensemble, il pourrait être intéressant d'attaquer par la force brute l'empreinte MD4 du mot de passe elle-même. Mais cette empreinte est de 128 bits, soit 2^{128} possibilités, ce qui est bien trop grand pour être réalisable en un temps raisonnable.

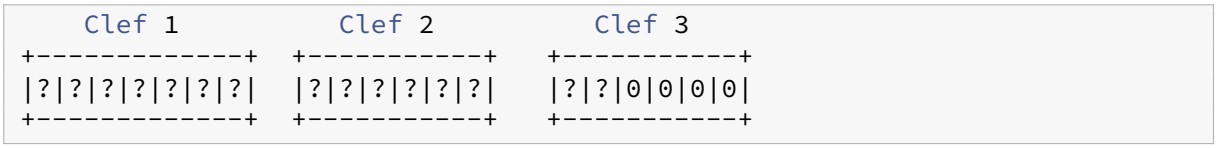
4.2.3 Diviser pour mieux régner

L'empreinte MD4 que nous essayons d'obtenir est utilisée comme la clef de trois chiffrements DES. Les clés DES étant de 7 octets, chaque opération DES utilise un morceau de 7 octets de l'empreinte MD4. Ainsi, au lieu de chercher l'empreinte MD4 elle-même, on pourrait chercher les trois clés DES qui permettent de la construire. Dès lors, plus besoin d'attaquer par la force brute une empreinte de 128 bits, mais trois clés de 56 bits chacune.

Comme il y a 3 opérations DES et que ces opérations sont indépendantes les unes des autres, on a une complexité globale de $3 \times 2^{56} = 2^{57.59}$, ce qui est bien mieux que 2^{128}.

Mais il y a quelque chose qui ne va pas. En l'espèce, nous avons besoin de trois clefs DES de 56 bits pour un total de 168 bits, alors que l'empreinte MD4 n'en fait que 128 : il manque 40 bits. Lors de la réalisation de MSCHAPv2, Microsoft a palié à ce problème en ajoutant 40 bits nuls à l'empreinte MD4, rendant la troisième clef DES d'une longueur effective de 16 bits.

Ainsi, il n'y a que 2^{16} possibilités pour cette dernière clef DES, ce qui est tout à fait réalisable par la force brute. La complexité totale de l'attaque est donc de $2^{56} + 2^{56} + 2^{16} = 2^{57}$. C'est considérablement mieux.



5 Géné

6 Sources

6.1 Pandoc

CI with pandoc - https://gitlab.com/pandoc/pandoc-ci-example

6.2 Templating

Templating with pandoc - https://gitlab.cylab.be/a.muls/pandoc-for-pdf