



Published on hacktimes.com (<http://www.hacktimes.com>)

Rompiendo el protocolo WPA con clave precompartida (PSK) y el protocolo de integridad de clave temporal (TKIP)

By modlok

Creado 20 Jun 2006 - 15:41

Los proveedores de servicios o ISPs que distribuyen puntos de acceso siguen sin tener presente la seguridad como una necesidad, por lo que en el mejor de los casos utilizan el inseguro protocolo WEP.

Poco a poco se van implementando en los dispositivos, configuraciones por defecto que dejan a un lado el lamentable protocolo WEP, que tantos y tantos quebraderos de cabeza ha generado, tanto en particulares como en entornos de pequeña y mediana empresa.

En este breve artículo veremos como las “nuevas” implementaciones como WPA-PSK (TKIP) son susceptibles a ataques de fuerza bruta.

WPA es un subconjunto del desarrollo del estándar 802.11i, donde con un porcentaje de éxito mucho menor que con el protocolo WEP, es posible la obtención de la contraseña a través de un ataque por diccionario.

Para entender un poco más el protocolo WPA-PSK es necesario conocer, aunque sea de forma muy superficial, el protocolo TKIP (Temporal Key Integrity Protocol), que fue diseñado para cubrir en gran medida las críticas continuas sobre la debilidad del protocolo WEP.

Podemos decir que TKIP:

- Es un vector de inicialización o IV de 58 bits
- Tiene una clave de 128 bits
- Generación de bloques de 4 Bytes (MIC) a partir de la dirección de origen, de destino y datos, es decir, cifra el checksum incluyendo direcciones MAC.

WPA-PSK es un protocolo de clave compartida (PSK) para aquellos entornos donde no existen servidores de autenticación (RADIUS). La clave esta compuesta por una cadena de 256 bits o una frase entre 8 y 63 caracteres.

PSK = PMK = PBKDF2 (frase, SSID, longitud SSID 4096, 256)

Paso a paso

- 1. Asociación del dispositivo cliente con el AP.
- 2. Autenticación y distribución de claves PMK (Pair-Wise MasterKey)
- 3. Creación del PTK (Pair Wise Transient Key) basado en el anterior PMK
- 4. Verificación de la integridad.
- 5. Establecimiento de sesión a través del protocolo TKIP.

Para realizar esta prueba se han utilizado un portátil **ASUS Pentium IV**, una tarjeta **PCMCIA AirPlus Extreme con chipset Atheros**. La distribución elegida para llevar a cabo nuestros planes es **BACKTRACK** con drivers **MADWIFI**.

Tanto los puntos de acceso como los clientes del mismo pertenecen a un entorno controlado y destinado exclusivamente a estas pruebas.

Datos de los dispositivos de laboratorio:

- BSSID del AP(Viene a ser la MAC): 00:30:BD:98:CE:62
- ESSID: HACKTIMES
- MAC dispositivo cliente: 00:15:E9:1B:33:53

Lo primero que haremos es buscar un punto de acceso que tenga este tipo de configuración (WPA-PSK(TKIP)). En nuestro caso y debido a que disponemos de puntos de acceso y dispositivos cliente debidamente configurados (no es objeto de este artículo el explicar como se configura un AP para que incorpore este tipo de cifrado, que en último caso dependerá del dispositivo instalado así como de su firmware) es muy sencillo encontrar un punto de acceso, aunque a veces se hace complicado localizar un AP con este tipo cifrado fuera del laboratorio.

El primer paso antes de realizar cualquier acción es establecer el modo monitor en nuestra tarjeta PCMCIA.

Modo 1

```
# airmon start ath0
```

Modo 2

```
# iwconfig ath0 mode monitor
```

A continuación lanzaremos la herramienta **airodump** para descubrir las redes cercanas, incluida la que hemos configurado para la ocasión, así como diversos datos que nos serán imprescindibles.

Nota: En caso de que **airodump** no sea capaz de identificar un AP con cifrado WPA-PSK, es necesario el uso de herramientas como **Kismet** o **NetStumbler**, entre otras.

Esta herramienta será vital para detectar nuestro AP, pero será también lo es el fichero que se genere donde intentaremos capturar los datos de autenticación de un determinado dispositivo conectado al AP.

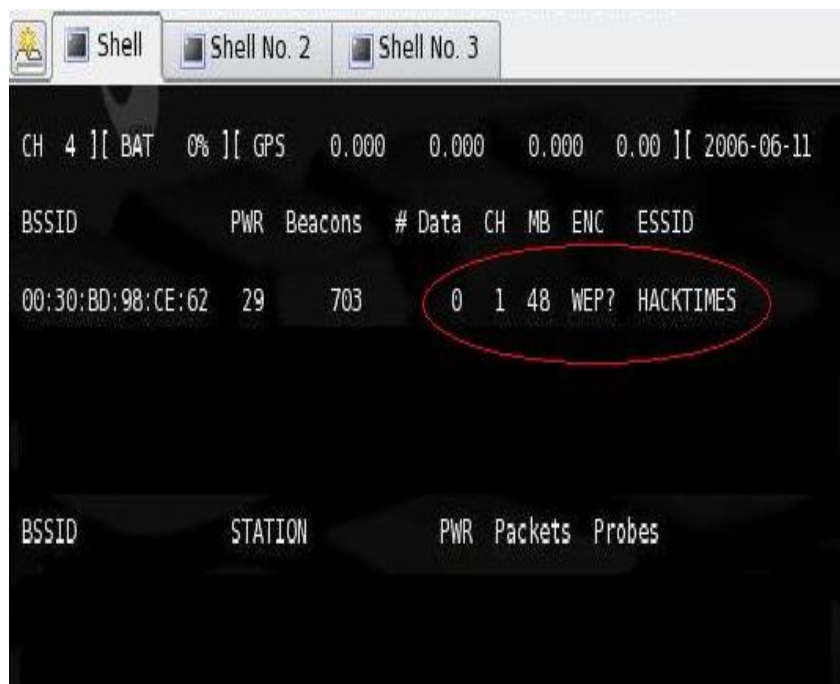
Una vez tengamos la tarjeta en modo monitor, ejecutaremos..

```
# airodump ath0 captura 0 0
```

Se puede observar que con estas opciones (el último 0) que estamos capturando todos los paquetes, a diferencia de cuando se rompe el protocolo WEP, que utilizamos la opción 1 que captura solo los paquetes de tipo IV, a fin de evitar que se generen ficheros de captura muy grandes.

Para llevar a cabo este tipo de ataque únicamente necesitamos capturar el “handshake” (negociación) que se produce cuando un cliente se autentica con un punto de acceso.

Una vez esto en marcha podemos ver en la siguiente imagen como **airodump** ha detectado el punto de acceso que nos interesa (ESSID HACKTIMES).

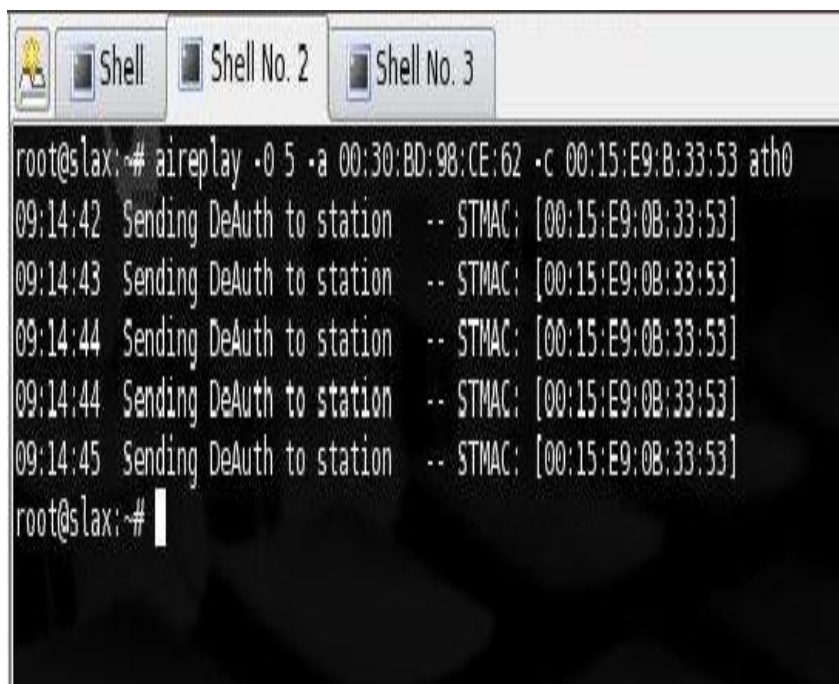


Los objetivos que nos planteamos llegados a este punto es capturar el “handshake”, que será reflejado en la columna **#data**, donde veremos incrementado el número de paquetes una vez realizado el ataque de denegación de servicio.

Aunque en un principio pueda que airodump no identifique el cifrado y aparezca con algo similar a lo siguiente “WEP?” en la columna “ENC”. En cuanto se capture un cierto volumen de tráfico, o mejor aún un “handshake” se establecerá ya como “WPA”.

A continuación se realizará un ataque de denegación de servicio contra el cliente conectado al AP, que en nuestro caso es **00:15:E9:1B:33:53**. Para ver que clientes están conectados a un determinado punto de acceso es recomendable el uso de **Kismet**, tanto por la sencillez como por la cantidad de información que muestra, pero existen más herramientas que nos pueden brindar resultados excelentes.

```
# aireplay -0 5 -a 00:30:BD:98:CE:62 -C 00:15:E9:1B:33:53
```

A terminal window with three tabs: 'Shell', 'Shell No. 2', and 'Shell No. 3'. The active tab shows a root shell on a machine named 'slax'. The user has entered the command 'aireplay -o 5 -a 00:30:BD:98:CE:62 -c 00:15:E9:0B:33:53 ath0'. The terminal output shows five lines of 'Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]' at timestamps 09:14:42, 09:14:43, 09:14:44, 09:14:44, and 09:14:45. The prompt 'root@slax:~#' is visible at the end of the output.

```
root@slax:~# aireplay -o 5 -a 00:30:BD:98:CE:62 -c 00:15:E9:0B:33:53 ath0
09:14:42 Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]
09:14:43 Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]
09:14:44 Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]
09:14:44 Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]
09:14:45 Sending DeAuth to station -- STMAC: [00:15:E9:0B:33:53]
root@slax:~#
```

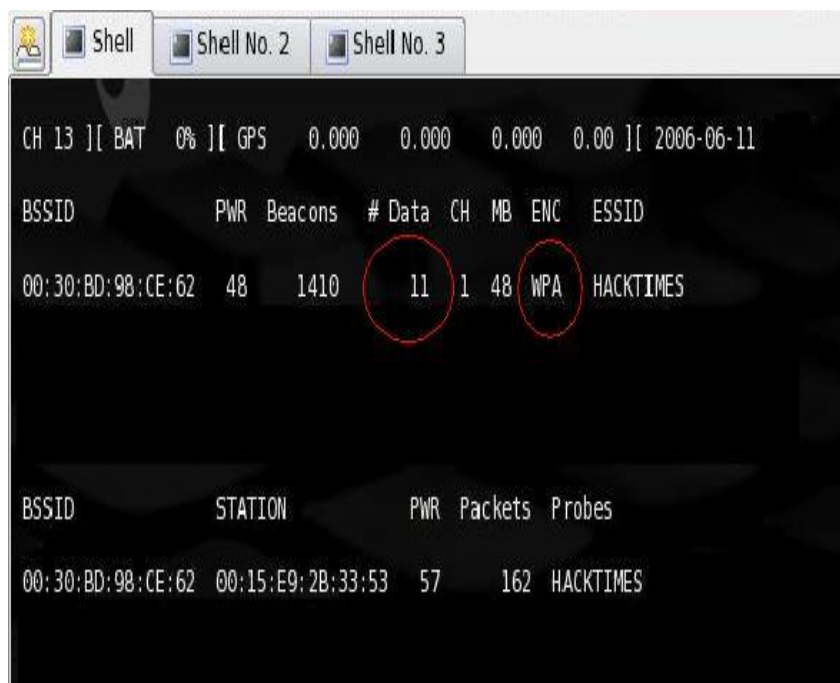
Ataque 0: deautenticación: este ataque es probablemente el más útil para recuperar un ESSID oculto así como para capturar el “handshake”.

Habrá que tener cuidado en no dejar el dispositivo cliente inoperativo, que es lo más normal en este tipo de ataques.

El efecto que se produce en el dispositivo cliente es inmediato, tal y como se puede apreciar.



Aunque los paquetes aumenten, por ahora no sabemos si realmente el cliente se ha desconectado y vuelto a autentificar al punto de acceso. Esto es necesario para poder capturar el “handshake”.



Una vez capturado los paquetes usaremos **aircrack** para primero verificar que realmente hemos obtenido el ansiado “handshake”, y si es así, lanzar el ataque.

```
# aircrack -a 2 -w ModlokHacktimesDICCsr.txt pskcrack-01.cap
```



```
root@slax:~/Hacktimes# aircrack -a 2 -w ModlokHacktimesDICCsr.txt pskcrack-01.cap
Opening pskcrack-01.cap
Read 9665 packets.

# BSSID          ESSID          Encryption
2 00:30:BD:98:CE:62 HACKTIMES      WPA (1 handshake)

Index number of target network ? 2
```

Hay que tener en cuenta que con un solo “handshake” es suficiente y que hay que compararlo con un diccionario, es decir, por fuerza bruta. Teniendo en cuenta estos dos factores dependerá de la calidad de nuestro diccionario el éxito ante este tipo de ataque, que en definitiva tiene un gran porcentaje de suerte.

Desde HackTimes hemos usado un fichero secuencial con más de 4 millones de líneas o posibles contraseñas.

```
aircrack 2.4
[05:03:02] 2215224 keys tested (172.59 k/s)
KEY FOUND! [ ZinedineZidane ]

Master Key   : 49 6D 52 00 13 FE C3 FF 6A 5F 8E 7F DB B2 66 8E
              04 F5 EA EA 0F 78 AF AA B2 9F FB 18 51 23 19 4E

Transcient Key : E7 25 F9 14 E0 EA 94 A1 58 C3 63 D8 1B EF FC CC
                8D CA 6E 00 96 0A 36 37 BA 9A B3 C4 FB 00 99 B7
                4A 2A 80 C2 4A 53 78 DC FA 3C A4 1B 9D 73 A3 B3
                86 36 6A 6D 13 C9 B0 52 71 AF 03 97 DC E6 F9 10

EAPOL HMAC   : 51 B1 F1 54 2E 40 8D E1 0E 76 02 D1 0E CA B3 AE
```

En la imagen superior observamos como se ha detectado la clave, con lo cual nuestro primer ataque al protocolo WPA-PSK(TKIP) ha sido todo un éxito.

Como todo proceso aquí existen 2 problemas, que pasaremos a comentar:

- **Tiempo:** más de 5 horas.
- **Rate:** 172 palabras/segundo

El tiempo que se tarda en completar un fichero de más de 4 millones de entradas, que no es mucho, es excesivo y es debido al rate o número de palabras verificadas por segundo.

Para acelerar el proceso es necesaria la creación de tablas de tipo Rainbow Tables, y el uso de herramientas como la versión 3.0 de **Cowpaty** y su utilidad **genpmk**. Una vez creadas las tablas podemos obtener un rate de más de 20000 palabras por segundo.

En una siguiente ocasión veremos como se realiza todo este proceso de forma detallada. ByeZZ

Enlaces:

http://www.remote-exploit.org/index.php/Main_Page

http://www.churchofwifi.org/default.asp?PageLink=Project_Display.asp?PID=86

<http://www.remote-exploit.org/index.php/Tutorials>

<http://madwifi.org/>

<http://www.wirelessve.org/>

<http://www.wardrive.net>

Source URL:

<http://www.hacktimes.com/?q=node/34>