

# Informe Laboratorio 3

## Sección 2

Iván Andrés Cáceres Satorres  
e-mail: ivan.caceres\_s@mail.udp.cl

Octubre de 2023

## Índice

<b>1. Descripción de actividades</b>	<b>2</b>
<b>2. Desarrollo (PASO 1)</b>	<b>2</b>
2.1. Identificar en qué se destaca la red del informante del resto . . . . .	3
2.2. Explicación Matemática de por qué se requieren más de 5000 paquetes para obtener la contraseña . . . . .	3
2.3. Obtención de la Contraseña con Ataque por Defecto de Aircrack-ng . . . . .	4
2.4. Indicación del Tiempo Empleado para Obtener la Contraseña . . . . .	4
2.5. Descifrado del Contenido Capturado . . . . .	5
2.6. Descripción de cómo se Obtiene la URL para Descargar el Archivo . . . . .	5
<b>3. Desarrollo (PASO 2)</b>	<b>6</b>
3.1. Indicación de Script para Modificar el Diccionario Original . . . . .	7
3.2. Cantidad de Contraseñas Finales en el Diccionario rockyou_mod.dic . . . . .	8
<b>4. Desarrollo (Paso 3)</b>	<b>9</b>
4.1. Obtiene contraseña con Hashcat con potfile . . . . .	9
4.2. Identifica nomenclatura del output . . . . .	10
4.3. Obtiene contraseña con Hashcat sin potfile . . . . .	11
4.4. Identifica nomenclatura del output . . . . .	12
4.5. Obtiene contraseña con aircrack-ng . . . . .	13
4.6. Identifica y modifica parámetros solicitados por pycrack . . . . .	13
4.7. Obtiene contraseña con pycrack . . . . .	16

## 1. Descripción de actividades

Su informante quiere entregarle la contraseña de acceso a una red, pero desconfía de todo medio para entregársela (aún no llega al capítulo del curso en donde aprende a comunicar una password sin que nadie más la pueda interceptar). Por lo tanto, le entregará un archivo que contiene un desafío de autenticación, que al analizarlo, usted podrá obtener la contraseña que lo permite resolver. Como nadie puede ver a su informante (es informante y debe mantener el anonimato), él se comunicará con usted a través de las redes inalámbricas y de una forma que solo usted, como experto en informática y telecomunicaciones, logrará esclarecer.

1. Identifique cual es la red inalámbrica que está utilizando su informante para enviarle información. Obtenga la contraseña de esa red utilizando el ataque por defecto de aircrack-ng, indicando el tiempo requerido para esto. Descifre el contenido transmitido sobre ella y descargue de Internet el archivo que su informante le ha comunicado a través de los paquetes que usted ha descifrado.
2. Descargue el diccionario de RockyouLinks to an external site. (utilizado ampliamente en el mundo del pentesting). Haga un script que para cada string contenido en el diccionario, reemplace la primera letra por su letra en capital y agregue un cero al final de la password.
3. Todos los strings que comiencen con número toca eliminarlos del diccionario. Indique la cantidad de contraseñas que contiene el diccionario modificado debe llamarse rock-you\_mod.dic A continuación un ejemplo de cómo se modifican las 10 primeras líneas del diccionario original.

## 2. Desarrollo (PASO 1)

Para la realización de este paso, se debe instalar aircrack-ng con el siguiente comando:

```
sudo apt-get install aircrack-ng
```

Luego, es necesario identificar la interfaz utilizando el comando `iwconfig`. En mi caso, la interfaz se llama `wlp1s0`. A continuación, se inicia la interfaz en modo monitor con el comando:

```
sudo airmon-ng start wlp1s0
```

Después, se realiza un escaneo del aire para identificar las diferentes redes presentes en la zona con el comando:

```
sudo airodump-ng wlp1s0
```

## 2.1 Identificar en qué se destaca la red del informante del resto DESARROLLO (PASO 1)

### 2.1. Identificar en qué se destaca la red del informante del resto

Después de escanear la zona, la red del informante se destaca del resto en los campos de `encode`, `cipher` y `ssid`, ya que en estos tres campos tiene el valor de WEP, como se puede ver en la siguiente imagen:

BSSID	PWR	Beacons	#Data, #/s		CH	MB	ENC	CIPHER	AUTH	ESSID
AC:F8:CC:1D:60:60	-77	1	0	0	1	130	WPA2	CCMP	PSK	VTR-8492879
B0:1F:8C:E2:14:A4	-67	2	0	0	11	130	WPA3	CCMP	OWE	<length: 0>
5C:03:39:0C:94:42	-1	0	8	3	5	-1	WPA			<length: 0>
3C:84:6A:87:7B:6E	-81	2	0	0	10	270	WPA2	CCMP	PSK	TP-Link_7B6E
E6:AB:89:1C:85:38	-1	0	0	0	10	-1				<length: 0>
36:71:AE:1E:F3:95	-40	7	0	0	11	130	WPA2	CCMP	PSK	Grumbly's Wifi
B0:48:7A:D2:DD:74	-46	8	301	68	6	54e	WEP	WEP		WEP
98:FC:11:86:B6:B9	-55	4	3	0	6	130	WPA2	CCMP	PSK	Telematica

Figura 1: Descripción de la imagen.

El BSSID de la red obtenida es B0:48:7A:D2:DD:74 y se encuentra en el canal 6.

A continuación, se inicia la captura de tráfico de esa red en específico con el comando:

```
sudo airodump-ng -c 6 --bssid B0:48:7A:D2:DD:74 -w captura wlp1s0mon
```

Este comando indica el canal y el BSSID que se escuchará por la interfaz en modo monitor `wlp1s0mon`. La captura se almacena en un archivo llamado `captura`.

### 2.2. Explicación Matemática de por qué se requieren más de 5000 paquetes para obtener la contraseña

Para explicar matemáticamente por qué se necesitan más de 5000 paquetes para obtener la contraseña en una red WEP, particularmente en relación con los Initialization Vectors (IV) de 24 bits y una longitud total de clave de 64 o 128 bits, se utiliza el concepto de la paradoja del cumpleaños.

En una red WEP, los IV forman parte de la información empleada en el cifrado de los paquetes. Los IV se generan de forma aleatoria y comienzan a repetirse después de cierta cantidad de paquetes debido a su longitud de 24 bits. Esto significa que, después de aproximadamente  $2^{24}$  paquetes (alrededor de 16.7 millones), los IV se repiten.

La paradoja del cumpleaños se aplica en este contexto para calcular la probabilidad de que dos IV idénticos se generen en un conjunto dado de paquetes. La fórmula para esta probabilidad se expresa de la siguiente manera:

$$P(N) = 1 - e^{\frac{-N(N-1)}{2M}}$$

Donde:

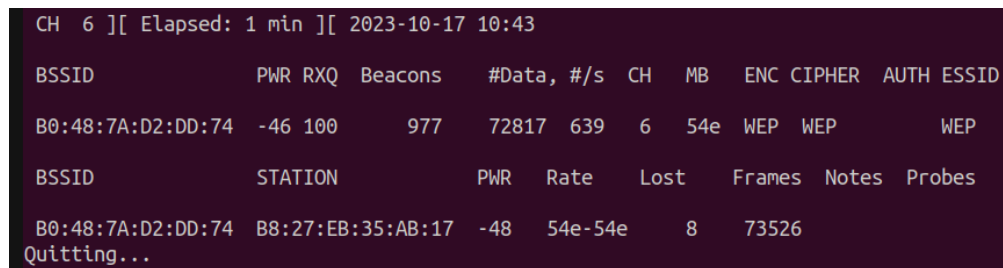
- $P(N)$  es la probabilidad de que se genere un IV duplicado después de  $N$  paquetes (en este caso, buscamos al menos el 50 % de probabilidad).

## 2.3 Obtención de la Contraseña con Ataque por Defecto de Aircrack-ng (PASO 1)

- $N$  es el número de paquetes (la incógnita).
- $M$  es el número total de posibles valores de IV, que es igual a  $2^{24}$ .

Resolviendo la fórmula para  $N$ , se obtiene  $N \approx 4823$ , lo que significa que se necesitan alrededor de 5000 paquetes para alcanzar una probabilidad del 50 % de generar un IV duplicado.

En el caso de la experiencia, capturé 73526 frames, lo que prácticamente garantiza la probabilidad de encontrar la contraseña, ya que la probabilidad es casi del 100 %.



```
CH 6 ][ Elapsed: 1 min ][ 2023-10-17 10:43
```

BSSID	PWR	RXQ	Beacons	#Data, #/s	CH	MB	ENC	CIPHER	AUTH	ESSID
B0:48:7A:D2:DD:74	-46	100	977	72817 639	6	54e	WEP	WEP		WEP

BSSID	STATION	PWR	Rate	Lost	Frames	Notes	Probes
B0:48:7A:D2:DD:74	B8:27:EB:35:AB:17	-48	54e-54e	8	73526		

Quitting...

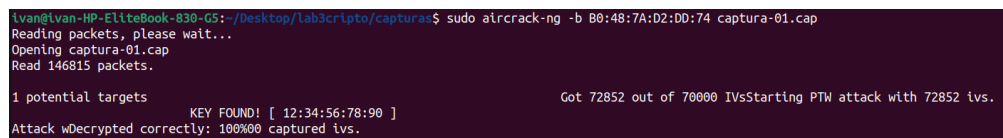
Figura 2: Captura de paquetes.

## 2.3. Obtención de la Contraseña con Ataque por Defecto de Aircrack-ng

Mediante el siguiente comando:

```
sudo aircrack-ng -b B0:48:7A:D2:DD:74 captura-01.cap
```

donde se especifica el BSSID de la red WEP y también se utiliza la captura previamente realizada en el archivo captura-01.cap, Aircrack-ng logra encontrar la contraseña en formato hexadecimal, que es 12:34:56:78:90.



```
Ivan@ivan-HP-EliteBook-830-G5: ~/Desktop/lab3cripto/captura$ sudo aircrack-ng -b B0:48:7A:D2:DD:74 captura-01.cap
Reading packets, please wait...
Opening captura-01.cap
Read 146815 packets.

1 potential targets
KEY FOUND! [ 12:34:56:78:90 ]
Attack wDecrypted correctly: 100%00 captured ivs.
Got 72852 out of 70000 IVsStarting PTW attack with 72852 ivs.
```

Figura 3: Obtención de la clave mediante Aircrack-ng.

## 2.4. Indicación del Tiempo Empleado para Obtener la Contraseña

Para obtener el tiempo que Aircrack-ng demoró en obtener la contraseña, utilicé el mismo comando, pero precediéndolo con el comando ‘time’. La instrucción completa se ve de la siguiente manera:

```
time sudo aircrack-ng -b B0:48:7A:D2:DD:74 captura-01.cap
```

El resultado de este comando mostró un tiempo real de ejecución de 0.854 segundos, lo que representa el tiempo que tomó el proceso de obtención de la contraseña.

```
ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto/capturas$ time sudo aircrack-ng -b B0:48:7A:D2:DD:74 captura-01.cap
Reading packets, please wait...
Opening captura-01.cap
Read 146815 packets.

1 potential targets
KEY FOUND! [ 12:34:56:78:90 ]
Got 72852 out of 70000 IVsStarting PTW attack with 72852 ivs.
Attack wDecrypted correctly: 100%00 captured ivs.

real    0m0.854s
user    0m0.008s
sys     0m0.022s
```

Figura 4: Obtención de la clave mediante Aircrack-ng, con el tiempo de ejecución incluido.

## 2.5. Descifrado del Contenido Capturado

Una vez obtenida la contraseña, se procede al descifrado de la captura realizada mediante el siguiente comando:

```
sudo airdecap-ng -w 12:34:56:78:90 captura-01.cap
```

En este comando, se indica la contraseña y el archivo de captura, y como resultado se obtiene un nuevo archivo llamado captura-01-dec.cap que contiene el contenido descifrado.

```
ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto/capturas$ sudo airdecap-ng -w 12:34:56:78:90 captura-01.cap
Total number of stations seen      1
Total number of packets read      146815
Total number of WEP data packets   72920
Total number of WPA data packets   0
Number of plaintext data packets   0
Number of decrypted WEP packets    72920
Number of corrupted WEP packets    0
Number of decrypted WPA packets    0
Number of bad TKIP (WPA) packets  0
Number of bad CCMP (WPA) packets  0
```

Figura 5: Descifrado de la captura realizada utilizando la contraseña obtenida.

## 2.6. Descripción de cómo se Obtiene la URL para Descargar el Archivo

La captura descifrada, mostrada a continuación, consta de múltiples paquetes ICMP generados por varios pings realizados como parte del experimento. En la carga de datos de cada paquete, al final de la misma, se encuentra la URL en texto plano. La URL que se ha encontrado en la captura es la siguiente: `bit.ly/wpa2_`

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10374/34344, ttl=64
2	0.000059	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10375/34600, ttl=64 (reply in 3)
3	0.000114	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10375/34600, ttl=64 (request in 2)
4	0.000153	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10376/34856, ttl=64
5	0.000189	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10377/35112, ttl=64 (no response found!)
6	0.000225	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10378/35368, ttl=64 (reply in 7)
7	0.000267	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10378/35368, ttl=64 (request in 6)
8	0.000304	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10379/35624, ttl=64 (reply in 9)
9	0.000340	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10379/35624, ttl=64 (request in 8)
10	0.000375	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10380/35880, ttl=64 (reply in 11)
11	0.000419	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10380/35880, ttl=64 (request in 10)
12	0.000455	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10381/36136, ttl=64 (reply in 13)
13	0.000491	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10381/36136, ttl=64 (request in 12)
14	0.000517	192.168.11.3	192.168.11.1	ICMP	54	Echo (ping) request id=0x0002, seq=10393/39208, ttl=64 (reply in 15)
15	0.000560	192.168.11.1	192.168.11.3	ICMP	54	Echo (ping) reply id=0x0002, seq=10393/39208, ttl=64 (request in 14)
▶ Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0 ▶ Ethernet II, Src: Tp-LinkT_d2:dc:74 (b8:48:7a:d2:dc:74), Dst: Raspberr_35:ab:17 (b8:27:eb:35:ab:17) ▶ Internet Protocol Version 4, Src: 192.168.11.1, Dst: 192.168.11.3 ▶ Internet Control Message Protocol Type: 0 (Echo (ping) reply) Code: 0 Checksum: 0xc22e [correct] [Checksum Status: Good] Identifier (BE): 2 (0x0002) Identifier (LE): 512 (0x0200) Sequence Number (BE): 10374 (0x2886) Sequence Number (LE): 34344 (0x8628) Data (12 bytes) Data: 6269742e6c792f77061325f [Length: 12]						
0000	b8 27 eb 35 ab 17 b8 48 7a d2 dc 74 08 00 45 00	. . . . . H z . t . E .				
0010	00 28 5b cc 00 00 40 01 87 b4 c0 a8 0b 01 c9 a8	. ([ . . . . . 0 . . . . .				
0020	0b 03 00 00 c2 2e 00 02 28 06 02 09 74 20 0c 75	. . . . . (- . . . . .				
0030	2f 77 70 01 32 5f	/wpa2				

Figura 6: Captura descifrada.

Al acceder a la URL encontrada, se obtiene una segunda captura de tráfico que se muestra a continuación:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	ee:de:67:8c:df:8b	Tp-LinkT_d2:dc:18	802.11	123	Association Request, SN=2292, FN=0, Flags=....., SSID=VTR-1645213
2	0.000002	ee:de:67:8c:df:8b	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
3	0.002401	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	802.11	102	Association Response, SN=1184, FN=0, Flags=.....
4	0.002402	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
5	0.007381	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	EAPOL	133	Key (Message 1 of 4)
6	0.009336	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
7	0.017080	ee:de:67:8c:df:8b	Tp-LinkT_d2:dc:18	EAPOL	155	Key (Message 2 of 4)
8	0.017082	ee:de:67:8c:df:8b	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
9	0.017087	ee:de:67:8c:df:8b	ee:de:67:8c:df:8b	802.11	10	Clear-to-send, Flags=.....
10	0.050774	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	EAPOL	189	Key (Message 3 of 4)
11	0.050776	Tp-LinkT_d2:dc:18	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
12	0.054559	ee:de:67:8c:df:8b	Tp-LinkT_d2:dc:18	EAPOL	133	Key (Message 4 of 4)
13	0.054560	ee:de:67:8c:df:8b	ee:de:67:8c:df:8b	802.11	10	Acknowledgement, Flags=.....
▶ Frame 1: 123 bytes on wire (984 bits), 123 bytes captured (984 bits) on interface 0 ▶ IEEE 802.11 Association Request, Flags: ..... ▶ IEEE 802.11 Wireless Management						
0000	00 00 3a 01 b0 48 7a d2 dc 18 ee de 67 8c df 8b	. . . . . H z . t . E .				
0010	00 48 7a d2 dc 18 40 0f 31 04 01 00 00 00 50 54	. H z . ( . . . . . V T				
0020	52 2d 31 36 34 35 32 31 33 01 00 02 04 00 90 0c	R -1645213				
0030	12 18 24 30 14 01 00 00 0f ac 04 01 00 00 0f ac	. \$0 . . . . .				
0040	04 01 00 00 0f ac 02 00 00 32 04 30 48 60 6c 3b	. . . . . 2 0H' L				
0050	10 51 51 53 54 73 74 75 76 77 78 7c 7d 7e 7f 80	..QGSTstu vwX )~				
0060	82 7f 85 84 00 00 00 01 dd 07 00 50 f2 02 00 01	. . . . . P . . . . .				
0070	00 dd 08 8c fd f0 01 01 02 01 00	. . . . .				

Figura 7: Captura descargada desde la URL encontrada.

### 3. Desarrollo (PASO 2)

En este paso, se utiliza y modifica el diccionario de contraseñas rockyou” para adaptarlo a la captura descargada.

### 3.1. Indicación de Script para Modificar el Diccionario Original

La modificación del diccionario rockyou consiste en cambiar la primera letra de cada contraseña a mayúscula, agregar un "0" al final de cada contraseña, eliminar todas las contraseñas que comiencen con un número y contar la cantidad de contraseñas en el nuevo diccionario modificado.

Para esto se realiza el siguiente script en python que genera el archivo rockyou\_mod.dic

```
with open('rockyou.txt', 'r', encoding='latin-1') as file:
    original_passwords = file.readlines()
password_nuevas = []
for password in original_passwords:
    password = password.strip()
    if password and not password[0].isdigit():
        password_nueva = password[0].upper() + password[1:] + '0'
        password_nuevas.append(password_nueva)

with open('rockyou_mod.dic', 'w') as file:
    for password in password_nuevas:
        file.write(password + '\n')

print(f'El diccionario modificado tiene {len(password_nuevas)} contraseñas.')
```

### 3.2 Cantidad de Contraseñas Finales en el Diccionario rockyou\_mod.dic DESARROLLO (PASO 2)

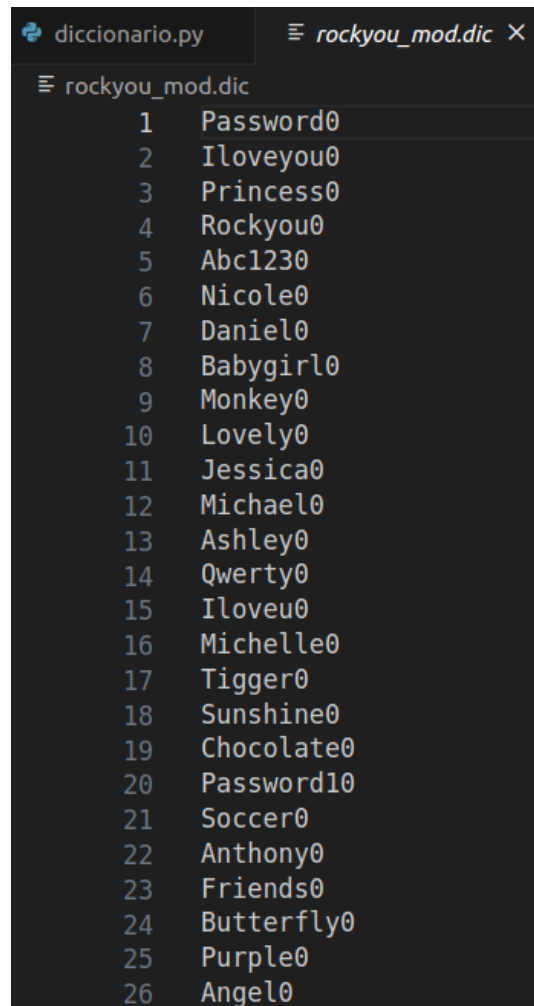


Figura 8: Algunas de las contraseñas que contiene el diccionario modificado.

### 3.2. Cantidad de Contraseñas Finales en el Diccionario rockyou\_mod.dic

Una vez que el script se ha ejecutado con éxito, se genera el archivo de diccionario modificado “rockyou\_mod.dic”. En la pantalla se imprime la cantidad de contraseñas que contiene, como se muestra en la siguiente captura:

```
ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto$ /bin/python3 /home/ivan/Desktop/lab3cripto/diccionario.py
El diccionario modificado tiene 11059798 contraseñas.
```

Figura 9: Cantidad de contraseñas finales en el archivo “rockyou\_mod.dic”.



## 4. Desarrollo (Paso 3)

### 4.1. Obtiene contraseña con Hashcat con potfile

Para trabajar con la captura en Hashcat, primero es necesario convertirla a un formato .hc22000, lo que resulta en el archivo 189978\_1697597121.hc22000. Para convertirlo se usa la página <https://hashcat.net/cap2hashcat/>.

Luego, para trabajar con Hashcat, se debe instalar la herramienta. Se puede hacer, con el siguiente comando:

```
sudo apt install hashcat
```

Una vez que todo está instalado y listo, se utiliza Hashcat para obtener la contraseña utilizando un “potfile”. Un “potfile” es un archivo utilizado por herramientas de recuperación de contraseñas, como Hashcat, para registrar y almacenar contraseñas descifradas o “crackeadas” durante un proceso de ataque.

El siguiente comando se utiliza para realizar el proceso de desciframiento, Hashcat registrará las contraseñas descifradas en el archivo “potfile.txt” .:

```
hashcat -m 22000 189978_1697597121.hc22000 rockyou_mod.dic --potfile-path  
potfile.txt --force
```

```

ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto$ hashcat -m 22000 189978_1697597121.hc22000 rockyou_mod.dic --potfile-path potfile.txt --force
hashcat (v6.2.5) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

OpenCL API (OpenCL 2.0 pocl 1.8 Linux, None+Asserts, RELOC, LLVM 11.1.0, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: pthread-Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz, 6858/13781 MB (2048 MB allocatable), 8MCU

Minimum password length supported by kernel: 8
Maximum password length supported by kernel: 63

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt
* Slow-Hash-SIMD-LOOP

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache built:
* Filename..: rockyou_mod.dic
* Passwords.: 11059725
* Bytes.....: 120106275
* Keyspace...: 11059707
* Runtime....: 1 sec

1813acb976741b446d43369fb96dbf90:b0487ad2dc18:eede678cdf8b:VTR-1645213:Security0

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 22000 (WPA-PBKDF2-PMKID+EAOL)
Hash.Target.....: 189978_1697597121.hc22000
Time.Started.....: Wed Oct 18 00:12:21 2023, (1 sec)
Time.Estimated...: Wed Oct 18 00:12:22 2023, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (rockyou_mod.dic)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 994 H/s (9.58ms) @ Accel:32 Loops:512 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 2907/11059707 (0.03%)
Rejected.....: 1371/2907 (47.16%)
Restore.Point....: 2443/11059707 (0.02%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: Lacrosse0 -> Dangerous0
Hardware.Mon.#1..: Temp: 43c Util: 85%

Started: Wed Oct 18 00:09:41 2023
Stopped: Wed Oct 18 00:12:25 2023

```

Figura 10: Hashcat con potfile.

```

1813acb976741b446d43369fb96dbf90:b0487ad2dc18:eede678cdf8b:VTR-1645213:Security0

```

Figura 11: Contraseña crackeada con hashcat con potfile.

Finalmente se puede observar que la contraseña crackeada con potfile fue “Security0”

## 4.2. Identifica nomenclatura del output

**Device:** Describe el dispositivo utilizado para el ataque. En este caso, se está utilizando un dispositivo Intel Core i5-8350U CPU.

**Compatibilidad con la longitud de contraseña:** Muestra la longitud mínima y máxima de contraseña soportada por el kernel de Hashcat.

**Hashes:** Indica el número de "digests" (hashes) y la cantidad de unique digests utilizados en el ataque.

**Rules:** Muestra cuántas reglas se están aplicando en el ataque.

**Optimizers applied:** Enumera los optimizadores que se han aplicado en el ataque.

**Watchdog:** Establece un límite de temperatura para detener el ataque si la temperatura del hardware alcanza un valor crítico (90°C en este caso).

**Host memory required for this attack:** Indica la cantidad de memoria del sistema requerida para el ataque (2 MB en este caso).

**Dictionary cache built:** Proporciona información sobre el diccionario utilizado, el número de contraseñas en el diccionario, el tamaño en bytes y el espacio de claves generado.

**Hash descifrado:** Muestra el hash descifrado y otra información relacionada.

**Session:** Proporciona detalles generales de la sesión, incluyendo el modo de hash, el archivo de destino y la hora de inicio y finalización del ataque.

**Speed:** Muestra la velocidad de desciframiento actual y otros detalles relacionados con el rendimiento del ataque.

**Started / Stopped:** Muestra cuándo se inició y finalizó el ataque.

### 4.3. **Obtiene contraseña con Hashcat sin potfile**

Para obtener la contraseña con Hashcat sin utilizar un potfile, el proceso es similar al paso anterior, pero en este caso, se asigna la opción '-potfile-disable' para deshabilitar el uso del potfile. Se usa el siguiente comando:

```
hashcat -m 22000 189978_1697597121.hc22000 rockyou_mod.dic --potfile-disable
```

Este comando ejecutará Hashcat en el modo 22000, pero deshabilitará el registro de las contraseñas descifradas en el potfile durante el ataque.

```

ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto$ hashcat -m 22000 189978_1697597121.hc22000 rockyou_mod.dic --potfile-disable
hashcat (v6.2.5) starting

OpenCL API (OpenCL 2.0 pocl 1.8 Linux, None+Asserts, RELOC, LLVM 11.1.0, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: pthread-Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz, 6858/13781 MB (2048 MB allocatable), 8MCU

Minimum password length supported by kernel: 8
Maximum password length supported by kernel: 63

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Single-Hash
* Single-Salt
* Slow-Hash-SIMD-LOOP

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 2 MB

Dictionary cache hit:
* Filename..: rockyou_mod.dic
* Passwords.: 11059707
* Bytes.....: 120106275
* Keyspace..: 11059707

1813acb976741b446d43369fb96dbf90:b0487ad2dc18:eede678cdf8b:VTR-1645213:Security0

Session.....: hashcat
Status.....: Cracked
Hash.Mode.....: 22000 (WPA-PBKDF2-PMKID+EAPOL)
Hash.Target.....: 189978_1697597121.hc22000
Time.Started....: Wed Oct 18 01:04:50 2023 (1 sec)
Time.Estimated...: Wed Oct 18 01:04:51 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (rockyou_mod.dic)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 3319 H/s (9.34ms) @ Accel:256 Loops:64 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 3817/11059707 (0.03%)
Rejected.....: 1769/3817 (46.35%)
Restore.Point....: 0/11059707 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: Password0 -> PASSWORD10
Hardware.Mon.#1..: Temp: 41c Util: 26%

Started: Wed Oct 18 01:04:46 2023
Stopped: Wed Oct 18 01:04:52 2023

```

Figura 12: Contraseña crackeada con hashcat sin potfile.

La ejecución de este crack fue mucho más rápida que la anterior, esto se debe a que el diccionario fue almacenado en caché y ahora se está rescatando desde la caché, llegando al mismo resultado, la contraseña “Security0”.

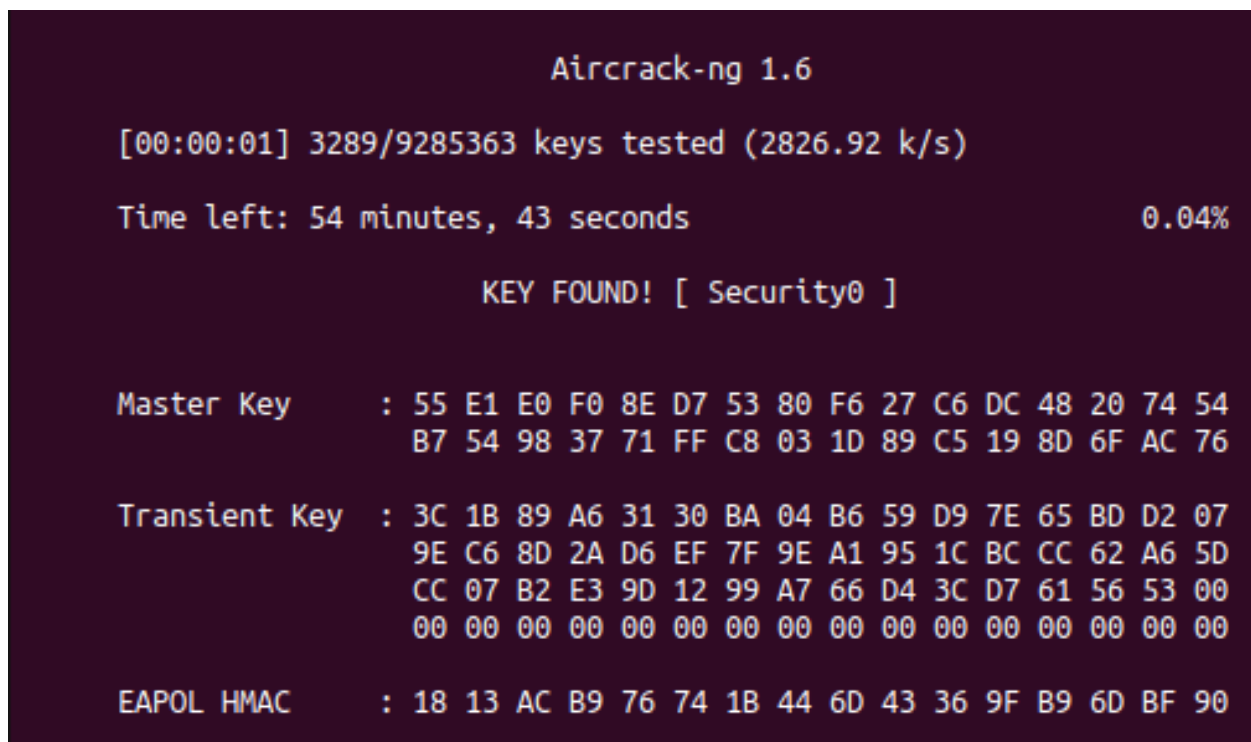
#### 4.4. Identifica nomenclatura del output

La nomenclatura del output es casi la misma, se observan los mismos campos de datos, sin embargo como ya se mencionó hay una diferencia en la carga del diccionario, ya que está vez se accede a la caché ya creada mientras que antes se debió crear el diccionario en la caché. Además en esta ejecución no se crea un archivo potfile.txt con la contraseña obtenida

## 4.5. Obtiene contraseña con aircrack-ng

Para utilizar aircrack se debe trabajar con la captura pero con formato .pcap y se usa el siguiente comando:

```
sudo aircrack-ng -a2 -w rockyou_mod.dic handshake.pcap
```



```

Aircrack-ng 1.6

[00:00:01] 3289/9285363 keys tested (2826.92 k/s)

Time left: 54 minutes, 43 seconds                                0.04%

KEY FOUND! [ Security0 ]

Master Key      : 55 E1 E0 F0 8E D7 53 80 F6 27 C6 DC 48 20 74 54
                  B7 54 98 37 71 FF C8 03 1D 89 C5 19 8D 6F AC 76

Transient Key   : 3C 1B 89 A6 31 30 BA 04 B6 59 D9 7E 65 BD D2 07
                  9E C6 8D 2A D6 EF 7F 9E A1 95 1C BC CC 62 A6 5D
                  CC 07 B2 E3 9D 12 99 A7 66 D4 3C D7 61 56 53 00
                  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

EAPOL HMAC      : 18 13 AC B9 76 74 1B 44 6D 43 36 9F B9 6D BF 90
  
```

Figura 13: Contraseña crackeada con aircrack.

Se obtiene la contraseña “Security0” una vez intentó con 3289 contraseñas.

## 4.6. Identifica y modifica parámetros solicitados por pycrack

Se realizaron modificaciones en varios campos, como el SSID, aNonce, sNonce, apMac y cliMac, además de ajustar los valores de mic y data de los paquetes específicos. Se obtiene la data requerida y se sustituye en el código pywd.py.



```

Tag: SSID parameter set: VTR-1645213
  
```

Figura 14: SSID.

#### 4.6 Identifica y modifica parámetros solicitados por pycrack 4 DESARROLLO (PASO 3)

```
WPA Key Nonce: 4c2fb7eca28fba45accefd3ac5e433314270e04355b6d95086031b004a31935
```

Figura 15: aNonce.

```
WPA Key Nonce: 30bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c038f5869b96bbe9cdc
```

Figura 16: sNonce.

```
Transmitter address: ee:de:67:8c:df:8b (ee:de:67:8c:df:8b)
Destination address: Tp-LinkT_d2:dc:18 (b0:48:7a:d2:dc:18)
```

Figura 17: apMac y cliMac.

```
WPA Key MIC: 1813acb976741b446d43369fb96dbf90
```

Figura 18: Primer MIC.

```
WPA Key MIC: a349d01089960aa9f94b5857b0ea10c6
```

Figura 19: Segundo MIC.

```
WPA Key MIC: 5cf0d63af458f13a83daa686df1f4067
```

Figura 20: Tercer MIC.

#### 4.6 Identifica y modifica parámetros solicitados por pycrack 4 DESARROLLO (PASO 3)

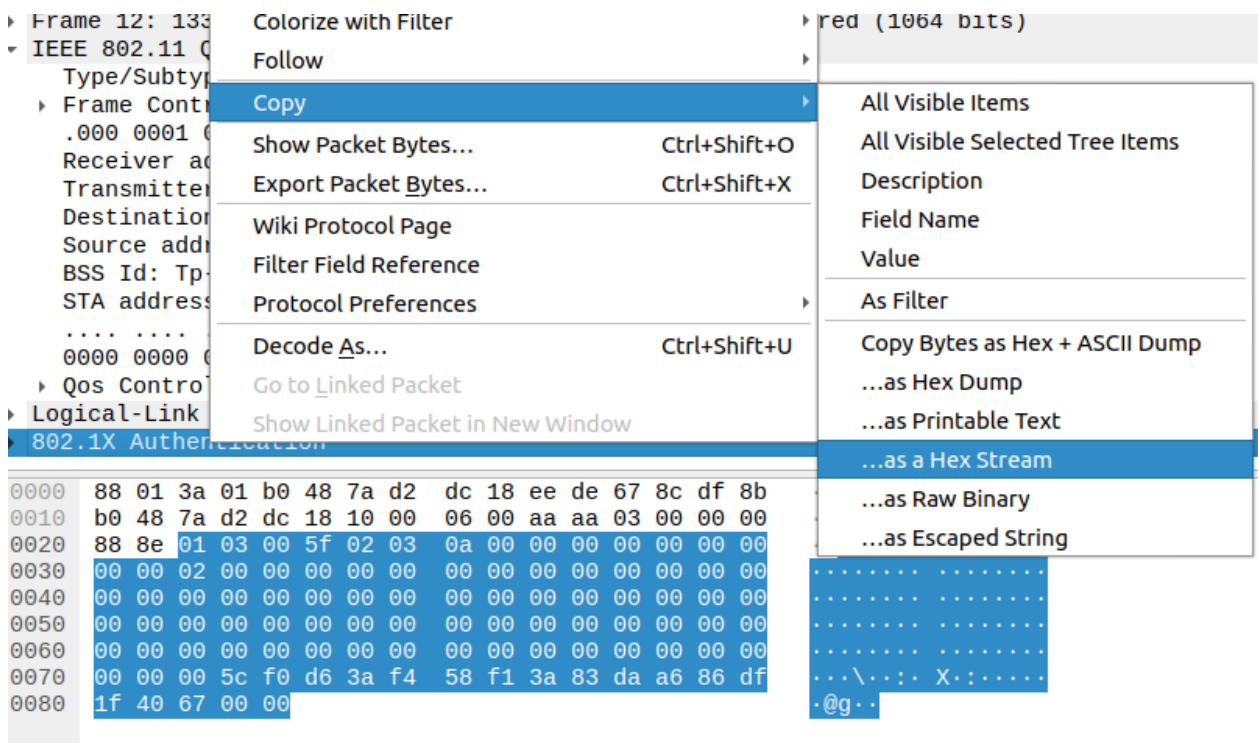


Figura 21: Data, esta es 1 de las 3 necesarias.

[illegible]

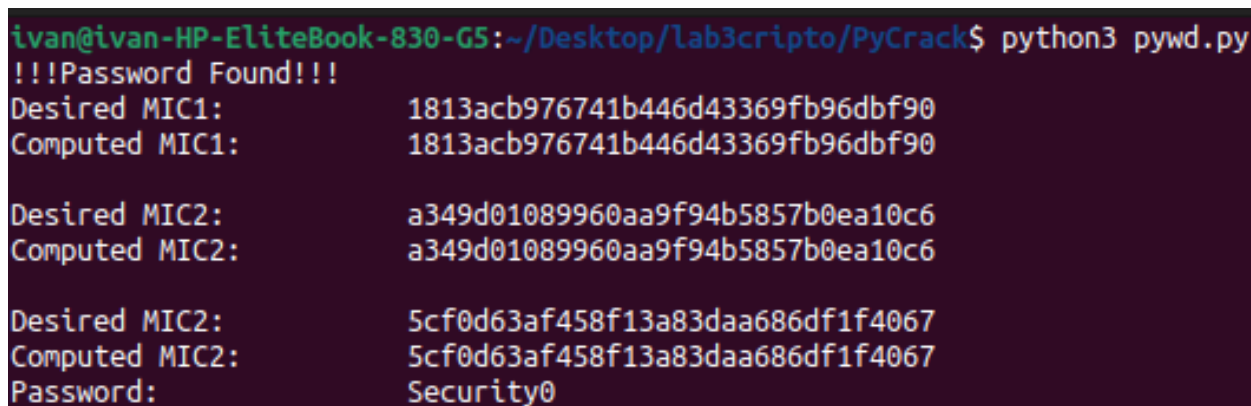
Figura 22: Código pycrack con parámetros modificados.

Todos los parámetros mencionados anteriormente se insertan directamente en el código

excepto el campo data, en donde se requiere una modificación más, los valores del MIC correspondiente se reemplazan con ceros, la misma cantidad de ceros que el largo original de cada MIC.

## 4.7. Obtiene contraseña con pycrack

Se ejecuta el código previamente modificado con `python3 pywd.py`



```
ivan@ivan-HP-EliteBook-830-G5:~/Desktop/lab3cripto/PyCrack$ python3 pywd.py
!!!Password Found!!!
Desired MIC1:      1813acb976741b446d43369fb96dbf90
Computed MIC1:     1813acb976741b446d43369fb96dbf90

Desired MIC2:      a349d01089960aa9f94b5857b0ea10c6
Computed MIC2:     a349d01089960aa9f94b5857b0ea10c6

Desired MIC2:      5cf0d63af458f13a83daa686df1f4067
Computed MIC2:     5cf0d63af458f13a83daa686df1f4067
Password:          Security0
```

Figura 23: Contraseña crackeada con pycrack.

Se obtiene que la contraseña es “Security0” al igual que en los métodos anteriores.

## Conclusiones y comentarios

En este laboratorio, se exploraron herramientas específicas para identificar vulnerabilidades en redes inalámbricas. Se destaca la obsolescencia del cifrado WEP y se utilizaron herramientas como Aircrack-ng, Hashcat y el script pycrack para demostrar su fragilidad.

Estas herramientas han permitido entender por qué el cifrado WEP no se recomienda en la actualidad, ya que su seguridad es insuficiente. La facilidad con la que se pudo descifrar una red WEP muestra sus serias vulnerabilidades. Cualquier persona con conocimientos y acceso a estas herramientas podría comprometer la seguridad de dicha red.

Este laboratorio resalta la importancia de utilizar cifrados más seguros, como WPA2 o WPA3, para proteger la privacidad y la confidencialidad de los datos en las redes inalámbricas.