Informe Laboratorio 3

Sección 2

Claudio Lopez
e-mail: claudio.lopez1@mail.udp.cl

Mayo de 2023

Índice

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 la 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 rockyou_mod.dic A continuación un ejemplo de cómo se modifican las 10 primeras líneas del diccionario original.

2. Desarrollo (PASO 1)

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

Para identificar la red, se instaló aircrack-ng, con el fin de acceder al modo monitor y realizar las siguientes actividades, para esto se utilizó el siguiente comando **sudo apt-get install aircrack-ng**. Posteriormente, se identificó la interfaz de la tarjeta wifi del equipo, el cual es **wlp2s0** y con el comando **sudo airmon-ng start wlp2s0**. Luego se utilizó la herramienta **Airodump-ng** con el siguiente comando **sudo airodump-ng wlp2s0mon -w cripto2**, con el fin de capturar paquetes y guardarlos en un archivo .cap, el resultado fue el referenciado en la figura 1.

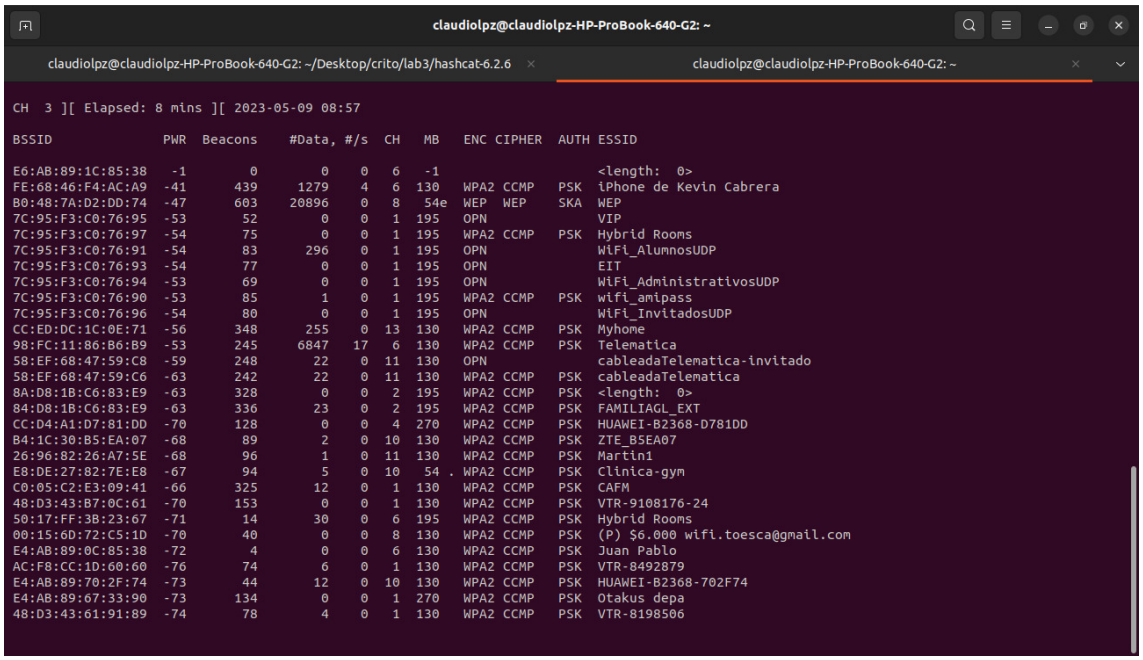

Figura 1: Captura de las redes a través de airodump.

Para finalizar esta sección se logró identificar la red del informante, la cual es **WEP**, se llegó a esta conclusión debido a que es la única red con encriptación WEP y además genera demasiado tráfico en comparación a las otras redes.

2.2. Explica matemáticamente por qué se requieren más de 5000 paquetes para obtener la pass

Se requieren más de 5000 paquetes debido a que WEP utiliza vectores de inicialización (IVs) de 24 bits, el problema que puede tener esto es la repetición de estos vectores en una red activa. como es de 24 bits, se tiene que el resultado de la probabilidad de encontrar vectores iguales es igual a lo mostrado en la ecuacion 1.

$$1 - [(2^{24} - 1)/2^{24}]. \tag{1}$$

Por lo tanto, el número esperado de IVs repetidos es de Probabilidad de IVs repetidos * Número total de paquetes capturados, para este caso el resultado se muestra en la ecuacion 2.

$$[1 - ((2^{24} - 1)/2^{24})] * 5000] \tag{2}$$

Por lo tanto, entre más paquetes, hay más probabilidad de encontrar un IV repetido.

2.3. obtiene la password con ataque por defecto de aircrack-ng

Para obtener la password con ataque por defecto de aircrack, se utilizó el siguiente comando **sudo aircrack-ng -b B0:48:7A:D2:DD:74 cripto2.cap**, como se muestra en la

figura 2.

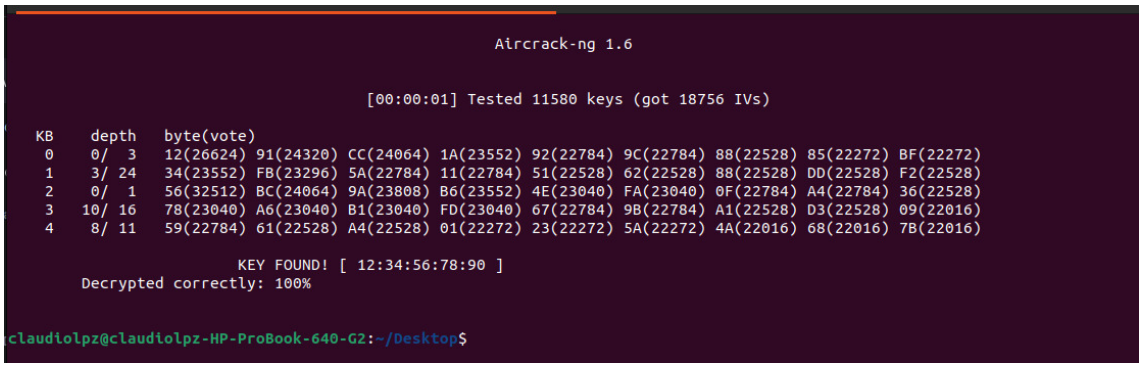

Figura 2: Búsqueda de la password de la red WEP.

La password que se encontró fue la siguiente **12:34:56:78:90**

2.4. indica el tiempo que demoró en obtener la password

Como indica la figura 2, el tiempo que se demoró para poder encontrar la Key fue de 1 segundo.

2.5. descifra el contenido capturado

Para descifrar se utiliza él comandó **sudo airdecap-ng -w 12:34:56:78:90 cripto2-01.cap**, una vez ejecutado este comando, se crea un archivo con los paquetes descifrados, el resultado se muestra en la figura 3.

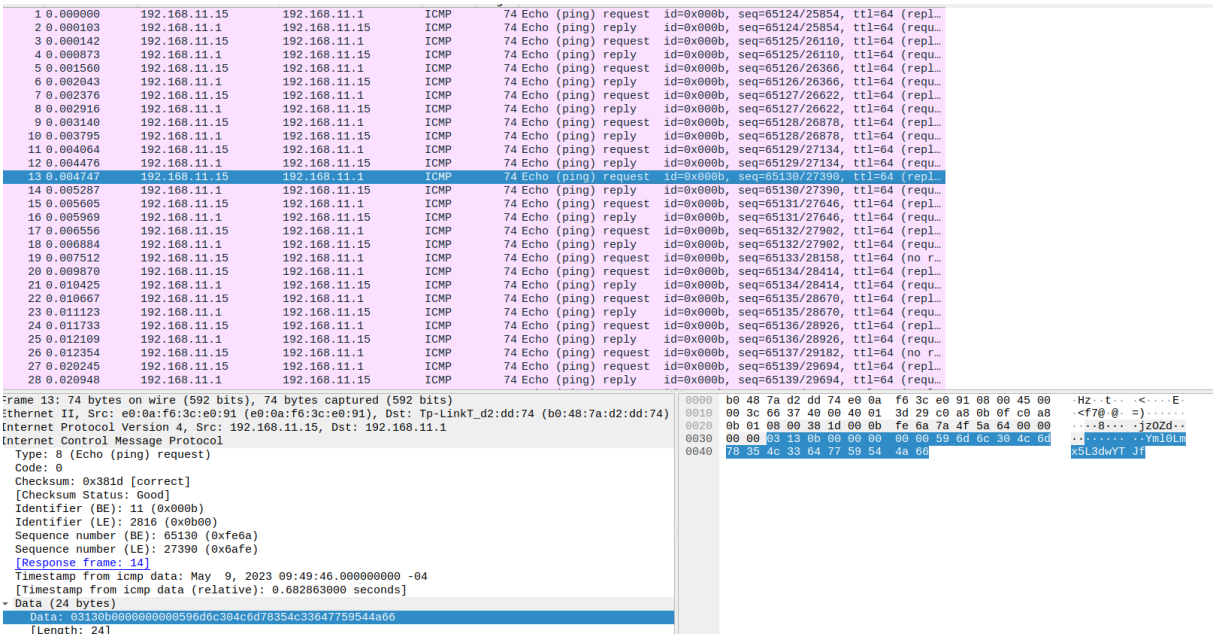

Figura 3: Captura de paquetes, descifrados.

2.6. describe como obtiene la url de donde descargar el archivo

Se observa que los paquetes de la captura del punto anterior, el texto que se repite es **Yml0Lmx5L3dwYTJf**, luego se procedió a decodificar el texto, el cual está en base 64 y se obtuvo el siguiente link acortado **http://bit.ly/wpa2_**, el cual lleva al link **https://www.cloudshark.org/captures/b5b39e1c51eb**. Este punto se puede evidenciar en las imágenes 3, 4 y 5.

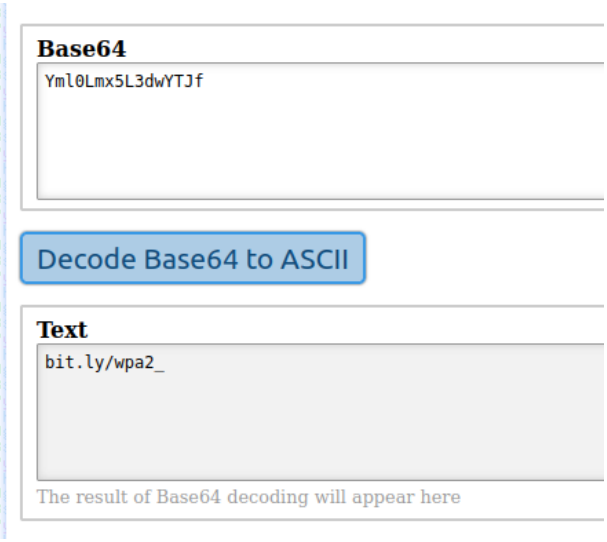

Figura 4: Decode de Base64 a ASCII.

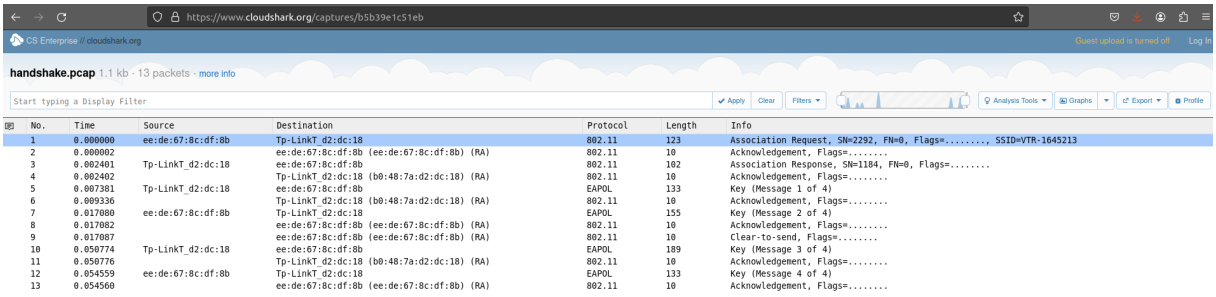

Figura 5: Página enviada por el informante.

3. Desarrollo (PASO 2)

3.1. indica script para modificar diccionario original

Se logró realizar el script mostrado en la figura 6 para modificar el diccionario.

```python
import re

def modificar_diccionario(diccionario):
    diccionario_modificado = diccionario.copy()
    for clave, valor in diccionario.items():
        if not re.match(r'^[a-zA-Z]', valor):
            del diccionario_modificado[clave]
        else:
            diccionario_modificado[clave] = valor.capitalize() + '0'
    return diccionario_modificado

def guardar_diccionario_modificado(diccionario, nombre_archivo):
    with open(nombre_archivo, 'w') as archivo:
        contador = 0
        for valor in diccionario.values():
            archivo.write(valor + '\n')
            contador += 1
    print(f"Se guardaron {contador} strings en el archivo {nombre_archivo}")


# Carga el diccionario desde un archivo de texto
def cargar_diccionario(nombre_archivo):
    diccionario = {}
    with open(nombre_archivo, 'r', encoding='latin-1') as archivo:
        for linea in archivo:
            linea = linea.strip()
            diccionario[linea] = linea
    return diccionario


# Nombre del archivo de entrada
archivo_entrada = "rockyou.txt"
# Nombre del archivo de salida modificado
archivo_salida = "rockyou_mod.dic"

# Cargar el diccionario original
diccionario_original = cargar_diccionario(archivo_entrada)

# Modificar el diccionario
diccionario_modificado = modificar_diccionario(diccionario_original)

# Guardar el diccionario modificado en un archivo
guardar_diccionario_modificado(diccionario_modificado, archivo_salida)

```

Figura 6: Script para modificar el diccionario.

3.2. cantidad de passwords finales que contiene rockyou_mod.dic

Se logró obtener un total de **10956580** contraseñas en el diccionario. Esto se puede evidenciar en la figura 7.

```
claudio@claudio-B365-M-AORUS-ELITE:~/Escritorio/cripto/lab3$ /bi
Se guardaron 10956580 strings en el archivo rockyou_mod.dic
claudio@claudio-B365-M-AORUS-ELITE:~/Escritorio/cripto/lab3$
```

Figura 7: Ejecución del script.

4. Desarrollo (Paso 3)

4.1. obtiene contraseña con hashcat con potfile

Para este punto se utilizó aircrack para pasar el archivo handshake.pcap a un archivo con la extension hccapx, el comando utilizado fue el siguiete: **aircrack-ng handshake.pcap -j converted1**. Luego se instaló hashcat a través de GitHub, luego se utilizó el comando **hashcat -m 2500 –deprecated-check-disable converted1.hccapx rockyou_mod.dic -o /home/claudio/Escritorio/cripto/lab3/hashcat.potfile**, con el fin de obtener la contraseña con potfile. Se obtuvo el siguiente resultado (figura 8).

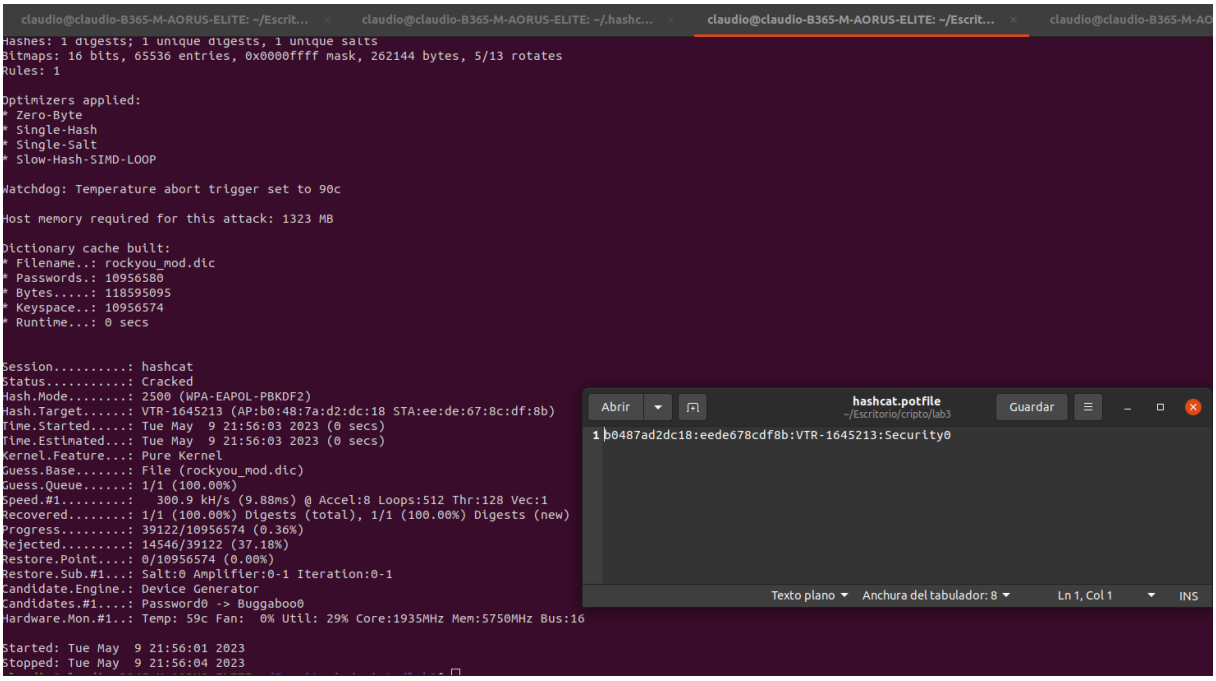

Figura 8: Obtención de contraseña con potfile.

4.2. identifica nomenclatura del output

En el archivo hashcat.potfile visto en la figura 8 se describen los siguientes campos, MAC del AP, MAC del cliente, SSID y contraseña (Security0).

4.3. obtiene contraseña con hashcat sin potfile

Para este punto se utilizó el comando **hashcat -m 2500 –deprecated-check-disable converted1.hccapx rockyou_mod.dic** y se obtuvo el siguiente resultado (figura 9)

```
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: 1323 MB

Dictionary cache built:
* Filename..: rockyou_mod.dic
* Passwords.: 10956580
* Bytes.....: 118595095
* Keyspace..: 10956574
* Runtime...: 0 secs

b0487ad2dc18:eede678cdf8b:VTR-1645213:Security0

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 2500 (WPA-EAPOL-PBKDF2)
Hash.Target......: VTR-1645213 (AP:b0:48:7a:d2:dc:18 STA:ee:de:67:8c:df:8b)
Time.Started.....: Tue May  9 21:57:37 2023 (0 secs)
Time.Estimated...: Tue May  9 21:57:37 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (rockyou_mod.dic)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   308.9 kH/s (9.72ms) @ Accel:64 Loops:256 Thr:32 Vec:1
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 75676/10956574 (0.69%)
Rejected.........: 26524/75676 (35.05%)
Restore.Point....: 0/10956574 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: Password0 -> Swimming!0
Hardware.Mon.#1..: Temp: 61c Fan:  0% Util:  7% Core:1935MHz Mem:5750MHz Bus:16

Started: Tue May  9 21:57:35 2023
Stopped: Tue May  9 21:57:38 2023
```

Figura 9: Obtención de contraseña sin potfile.

4.4. identifica nomenclatura del output

Se puede observar que al no utilizar potfile, este no se guarda en un archivo. Pero el output es mostrado en la consola, el punto importante para esta actividad está abajo del campo **Diccionary cache built**, se obtiene MAC del AP, MAC del cliente, SSID y contraseña (Security0).

4.5. obtiene contraseña con aircrack-ng

En este punto se procedió a obtener la contraseña con la herramienta aircrack-ng, el comando utilizado fue el siguiente: **aircrack-ng -w rockyou_mod.dic -b b0:48:7a:d2:dc:18**

handshake.pcap -l resultado.txt. La constraseña encontrada es la misma que en los puntos anteriores (Security0), el resultado se muestra en la imagen 10.

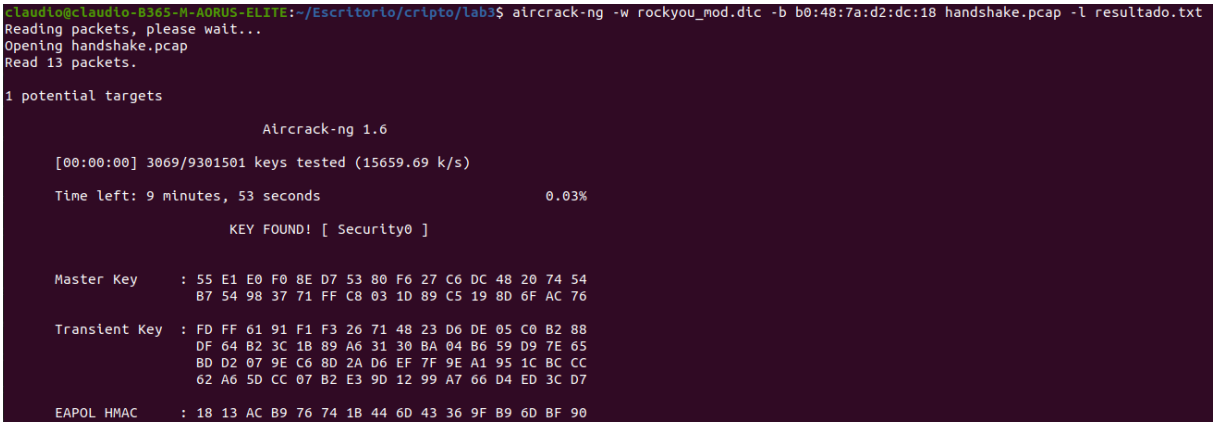

Figura 10: Obtención de contraseña a través de aircrack.

4.6. identifica y modifica parámetros solicitados por pycrack

Para este punto se descargó el código de Python Pycrack, además se modificaron los campos ssid ("VTR-1645213"), aNonce (WPA Key Nonce del AP), sNonce (WPA Key Nonce del cliente), apMac (MAC del AP), cliMac (MAC del cliente), mic1 (WPA Key MIC del segundo paquete y para los siguientes mic se repite esta instrucción, pero con los datos de los próximos dos paquetes), data1 (Campo completo del 802.1X Authentication, pero con los bytes del mic en seteados en cero, cabe destacar que este paso se repite para los dos paquetes siguientes), mic2, data2, mic3, data3. Estos datos se obtienen por la captura del handshake descargada. Los parámetros se muestran en la figura 11 y además en la figura 12 se evidencia los parametros del paquete 2, el cual se utiliza en el mic1 y data1.

```
178
179 if __name__ == "__main__":
180
181     # RunTest()
182     #Read a file of passwords containing
183     #passwords separated by a newline
184     with open('rockyou_mod.dic') as f:
185         S = []
186         for l in f:
187             S.append(l.strip())
188
189     #ssid name
190     ssid = "VTR-1645213"
191     #ANonce
192     aNonce = a2b_hex('4c2fb7eca28fba45accefde3ac5e433314270e04355b6d95086031b004a31935')
193     #SNonce
194     sNonce = a2b_hex("30bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c038f5869b96bbe9cdc")
195     #Authenticator MAC (AP)
196     apMac = a2b_hex("b0487ad2dc18")
197     #Station address: MAC of client
198     cliMac = a2b_hex("eede678cdf8b")
199     #The first MIC
200     mic1 = "1813acb976741b446d43369fb96dbf90"
201
202     #The entire 802.1x frame of the second handshake message with the MIC field set to all zeros
203     data1 = a2b_hex("0103007502010a00000000000000000000130bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c038f5869b96bb
204     #The second MIC
205     mic2 = "a349d01089960aa9f94b5857b0ea10c6"
206     #The entire 802.1x frame of the third handshake message with the MIC field set to all zeros
207     data2 = a2b_hex("020300970213ca001000000000000000024c2fb7eca28fba45accefde3ac5e433314270e04355b6d95086031b004a
208     #The third MIC
209     mic3 = "5cf0d63af458f13a83daa686df1f4067"
210     #The entire 802.1x frame of the forth handshake message with the MIC field set to all zeros
211     data3 = a2b_hex("0103005f02030a0000000000000000000200000000000000000000000000000000000000000000000000000000000000
212     #Run an offline dictionary attack against the access point
213     TestPwds(S, ssid, aNonce, sNonce, apMac, cliMac, data1, data2, data3, mic1, mic2, mic3)
```

Figura 11: Código modificado en Pycrack.

```
    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 (… 802.11      10 Acknowledgement, Flags=........
    9 0.017087                           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 (… 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 (… 802.11      10 Acknowledgement, Flags=........

Frame 7: 155 bytes on wire (1240 bits), 155 bytes captured (1240 bits)
IEEE 802.11 QoS Data, Flags: .......T
Logical-Link Control
802.1X Authentication
  Version: 802.1X-2001 (1)
  Type: Key (3)
  Length: 117
  Key Descriptor Type: EAPOL RSN Key (2)
  [Message number: 2]
▸ Key Information: 0x010a
  Key Length: 0
  Replay Counter: 1
  WPA Key Nonce: 30bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c03…
  Key IV: 00000000000000000000000000000000
  WPA Key RSC: 0000000000000000
  WPA Key ID: 0000000000000000
  WPA Key MIC: 1813acb976741b446d43369fb96dbf90
  WPA Key Data Length: 22
▸ WPA Key Data: 30140100000fac040100000fac040100000fac020000

0000  88 01 3a 01
0010  b0 48 7a d2
0020  88 8e 01 03
0030  00 00 01 30
0040  78 8e 95 b6
0050  be 9c dc 00
0060  00 00 00 00
0070  00 00 00 18
0080  6d bf 90 00
0090  0f ac 04 01
```

Figura 12: Captura de Handshake.

4.7. obtiene contraseña con pycrack.

Una vez identificamos y modificamos los campos se procedió a ejecutar el código, dando el resultado mostrado en la figura 13.

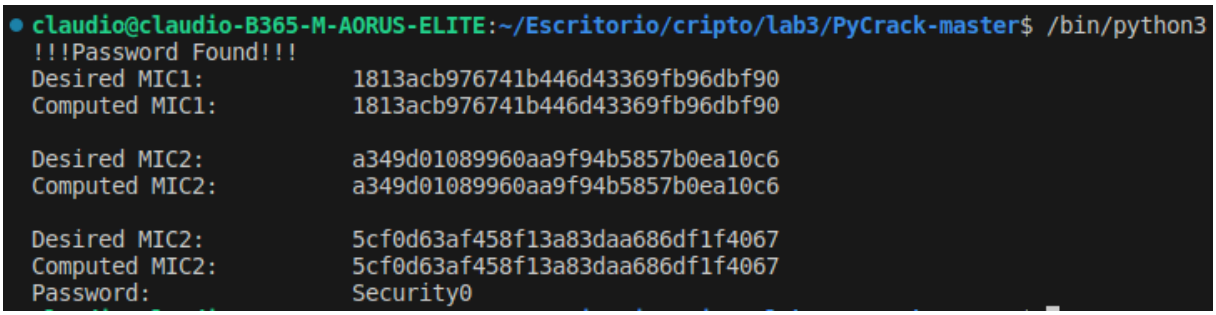

Figura 13: Resultado de Pycrack.

Conclusiones y comentarios

Para el ámbito de romper contraseñas se puede evidenciar que existen varias herramientas, en el caso de este laboratorio se logró trabajar con tres de estas herramientas (aircrack-ng, hashcat y pycrack).

Cabe destacar que unos de los problemas presentados en este trabajo fue en al emplear por segunda vez hashcat, esto es debido a que para volver a utilizar esta herramienta con los mismos parámetros había que borrar los logs y los archivos que este generaba.