Informe Laboratorio 3

Sección 1

Sofía Ignacia Belmar Alvarez
e-mail: sofia.belmar@mail.udp.cl

Mayo de 2024

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 Rockyou (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.

   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.

3. A partir del archivo que descargó de Internet, obtenga la password asociada a la generación de dicho archivo. Obtenga la llave mediante un ataque por fuerza bruta.
   Para esto deberá utilizar tres herramientas distintas para lograr obtener la password del archivo: hashcat, aircrack-ng, pycrack. Esta última, permite entender paso a paso de qué forma se calcula la contraseña a partir de los valores contenidos en el handshake, por lo que deberá agregar dichos valores al código para obtener la password a partir de ellos y de rockyou_mod.dic. Antes de ejecutar esta herramienta deberá deshabilitar la función RunTest().
   Al calcular la password con hashcat utilice dos técnicas: una donde el resultado se guarda en el potfile y otra donde se deshabilita el potfile. Indique qué información retorna cada una de las 2 técnicas, identificando claramente cada campo.
   Recuerde indicar los 4 mayores problemas que se le presentaron y cómo los solucionó.

2. Desarrollo (PASO 1)

Para comenzar con el desarrollo de esta sección, se debe inicializar el monitor utilizando el comando 'sudo airmon-ng'. Este mostrará la lista de las interfaces de red disponibles y su estado. Al hacer resto, se hace uso de la interfaz específica que se quiera utilizar. En este

caso, la interfaz correspondió a 'wlp4s0f4u2mon'. Continuando con esto, se inicia el modo monitor en esta interfaz específica, permitiendo así, obtener todo el tráfico inalámbrico en un canal específico. El comando utilizado para esto se puede ver en la Figura 1.

```
1  sudo airmon-ng start wlp4s0f4u2mon
2
```

Figura 1: Comando para iniciar el modo monitor en una interfaz específica.

2.1. En qué se destaca la red del informante del resto

A partir del comando mencionado anteriormente, se obtuvo lo siguiente:

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3$ sudo airodump-ng wlp4s0f4u2mon


CH  7 ][ Elapsed: 0 s ][ 2024-05-17 09:44

BSSID              PWR  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSID

B0:1F:8C:E1:B2:04  -83        2        0    0   1  130   WPA3 CCMP   OWE  <length:  0>
B0:1F:8C:E2:14:A1  -76        1        0    0  11  130   OPN              Invitados-UDP
18:35:D1:90:C7:99  -91        2        0    0  11  130   WPA2 CCMP   PSK  VTR-6733269
B0:1F:8C:E2:14:A3  -77        2        0    0  11  130   OPN              Alumnos-UDP
B0:1F:8C:E2:14:A0  -77        2        0    0  11  130   WPA3 CCMP   SAE  Sala Hibrida-UDP
98:FC:11:86:B6:B9  -75        3       12    5  11  130   WPA2 CCMP   PSK  Telematica
82:C3:31:4C:91:31  -66        4        2    0  11  180   WPA2 CCMP   PSK  Alexis
00:25:00:FF:94:73   -1        0        0    0  -1   -1                    <length:  0>
B0:1F:8C:E1:B2:03  -82        1        0    0   1  130   OPN              Alumnos-UDP
82:45:6B:0D:79:DA  -87        5        0    0   6  130   WPA2 CCMP   PSK  <length: 15>
58:EF:68:47:59:C8  -83       15        0    0   6  130   OPN              cableadaTelematica-invitado
14:CC:20:E8:EB:35  -84        2        0    0   8  270   WPA2 CCMP   PSK  Jpablov_EXT
58:EF:68:47:59:C6  -82       12        0    0   6  130   WPA2 CCMP   PSK  cableadaTelematica
B0:1F:8C:E1:B2:05  -83        3        0    0   1  130   OPN              VIP-UDP
FA:8F:CA:50:A8:EF  -92        2        0    0   1   65   OPN              Dormitorio grande
B0:1F:8C:E0:E8:84  -88        1        1    0   1  130   WPA3 CCMP   OWE  <length:  0>
B0:1F:8C:E1:B2:02  -81        2        0    0   1  130   WPA3 CCMP   OWE  <length:  0>
24:FB:65:8A:A0:F1  -51        3        0    0   1   65   WPA2 CCMP   PSK  HUAWEI P20 lite
E4:AB:89:07:57:38  -88        3        0    0   1  130   WPA2 CCMP   PSK  Sofia522 2,4G
E6:AB:89:1C:85:38   -1        0        0    0   6   -1                    <length:  0>
B0:48:7A:D2:DD:74  -59       15      552  130   6   54e  WEP  WEP          WEP
```

Figura 2: Línea de comandos tras ejecutar el comando.

Como se puede observar al final de la Figura 2, se muestra una red tipo WEP, la cual se destaca dentro de las otras redes ya que es la única red con el protocolo WEP. Además, se destaca dentro de las otras por utilizar un protocolo obsoleto dado a su falta de seguridad y vulnerabilidad frente a ataques.

Una vez que ya se tiene la red, se escanea el canal específico donde se encuentre la red. Esto con el objetivo de concentrar la captura en una frecuencia específica.

Como el canal en el que se encuentra la red WEP corresponde al 6, el comando final se puede observar en la Figura 3:

```
1  sudo airodump-ng -c 6 wlp4s0f4u2mon
```

Figura 3: Comando para escanear un canal específico.

Luego, se procede con la captura de tráfico en este canal. Dentro de esta captura, se debe tener en consideración

```
1  sudo airodump-ng -c 6 -w capture wlp4s0f4u2mon
```

Figura 4: Comando para capturar tráfico.

En la Figura 4 se enseña el comando utilizado para realizar esta captura de tráfico.

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

Se debe tener en cuenta que al momento de realizar la captura, esta debe tener entre de 5.000 a 10.000 paquetes para obtener la clave WEP. Para entender esto, de debe tener en cuenta el concepto de la "paradoja del cumpleaños", la cual está relacionada con conceptos de probabilidad y estadística, y explica porqué se requiere una cierta cantidad de paquetes para poder obtener la clave. La fórmula de este concepto se puede ver a continuación:

$$P(\text{colisión}) = 1 - e^{-\frac{n^2}{2N}} \tag{1}$$

En donde P corresponde a la probabilidad de que ocurra al menos una colisión, n el número de elementos capturados o vectores de inicialización y N el total de posibles valores.

En las redes WEP, los vectores de inicialización (IV) se utilizan junto con una clave para cifrar los datos transmitidos. Debido al espacio de IV (24 bits), hay 2^{24} posibles valores de IV, es decir, aproximadamente 16.7 millones de posibles IVs. Por lo tanto, para obtener una probabilidad alta de éxito, es necesario acumular una alta cantidad de paquetes. Esta cantidad debe ser superior a 5.000 paquetes.

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

Una vez que se tenga la captura, se procede con el ataque con aircrack-ng. Para esto es importante tener el BSSID de la red que se quiere atacar, la cual en este caso corresponde a 'B0:48:7A:D2:DD:74'. Este se puede verificar en la Figura 2.

Teniendo esto en cuenta, a continuación se muestra el comando para realizar el ataque a esta red. Para esto, se utiliza el comando de la Figura 5.

```
1  sudo aircrack-ng -b B0:48:7A:D2:DD:74 capture-04.cap
```

Figura 5: Comando para ataque con aircrack-ng a red WEP.

Al aplicarlo, se obtuvo lo siguiente.

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3/capture$ sudo aircrack-ng -b B0:48:7A:D2:DD:74 capture-04.cap
[sudo] password for sofiabelmar:
Reading packets, please wait...
Opening capture-04.cap
Read 655160 packets.
                                                    Got 227503 out of 225000 IVsStarting PTW attack with 227503 ivs.
                      KEY FOUND! [ 12:34:56:78:90 ]
       Decrypted correctly: 100%
Attack will be restarted every 5000 captured ivs.
```

Figura 6: Ataque con aircrack-ng a red WEP.

En la Figura 6 se puede observar la terminal, en donde muestra el valor de la contraseña tras el ataque.

2.4. Indica el tiempo que demoró en obtener la password

Para obtener el tiempo que tardó aircrack-ng en obtener, se ejecuta el comando de la Figura 7.

```
1  time sudo aircrack-ng -b B0:48:7A:D2:DD:74 capture-04.cap
```

Figura 7: Comando para medir el tiempo del ataque.

Este comando proporciona el tiempo real que toma la ejecución del ataque, es decir, de la ejecución del comando de la Figura 5. Seguidamente se presenta la línea de comandos tras utilizar este comando.

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3/capture$ time sudo aircrack-ng -b B0:48:7A:D2:DD:74 capture-04.cap
Reading packets, please wait...
Opening capture-04.cap
Read 655160 packets.

1 potential targets                                    Got 227503 out of 225000 IVsStarting PTW attack with 227503 ivs.
                        KEY FOUND! [ 12:34:56:78:90 ]
Attack wDecrypted correctly: 100%00 captured ivs.



real    0m0.807s
user    0m0.007s
sys     0m0.011s
```

Figura 8: Tiempo demora del ataque.

Como se puede notar en la Figura 8 el proceso toma 0.807s para obtener la contraseña a través del ataque por aircrack-ng.

2.5. Descifra el contenido capturado

Para descifrar el contenido de la captura, se ejecuta el comando de la Figura 9.

```
1  sudo airdecap-ng -w 12:34:56:78:90 capture-04.cap
```

Figura 9: Comando para descifrar el contenido.

Dentro de este contenido se debe colocar la contraseña obtenida anteriormente para desencriptar el tráfico capturado. Esto permite visualizar el contenido del tráfico que fue cifrado originalmente. Al ejecutar este comando, se crea un archivo DEC con el tráfico desencriptado.

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

Para visualizar el archivo DEC previamente creado, se utiliza el software Wireshark. Dentro de este se puede observar lo siguiente:

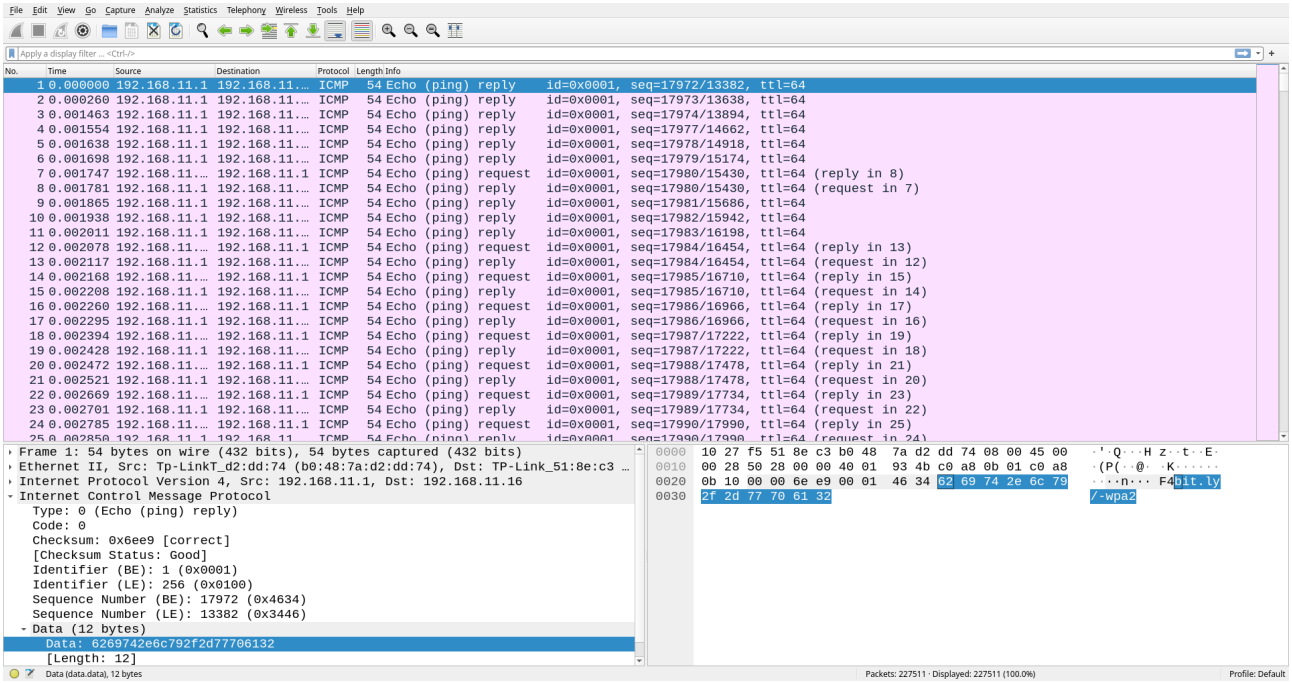

Figura 10: Captura desencriptada.

En la Figura 10 se pueden notar varios paquetes ICMP. Para encontrar la URL, se debe seleccionar cualquiera de estos paquetes. Al hacerlo, dentro del payload se puede observar una URL, que corresponde a `bit.ly/wpa2_`. Al acceder a este enlace, se puede ver otra captura, la cual se presenta en la Figura 11.

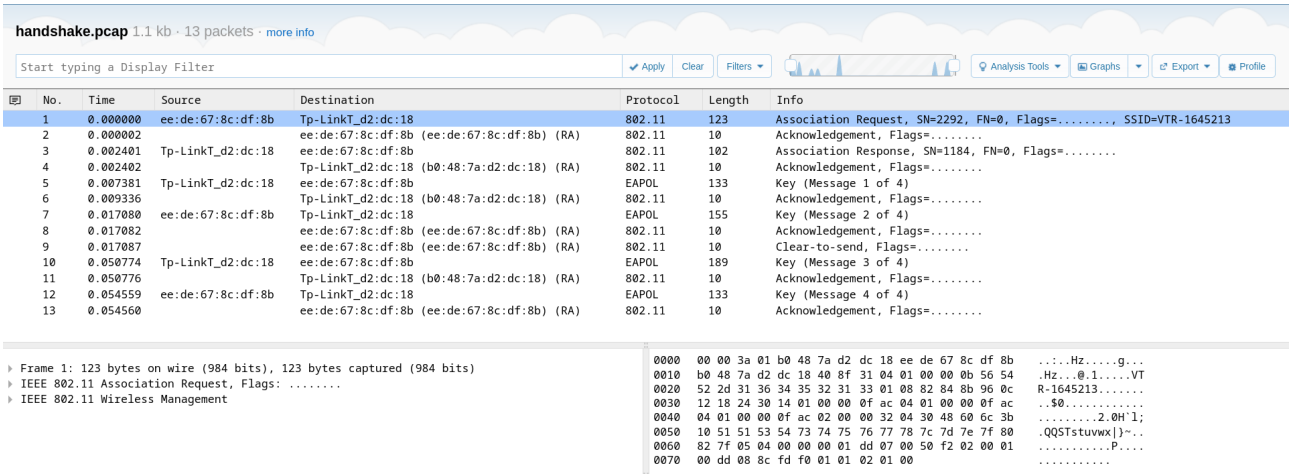

Figura 11: URL obtenida a través de la desencriptación.

3. Desarrollo (PASO 2)

3.1. Script para modificar el diccionario original

Para esta parte, se creó un archivo llamado 'rules.rule'. Dentro de este archivo, hay una línea con 'c $0'. Estas son reglas de hashcat donde 'c' capitaliza la primera letra y '$0' agrega un 0 al final de cada contraseña. Una vez configurado esto, se utiliza el comando que se muestra en la Figura 12.

```
1  hashcat --stdout -r rules.rule rockyou.txt | sed '/^[0-9]/d' > rockyou_mod.dic
```

Figura 12: Script para modificar el diccionario original.

Este comando aplica las reglas de hashcat definidas en el archivo 'rules.rule' a cada entrada del diccionario 'rockyou.txt'. Luego, se utiliza 'sed' para eliminar todas las líneas que comienzan con un dígito. Finalmente, el resultado se redirige a un nuevo archivo llamado 'rockyou_mod.dic', que contendrá las contraseñas modificadas y filtradas.

3.2. Cantidad de passwords finales que contiene rockyou_mod.dic

Para visualizar la cantidad de contraseñas finales en el nuevo archivo, se ejecuta el comando de la Figura 13.

```
1  wc -l rockyou_mod.dic
```

Figura 13: Comando para visualizar cantidad de contraseñas en rockyou_mod.dic.

Al ejecutar este comando, se muestra la cantidad de contraseñas en el archivo rockyou_mod.dic, que corresponde a 11.059.790 contraseñas en total. Esta salida se puede observar en la Figura 14.

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3/files$ wc -l rockyou_mod.dic
11059790 rockyou_mod.dic
```

Figura 14: Cantidad de contraseñas finales en rockyou_mod.dic.

4. Desarrollo (Paso 3)

4.1. Obtención de contraseña con hashcat utilizando potfile

Para comenzar, es necesario cambiar el tipo de archivo obtenido a través de la URL a uno compatible con hashcat, ya que este último no soporta archivos '.cap' ni '.pcapng'. Teniendo esto en cuenta, se procede a utilizar el comando de la Figura 15, el cual toma el archivo '.cap' y lo convierte en un archivo '22000', compatible con hashcat.

```
1  hcxpcapngtool -o handshake.22000 handshake.pcap
```

Figura 15: Comando para pasar de '.cap' a '.22000'.

Luego, se procede con el comando para el hashcat con potfile. El comando utilizado se puede observar en la Figura 16.

```
1  hashcat -D 1 -m 22000 handshake.22000 rockyou_mod.dic --potfile-path potfile.txt
```

Figura 16: Comando para obtener contraseña con hashcat con potfile.

4.2. Nomenclatura del output

Al ejecutar el comando de la Figura 16, se obtiene el siguiente output:

```
Host memory required for this attack: 3 MB


Dictionary cache built:

* Filename..: rockyou_mod.dic
* Passwords.: 11059790
* Bytes.....: 119974994
* Keyspace..: 11059790
* Runtime...: 1 sec

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


Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 22000 (WPA-PBKDF2-PMKID+EAPOL)
Hash.Target......: handshake.22000
Time.Started.....: Mon May 20 22:34:50 2024, (0 secs)
Time.Estimated...: Mon May 20 22:34:50 2024, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (rockyou_mod.dic)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:    14464 H/s (5.87ms) @ Accel:128 Loops:256 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 2907/11059790 (0.03%)
Rejected.........: 1371/2907 (47.16%)
Restore.Point....: 0/11059790 (0.00%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#2....: Password0 -> Dangerous0
Hardware.Mon.#2..: Temp: 97c Util: 47%
```

Figura 17: Contraseña obtenida con hashcat con potfile.

Considerando el output obtenido, se definen las siguientes nomenclaturas:

- **Host memory required for this attack**: Cantidad de memoria del host necesaria para el ataque.
- **Dictionary cache built**: Información sobre el diccionario.
- **Session**: Nombre de la sesión de hashcat.
- **Status**: Estado actual del ataque.
- **Hash.Mode**: Modo de hash utilizado.
- **Hash.Target**: Nombre del archivo objetivo.
- **Time.Started**: Fecha y hora de inicio del ataque.
- **Time.Estimated**: Fecha y hora del término del ataque.
- **Kernel.Feature**: Características del kernel.

- **Guess.Base**: Método base para adivinar contraseñas
- **Guess.Queue**: Progreso de la cola.
- **Speed**: Velocidad del ataque.

4.3. Obtiene contraseña con hashcat sin potfile

Para obtener la contraseña con hashcat sin potfile se debe utilizar el mismo archivo '.22000' creado en el ítem anterior. El comando utilizado en este ítem es similar al anterior, pero se modifica al final, quedando de la siguiente manera:

```
1  hashcat -D 1 -m 22000 handshake.22000 rockyou_mod.dic --potfile-disable
```

Figura 18: Comando para obtener contraseña con hashcat sin potfile.

4.4. Nomenclatura del output

Al ejecutar el comando de la Figura 18, se visualiza lo siguiente:

```
Host memory required for this attack: 3 MB

Dictionary cache hit:
* Filename..: rockyou_mod.dic
* Passwords.: 11059790
* Bytes.....: 119974994
* Keyspace..: 11059790

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

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 22000 (WPA-PBKDF2-PMKID+EAPOL)
Hash.Target......: handshake.22000
Time.Started.....: Tue May 21 00:52:06 2024 (0 secs)
Time.Estimated...: Tue May 21 00:52:06 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (rockyou_mod.dic)
Guess.Queue......: 1/1 (100.00%)
Speed.#2.........:    10049 H/s (8.63ms) @ Accel:128 Loops:256 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 2907/11059790 (0.03%)
Rejected.........: 1371/2907 (47.16%)
Restore.Point....: 0/11059790 (0.00%)
Restore.Sub.#2...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#2....: Password0 -> Dangerous0
Hardware.Mon.#2..: Temp: 63c Util: 49%

Started: Tue May 21 00:52:04 2024
Stopped: Tue May 21 00:52:08 2024
```

Figura 19: Contraseña obtenida con hashcat sin potfile.

Considerando el output obtenido de la Figura 19, se definen las siguientes nomenclaturas:

- **Host memory required for this attack**: Cantidad de memoria del host necesaria para el ataque.
- **Dictionary cache hit**: Información sobre el diccionario.
- **Hash**: Hash que se está atacando.
- **Session**: Nombre de la sesión de hashcat.
- **Status**: Estado actual del ataque.
- **Hash.Mode**: Modo de hash utilizado.
- **Hash.Target**: Nombre del archivo objetivo de hash.
- **Time.Started**: Fecha y hora de inicio del ataque.
- **Time.Estimated**: Fecha y hora estimada de finalización del ataque.

- **Kernel.Feature**: Características del kernel utilizado.
- **Guess.Base**: Método base utilizado para adivinar contraseñas.
- **Speed**: Velocidad del ataque en hashes por segundo.

Dentro de la nomenclatura no se encuentran muchas diferencias con la nomenclatura de hashcat con potfile. Sin embargo, lo que diferencia notoriamente una de otra, es la creación del archivo 'potfile.txt' con la contraseña obtenida.

4.5. Obtiene contraseña con aircrack-ng

Para la obtención de la contraseña con aircrack-ng se debe utilizar el comando de la Figura 20. Este comando inicia con la red y carga las contraseñas del archivo 'rockyou_mod.dic'. Una vez que estas con cargadas, se realiza el ataque por diccionario, utilizando cada contraseña para descifrar la clave del handshake.

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

Figura 20: Comando para obtener contraseña con aircrack-ng.

Al ejecutar este comando se observa lo siguiente:

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3/files$ aircrack-ng -a2 -w rockyou_mod.d
ic handshake.pcap
Reading packets, please wait...
Opening handshake.pcap
Read 13 packets.

   #  BSSID              ESSID                     Encryption

   1  B0:48:7A:D2:DC:18  VTR-1645213               WPA (1 handshake)

Choosing first network as target.

Reading packets, please wait...
Opening handshake.pcap
Read 13 packets.

1 potential targets
```

Figura 21: Output aircrack-ng.

Figura 22: Contraseña obtenida con aircrack-ng.

En la Figura 21 y 22 se puede observar el output obtenido, en donde se muestran las llaves probadas, la velocidad, el tiempo que tomó y finalmente la contraseña.

4.6. Identifica y modifica parámetros solicitados por pycrack

Primeramente, es necesario instalar PyCrack ejecutando el siguiente comando:

```
1  git clone https://github.com/nogilnick/PyCrack
```

Figura 23: Comando para instalar PyCrack.

Para este ataque es necesario modificar un script en Python, dentro del cual es necesario modificar los parámetros que solicita PyCrack. Dichos parámetros corresponden a SSID, aNonce, aNonce, apMac, cliMac, Mic y Data. Para obtener los valores de estos parámetros, es necesario analizar la captura y obtener los parámetros de esta. A continuación, se enseñarán los valores de los parámetros obtenidos de la captura.

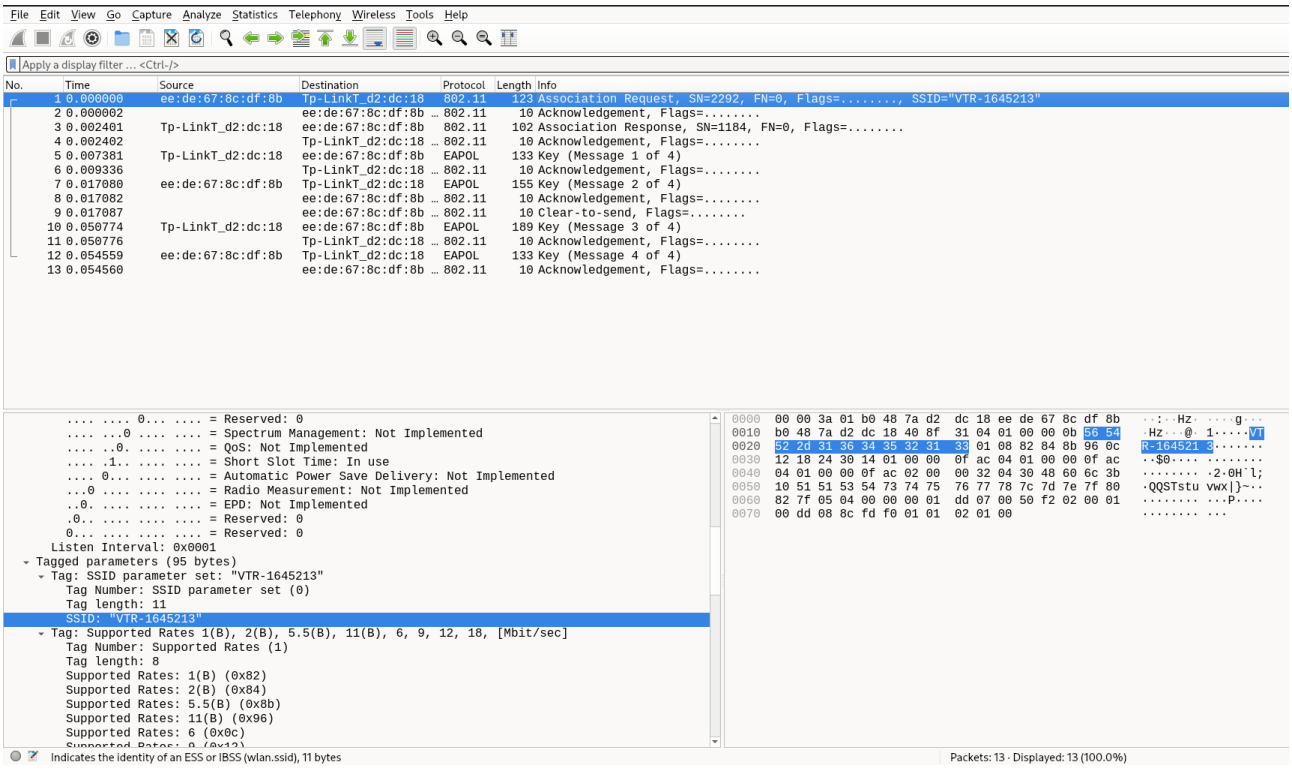

Figura 24: Parámetro SSID.

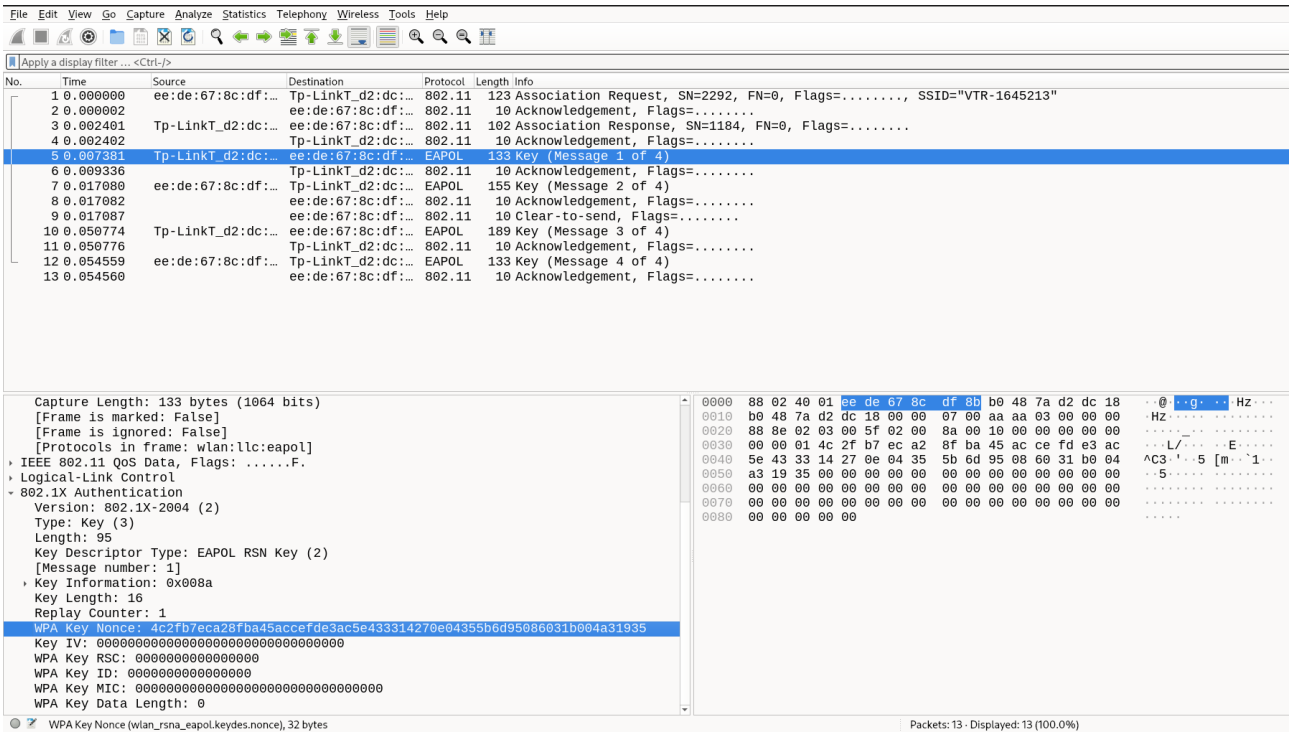

Figura 25: Parámetro aNonce.

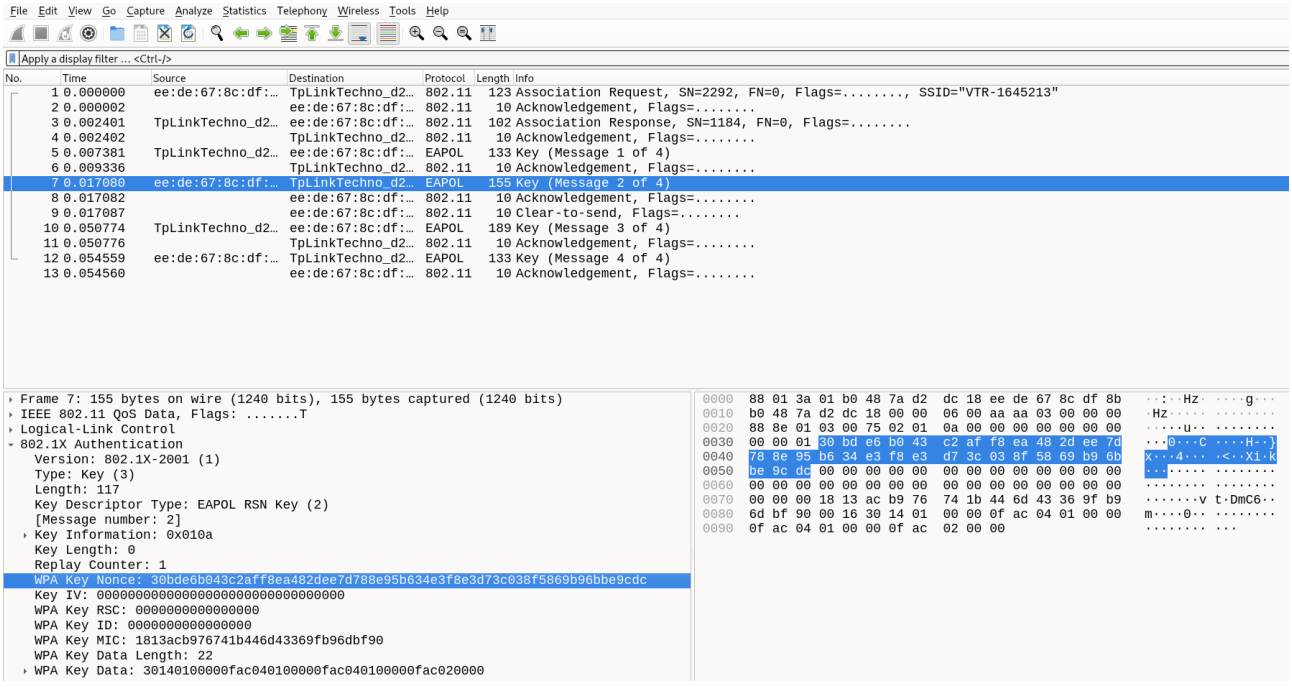

Figura 26: Parámetro sNonce.

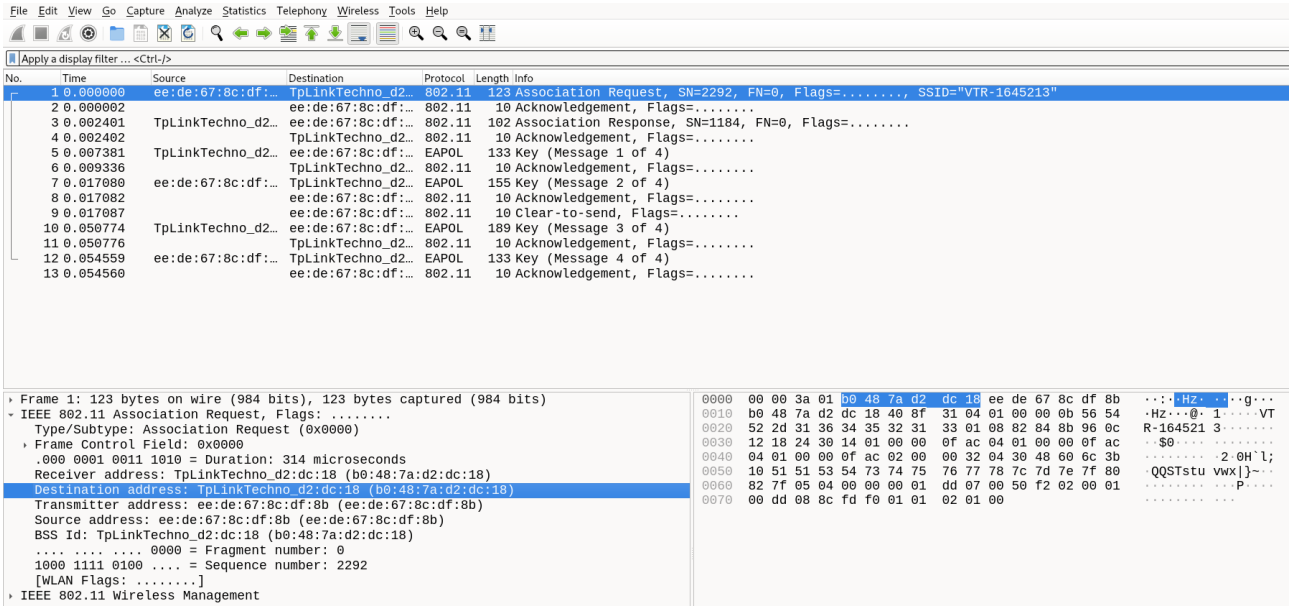

Figura 27: Parámetros apMac y cliMac.

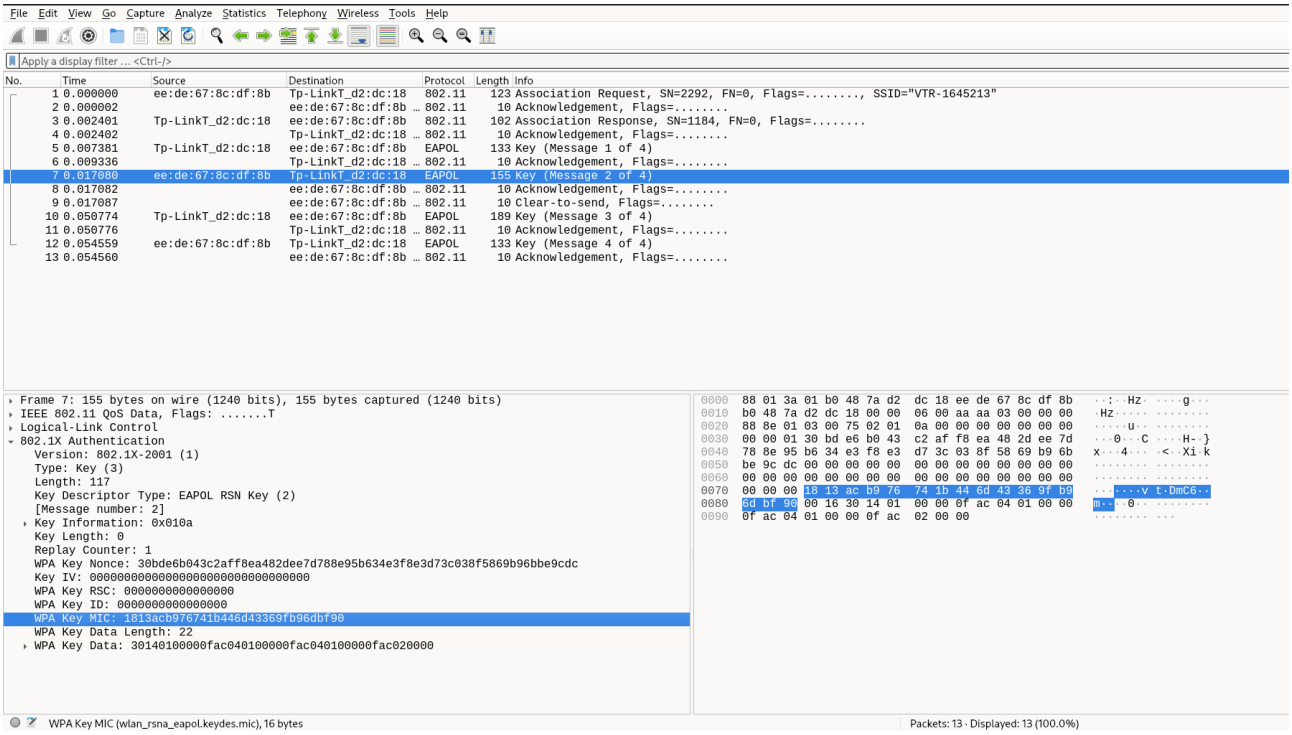

Figura 28: Parámetro primer Mic.

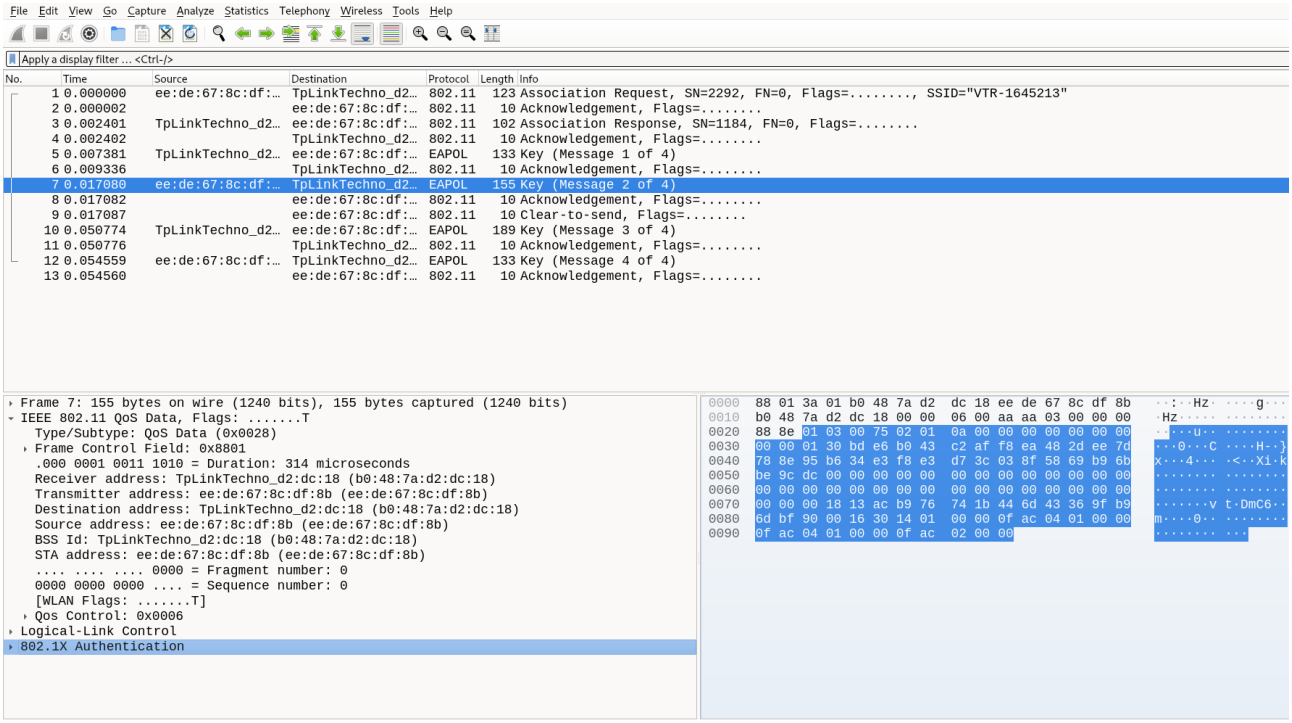

Figura 29: Data 1.

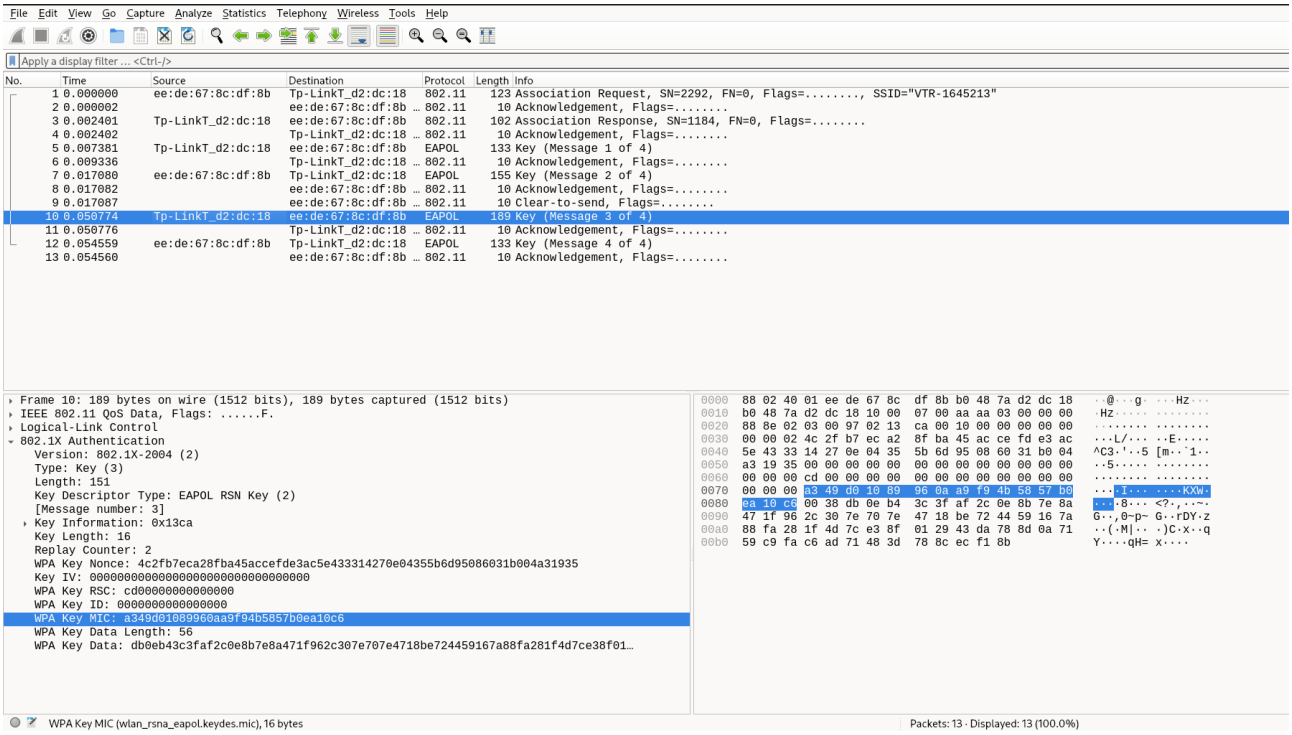

Figura 30: Parámetro segundo Mic.

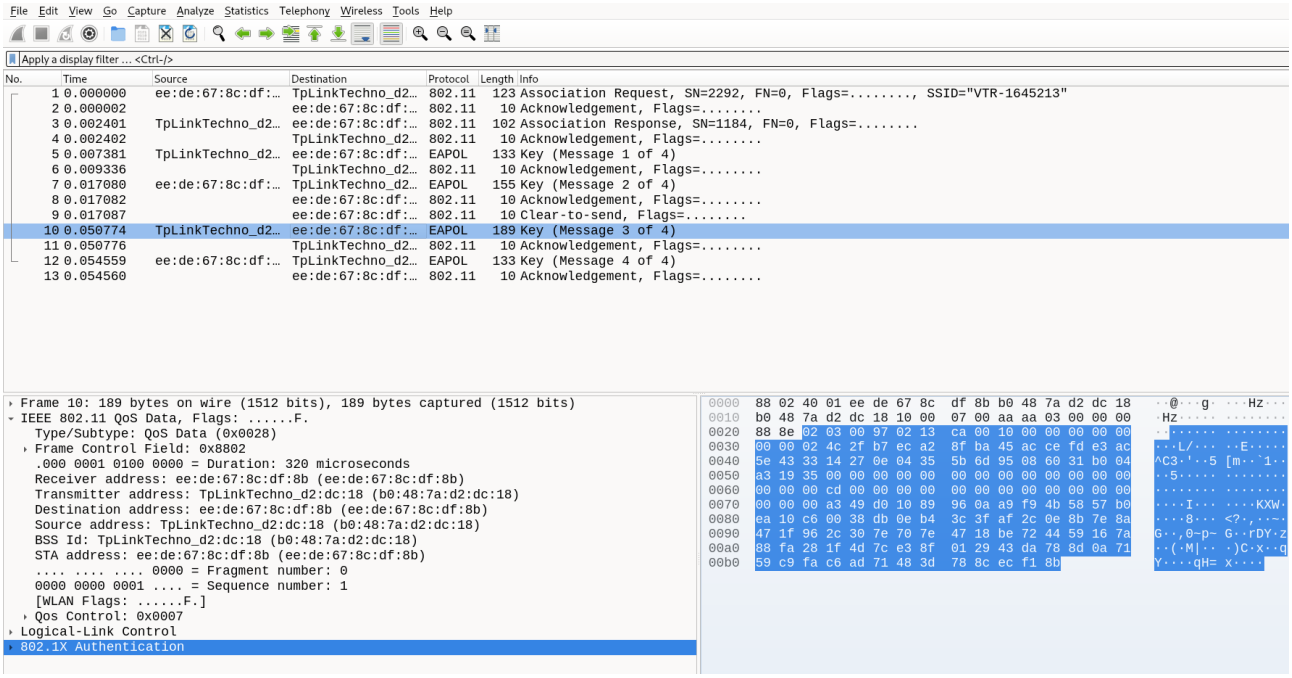

Figura 31: Data 2.

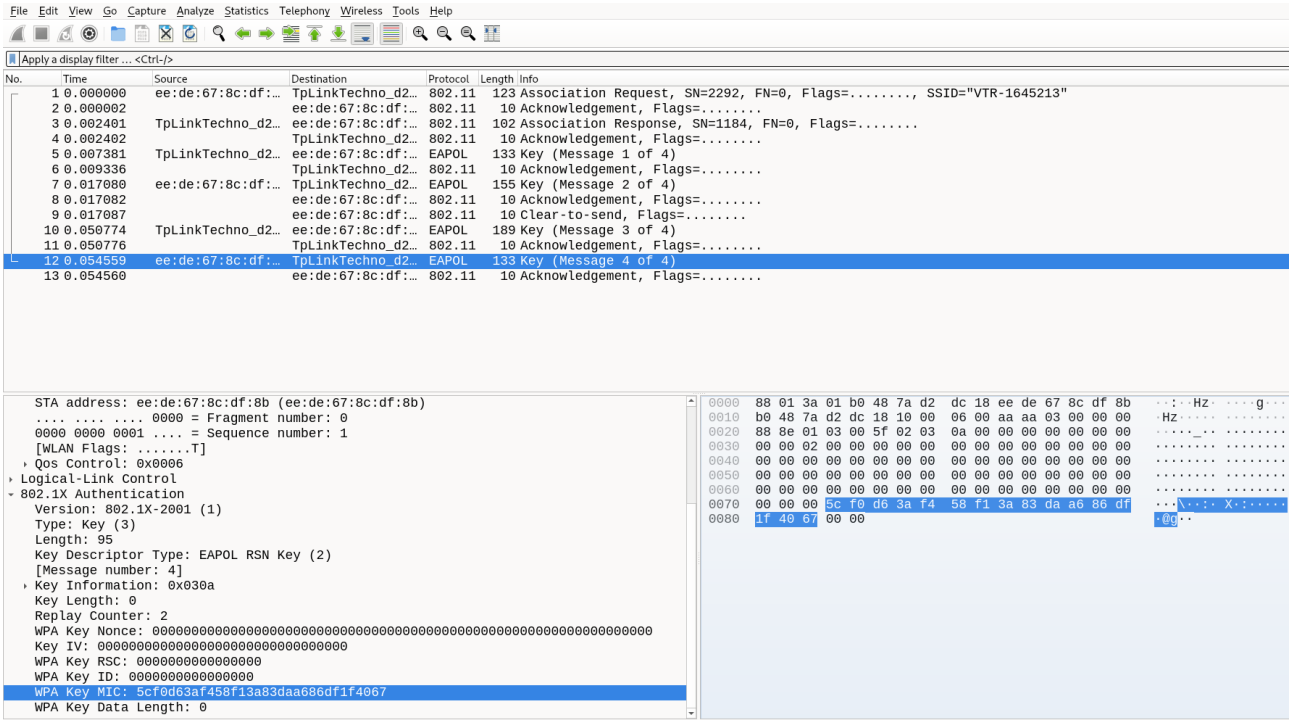

Figura 32: Parámetro tercer Mic.

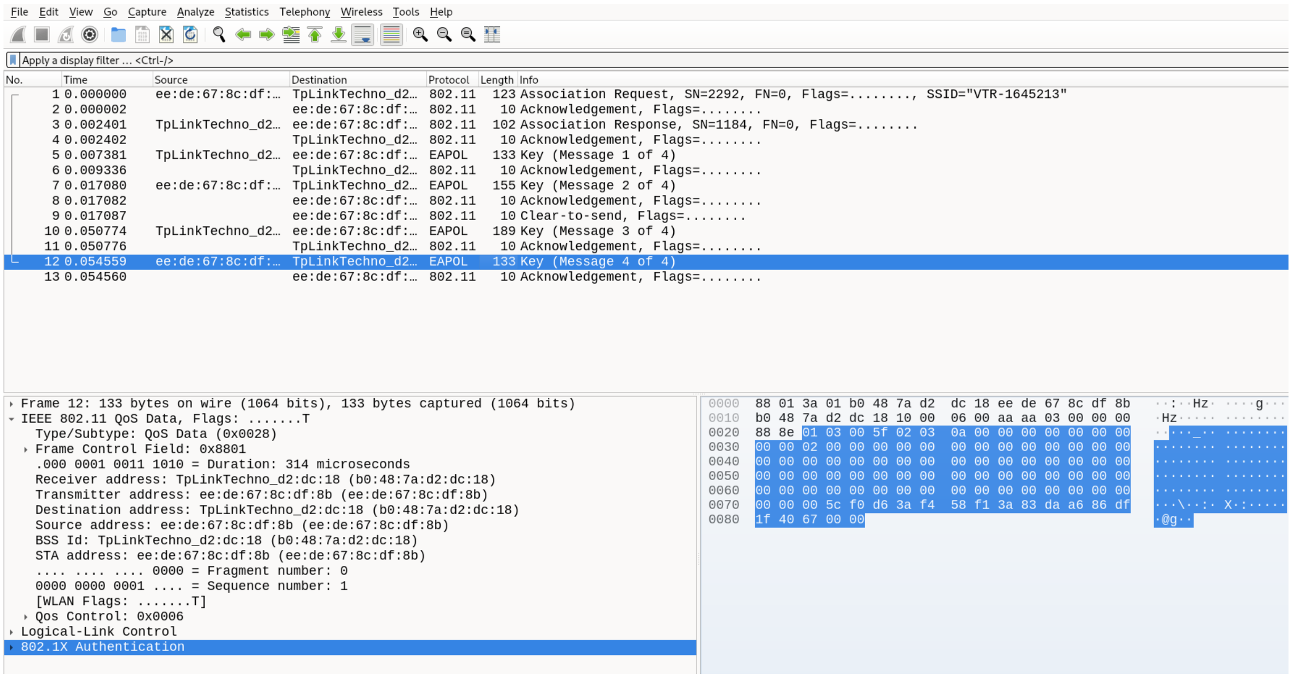

Figura 33: Data 3.

Una vez que se tengan estos valores, estos se reemplazan en los campos respectivos quedando de la siguiente manera:

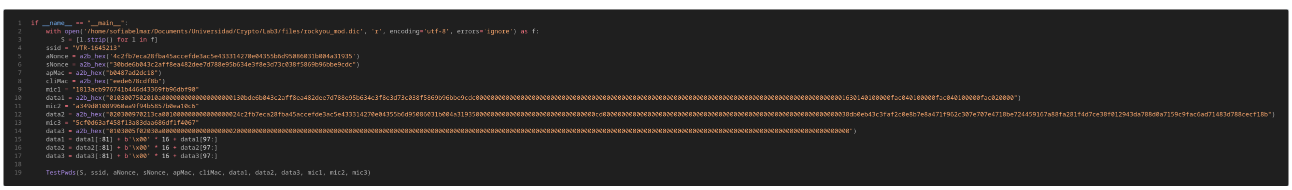

Figura 34: Script con los parámetros modificados.

4.7. Obtiene contraseña con pycrack

Para obtener la contraseña con PyCrack, se debe ejecutar el script anterior.

```
sofiabelmar@fedora:~/Documents/Universidad/Crypto/Lab3/pycrack$ python3 pywd.py
!!!Password Found!!!
Desired MIC1:           1813acb976741b446d43369fb96dbf90
Computed MIC1:          1813acb976741b446d43369fb96dbf90

Desired MIC2:           a349d01089960aa9f94b5857b0ea10c6
Computed MIC2:          a349d01089960aa9f94b5857b0ea10c6

Desired MIC3:           5cf0d63af458f13a83daa686df1f4067
Computed MIC3:          5cf0d63af458f13a83daa686df1f4067
Password:               Security0
```

Figura 35: Contraseña obtenida con PyCrack.

Como se puede observar en la Figura 35, la contraseña se logra obtener de manera exitosa con el uso de PyCrack.

Conclusiones y comentarios

En el desarrollo del laboratorio, se exploraron técnicas de seguridad informática relacionadas la vulnerabilidad en redes inalámbricas y la manipulación de contraseñas. Se tuvo un enfoque en la vulnerabilidad que tiene el cifrado WEP y se demostró esto mismo a través de herramientas como Aircrack-ng, Hashcat y PyCrack. Estas herramientas permitieron obtener una visión clara de la falta de robustez del cifrado WEP, lo que también permitió la comprensión del porqué este cifrado ya no se considera seguro en la actualidad.

Además, se adquirieron conocimientos relacionados con la modificación de diccionarios y sobre herramientas fundamentales para identificar vulnerabilidades, donde a raíz de esto, se logró comprender la fragilidad de sistemas de seguridad obsoletos. Este conocimiento es crucial para la implementación de medidas de seguridad más robustas y efectivas en entornos de red.

Issues

Durante el desarrollo del laboratorio, se tuvieron los siguientes problemas:

- **OpenCL**: Durante la ejecución de los comandos de hashcat en la parte 3 del laboratorio, la terminal arrojaba un error relacionado con el driver OpenCL, lo cual no impedía que hashcat funcionada de manera correcta. Para solucionar esto, fue necesario incluir '-D' para ejecutarlo con CPU, y además incluir el '–force'. Incluyendo estos cambios, fue posible lograr lo requerido.

- **Formato Hash**: Dado a la falta de conocimiento, en un inicio se estaba intentando utilizar hashcat con la captura en su formato 'pcapng'. Esto arrojaba un error ya que hashcat no soporta estos formatos. Teniendo esto en cuenta, la captura se pasó al formato 2500, pero este formato ya estaba obsoleto, por lo que hashcat seguía sin poder ejecutarse. Finalmente, al hacer una búsqueda, se dió con el formato permitido

por hashcat, el cual correspondía al '.22000'. Al pasarlo a este formato, hashcat logró ejecutarse de manera exitosa.

- **Aircrack-ng**: Dado a los desconocimientos sobre aircrack-ng, fue completo encontrar la manera de encontrar la contraseña. Luego de varios intentos fallidos, con apoyo del profesor y de la documentación de aircrack-ng, se pudo obtener la contraseña.
- **Parámetros de PyCrack**: Al momento de buscar los parámetros en la captura para posteriormente modificar el archivo de PyCrack, se hizo complejo dado a la cantidad de datos existentes, y nuevamente dado al desconocimiento que se tenía sobre este. Sin embargo, luego de hacer una búsqueda intensiva sobre los parámetros, se logra obtener todos los valores de los parámetros.