Informe Laboratorio 3

Sección 1

Alumno : Alejandro Lagos
e-mail : alejandro.lagos@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 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. En qué se destaca la red del informante del resto

Luego de configurar la tarjeta de red externa de nombre **wlx1027f5518cfe** en modo monitor, y usando una de las herramientas de **Aircrack-ng** se hace un escaneo de las redes inalámbricas que están alrededor con el siguiente comando:

```
metroh@METROH-NOTEBOOK-LINUX:~$
sudo airodump-ng wlx1027f5518cfe
```

Figura 1: Comando para escanear las redes inalámbricas cercanas

Al hacer dicho escaneo en la red, notamos que hay una red que destaca sobre el resto debido a que esta configurada con el obsoleto tipo de encriptación **WEP** (Wired Equivalent Privacy).

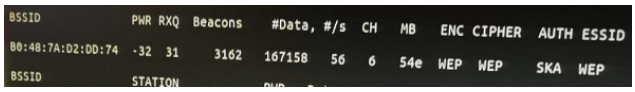


Figura 2: Red inalambrica del informante

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

Para obtener la contraseña necesitamos que hallan colisiones entre los vectores de inicialización. Para este caso, asumimos que los vectores de inicialización en esta red tipo WEP son de un largo de 24 bits, por lo que el espacio de vectores esta dado por:

$$H = 2^{24} = 16777216\ IVs$$

Luego, la cantidad de intentos que se necesitan antes de que halla una colisión con una probabilidad del 50 % esta dada por:

$$n(50\,\%) = 1{,}1774 \cdot \sqrt{H}$$

$$n(50\,\%) = 1{,}1774 \cdot \sqrt{2^{24}}$$

$$n(50\,\%) \approx 5000$$

Por lo tanto, se requiere de aproximadamente 5000 intentos para que halla una colisión de vectores de inicialización.

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

Primero se hace un escaneo a los paquetes que circulan en la red tipo WEP y se guarda el registro en un archivo .cap, esto se puede hacer gracias al comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop$
sudo airodump-ng -c 6 --bssid B0:48:7A:D2:DD:74 -w captura_lab_3 wlx1027f5518cfe
```

Figura 3: Rastreo de paquetes en la red tipo WEP

Luego de esto, se puede ejecutar el ataque por defecto que tiene Aircrack-ng usando los paquetes capturados anteriormente y aprovechándose del como se repiten los vectores de inicialización en las redes tipo WEP. Esto se hace mediante el comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop$
sudo aircrack-ng -b B0:48:7A:D2:DD:74 captura_lab_3-01.cap
```

Figura 4: Ataque por defecto de Aircrack

La key encontrada es la siguiente : **12:34:56:78:90**

2.4. Indica el tiempo que demoró en obtener la password

2.5. Descifra el contenido capturado

Ya teniendo la key de la red, podemos descifrar los mensajes de los paquetes capturados y guardados anteriormente en un archivo .cap. Para hacer esto, usamos el siguiente comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop/lab_cripto_3/capturas$
airdecap-ng -w 12:34:56:78:90 captura_lab_3-01.cap
Total number of stations seen            8
Total number of packets read        367832
Total number of WEP data packets    163389
Total number of WPA data packets         0
Number of plaintext data packets         1
Number of decrypted WEP  packets    163389
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 paquetes capturados con llave obtenida

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

Ya habiendo descifrado los paquetes del archivo .cap gracias al comando anterior, podemos analizar la data de los paquetes en texto plano gracias a Wireshark:

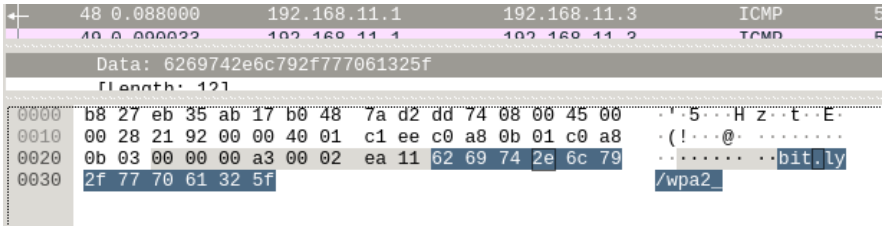


Figura 6: Url del archivo a descargar

3. Desarrollo (PASO 2)

3.1. Script para modificar diccionario original

Habiendo descargado el wordlist rockyou.txt, a partir de este ultimo creamos un nuevo wordlist en el cual se elimina cada palabra que empiece con un numero como primer carácter, en caso contrario se cambia su primer carácter por su versión capital y se le agrega un cero al final de la palabra. Para hacer esto se utilizó el siguiente script de Python:

```python
def process_word(word):
    if word[0].isdigit():
        return None
    else:
        new_word = word.capitalize() + '0'
        return new_word

with open('rockyou.txt', 'r', encoding='utf-8') as input_file:
    with open('rockyou_mod.dic', 'w', encoding='utf-8') as output_file:
        word_count = 0
        for line in input_file:
            word = line.strip()
            if word:
                new_word = process_word(word)
                if new_word:
                    output_file.write(new_word + '\n')
                    word_count += 1

print(f'\nEl nuevo archivo tiene {word_count} palabras.\n')
```

Figura 7: Script para generar el nuevo wordlist

3.2. Cantidad de passwords finales que contiene rockyou_mod.dic

Al ejecutar el script podemos ver los resultados finales junto a la cantidad de palabras generadas

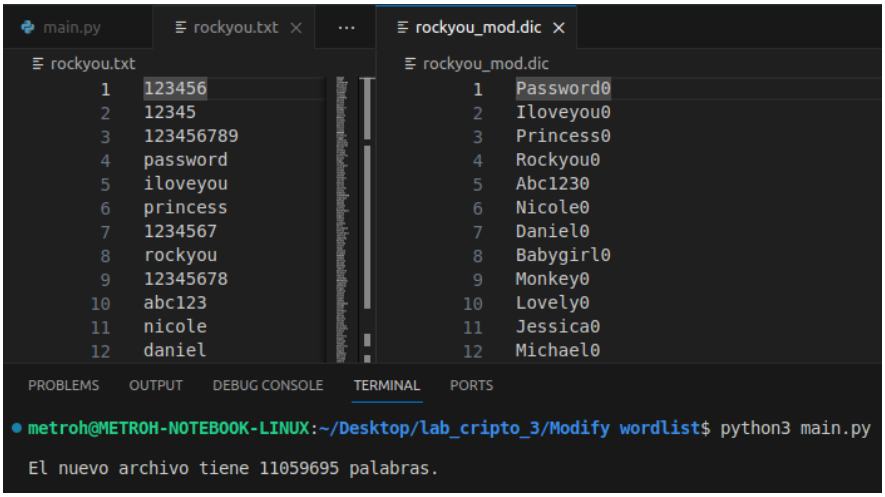


Figura 8: Resultados de la creación del nuevo wordlist

4. Desarrollo (Paso 3)

4.1. Obtiene contraseña con hashcat con potfile

Para obtener la contraseña a partir del archivo pcap que capturo el handshake usando Hashcat necesitamos convertir el archivo .pcap a un formato que Hashcat entienda (.hc2000), para eso se utilizo el programa Hashtools con el siguiente comando:

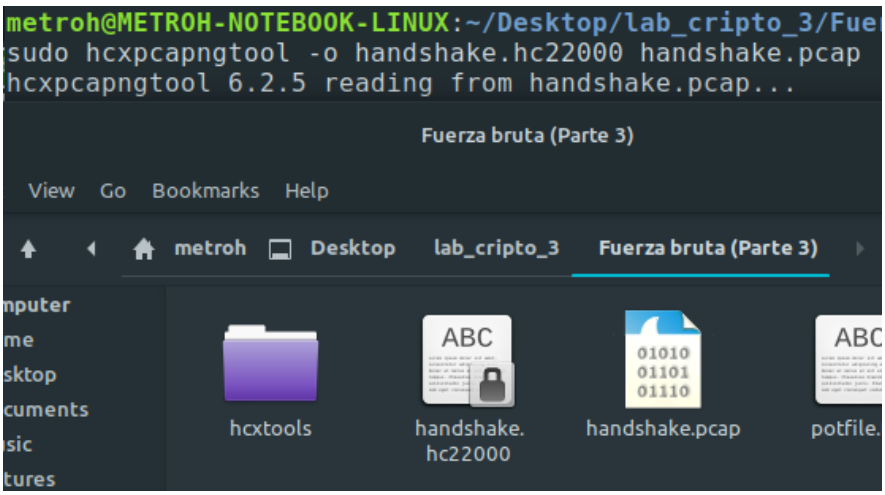


Figura 9: Comando para convertir archivo pcap a hc22000

Luego, intentamos descifrar la contraseña usando el wordlist modificado y el handshake convertido, guardando en un potfile el hash de la contraseña encontrada, usando al siguiente comando:

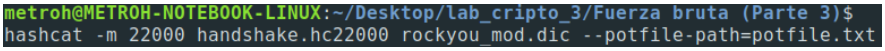


Figura 10: Comando de hashcat para descifrar contraseña usando potfile

El output generado en nuestro archivo potfile nos muestra el hash de la contraseña, información de la red y la contraseña en texto plano:

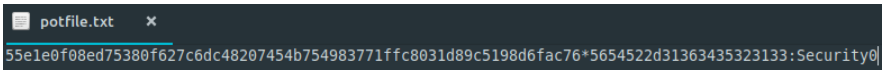


Figura 11: Contraseña descifrada con hashcat usando potfile

La contraseña de la red es : **Security0**

4.2. Nomenclatura del output

- **55e1e0f08ed75380f627c6dc48207454b754983771ffc8031d89c5198d6fac76**: Es el **PMKID** (Pairwise Master Key Identifier).

- **5654522d31363435323133**: Es el SSID (nombre) de la red Wi-Fi en formato hexadecimal. Si lo conviertes a ASCII, obtendrás **VTR-1645213**.
- **Security0**: Es la contraseña que Hashcat ha descifrado para esa red Wi-Fi.

4.3. Obtiene contraseña con hashcat sin potfile

Para descifrar la contraseña usando el wordlist modificado y el handshake convertido, sin guardar en un potfile el hash de la contraseña encontrada, ejecutamos el siguiente comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop/lab_cripto_3/Fuerza bruta (Parte 3)$
    hashcat -m 22000 handshake.hc22000 rockyou_mod.dic --show
1813acb976741b446d43369fb96dbf90:b0487ad2dc18:eede678cdf8b:VTR-1645213:Security0
```

Figura 12: Contraseña descifrada con hashcat sin usar potfile

El output de la terminal nos muestra el hash de la contraseña, información de la red y la contraseña en texto plano. La contraseña nuevamente es : **Security0**

4.4. Nomenclatura del output

- **1813acb976741b446d43369fb96dbf90**: Es el **PMKID** (Pairwise Master Key Identifier).
- **b0487ad2dc18**, **eede678cdf8b**: Son las direcciones MAC del punto de acceso y del cliente, respectivamente.
- **VTR-1645213**: Es el SSID (nombre) de la red Wi-Fi en texto plano
- **Security0**: Es la contraseña que Hashcat ha descifrado para esa red Wi-Fi.

4.5. Obtiene contraseña con aircrack-ng

Para allar la contraseña con aircrack-ng basta con utilizar nuestra wordlist modificada, la captura del handshake.pcap y el siguiente comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop/lab_cripto_3/Fuerza bruta (Parte 3)$
    aircrack-ng -w rockyou_mod.dic handshake.pcap
```

Figura 13: Comando de Aircrack-ng para descifrar contraseña

La contraseña encontrada gracias a Aircrack-ng se puede ver en la siguiente figura:

```
                          Aircrack-ng 1.6

[00:00:00] 4209/9296251 keys tested (10122.48 k/s)

Time left: 15 minutes, 17 seconds                          0.05%

                      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 14: Contraseña descifrada con Aircrack-ng

4.6. Identifica y modifica parámetros solicitados por pycrack

Primero bajamos el código del repositorio de Pycrack con el siguiente comando:

```
metroh@METROH-NOTEBOOK-LINUX:~/Desktop/lab_cripto_3$
git clone https://github.com/nogilnick/PyCrack
```

Figura 15: Comando para importar codigo de Pycrack

Luego, revisando el código y el blog referenciado en el repositorio nos damos cuenta de que hay que revisar la información de los paquetes capturados en el archivo .pcap con wireshark:

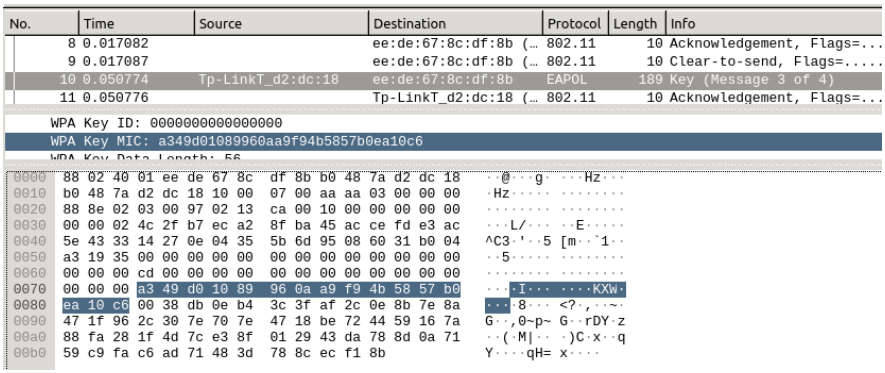
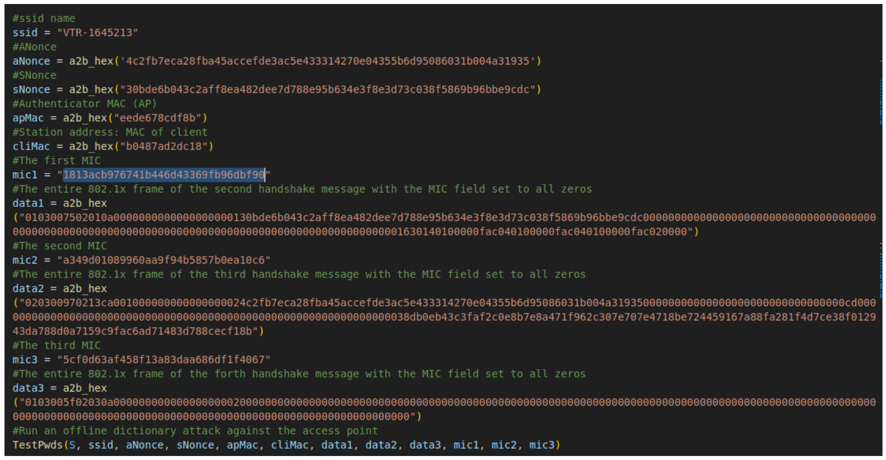


Figura 16: Ejemplo de revisión de los paquetes del handshake

Los datos reemplazados en el codigo se pueden ver en la siguiente figura:



```
#ssid name
ssid = "VTR-1645213"
#ANonce
aNonce = a2b_hex('4c2fb7eca28fba45accefde3ac5e433314270e04355b6d95086031b004a31935')
#SNonce
sNonce = a2b_hex("30bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c038f5869b96bbe9cdc")
#Authenticator MAC (AP)
apMac = a2b_hex("eede678cdf8b")
#Station address: MAC of client
cliMac = a2b_hex("b0487ad2dc18")
#The first MIC
mic1 = "1813acb976741b446d43369fb96dbf90"
#The entire 802.1x frame of the second handshake message with the MIC field set to all zeros
data1 = a2b_hex
("0103007502010a000000000000000130bde6b043c2aff8ea482dee7d788e95b634e3f8e3d73c038f5869b96bbe9cdc00000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000001630140100000fac040100000fac040100000fac020000")
#The second MIC
mic2 = "a349d01089960aa9f94b5857b0ea10c6"
#The entire 802.1x frame of the third handshake message with the MIC field set to all zeros
data2 = a2b_hex
("020300970213ca001000000000000000024c2fb7eca28fba45accefde3ac5e433314270e04355b6d95086031b004a3193500000000000000000000000000000000cd000
0000000000000000000000000000000000000000000000000000000000038db0eb43c3faf2c0e8b7e8a471f962c307e707e4718be724459167a88fa281f4d7ce38f0129
43da788d0a7159c9fac6ad71483d788cecf18b")
#The third MIC
mic3 = "5cf0d63af458f13a83daa686df1f4067"
#The entire 802.1x frame of the forth handshake message with the MIC field set to all zeros
data3 = a2b_hex
("0103005f02030a000000000000000200000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000")
#Run an offline dictionary attack against the access point
TestPwds(S, ssid, aNonce, sNonce, apMac, cliMac, data1, data2, data3, mic1, mic2, mic3)
```

Figura 17: Campos del código modificados para el caso

- **SSID**: Es el nombre de la red
- **ANonce**: Es el Wpa key nonce del primer paquete del handshake
- **SNonce**: Es el Wpa key nonce del segundo paquete del handshake
- **apMac**: La mac del punto de acceso
- **cliMac**: La mac del cliente
- **mic1**: Campo message integrity check del segundo paquete del handshake
- **data1**: Todo el campo 802.1x Authentication del segundo paquete del handshake, reemplazando por cero los caracteres hexadecimales que tengan el mic correspondiente
- **mic2**: Campo message integrity check del tercer paquete del handshake
- **data2**: Todo el campo 802.1x Authentication del tercer paquete del handshake, reemplazando por cero los caracteres hexadecimales que tengan el mic correspondiente
- **mic3**: Campo message integrity check del cuarto paquete del handshake
- **data3**: Todo el campo 802.1x Authentication del cuarto paquete del handshake, reemplazando por cero los caracteres hexadecimales que tengan el mic correspondiente

4.7. Obtiene contraseña con pycrack

A continuación se muestra el resultado de la ejecución del código:

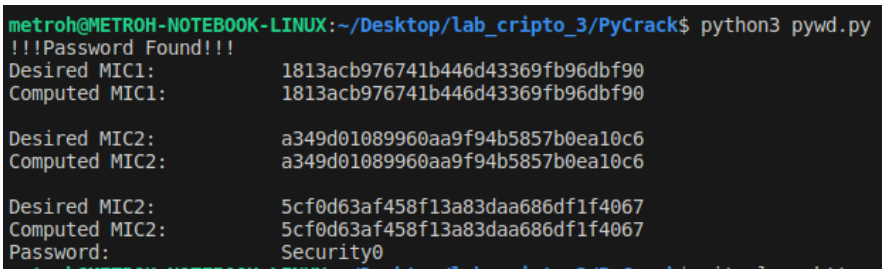


Figura 18: Campos del código modificados para el caso

Se puede ver que se encontró la contraseña **Security0**, la cual como comprobamos anteriormente, es correcta.

Conclusiones y comentarios

En este laboratorio, se ha demostrado la importancia de comprender y aplicar técnicas de seguridad en redes inalámbricas. Mediante el uso de herramientas como Aircrack-ng, Hashcat y Pycrack, pudimos llevar a cabo ataques controlados para recuperar contraseñas en situaciones específicas. Asimismo, se evidenció la necesidad de mantener contraseñas robustas y evitar el uso de métodos de encriptación obsoletos como WEP. Estos conocimientos resultan fundamentales en la protección de la información y la garantía de la confidencialidad en entornos de comunicaciones inalámbricas.