SWI - Laboratoire 1

Julien Huguet & Antoine Hunkeler

Mars 2020

Etape 1

Point a)

Pour la déauthentification avec aircrack, nous avons utilisé cette commande : `aireplay-ng -0 0 -a <MAC address access point> -c <MAC address station> <interface name>` dont le premier `-0` siginifie que nous envoyons un paquet de déathentification et le deuxième `0` signifie d'envoyer plusieurs paquets continuellement. Il est possible de remplaçer par `1` pour indiquer d'envoyer un seul paquet.

Lien : https://www.aircrack-ng.org/doku.php?id=deauthentication

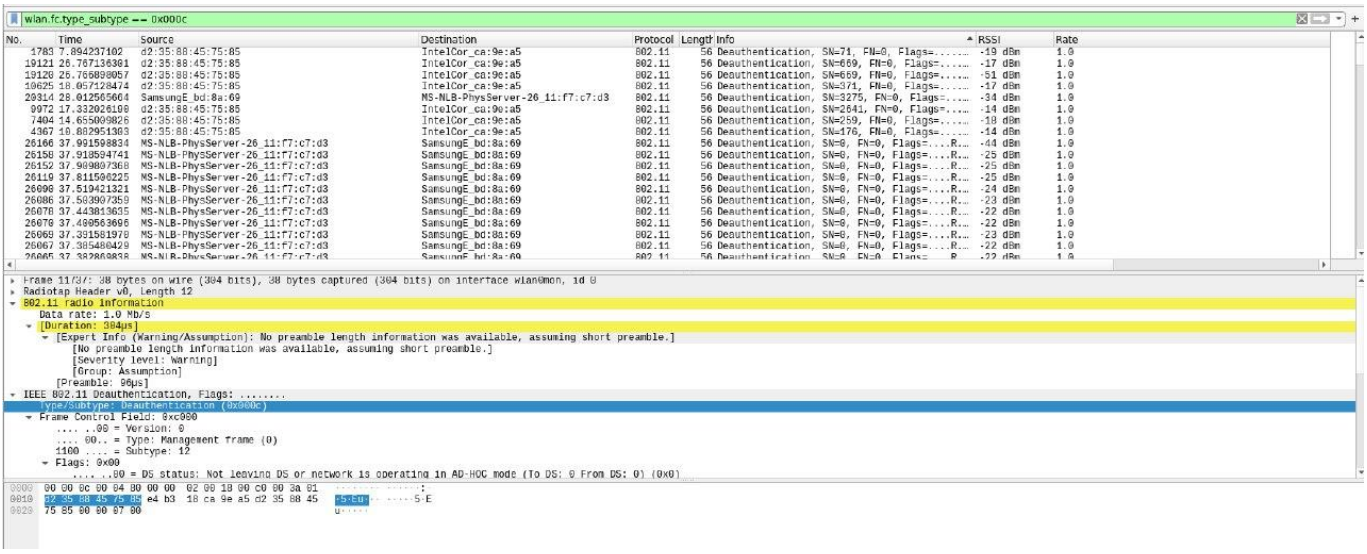
Quel code est utilisé par aircrack pour déauthentifier un client 802.11?

aircrack-ng utilise le code 0x0007 comme le montre la capture d'écran ci-dessous. L'interprétation possible est qu'il retire la station parce qu'elle tentait d'envoyer une trame à un point d'accès dont elle nêst plus associé.

```
▾ IEEE 802.11 Wireless Management
  ▾ Fixed parameters (2 bytes)
      Reason code: Class 3 frame received from nonassociated STA (0x0007)
```

A l'aide d'un filtre d'affichage, essayer de trouver d'autres trames de déauthentification dans votre capture. Avez-vous en trouvé d'autres ? Si oui, quel code contient-elle et quelle est son interpretation ?

Ci-dessous une capture d'écran représentant d'autres trames capturées incluant un exemple d'une trame de déauthentification :



Point b)

Quels codes/raisons justifient l'envoie de la trame à la STA cible et pourquoi ?

Le code 1 avec la raison Unspecified peut être envoyé à la station cible, car la raison peut être quelconque, donc elle peut aussi être envoyé à un point d'accès.

Le code 5 peut être aussi envoyé à une station pour signifier un problème de prise en compte de toutes les stations associés au point d'accès.

Le code 4 peut être envoyé à la station pour avertir sa déathentification due à son inactivité.

Quels codes/raisons justifient l'envoie de la trame à l'AP et pourquoi ?

Le code 8 est envoyé au point d'accès pour avertir qu'une station associée a quitté la BSS.

Comment essayer de déauthentifier toutes les STA ?

Il est possible d'essayer de déauthentifier toutes les stations en remplaçant l'adresse MAC de destination par `ff:ff:ff:ff:ff:ff` qui correspond à l'adresse MAC broadcast pour envoyer des trames à toutes les stations dans le même réseau.

Quelle est la différence entre le code 3 et le code 8 de la liste ?

Le code 3 signifie que des paquets de déathentification sont envoyés car une station est en train de partir ou a quitté le IBSS ou ESS.

Le code 8 signifie que la station est cette fois-ci disassocié du point d'accès, car il n'est plus présent dans le BSS.

Expliquer l'effet de cette attaque sur la cible

Lors de l'exécution du script, si la station envoie des requêtes ICMP (`ping`), il verra qu'il ne pourra plus atteindre le destinataire.

Aussi, ce script engendre des problèmes réseau, comme un déni de service pour le point d'accès.

Liens :

- https://mrncciew.com/2014/10/11/802-11-mgmt-deauth-disassociation-frames/
- https://support.zyxel.eu/hc/en-us/articles/360009469759-What-is-the-meaning-of-802-11-Deauthentication-Reason-Codes-
- https://kb.fortinet.com/kb/documentLink.do?externalID=FD37576

Script

Ci-dessous les explications du fonctionnement du script à l'aide de capture d'écran.

Pour pouvoir tester le bon fonctionnement du script, nous avons créé un hotspot via un téléphone portable. L'ordinateur est donc connecté à cet hotspot afin d'effectuer les tests des reasons codes utilisé.

Une fois la connexion établie, nous avons lancé la commande `ping 8.8.8.8 -t` qui permettra de voir à quel moment l'ordinateur est déconnecté de l'AP. Voici une capture comportant les pings avant l'utilisation du

script :

```
Pinging 8.8.8.8 with 32 bytes of data:
Reply from 8.8.8.8: bytes=32 time=49ms TTL=52
Reply from 8.8.8.8: bytes=32 time=62ms TTL=52
Reply from 8.8.8.8: bytes=32 time=29ms TTL=52
Reply from 8.8.8.8: bytes=32 time=35ms TTL=52
```

Ensuite, nous ponvons lancer le script qui va dans un premier temps demander à l'utilisateur quelle reason code il souhaite utiliser entre le 1, 4, 5 ou 8:

```
Reason code available : 1 - 4 - 5 - 8
Choose the reason code :
```

Pour finir le script va générer des paquets de type deauth afin de déconnecter l'AP ou le client du réseau. Voici ce que nous obtenons lors du ping :

```
Request timed out.
```

Nous obtenons donc une réponse de type timed out lors de l'execution du script. L'ordinateur ne dispose plus d'un accès à l'AP lors de l'execution du script.

Etape 2

Point a)

Expliquer l'effet de cette attaque sur la cible

L'objectif de cette attaque est de sniffer à la recherche de points d'accès, sélectionner un point d'accès cible et de le proposer sur un canal différent avec un décalage de 6 et de pertuber la connexion du point d'accès cible afin de déconnecter les stations présentes pour qu'elles se connectent sur le point d'accès attaqué.

Cette attaque nous permet de sniffer directement sur notre fake AP les différents paquets émis sur la connexion. Il est alors possible de retrouver des informations confidentielle de la cible.

Nous n'avons malheuresement pas réussi à compléter ce script, faute de temps et de manque de connaissances en Python. La liste contient plusieurs fois le SSID, malgré un test pour tester sa présence ou non dans la liste.

Aussi, le script n'inclue pas la sélection d'un SSID et l'envoi d'un beacon sur un channel différent.

Néanmoins, nous avons quelques pistes pour le changement de channel :

- Proposer une entrée utilisateur pour sélectionner un SSID capturé
- Forger une trame beacon avec ce SSID sur un channel différent
- Ou une autre solution est de stocker dans un tableau le paquet capturé et ensuite le sélectionner et modifier son channel (et non pas le channel de l'interface) avant l'envoi.

Voici un exemple de capture :

	BSSID	SSID	dBm_Signal	Channel
0	c9:39:23:04:ec:6c	mg130d0orz	None	None
1	c9:39:23:04:ec:6c	mg130d0orz	None	None
2	c9:39:23:04:ec:6c	mg130d0orz	None	None
3	c9:39:23:04:ec:6c	mg130d0orz	None	None
4	c9:39:23:04:ec:6c	mg130d0orz	None	None
...	...	...	...	...
2488	c9:39:23:04:ec:6c	mg130d0orz	None	None
2489	c9:39:23:04:ec:6c	mg130d0orz	None	None
2490	c9:39:23:04:ec:6c	mg130d0orz	None	None
2491	c9:39:23:04:ec:6c	mg130d0orz	None	None
2492	c9:39:23:04:ec:6c	mg130d0orz	None	None

Notre hypothèse pour les effets de cette attaque est que un utilisateur peut voir un SSID, qui est en fait attaqué, se connecter dessus et l'attaquant peut voir les données de l'utilisateur, car il n'a aucune idée du concept des channels dans les réseaux sans fil.

Etape 3

Ci-dessous quelques captures d'écran pour montrer et expliquer le fonctionnement du script.

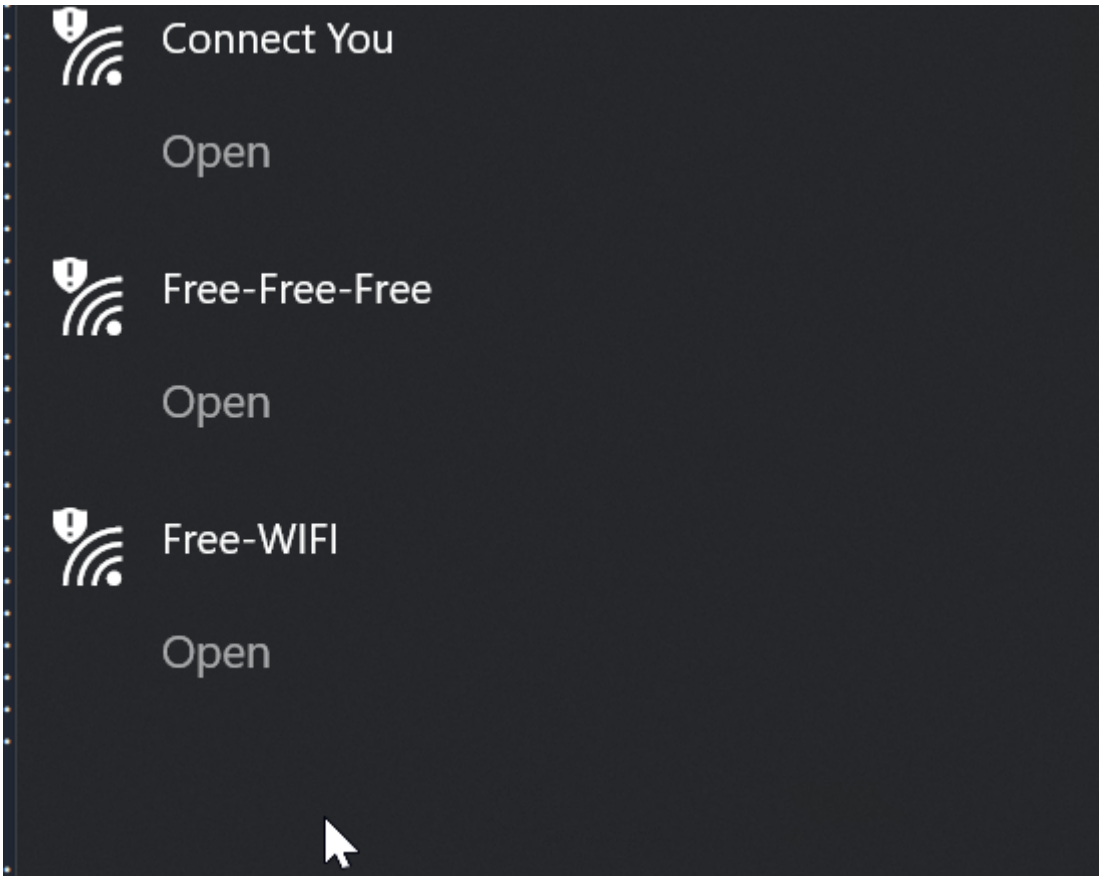
Tout d'abord, le fichier contient ces noms de faux SSID :

```
Free-WIFI
Free-Free-Free
Connect You
FREE-Cafeteria
Yverdon-Free-Wifi
```

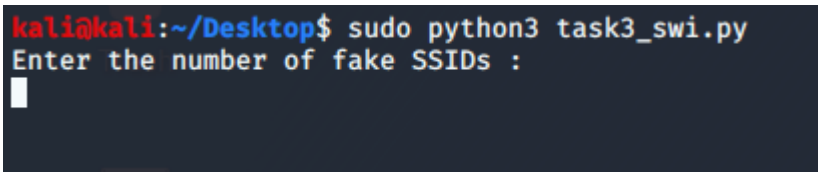
Une fois le script lancé, il va prendre automatiquement chaque ligne, forger une trame de beacon 802.11 et l'envoyer 10000 fois, qui est suffisant pour que le faux SSID s'affiche dans les réseaux sans fils disponibles.

```
kali@kali:~/Desktop$ sudo python3 task3_swi.py
SSID : Free-WIFI
```

```
Sent 10000 packets.
SSID : Free-Free-Free
```



Si le fichier n'existe pas, il demande le nombre de SSID à générer à l'utilisateur



Ensuite de cela, il génère deux faux SSID et envoie aussi 10000 fois une trame beacon forgé.