République **A**lgérienne **D**émocratique et **P**opulaire

Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DE **G**ENIE **E**LECTRIQUE ET **D**'INFORMATIQUE

DEPARTEMENT D'ELECTRONIQUE

Mémoire de Fin d'Etude de

MASTER ACADEMIQUE

Spécialité : **Réseaux et Télécommunication**

Filière : **Génie Electrique**

Thème

Conception et développement d'un système de détection d'intrusions sans fil(WIDS)

mémoire soutenu publiquement le 28/09/2016 devant les membres de jurys composée de :

Le présidant :	Lazri Mourad	**MCA**
Encadreur :	LAHDIR Mourad	**MCA**
Les examinateurs :	BOUZBOUDJA Ourida	**MAA**
	ATTAF Youcef	**MCB**

Présenté par:

Melle : TAIBI Lilia

Melle : OUERDANE Fatima

Co-encadreur:

KADDOUR Boumediene

Promotion 2015/2016

Remerciements

En premier lieu, nous remercions notre Dieu le tout puissant de nous avoir donné la foi, la santé et nous a permit de bien mener ce travail.

Nous tenons à remercier en cette occasion tout le corps professoral et administratif de département d'électronique de l'université ***Mouloud Mammeri de Tizi-Ouzou*** *pour la richesse et la qualité de leurs enseignements et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.*

Nous tenons remercier notre Encadreur ***Mr KADDOR Boumediene*** *pour son aide tout au long de la réalisation de ce mémoire, et ses orientations et son soutien.*

Nous tenons aussi à remercier sincèrement notre promoteur ***Mr LAHDIR Mourad*** *pour ses remarques et ses conseils.*

Nous exprimons également notre gratitude aux membres du jury, qui nous ont honorés en acceptant de juger ce modeste travail.

Nous souhaitons d'adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire.

Dédicaces

Je dédie ce modeste travail à :

Mes très chers parents qui m'ont soutenu tout au long de mes études et qui ont contribué à ma réussite, que dieu les garde et leurs donne une longue vie.

Mes très chers frères Mohamed lamine et Sidali auxquels je souhaite une bonne réussite dans leurs études et leurs vie.

Mes très chers sœurs Sabrina, Kamilia et leurs maris Walid et Farid.

Mon fiancé Mohamed, celui à qui je souhaite une longue vie plein de joie, de bonheur et de santé.

Mon ange Wassim.

Toute ma famille.

Tous mes amis.

Mon binôme Fatima.

Lilia

Dédicaces

Je dédie ce modeste travail à

A mes chers parents, pour leur patience, leurs sacrifices, leur tendresses et soutien durant mes études, aucun hommage ne pourrait être à la hauteur de l'amour dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie.

A mes très chers frères Djamel, Kaci et sa femme Nabila.

A ma très chère sœur Ouahiba et son mari Hakim.

A mes princes Ilyas, Nélia, Chanez,Adem.

A tous mes chers amis.

A mon binôme Lilia.

Et à tous ceux qui ont contribué de près ou de loin pour leurs soutiens, je vous dis merci.

FATIMA

Glossaire

AES: Advanced Encryption Standard.

AP: Access Point**.**

ARP: Adresse Résolution Protocol.

BSS: Basic Service Set**.**

BSSID: Basic Service Set Identifier.

CCA: Clear Channel Assessment.

DS: Distribution System.

ESS: Extended Service Set.

ESSID: Extended Service Set Identifier.

FCS: Frame Check Sequence.

IBSS: Independent Basic Set Service.

IDLE : Integrated Development Environment.

IEEE: Institute of Electrical and Electronics Engineers

IV: Initialisation Vector.

LLC: Logical Link Control.

MAC: Medium Access Controller**.**

MIC: Message Integrity Code.

OSI: Open Systems Interconnection.

PLCP: Physical Layer Control Protocol.

PMD: Physical Medium Dependant.

PSK: Pré-Shared Key..

RADIUS : Remote Authentication Dial In User Service.

Glossaire

RADIUS : Remote Authentication Dial In User Service.

RC4: Ron’s Code #4.

RTS: Request To Send.

SSID: Service Set Identifier.

STA: Station.

TKIP: Temporal Key Integrity Protocol.

WECA: Wireless Ethernet Compatibility Alliance.

WEP: Wired Equivalent Privacy.

Wi-Fi: Wireless Fidelity. .

WLAN: Wireless Local Area Network.

WPA: Wireless Protected Access.

WPA2: Wi-Fi Protected Access version 2.

WPS: Wi-Fi Protected Setup.

Liste des figures

CHAPITRE I : Les bases du réseau Wi-Fi

CHAPITRE II : Sécurité Wi-Fi et ses faiblesses

CHAPITRE III : Développement d'un système de détection d'intrusions sans fil WaveShark

Liste des tableaux

Sommaire

Sommaire

ChapitreIII :Développement d'un système de détection d'intrusions sans filWaveShark

Introduction générale

Introduction générale

Les réseaux sans fil ont été créés pour permettre aux utilisateurs d'effectuer des communications sans utiliser les câbles de connexion. De ce fait, nous avons assisté ces dernières années à un essor en puissance des réseaux locaux sans fil, ou encore le Wi-Fi, qui sont en passe de devenir l'une des principales solutions de connexion.

La sécurité du réseau Wi-Fi constitue un enjeu crucial, du fait que ce dernier est potentiellement vulnérable à des attaques nombreuses. Parmi ces attaques, on trouve le vol d'informations confidentielles, destructions de données, voir même des dégâts matériels. L'application d'une stratégie de sécurité efficace est l'étape la plus importante qu'on doit franchir pour protéger ce réseau.

Dans ce travail, nous nous intéressons à la sécurité du réseau Wi-Fi. A cet effet, les différents mécanismes de sécurité sont étudiés afin de mieux comprendre le fond d'interception des données transmises, nous représenterons les différentes types d'attaques les plus fréquentes sur ce réseau. Aussi nous avons conçu et développé l'application ***WaveShark*** pour détecter ces attaques.

Pour mettre en évidence notre travail, nous avons décomposé notre mémoire en trois chapitres.Dans le premier chapitre, nous avons définis les principales bases du réseau sans fil Wi-Fi.Le deuxième chapitre est consacré à l'étude de l'évolution des différents mécanismes de sécurité du réseau Wi-Fi, ainsi que les différentes attaques qui peuvent être appliquées sur ces mécanismes.Dans le dernier chapitre, nous avons développé notre application ***WaveShark*** en utilisant le langage de programmation Python.Nous terminons notre mémoire par une conclusion générale ainsi que par des perspectives permettant d'améliorer notre travail.

Chapitre I :

Les bases du réseau Wi-Fi

1. Préambule

Les réseaux sans-fil connaissent actuellement un succès très important. Ils offrent en effet, une flexibilité largement supérieure aux réseaux filaires, en s'affranchissant notamment des problèmes de câblage et de mobilité des équipements. Il existe plusieurs types des réseaux sans fil, en particulier le réseau Wi-Fi, qui est une technologie actuellement très utilisée soit chez lesparticuliers ou bien au niveau des entreprises.

Avant de développer la partie technique de ce travail, il nous a semblé nécessaire de définirdans ce premier chapitre toutes les notions de bases du terme Wi-Fi, afin de mieux comprendre la suite de notre travail.

2.IEEE

IEEE est une norme professionnelle qui compte plus de 400 000 membres, elle possède différentes branches dans plusieurs parties du monde, IEEE est constituée d'ingénieursélectriciens, informaticiens, etc. L'organisation a pour but de promouvoirla connaissance dans le domaine de l'ingénierie électrique (électricité et électronique).

3.Le Wi-Fi

Le Wi-Fi est le nom commercial donné à la norme IEEE 802.11 par la Wi-Fi Alliance. Le Wi-Fi est une technologie de transmission sans fil qui utilise les ondes électromagnétiques, dans la pratique il permet de relierdes objets communicants à une liaison haut débit, sur un rayon de plusieurs dizaines de mètres.

3.1. Evolution de la norme IEEE 802.11

La norme IEEE 802.11 est en réalité la norme initiale offrant des débits de 1 ou 2 Mbps, des révisions ont été apportées à la norme originale afin d'optimiser le débit c'est le cas des normes 802.11b, 802.11a, 802.11g et 802.11n.

3.1.1. Les normes physiques

3.1.1.1. IEEE 802.11b (Wi-Fi)

Cette norme est la norme la plus répondueactuellement, elle propose un débit (11 Mbps théorique ,6Mbps réel) avec une portée de 300 mètres, la plage de fréquence utilisée est la bande de 2.4 GHz.

3.1.1.2. IEEE 802.11a (Wi-Fi 5)

La norme IEEE 802.11a permet d'obtenir un haut débit (54 Mbps théorique, 30 Mbps réel), avec une faible portée de 15 mètres,elle utilise une bandede fréquence de 5GHz.

3.1.1.3 IEEE 802.11g

La norme IEEE 802.11g offre un haut débit (54 Mbps théorique, 30 Mbps réel) sur la bande de fréquence de 2.4 GHz. Cette norme a une compatibilité avec la norme IEEE 802.11b, ce qui signifie que des matériels conformes à la norme IEEE 802.11g peuvent fonctionner en IEEE 802.11b.

3.1.1.4. IEEE 802.11n

La norme IEEE 802.11n permet d'atteindre un débit théorique 300Mbps et une portée de 100m, cette norme supporte deux bandes de fréquences différentes 2 ,4 GHz et 5 GHz.

3.1.2. Les normes d'améliorations

3.1.2.1. IEEE 802.11i

La norme 802.11i est connue sous le nom WPA2, elle a pour but d'améliorer la sécurité des transmissions, et des échanges au niveau des réseaux informatiques locaux utilisant une liaison sans fil.

3.1.2.2.802.1x

Le standard 802.1x est une solution de sécurisation mise au point par l'IEEE, il permet d'authentifier un utilisateur souhaitant accéder à un réseau (filaire ou Wi-Fi) grâce à un serveur d'authentification, ce serveur est souventappelé Serveur RADIUS.

Le 802.1x se base sur trois éléments :

Suppliant:c'estle client qui demande à s'authentifier avant de pouvoir accéder au réseau.

- **La borne d'accès :**c'est l'équipement réseau (point d'accès) auquel le client se connecte, suivant la réponse du serveur d'authentification la borne laissera passer ou non le trafic du client.
- **Le serveur d'authentification:**ce serveur vérifie la demande de la borne d'accès si le client peut ou non accéder au réseau.

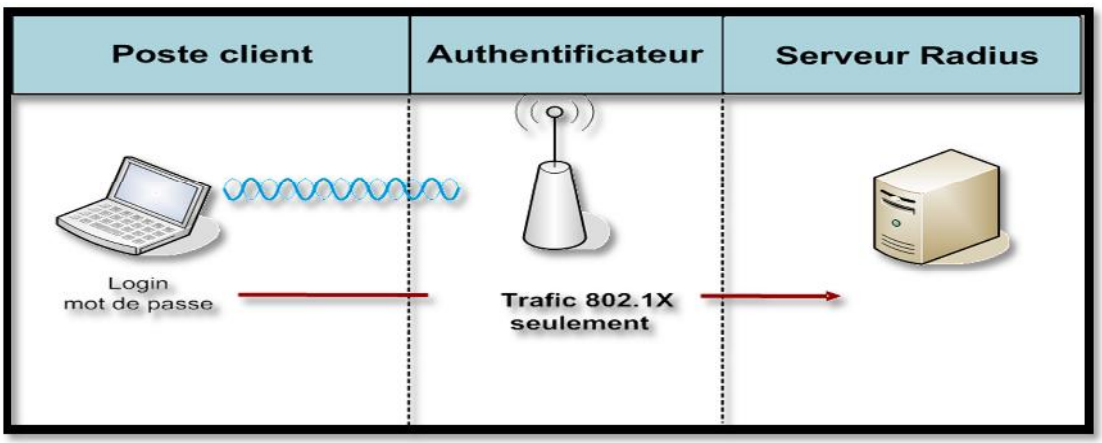

Figure 1 : Principe de la norme 802.1x.

3.2. Les modes de fonctionnement du réseau Wi-Fi

3.2.1. Le mode infrastructure (Managed)

En mode infrastructure chaque station cliente (notée STA) se connecte à un point d'accès via une liaison sans fil.

.

Figure 2 : Le mode infrastructure.

Il est possible de relier plusieurs points d'accès entre eux, ou plus exactement plusieurs BSS par une liaison appelée système de distribution DS, afin de constituer un ensemble de services étendus ESS. Un ESS est repéré par un ESSID et aussi un BSS est identifié par un BSSID.

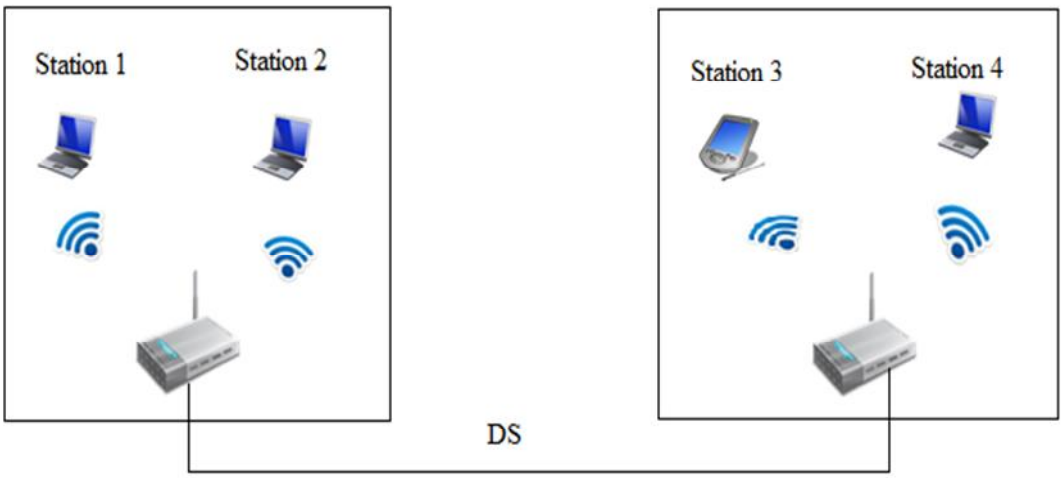

Figure 3 : Système de distribution.

- **La connexion des stations clientes aux réseaux sans**

  Soit la station cliente a été configurée pour se connecter à un réseau sans fil, elle possèdedonc le nom du réseau sans fil. Lorsque la station est à la portée d'un point d'accès, elle diffuse une requête de sondage (probe request) contenant l'ESSID pour lequel elle est configurée.

  Soit aucun ESSID n'aété configuré, la station se mis alors à l'écoute et à la recherche d'un ESSID.

  En effet, chaque point d'accès diffuse régulièrement toutes les 0.1 secondes une trame balise (beacon),ces trames balises donnent des informations sur le BSSID du point d'accès, ses caractéristiques et éventuellement son ESSID. En écoutant les trames balises, la station peut découvrir quels sont les réseaux sans-fils disponibles et en sélectionnant un en envoyant une requête de sondage.

- **Le traitement des requêtes par le point**

  A chaque requête de sondage reçue, le point d'accès vérifie l'ESSID, si l'ESSID correspond à celui du point d'accès, ce dernier envoie une réponse contenant

des informations sur sa charge. En effet d'une manière générale, plus un point d'accès est proche, meilleur est le débit.

3.2.2. Le mode Ad-Hoc (peer to peer)

En mode Ad-Hoc les machines sans fil clientes se connectent les unes aux autres directement afin de constituer un réseau point à point, c'est-à-dire un réseau dans lequel chaque machine joue au même temps le rôle du client et le rôle du point d'accès. L'ensemble formé par les différentes stations est appelé IBSS.

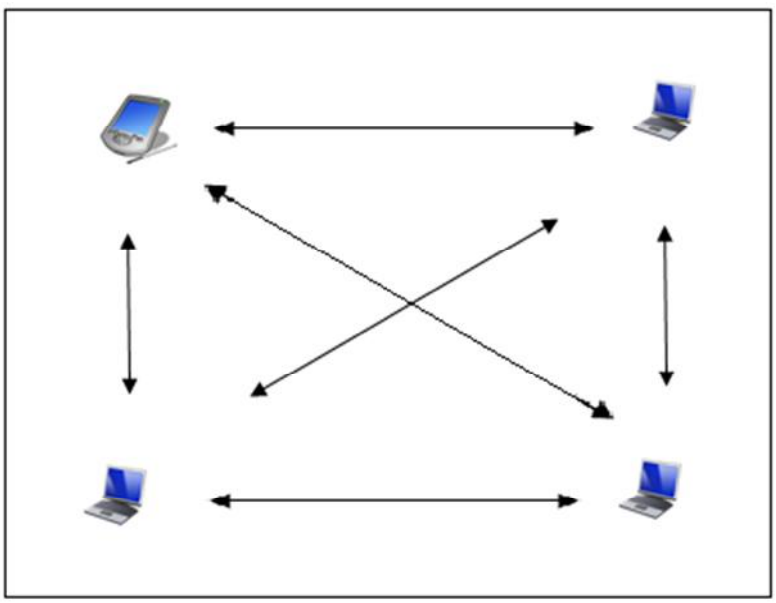

Figure 4 : Le mode Ad Hoc.

3.2.3. Le mode moniteur

Ce mode permet d'intercepter tous les paquets transitant dans l'air, qui ne sont pas forcément à destination de notre machine. Le mode moniteur n'existe que sur les réseaux sans fil.

3.3. Les couches IEEE802.11

La norme IEEE 802.11 repose sur une architecture en couche définie par le standard IEEE etcouvre les deux premières couches du modèle OSI, c'est à dire la couche physique et lacouche liaison de données.

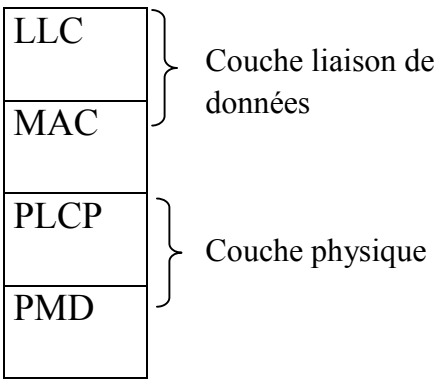

Figure 5 : Les couches IEEE 802.11.

a. La couche physique

Elle assure la transmission des données sur le support, elle est constituée de deux souscouches : **PMD** et **PLCP.**

- **La sous couche PMD**

Elle spécifie le type de support de transmission.

- **La sous couche PLCP**

Elle s'occupe de la détection du support et fournit un signal appelé CCA à la sous couche MAC pour lui indiquer si le support est occupé ou non.

b.La couche liaison de donnée :

Elle est aussi composée de deux sous couches.

- **Sous couche LLC**

La sous couche LLC 802.11 a pour rôle d'adapter les données venant des couches supérieures à la couche physique.

- **Sous couche MAC**

Le fonctionnement de la sous couche MAC est d'écouter le canal, attendre s'il est occupé puis transmettre les données lorsqu'il sera libéré.

3.4. Les différents types de la trame Wi-Fi

Il existe trois types de trames Wi-Fi, les trames de gestion, les trames de données, les trames de contrôle.

a.Les trames de gestion

Elles sont utilisées lors des procédures d'association et de l'authentificationd'une station avec le point d'accès.

b.Les trames de données

Elles contiennent les données utilisateurs, notamment les adresses source, destination, ce qui permet aux points d'accès d'acheminer correctement les trames vers leurs destinations.

c.Les trames de contrôle

Elles sont utilisées pour contrôler l'accès au support.

3.5) Format de la trame MAC

La trame MAC est la trame encapsulée au niveau de la sous couche MAC, chaque trame est constituée d'un en-tête (appelé MAC header, d'une longueur de 30 octets), d'un corps et d'un FCS.

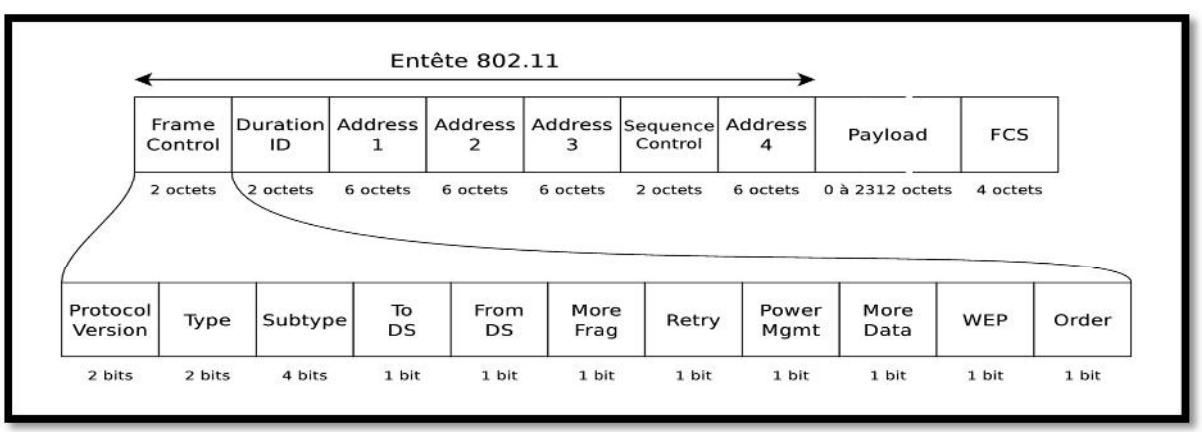

Figure 6: La trame MAC de l'IEEE 802.11.

- **Frame Control (FC) :** est constitué des informations suivantes :
  - **Version de protocole:** cechamp permettra de prendre en compte les évolutions de version du standard 802.11. La valeur est égale à zéro pour la première version.
  - **Type et Sous-type:** ces champs, respectivement de 2 et 4 bits, définissent le type et le sous-type des trames.

Type	00	01	10	11
Nature	Gestion	Contrôle	Données	Réservé

Tableau 1 : Types de trames.

  - **To DS:** Ce bit vaut 1 lorsque la trame est destinée au système de distribution (DS), il vaut zéro dans les autres cas. Toute trame envoyée par une station à destination d'un point d'accès possède ainsi un champ To DS positionné à 1.
  - **From DS:** ce bit vaut 1 lorsque la trame provient du système de distribution (DS), il vaut zéro dans les autres cas. Ainsi, lorsque les deux champs To et From sont positionnés à zéro il s'agit d'une communication directe entre deux stations (mode ad hoc).
  - **More Fragments (d'autres fragments):** ce bit est mis à 1 quand il y a d'autres fragments qui suivent le fragment en cours, il est à 0 s'il ne reste plus de fragments à transmettre. Un ensemble de fragments forme un paquet.
  - **Retry (Retransmission):**ce champ renseigne si la trame est transmise pour la première fois, ou si elle est retransmise.
  - **Power management (gestion d'énergie):**ce champ indique l'état de la station après la transmission, si le bit est à 0, la station terminale est en mode normal, si le bit est à 1, la station terminale est en état d'économie d'énergie.
  - **More Data (d'autres données) :**le point d'accès utilise ce champ pour indiquer à une station terminale en état d'économie d'énergie, s'il a ou non des trames en attente qui lui sont destinées.
  - **WEP:** ce bit indique que l'algorithme de chiffrement WEP a été utilisé pour chiffrer le corps de la trame.

- **Order (ordre):** Ce champ permet de vérifier si l'ordre de réception des fragments est le bon.

- **Durée / ID:** Ce champ indique la durée d'utilisation du canal de transmission.
- **Adresse 1:** est toujours l'adresse du récepteur (i.e. la station de la cellule qui est le récepteur du paquet).
- **Adresse 2:** est toujours l'adresse de l'émetteur (i.e. celui qui transmet le paquet)
- **Adresse 3:** est l'adresse de l'émetteur original quand le champ FromDS est à 1, sinon, et si ToDS est à 1, Adresse 3 est l'adresse destination.
- **Adresse 4:** est spécialement utilisée dans le cas d'une communication entre deux points d'accès faisant intervenir le système de distribution DS. Les bits To DS et From DS seront donc tous les deux à 1.
- **Contrôle de séquence:**

Cechamp permet de distinguer les divers fragments d'une même trame, il est composé de deux sous-champs permettant de réordonner les fragments :

- Le numéro de fragment
- Le numéro de séquence

- **FCS:**Une somme de contrôle servant à vérifier l'intégrité de la trame**.**

4.Discussion

La croissance du développement des technologies sans fil, et particulièrement la technologie Wi-Fi, ne cesse pas de s'arrêter et cela afin de satisfaire les utilisateurs, en leurs offrants une liberté de se déplacer tout en gardant la connectivité,mais la sécurité dans ce domaine reste un sujet très délicat, car depuis l'utilisation de ce type du réseau, plusieurs failles ont été détectées.

Dans ce chapitre,nous avons présenté quelques définitions nécessaires du réseau Wi-Fi avec des explications simples à assimiler.

Dans le chapitre suivant, on présentera les différents mécanismes de sécurité, et les attaques qui permettent de tester leurs degrés de sécurité.

Chapitre II :

La Sécurité WI-FI et ses faiblesses

1. Préambule

L'installation d'un réseau sans fil avec le manque de la mise en place d'éléments de protection, il est totalement permissif aux personnes indésirables d'écouter, de modifier et d'accéder à ce réseau facilement. La sécurité atoujours été le point faible des réseaux Wi-Fi, il est donc indispensable de mettre en place un grand nombre de mécanismes de sécurité.

Dans ce chapitre, nous allons décrire l'évolution des différents mécanismes de sécurité Wi-Fi. Afin de mieux comprendre le fond d'interception, nous représenterons les attaques les plus fréquentes sous KALI LINUX.

2.Les principes de la sécurité informatique

La sécurité informatique est l'ensemble de moyens mises en œuvre pour minimiser lavulnérabilité d'un système contre les menaces, ce qui implique la réalisation des fonctions essentielles suivantes :

- **Disponibilité**

  Les services (ordinateurs, périphériques….) et les informations (données, fichiers) doivent être accessibles aux personnes autorisées quand elles en ont besoin.

- **Confidentialité**

  Les informations ne peuvent être lues que par les personnes autorisées.

- **Intégrité**

  Les informations ne peuvent être modifiées que par les personnes autorisées.

- **Authentification**

  Consiste à assurer que seules les personnes autorisées aient accès aux ressources.

3.Mécanismes de sécurité Wi-Fi et ses faiblesses

La sécurité d'un réseau Wi-Fi est l'élément essentiel,qui s'est évoluée avec le temps, nous allons détailler cette évolution pour en comprendre la finalité. Au début le réseau Wi-Fin'était pas sécurisé, donc tout le monde ait accès, de plus, les messages transmis n'étaient pas cryptés, c'est le cas d'un système ouvert.

3.1. Système ouvert

- Le client diffuse une requête (probe request) pour sonder les réseaux présents dans le voisinage.
- Les points d'accès présents dans la zone répondent.
- Le client décide quel point d'accès utilise et envoie une demande d'authentification.
- Le point d'accès renvoie une réponse pour l'authentification.
- Une fois l'authentification réussie, le client envoie une demande d'association avec le point d'accès.
- Le point d'accès répond à cette demande d'association.
- Le client peut maintenant utiliser le point d'accès pour accéder àce réseau si toutes les étapes précédentes ont réussi.

3.1.1.Faiblesses d'un système ouvert

Dans ce type du réseautoute personne non autorisépeutécouter, intercepter et modifier les données échangées au niveau du réseau, ce qui pose de sérieux problèmes de sécurité, c'est pourquoi les chercheures ont mis en œuvre de plus en plus des mécanismes de sécurité plus fiables, qu'on va détailler chronologiquement ci-dessous.

3.2. Cacher le SSID

Afin d'accéder à tout réseau Wi-Fi, il est nécessaire de connaitre son identifiant, c'est-à-dire son SSID. Comme nous l'avons vu dans lechapitre précèdent, le point d'accès envoie dans des intervallesréguliers son SSID en broadcast pour informer de sa présence, ce qui permet au réseau d'apparaître dans la liste des réseaux disponibles.

Mais sachez qu'il y a possibilité de le cacher, simplement en désactivant le broadcast du SSID qui se trouve dans les fameuses beacon frames sur notre point d'accès, donc ce réseau n'apparait pas dans la liste des réseaux disponibles, il faut saisir manuellement son nom pour pouvoir s'y connecter.

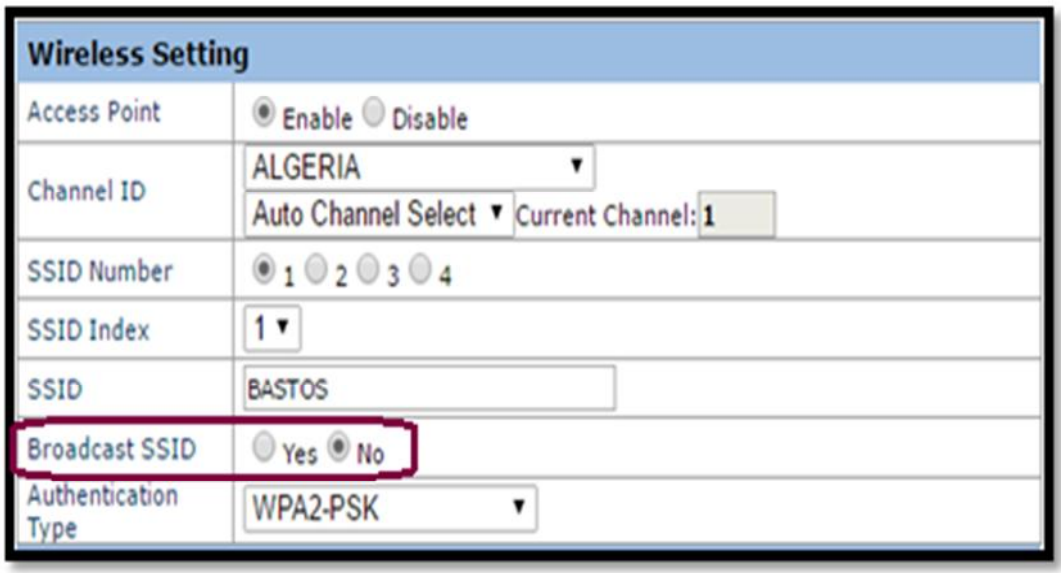

Figure7 : Désactiver le broadcaste du SSID.

Pour ce qui suit, nous allons définir les différentes étapes à suivre afin de détecter les réseaux sans fil disponibles.

- La saisie de la commande airmon-ngaffiche l'état des interfaces.
  Usage : Airmon-ng.

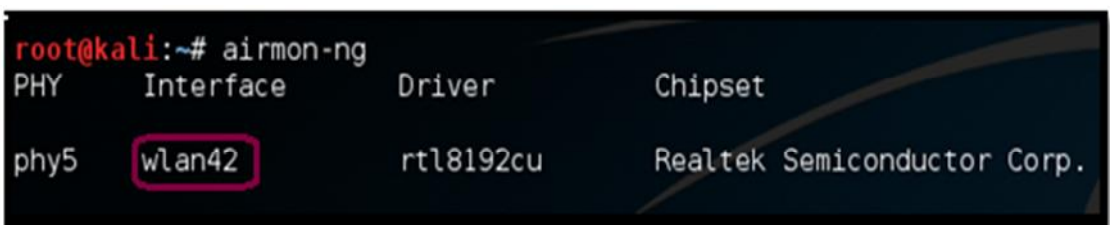

Figure 8 : Résultats de la commande airmon-ng.

.

- On bascule notre carte Wi-Fi dans le mode moniteur.
  Usage: Airmon-ngstart wlan42.

Figure 9 : Basculement dans le mode moniteur.

Une fois la commande exécuté, nous avons une nouvelle interface (wlan42mon) que nous allons utiliser pour tester les attaques.

- Une fois cela est fait en lançant airodump-ngsur notre interface wlan42mon, on remarque bien que l'ESSID n'est pas reconnu.
  Usage : Airodump-ng wlan42mon.

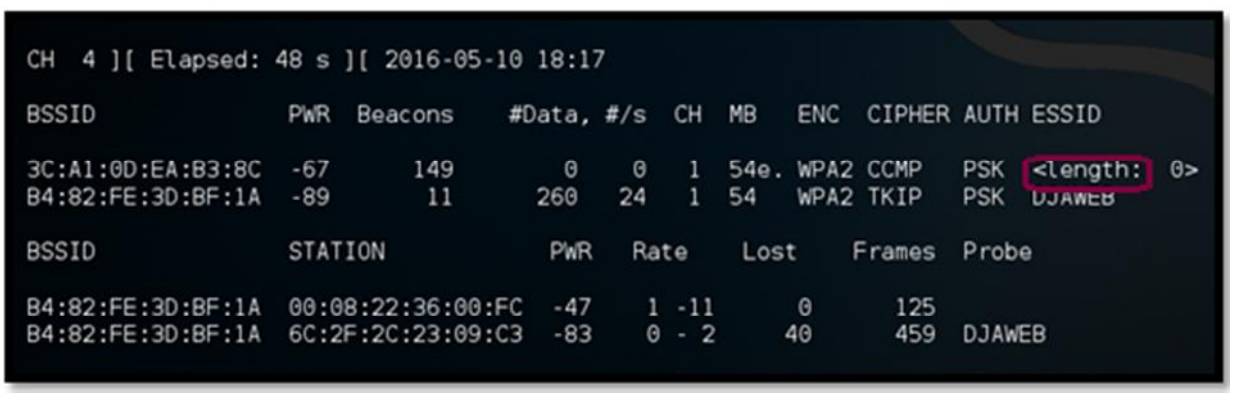

Figure 10 : Le SSID caché.

3.2.1.Les attaques sur l'ESSID caché

a. Attaque passive

On lance une capture avec Wireshark et on analyse le trafic jusqu'à ce qu'un client se connecte, lorsqu'il se connectera il échangera des paquets de probes request / response avec le

point d'accès ce qui nous permettra d'obtenir notre fameux SSID.Mais comme nous ne sommes pas dotés d'une grande patience, nous allons utiliser la deuxième attaque dite active.

b. Attaque active

Cette attaque consiste à déconnecter tous les clients pour qu'ils se reconnectent, de manière générale la reconnexion à un réseau se fait automatiquement.

D'abord on relance la commande airodump-ng et on voit toujours notre réseau avec l'ESSID caché « length ».On va lancer en parallèle Wireshark pour suivre ce qui va se passer.

Usage: Aireplay-ng --deauth 10 –a 3C:A1:0D: EA: B3:8C wlan42mon.

```
root@kali:~# aireplay-ng --deauth 10 -a 3C:A1:0D:EA:B3:8C wlan42mon
18:22:57  Waiting for beacon frame (BSSID: 3C:A1:0D:EA:B3:8C) on channel 1
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
18:22:57  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:22:58  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:22:58  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:22:59  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:22:59  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:23:00  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:23:00  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:23:01  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:23:03  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
18:23:03  Sending DeAuth to broadcast -- BSSID: [3C:A1:0D:EA:B3:8C]
```

Figure11: Désauthentification des clients du point d'accès.

Si nous regardons les beacons frames envoyées par ce réseau caché à l'aide de Wireshark on remarque aussi l'absence du SSID dans ces paquets.

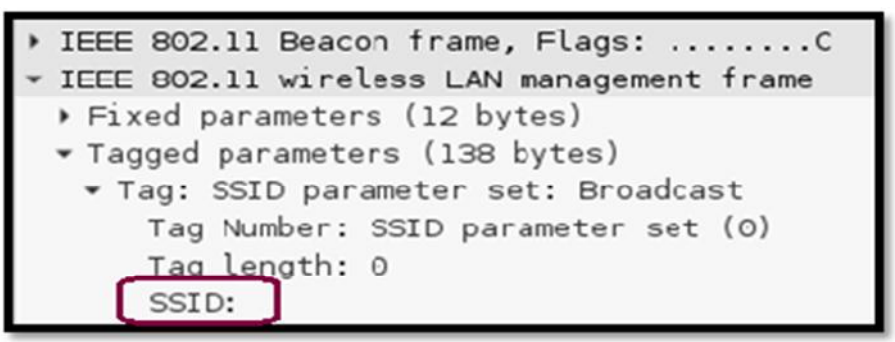

Figure12 : Absence de SSID dans les beacons frames.

En, patientant quelques secondes,on remarque sous Wireshark l'échange de paquet de type « **probe** », c'est normal puisque les clients déconnectés du réseau essaient de se reconnecter automatiquement et le SSID apparait.

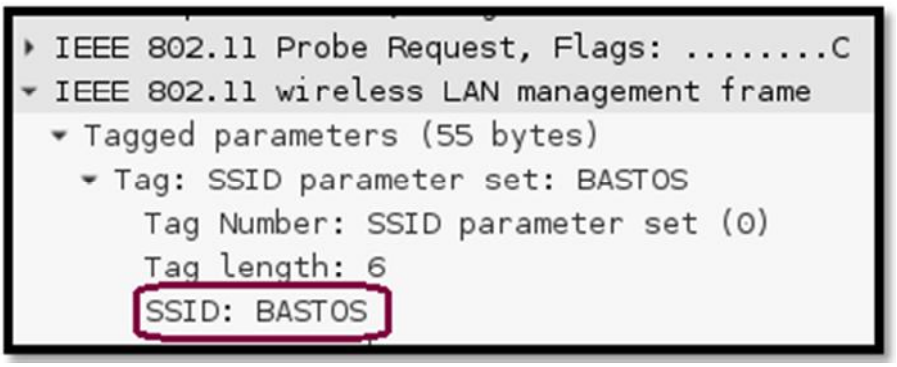

Figure 13 : Apparition de SSID dans les probes request.

Parallèlement sous la fenêtre airodump-ng au moment où le client se reconnecte, le SSID apparait à la place du « length **».**

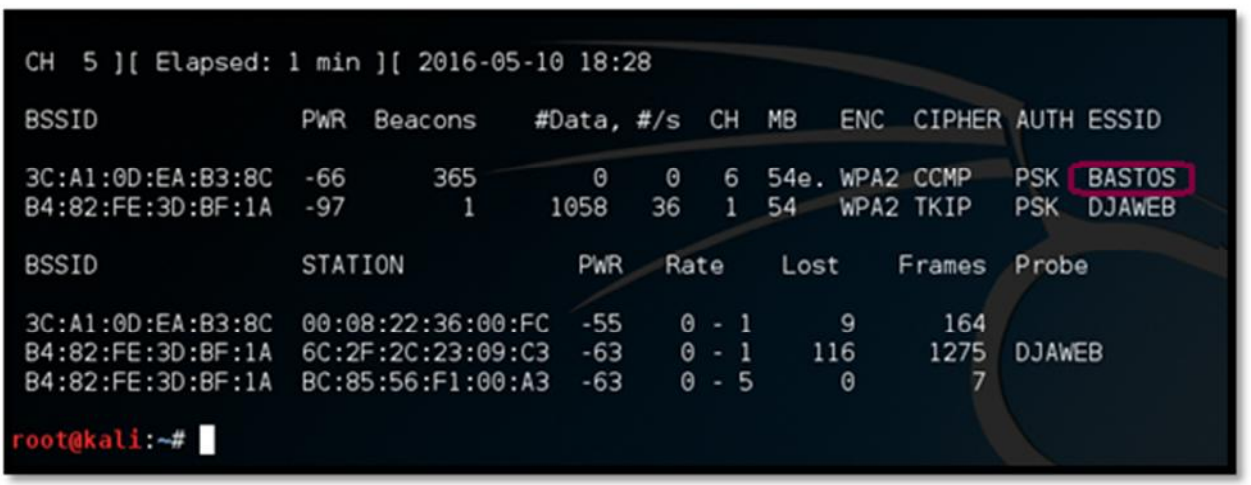

Figure 14: Démasquer le SSID caché.

3.3.Le filtrage MAC

Pour sécuriser un réseau wlan par le filtrage d'adresse MAC il faut définir au sein de la borne Wi-Fi une liste d'adresses MAC autorisées dite « liste blanche » pour une liste contenant des adresses MAC légitimes, au contraire une « liste noire » contiendra des adresses qui seront considérées comme illégitimes et donc à interdire.

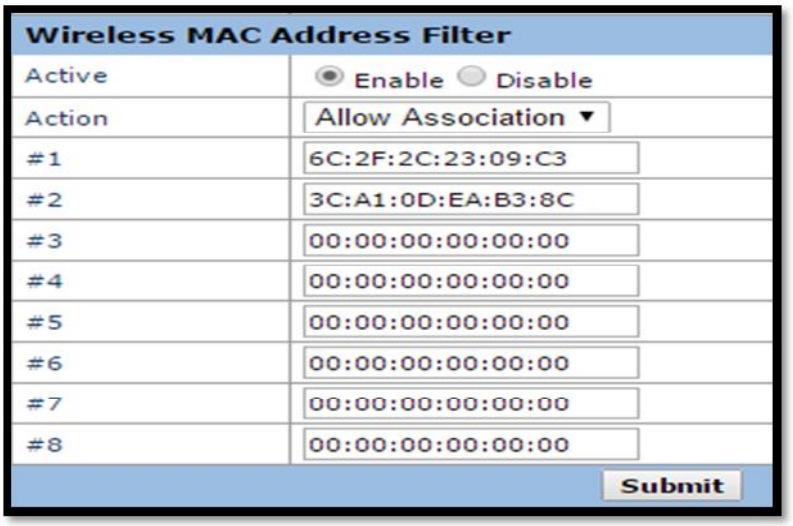

Figure 15: Activation de filtrage d'adresse MAC.

3.3.1.Outre passe le filtrage MAC

Le pirate souhaite se connecter à un réseau Wi-Fi protégé par un filtrage MAC, il commence par écouter les échanges d'informations qui circulent sur le réseau ciblé, une fois une machine autorisée dialogue avec le point d'accès, le pirate capture son adresse MAC, et l'attribue à sa propre machine, en quelques minutes la machine du pirate devient un appareil accepté sur le réseau Wi-Fi visé.

Notre travail sera comme suit :

- On bascule notre carte Wi-Fi dans le mode moniteur.
  Usage: Airmon-ngstart wlan46.
- Scanner les réseaux sans fil pour voir la liste des points d'accès et les stations disponibles.
  Usage: Airodump-ng wlan46mon.

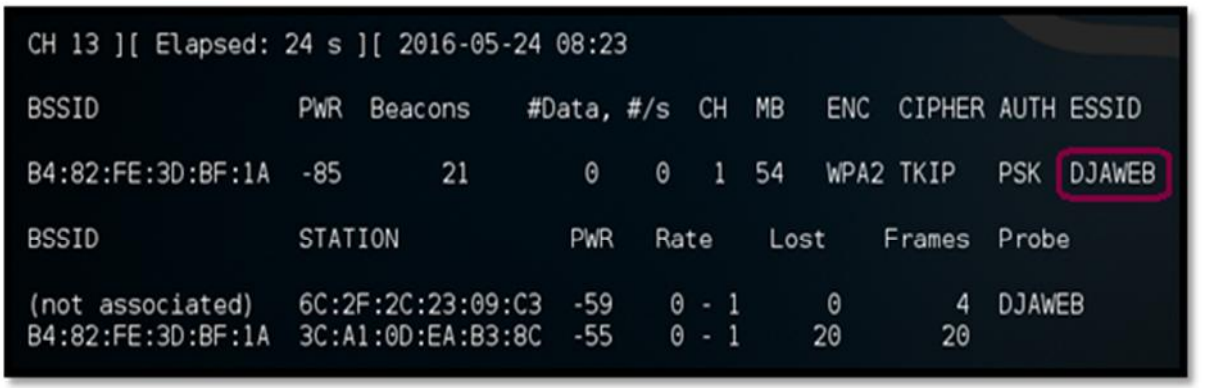

Figure 16: Résultats de la commande airodump-ng wlan46mon.

- Désactiver le mode moniteur pour pouvoir changé l'adresse MAC.

  Usage : Airmon-ng stop wlan46mon.

Figure 17: Désactivation du mode moniteur**.**

- Désactiver la carte Wi-Fi.

  Usage :ifconfig wlan46 down.

- Ensuite on va changer l'adresse MAC de la carte réseau Wi-Fi.

  Usage :macchanger–m 3C :A1 :0D : EA : B3 :8C wlan46.

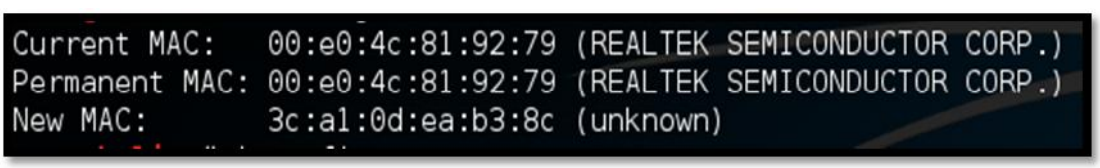

Figure 18: Changement d'adresse MAC.

- Réactiver la carte Wi-Fi

  Usage : ifconfig wlan46 up.

- Réactiver le mode moniteur

  Usage:airmon-ngstart wlan46.

- Dans cette dernière étape nous vérifions si l'adresse MAC est effectivement changée.

  Usage :ifconfig.

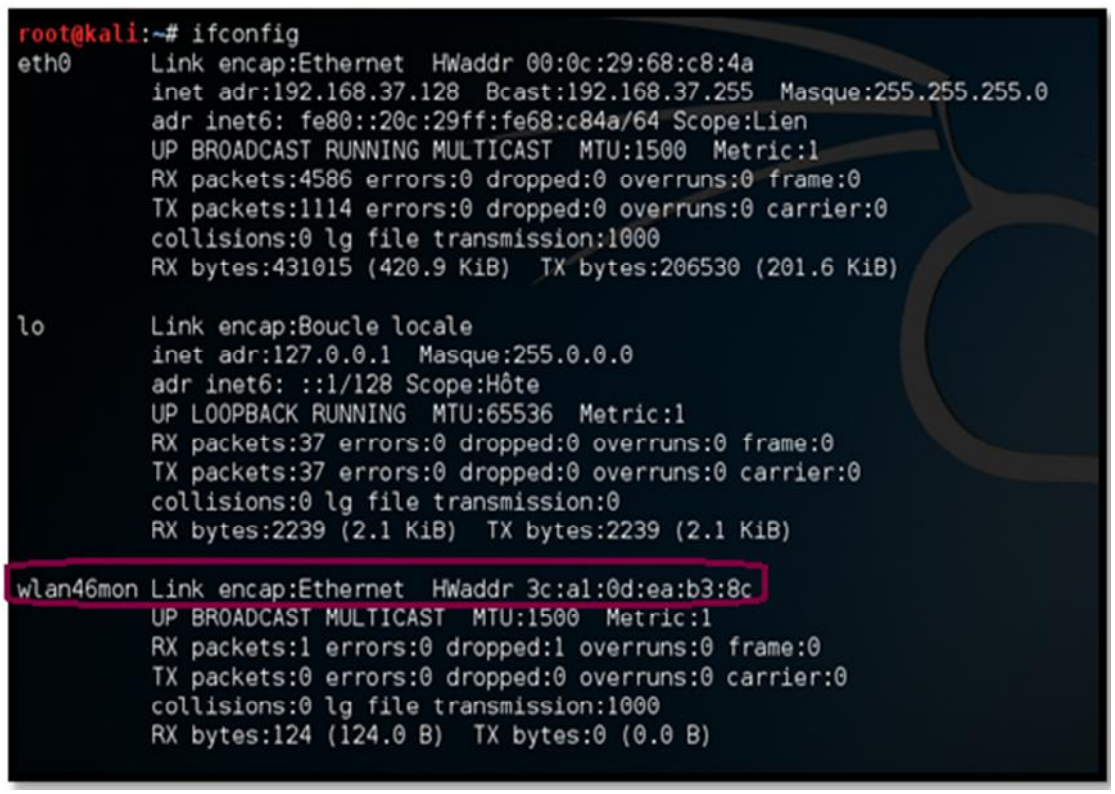

Figure 19 : Vérification si l'adresse MAC est changée.

- Maintenant si on envoie une requête fausse avec une adresse MAC source autorisée par le point d'accès, notre requête sera acceptée par le point d'accès.

  Usage: aireplay-ng –fakeauth 6 –e DJAWEB –h 3C:A1:0D: EA: B3:8C wlan46mon.

Figure 20 : Requête fausse d'authentification vers le point d'accès.

3.4. Le protocole WEP

Le protocole WEP constitue le premier mécanisme de chiffrement implanté dans lesréseaux sans fil, dans le but de sécuriser les échanges de données. Le protocole WEP se base sur uneclé secrète partagée entre les différentes parties communicantes, pour protéger le corps de la trame de donnée transmise.

a. Le chiffrement WEP

Le principe du WEP consiste à définir dans un premier temps un mot de passe qu'on appelle clé secrète de 64 bits ou 128 bits, cette clé secrète doit être déclarée au niveau du point d'accès et au niveau des clients.Le WEP utilise l'algorithmeRC4, un algorithme de chiffrement, ainsi à partir d'une clé et d'un vecteur d'initialisation de 24 bits transporté en clair dans chaque paquet, le WEP génère une séquence aléatoire d'une longueur égale à la longueur de la trame, chaque transmission de donnée est ainsi chiffrée en utilisant un OU Exclusif entre la séquence aléatoire et la donnée à transmettre.

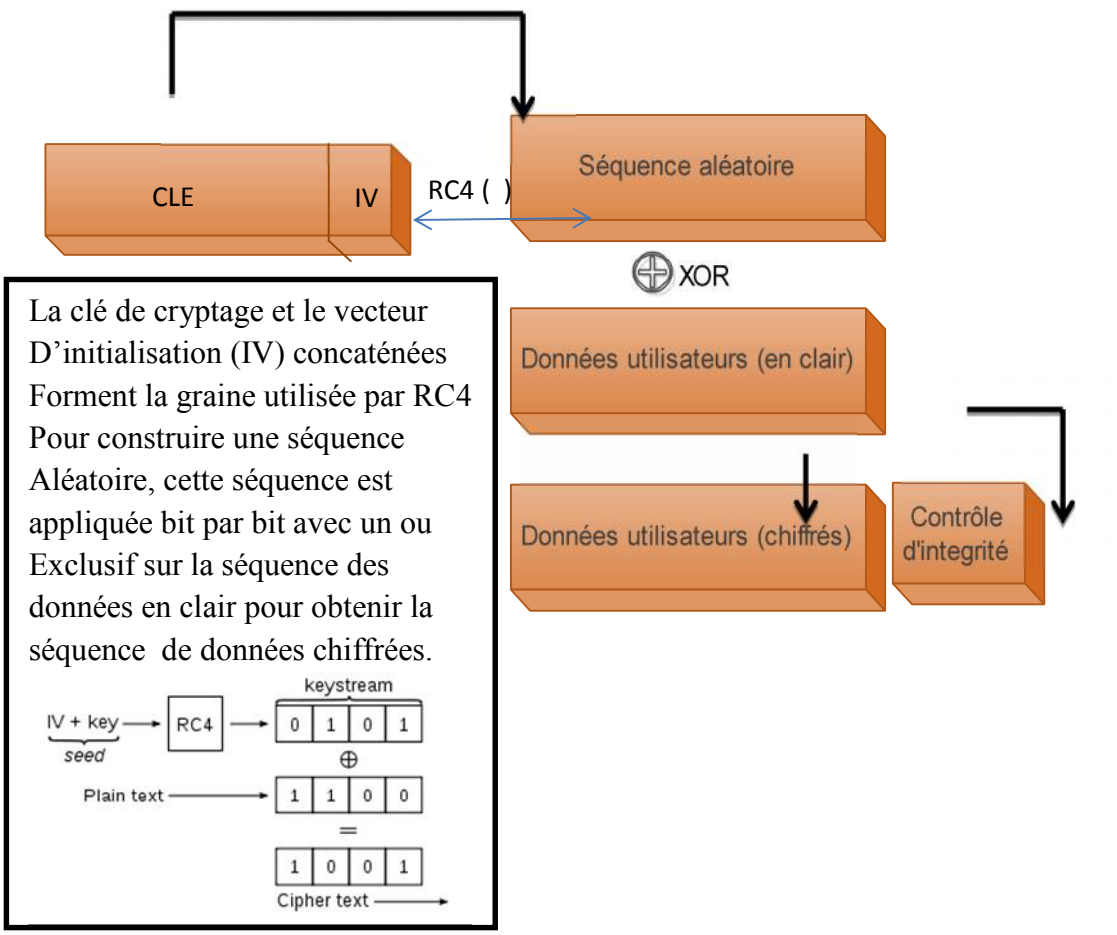

Figure 21 : Le chiffrement WEP.

b. Le déchiffrement WEP

Dans le déchiffrement dès qu'on connait la séquence aléatoireon peut déchiffrer un message chiffré, car le XOR est une fonction réversible:

- [séquence aléatoire] XOR [message en clair] = message chiffré
- [séquence aléatoire] XOR [message chiffré] = message en clair

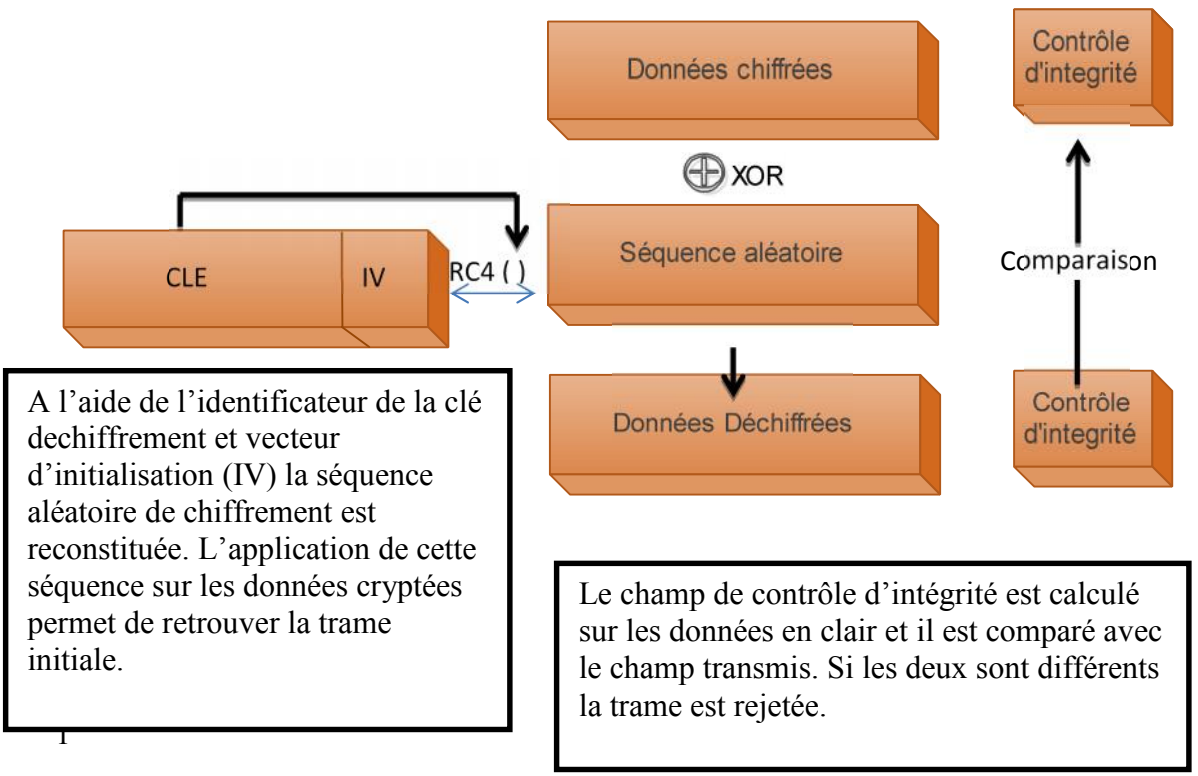

Figure 22 : Le déchiffrement WEP.

3.4.1. Lestypes d'attaques WEP

Parmi les différents types d'attaques WEP on a choisi deux attaques, une attaque du côté serveur dite ARP Replay et une autre du côté client dite caffe- latte.

3.4.1.1.L'attaque ARP Replay

L'ARP Replay est une attaque classique contre le WEP,elle permet d'obtenir une clé WEP à partir d'un point d'accès. Lehacker intercepte un paquet ARP transitant entre le point d'accès et la station victime, puis il le retransmet vers le point d'accès à l'intermédiaire de la victime, ceci, à son tour, provoque le point de répéter le paquet ARP avec un nouveau IV,le hacker réémet le même paquet ARP plus et plus, cependant chaque paquet ARP répété par le point

d'accès à un nouveau IV, ce sont tous ces nouveaux IVs qui nous permettent de déterminer la clé WEP.

Notre travail sera comme suit :

- On dispose d'un point d'accès sécurisé par la clé WEP(1301199200).

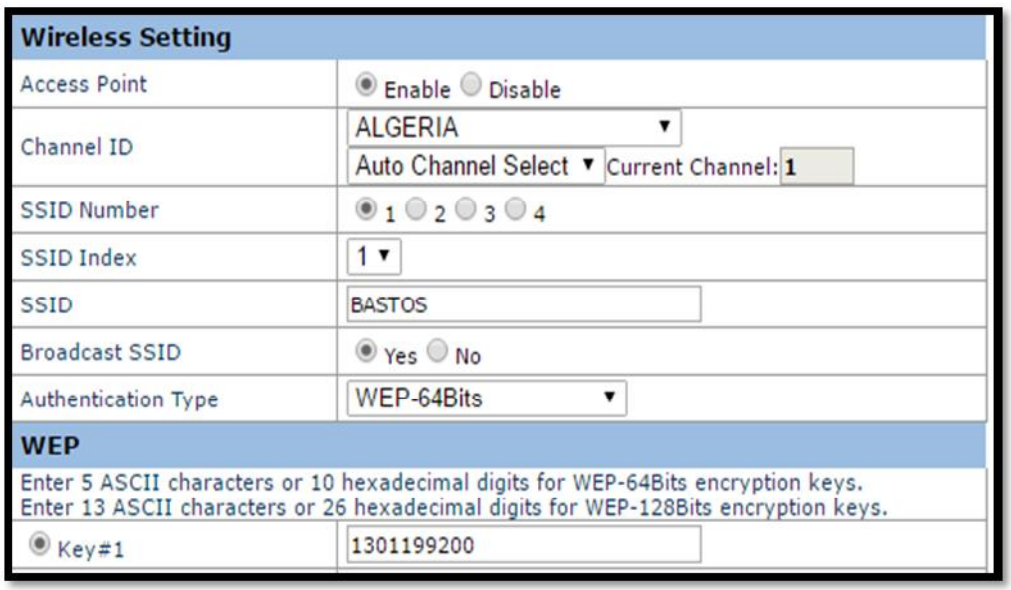

Figure 23 : Activation du WEP.

- On bascule notre carte Wi-Fi dans le mode moniteur.

  Usage: airmon-ngstart wlan15.

- Scanner les réseaux sans fil pour voir la liste des points d'accès, et les stations disponibles.

  Usage : Airodump-ng wlan15.

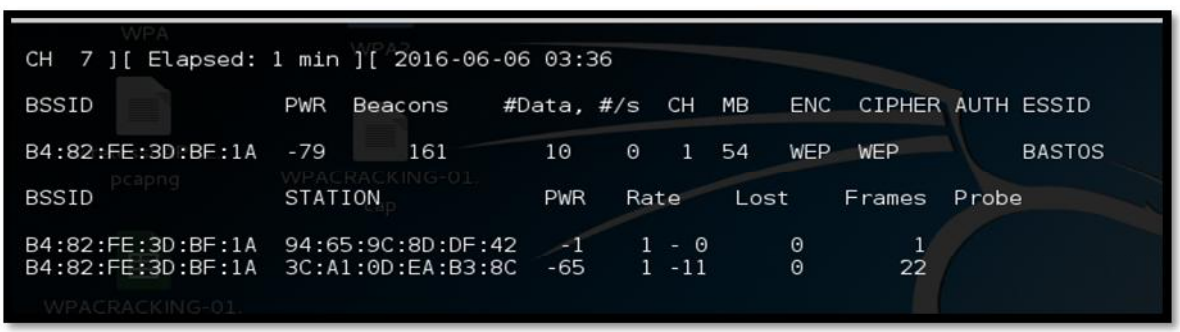

Figure 24 : Résultat de la commande airodump-ng.

- Dans une nouvelle console, on va procéder àl'injection de paquets ARP sur la machineassocié au point d'accèsciblé, pour capturer un le nombre nécessaire des IVs afin de réussir l'attaque.

  Usage :aireplay-ng --arpreplay -e BASTOS -b B4 :82 :FE :3D :BF :1A –h 94 :65 :9C :8D :DF :42 wlan15.

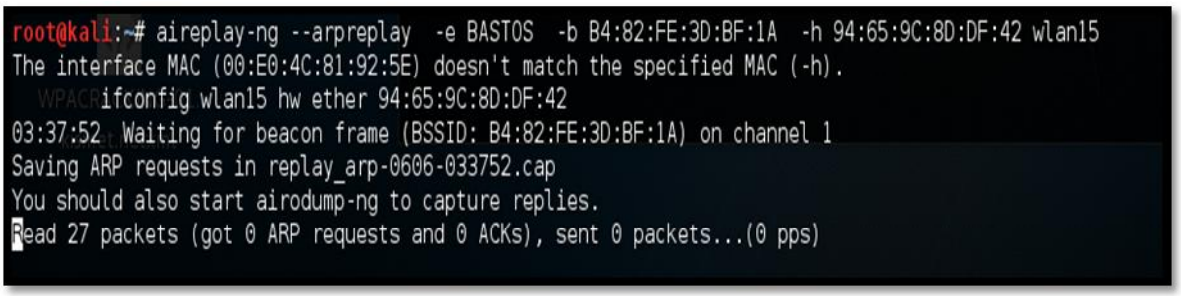

Figure 25: Résultats de la commande aireplay-ng.

- On remarque maintenant plus les clients essayent de s'authentifier plus les paquets ARP augmentent.

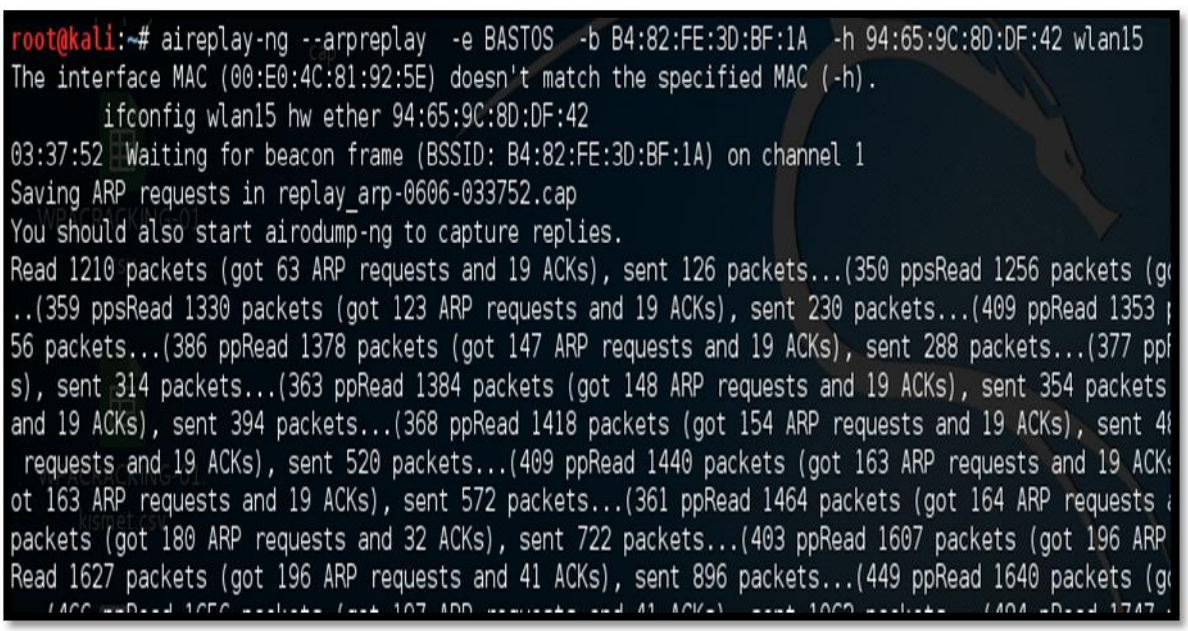

Figure 26 : Augmentation des paquets ARP.

- On remarque aussi l'augmentation du Data (données)comme nous montre la figure suivante.

Figure 27 : Augmentation du Data.

- Après un nombre suffisant des IVs, On lance l'outil aircrack-ng pour casser la clé WEP en définissant le fichier de capture.

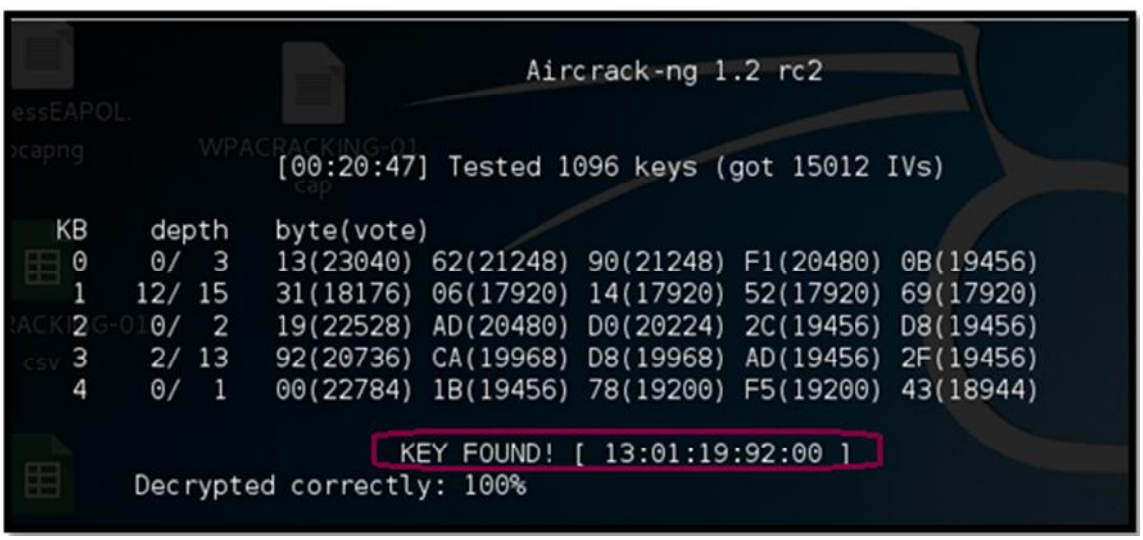

Figure 28:cracker la clé WEP.

D'après le résultat (Key found), notre manipulation a été bien fonctionnée, et le mot de passe a été cracké.

3.4.1.2. L'Attaque caffe latte

D'où le nom d'attaque vient? Le concept est qu'une clé WEP peut être obtenue à partir d'un client innocent dans une cafétéria, dans le temps qu'il faut pour boire son café.

Caffe Latte est une nouvelle attaque contre le WEP, qui permet d'obtenir une clé WEP à partir d'un système client, où l'attaquant et le client sont loin de la zone du réseau ciblé. En bref, cela se fait en capturant un paquet ARP du client, en le manipulant et puis en le renvoyant à ce dernier, ceci à son tour génère des paquets, la suite, aircrack-ng peut être utilisé pour déterminer la clé WEP.

Notre travail sera comme suit :

- On dispose d'un point d'accès sécurisé par la clé WEP(1234567890).

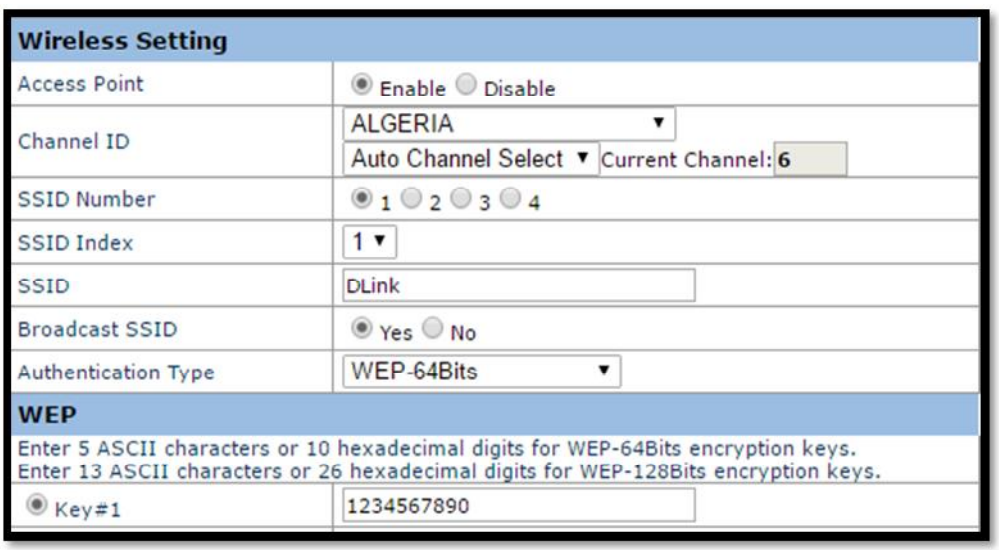

Figure 29: Activation du WEP.

- Nous devons activer notre carte Wi-Fi en mode moniteur.

  Usage: Airmon-ngstart wlan0.
- Les clients Wi-Fi sondent activement pour tous les réseaux où ils ont été associés dans le passé, lorsqu'un point d'accès est trouvé avec un nom du réseau connu, les clients s'associent automatiquement, pour cela, nous devons créer un faux point d'accès avec un SSID qui apparait dans la liste d'un client qui envoie des probes (SSID), pour associer au réseau.
- **Usage:**airbase-ng - -essidDLink –W 1 –L wlan0.

Figure 30: Création d'un faux point d'accès nommé « DLink ».

- Ensuite scanner les réseaux sans fil pour voir le faux point d'accès créé, et la station connectée

  Usage :airodump-ng wlan0.

Figure 31: Résultats de la commande airodump-ng wlan0.

- Dans une nouvelle console, on va procéder à l'injection de paquets ARP pour la capture des IVs nécessaires à la réussite de l'attaque.

Usage: aireplay-ng –caffe-latte –D –b AA: 98: AB: B7:C3: EF –h C0: EE: FB: 26:F5:75

Wlan0.

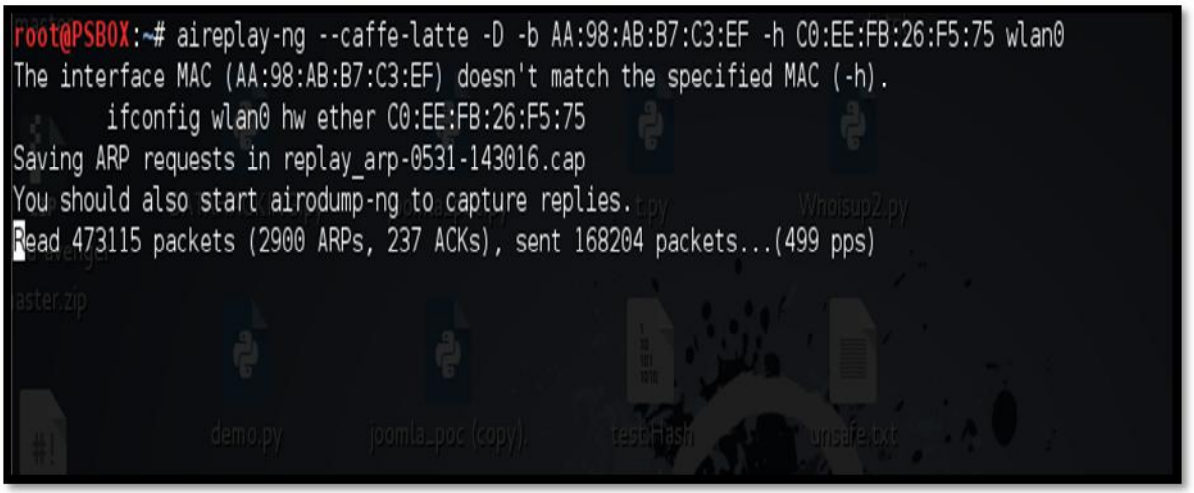

Figure 32: Résultats de la commande aireplay-ng.

On remarque que les données (Data) augmentent 184 data par seconde.

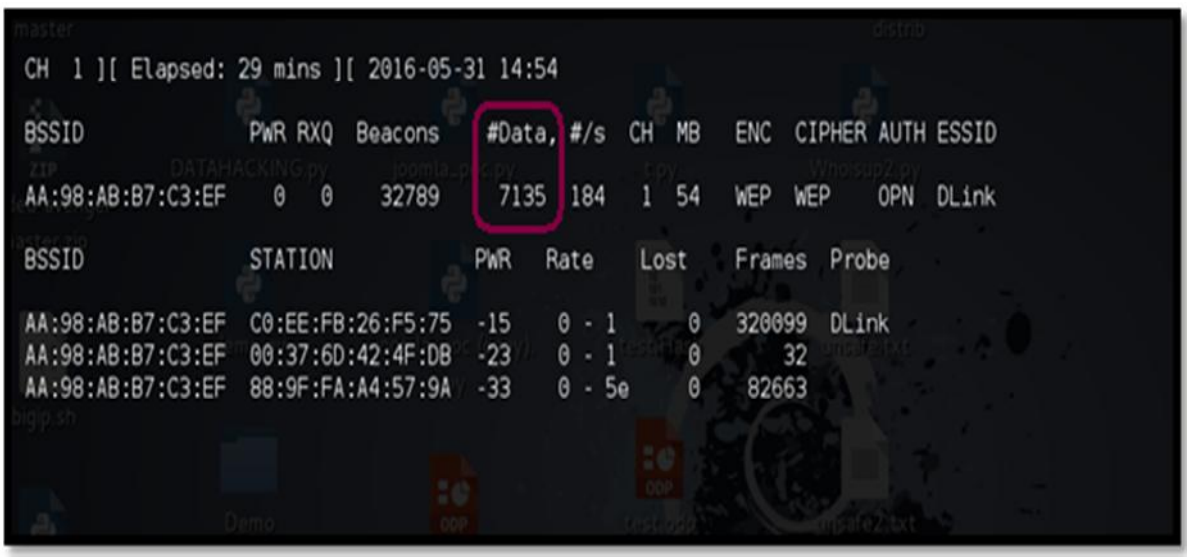

Figure 33 : Augmentation du Data.

- Apres un nombre suffisant des paquets IVs, on lance l'outil aircrack-ng pour casser la clé WEP.

  Usage: aircrack-ng wepcracking-01.cap.

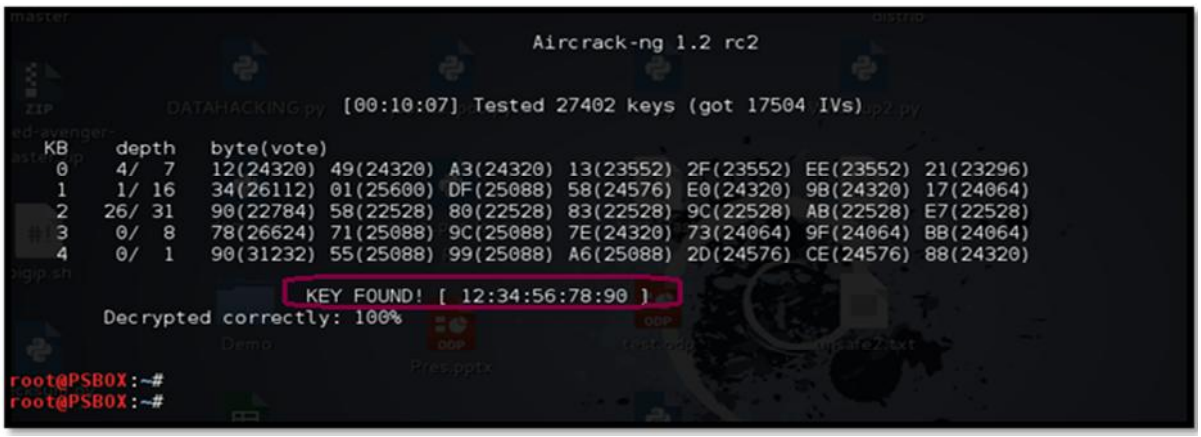

Figure 34: Cracker la clé WEP.

D'après le résultat (Key found), notre manipulation a été bien fonctionnée, et le mot de passe a été cracké.

3.5.802.11i (WPA /WPA2)

- **WPA**

WPA est une version allégée de la norme 802.11i, il peut être considérer comme une deuxième génération pour la sécurité des réseaux sans filaprès le WEP, dans ce protocole les données sont chiffrées de la mêmefaçon que pour le WEP (chiffrement par RC4), mais le WPA utilise le protocole TKIP, qui permet de générer des clés différentes, et même de les changer plusieurs fois par seconde,de plus, les clés sont plus grandes, le vecteur d'initialisation aussi.

- **WPA2**

Le 802.11i a été ratifié afin de fournir une solution de sécurisation forte pour les réseaux Wi-Fi, il s'appuie sur l'algorithme de chiffrement TKIP, comme le WPA, mais il supporte également le chiffrement AES beaucoup plus robuste que TKIP.

La norme IEEE802.11i définit deux modes de fonctionnement :

- **WPA Personnel**

Le mode WPA personnel permet de mettre en œuvre une infrastructure sécurisée sans mettre en œuvre de serveur d'authentification. Le WPA personnel

repose sur l'utilisation d'une clé partagé appelée PSK, renseignée dans le point d'accès ainsi que dans les postes clients.

- **WPA Entreprise**

  Le mode entreprise impose l'utilisation d'une infrastructure d'authentification 802.1x, basée sur l'utilisation d'un serveur d'authentification, généralement un serveur RADIUS et un contrôleur réseau (point d'accès).

3.5.1.Attaquepar force brute

Malgré que WPA2 possède un algorithme de chiffrement robuste(AES), mais ça ne veut pas dire qu'il est fiable à 100%, car une attaque par force brute (attaque par dictionnaire) peut facilement cracker la clé.

Notre travail sera comme suit :

- On dispose d'un point d'accès sécurisé par la clé WPA2 (tiziouzou123).

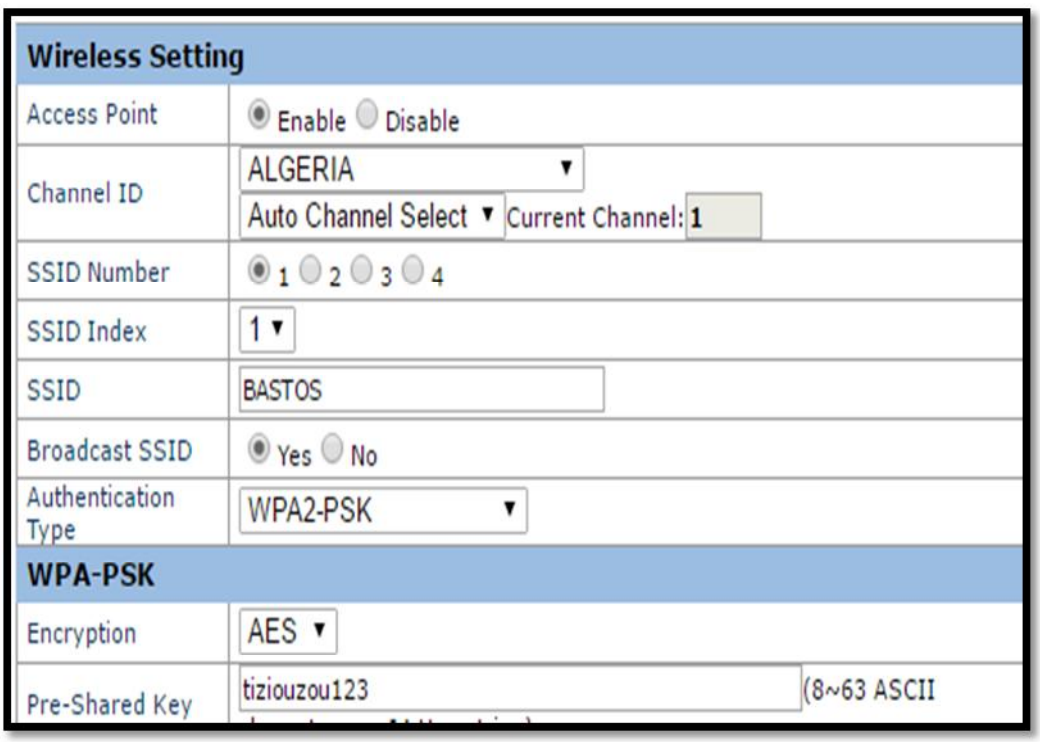

Figure 35 : Activation de WPA2.

- Nous devons activer notre carte Wi-Fi en mode moniteur.

  Usage: Airmon-ngstart wlan14.

- Scanner les réseaux sans fil pour voir la liste des points d'accès, et les stations disponibles.

  Usage :airodump-ng wlan14.

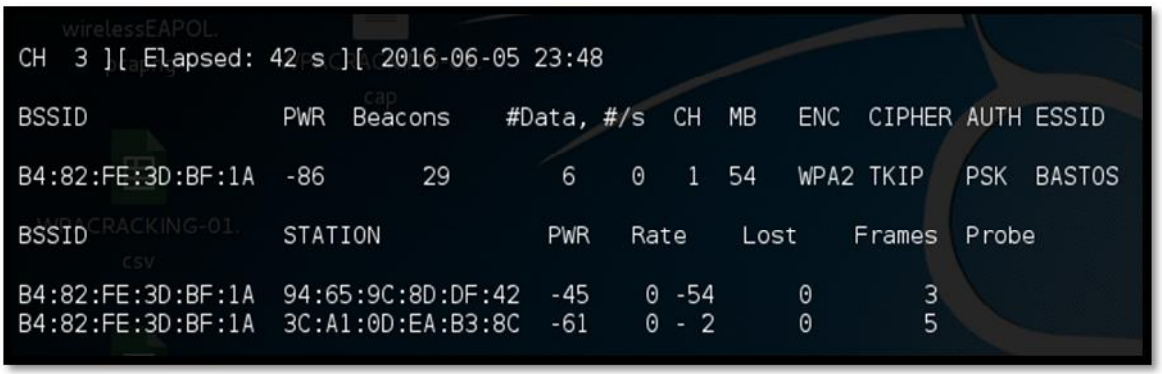

Figure 36 : Résultats de la commande airodump-ng.

- Ici on se focalise donc sur le réseau BASTOS qui se trouve dans le canal 1, et on enregistre les paquets dans un fichier de capture wpacrack.

  Usage: airodump-ng --bssid B4 :82 :FE :3D :BF :1A -c 1 –w wpacrack wlan14.

Figure 37 :Résultats de la commande airodump-ng.

Pour réussir un crack WPA il est primordial qu'une station soit connectée, en effet le 4 wayhandshake nécessaire au crack, nepeut être capturé que si une station est connectée au point d'accès.

En patientant quelques minutes,en voyant l'apparition du WPA handshake en haut à droite de la fenêtre, ce qui indique la réussite de teste d'attaque.

Figure 38 : Apparition de WPA handshake.

On va lancer en parallèle wireshark,en voyant les quatre EAPOL-Key échangé entre le client et le point d'accès comme illustré ci-dessous.

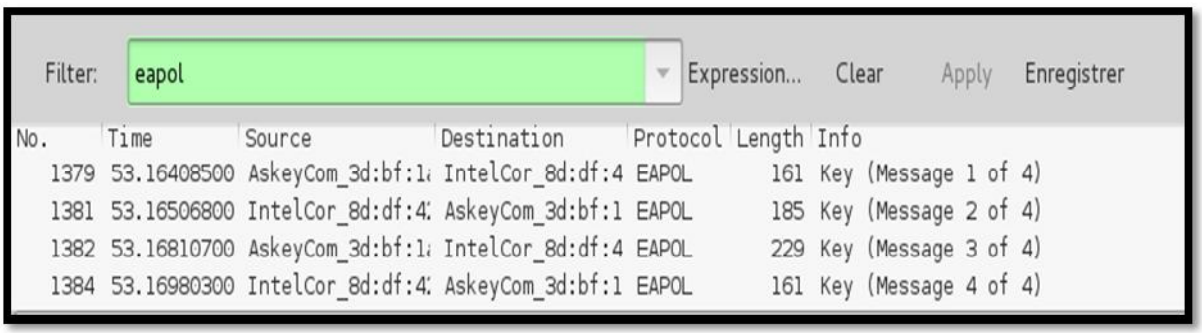

Figure 39: Le 4-Way Handshake sous wireshark.

- Dans une nouvelle console on fait entrer le mot de passe « tiziouzou123 » au dictionnaire qui se trouve dans kalilinux par la commande (nano /usr/share/wordlists/dirb/common.txt), puis on clique sur CTRL x SHIFT y.

- L'étape finale consiste à lancer le test d'attaque par dictionnaire, en utilisant aircrack-ng, qui va tester tous les mots de passes contenus dans le dictionnaire.
  Usage:aircrack-ng wpacrack-01.cap –w /usr/share /wordlists/dirb/common.txt.

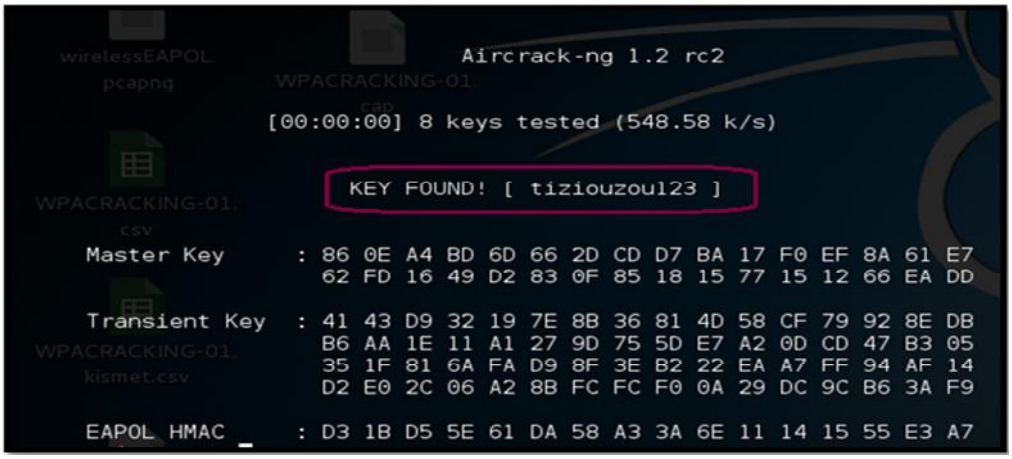

Figure 40 : Cracker la clé WPA2.

D'après le résultat (Key found), notre manipulation a été bien fonctionnée, et le mot de passe a été cracké.

3.5.2. Vulnérabilité WPS

Le WPS est une technologie lancée par la Wi-Fi Alliance, elle utilise un code PIN comme un secret partagé pour authentifier un point d'accès et un client sans utiliser le mot de passe, mais malheureusementcette fonction ne déroge pas à des failles de sécurité, cesfailles permettent à un attaquant de récupérer le code PIN WPS en quelques heures avec une attaque par force brute.

3.6.2.1.Attaque par force brute

Notre travail sera comme suit :

- On dispose d'un point d'accès sécurisé par la clé WPA2vidarosa.

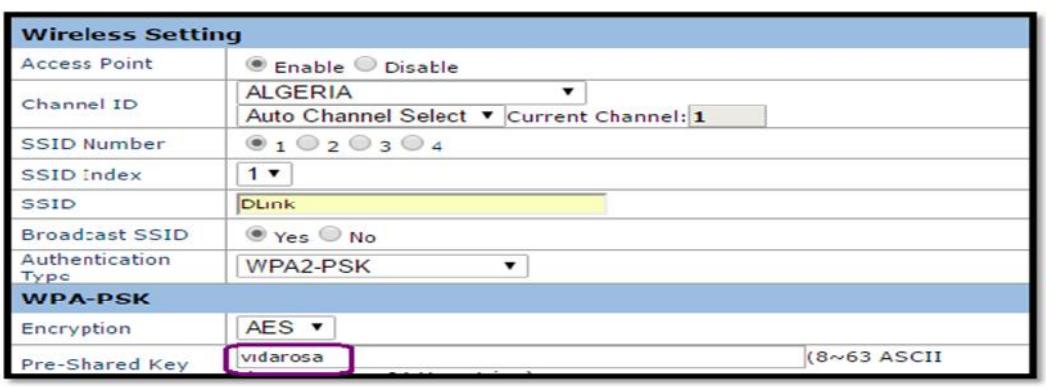

Figure 41 : Activation du WPA2.

- Nous devons activer WPS dans notre point d'accès en entrant le code PIN 21464065.

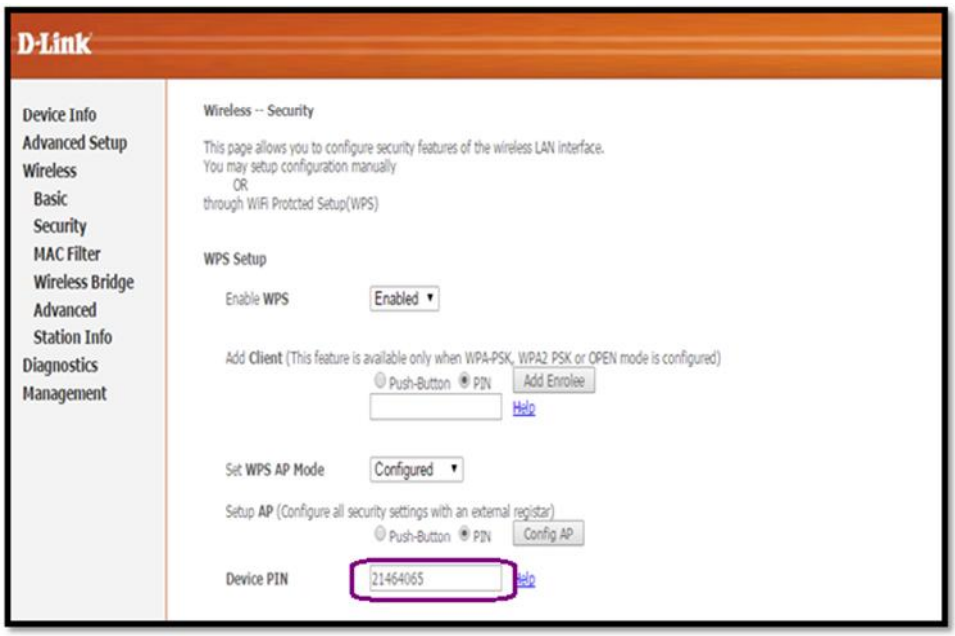

Figure 42 : Activation du WPS

- Nous devons activer notre carte Wi-Fi en mode moniteur.
  Usage: Airmon-ngstart wlan0.
- Scanner les réseaux sans fil pour voir la liste des points d'accès, et les stations disponibles. **Usage :**airodump-ng wlan0.

Figure 43 : Résultats de la commande airodump-ng.

- On lance l'attaque par force brute.

  Usage :reaver -v –b 84 :C9 :B2 :EF :BB:C1 –i wlan0 –d 0

Figure 44 : Résultats de la commande reaver.

- Dans cette étape on va faire entrer le code PIN 21644065 au lieu d'attendre des heures,après on voit l'apparition de la clé vidarosa.

  Usage :reaver -v –b 84 :C9 :B2 :EF :BB:C1 –i wlan0 –d 0 –p 21464065.

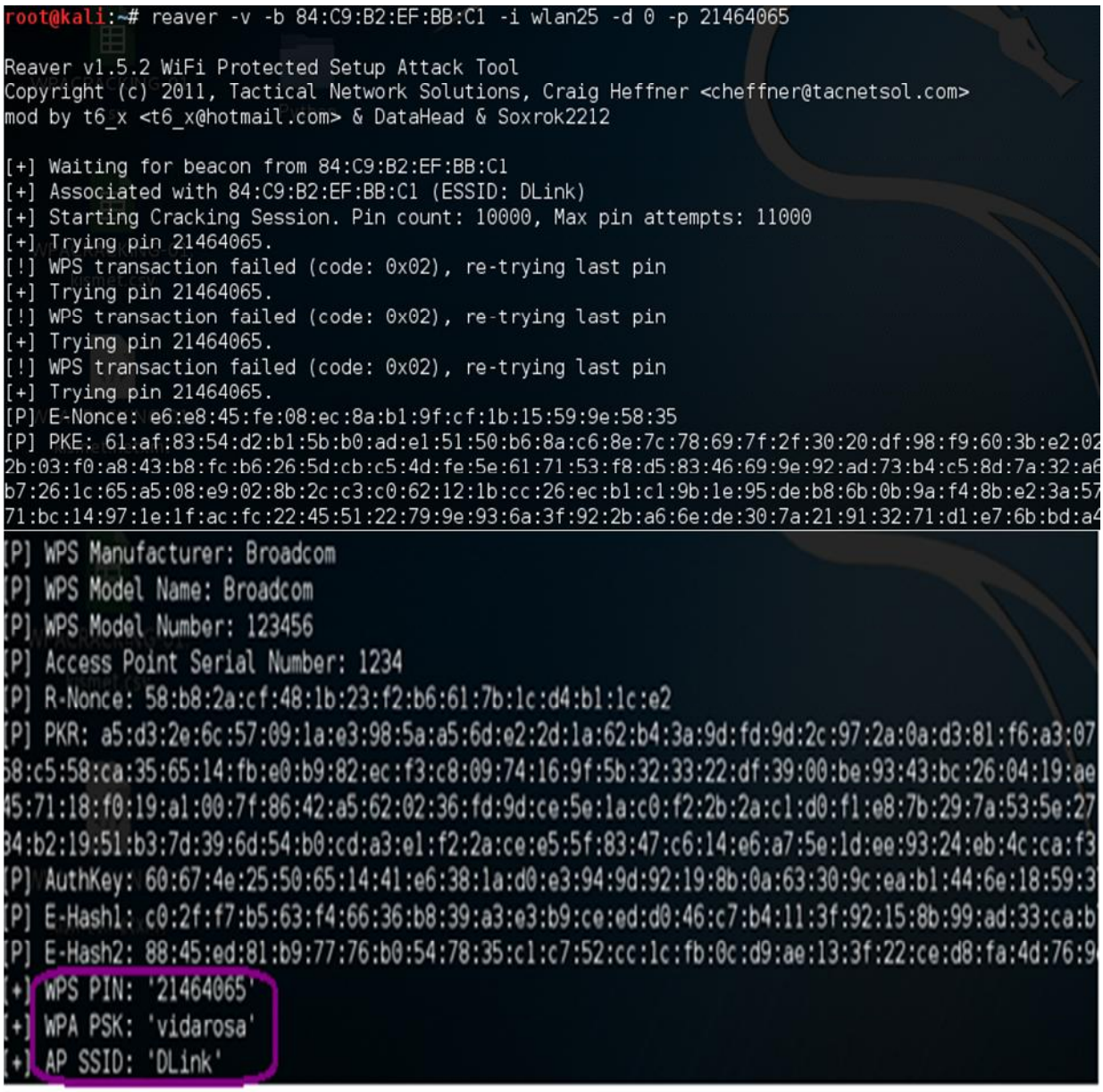

Figure 45: Apparition de la clé « vidarosa ».

4.Discussion

Bien que les mécanismes de sécurité Wi-Fiont évolués au fil du temps, les attaques qu'on a appliquées dans ce chapitre, nécessitent des questions qui ont besoin des réponses. C'est pourquoi, il nous a semblé nécessaire au moins de détecter ces attaques afin d'avoir une visibilité sur notre réseau Wi-Fi de point de vue de sécurité, pour minimiser les risques.

Dans ce cadre, dans le troisièmechapitre, nous allons développer un système de détection d'intrusion sans fil, en donnant quelques exemples de signatures les plus utilisées.

Chapitre III :

Développement d'un Wireless Intrusion Détection System - WaveShark

1. Préambule

Aujourd'hui, les réseaux sans fil deviennent de plus en plus répondus, c'est pourquoi des attaques sont développées pour exploiter ces réseaux. En réponse, un système de détection d'intrusion sans fil (WIDS) est développé afin d'offrir une surveillance contre ces attaques.

Dans ce chapitre, nous allons développer une application qu'on a appelée ***WaveShark***, qui nous permet de détecter quelques attaques applicables sur le réseau Wi-Fi, en utilisant lelangage de programmation Python.

2. un Framework

Un Framework est un cadre de travailqui va nous permettre d'être plus efficace dans la conception des codes. Un Framework est l'ensemble d'outils formant le squelletteduprogramme.L'objectif d'un Framework est généralement de simplifier le travail des développeurs informatiques, en leur offrant une architecture prête à l'emploi, et qui leur permette de ne pas repartir de zéro à chaque nouveau projet.

3. Le choix du langage

3.1.La différence entre un langage haut niveau et bas niveau

Le terme langage de haut niveau n'implique pas que ce type de langage soit supérieur à un langage de bas niveau.Un langage de programmationest dit de bas niveau lorsque le codage de celui-ci se rapproche du langage machine dit binaire, par contre le langage haut niveau permet d'écrire des programmes en utilisant des mots usuels des langues naturelles (très souvent l'anglais).

3.2. Le langage Python

Le langage Python est développé depuis 1989 par Guido van Rossum, c'est un langage de programmation haut niveau, relativement simple à apprendre, avec lequel on peut rapidement réaliser de simples programmes.

Python a la particularité d'être un langage interprété, orienté objet, multiplateforme, gratuitet aussi complet grâce aux nombreuses bibliothèques spécialisées, appelées modules. Python contient un environnement de développement intégré appelé IDLE qui propose un certain nombre d'outils :

- un éditeur de texte (pour écrire le programme)
- un interpréteur (pour exécuter le programme).

3.3. Les bases du langage python

a. Le calcule avec python

Dans ce cas on utilise l'interpréteur comme une simple calculatrice comme c'est illustrer ci-dessous.

```
>>> 7-2
5
>>> 7+3
10
>>>
```

b. Les variables

En Python pour créer une variable, il suffit de l'initialiser une première fois, c'est à dire d'écrire nom variable=valeur de la variable.

```
>>> x=1
>>> y=1+1
>>> z=x+y
```

En Python, le symbole égalest également nommé l'opérateur d'affectation.

- **Les opérateurs**

Lesopérateurs de base sont les suivant :

opérateurs	signification
+	l'adition
-	la soustraction
*	la multiplication
/	la division
%	le modulo
**	la puisance

Tableau 2 : Les opérateurs de base.

- **La comparaison**

Python est capable d'effectuer toute une série de comparaison entre le contenu de différentes variables, telles que :

syntaxe python	signification
==	égal à
!=	différent de
>	supérieur à
>=	superieur ou égal à
<	inférieur à
<=	inférieur ou égal à

Tableau 3 : Les instructions de comparaison.

c. Les types de données

Python est un langage dont le typage est automatique, cela signifie que lorsque lavariable est affectée, l'interpréteur trouvera automatiquement son type.

- **Les nombres**

Dans les nombres on trouve plusieurs types : les entiers, les réels……etc.

```
>>> x=1
>>> x=0
>>> x=-33
>>> x=1.234
>>> x=1.0
```

- **Les chaines de caractères**

Une chaîne de caractères (string en anglais) est une suitede caractères, c'est–à–dire des caractères qui se suivent les uns derrière les autres.Les valeurs littéralesde chaînes de caractères s'écrivent de plusieurs manières en Python.Il est possible d'utiliser indifféremment des apostrophes, des guillemets, des triples apostrophes ou des triples guillemets.

```
>>> x='hello'
>>> y="world"
>>> z='''hello world'''
```

- **Les listes**

Une liste est un ensemble ordonné d'éléments entouré par des crochets, elle peut être utilisée comme un tableau dont l'indice de base est zéro. Le premier élément de toute liste non vide est toujours liste [0]. Un indice négatif permet d'accéder aux éléments à partir de la fin de la liste.

```
>>> liste=["a","b","mpilgrim","z","example"]
>>> liste
['a', 'b', 'mpilgrim', 'z', 'example']
>>> liste[0]
'a'
>>> liste[-1]
'example'
>>> liste[-3]
'mpilgrim'
```

- **Les dictionnaires**

Les dictionnaires sont des collections non ordonnées d'objets entourés par des accolades. Chaque objet est une paire clé-valeur. On accède aux valeursd'un dictionnaire par ses clés.

```
>>> dic={'prenom':'lilia','nom':'taibi'}
>>> print dic['nom']
taibi
```

- **Les tuples**

Un tuple est une liste non-mutable, unefois créé, il ne peut en aucune manière être modifié.

Un tuple est défini de la même manière qu'une liste, sauf que l'ensemble d'éléments est entouré de parenthèses plutôt que de crochets. Les éléments d'un tuple ont un ordre défini. Les indices de tuple débutent à zéro, le premier élément d'un tuple non vide est toujours t [0].

```
>>> t= ("a","b","mpilgrim","z","example")
>>> t
('a', 'b', 'mpilgrim', 'z', 'example')
>>> t[0]
'a'
>>> t[-1]
'example'
```

d.Les fonctions

Une fonction est une portion du code, c'est une sorte de sous-programme.

L'utilisation des fonctions évite des redondances dans le code, on obtient ainsi des programmes plus courts et plus lisibles

Dans le langage python on définit une fonction en utilisant le mot clé « **def** » suivie du nom de la fonction et de ses paramètres.

```
>>> def nom de la fonction(param etre1,param etre2,param etre3,...)
```

Comme on a aussi des fonctions prédéfinies comme :

- **La fonction type ()**

cette fonction nous permet de définir le type d'une variable.

```
>>> x=2
>>> type(x)
<type 'int'>
>>> y='3'
>>> type(y)
<type 'str'>
```

- **La fonction range ()**

  cette fonction nous permet de créer des listes d'entiers d'une manière simple et rapide.

```
>>> range(5)
[0, 1, 2, 3, 4]
>>> range(0,5)
[0, 1, 2, 3, 4]
```

- **La fonction print ()**

cette fonction nous permet d'afficher un résultat.

```
>>> message='bonjour'
>>> print message
bonjour
```

- **La fonction Len ()**

  Cette fonction nous donne la langueur de la chaine de caractère.

```
>>> a=list(range(7,10))
>>> a
[7, 8, 9]
>>> [7,8,9]
[7, 8, 9]
>>> len(a)
3
```

- **La fonction count ()**

cette fonction additionne le nombre de fois qu'un caractère apparaît dans une chaîne.

```
>>> animaux="girafe tigre"
>>> animaux.count("i")
2
>>> animaux.count("z")
0
```

e. Les modules

Un module est un programme Python qui contient un ensemble de fonctions,on l'appelleaussi bibliothèque ou librairieque l'on peut importer et utiliser dans le programme principal. Pour importer un module ou une fonction d'un module on utilise le mot clé importe.

```
>>> import random
>>> random.randint(0,1)
0
```

f. Les commentaires

Un commentaire est un texte qui ne sera pas pris en compte par l'interpréteur, pour se repérer il suffit de le mettre à côté d'une ligne dans le code source précédé d'un #

```
>>> x=1#on a affecter la valeur 1 à la variable x
```

h. Instruction de teste (if)

Cette notion est importante en programmation, ellesert à vérifier si une condition est vraie ou non.

```
>>> a=20
>>> if a==20:
        print "c'est vraie"


c'est vraie
>>>
```

i.Les boucles

Il existe deux types de boucles en Python,la plus couramment utilisé est **for…in ,**qui permet de parcourir les éléments d'une liste.

```
>>> for i in range(10):
        print i


0
1
2
3
4
5
6
7
8
9
```

La deuxième est très similaire à la boucle **while** en C.

```
>>> a=0
>>> while a<10:
        a+=1
        print a


1
2
3
4
5
6
7
8
9
10
>>>
```

4.Squelette du Framework

Nous avons exploité les avantagesdu l'orienté objet du Python pour créerune classe ***WaveShark***, qui représente le corps du notre programme. Cette classe contient cequ'on appelle un constructeur et plusieurs méthodes, le constructeur va nous permet d'initialiser plusieurs attributs qui vont être utilisés dans notre classe, ou peut-être dans une autre classe en terme d'héritage.

```
#!/usr/bin/env python
# Author :Lilia, Fatima,
# Purpose: Wireless Intrusion Detection
# Start Date: 02/06/2016
# Issue Date:17 /09/2016
# Program version: v1.0

import logging
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
from re import search
from threading import Thread
from tabulate import tabulate
from multiprocessing import Process
import signal
import os
ap_list = set()

class WIDS:
def __init__(self):
        attributes


defPacketHandler
code

defProbeReq(self, pkt):
code

def SSID(self, pkt,cap, vrai):
code

def HIDDEN(self,pkt,cap, vrai):
code

defDeauth(self, pkt, cap, vrai):
code

defAssociationReq(self, pkt):
            code

defAuthenticationReq(self, pkt):
            code
```

```
def __del__(self):
 code

defchannel_hopper(interface):
            code

defstop_channel_hop(signal, frame):
            code


def main():

channel_hop = Process(target = channel_hopper, args=("wlan0",))
channel_hop.start()
signal.signal(signal.SIGINT, stop_channel_hop)
        while True:
                try:
                        pkt = sniff(iface="wlan0", count=1)
                        if pkt[0].haslayer(Dot11):
                                obj = WIDS()
                                obj.PacketHandler(pkt[0])
                except KeyboardInterrupt:
                        exit()

if __name__== "__main__":

main()
```

Dans ce squelette on dispose d'une classe appelée WIDS,qui contient une fonction main. La fonction main sniffe un paquet, puis elle le passe à l'objet de la classe, cette classe fait appelle à la méthode PacketHandler.

La méthode PacketHandler va traiter le paquet et fait appelle à la méthode appropriée, cette dernière va faire des mises à jour aux attributs du constructeur pour que le déconstructeur affiche automatiquement les nouvelles mises à jour.

5.Extension du Framework

Dans notre application ***WaveShark***, nous avons implémentées quelques signatures des attaques Wi-Fi. ***WaveShark*** a été mis d'une façon extensible aux personnes qui veulent contribuer leurs signatures afin d'améliorer la qualité du programme.

Pour qu'une personne contribue sa signature, il n'a qu'à la créer et l'appeler depuis la méthode PacketHandler.

6.Sniffer des SSIDs

```
defPacketHandler(pkt):
          if pkt.haslayer(Dot11):
        if pkt.type == 0 and pkt.subtype == 8 :
        if pkt.haslayer(Dot11Beacon):
          capstring=pkt.sprintf("{Dot11Beacon:%Dot11Beacon.cap%}")
                                    cap = {0: "Open", 1: "Enc"}
                                    vrai = 0
                                    pkt.getlayer(Dot11Beacon)
                                    if pkt.addr2 not in ap_list and "\x00" in pkt.info:
                    ap_list.add(pkt.addr2)
                                              if search("privacy", capstring):
                                                        vrai = 1
                    return self.SSID(pkt, cap, vrai)


                                    elif pkt.addr2 not in ap_list:
                                              ap_list.add(pkt.addr2)
                                              if search("privacy", capstring):
                                    vrai = 1
        return self.HIDDEN(pkt, cap, vrai)


def SSID(self, pkt,cap, vrai):
self.channel = int( ord(pkt[Dot11Elt:3].info))
self.ssid = pkt.info
self.bssid = pkt.addr2
self.privacy = cap [vrai]
                if self.ssid not in Values:
                    Values.append([self.ssid, self.channel, self.privacy, self.bssid, self.attackType,
self.attacker, self.target, self.isHidden])
                    self.indicator = 1
                return

defHIDDEN (self,pkt,cap, vrai):
self.channel = int( ord(pkt[Dot11Elt:3].info))
self.ssid = pkt.info
self.bssid = pkt.addr2
self.len = pkt.getlayer(Dot11Beacon).len
self.privacy = cap[vrai]
self.isHidden = True
```

```
            if self.ssid not in self.hidden:

Values.append([self.ssid, self.channel, self.privacy, self.bssid, self.attackType, self.attacker, self.target, self.isHidden]
Return
```

Chaque paquet sniffé est envoyé à la méthode PacketHandler pour qu'il soit traité. Lors du traitement, la méthode vérifie si le type du paquet égale à 0, son sous type égale à 8 ou si ce paquet se trouve dans la couche Do11beacon.Si cela est vérifié donc le paquet est de type beacon frame,ce type de paquet contient SSID et BSSID, elle vérifie aussi l'existence des BSSIDs qui sont déjà reçu par la méthode PacketHandler. Si un BSSID existe dans la liste **ap_list**, il sera écarté, sinon les informations liés à ce BSSID seront envoyé soit à la méthode SSID ou bien à la méthode HIDDEN, ces dernières vont mettre à jour les attributs du constructeur, ces mises à jour vont être affichés par le déconstructeur.

Cette fonctionnalité est aussi capable de différencier entre le SSID caché et le SSID lisible, qui sera stocké pour faciliter l'identification du type d'attaque deauthentification, afin de catégoriser cette dernière soit comme déni de service ou SSID cloaking.

7. Sniffer des requêtes probes

```
defPacketHandler(self, pkt):

      if pkt.haslayer(Dot11):
# interception des probes request
            if pkt.type == 0 and pkt.subtype == 4:
                  if pkt.info:
return self.ProbeReq(pkt)
defProbeReq(self, pkt):
self.STA = pkt.addr2
self.ssid = pkt.info
                  if self.STA not in Probes and self.ssid not in Probes:
         Probes.append([self.STA, self.ssid])
                        self.indicator = 1

                  Return
```

Chaque paquet sniffé est envoyé à la méthode PacketHandler pour qu'il soit traité. Lors du traitement la méthode vérifie si ce dernier se trouve dans la couche Dot11, son type égale à 0 et son sous-type égale à 4.Si cela est vérifié ce paquet est donc de type probe. Une fois le paquet est identifié, la méthode ProbeReq est appelée pour qu'elle fasse des mises à jour aux attributs du constructeur.

8. Sniffer des paquets Deauthentification

```
if pkt.haslayer(Dot11Deauth):
      return self.Deauth(pkt,cap, vrai)

defDeauth(self, pkt, cap, vrai):
self.channel = int( ord(pkt[Dot11Elt:3].info))
self.bssid = pkt.addr2
self.ssid = pkt.info
self.attacker = pkt.addr1
self.attackType = "Deauth"
self.reason = pkt.reason
self.privacy = cap[vrai]
            return Values.append([self.ssid, self.channel, self.privacy, self.bssid,
self.attackType+"|"+self.reason, self.attacker, self.target, self.isHidden])
```

Chaque paquet sniffé est envoyé à la méthode PacketHandler pour qu'il soit traité, lors du traitement la méthode vérifie si ce dernier se trouve dans la couche Dot11 Deauth, une fois le paquet est identifié, la méthode Deauth est appelée pour qu'elle fasse des mises à jour aux attributs du constructeur.

9. Résultat finale

```
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec  5 2015, 20:32:19) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> #!/usr/bin/env python
# Author:Lilia,Fatima
# Purpose: Wireless Intrusion Detection
# Start Date: 02/06/2016
# Issue Date: 17/09/2016
# Program version: v1.0

import logging
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
from re import search
from threading import Thread
from tabulate import tabulate
from multiprocessing import Process
import signal
import os


Table = [["SSID","CH","ENC","BSSID", "AttackType","Attacker","TargeT", "IsHidden"]]
Values = []
Probes = [["STA","ProbesFor ","SSID"]]
ap_list = set()
class WIDS(Thread):
        global Table
        global ap_list
        global Values
        global Probes
        def __init__(self):
                Thread.__init__(self)
                self.iface        = "wlan0"
                self.devices      = set()
                self.hidden       = {}
                self.capabilities = {0: "Encrypted", 1: "Open"}
                self.isencypted   = False
                self.channel      = 0
                self.bssid        = "00:00:00:00:00:00"
                self.STA          = None
                self.ssid         = None
                self.len          = None
```

```
            self.privacy      = None
            self.isHidden     = False
            self.attacker     = "00:00:00:00:00:00"
            self.attackType   = None
            self.reason       = None
            self.client       = None
            self.target = self.bssid
            self.indicator = 0


    def PacketHandler(self, pkt):

            if pkt.haslayer(Dot11):
                    if pkt.type == 0 and pkt.subtype == 4:
                          if pkt.info:
                                  return self.ProbeReq(pkt)

                    elif pkt.type == 0 and pkt.subtype == 8 :
                            if pkt.haslayer(Dot11Beacon):
                                    capstring=pkt.sprintf("{Dot11Beacon:%Dot11Beacon.cap%}")
                                    cap = {0: "Open", 1: "Enc"}
                                    vrai = 0
#                                   pkt.getlayer(Dot11Beacon)
                                    if pkt.addr2 not in ap_list and "\x00" in pkt.info:
                                            ap_list.add(pkt.addr2)

                                            if search("privacy", capstring):
                                                    vrai = 1
                                            return self.HIDDEN(pkt, cap, vrai)

                                    elif pkt.addr2 not in ap_list:
                                            ap_list.add(pkt.addr2)
                                            if search("privacy", capstring):
                                                    vrai = 1
                                            return self.SSID(pkt, cap, vrai)


                    elif pkt.haslayer(Dot11Deauth):
                            return self.Deauth(pkt)
```

```
            elif pkt.haslayer(Dot11AssoReq):
                  return self.AssociationReq(pkt)

            elif pkt.haslayer(Dot11Auth):
                  return self.AuthenticationReq(pkt)
            elif pkt.haslayer(Dot11WEP):
                  return self.ARPReplay(pkt)

      else:
            pass

def ProbeReq(self, pkt):
      self.STA = pkt.addr2
      self.ssid = pkt.info
      if self.STA not in Probes and self.ssid not in Probes:
            Probes.append([self.STA, self.ssid])
            self.indicator = 1
      returnelf

def SSID(self, pkt,cap, vrai):
     self.channel = int( ord(pkt[Dot11Elt:3].info))
     self.ssid = pkt.info
     self.bssid = pkt.addr2
     self.privacy = cap[vrai]
     if self.ssid not in Table:
           Table.append([self.ssid, self.channel, self.privacy, self.bssid, self.attackType, self.attacker, self.target, self.isHidden])
           self.indicator = 1
     return

def HIDDEN(self,pkt,cap, vrai):
      self.channel = int( ord(pkt[Dot11Elt:3].info))
      self.ssid = pkt.info
      self.bssid = pkt.addr2
      self.len = pkt.getlayer(Dot11Beacon).len
      self.privacy = cap[vrai]
      self.isHidden = True
      if self.ssid not in self.hidden:
            self.hidden[self.ssid] = self.isHidden
      Table.append([self.ssid, self.channel, self.privacy, self.bssid, self.attackType, self.attacker, self.target, self.isHidden])
```

```
                return

        def Deauth(self, pkt):
                self.bssid = pkt.addr2
                self.target = pkt.addr1
                self.attacker = pkt.addr2
                self.attackType = "Deauth"
                for table in Table:
                        for item in table:
                                if item == self.bssid:
                                        table[4] = self.attackType
                                        table[5] = self.attacker
                                        table[6] = self.target
                return

        def AssociationReq(self, pkt):
                self.STA = pkt.addr1
                self.client = pkt.addr2
                self.ssid = pkt.info
                return [self.STA, self.client, self.ssid]

        def AuthenticationReq(self, pkt):
                self.client = pkt.addr1
                self.addr2 = pkt.addr2
                return [self.addr1, self.addr2]


       def __del__(self):
                if self.indicator == 1:
                        os.system("clear")
                        print tabulate(Table)
                else:
                        pass

import time
def channel_hopper(interface):
        while True:
                try:
                        channel = random.randrange(1,13)
                        #channel =8
```

```
                        os.system("iwconfig %s channel %d" % (interface, channel))
                        time.sleep(1)
                except KeyboardInterrupt:
                        break


def stop_channel_hop(signal, frame):
    try:
        stop_sniff = True
        channel_hop.terminate()
        channel_hop.join()
    except Exception:
        pass

def main():
        channel_hop = Process(target = channel_hopper, args=("wlan0",))
        channel_hop.start()
        signal.signal(signal.SIGINT, stop_channel_hop)
        while True:
                try:
                        pkt = sniff(iface="wlan0", count=1)
                        if pkt[0].haslayer(Dot11):
                                obj = WIDS()
                                obj.PacketHandler(pkt[0])
                except KeyboardInterrupt:
                        exit()
try:
        if __name__ == "__main__":
                main()
except Exception:
        exit()
```

10.Détection de l'attaque déauthentification

Comme nous l'avons vue précédemment le masquage du SSID, nous a pas permet d'assurer une sécurité fiable à notre point d'accès, ce qui a été vérifié lorsque nous avons appliqué les deux attaques,l'attaque passive et l'attaque active. Pour faire face à ce problème nous avons développé une signature qui permet de détecter ce type d'attaque en basant sur la détection des paquets de type deauthentification comme nous montre la figure suivante :

SSID	CH	ENC	BSSID	AttackType	Attacker	TargeT	IsHidden
AT-VOD	2	Enc	ec:22:80:9c:60:9b		00:00:00:00:00:00	00:00:00:00:00:00	False
peace	5	Enc	c0:ee:fb:26:f5:75	Deauth	88:9f:fa:a4:57:9a	c0:ee:fb:26:f5:75	False
tests	11	Enc	2a:76:3f:45:6b:83		00:00:00:00:00:00	00:00:00:00:00:00	False
HUAWEI-E5172-1888	1	Enc	34:cd:be:c7:18:88		00:00:00:00:00:00	00:00:00:00:00:00	False
HG520	1	Enc	00:1b:9e:e8:5e:e2		00:00:00:00:00:00	00:00:00:00:00:00	False
DJAWEB_23CFD	9	Enc	9c:c1:72:52:3d:04		00:00:00:00:00:00	00:00:00:00:00:00	False

Figure 46 : Détection de l'attaque déauthentification.

11. Discussion

Avec l'évolution des réseaux sans fil et le Wi-Fi en particulier, plusieurs méthodes de sécurité sont créées pour protéger les systèmes informatiques contre les différentes attaques.

Dans ce chapitre, nous avons donné en premier lieu une idée générale sur le langage de programmation Python, sa définition ainsi que ses bases. Nous avons développé aussi, une application ***WaveShark*** qui est capable de détecter plusieurs attaques applicables sur le réseau Wi-Fi.

Conclusion générale

Dans ce mémoire, nous avons présenté une synthèse sur les réseaux Wi-Fi tout en montrant l'évolution de la normalisation en termes de standard de sécurité 802.11.

Nous avons également présenté une étude détaillée sur les vulnérabilités des mécanismes de sécurité Wi-Fi, et les différentes attaques qui exploitent leurs faiblesses. Cette étude nous a permis de prendre conscience de l'étendue des dégâts qui peuvent provoquer ce type du réseau, c'est pourquoila sécurité informatique demeure un sujet très essentielle et aussi sensiblevoire même complexe, car l'évolution des technologies a permis d'améliorer plusieurs mécanismes de sécurité dans un réseauWi-Fi, mais il est toujours difficile de la garantir à 100%.

Afin d'optimiser la sécurité et de minimiser toutes sortes de piratage informatique, on a développé dans notre mémoire une application ***WaveShark***,qui est capable de détecter quelques attaques applicables sur le réseau Wi-Fi, à savoir l'attaque ARP Replay, l'attaque déauthentification.

Pour améliorer notre travail, nous proposons d'améliorer ***WaveShark***en ajoutant des signatures qui permettent d'identifier d'autres types d'attaques comme caffe latte et d'utiliser un langage de programmation orienté objet plus robuste comme java.

Références

Références

[1] :http://fr.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers

[2] :http://www.commentcamarche.net/faq/7257-windows-utilisation-d-adaptateurs-wifi-et-problemes-eventuels

[3] :https://fr.wikipedia.org/wiki/Hacker_(s%C3%A9curit%C3%A9_informatique)

[4] : http://www.simpliweb.fr/blog/les-normes-wifi-a-quoi-ca-sert/

[5] :https://fr.wikipedia.org/wiki/IEEE_802.11i

[6] :https://fr.wikipedia.org/wiki/IEEE_802.1X

[7] :http://www.memoireonline.com/04/12/5653/m_Mise-en-place-sur-le-point-dacces-dun-reseau-wifi9.html

[8] : http://www.commentcamarche.net/contents/1282-les-modes-de-fonctionnement-du-wifi-802-11-ou-wi-fi

[9] : magasine Réseaux et télécoms, Claude Servin, 980 pages,3 ème édition, 2009.

[10] : http://www.pouf.org/documentation/securite/html/node15.html

[11] : http://www.commentcamarche.net/contents/1284-securiser-un-reseau-wifi

[12] : Magasine Misc multi System&internet Security cookbook, Frédéric Raynal, 82 pages, 3éme édition, 2011.

[13] :python.developpez.com

Annexes

Wifi Alliance :

La Wi-Fi Alliance a été connu en 1999 sous le nom (WECA).La Wi-Fi Alliance est une organisation mondiale qui traite des produits de différents fabricants qui sont certifiés sur la base de la norme IEEE 802.11.

Trafic :

C'est l'ensemble de données qui transitent sur le réseau.

BSS :

Il correspond à L'ensemble formé par le point d'accès et les stations situées dans sa zone de couverture .Un BSS forme une cellule.

BSSID :

Un identifiant de 6 octets (48 bits). Le BSSID correspond à l'adresse MAC d'un périphérique.

DS :

Système de distribution permet de connecter plusieurs points d'accès afin qu'ils puissent constituer un ESS, il peut être un réseau filaire ou bien sans fil.

ESS :

Il est formé par un ensemble de BSS et un système de distribution.

ESSID :

ESSID c'est un identifiant de 32 caractères de longueur, il représente le nom du réseau, souvent abrégé en SSID, et représente en quelque sorte un premier niveau de sécurité dans la mesure où la connaissance du SSID est nécessaire pour qu'une station se connecte au réseau étendu.

IBSS :

Un IBSS est un réseau sans fil constitué au minimum de deux stations et qui n'utilise pas de point d'accès.

Authentification :

Action de prouver son identité afin de faire partie du BSS ou ESS.

Association :

C'est une opération qui permet d'obtenir une association afin de pouvoir communiquer sur le réseau.

Mise en place de machine virtuelle

Utiliser un environnement virtuel, nous permet d'installer des machines virtuelles sur notre ordinateur. Ces machines virtuelles seront accessibles par le réseau comme si nous avions vraiment d'autres ordinateurs connectés sur notre réseau local.

VMware Workstation

VMware Workstation est hyper viseur qui fonctionne sur des ordinateurs, il permet aux utilisateurs de mettre en place une ou plusieurs machines virtuelles sur une seule machine physique, et de les utiliser simultanément avec la machine réelle. Chaque machine virtuelle peut exécuter son propre système d'exploitation.

Kali Linux :

Kali Linux est une distribution Linux de tests de pénétration et d'audit de sécurité informatique très avancé. Ce système permet de faire des exploits et de pirater les réseaux. Bien entendu, Kali Linux a pour but de permettre aux professionnels de la sécurité et aux entreprises de tester la robustesse de leur système et de leurs réseaux.

Adresse de diffusion (broadcast) :

Diffusion de données à un ensemble de machines d'un même réseau dont on ne connait pas l'adresse MAC.

Adresse MAC :

Est un identifiant physique unique d'une machine codée sur 6 octets et représentée en générale sous forme hexadécimale comme illustrer dans l'exemple suivant : **5E:FF:56:A2:AF:15.**

Aircrack-ng

Aircrack-ng est un logiciel utilitaire destiné à tester la qualité de la protection autour de votre connexion Wi-Fi. Aircrack-ng est un « sniffer », c'est-à-dire qu'il teste des paquets IP transitant entre le routeur et votre ordinateur à la recherche de failles.

Cette suite contient les différents outils, parmi ses outils on a :

- **airmon-ng** : permet d'activer/désactiver le mode moniteur d'une carte wifi.
- **airodump-ng** : Cet outil permet de scanner les réseaux Wi-Fi disponibles et par la suite de capturer les données (paquets) qui transitent par le Wi-Fi.
- **aireplay-ng** : permet l'injection de paquets 802.11.
- **aircrack-ng** : casseur de clés WEP, WPA, WPA2.
- **airbase-ng** : permet de créer un faux point d'accès.

Procédures d'utilisation d'Aircrack-ng

Deauth : désauthentifier 1 ou toutes les stations (-0).

-a : définir l'adresse MAC du point d'accès.

Fakeauth : fausse authentification avec le point d'accès.

-e (--essid) : nom du réseau.

-h : adresse MAC d'un client.

-b (--bssid) : adresse MAC du point d'accès.

-W: 0|1 : règle [ou pas] le flag WEP flag dans les beacons 0|1 (défaut: auto).

-L : attaque « Caffe-Latte ».

-D: désactiver la détection du point d'accès.

-**c** : canal.

-w: spécifie le nom du fichier de capture qui sera créé exemple: -w wpacrack.
-**v** : afficher plus de détaille.

-i: Interface utilisée.

-d 0 : établir un délai entre deux tentatives des PIN.

Attaque informatique

Une attaque est une action malveillante consistant à tenter de contrôle les fonctions et les mesures de sécurités d'un système informatique, voler ses données confidentielles, détruire, endommager ou altérer son fonctionnement normal.

Attaque passive :

Consiste à écouter sans modifier les données ou le fonctionnement d'un réseau.

Attaque active :

Consiste à modifier des données ou des messages ou à perturber le bon fonctionnement d'un réseau.

Wireshark

Wireshark est un analyseur de paquets libre utilisé dans le dépannage et l'analyse de réseaux informatiques.

Hacker

Un hacker est un passionné des réseaux informatiques, doté d'une connaissance très développée, cherchant toujours à repousser les limites de l'impossible, et à défier les systèmes sécurisés. Un hacker peut être bienfaiteur ou malfaiteur ou bien quelque part entre les deux. Tout dépend du but et des moyens qu'il choisit pour écouter, changer ou faire évoluer les systèmes informatiques.

Protocole :

Ensemble de messages échangés entre plusieurs entités.

Le protocole ARP :

Le protocole ARP est utilisé pour faire la correspondance entre des adresses IP sur 32 bits et l'adresse Ethernet sur 48 bits. Par exemple, quand une machine A (192.168.1.1) veut communiquer avec une machine B (192.168.1.2), l'adresse IP connue de B doit être traduite avec son adresse MAC correspondante. Pour réaliser cela, la machine A envoie un message en diffusion contenant l'adresse IP de la machine B (*Who has 192.168.1.2? Tell 192.168.1.1*). La machine cible, reconnaissant que l'adresse IP du paquet correspond à la sienne, retourne une réponse révélant son adresse MAC (*192.168.1.2 is at 01:23:45:67:89:0A*). La réponse est généralement mise en cache.

4 way handshake : (la poignée de mains) est un ensemble de paquets émis par le point d'accès et la station lorsque la station se connecte.

Intrusion

Une intrusion est un événement permettent d'avoir indument accès à un système et ses ressources. L'intrusion est généralement vue comme une personne étrangère au système informatique qui réussit à en prendre le contrôle.

PWR : puissance du signal.

Beacons : nombre de paquets d'annonces envoyés par le point d'accès pour se faire connaitre sur le réseau.

#Data : nombre de paquets capturés.

#/S: nombres de paquets par les seconds.

CH: numéro de canal d'émission utilisé par le point d'accès.

MB: vitesse maximum supporté par le point d'accès Wi-Fi.

ENC: protocole de cryptage.

CIPHER: sur quel type de chiffrement se base le protocole de cryptage.

AUTH: protocole d'authentification utilisé.

Scapy :

Scapy est une bibliothèque Python permettant, d'envoyer, réceptionner et manipuler des paquets réseau. Scapy peut fonctionner à la manière d'un analyseur de trafic réseaux afin d'effectuer des captures de données en vue de les visualiser par la suite.

Sniffer :

Le sniffing est l'action d'écouter le trafic sur un réseau à l'aide d'un logiciel appelé « **renifleur** » ou « **sniffer** » en anglais, ce dernier est capable d'intercepter toutes les trames que notre carte reçoit et qui ne nous sont pas destinées.