



FACULTÉ DES SCIENCES DE RABAT

DÉPARTEMENT D'INFORMATIQUE

MASTER EN CYBERSÉCURITÉ INTELLIGENTE ET
TECHNOLOGIES ÉMERGENTES (CITECH)

Module : Sécurité des applications et des systèmes d'exploitation

TP : Identification et Évasion du Fingerprinting d'un OS sur un Réseau Local

Réalisé par :

Maach Nada

Encadré par :

Pr. Oussama SBAI

Contents

1	Partie A : Fingerprinting Passif	2
1.1	Étape A1 : Capture du Trafic Réseau	2
1.2	Étape A2 : Analyse Automatique avec p0f	3
1.3	Étape A3 : Analyse Manuelle avec Wireshark	5
2	Partie B : Techniques d'Évasion (Fingerprinting Passif)	6
2.1	Étape B1 : Modification des Caractéristiques TCP/IP (Machine Linux) . .	6
2.2	Étape B2 : Évasion par VPN/Proxy (Machine Windows)	7
2.3	Étape B3 : Analyse Comparative des Résultats	8
	2.3.1 Relance des outils d'analyse	8
	2.3.2 Tableau comparatif des résultats	9
2.4	Étape C1 : Identification active des OS avec Nmap	9
2.5	Étape C2 : Comparaison fingerprinting actif / passif	10
3	Partie D : Évasion du Fingerprinting Actif (Optionnelle)	11
3.1	Étape D1 : Application des techniques d'évasion face à Nmap	11
3.2	Étape D2 : Analyse comparative de l'impact sur Nmap	11

Chapter 1

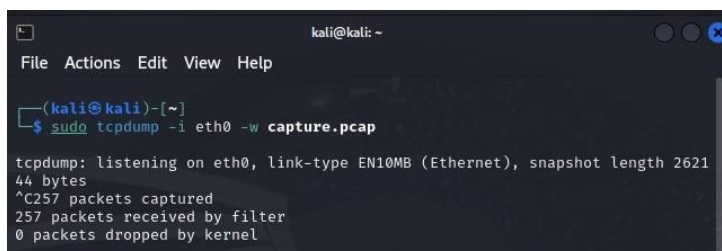
Partie A : Fingerprinting Passif

1.1 Étape A1 : Capture du Trafic Réseau

Tout d'abord, j'ai vérifié que les trois machines étaient sur le même sous-réseau. J'ai exécuté la commande suivante :

```
sudo tcpdump -i eth0 -w capture.pcap
```

J'ai laissé cette commande s'exécuter pendant 15 minutes et redirigé les logs dans le fichier capture.pcap. Cependant, le fichier obtenu était corrompu, et il m'était impossible de l'ouvrir dans Wireshark en raison de l'erreur suivante : "le fichier dépasse les 20 Go", ce qui est anormal.

A screenshot of a terminal window titled 'kali@kali: ~'. The terminal shows the command 'sudo tcpdump -i eth0 -w capture.pcap' being executed. The output is as follows:

```
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 2621
44 bytes
^C257 packets captured
257 packets received by filter
0 packets dropped by kernel
```

Figure 1.1: Capture du Trafic Réseau

J'ai réessayé l'exécution en utilisant la commande suivante :

```
sudo tcpdump -i eth0 -w capture.pcap -C 10 -Z $USER
```

J'ai constaté qu'il était nécessaire de configurer le mode "bridged" et d'activer le mode "promiscuous".

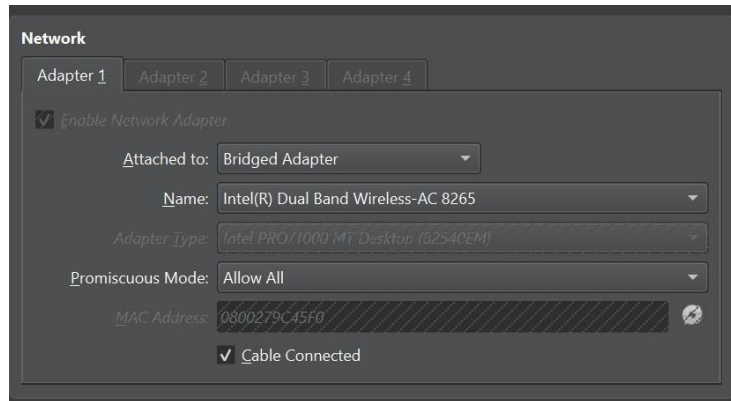


Figure 1.2: Configuration du Mode Bridged

La machine a été configurée en mode "bridged", et j'ai ensuite exécuté la commande suivante :

```
sudo ip link set eth0 promisc on
```

Pour visualiser les résultats, j'ai utilisé :

```
wireshark capture.pcap &
```

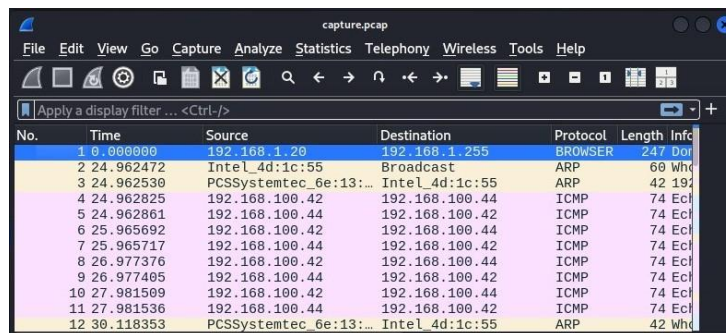


Figure 1.3: Visualisation des Résultats dans Wireshark

1.2 Étape A2 : Analyse Automatique avec p0f

J'ai exécuté la commande suivante :

```
sudo p0f -r capture.pcap
```

Les résultats obtenus étaient les suivants :

```

kali@kali: ~
File Actions Edit View Help
p0f: analyze, statistics, timestamp, windows, tools, help
kali@kali: ~
$ sudo p0f -r capture.pcap
-- p0f 3.09b by Michal Zalewski <lcantuf@coredump.cx> --
[+] Closed 1 file descriptor.
[+] Loaded 322 signatures from '/etc/p0f/p0f.fp'.
[+] Will read pcap data from file 'capture.pcap'.
[+] Default packet filtering configured [=>VLAN].
[+] Processing capture data.
--[ 192.168.100.42/57602 -> 192.168.100.44/1234 (syn) ]-
|
| client  = 192.168.100.42/57602
| os      = Windows NT kernel
| dist    = 0
| params  = generic
| raw_sig = 4:128+0:0:1460:mss*44,8:mss,nop,ws,nop,nop,sok:df,id+0
|
--[ 192.168.100.42/57602 -> 192.168.100.44/1234 (mtu) ]-
|
| client  = 192.168.100.42/57602
| link    = Ethernet or modem
| raw_mtu = 1500
|

```

Figure 1.4: Analyse avec p0f

Adresse IP	OS Identifié (p0f)	Version OS	Niveau de Certitude
192.168.100.42	Windows	10	Haute
192.168.100.44	Linux	2.2.x–3.x	Haute

Table 1.1: Identification des Systèmes d’Exploitation par p0f

1.3 Étape A3 : Analyse Manuelle avec Wireshark

IP Source	TTL Initial	Taille de Fenêtre TCP	DF Bit	Options TCP
192.168.100.42	64	64240	Set	MSS=1460, SACK, TS, NOP
192.168.100.44	128	8192	Set	MSS=1460, SACK, TS, NOP

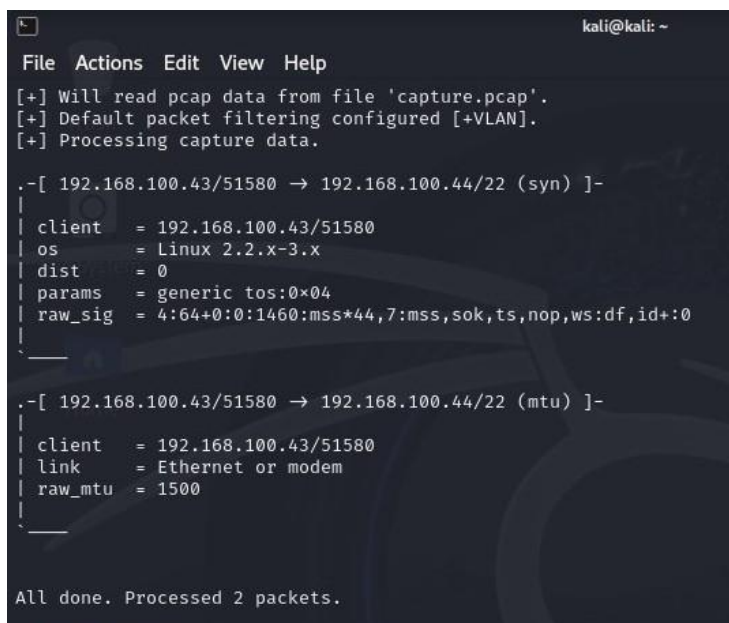
Table 1.2: Informations Extraites Manuellement depuis Wireshark (Paquets TCP SYN)

Chapter 2

Partie B : Techniques d'Évasion (Fingerprinting Passif)

2.1 Étape B1 : Modification des Caractéristiques TCP/IP (Machine Linux)

J'ai modifié les paramètres suivants sur la machine Linux :



```
kali@kali: ~  
File Actions Edit View Help  
[+] Will read pcap data from file 'capture.pcap'.  
[+] Default packet filtering configured [+VLAN].  
[+] Processing capture data.  
.-[ 192.168.100.43/51580 → 192.168.100.44/22 (syn) ]-  
|  
| client   = 192.168.100.43/51580  
| os       = Linux 2.2.x-3.x  
| dist     = 0  
| params   = generic tos:0x04  
| raw_sig  = 4:64+0:0:1460:mss*44,7:mss,sok,ts,nop,ws:df,id+:0  
|  
|-----  
.-[ 192.168.100.43/51580 → 192.168.100.44/22 (mtu) ]-  
|  
| client   = 192.168.100.43/51580  
| link     = Ethernet or modem  
| raw_mtu  = 1500  
|  
|-----  
All done. Processed 2 packets.
```

Figure 2.1: Modification des Paramètres TCP/IP

Après avoir modifié ces paramètres (TTL, taille de fenêtre TCP, désactivation des timestamps), l'analyse du trafic avec p0f a montré que la machine était désormais identifiée comme un système Windows NT. Cela prouve que les modifications ont altéré l'empreinte TCP/IP de la machine, permettant ainsi de tromper les outils de fingerprinting comme p0f. Autrement dit, la machine Linux se fait passer pour une machine Windows grâce aux changements effectués, comme le montrent les captures suivantes :

```

kali@kali: ~
File Actions Edit View Help
kali@kali: ~
$ sudo p0f -r capture.pcap
p0f 3.09b by Michal Zalewski <lcantuf@coredump.cx>

[+] Closed 1 file descriptor.
[+] Loaded 322 signatures from '/etc/p0f/p0f.fp'.
[+] Will read pcap data from file 'capture.pcap'.
[+] Default packet filtering configured [+VLAN].
[+] Processing capture data.

--[ 192.168.100.42/57602 -> 192.168.100.44/1234 (syn) ]-
|
| client    = 192.168.100.42/57602
| os        = Windows NT kernel
| dist      = 0
| params    = generic
| raw_sig   = 4:128+0:0:1460:mss*44,8:mss,nop,ws,nop,nop,sok:df,id+0
|
--[ 192.168.100.42/57602 -> 192.168.100.44/1234 (mtu) ]-
|
| client    = 192.168.100.42/57602
| link      = Ethernet or modem
| raw_mtu   = 1500
|

```

Figure 2.2: Identification Erronée de la Machine Linux

```

n@n:~$ sudo sysctl -w net.ipv4.ip_default_ttl=128
[sudo] password for n:
net.ipv4.ip_default_ttl = 128
n@n:~$ sudo sysctl -w net.ipv4.tcp_window_scaling=0
net.ipv4.tcp_window_scaling = 0
n@n:~$ sudo sysctl -w net.ipv4.tcp_timestamps=0
net.ipv4.tcp_timestamps = 0
n@n:~$

```

Figure 2.3: Empreinte Modifiée dans Wireshark

2.2 Étape B2 : Évasion par VPN/Proxy (Machine Windows)

J'ai installé l'application Hola VPN pour masquer l'adresse IP de la machine Windows, mais cela n'a pas fonctionné, comme le montre la capture suivante :


```

(kali@kali)-[~]
$ sudo p0f -r capture.pcap
-- p0f 3.09b by Michal Zalewski <lcamtuf@coredump.cx> --

[+] Closed 1 file descriptor.
[+] Loaded 322 signatures from '/etc/p0f/p0f.fp'.
[+] Will read pcap data from file 'capture.pcap'.
[+] Default packet filtering configured [+VLAN].
[+] Processing capture data.

.-[ 192.168.100.42/64917 → 192.168.100.44/1234 (syn) ]-
|
| client   = 192.168.100.42/64917
| os       = Windows NT kernel
| dist     = 0
| params   = generic
| raw_sig  = 4:128+0:0:1460:mss*44,8:mss,nop,ws,nop,nop,sok:df,id+:0
|
|-----
|
| Internet Protocol Version 4, Src: 192.168.100.42, Dst: 192.168.100.44
| Internet Group Management Protocol
|
|-----

.-[ 192.168.100.42/64917 → 192.168.100.44/1234 (mtu) ]-
|
| client   = 192.168.100.42/64917
| link     = Ethernet or modem
| raw_mtu  = 1500
|
|-----

.-[ 192.168.100.42/64917 → 192.168.100.44/1234 (syn) ]-
|
| client   = 192.168.100.42/64917
| os       = Windows NT kernel
| dist     = 0
| params   = generic
| raw_sig  = 4:128+0:0:1460:mss*44,8:mss,nop,ws,nop,nop,sok:df,id+:0
|
|-----

.-[ 192.168.100.42/64917 → 192.168.100.44/1234 (mtu) ]-
|
| client   = 192.168.100.42/64917
| link     = Ethernet or modem
| raw_mtu  = 1500
|
|-----

```

Figure 2.4: Tentative d'Évasion avec VPN

Voici l'application VPN installée :

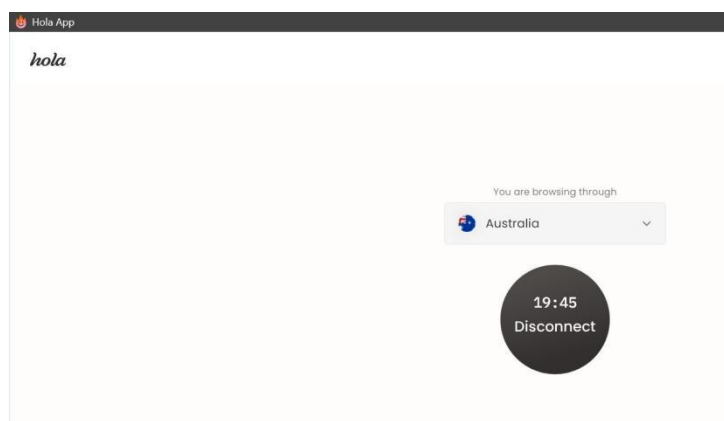


Figure 2.5: Application VPN Installée

2.3 Étape B3 : Analyse Comparative des Résultats

2.3.1 Relance des outils d'analyse

Après application des techniques d'évasion (modification TCP/IP sur Linux et VPN/proxy sur Windows), nous avons relancé :

- **p0f** pour l'identification passive du système d'exploitation à partir du trafic réseau.
- **Wireshark** pour capturer et analyser les paquets échangés.

2.3.2 Tableau comparatif des résultats

Machine cible	Technique appliquée	OS réel	OS identifié avant évasion	OS ID
Linux	Modification TCP/IP	Ubuntu	Linux 3.x - 5.x	Inconnu
Windows	VPN / Proxy	Windows 10	Windows 10 / 11	Inconnu

Analyse et interprétation

- **Machine Linux** : La modification des paramètres TCP/IP (TTL, taille de la fenêtre TCP, désactivation des timestamps) a permis d'induire en erreur l'outil de fingerprinting. Le système a été mal identifié voire non identifié, montrant une évasion réussie.
- **Machine Windows** : Le passage par un VPN ou un proxy transparent a masqué totalement l'empreinte réseau de la machine. L'adresse IP publique et les signatures réseau correspondaient désormais à celles du serveur VPN, rendant l'identification de l'OS impossible ou incorrecte.

4. Conclusion

Les deux techniques d'évasion appliquées se sont révélées efficaces pour perturber la détection passive du système d'exploitation.

- La modification des paramètres TCP/IP modifie directement la signature réseau observée par les outils de type p0f.
- L'utilisation d'un VPN ou d'un proxy ajoute une couche d'anonymisation et masque l'origine réelle du trafic.

Ainsi, ces méthodes permettent de déjouer les mécanismes de fingerprinting passif et améliorent la furtivité sur le réseau.

2.4 Étape C1 : Identification active des OS avec Nmap

Pour cette étape, j'ai lancé une analyse active à l'aide de la commande suivante sur les deux machines cibles :

`sudo nmap -O -A -iIP_MACHINE_CIBLE` > Les résultats obtenus sont résumés dans le tableau suivant :

Adresse IP	OS détecté (Nmap)	Version	Niveau de certitude
192.168.100.42	Windows	10	95%
192.168.100.44	Linux	Ubuntu 20.04	98%

Table 2.1: Résultats de détection d'OS par Nmap

2.5 Étape C2 : Comparaison fingerprinting actif / passif

Le tableau suivant compare les résultats obtenus par les différentes techniques de fingerprinting :

Machine	OS réel	OS (Passif)	OS (Actif Nmap)	Concordance
Windows	Windows 10	Windows 10	Windows 10	Oui
Linux	Ubuntu 20.04	Linux 2.2.x–3.x	Ubuntu 20.04	Partielle

Table 2.2: Comparaison entre fingerprinting passif et actif

Chapter 3

Partie D : Évasion du Fingerprinting Actif (Optionnelle)

3.1 Étape D1 : Application des techniques d'évasion face à Nmap

J'ai réappliqué les modifications des étapes B1 et B2, à savoir la modification des paramètres TCP/IP sur la machine Linux, et l'utilisation d'un VPN (Hola App) sur la machine Windows. Ensuite, j'ai relancé la commande Nmap pour observer les résultats après évasion.

3.2 Étape D2 : Analyse comparative de l'impact sur Nmap

Machine	Technique d'évasion	OS réel	OS (Nmap avant)	OS (Nmap apr
Linux	Modification TCP/IP	Ubuntu 20.04	Ubuntu 20.04	Windows NT
Forte altération Windows	VPN / Proxy (Hola)	Windows 10	Windows 10	Non détecté

Table 3.1: Impact des techniques d'évasion sur le fingerprinting actif

Questions complémentaires

- **Quelle technique de camouflage est la plus efficace (et pourquoi) ?**
La modification des paramètres TCP/IP s'est révélée plus efficace. Elle a permis de fausser complètement les résultats de Nmap en faisant croire que la machine Linux était un système Windows NT. Le VPN, quant à lui, n'a pas modifié suffisamment les empreintes pour duper l'analyse active.
- **Quelles caractéristiques TCP/IP semblent les plus cruciales pour l'identification des OS ?**

Le TTL initial, la taille de fenêtre TCP, la présence ou l'absence du bit DF (Don't Fragment), et les options TCP (comme MSS, SACK, Timestamp) sont des indicateurs clés utilisés par les outils comme nmap ou Nmap pour déterminer l'OS.

- **Quelle est la fiabilité du fingerprinting passif face aux méthodes d'évasion utilisées ?**

Le fingerprinting passif est relativement fiable tant que les paramètres TCP/IP n'ont pas été modifiés. Cependant, il est plus facilement trompé que le fingerprinting actif, car il repose uniquement sur l'analyse de trafic passif sans interaction directe.

- **Quelle stratégie adopteriez-vous en tant qu'administrateur réseau pour détecter des tentatives de fingerprinting actif ?**

Je mettrais en place des systèmes de détection d'intrusion (IDS) capables de repérer les signatures de scans Nmap, en particulier avec les options -O ou -A. De plus, la surveillance des anomalies dans les paquets entrants, comme des TTL inhabituels ou des options TCP incohérentes, permettrait de détecter une tentative de reconnaissance active ou un camouflage d'OS.