

# 1 Choix et solutions

Pour le sniffing avec Scapy, nous avons implémenté une seule fonction `listen.py` qui s'exécute sur les deux `netns` (avec détection du `netns` et setup des variables à l'aide de macros).

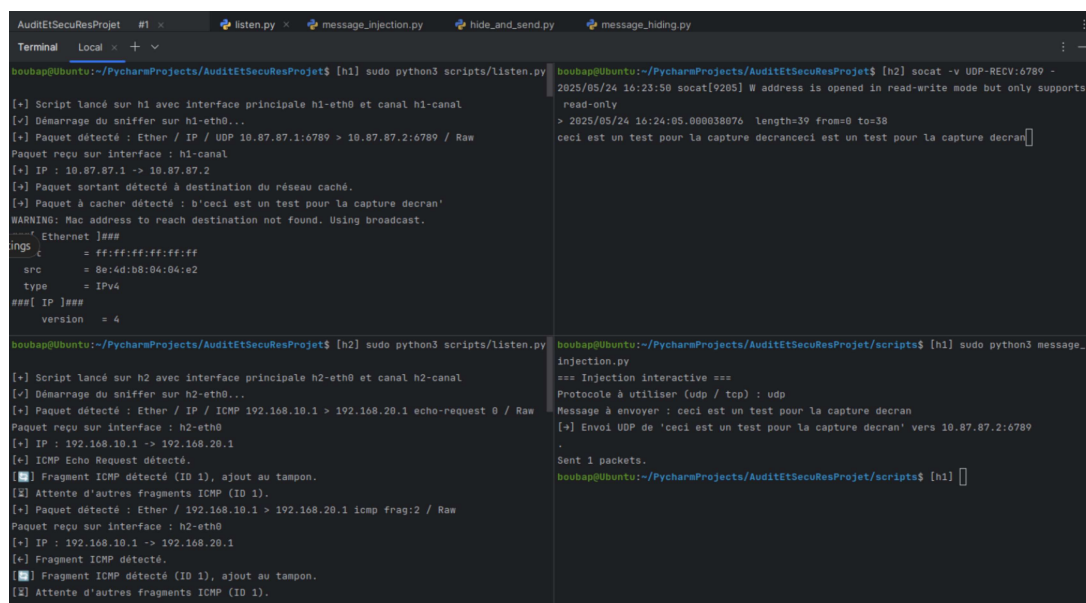
Il détecte les paquets cachés (donc, à destination des `veth`), et initie le processus d'encapsulation pour cacher le paquet dans un paquet ICMP.

Nous avons décidé de chiffrer à l'aide d'un *One Time Pad* généré au moment de la réception d'un paquet à destination du canal. Une fonction dédiée génère une clé `key.pem` partagée entre les deux hôtes (ici, pour la simplicité des tests, on garde la clé dans le système de fichiers...) qui permet de chiffrer le message à l'aide d'un XOR.

Pour envoyer un message à un hôte, il suffit d'exécuter `message_injection.py` et de suivre les instructions. Le fichier `READ.ME` contient des instructions plus détaillées si besoin.

L'ensemble des détails des transactions peuvent être observés dans la console lors de l'exécution de `listen.py`, qui affiche des détails sur les paquets envoyés/reçus sur les interfaces sniffées, ainsi que le contenu du payload (pour simuler un sniffing de sécurité et montrer que le payload est chiffré correctement).

Voici les essais obtenus en UDP et TCP :



```
AuditEtSecuResProjet #1 x listen.py message_injection.py hide_and_send.py message_hiding.py
Terminal Local x + v
boubap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet$ [h1] sudo python3 scripts/listen.py
[*] Script lancé sur h1 avec interface principale h1-eth0 et canal h1-canal
[*] Démarrage du sniffer sur h1-eth0...
[*] Paquet détecté : Ether / IP / UDP 10.87.87.1:6789 > 10.87.87.2:6789 / Raw
Paquet reçu sur interface : h1-canal
[*] IP : 10.87.87.1 -> 10.87.87.2
[*] Paquet sortant détecté à destination du réseau caché.
[*] Paquet à cacher détecté : b'ceci est un test pour la capture decran'
WARNING: Mac address to reach destination not found. Using broadcast.
--- Ethernet ---
Ingress:
  c = ff:ff:ff:ff:ff:ff
  src = 8e:4d:b8:04:04:e2
  type = IPv4
### IP ###
version = 4

boubap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet$ [h2] sudo python3 scripts/listen.py
[*] Script lancé sur h2 avec interface principale h2-eth0 et canal h2-canal
[*] Démarrage du sniffer sur h2-eth0...
[*] Paquet détecté : Ether / IP / ICMP 192.168.10.1 > 192.168.20.1 echo-request 0 / Raw
Paquet reçu sur interface : h2-eth0
[*] IP : 192.168.10.1 -> 192.168.20.1
[*] ICMP Echo Request détecté.
[ ] Fragment ICMP détecté (ID 1), ajout au tampon.
[ ] Attente d'autres fragments ICMP (ID 1).
[*] Paquet détecté : Ether / 192.168.10.1 > 192.168.20.1 icmp frag:2 / Raw
Paquet reçu sur interface : h2-eth0
[*] IP : 192.168.10.1 -> 192.168.20.1
[*] Fragment ICMP détecté.
[ ] Fragment ICMP détecté (ID 1), ajout au tampon.
[ ] Attente d'autres fragments ICMP (ID 1).

boubap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet/scripts$ [h1] sudo python3 message_injection.py
=== Injection interactive ===
Protocole à utiliser (udp / tcp) : udp
Message à envoyer : ceci est un test pour la capture decran
[*] Envoi UDP de 'ceci est un test pour la capture decran' vers 10.87.87.2:6789
.
Sent 1 packets.
boubap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet/scripts$ [h1]
```

FIGURE 1 – Essai en UDP

```

dst      = 10.87.87.2
\options \
###[ TCP ]###
sport    = 6789
dport    = 6789
seq      = 0
ack      = 0
dataofs  = None
reserved = 0
flags    = 5
window  = 8192
chksum   = None
urgste   = 0
options  = ''
###[ Raw ]###
load     = 'HIDE:ceci est un tcp syn'

[ ] Fragment ICMP détecté (ID 1), ajout au tampon.
[ ] Attente d'autres fragments ICMP (ID 1).
[+] Paquet détecté : Ether / 192.168.10.1 > 192.168.20.1 icmp frag:0 / Raw
Paquet reçu sur interface : h2-eth0
[+] IP : 192.168.10.1 -> 192.168.20.1
[+] Fragment ICMP détecté.
[ ] Fragment ICMP détecté (ID 1), ajout au tampon.
[ ] Attente d'autres fragments ICMP (ID 1).
Python Packages : détecté : Ether / 192.168.10.1 > 192.168.20.1 icmp frag:0 / Raw
Paquet reçu sur interface : h2-eth0
[+] IP : 192.168.10.1 -> 192.168.20.1
[+] Fragment ICMP détecté.
[ ] Fragment ICMP détecté (ID 1), ajout au tampon.
[+] Tous les fragments reçus, message complet prêt.
[+] Message caché détecté : ceci est un tcp syn

[sudo] password for bouhap:
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
bouhap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet$ [h2] sudo tcpdump -i any 'tcp[tcpflags] & tcp-syn != 0 and tcp port 6789'
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
16:26:11.690469 lo M IP 192.168.10.1.5678 > 192.168.20.1.6789: Flags [S], seq 0:19,
win 8192, length 19
[ ]

injection.py
=== Injection interactive ===
Protocole à utiliser (udp / tcp) : tcp
Message à envoyer : ceci est un paquet syn
[+] Envoi TCP SYN vers 10.87.87.2:6789
.
Sent 1 packets.
bouhap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet/scripts$ [h1] sudo python3 message_
injection.py
=== Injection interactive ===
Protocole à utiliser (udp / tcp) : tcp
Message à envoyer : ceci est un tcp syn
[+] Envoi TCP SYN vers 10.87.87.2:6789
.
Sent 1 packets.
bouhap@Ubuntu:~/PycharmProjects/AuditEtSecuResProjet/scripts$ [h1]

```

FIGURE 2 – Essai en TCP