26/01/2026

SAE503 : Assurer la sécurisation et la supervision avancées d'un système d'information

Superviseur : Dr. Mohamed Amine Ouamri

Aliya MAATIT
BUT3 RESEAUX ET TELECOMMUNICATIONS - TPB

Table des matières

Introduction

Dans un monde où les systèmes d'information sont de plus en plus exposés aux menaces, la sécurisation et la supervision des réseaux constituent des enjeux majeurs.

La SAE 503 s'inscrit dans cette problématique en proposant une mise en pratique des mécanismes de sécurité réseau, de détection d'attaques et d'analyse de comportements malveillants.

L'objectif de ce travail est d'étudier et de mettre en œuvre différents outils de protection et d'attaque afin de mieux comprendre leur fonctionnement, leurs limites et les risques associés. Les manipulations réalisées portent notamment sur la configuration d'un pare-feu sous Linux, la mise en place du NAT, le filtrage du trafic réseau, ainsi que l'étude d'attaques de type Man-in-the-Middle, ARP Spoofing et détournement DNS.

Ce rapport présente de manière structurée l'ensemble des expérimentations réalisées, en s'appuyant sur des captures, des analyses techniques et des explications détaillées, dans le but de démontrer les vulnérabilités possibles d'un réseau mal sécurisé et l'importance de mécanismes de défense adaptés.

1. Observation d'Internet et Analyse Malware

1.1. Partie 1 : Configuration Réseau Firewall

1.1.1. Principe du pare-feu et rôle du noyau Linux

Un pare-feu est un dispositif de sécurité qui permet de contrôler le trafic réseau entrant et sortant en appliquant des règles de filtrage. Il analyse les paquets selon différents critères (adresse IP, port, protocole) pour d'autoriser les communications légitimes et bloquer les accès non autorisés ou malveillants.

Le noyau Linux permet ceci grâce au NAT et MASQUERADE. Ce mécanisme remplace l'adresse IP privée des machines internes par l'adresse IP publique du routeur, rendant ainsi les machines du réseau privé capables d'accéder à Internet même si leurs adresses ne sont pas routables.

1.1.2. Tables et chaînes iptables

Table	Description	Chaines utilisées

filter	Table par défaut qui sert au filtrage du trafic réseau	INPUT, OUTPUT, FORWARD
nat	Table permettant la traduction d'adresses réseau (NAT)	PREROUTING, POSTROUTING, OUTPUT

1.1.3. Mise en place de l'architecture réseau

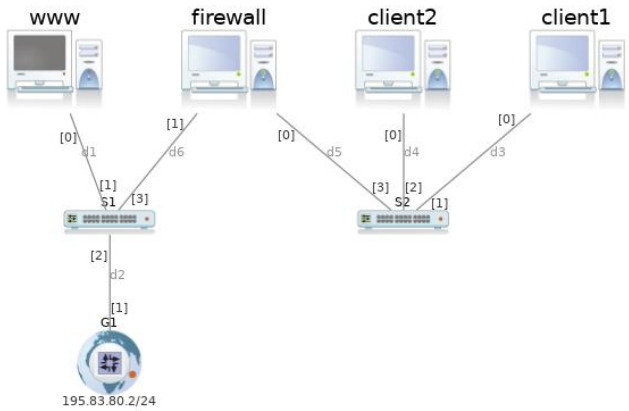


Figure 1: Architecture réseau mise en place

L'architecture réseau mise en place repose sur une séparation entre un réseau public et un réseau privé, reliés par un routeur jouant le rôle de pare-feu.

Le réseau privé (192.168.20.0/24) contient deux machines clientes, tandis que le réseau public (195.83.80.0/24) représente l'accès à Internet. Le firewall possède deux interfaces réseau : l'une connectée au réseau privé et l'autre au réseau public.

1.1.4. Configuration des adresses IP et du routage

Les adresses IP ont été configurées manuellement sur chaque machine. Les clients disposent d'adresses IP privées, tandis que le firewall possède une interface privée et une interface publique.

```
[0 root@client1 ~]$ ip addr add 192.168.20.1/24 dev eth0
[0 root@client1 ~]$ sudo ip link set dev eth0 up
[0 root@client1 ~]$
```

Figure 2 : Adresse IP client1

```
[0 root@client2 ~]$ sudo ip addr add 192.168.20.10/24 dev eth0
[0 root@client2 ~]$ sudo ip link set dev eth0 up
[0 root@client2 ~]$
```

Figure 3 : Adresse IP client2

```
[0 root@firewall ~]$ ip addr add 192.168.20.254/24 dev eth0
[0 root@firewall ~]$ sudo ip link set dev eth0 up
[0 root@firewall ~]$
[0 root@firewall ~]$ ip addr add 195.83.80.254/24 dev eth1
[0 root@firewall ~]$ sudo ip link set dev eth1 up
[0 root@firewall ~]$
```

Figure 4 : Adresse IP firewall

Les routes par défaut des clients ont été configurées vers l'adresse du firewall (192.168.20.254). Le transfert IP a été activé sur le firewall afin de permettre le routage entre les réseaux. Les tests de connectivité confirment le bon fonctionnement du routage.

```
0 root@client1 ~]$ sudo ip route add default via 192.168.20.254
0 root@client1 ~]$ ip route
efault via 192.168.20.254 dev eth0
92.168.20.0/24 dev eth0  proto kernel  scope link  src 192.168.20.1
0 root@client1 ~]$
```

Figure 5 : Route sur le client1

```
[0 root@client2 ~]$ sudo ip route add default via 192.168.20.254
[0 root@client2 ~]$ sudo ip route
default via 192.168.20.254 dev eth0
192.168.20.0/24 dev eth0  proto kernel  scope link  src 192.168.20.10
[0 root@client2 ~]$
```

Figure 6 : Route sur le client2

```
[130 root@firewall ~]$ echo 1 > /proc/sys/net/ipv4/ip_forward
[0 root@firewall ~]$
```

Figure 7 : Activation du routage

```
[0 root@client1 ~]$ ping 192.168.20.254
PING 192.168.20.254 (192.168.20.254) 56(84) bytes of data.
64 bytes from 192.168.20.254: icmp_req=1 ttl=64 time=0.847 ms
64 bytes from 192.168.20.254: icmp_req=2 ttl=64 time=1.98 ms
^C
--- 192.168.20.254 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1009ms
rtt min/avg/max/mdev = 0.847/1.417/1.987/0.570 ms
[0 root@client1 ~]$
[0 root@client1 ~]$ ping 192.168.20.10
PING 192.168.20.10 (192.168.20.10) 56(84) bytes of data.
64 bytes from 192.168.20.10: icmp_req=1 ttl=64 time=0.867 ms
64 bytes from 192.168.20.10: icmp_req=2 ttl=64 time=2.71 ms
^C
--- 192.168.20.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.867/1.791/2.715/0.924 ms
[0 root@client1 ~]$
```

Figure 8 : Test de connectivité

1.1.5. Sécurisation du routage (ICMP Redirect)

Les messages ICMP Redirect ont été désactivés afin d'éviter toute modification dynamique et malveillante des routes réseau.

```
[0 root@firewall ~]$ echo 0 > /proc/sys/net/ipv4/conf/all/send_redirects
[0 root@firewall ~]$ echo 0 > /proc/sys/net/ipv4/conf/eth0/send_redirects
[0 root@firewall ~]$
```

Figure 9 : Désactivation des messages ICMP

1.1.6. Mise en place du NAT (camouflage IP)

Le NAT a été mis en place à l'aide de la cible MASQUERADE dans la table nat. Cette règle permet de masquer les adresses IP privées du réseau interne derrière l'adresse IP publique du firewall.

- -t nat : Indique que la règle est appliquée dans la table NAT
- A POSTROUTING : Ajoute la règle dans la chaîne POSTROUTING (après routage)
- o eth1 : Applique la règle aux paquets sortant par l'interface publique
- j MASQUERADE : Remplace l'adresse IP source privée par l'adresse IP publique du firewall

```
[0 root@firewall ~]$ iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
[0 root@firewall ~]$
```

Figure 10 : Activation du NAT avec MASQUERADE

Figure 11 : Vérification de la règle NAT

1.1.7. Autorisation du trafic du réseau privé

Cette règle autorise le trafic provenant du réseau privé (192.168.20.0/24) à traverser le pare-feu vers le réseau public.

- FORWARD : Gère le trafic traversant le firewall
- s 192.168.20.0/24 : Autorise les paquets provenant du réseau privé
- j ACCEPT : Autorise le passage des paquets

Figure 12 : Autorisation du trafic du réseau privé

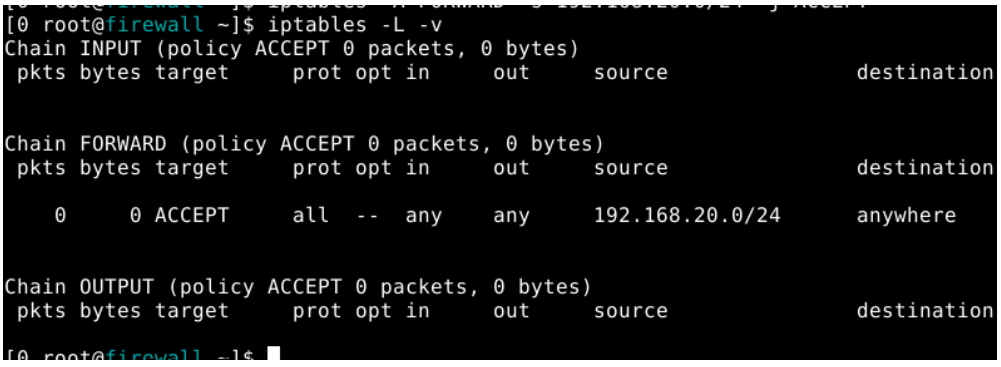


Figure 13 : Vérification de l'autorisation du trafic

1.1.8. Vérification de l’accès Internet

L'adresse affichée correspond à l'adresse IP publique du firewall (195.83.80.254). Cela confirme le bon fonctionnement du NAT : l'adresse IP privée du client n'apparaît pas.

```
[0 root@client1 ~]$ ping 195.83.80.10
PING 195.83.80.10 (195.83.80.10) 56(84) bytes of data.
64 bytes from 195.83.80.10: icmp_req=1 ttl=63 time=1.02 ms
^C
--- 195.83.80.10 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.025/1.025/1.025/0.000 ms
[0 root@client1 ~]$
```

Figure 14 : Vérification de l'accès Internet

```
[0 root@firewall ~]$ epiphany http://195.83.80.10/whatismyip.php

** (epiphany-browser:1655): WARNING **: Unable to start Zeroconf subsystem

(epiphany-browser:1655): GLib-Net-WARNING **: couldn't load TLS file database: F
ailed to open file '/etc/ssl/certs/ca-certificates.crt': No such file or directo
ry

(epiphany-browser:1655): GLib-GObject-CRITICAL **: g_object_unref: assertion `G_
IS_OBJECT (object)' failed

^C
[130 root@firewall ~]$
```

Figure 15 : Vérification d'epiphany

Figure 16 : Vérification de l'adresse IP publique

1.1.9. Filtrage ICMP et journalisation

Une chaîne personnalisée LOG_DROP a été créée pour journaliser les paquets ICMP avant de les rejeter. La cible LOG permet l'enregistrement des événements dans les journaux système, tandis que la cible DROP bloque définitivement les paquets.

```
[0 root@firewall ~]$ iptables -N LOG_DROP
[0 root@firewall ~]$ iptables -A LOG_DROP -j LOG --log-prefix "DROP_ICMP: "
[0 root@firewall ~]$ iptables -A LOG_DROP -j DROP
[0 root@firewall ~]$
```

Figure 17 : Création de la chaîne LOG_DROP

Un filtre ICMP a été appliqué sur la chaîne INPUT afin de bloquer les requêtes ICMP provenant de l'adresse de bouclage. L'affichage des règles iptables confirme que LOG_DROP est bien pris en compte par le firewall.

```
[0 root@firewall ~]$ iptables -A INPUT -p icmp -s 127.0.0.1 -j LOG_DROP
```

Figure 18 : Application du filtrage ICMP

```
[130 root@firewall ~]$ iptables -L -v
Chain INPUT (policy ACCEPT 2 packets, 142 bytes)
 pkts bytes target     prot opt in     out     source               destination

    2   198 LOG_DROP   icmp --  any    any     localhost            anywhere


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

    3   252 ACCEPT     all  --  any    any     192.168.20.0/24      anywhere


Chain OUTPUT (policy ACCEPT 4 packets, 340 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain LOG_DROP (1 references)
 pkts bytes target     prot opt in     out     source               destination

    2   198 LOG        all  --  any    any     anywhere             anywhere
         LOG level warning prefix "DROP_ICMP: "
    2   198 DROP       all  --  any    any     anywhere             anywhere

[0 root@firewall ~]$
```

Figure 19 : Vérification des règles

```
[0 root@firewall ~]$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
^C
--- 127.0.0.1 ping statistics ---
179 packets transmitted, 0 received, 100% packet loss, time 178680ms

[1 root@firewall ~]$
```

Figure 20 : Paquets ICMP bloqués

1.1.9.1. Suppression de la règle de filtrage ICMP

Une fois les tests réalisés, la règle de filtrage ICMP a été supprimée afin de rétablir le fonctionnement normal du système.

```
[1 root@firewall ~]$ iptables -D INPUT 1
[0 root@firewall ~]$
```

Figure 21 : Suppression de la règle de blocage ICMP

1.1.10. Blocage des pings depuis le réseau privé

Une règle a été ajoutée sur le firewall afin de bloquer les requêtes ICMP provenant de l'ensemble du réseau privé (192.168.20.0/24). Cette règle empêche les machines du réseau privé de pinger le firewall.

```
[0 root@firewall ~]$ iptables -A INPUT -p icmp -s 192.168.20.0/24 -j DROP
[0 root@firewall ~]$
```

Figure 22 : Blocage ICMP du réseau privé

```
[0 root@client1 ~]$ ping 192.168.20.254
PING 192.168.20.254 (192.168.20.254) 56(84) bytes of data.
^C
--- 192.168.20.254 ping statistics ---
9 packets transmitted, 0 received, 100% packet loss, time 8039ms

[1 root@client1 ~]$
```

Figure 23 : : Échec du ping vers firewall

```
[0 root@firewall ~]$ iptables -D INPUT -p icmp -s 192.168.20.0/24 -j DROP
[0 root@firewall ~]$
```

Figure 24 : Suppression de la règle de blocage

```
[1 root@client1 ~]$ ping 192.168.20.254
PING 192.168.20.254 (192.168.20.254) 56(84) bytes of data.
64 bytes from 192.168.20.254: icmp_req=1 ttl=64 time=0.866 ms
64 bytes from 192.168.20.254: icmp_req=2 ttl=64 time=0.599 ms
^C
--- 192.168.20.254 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.599/0.732/0.866/0.136 ms
[0 root@client1 ~]$
```

Figure 25 : Réussite du ping après la suppression de la règle

Tableau iptables

Commande	Syntaxe	Rôle
iptables -N test	-N	Créer une nouvelle chaîne utilisateur
iptables -X test	-X	Supprimer une chaîne utilisateur
iptables -P FORWARD DROP	-P	Définir la politique par défaut d'une chaîne
iptables -L INPUT	-L	Lister les règles d'une chaîne spécifique

iptables -L	-L	Lister toutes les chaînes/règles de la table par défaut
iptables -nL	-n	Afficher en numérique (pas de résolution DNS / ports)
iptables -F INPUT	-F	Vider toutes les règles d'une chaîne
iptables -F	-F	Vider toutes les règles de toutes les chaînes
iptables -A INPUT -s 0/0 -j DENY	-A	Ajouter une règle à la fin d'une chaîne. (-s = source ; -j = action).
iptables -I …	-I	Insérer une règle en tête (ou à une position donnée) dans une chaîne.
iptables -R …	-R	Remplacer une règle existante précise dans une chaîne.
iptables -D INPUT 1	-D	Supprimer une règle par son numéro dans la chaîne INPUT.
iptables -D INPUT -s 127.0.0.1 -p icmp -j DENY	-D	Supprimer une règle en redonnant exactement ses paramètres.

1.2. Partie 2 : ARP Spoofing

1.2.1. Présentation de l'environnement et objectif de l'attaque

L'objectif de cette partie est de comprendre et de mettre en œuvre une attaque de type ARP Spoofing, afin d'intercepter le trafic entre deux machines d'un réseau local. L'infrastructure utilisée est fournie par l'outil MI-LXC, qui permet de déployer un environnement réseau réaliste composé de plusieurs machines virtuelles représentant une petite entreprise.

Une fois l'infrastructure démarrée, seule la machine hacker est utilisée pour réaliser l'attaque.

1.2.2. Plan d'adressage du réseau cible

Les adresses IP et MAC de chaque machine ont été récupérées à l'aide des commandes réseau ip a, ip neigh, arp -a.

Nom de la machine	Adresse IP	Adresse MAC
target-router	eth1 : 100.80.0.1/16	2a:e1:e3:8c:e7:69
target-admin	100.80.0.4/16	ea:d1:cb:b2:29:3e
target-commercial	100.80.0.2/16	8e:e5:86:38:21:0f
target-dmz	100.80.1.2/16	d2:11:b0:d4:9c:fc
target-dev	100.80.0.3/16	e2:8e:16:f7:67:b7
target-ldap	100.80.0.10/16	1a:c2:2f:54:1e:19
target-filer	100.80.0.6/16	4a:9c:c3:82:d2:83
target-intranet	100.80.0.5/16	6a:67:03:b3:2d:dd

1.2.3. Rappel théorique : ARP Spoofing

Dans un réseau local, les machines communiquent à l'aide des adresses MAC, mais identifient les hôtes grâce aux adresses IP. Le protocole ARP permet de faire la correspondance entre une adresse IP et une adresse MAC.

Une attaque ARP Spoofing consiste à empoisonner le cache ARP d'une ou plusieurs victimes en leur faisant croire que l'adresse MAC de l'attaquant correspond à l'adresse IP d'une autre machine légitime (serveur par exemple).

Cette attaque permet de se positionner en homme du milieu (Man-in-the-Middle), rendant possible l'interception, l'analyse ou la modification du trafic réseau.

1.2.4. Vérification initiale des caches ARP

Avant toute attaque, les caches ARP des machines concernées sont vérifiés afin de s'assurer de leur état initial.

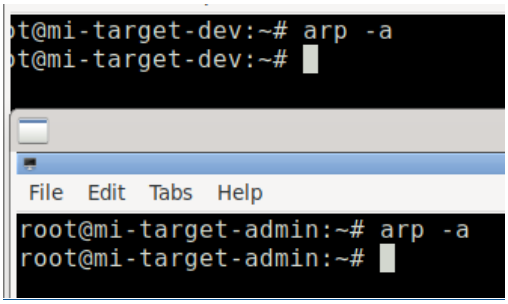


Figure 26 : État initial de la table ARP avant l'attaque

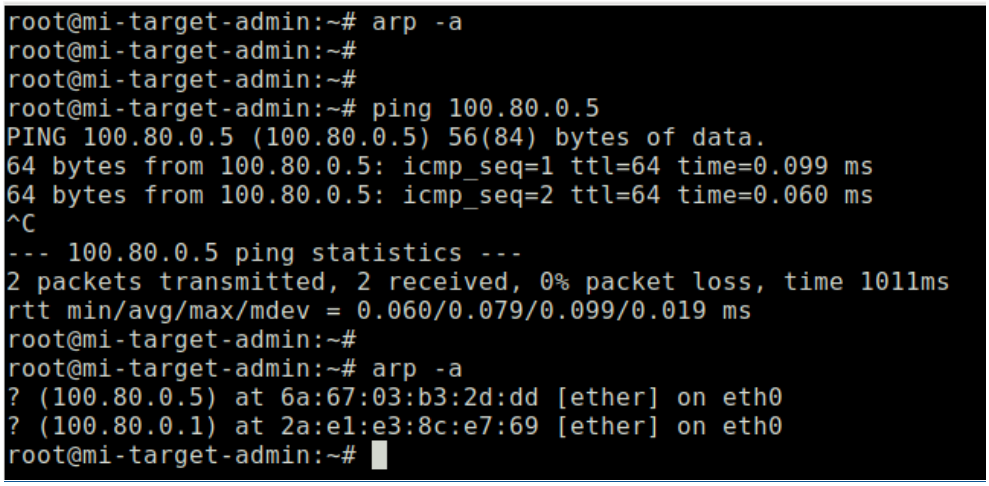


Figure 27 : Ping avec les adresses légitimes

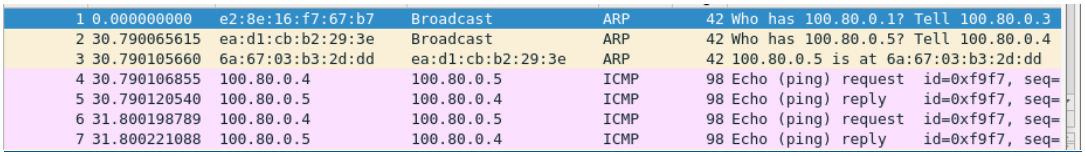


Figure 28 : Trames Wireshark légitime

1.2.5. Mise en œuvre de l'attaque ARP Spoofing

1.2.5.1. Principe de l'attaque réalisée

L'attaque consiste à faire croire à la machine target-admin (100.80.0.4) que la machine target-intranet (100.80.0.5) correspond à l'adresse MAC de l'attaquant (machine développeur/hacker).

Pour observer les changements en temps réel, la commande suivante est utilisée sur la machine administrateur :

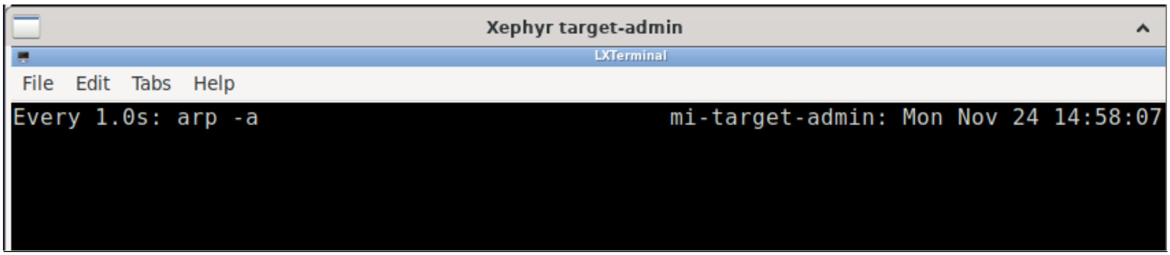


Figure 29 : Observation de la table ARP avant l'attaque

1.2.5.2. Lancement de l'attaque

L'outil arpspoof, est déjà installé sur la machine attaquante et est utilisé avec les paramètres suivants afin d'empoisonner la table ARP de la victime.

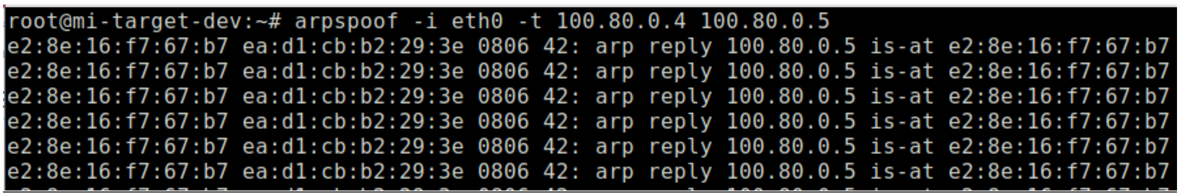


Figure 30 : Lancement de l'attaque ARP Spoofing depuis machine attaquante

1.2.6. Analyse du comportement de la table ARP

Dans un premier temps, la table ARP de la machine administrateur ne change pas. Cela s'explique par le fait que Linux conserve les entrées ARP tant qu'elles sont valides et qu'aucune nouvelle résolution ARP n'est nécessaire.

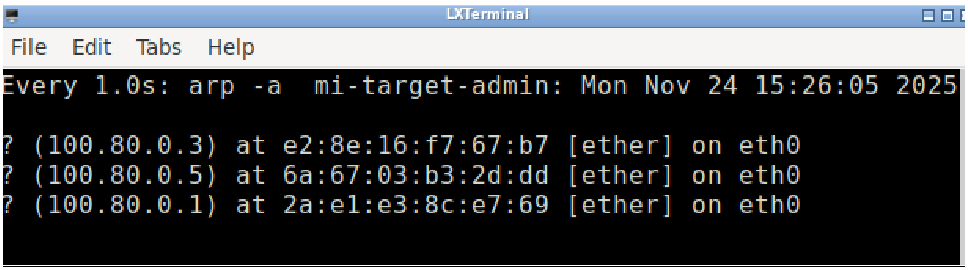


Figure 31 : Table ARP avant nouvelle résolution

Après l'arrêt de l'attaque et l'envoi d'une requête ICMP vers le serveur intranet, une nouvelle résolution ARP est déclenchée, ce qui provoque la mise à jour de la table ARP.

```
root@mi-target-admin:~# ping 100.80.0.5
PING 100.80.0.5 (100.80.0.5) 56(84) bytes of data.
64 bytes from 100.80.0.5: icmp_seq=1 ttl=64 time=0.076 ms
64 bytes from 100.80.0.5: icmp_seq=2 ttl=64 time=0.048 ms
64 bytes from 100.80.0.5: icmp_seq=3 ttl=64 time=0.272 ms
64 bytes from 100.80.0.5: icmp_seq=4 ttl=64 time=0.059 ms
^C
--- 100.80.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/mdev = 0.048/0.113/0.272/0.091 ms
root@mi-target-admin:~#
```

Figure 32 : Requête ICMP pour forcer la résolution

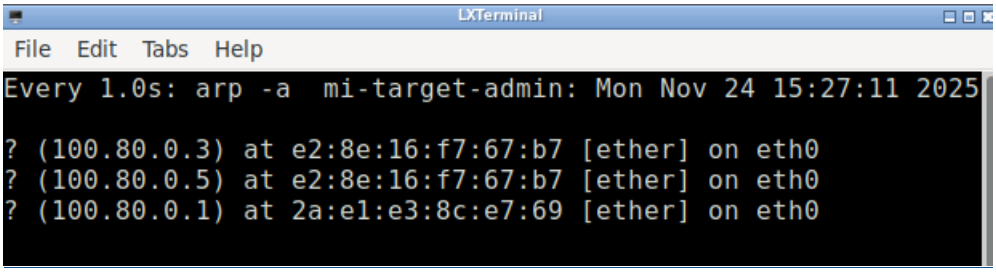


Figure 33 : Modification de la table ARP après nouvelle résolution

1.2.7. Vérification de l'interception du trafic ICMP

Une fois l'attaque relancée, il est possible de vérifier que la machine attaquante reçoit bien le trafic ICMP échangé entre la machine administrateur et le serveur intranet. Cela confirme que l'attaquant est correctement positionné en homme du milieu.

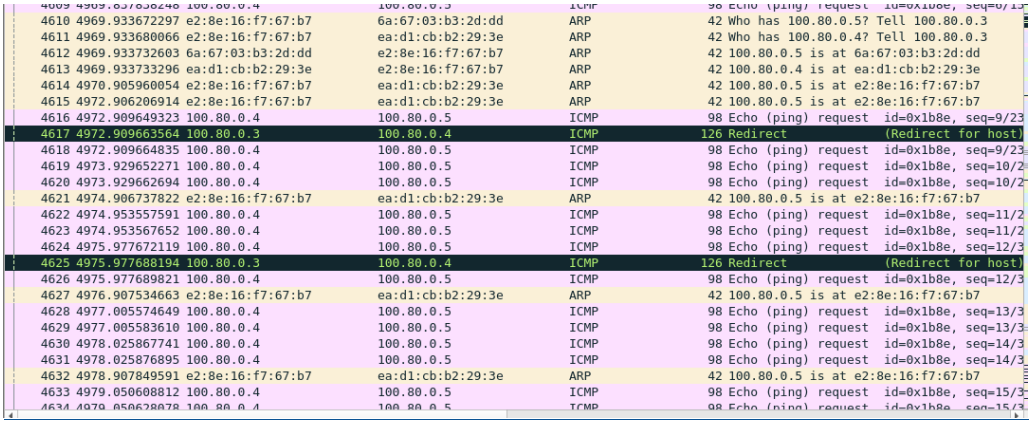


Figure 34 : Interception du trafic ICMP sur la machine attaquante après ARP Spoofing

1.2.8. Tentative d'interception du trafic HTTP avec urlsnarf

Une fois l'attaque ARP Spoofing opérationnelle et l'attaquant positionné en homme du milieu, une tentative d'interception du trafic applicatif est réalisée. Pour cela, l'outil urlsnarf est lancé sur la machine attaquante.

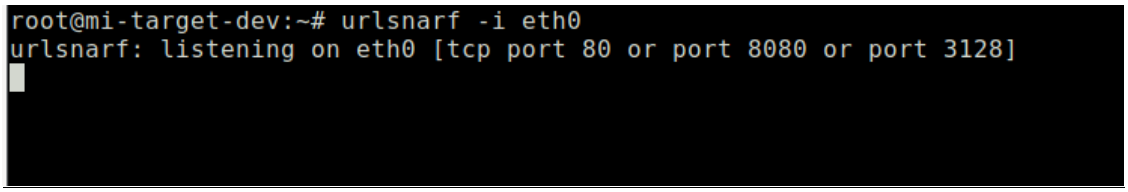


Figure 35 : Lancement d'urlsnarf sur la machine attaquante

Depuis la machine administrateur, une requête HTTP est effectuée à l'aide de la commande curl vers le serveur intranet.

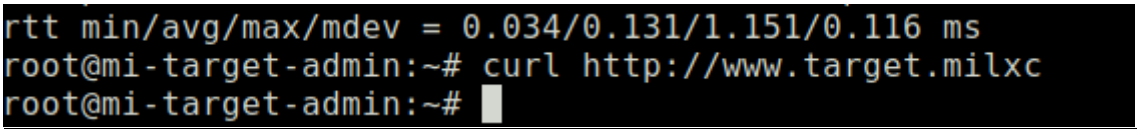


Figure 36 : Envoi d'une requête HTTP depuis la machine admin vers le serveur intranet

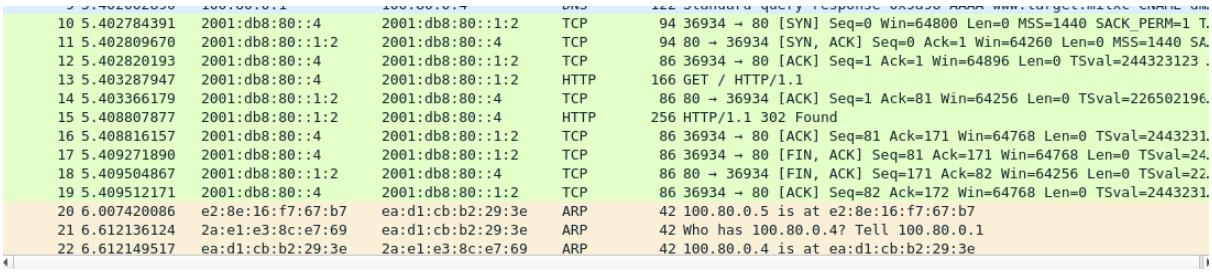
Aucun résultat n'apparaît dans la sortie de urlsnarf. Cela signifie que, malgré l'attaque ARP Spoofing, l'outil ne parvient pas à intercepter de requêtes HTTP exploitables.

1.2.9. Analyse des causes de l'échec de l'interception HTTP

L'absence de résultats avec urlsnarf peut s'expliquer par plusieurs raisons techniques :

- urlsnarf intercepte uniquement le trafic HTTP en clair
- la requête HTTP peut être redirigée automatiquement vers HTTPS (port 443)
- le nom de domaine peut ne pas pointer directement vers le serveur intranet
- le trafic HTTP ne transite pas encore correctement par la machine attaquante

Dans ce contexte, l'outil ne dispose donc d'aucune donnée pertinente à afficher.



10	5.402784391	2001:db8:80::4	2001:db8:80::1:2	TCP	94	36934 → 80 [SYN] Seq=0 Win=64800 Len=0 MSS=1440 SACK_PERM=1 T
11	5.402809670	2001:db8:80::1:2	2001:db8:80::4	TCP	94	80 → 36934 [SYN, ACK] Seq=0 Ack=1 Win=64260 Len=0 MSS=1440 SA
12	5.402820193	2001:db8:80::4	2001:db8:80::1:2	TCP	86	36934 → 80 [ACK] Seq=1 Ack=1 Win=64896 Len=0 TSval=244323123
13	5.403287947	2001:db8:80::4	2001:db8:80::1:2	HTTP	166	GET / HTTP/1.1
14	5.403366179	2001:db8:80::1:2	2001:db8:80::4	TCP	86	80 → 36934 [ACK] Seq=1 Ack=81 Win=64256 Len=0 TSval=226502196
15	5.408807877	2001:db8:80::1:2	2001:db8:80::4	HTTP	256	HTTP/1.1 302 Found
16	5.408816157	2001:db8:80::4	2001:db8:80::1:2	TCP	86	36934 → 80 [ACK] Seq=81 Ack=171 Win=64768 Len=0 TSval=2443231
17	5.409271890	2001:db8:80::4	2001:db8:80::1:2	TCP	86	36934 → 80 [FIN, ACK] Seq=81 Ack=171 Win=64768 Len=0 TSval=24
18	5.409504867	2001:db8:80::1:2	2001:db8:80::4	TCP	86	80 → 36934 [FIN, ACK] Seq=171 Ack=82 Win=64256 Len=0 TSval=22
19	5.409512171	2001:db8:80::4	2001:db8:80::1:2	TCP	86	36934 → 80 [ACK] Seq=82 Ack=172 Win=64768 Len=0 TSval=2443231
20	6.007420086	e2:8e:16:f7:67:b7	ea:d1:cb:b2:29:3e	ARP	42	100.80.0.5 is at e2:8e:16:f7:67:b7
21	6.612136124	2a:e1:e3:8c:e7:69	ea:d1:cb:b2:29:3e	ARP	42	Who has 100.80.0.4? Tell 100.80.0.1
22	6.612149517	ea:d1:cb:b2:29:3e	2a:e1:e3:8c:e7:69	ARP	42	100.80.0.4 is at ea:d1:cb:b2:29:3e

Figure 37 : Absence de trafic HTTP intercepté par urlsnarf malgré l'attaque ARP Spoofing

1.2.10. Analyse du trafic réseau avec tcpdump

Afin de mieux comprendre le comportement du réseau et de vérifier la présence effective de trafic, une capture plus bas niveau est réalisée à l'aide de l'outil tcpdump. Contrairement à urlsnarf, tcpdump permet d'observer l'ensemble des paquets transitant sur une interface réseau, indépendamment du protocole applicatif.

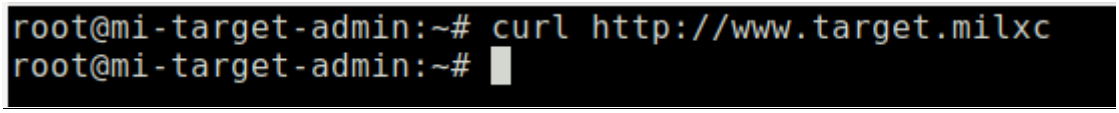


Figure 38 : Renvoi d'une requête HTTP depuis la machine admin vers le serveur intranet

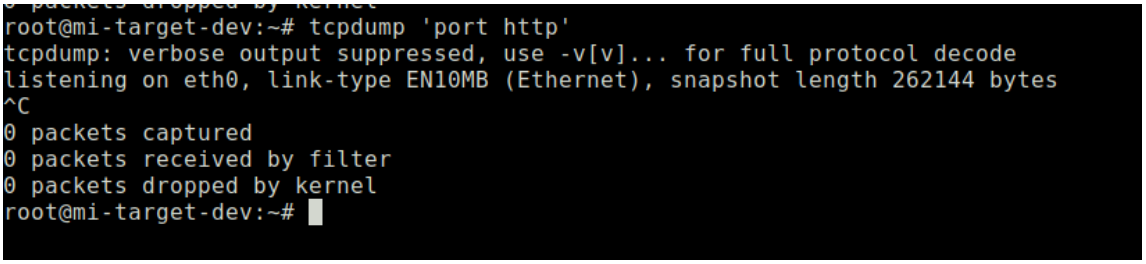


Figure 39 : Absence de paquets HTTP en clair lors de la capture tcpdump

L'analyse de cette capture montre la présence de paquets ARP, ICMP et TCP. Cela confirme que du trafic transite bien par la machine attaquante, mais que celui-ci n'est pas exploitable au niveau applicatif (HTTP en clair).

1.2.11. Ajout d'une interface réseau virtuelle sur la machine attaquante

Par défaut, le noyau Linux ne traite que les paquets dont l'adresse IP de destination correspond à une interface locale. Même si la machine attaquante est positionnée en homme du milieu grâce à l'ARP Spoofing, certains paquets peuvent ne pas être traités par la pile réseau.

Pour contourner cela, une interface réseau virtuelle est ajoutée sur la machine attaquante à l'aide de la commande ifconfig. Cette interface utilise la même adresse IP que la machine cible afin de forcer le noyau Linux à accepter les paquets détournés.

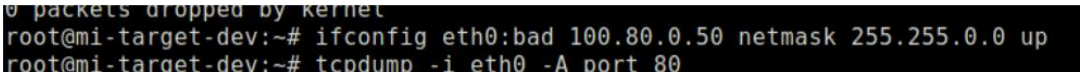


Figure 40 : Ajout d'une interface réseau virtuelle pour accepter le trafic détourné

1.2.12. Vérification de la configuration de l'interface virtuelle

Après l'ajout de l'interface virtuelle, la configuration réseau est vérifiée avec la commande ifconfig. L'interface eth0:bad est configurée avec l'adresse IP attendue.

```
root@mi-target-dev:~# ifconfig eth0:bad 100.80.0.50 netmask 255.255.0.0 up
root@mi-target-dev:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 100.80.0.3  netmask 255.255.0.0  broadcast 100.80.255.255
        inet6 2001:db8:80::3  prefixlen 48  scopeid 0x0<global>
        inet6 fe80::60c0:c6ff:fe42:ddb9  prefixlen 64  scopeid 0x20<link>
        ether 62:c0:c6:42:dd:b9  txqueuelen 1000  (Ethernet)
        RX packets 478  bytes 42970 (41.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 410  bytes 29469 (28.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0:bad: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 100.80.0.50  netmask 255.255.0.0  broadcast 100.80.255.255
        ether 62:c0:c6:42:dd:b9  txqueuelen 1000  (Ethernet)

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 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@mi-target-dev:~#
```

Figure 41: Vérification de la configuration de l'interface réseau virtuelle

1.2.13. Accès légitime au site web depuis la machine isp-a-home

Depuis la machine isp-a-home, un navigateur web est utilisé pour accéder à l'URL suivante :

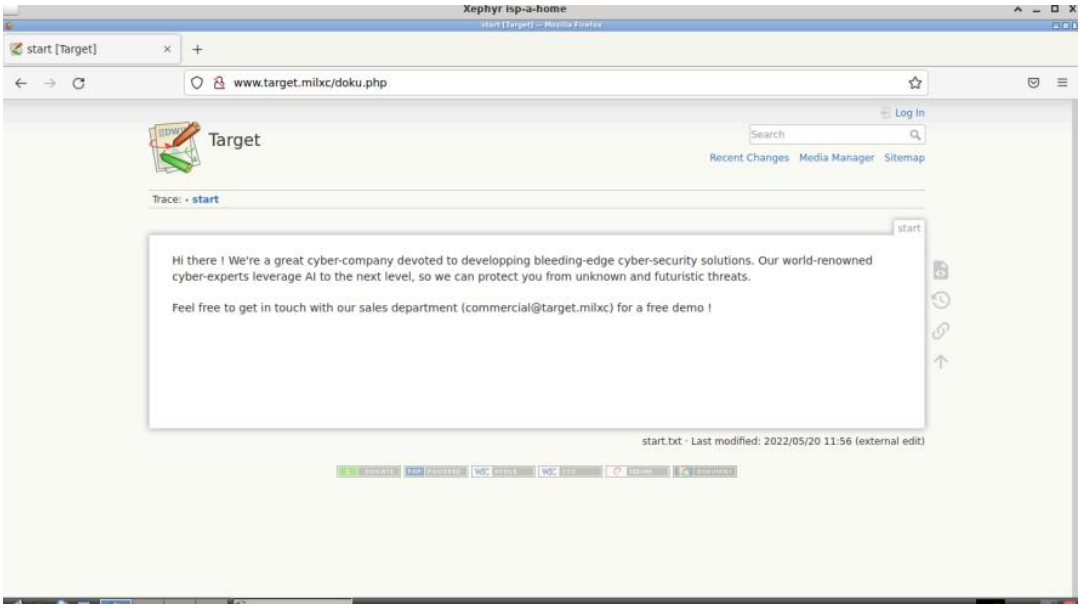


Figure 42 : Accès légitime au site www.target.milxc

1.2.14. Simulation d'une compromission du résolveur DNS

Nous supposons désormais que le résolveur DNS a été compromis. L'objectif est de rediriger l'utilisateur vers un site frauduleux lorsqu'il tente d'accéder à l'URL www.target.milxc.

Cette attaque est simulée en modifiant l'enregistrement DNS afin que le nom de domaine pointe vers la machine malveillante ecorp-infra. Pour simuler cette compromission, l'enregistrement DNS est modifié dans le fichier :

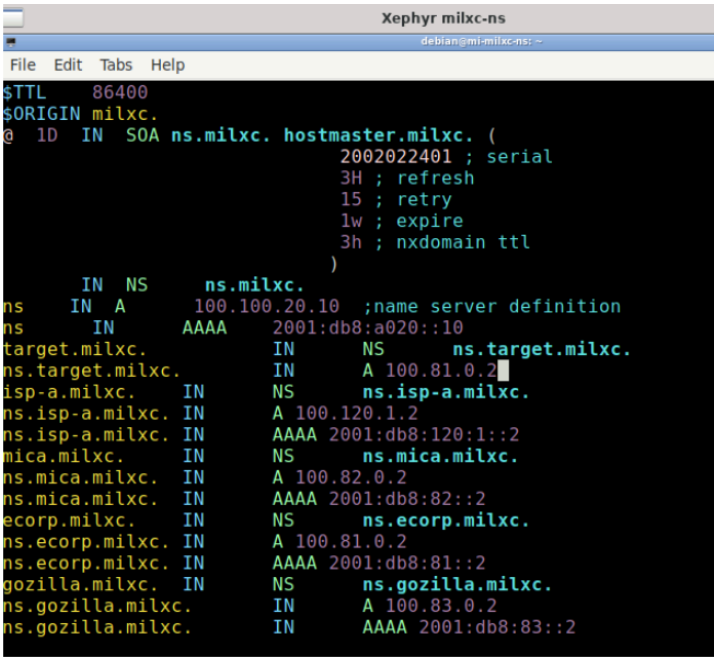


Figure 43 : Modification de l'enregistrement DNS A de www.target.milxc

1.2.15. Redémarrage du service DNS et vérification

Après modification de la zone DNS, le service est redémarré :

```
root@mi-milxc-ns:~# service nsd restart
root@mi-milxc-ns:~# dig @127.0.0.1 www.target.milxc

; <<>> DiG 9.16.27-Debian <<>> @127.0.0.1 www.target.milxc
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39604
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;www.target.milxc.              IN      A

;; AUTHORITY SECTION:
target.milxc.           86400   IN      NS      ns.target.milxc.

;; ADDITIONAL SECTION:
ns.target.milxc.        86400   IN      A       100.81.0.2

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Dec 03 14:10:55 CET 2025
;; MSG SIZE  rcvd: 78

root@mi-milxc-ns:~#
```

Figure 44 : Redémarrage du service DNS

1.2.16. Redirection transparente vers un site frauduleux

Depuis la machine isp-a-home, l'utilisateur tente à nouveau d'accéder à www.target.milxc. Le site affiché n'est plus la page DokuWiki légitime, mais un site frauduleux hébergé sur ecorp-infra.

Pour l'utilisateur, l'attaque est totalement transparente :

- l'URL affichée reste identique
- aucun message d'erreur ou d'avertissement n'apparaît
- la connexion utilise HTTP

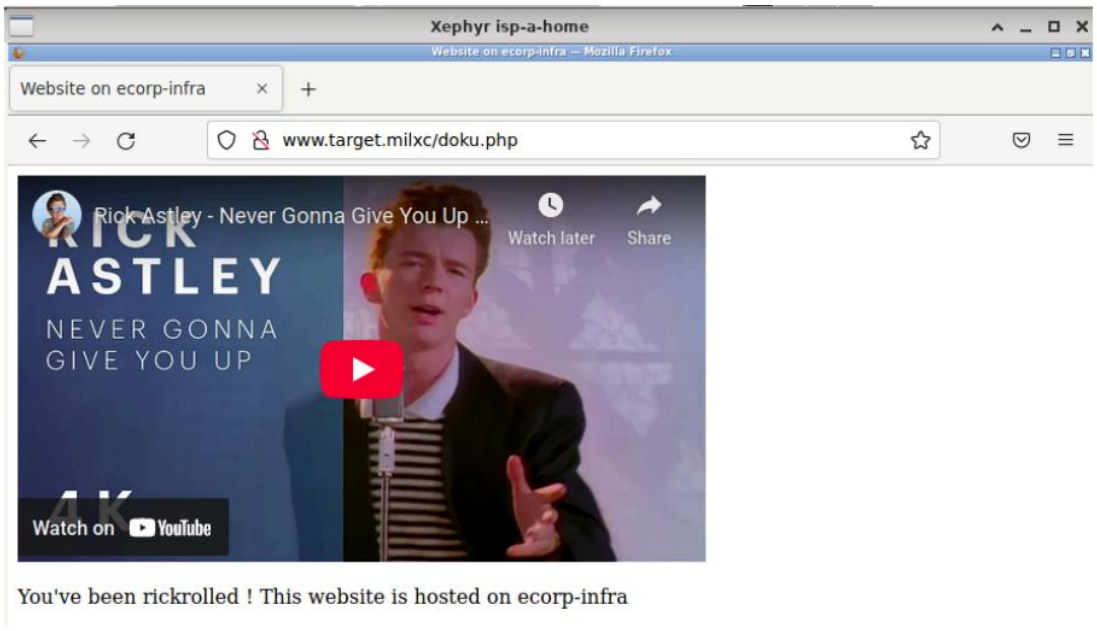


Figure 45 : Redirection transparente de user vers un site frauduleux

1.2.17. Remise en état du système

Afin de rétablir le fonctionnement normal du système et de poursuivre les manipulations dans un environnement sain, on effectue une remise en état du service DNS.

Sur la machine milxc-ns, le fichier de zone /etc/nsd/milxc.zone est de nouveau modifié afin que l'enregistrement A de www.target.milxc pointe vers l'adresse IP correcte du serveur légitime, à savoir 100.80.1.2.

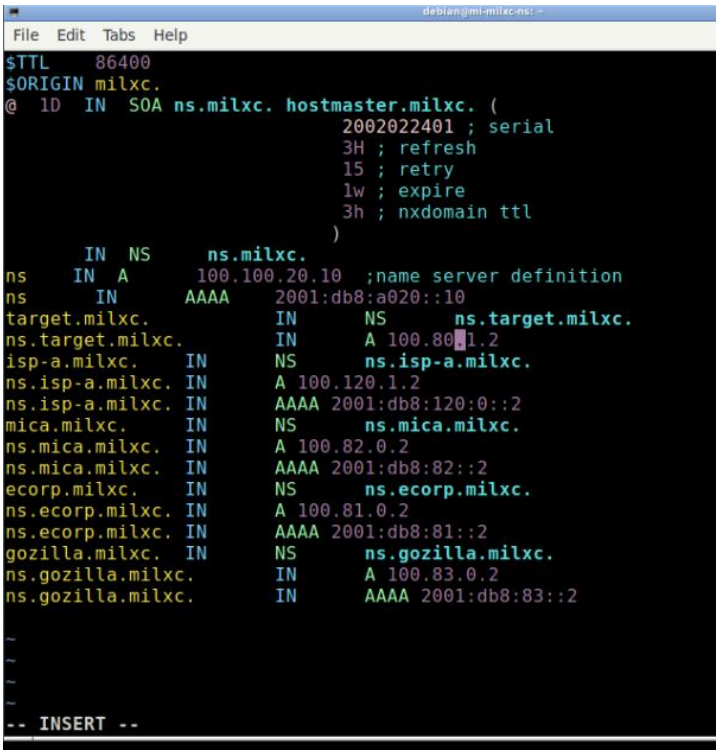


Figure 46 : Restauration de l'enregistrement DNS A de www.target.milxc

Une fois la correction appliquée, le service DNS est redémarré. La résolution DNS est vérifiée à l'aide de la commande dig, qui confirme que le nom www.target.milxc pointe maintenant vers l'adresse IP légitime du serveur cible.

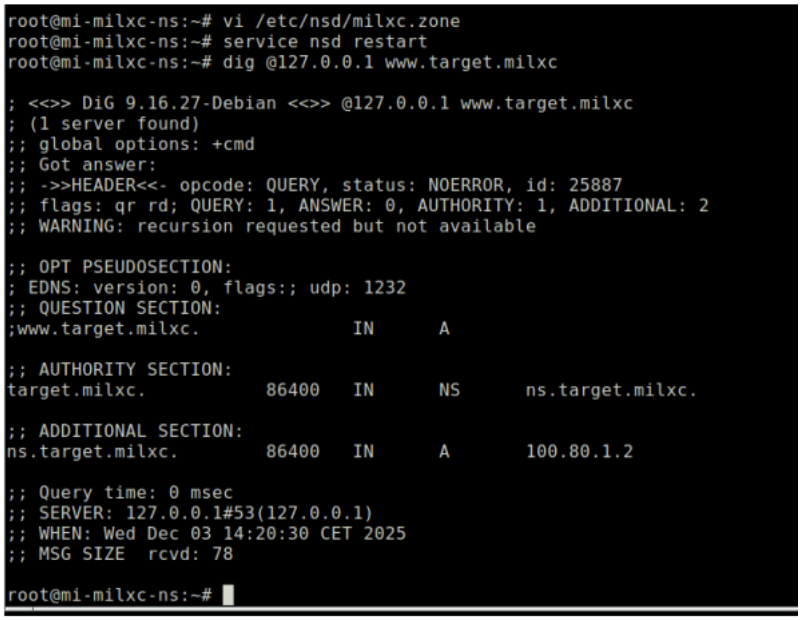


Figure 47 : Vérification de la résolution DNS après correction de la zone milxc

Pour finir, un nouvel accès au site web est effectué depuis la machine isp-a-home. La page affichée correspond à la page DokuWiki légitime, confirmant que le détournement DNS a bien été corrigé et que le système est revenu à un état de fonctionnement normal.

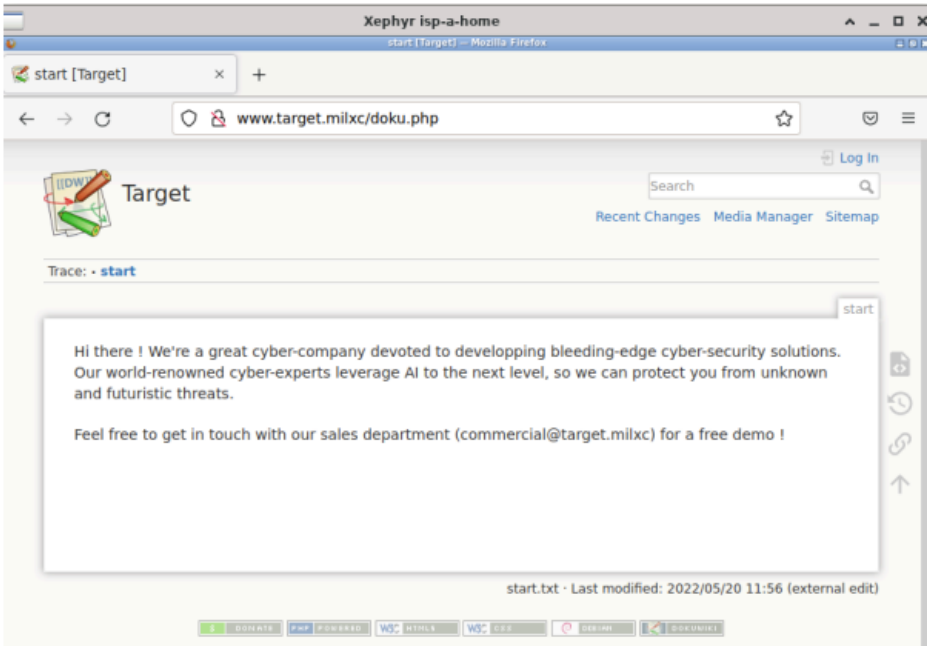


Figure 48 : Accès rétabli au site légitime www.target.milxc

Conclusion

Cette SAE a permis de mettre en évidence les mécanismes fondamentaux de sécurisation et de supervision d'un système d'information, ainsi que les vulnérabilités auxquelles un réseau peut être exposé en l'absence de protections adaptées.

La configuration du pare-feu et du NAT a montré l'importance du contrôle du trafic réseau et du cloisonnement entre réseaux privé et public. L'étude de l'attaque ARP Spoofing a permis de comprendre comment un attaquant peut se positionner en homme du milieu et intercepter des échanges, même dans un réseau commuté.

Les expérimentations ont également mis en lumière les limites de ce type d'attaque face au chiffrement HTTPS, ainsi que les risques liés à une compromission du service DNS, capable de rediriger un utilisateur vers un site frauduleux de manière totalement transparente.

Enfin, ce travail souligne l'importance de la détection, de la supervision et de la remise en état des systèmes après incident, qui sont des éléments essentiels dans la cybersécurité.