

## Remerciement

Nous tenons à exprimer notre profonde gratitude envers M. Evangelista pour son dévouement exceptionnel envers ses élèves, tant sur Discord que par e-mail. Nous avons grandement apprécié ce deuxième semestre, en grande partie grâce à M. Evangelista et à sa gestion exemplaire de ses modules.

## *Table des matières*

<b>1. Introduction.....</b>	<b>1</b>
<b>2. Environnement de travail.....</b>	<b>1</b>
<b>3. Partie 1 : Attaque ARP spoofing avec le paquet mitm.....</b>	<b>2</b>
3.1 Réalisation.....	2
3.2 Tests.....	2
<b>4. Partie 2 : Attaque man-in-the-middle sur SSH.....</b>	<b>2</b>
<b>5. Partie 3 : Sécurisation du réseau local.....</b>	<b>2</b>

## 1. Introduction

Cette SAÉ24 a été mis en place par M. Sami Evangelista dans le but de nous initier à la cybersécurité. L'objectif était de réaliser une attaque MITM, également connue sous le nom de "man in the middle" (l'homme du milieu). Pour mener à bien cette SAÉ, nous devons mettre en pratique nos compétences en python, en réseau et, dans notre cas, en HTML acquises lors de notre première année.

L'objectif de la SAÉ24 est de programmer et d'expérimenter des attaques de type Man In The Middle (MITM) sur un réseau local et d'étudier les mesures de protection pour les contrer. Cet exercice vise à comprendre les mécanismes des attaques réseau pour mieux s'en protéger, tout en s'assurant de les réaliser uniquement dans un cadre contrôlé comme une salle de TP ou des machines virtuelles confinées.

La SAÉ24 se compose de trois parties. Dans la première, nous programmerons en Python et expérimenterons l'attaque ARP spoofing, permettant à un attaquant d'intercepter les paquets échangés entre un client et un serveur en empoisonnant leur cache ARP. La deuxième partie portera sur une attaque MITM sur le protocole SSH, où l'attaquant usurpe l'identité du serveur auprès du client pour intercepter des communications sécurisées. Enfin, la troisième partie consistera à étudier et expérimenter des techniques de protection intégrées aux équipements CISCO, telles que les listes de contrôle d'accès (ACL) et la protection contre l'empoisonnement ARP, pour bloquer ces attaques sur un réseau local.

Ces trois parties visent à fournir une compréhension pratique des attaques MITM et des stratégies de défense efficaces, en mettant l'accent sur des expérimentations concrètes dans un environnement sécurisé et contrôlé.

## 2. Environnement de travail

Nous avons configuré 3 machines : une qui jouera le rôle de l'attaquant, une autre machine qui jouera le rôle d'un serveur et une troisième machine qui jouera le rôle d'un client.

Le code Python est placé et exécuté sur la machine de l'attaquant. Dans la première partie, l'attaquant espionnera les connexions TCP et les demandes d'écho échangées entre le client et le serveur. Dans la deuxième partie, l'attaquant usurpe l'identité du serveur auprès du client lors d'une connexion SSH. Les deux premières parties peuvent être réalisées dans un environnement virtuel (machines virtuelles) ou physique (PC des salles de TP). Pour la troisième partie, vous devrez travailler sur des équipements physiques (PC et switch Cisco).

### 3. Partie 1 : Attaque ARP spoofing avec le paquet mitm

Une attaque ARP (Address Resolution Protocol) est une technique où un attaquant envoie des messages ARP falsifiés sur un réseau local pour associer son adresse MAC à l'adresse IP d'une autre machine, comme un routeur. Cela permet à l'attaquant de rediriger le trafic destiné à cette machine vers sa propre machine, réalisant ainsi une attaque de type Man-In-The-Middle (MITM).

L'attaquant envoie des réponses ARP non sollicitées aux autres machines du réseau, qui mettent à jour leur cache ARP avec les informations falsifiées. Tout le trafic destiné à l'adresse IP cible est alors envoyé à l'attaquant, qui peut intercepter, lire, modifier ou rediriger les communications.

Les objectifs des attaques ARP incluent le vol de données sensibles, l'usurpation d'identité et la perturbation des communications réseau. Pour se protéger contre ces attaques, il est recommandé d'utiliser des protocoles sécurisés comme HTTPS, de surveiller le réseau pour détecter des comportements ARP suspects, de configurer des entrées ARP statiques pour les machines critiques, de segmenter le réseau avec des VLANs et d'utiliser des systèmes de prévention d'intrusion (IPS) et des pare-feux.

Les attaques ARP exploitent une faiblesse du protocole ARP pour manipuler le trafic réseau, constituant une menace sérieuse pour la sécurité des réseaux locaux.

Une attaque de type Man-In-The-Middle (MITM) est une méthode où un attaquant intercepte et modifie la communication entre deux parties sans qu'elles ne s'en aperçoivent. L'attaquant peut écouter, altérer ou insérer des informations dans les messages échangés, souvent en utilisant des réseaux non sécurisés ou en redirigeant le trafic réseau.

Les principaux objectifs des attaques MITM incluent le vol de données sensibles (identifiants, mots de passe, numéros de carte de crédit), l'usurpation d'identité, la fraude (modification des transactions financières) et l'espionnage des communications.

Il existe plusieurs types d'attaques MITM, comme le sniffing (écoute du trafic non crypté), le session hijacking (vol des cookies de session), l'IP spoofing (usurpation d'adresse IP), le DNS spoofing (redirection vers des sites malveillants) et le SSL stripping (transformation des connexions HTTPS en HTTP).

Pour se protéger contre ces attaques, il est recommandé d'utiliser HTTPS, de vérifier les certificats SSL/TLS, d'utiliser des VPN pour sécuriser les connexions, d'activer

l'authentification à deux facteurs (2FA), de maintenir les systèmes et logiciels à jour et de sensibiliser les utilisateurs aux bonnes pratiques de sécurité.

Les attaques MITM représentent une menace sérieuse pour la sécurité des communications en ligne, et il est crucial de prendre des mesures pour se protéger.

Le paquet mitm contient le code de l'attaque d'empoisonnement ARP ainsi que du code permettant d'observer les connexions TCP et les paquets ICMP échangés entre le client et le serveur une fois l'attaque lancée.

### 3.1 Réalisation

- `__init__.py` : Code d'initialisation du paquet. Il définit un numéro de version du paquet et affiche un message de bienvenue contenant ce numéro.

Ce module définit un numéro de version pour le paquet et affiche un message de bienvenue avec ce numéro de version, similaire au paquet `biblio` écrit en R208.

- `arp.py` : Contient le code de l'attaque ARP.

Cette fonction réalise l'attaque d'ARP spoofing. L'attaquant capture tous les paquets échangés entre les deux hôtes dont les IP sont passées en paramètre (`ipa` et `ipb`). Le paramètre `delay` est le délai entre l'envoi de paquets d'attaque, en secondes. Utilisez la fonction `sleep` du module `time` pour introduire un délai dans le programme.

Pour éviter que Scapy n'envoie pas les paquets sur la bonne interface, précisez explicitement l'interface d'envoi en appelant `send`, `sendp`, `sr1` ou `srp1` avec le paramètre `iface` ayant comme valeur le nom de l'interface sur laquelle on veut envoyer le paquet.

- `capture.py` : Contient le code permettant d'analyser les paquets capturés.

Cette fonction capture pendant `timeout` secondes les paquets allant de `ip_client` vers `ip_server`. Elle affiche dans le terminal :

Une ligne pour chaque demande de connexion TCP capturée.

Une ligne pour chaque demande d'écho capturée.

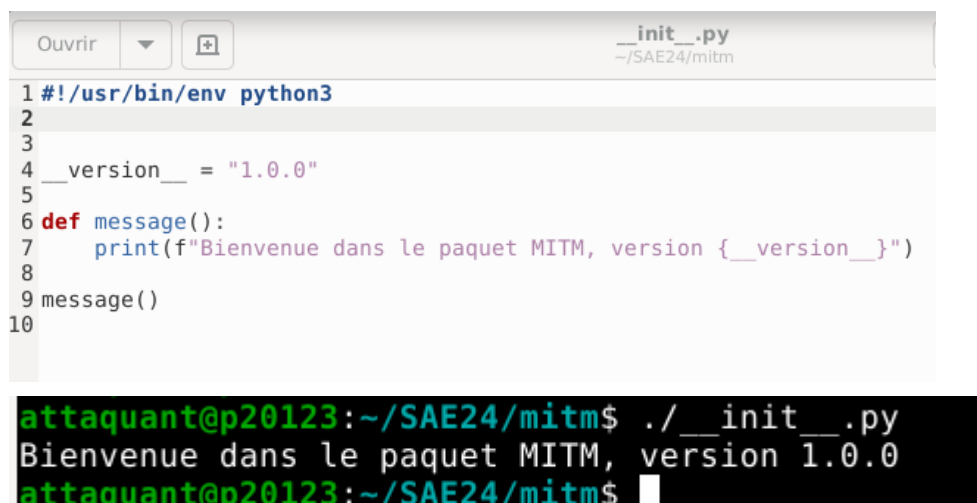
Les deux premières lignes sont affichées suite à la capture de deux paquets TCP de demande de connexion des ports 23443 et 12522 vers les ports 80 et 22 respectivement. Les IP 10.1.0.1 et 10.1.0.100 sont respectivement les IP du client (`ip_client`) et du

serveur (`ip_server`). La troisième ligne est affichée suite à la capture d'une demande d'écho de 10.1.0.1 vers 10.1.0.100.

Les données affichées dans le terminal seront également sauvegardées au format JSON dans le fichier passé en paramètre (`output`). Les données JSON consisteront en une liste de dictionnaires. Chaque dictionnaire de la liste correspondra à un paquet capturé et aura trois clés : `prot` pour le protocole (TCP ou ICMP), `src` pour la source et `dst` pour la destination. Par exemple, pour l'affichage donné plus haut, le fichier aura le contenu ci-dessous :

- `setup.py` : Script d'installation du paquet

Il contient le code d'installation du paquet `mitm`, similaire à celui écrit en R208. Notez que le paquet `mitm` ne contient pas de script exécutable, donc il est inutile de passer l'argument `scripts` à la fonction `setup`.



```
Ouvrir  [icon]  __init__.py
~/SAE24/mitm

1 #!/usr/bin/env python3
2
3
4 __version__ = "1.0.0"
5
6 def message():
7     print(f"Bienvenue dans le paquet MITM, version {__version__}")
8
9 message()
10
```

```
attaquant@p20123:~/SAE24/mitm$ ./__init__.py
Bienvenue dans le paquet MITM, version 1.0.0
attaquant@p20123:~/SAE24/mitm$
```

client 10.13.0.1/24 : Adressage IP

```

client@p20122:~$ sudo ip addr add 10.13.0.1/24 dev eth0
client@p20122:~$ sudo ip link set dev eth0 up
client@p20122:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 40:a6:b7:ae:06:21 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.13.0.1/24 scope global eth0
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether bc:0f:f3:cb:93:cb brd ff:ff:ff:ff:ff:ff
    altname eno1
    altname enp0s31f6
    inet 192.168.51.22/24 brd 192.168.51.255 scope global dynamic noprefixroute eth1
        valid_lft 263sec preferred_lft 263sec
    inet6 fe80::be0f:f3ff:fe0b:93cb/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

```

Attaquant 10.13.0.2/24 Adressage IP :

```

attaquant@p20123:~/SAE24$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 40:a6:b7:ae:0a:86 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.13.0.2/24 scope global eth0
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether bc:0f:f3:cb:93:e0 brd ff:ff:ff:ff:ff:ff
    altname eno1
    altname enp0s31f6
    inet 192.168.51.23/24 brd 192.168.51.255 scope global dynamic noprefixroute eth1
        valid_lft 210sec preferred_lft 210sec
    inet6 fe80::be0f:f3ff:fe0b:93e0/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
attaquant@p20123:~/SAE24$

```

serveur 10.13.0.254 :

```

serveur@p20124:~$ sudo ip addr add 10.13.0.254/24 dev eth0
serveur@p20124:~$ sudo ip link set dev eth0 up
serveur@p20124:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 40:a6:b7:ae:05:a6 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.13.0.254/24 scope global eth0
        valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether bc:0f:f3:cb:93:cf brd ff:ff:ff:ff:ff:ff
    altname eno1
    inet 192.168.51.24/24 brd 192.168.51.255 scope global dynamic noprefixroute eth1
        valid_lft 187sec preferred_lft 187sec
    inet6 fe80::be0f:f3ff:fe3b:93cf/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

```

\*ETH0 est bien uP dans client

Pc attaquant (fichier arp.py fonctionne)

```

root@p20123:/home/attaquant/SAE24/mitm# ./arp.py
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
^CARP Spoofing stopped.
root@p20123:/home/attaquant/SAE24/mitm#

```

```

root@p20122:/home/client# arp
Adresse                TypeMap AdresseMat          Indicateurs          Iface
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c      C                     eth1
10.13.0.2               ether  40:a6:b7:ae:0a:86      C                     eth0
10.13.0.254             ether  40:a6:b7:ae:0a:86      C                     eth0
root@p20122:/home/client# sudo arp -d 10.13.0.2
root@p20122:/home/client# arp
Adresse                TypeMap AdresseMat          Indicateurs          Iface
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c      C                     eth1
10.13.0.254            ether  40:a6:b7:ae:0a:86      C                     eth0
root@p20122:/home/client#

```



## 3.2 Tests

On lance arp.py sur le terminal python3 :

```
root@p20123:/home/attaquant/SAE24# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from mitm import arp
Bienvenue dans le paquet MITM version1.0.0
>>> arp.arp("10.13.0.1", "10.13.0.254")
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
```

Puis on lance capture.py sur le terminal python3 :

```
root@p20123:/home/attaquant/SAE24# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from mitm import capture
Bienvenue dans le paquet MITM version1.0.0
```

On effectue ensuite des ping et des tentatives de connctitions ssh pour les capturer à l'aide de capture.py :

```
root@p20122:/home/client# ping 10.13.0.254
PING 10.13.0.254 (10.13.0.254) 56(84) bytes of data.
^C
--- 10.13.0.254 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4103ms

root@p20122:/home/client# ssh server@10.13.0.254
^C
root@p20122:/home/client# █
```

Voici le résultat sur le terminal ou capture.py a été lancé :

```
root@p20123:/home/attaquant/SAE24# python3
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from mitm import capture
Bienvenue dans le paquet MITM version1.0.0
>>> capture.tcp_icmp("10.13.0.1", "10.13.0.254")
ICMP 10.13.0.1 -> 10.13.0.254
ICMP 10.13.0.1 -> 10.13.0.254
ICMP 10.13.0.1 -> 10.13.0.254
ICMP 10.13.0.1 -> 10.13.0.254
ICMP 10.13.0.1 -> 10.13.0.254
TCP 10.13.0.1:33180 -> 10.13.0.254:22
TCP 10.13.0.1:33180 -> 10.13.0.254:22
Captured packets saved to capture.json
>>>
```

Ainsi que sur le fichier capture.json :



```
1 [{"prot": "ICMP", "src": "10.13.0.1", "dst": "10.13.0.254"}, {"prot": "ICMP", "src": "10.13.0.1", "dst": "10.13.0.254"}, {"prot": "ICMP", "src": "10.13.0.1", "dst": "10.13.0.254"}, {"prot": "ICMP", "src": "10.13.0.1", "dst": "10.13.0.254"}, {"prot": "ICMP", "src": "10.13.0.1", "dst": "10.13.0.254"}, {"prot": "TCP", "src": "10.13.0.1:33180", "dst": "10.13.0.254:22"}]
```

\*on peut voir qu'avant le ping la table arp est bien vide:

```
root@p20122:/home/client# arp
root@p20122:/home/client#
```

Et après le ping la table est rempli :

```
root@p20122:/home/client# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c  C                eth1
10.13.0.2               ether  40:a6:b7:ae:0a:86  C                eth0
10.13.0.254             ether  40:a6:b7:ae:0a:86  C                eth0
root@p20122:/home/client# sudo arp -d 10.13.0.2
root@p20122:/home/client# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c  C                eth1
10.13.0.254             ether  40:a6:b7:ae:0a:86  C                eth0
root@p20122:/home/client#
```

On regarde la table arp du serveur avant l'attaque :

```

root@p20124:/home/serveur# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c    C                eth1

```

On regarde la table arp su serveur après l'attaque :

```

root@p20124:/home/serveur# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
10.13.0.1              ether  40:a6:b7:ae:0a:86    C                eth0
clonez-p201.iutv.univ-p ether  14:18:77:4c:87:0c    C                eth1
10.13.0.2              ether  40:a6:b7:ae:0a:86    C                eth0
root@p20124:/home/serveur# █

```

## 4. Partie 2 : Attaque man-in-the-middle sur SSH

La sécurité des communications réseau est une préoccupation majeure dans le domaine de l'informatique, en particulier lorsqu'il s'agit de protéger les échanges sensibles, tels que les connexions SSH (Secure Shell). Dans cette deuxième partie, nous aborderons une attaque bien connue et redoutée : l'attaque man-in-the-middle (MITM) sur SSH.

L'attaque MITM sur SSH exploite les faiblesses du protocole de résolution d'adresses ARP (Address Resolution Protocol) pour intercepter et manipuler les communications entre un client et un serveur SSH. En usurpant l'identité du serveur auprès du client et vice versa, l'attaquant parvient à déchiffrer toutes les données échangées, y compris les mots de passe.

Dans cette partie, nous explorerons les étapes nécessaires pour réaliser cette attaque. Nous commencerons par mettre en place une attaque ARP spoofing pour empoisonner les caches ARP du client et du serveur, puis nous utiliserons l'outil `ssh-mitm` pour intercepter et manipuler les connexions SSH. Enfin, nous observerons les données capturées, démontrant ainsi la vulnérabilité des communications SSH face à cette attaque sophistiquée.

Cette démonstration mettra en lumière l'importance de la sécurisation des communications réseau et soulignera la nécessité pour les administrateurs système de prendre des mesures proactives pour protéger leurs infrastructures contre de telles menaces.

```

root@p20122:/home/client# ssh serveur@10.13.0.254
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
SHA256:yAeolfl03kTgN3UTIYfKFcE4wisbAuL0o/ohYSimcLo.
Please contact your system administrator.
Add correct host key in /root/.ssh/known_hosts to get rid of this message.
Offending RSA key in /root/.ssh/known_hosts:2
  remove with:
  ssh-keygen -f "/root/.ssh/known_hosts" -R "10.13.0.254"
RSA host key for 10.13.0.254 has changed and you have requested strict checking.
Host key verification failed.
root@p20122:/home/client# █

```

## Sur l'attaquant :

```

root@p20123:/home/attaquant/SAE24# sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-ports 10022
root@p20123:/home/attaquant/SAE24# sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination
REDIRECT   tcp  --  anywhere               anywhere            tcp dpt:ssh redir ports 10022
REDIRECT   tcp  --  anywhere               anywhere            tcp dpt:ssh redir ports 10022

Chain INPUT (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source                destination
root@p20123:/home/attaquant/SAE24#

```

```

attaquant@p20123:~/SAE24$ sudo python3
[sudo] Mot de passe de attaquant :
Python 3.9.2 (default, Feb 28 2021, 17:03:44)
[GCC 10.2.1 20210110] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from mitm import arp
Bienvenue dans le paquet MITM version1.0.0
>>> arp.arp("10.13.0.1", "10.13.0.254")
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets

```

```

root@p20123:/home/attaquant# ssh-mitm server --remote-host 10.13.0.254
SSH-MITM - ssh audits made simple
Documentation: https://docs.ssh-mitm.at
Issues: https://github.com/ssh-mitm/ssh-mitm/issues
Configuration
SSH-Host-Keys:
generated temporary RSAKey key with 2048 bit length
MD5:6e:3d:ae:bd:61:33:1f:da:4c:4d:32:58:22:80:cd:48
SHA256:yAeoLfL03kTgN3UTIYfKFcE4wisbAuL0o/ohYSimcLo
SHA512:3nVVREwZ/GE0BeSyyEfQ+mPKjmMo0WIW4devNzVn2aM3Hef+yKm7FLUZex0CDKLQcYIVbdTQMgkbjt17kw066g
.....
listen interfaces 0.0.0.0 and :: on port 10022
waiting for connections
[06/03/24 17:05:35] INFO i session 287a60df-ef8c-47e1-a2d7-9f1a79adcf9d created
INFO i client information:
- client version: ssh-2.0-openssh_8.4p1 debian-5+deb11u3
- product name: OpenSSH
- vendor url: https://www.openssh.com/
- client address: ip::ffff:10.13.0.1 port=39766
Δ client affected by CVEs:
* CVE-2021-28041: https://nvd.nist.gov/vuln/detail/CVE-2021-28041
* CVE-2020-14145: https://docs.ssh-mitm.at/vulnerabilities/CVE-2020-14145.html
Δ client audit tests:
* client has a locally cached remote fingerprint.
* Preferred server host key algorithm: rsa-sha2-512-cert-v01@openssh.com
WARNING (('::ffff:10.13.0.1', 39766, 0, 0)->('::ffff:10.13.0.2', 10022, 0, 0))
session not started
INFO i session 287a60df-ef8c-47e1-a2d7-9f1a79adcf9d closed

```

## 5. Partie 3 : Sécurisation du réseau local

```

+-----+
| A -      Port série : /dev/ttyUSB0
| B - Emplacement fichier verr. : /var/lock
| C -      Prog. d'appel entrant :
| D -      Prog. d'appel sortant :
| E -      Débit/Parité/Bits : 9600 8N1
| F - Contrôle de flux matériel : Oui
| G - Contrôle de flux logiciel : Non
| H -      RS485 Enable      : No
| I -      RS485 Rts On Send  : No
| J -      RS485 Rts After Send : No
| K -      RS485 Rx During Tx  : No
| L -      RS485 Terminate Bus : No
| M - RS485 Delay Rts Before: 0
| N - RS485 Delay Rts After : 0
|
|      Changer quel réglage ? █
+-----+

```

### --- System Configuration Dialog ---

Enable secret warning

```

-----
In order to access the device manager, an enable secret is required
If you enter the initial configuration dialog, you will be prompted for the enat
If you choose not to enter the initial configuration dialog, or if you exit setu,
please set an enable secret using the following CLI in configuration mode-
enable secret 0 <cleartext password>
-----

```

Would you like to enter the initial configuration dialog? [yes/no]: no

Switch>

Switch>

\*Jun 4 08:17:47.211: %PNP-6-PNP\_SAVING\_TECH\_SUMMARY: Saving PnP tech summary (.

\*Jun 4 08:17:48.477: %PNP-6-PNP\_TECH\_SUMMARY\_SAVED\_OK: PnP tech summary (/pnp-.

\*Jun 4 08:17:48.477: %PNP-6-PNP\_DISCOVERY\_STOPPED: PnP Discovery stopped (Conf)

Switch>

Switch>

Switch>

Switch>

Switch>

Switch>█

Switch#config

Configuring from terminal, memory, or network [terminal]?

Enter configuration commands, one per line. End with CNTL/Z.

Switch(config)#interface vlan 1

Switch(config-if)#ip address 10.13.0.254 255.255.255.0

Switch(config-if)#no sh

Switch(config-if)#exit

Switch(config)#ip dhcp pool P00L

Switch(dhcp-config)#network 10.13.0.0 255.255.255.0

Switch(dhcp-config)#default-router 10.13.0.254

Switch(dhcp-config)#exit

Switch(config)#

```
Switch(config)#
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan 1
Switch(config)#
```

```
Switch(config)#interface GigabitEthernet0/1
```

```
% Invalid input detected at '^' marker.
```

```
Switch(config)#interface GigabitEthernet1/0/1
Switch(config-if)#ip dhcp snooping
% Incomplete command.
```

```
Switch(config-if)#ip dhcp snooping trust
Switch(config-if)#
```

```
Switch(config)#
Switch(config)#ip arp inspection vlan 1
Switch(config)#
```

```
Switch#
Switch#show ip dhcp snooping
Switch DHCP snooping is enabled
Switch DHCP gleaning is disabled
DHCP snooping is configured on following VLANs:
1
DHCP snooping is operational on following VLANs:
1
DHCP snooping is configured on the following L3 Interfaces:
```

```
Insertion of option 82 is enabled
  circuit-id default format: vlan-mod-port
  remote-id: 6c4e.f66b.dc80 (MAC)
Option 82 on untrusted port is not allowed
Verification of hwaddr field is enabled
Verification of giaddr field is enabled
DHCP snooping trust/rate is configured on the following Interfaces:
```

Interface	Trusted	Allow option	Rate limit (pps)
GigabitEthernet1/0/1	yes	yes	unlimited

```
Custom circuit-ids:
Switch#
```

```

Switch#
Switch#show ip arp inspection

Source Mac Validation      : Disabled
Destination Mac Validation : Disabled
IP Address Validation      : Disabled

Vlan      Configuration      Operation      ACL Match      Static ACL
-----
  1      Enabled              Active

Vlan      ACL Logging        DHCP Logging    Probe Logging
-----
  1      Deny                Deny            Off

Vlan      Forwarded          Dropped        DHCP Drops      ACL Drops
-----
  1              0              0              0              0

Vlan      DHCP Permits      ACL Permits    Probe Permits    Source MAC Failures
-----
  1              0              0              0              0

Vlan      Dest MAC Failures    IP Validation Failures    Invalid Protocol Data
-----
--More-- █

```

```

+-----+---[Paramètres de communication]---+-----+
| A -   |                                         |           | |
| B - Emplacement | Actuelle: 9600 8N1 |           |
| C -   Prog. d' | Vitesse           | Parité     | Données   |
| D -   Prog. d' | A: <suiv>         | L: Aucune  | S: 5      |
| E -   Débi     | B: <prev>         | M: Paire   | T: 6      |
| F - Contrôle de | C: 9600           | N: Impaire | U: 7      |
| G - Contrôle de | D: 38400          | O: Marque  | V: 8      |
| H - RS485 En   | E: 115200         | P: Espace  |           |
| I - RS485 Rts  |                   |           |           |
| J - RS485 Rts A | Bits de stop      |           |           |
| K - RS485 Rx Du | W: 1              | Q: 8-N-1   |           |
| L - RS485 Termi | X: 2              | R: 7-E-1   |           |
| M - RS485 Delay |                   |           |           |
| N - RS485 Delay |                   |           |           |
|           | Choix ou <Entrée> pour sortir ? █ |           |
| Changer quel | +-----+-----+ |           |
+-----+-----+-----+

```



```

+-----+
| A -          Port série : /dev/ttyUSB0
| B - Emplacement fichier verr. : /var/lock
| C -   Prog. d'appel entrant :
| D -   Prog. d'appel sortant :
| E -      Débit/Parité/Bits : 9600 8N1
| F - Contrôle de flux matériel : Oui
| G - Contrôle de flux logiciel : Non
| H -   RS485 Enable      : No
| I -   RS485 Rts On Send : No
| J -   RS485 Rts After Send : No
| K -   RS485 Rx During Tx  : No
| L -   RS485 Terminate Bus : No
| M - RS485 Delay Rts Before: 0
| N - RS485 Delay Rts After : 0
|
|   Changer quel réglage ? █
+-----+

```

```

Switch>en
Switch#config
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#interface vlan 1
Switch(config-if)#ip address 10.13.0.1 255.255.255.0
Switch(config-if)#no sh

```

```

Switch(config-if)#exit
Switch(config)#ip dhcp pool MYP00L
Switch(dhcp-config)#network 10.13.0.0 255.255.255.0
Switch(dhcp-config)#default-router 10.13.0.1
Switch(dhcp-config)#dns-server 8.8.8.8
Switch(dhcp-config)#exit
Switch(config)#

```

```

Switch(config)#
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan 1

```

```

Switch(config)#interface range GigabitEthernet1/0/1 - 24
Switch(config-if-range)#ip dhcp snooping trust
Switch(config-if-range)#exit
Switch(config)#
Switch(config)#
Switch(config)#ip arp inspection vlan 1
Switch(config)#interface range GigabitEthernet1/0/1 - 24
Switch(config-if-range)#ip arp inspection trust
Switch(config-if-range)#exit
Switch(config)#exit

```



```

attaquant@p20123:~$ sudo dhclient eth0
attaquant@p20123:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group defau
lt qlen 1000
    link/ether 40:a6:b7:ae:0a:86 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.13.0.2/24 brd 10.13.0.255 scope global dynamic eth0
        valid_lft 86394sec preferred_lft 86394sec
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gro
up default qlen 1000
    link/ether bc:0f:f3:cb:93:e0 brd ff:ff:ff:ff:ff:ff
    altname eno1
    altname enp0s31f6
    inet 192.168.51.23/24 brd 192.168.51.255 scope global dynamic noprefixroute e
th1
        valid_lft 246sec preferred_lft 246sec
    inet6 fe80::be0f:f3ff:fe0b:93e0/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
attaquant@p20123:~$ █

```

## PC client

```

client@p20124:~$ sudo dhclient eth0

Nous espérons que vous avez reçu de votre administrateur système local
les consignes traditionnelles. Généralement, elles se concentrent sur ces trois éléments :

#1) Respectez la vie privée des autres.
#2) Réfléchissez avant d'utiliser le clavier.
#3) De grands pouvoirs confèrent de grandes responsabilités.

[sudo] Mot de passe de client :
client@p20124:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 40:a6:b7:ae:05:a6 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 10.13.0.3/24 brd 10.13.0.255 scope global dynamic eth0
        valid_lft 86397sec preferred_lft 86397sec
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether bc:0f:f3:cb:93:cf brd ff:ff:ff:ff:ff:ff
    altname eno1
    altname enp0s31f6
    inet 192.168.51.24/24 brd 192.168.51.255 scope global dynamic noprefixroute eth1
        valid_lft 197sec preferred_lft 197sec
    inet6 fe80::be0f:f3ff:fe0b:93cf/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
client@p20124:~$ █

root@p20124:/home/client# ping 10.13.0.4
PING 10.13.0.4 (10.13.0.4) 56(84) bytes of data.
64 bytes from 10.13.0.4: icmp_seq=9 ttl=64 time=5.77 ms
64 bytes from 10.13.0.4: icmp_seq=10 ttl=64 time=0.737 ms
64 bytes from 10.13.0.4: icmp_seq=11 ttl=64 time=0.759 ms
^C
--- 10.13.0.4 ping statistics ---
18 packets transmitted, 3 received, 83.3333% packet loss, time 17370ms
rtt min/avg/max/mdev = 0.737/2.420/5.765/2.365 ms
root@p20124:/home/client# █

```

ssh:

```
root@p20124:/home/client# ssh user@10.13.0.4
```

ARP VIDE:

```
root@p20124:/home/client# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
192.168.51.254          ether    14:18:77:4c:87:0c    C                eth1
```

ARP rempli ;

```
root@p20124:/home/client# arp
Adresse                TypeMap AdresseMat      Indicateurs      Iface
192.168.51.254          ether    14:18:77:4c:87:0c    C                eth1
10.13.0.2                ether    40:a6:b7:ae:0a:86    C                eth0
10.13.0.4                ether    40:a6:b7:ae:06:21    C                eth0
root@p20124:/home/client# █
```

## Conclusion

Durant notre projet intégratif SAE24 sur les attaques MITM sur un réseau local durant l'année académique 2023-2024, nous avons exploré ensemble le monde de la sécurité informatique. En travaillant en équipe, nous avons mis en pratique des techniques d'attaque comme l'ARP spoofing et le man-in-the-middle sur SSH, tout en étudiant les moyens de protéger un réseau local.

À travers nos expériences, nous avons appris comment ces attaques fonctionnent et comment les contrer en utilisant des méthodes comme le DHCP snooping et l'ARP inspection. Ces connaissances nous ont permis de mieux comprendre les vulnérabilités des réseaux locaux et d'acquérir des compétences pratiques en sécurisation informatique.

En collaborant et en partageant nos connaissances, nous avons atteint nos objectifs pédagogiques tout en développant nos compétences en programmation, en analyse réseau et en gestion de projet. Ce projet nous a préparés à relever les défis de la sécurité informatique dans notre future carrière professionnelle.