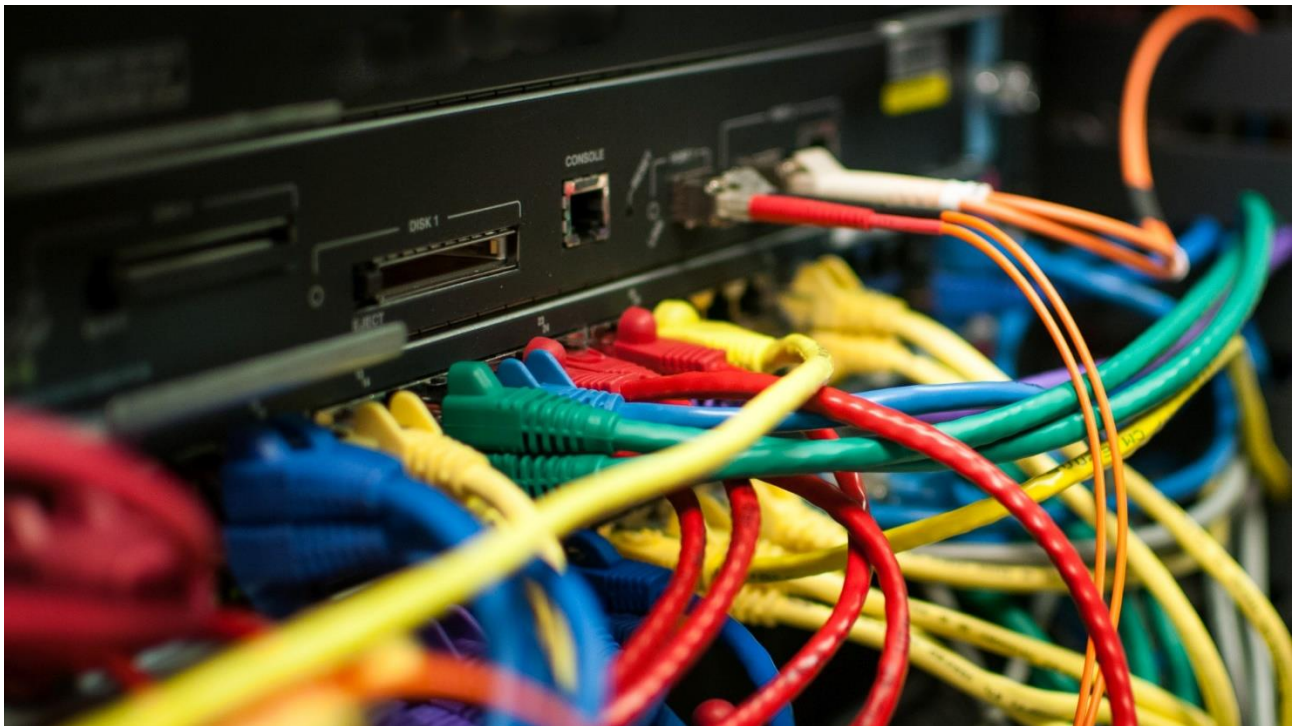


06 JUIN 2022

COMPTE RENDU SAE 21

CONSTRUIRE UN RESEAU



THOMAS BILGER- MASSIMO AURELI – RAYANE DIB - LOUIS DESVERNOIS

Table des matières

| | |
|---|----|
| I/ Introduction | 2 |
| II/ Mise en place du réseau | 3 |
| II.1 Découpage du réseau local | 3 |
| II.2 Adressage IP | 4 |
| II.3 Routage inter-vlan | 4 |
| II.4 Accès SSH sur le commutateur et le routeur | 5 |
| II.5 DHCP | 6 |
| II.6 Interconnexion des routeurs et routage | 6 |
| III/ Sécurité du réseau | 14 |
| III.1 Empoisonnement du cache ARP..... | 14 |
| III.2 Sécurité des ports du commutateur..... | 19 |
| III.3 Verrouillage du port SSH du routeur | 20 |
| IV/ Mise en place des services..... | 22 |
| IV.1 Serveur FTP..... | 22 |
| IV.2 Serveur web | 23 |
| V/ Conclusion | 25 |
| Références..... | 26 |

Tables des figures

| | |
|---|----|
| Figure 1: Structure générale du réseau | 3 |
| Figure 2: Paramètres SSH du routeur et du commutateur | 5 |
| Figure 3: Ordre de parcours en largeur | 8 |
| Figure 4: Ordre de parcours en profondeur | 9 |
| Figure 5: Exemple de graphe que nous allons utiliser..... | 10 |
| Figure 6: Première étape de notre exemple | 11 |
| Figure 7: Deuxième étape de notre exemple | 11 |
| Figure 8: Troisième étape de notre exemple | 12 |
| Figure 9: Les dernières étapes de notre exemple | 12 |
| Figure 10: "Table de routage" du nœud A | 13 |
| Figure 11: Configuration IP de l'attaquant | 14 |
| Figure 12 : Configuration IP de l'attaquant | 15 |
| Figure 13: PC recevant les pings..... | 15 |
| Figure 14: PC attaquant avec ettercap-graphical ouvert | 16 |
| Figure 15: Scan du réseau par le PC attaquant | 16 |
| Figure 16: Liste des hôtes présents sur le réseau | 17 |
| Figure 17: Menu des attaques Man In The Middle (MITM) | 17 |
| Figure 18: Paramètres de l'attaque empoisonnement de cache ARP | 18 |
| Figure 19: PC attaquant écoutant le réseau..... | 18 |
| Figure 20: PC cible montrant le ralentissement des requêtes pendant l'attaque | 19 |
| Figure 21: Table ARP corrompue du PC cible | 19 |
| Figure 22: Connexion SSH utilisateur | 21 |
| Figure 23: Connexion SSH administrateur..... | 21 |
| Figure 24: Status du démon proftpd | 22 |
| Figure 25: Commande pour ajouter l'utilisateur Antoine | 22 |
| Figure 26: Rajoute l'autorisation groupe au dossier www..... | 22 |
| Figure 27: Commande pour ajouter l'utilisateur Cathy..... | 22 |
| Figure 28: Mise à jour de l'autorisation au dossier www | 22 |
| Figure 29: L'utilisateur Cathy peut lire/écrire sur FileZilla | 23 |
| Figure 30 : Page par défaut d'apache..... | 23 |
| Figure 31: Fichier squid.conf | 24 |
| Figure 32: Etat des ports de la machine | 24 |
| Figure 33: Log d'accès du proxy | 25 |

I/ Introduction

Cette SAE s'est déroulée durant le second semestre et avait pour objectif de consolider notre expertise technique sur le matériel réseau de niveau deux (commutateur) et trois (routeur). Nous avons mis en place un réseau local simple, nous y avons ajouté des règles de sécurité et nous avons également mis en place des services tels qu'un serveur HTTP Apache2, un serveur FTP proftpd ainsi qu'un proxy Squid. Nous avons travaillé en groupe de quatre, chaque groupe a géré de manière indépendante son réseau, mais pour la segmentation des adresses IP et l'interconnexion des routeurs via OSPF, une coopération avec les membres des autres groupes a été nécessaire. Dans ce rapport, nous commencerons par décrire la mise en place du réseau en lui-même (attributions IP, créations VLANs, etc) puis nous continuerons sur la sécurisation du réseau et au déploiement des services.

II/ Mise en place du réseau

II.1 Découpage du réseau local

Pour découper le réseau, nous avons mis en place trois VLANs sur les commutateurs. La segmentation du réseau est une stratégie qui consiste à isoler les parties du réseau les unes des autres afin d'améliorer la sécurité et de réduire la congestion.

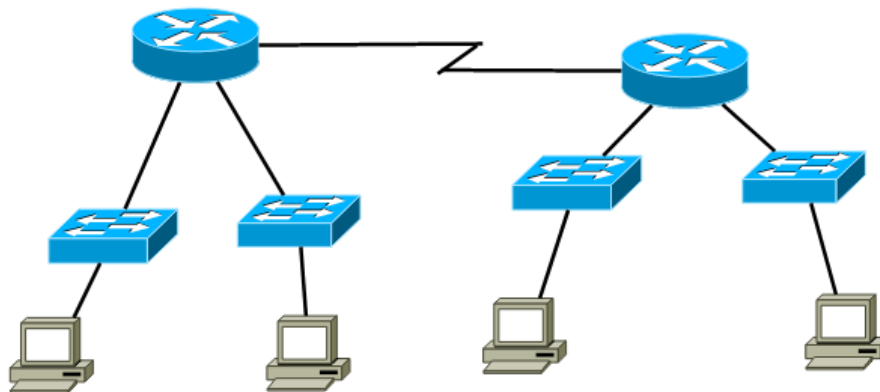


Figure 1: Structure générale du réseau

La structure générale du réseau que nous avons dû mettre en place (sont représenté ici le réseau de deux groupes). Chaque groupe ont été équipé d'un routeur et de deux commutateurs. Chacun de nos commutateurs ont été configuré de la même manière, trois VLANs *administrateur*, *utilisateur* et *test*, deux ports par VLANs et un port trunk vers le routeur.

```
Switch(config)#vlan 11
Switch(config-vlan)#name admin
Switch(config-vlan)#vlan 21
Switch(config-vlan)#name util
Switch(config-vlan)#vlan 31
Switch(config-vlan)#name test
Switch(config-vlan)#exit
Switch(config)#int range fastEthernet 0/1-2
Switch(config-if-range)#sw access vlan 11
Switch(config)#int range fastEthernet 0/3-4
Switch(config-if-range)#sw access vlan 21
Switch(config)#int range fastEthernet 0/5-6
Switch(config-if-range)#sw access vlan 31
```

Les commandes utilisées pour créer et configurer les noms et les ports de chaque VLAN sont les mêmes sur les deux commutateurs.

II.2 Adressage IP

Le groupe RT11 s'est vu attribuer le réseau 172.16.128.0/17. Étant situé en salle C007, nous avons choisi, après discussion avec les autres groupes de la salle, d'attribuer à chacun de nos VLANs un /24 divisé en deux /25 pour chaque commutateur.

| | Vlan11 | Vlan21 | Vlan31 |
|-------------------------------------|-------------------|-------------------|-------------------|
| Réseau | 172.16.160.0/24 | 172.16.161.0/24 | 172.16.162.0/24 |
| Sw1 | 172.16.160.0/25 | 172.16.161.0/25 | 172.16.162.0/25 |
| Sw2 | 172.16.160.128/25 | 172.16.161.128/25 | 172.16.162.128/25 |
| | | | |
| Réservé pour le routage de la salle | | | 172.16.255.0/24 |

Tableau 1: Adressage IP

Cette segmentation des adresses IP nous permet de largement attribuer assez d'adresses aux hôtes de chaque VLANs. Pour le routage entre les routeurs des groupes, nous avons réservé le réseau 172.16.255.0/24 que nous allons séparer en plusieurs /30.

II.3 Routage inter-vlan

Pour effectuer le routage inter-vlan de notre topologie "router on a stick" nous avons configuré les ports trunks du commutateur et les interfaces virtuelles du routeur pour utiliser l'encapsulation IEEE 802.1Q (dot1Q) et pour que seulement les VLANs admin et util peuvent être routés pour isoler le réseau de test. Pour les commutateurs, nous avons utilisé ces commandes ci-dessous¹.

```
Sw1(config)#int gigabitEthernet 0/1
Sw1(config-if)#sw mode trunk
Sw1(config-if)#sw trunk allowed vlan 11,21
```

Chaque interface virtuelle sur le routeur a été configurés avec la dernière adresse IP du réseau sur laquelle elle se situe (e.g. 172.16.160.126/24 pour le VLAN11 sur Sw1). Après l'attribution des IP, aucune autre configuration supplémentaires n'est nécessaire, car tous les réseaux sont « directement connectés » au routeur. Par exemple, voici ci-dessous les commandes nécessaires pour configurer une interface virtuelle².

```
Router(config)#int gigabitEthernet 0/0/0.11
Router(config-subif)#encapsulation dot1Q 11
Router(config-subif)#ip address 172.16.160.126 255.255.255.128
```

Nous avons également mis en place le port-mirroring. La mise en miroir de ports est une fonction des commutateurs Cisco qui permet de transférer des copies de tous les paquets d'un VLAN vers un autre port désigné, ce qui permet par exemple de connecter un ordinateur sur lequel le logiciel Wireshark est utilisé pour surveiller ou analyser le réseau en cas de problèmes.

¹ Les commandes sont les mêmes pour les deux commutateurs

² Nous avons exécuté ces commandes quatre fois (deux fois par VLANs, car les deux commutateurs partagent uniquement des VLAN différents ayant le même nom), ici on configure le VLAN 11 de Sw1.

Pour activer cette fonction nous avons utilisé les commandes suivantes :

```
Switch(config)#monitor session 9 source vlan 31
Switch(config)#interface fa 0/10
Switch(config-if-range)#switchport access Vlan11
Switch(config)#monitor session 6 source vlan 21
Switch(config)#interface range fa 0/11
Switch(config-if-range)#switchport access Vlan11
Switch(config)#monitor session 3 source vlan 11
```

II.4 Accès SSH sur le commutateur et le routeur

Pour la partie SSH, nous avons attribué les adresse IP au commutateur comme suit :

| | |
|----------------|-----|
| 172.16.160.125 | SW1 |
| 172.16.160.253 | SW2 |

Tableau 2: Attribution IP commutateur

Ensuite nous avons mis en place l'accès SSH sur les commutateurs et le routeur pour cela nous utilisons les commandes suivantes³ [1].

```
Routeur(config)#ip domain-name uha.fr
Routeur(config)#crypto key generate rsa general-keys modulus 1024
Routeur(config)#ip ssh version 2
Routeur(config)#ip ssh time-out 60
Routeur(config)#ip ssh authentication-retries 3
Routeur(config)#service password-encryption
Routeur(config)#username admin password 0 toto
Routeur(config)# line vty 0 4
Routeur(config-line)# login local
Routeur(config-line)# transport input ssh
```

```
!
ip ssh version 2
ip ssh time-out 60
ip domain-name uha.fr
!
```

Figure 2: Paramètres SSH du routeur et du commutateur

³ Les commandes sont les mêmes sur les commutateurs

II.5 DHCP

Nous avons mis DHCP, nous avons configuré le routeur, le DHCP nous permet d'attribuer automatiquement des adresses IP aux clients.

Nous avons appliqué ces commandes pour mettre en place DHCP sur le routeur.

```
Switch(config)#ip dhcp pool admin1
Switch(dhcp-config)#network 172.16.160.0 255.255.255.128
Switch(dhcp-config)#default-router 172.16.160.126
Switch(config)#ip dhcp pool admin2
Switch(dhcp-config)#network 172.16.160.128 255.255.255.128
Switch(dhcp-config)#default-router 172.16.160.254
Switch(config)#ip dhcp pool util1
Switch(dhcp-config)#network 172.16.161.0 255.255.255.128
Switch(dhcp-config)#default-router 172.16.161.126
Switch(config)#ip dhcp pool util2
Switch(dhcp-config)#network 172.16.161.128 255.255.255.128
Switch(dhcp-config)#default-router 172.16.161.254
```

Nous pouvons maintenant configurer les hôtes pour utiliser le protocole DHCP pour leur adressage IP.

II.6 Interconnexion des routeurs et routage

II.6.1 Mise en place du protocole OSPF sur le routeur

Nous avons ensuite activé le protocole OSPF "open shortest path first", un protocole de routage par état de lien, afin d'automatiser la distribution des tables de routage entre les routeurs des différents groupes. Voici les commandes que nous avons exécutées.

```
Router(config)#router ospf 1
Router(config-router)#network 172.16.160.0 255.255.255.128 area 0
Router(config-router)#network 172.16.160.128 255.255.255.128 area 0
Router(config-router)#network 172.16.161.0 255.255.255.128 area 0
Router(config-router)#network 172.16.161.128 255.255.255.128 area 0
Router(config)#int serial 0/1/0
Router(config-if)#ip ospf 1 area 0
```

Ici, nous spécifions quels réseaux doit annoncer le routeur et sur quelle zone ("area"), nous utilisons la zone 0, la zone dite de "backbone", car notre installation n'en nécessite pas plus. Voici le résultat de ces commandes avec un autre routeur configuré pour utiliser le protocole OSPF.⁴

```
Router#sh ip route
[...]
    172.16.0.0/16 is variably subnetted, 5 subnets, 3 masks
O       172.16.160.0/25 [110/65] via 172.16.255.2, 00:00:11, Serial0/1/0
O       172.16.160.128/25 [110/65] via 172.16.255.2, 00:00:11,
Serial0/1/0
O       172.16.161.0/25 [110/65] via 172.16.255.2, 00:00:11, Serial0/1/0
O       172.16.161.128/25 [110/65] via 172.16.255.2, 00:00:11, Serial0/1/0
C       172.16.255.0/30 is directly connected, Serial0/1/0
L       172.16.255.1/32 is directly connected, Serial0/1/0
[...]
```

Comme nous pouvons le voir avec les lignes commençant par la lettre "O", les réseaux configurés précédemment sont donc correctement annoncés aux autres routeurs utilisant OSPF.

⁴ Résultats copiés de PacketTracer

II.6.2 OSPF en détails.

OSPF est un protocole de routage par état de liens, il permet le partage d'information de routage en les différents routeurs d'une organisation (Autonomous System), mais il ne permet le routage sur Internet comme BGP. Les trames OSPF sont envoyées en broadcast sur les interfaces configurées pour utiliser le protocole. Puis chaque route apprise et associée à un métrique de distance, de plus quand plusieurs routes de même coût vers une même destination existent, le trafic est distribué équitablement entre elles .

OSPF envoie périodiquement des paquets « hello », qui est un sous-protocole d'OSPF qui permet notamment de découvrir ces voisins ainsi que de s'assurer que ces mêmes voisins sont encore fonctionnels. OSPF communique l'ID du routeur ainsi que l'ID de la zone « area » dans toutes ces trames .

OSPF utilise quatre autres types de message utilisant leur propre type de paquets :

- les échanges de bases de données (« database description ») : première synchronisation des bases de données entre voisins
- les demandes d'état de lien (« link state request ») : demande d'un routeur lorsqu'il détecte qu'une partie de la base de données doit être modifiée
- les mises à jour d'état de lien (« link state update ») : réponse à une demande d'état de lien, contient les informations demandées, ces paquets sont envoyés en multicast si le réseau le supporte.
- les acquittements d'état de lien (« link state acknowledgment ») : envoyé après qu'une mise à jour d'état de lien a été correctement appliquée.

II.6.2.1 : Algorithme de parcours en largeur

Le parcours en largeur (BFS, pour Breadth-First Search) est un algorithme permettant de calculer la distance de tous les nœuds depuis un nœud racine. Pour cela, l'algorithme explore⁵ puis « visite »⁶ en premier lieu tous ses voisins, une fois que cela est fait, les successeurs des voisins sont explorés puis visités à leur tour. On utilise ici une structure de file, c'est-à-dire que le premier nœud exploré est le premier à être visité. Chaque nœud visité sont marqués pour éviter de créer des boucles infinies. La FIGURE montre dans quel ordre les nœuds sont parcourus .

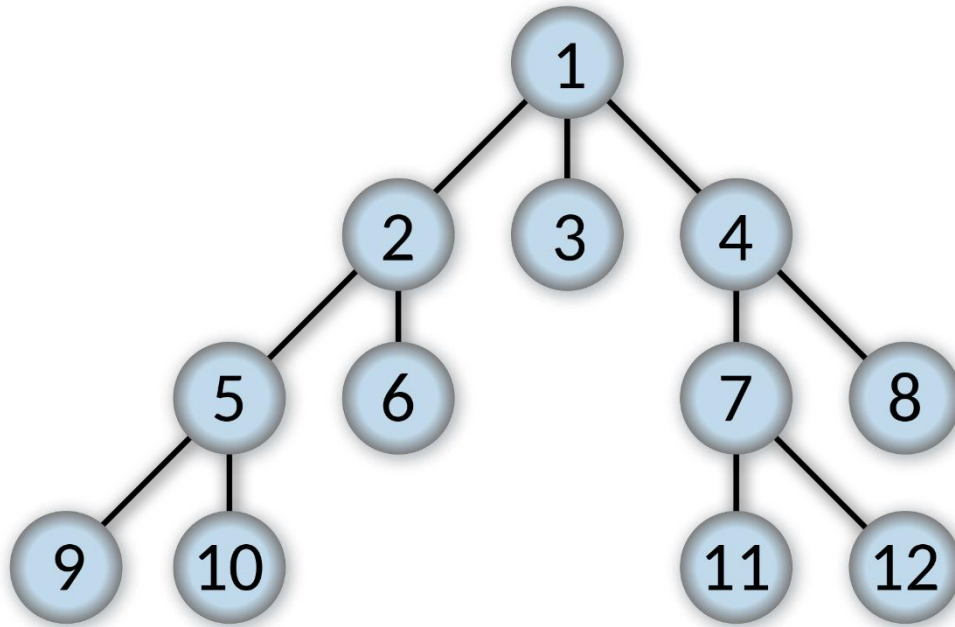


Figure 3: Ordre de parcours en largeur

⁵Quand l'algorithme "explore", il l'ajoute dans une liste d'attente

⁶Visiter signifie ici qu'il enregistre la distance du nœud

II.6.2.2 : Algorithme de parcours en profondeur

L'algorithme de parcours en profondeur, contrairement à l'algorithme de parcours en largeur, continue l'exploration jusqu'à arriver un cul-de-sac. C'est seulement à ce moment-là qu'il revient en arrière pour explorer depuis un autre nœud voisin. Ici, nous utilisons plutôt une structure en pile. Chaque nœud est également marqué quand celui-ci est exploré, pour éviter les boucles. Nous pouvons voir en FIGURE, l'ordre dans lequel les nœuds sont explorés .

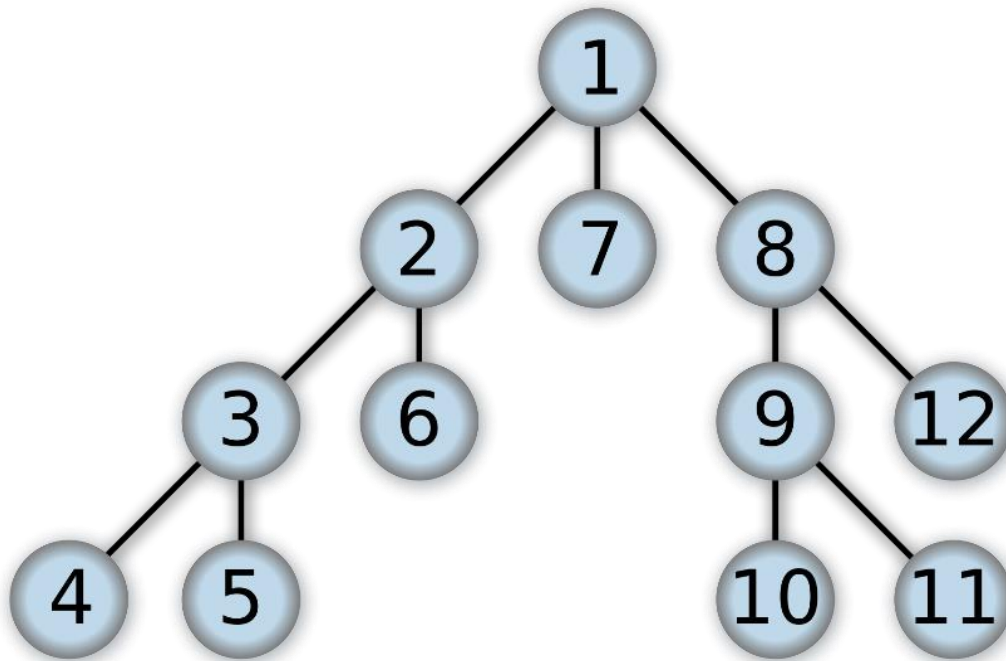


Figure 4: Ordre de parcours en profondeur

Il est facile de remarquer la différence entre ces deux algorithmes similaires après la visualisation de ces deux graphes. Les noms de ces algorithmes qui font donc références à l'ordre dans lequel les nœuds sont explorés.

II.6.2.3 : Algorithme de Dijkstra.

Le protocole OSPF utilise l'algorithme de Dijkstra qui permet de trouver le chemin le plus court vers un autre réseau. L'algorithme a été créé par le mathématicien et informaticien néerlandais, Dr. Edsger W. Dijkstra en 1959 . Pour mieux comprendre le fonctionnement de cet algorithme, nous allons l'exécuter étapes par étapes, pour cela, nous pouvons le représenter par un graphe comme celui en Figure 5, on a ici un exemple composé de sept nœuds.⁷

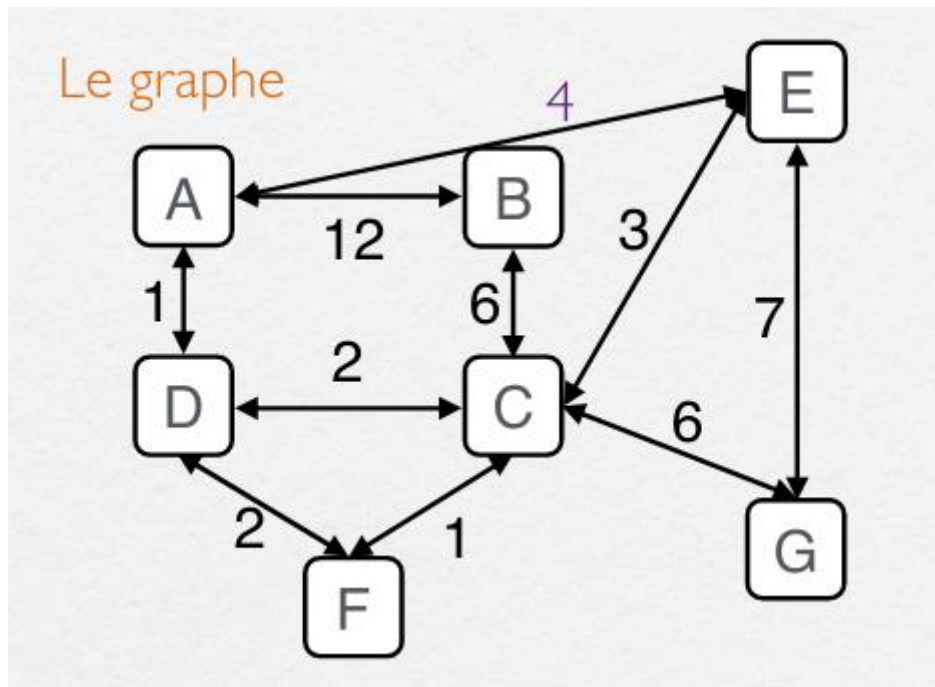


Figure 5: Exemple de graphe que nous allons utiliser

⁷Il s'agit ici des illustrations du cours de M. Hilt

Sur le graphe on remarque les liens entre chaque nœud (routeurs) ainsi que leurs coûts. Nous allons ici nous intéresser sur le processus de découverte du réseau par le nœud A. Pour représenter le choix des routes, nous pouvons séparer le graphe en deux comme ci-dessous en Figure 6. À gauche l'arbre optimal c'est-à-dire le graphe contenant uniquement les liens les plus courts. À droite nous avons les candidats, c'est-à-dire les nœuds directement connectés à n'importe quel nœud de l'arbre optimal. La distance est notée à côté des nœuds.

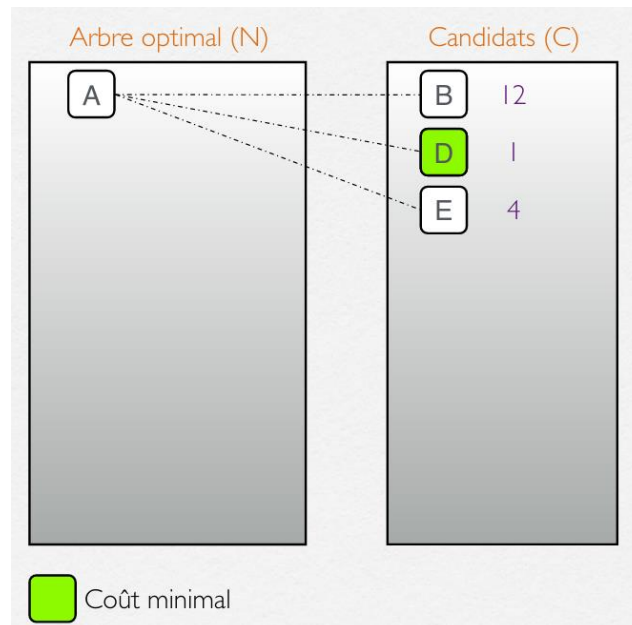


Figure 6: Première étape de notre exemple

Dans la liste des candidats, nous choisissons le coût minimal, ici le nœud D, et nous le déplaçons dans l'arbre optimal. Ce qui nous donne ce tableau en Figure 7. Après avoir ajouté le nœud avec un coût minimal, nous pouvons mettre à jour la liste de candidats avec tous les nœuds connectés directement à l'arbre optimal. Parmi la liste des candidats, nous choisissons de nouveau le (premier) nœud avec un coût minimal, ici c'est C avec un coût de trois. On remarque que A, un nœud déjà présent dans l'arbre optimal, est dans les candidats, nous pouvons le retirer de la liste.

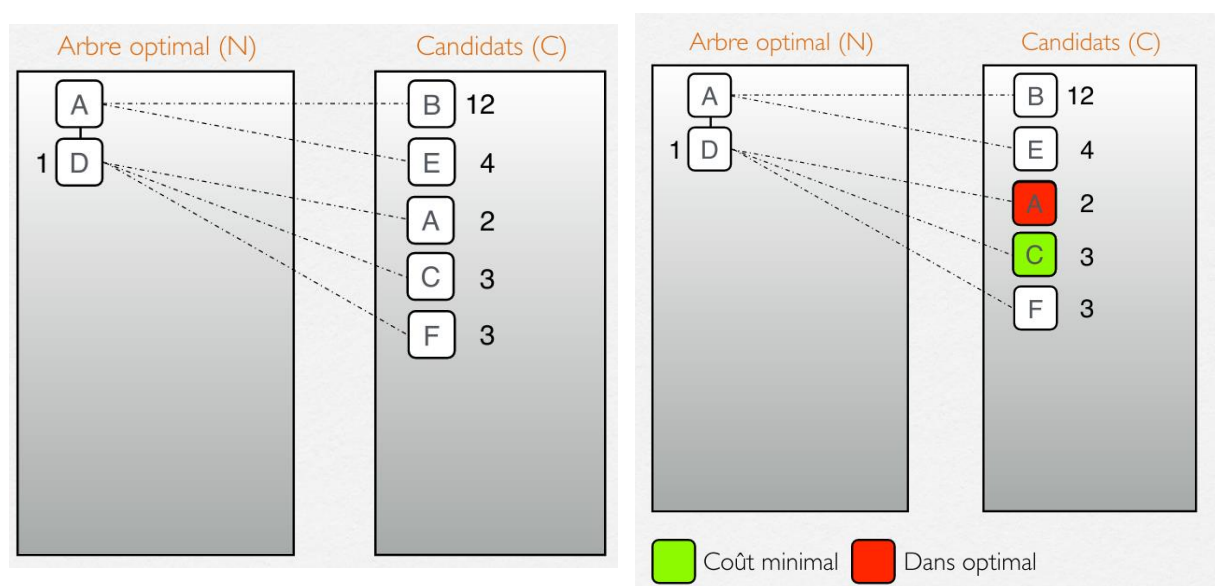


Figure 7: Deuxième étape de notre exemple

Ensuite, répétons le choix de l'étape précédente (Figure 8), c'est-à-dire que nous ajoutons le nœud choisi à l'étape précédente à l'arbre optimal et nous choisissons le nœud qui sera ajouté à l'étape suivante (en vert) et ignorons le nœud déjà présent (en rouge). Cette fois-ci, nous remarquons que des liens redondants sont présents dans les candidats, nous pouvons ignorer les liens ayant un coût non-minimal (en orange).

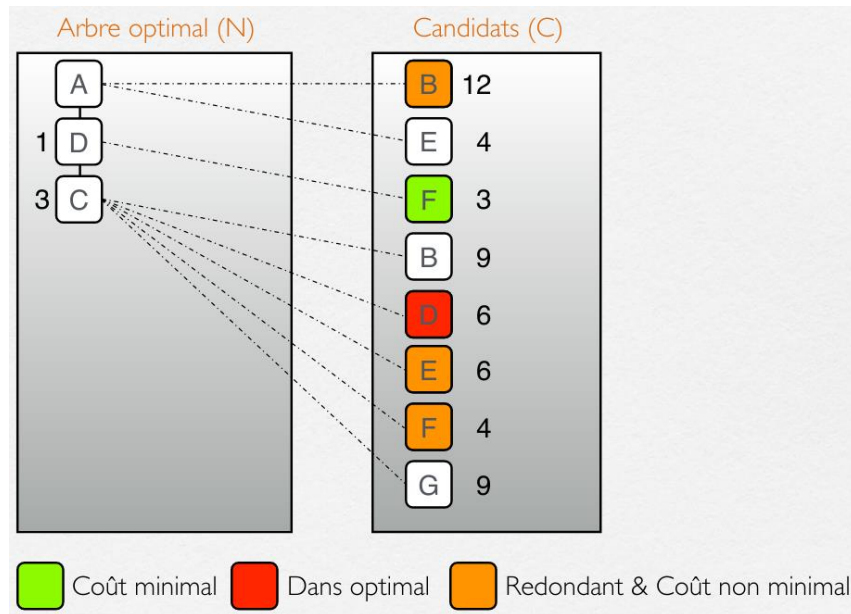


Figure 8: Troisième étape de notre exemple

Cet algorithme continue jusqu'à ce que l'arbre optimal contienne tous les nœuds, nous pouvons donc rapidement déduire l'arbre optimal final. En effet, en Figure 9, nous pouvons voir les dernières étapes ainsi que le résultat.

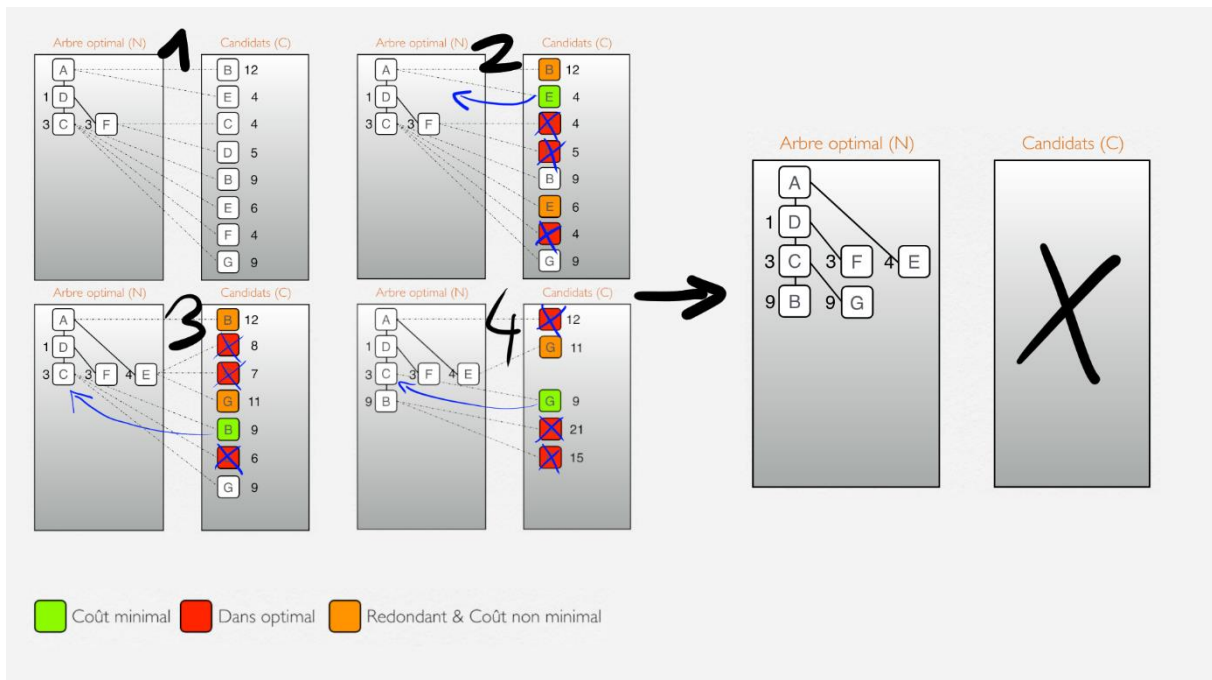


Figure 9: Les dernières étapes de notre exemple

À l'aide de ce graphe optimal, nous pouvons dresser la « table de routage » (en Figure 10) de A, c'est-à-dire les distances minimales entre chaque nœud.

Table de routage

| Pour aller à | Passer par | Coût |
|--------------|------------|------|
| A | local | 0 |
| B | D | 9 |
| C | D | 3 |
| D | D | 1 |
| E | E | 4 |
| F | D | 3 |
| G | D | 9 |

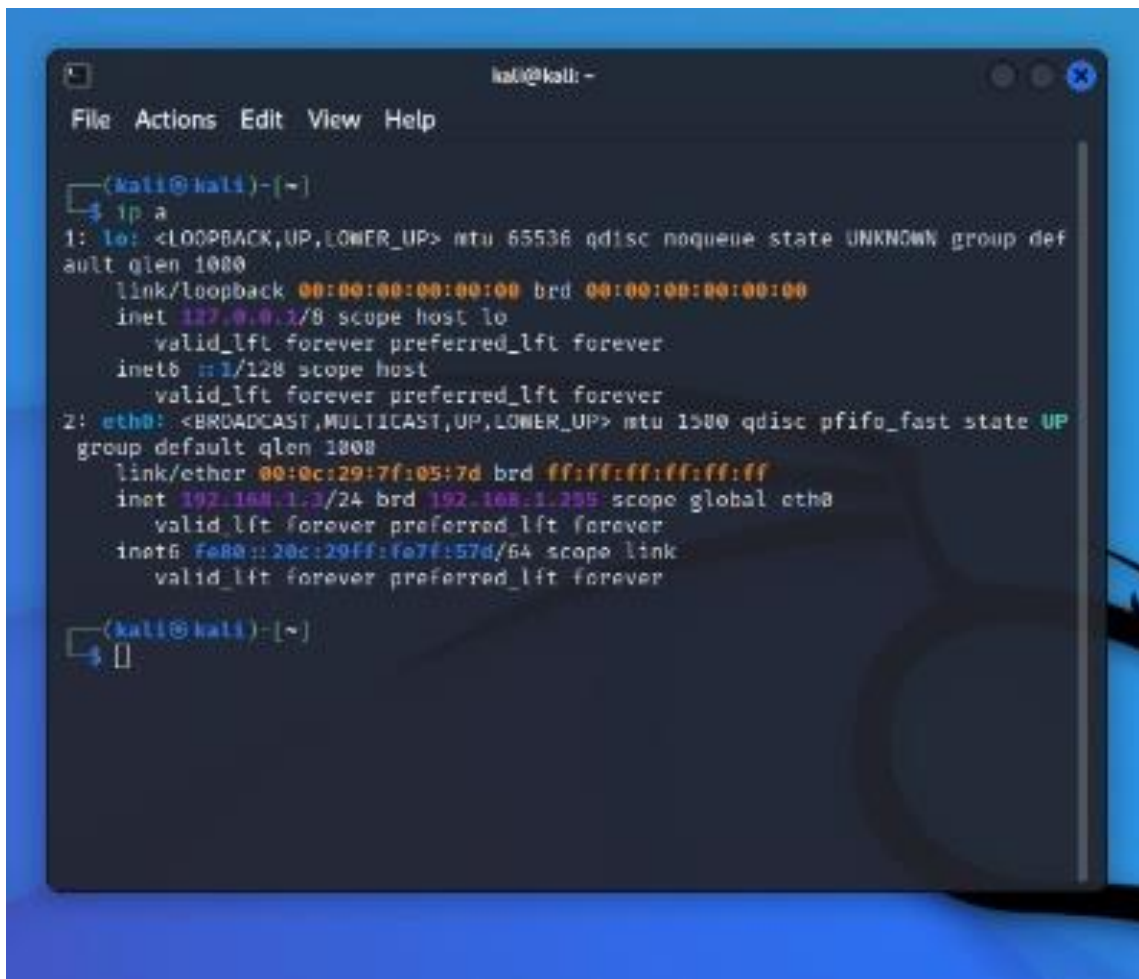
Figure 10: "Table de routage" du nœud A

III/ Sécurité du réseau

III.1 Empoisonnement du cache ARP

Pour effectuer un empoisonnement du cache ARP avec le logiciel ettercap-graphical⁸, nous utiliserons l'interface graphique de ce logiciel pour sa facilité d'utilisation et de compréhension. Pour cette partie nous avons suivi un tutoriel du site ITIGIC . Pour commencer nous aurons besoin d'une machine sous Linux⁹ (dans cet exemple nous en aurons trois afin de simuler un réseau local).

Voici donc la machine Attaquant qui est sous kali linux et qui possède l'IP 192.168.1.3 :



```
kali@kali: -
File Actions Edit View Help

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
  ault 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 pfifo_fast state UP
  group default qlen 1000
    link/ether 00:0c:29:7f:05:7d brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.3/24 brd 192.168.1.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe7f:57d/64 scope link
        valid_lft forever preferred_lft forever

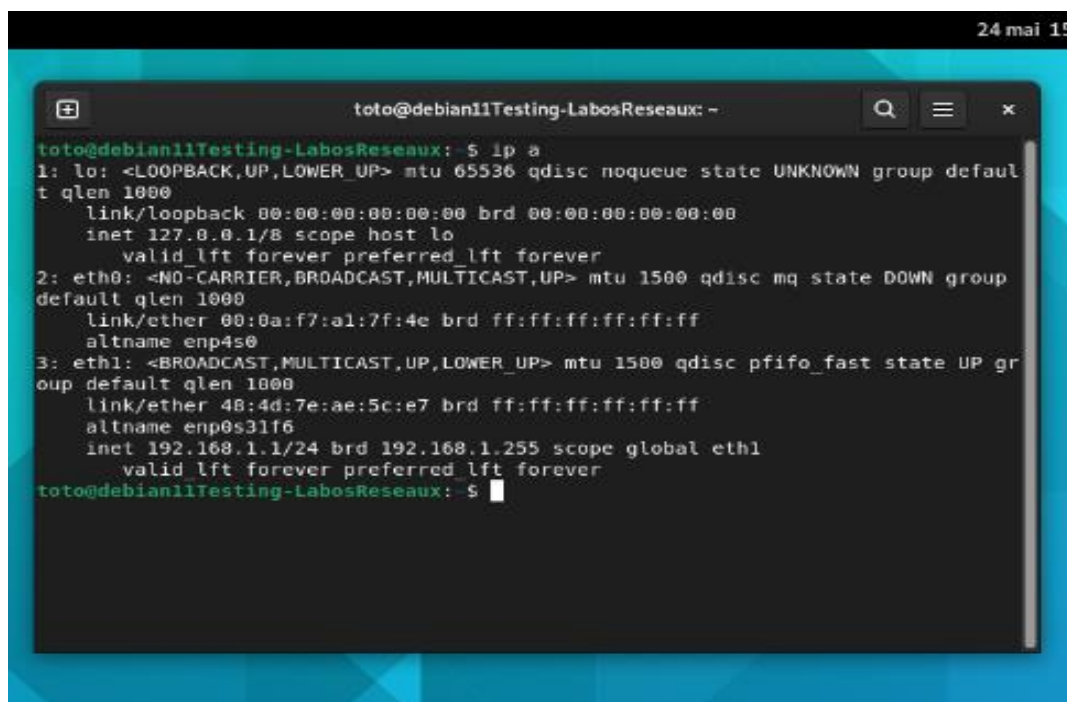
(kali@kali)-[~]
$
```

Figure 11: Configuration IP de l'attaquant

⁸ Logiciel disponible sur *nix et Microsoft Windows

⁹ Nous utilisons également Linux pour la victime pour éviter tout problème lié au pare-feu Windows

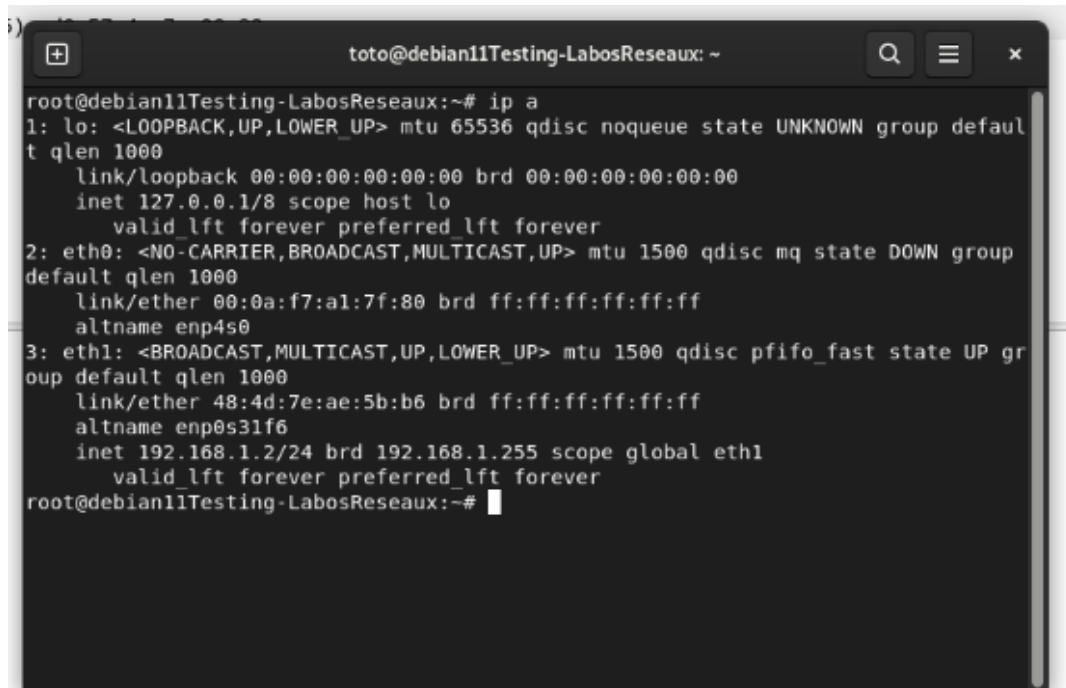
Ensuite nous avons la machine Victime qui tourne sous Debian, c'est la machine sur laquelle nous allons faire l'empoisonnement de cache ARP et qui possède l'IP 192.168.1.1 :



```
toto@debian11Testing-LabosReseaux: ~  
toto@debian11Testing-LabosReseaux:~$ ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default  
    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  
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group  
    default qlen 1000  
    link/ether 00:0a:f7:a1:7f:4e brd ff:ff:ff:ff:ff:ff  
    altname enp4s0  
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gr  
    oup default qlen 1000  
    link/ether 48:4d:7e:ae:5c:e7 brd ff:ff:ff:ff:ff:ff  
    altname enp0s31f6  
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth1  
        valid_lft forever preferred_lft forever  
toto@debian11Testing-LabosReseaux:~$
```

Figure 12 : Configuration IP de l'attaquant

Pour des fins de démonstration, nous avons la machine possédant l'IP 192.168.1.2 qui recevra les pings.



```
root@debian11Testing-LabosReseaux:~# ip a  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default  
    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  
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group  
    default qlen 1000  
    link/ether 00:0a:f7:a1:7f:80 brd ff:ff:ff:ff:ff:ff  
    altname enp4s0  
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gr  
    oup default qlen 1000  
    link/ether 48:4d:7e:ae:5b:b6 brd ff:ff:ff:ff:ff:ff  
    altname enp0s31f6  
    inet 192.168.1.2/24 brd 192.168.1.255 scope global eth1  
        valid_lft forever preferred_lft forever  
root@debian11Testing-LabosReseaux:~#
```

Figure 13: PC recevant les pings

Maintenant que nous avons présenté la configuration nous allons commencer l'empoisonnement.

Etape 1 :

- Lancez Ettercap-graphical laissez la configuration par défaut et cliquez sur l'icône « ✓ » :



Figure 14: PC attaquant avec ettercap-graphical ouvert

Etape 2 :

- Cliquez sur la loupe cet écran devrait apparaître après un court temps de chargement ou il analyse tous les hôtes disponibles (ce n'est pas la méthode la plus discrète mais ce n'est pas la discrétion que nous recherchons) :

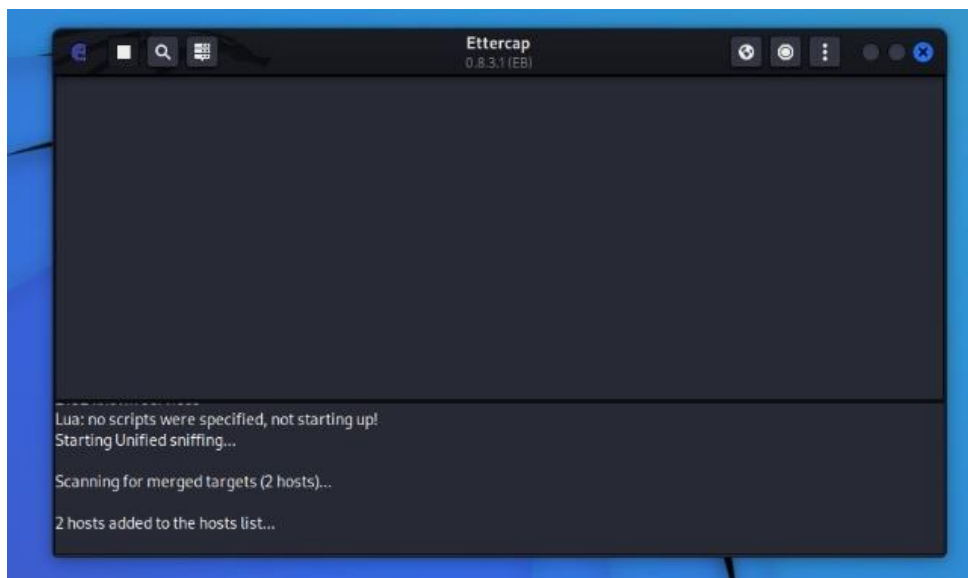


Figure 15: Scan du réseau par le PC attaquant

Etape 3 :

- Nous avons donc la fenêtre Hosts List qui s'ouvre et étant donné que nous voulons lancer une attaque dirigée sur une seule cible nous allons établir deux objectifs. Dans Target1 nous allons mettre la cible (donc l'adresse de la machine que nous souhaitons empoisonner) et dans Target2 nous allons mettre la machine à usurper¹⁰:

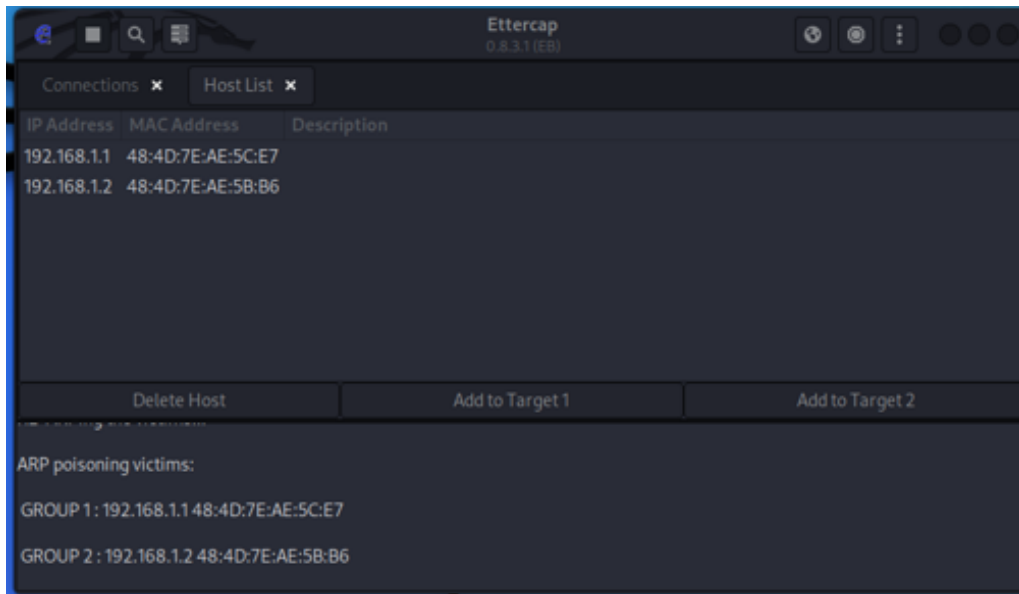


Figure 16: Liste des hôtes présents sur le réseau

Etape 4 :

- Maintenant cliquons sur la planète et allons dans ARP poisoning puis il faut cliquer sur ok pour lancer l'attaque :

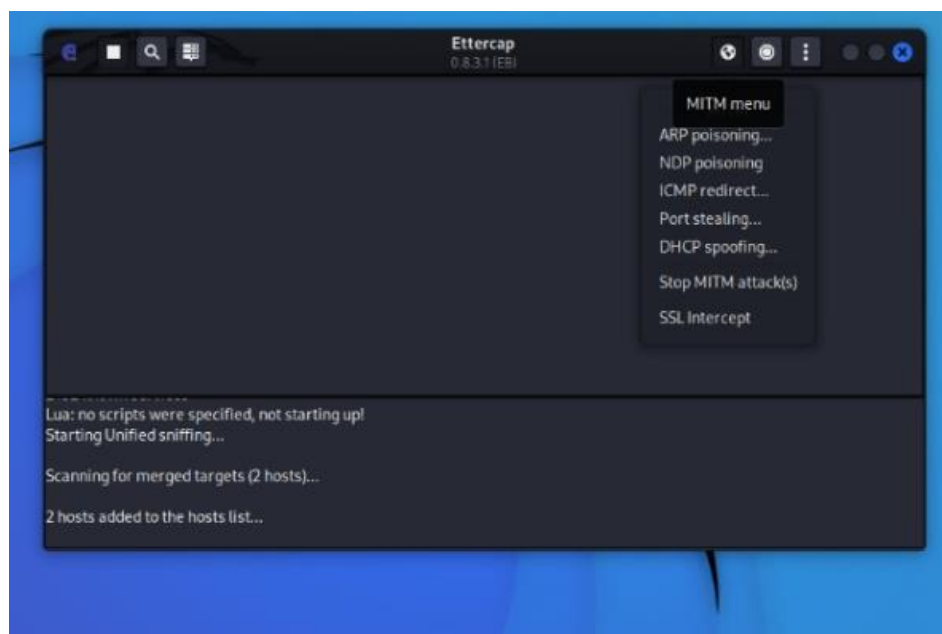


Figure 17: Menu des attaques Man In The Middle (MITM)

¹⁰ Nous allons usurper l'identité de la machine qui va recevoir les pings cependant il reste plus pertinent d'usurper la Gateway

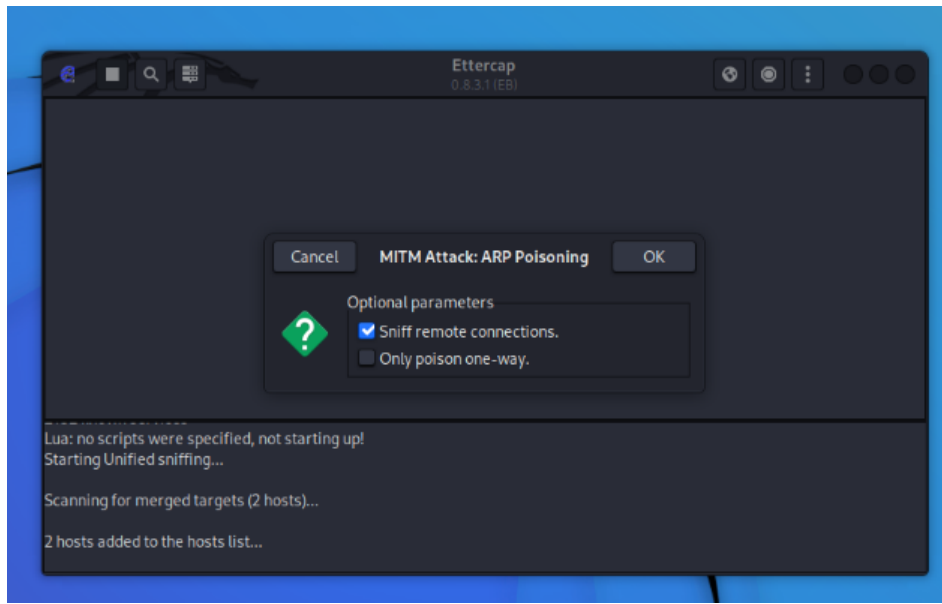


Figure 18: Paramètres de l'attaque empoisonnement de cache ARP

Etape 5 :

Écoutons !

- Maintenant il ne reste plus qu'à voir ce qu'il se passe avec Wireshark et comme vous pouvez le constater nous voyons tous les pings qui sont envoyés depuis la cible vers la machine qui reçoit les pings :

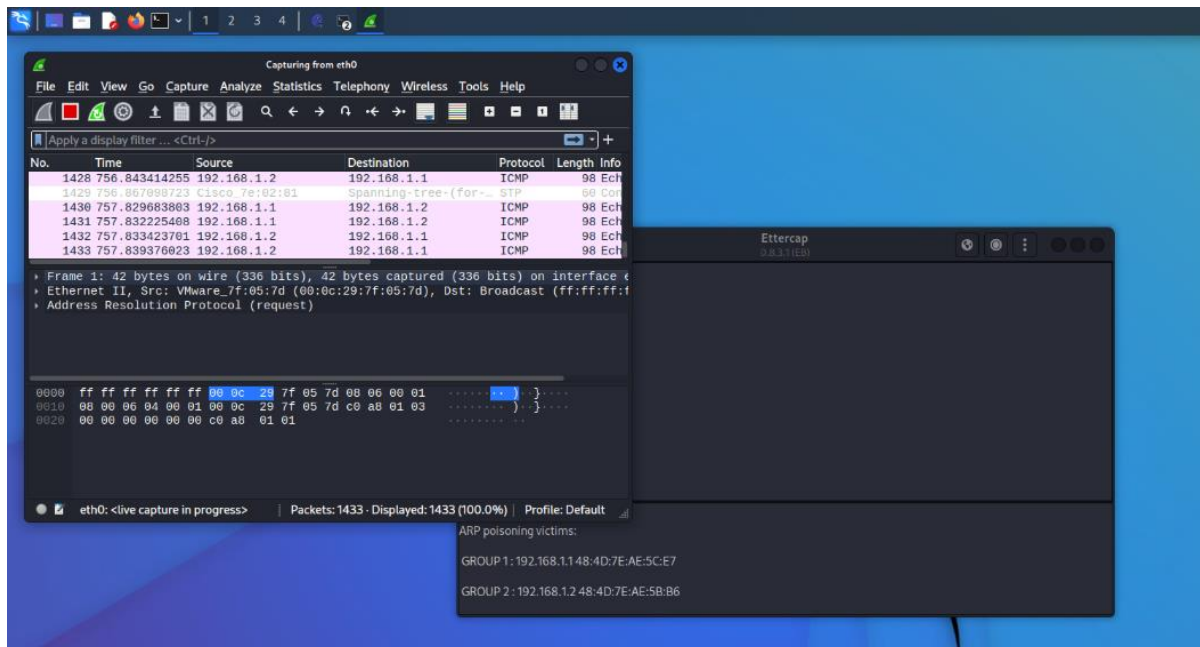


Figure 19: PC attaquant écoutant le réseau

- Voici les pings envoyés par la cible, Vous pourrez noter que quand l'attaque n'est pas lancée les pings mettent beaucoup moins de temps à arriver car ils ne transitent pas par une autre machine qui va les renvoyer :

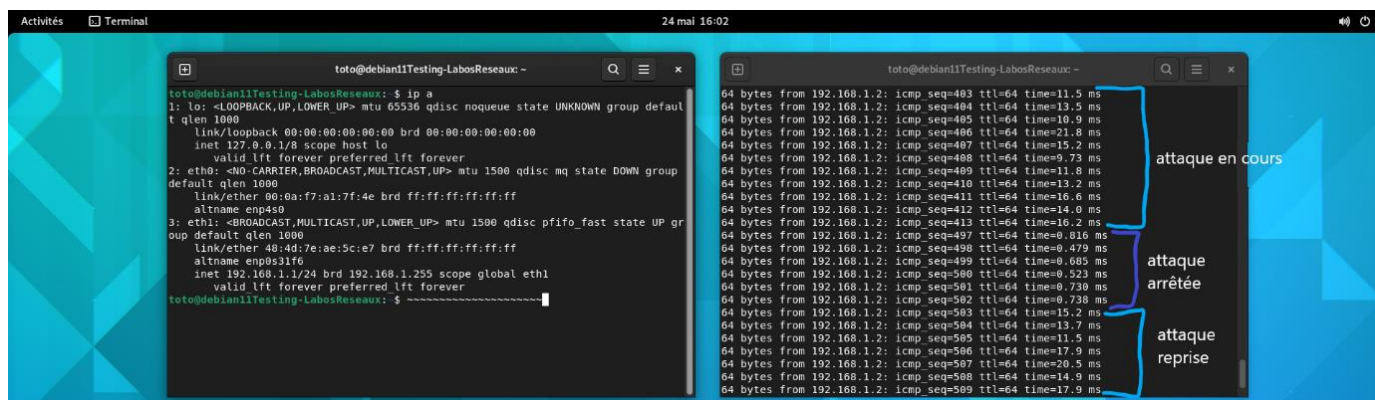


Figure 20: PC cible montrant le ralentissement des requêtes pendant l'attaque

- Voilà ! Nous avons effectué un empoisonnement du cache ARP en utilisant ettercap-graphical. Mais comment détecter un empoisonnement du cache ARP ? Dans notre table ARP si deux IP ont la même adresse MAC, alors le cache ARP est empoisonné :

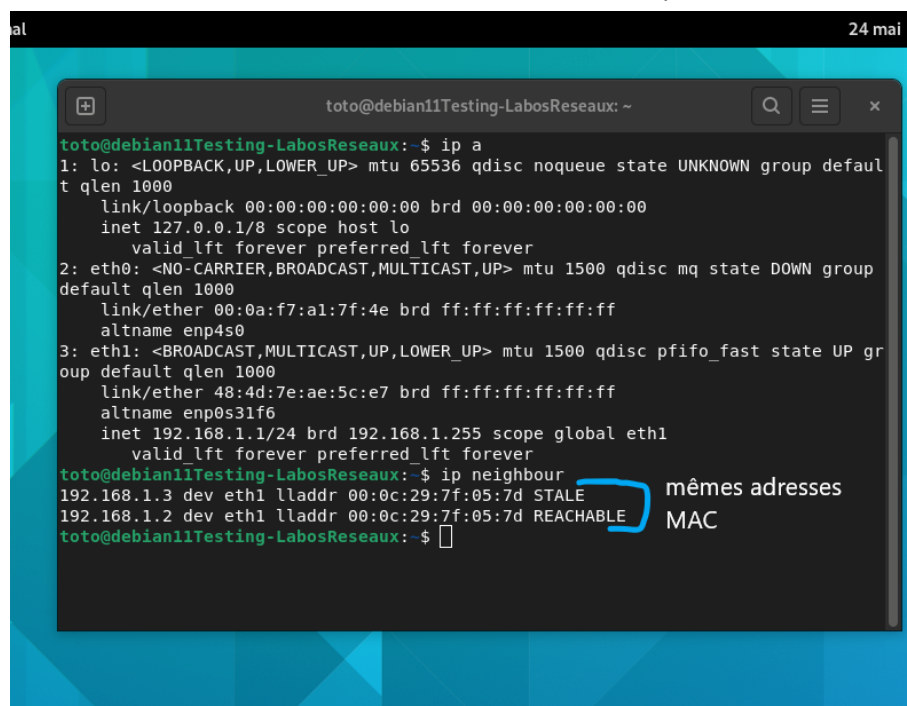


Figure 21: Table ARP corrompue du PC cible

III.2 Sécurité des ports du commutateur

Pour éviter l'empoisonnement du cache ARP il existe une solution qui consiste en la sécurisation des ports du commutateur en autorisant uniquement la première machine connectée sur le port. Pour cela on utilise la commande `port-security`

Par défaut, cette fonction est désactivée. Si elle est simplement activée, par défaut :

Une seule adresse MAC est apprise dynamiquement et elle la seule autorisée. En cas de "violation", le port tombe en mode shutdown.

Comme dans notre cas on souhaite d'une seule machine par port sur le commutateur nous allons rester sur la configuration par défaut :

```
(config)#interface G0/1
(config-if)#switchport mode access
(config-if)#switchport port-security
```

III.3 Verrouillage du port SSH du routeur

Afin d'activer le verrouillage du port SSH nous utilisons une « acces-list » afin d'autoriser les machines du vlan administrateur l'accès au port SSH du routeur. Cela empêche quiconque d'accéder au port SSH.

```
Router(config)#access-list extended SSH
Router(config-ext-nacl)#permit tcp 172.16.160.0 0.0.0.127 any eq 22
Router(config-ext-nacl)#permit tcp 172.16.160.128 0.0.0.127 any eq 22
Router(config-ext-nacl)#deny tcp any any eq 22
Router(config-ext-nacl)#permit ip any any
Router(config)#int gigabitEthernet 0/0/0.1111
Router(config-subif)#ip access-group SSH in
```

Les commandes ci-dessus permettent de définir les propriétés de l'ACL (ici l'accès au port 22 pour les réseaux admin) ensuite on interdit l'accès au port SSH pour les autres réseaux. Les deux dernières commandes permettent d'appliquer les ACL aux interfaces du routeur

¹¹ Ces commandes sont à répéter sur chaque interface du routeur

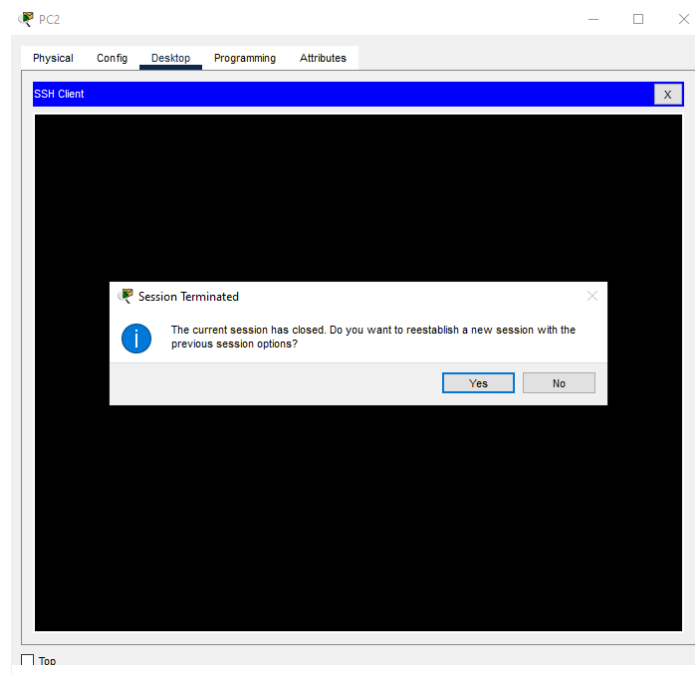


Figure 22: Connexion SSH utilisateur

Ici on peut voir qu'on ne peut pas se connecter au port SSH sur une machine client

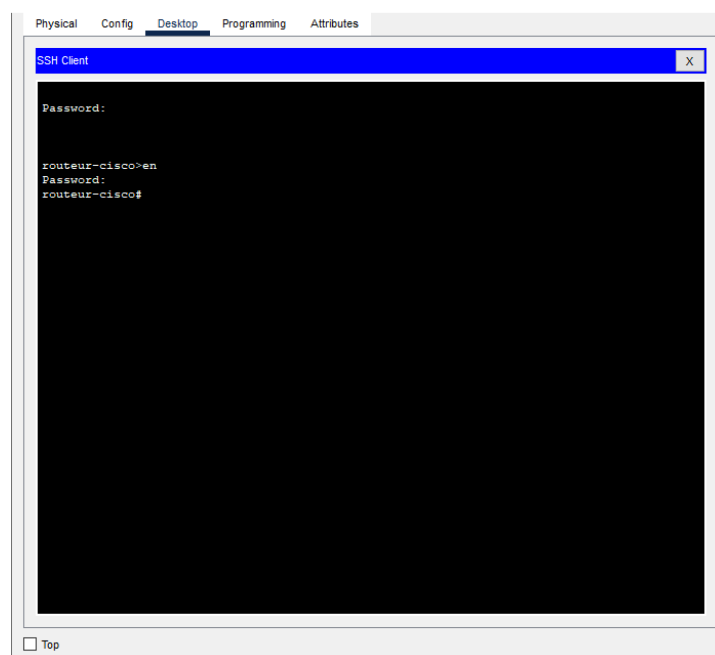


Figure 23: Connexion SSH administrateur

Sur cet exemple on peut se connecter en SSH depuis une machine sur le réseau administrateur

IV/ Mise en place des services

IV.1 Serveur FTP

Pour mettre en place le serveur FTP, nous avons utilisé le programme `proftpd` sur un ordinateur portable d'un des membres de notre groupe utilisant Kali Linux, une distribution basée sur Debian.

```
(sterben@Sterben)-[/etc/proftpd]
$ systemctl status proftpd
● proftpd.service - ProFTPD FTP Server
   Loaded: loaded (/lib/systemd/system/proftpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2022-06-03 23:11:52 CEST; 3s ago
     Docs: man:proftpd(8)
   Process: 36475 ExecStartPre=/usr/sbin/proftpd --configtest -c $CONFIG_FILE $OPTIONS (code=exited, status=0/SUCCESS)
   Process: 36476 ExecStart=/usr/sbin/proftpd -c $CONFIG_FILE $OPTIONS (code=exited, status=0/SUCCESS)
  Main PID: 36477 (proftpd)
    Tasks: 1 (limit: 8677)
   Memory: 1.5M
      CPU: 21ms
   CGroup: /system.slice/proftpd.service
           └─36477 "proftpd: (accepting connections)"

Jun 03 23:11:52 Sterben systemd[1]: Starting ProFTPD FTP Server...
Jun 03 23:11:52 Sterben proftpd[36475]: Checking syntax of configuration file
Jun 03 23:11:52 Sterben systemd[1]: Started ProFTPD FTP Server.
```

Figure 24: Status du démon `proftpd`

Ensuite, nous avons affecté les utilisateurs Cathy et Antoine au groupe `www-data`, ce qui nous permet de leur donner une autorisation de lecture-écriture.

```
(sterben@Sterben)-[/var/www]
$ sudo adduser --home /var/www/ --shell /bin/false --ingroup www-data antoine
```

Figure 25: Commande pour ajouter l'utilisateur Antoine

```
(sterben@Sterben)-[/var/www]
$ sudo adduser --home /var/www/ --shell /bin/false --ingroup www-data cathy
```

Figure 27: Commande pour ajouter l'utilisateur Cathy

```
(sterben@Sterben)-[/var/www]
$ sudo chgrp -R www-data /var/www/
```

Figure 26: Rajoute l'autorisation groupe au dossier `www`

```
(sterben@Sterben)-[/var/www]
$ sudo chmod -R 775 /var/www
```

Figure 28: Mise à jour de l'autorisation au dossier `www`

Ensuite, nous avons utilisé FileZilla pour nous connecter en tant utilisateur Cathy pour vérifier les droits d'lecture/écriture.

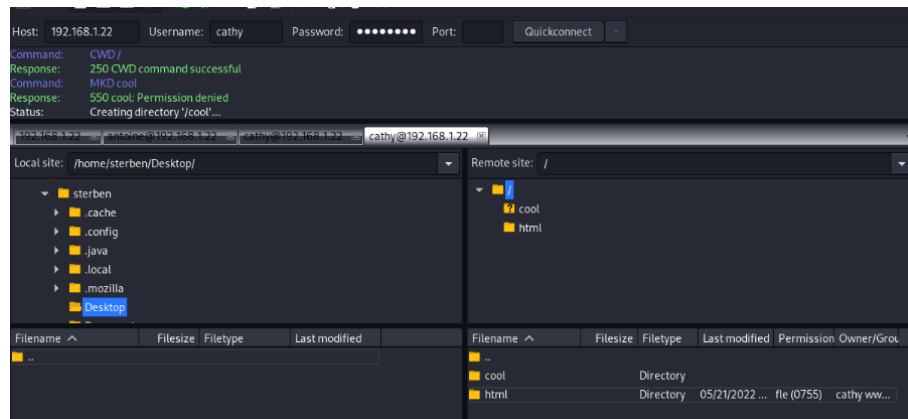


Figure 29: L'utilisateur Cathy peut lire/écrire sur FileZilla

Nous pouvons voir que Cathy a les droits de lecture/écriture et que nous pouvons créer un dossier nommé /cool.

IV.2 Serveur web

Nous avons également déployé un serveur web apache2 ainsi qu'un proxy Squid sur une machine Debian. Nous n'avons pas documenté correctement le déploiement sur le matériel physique, c'est pour cela que nous avons déployé ces services dans un conteneur LXC Debian sur une machine personnelle.

L'installation du serveur apache est très aisée et rapide, sur Debian, il suffit d'installer le paquet apache2 et d'activer le service avec ces commandes :

```
root@apache-serv:~# apt install apache2
root@apache-serv:~# systemctl enable --now apache2
```

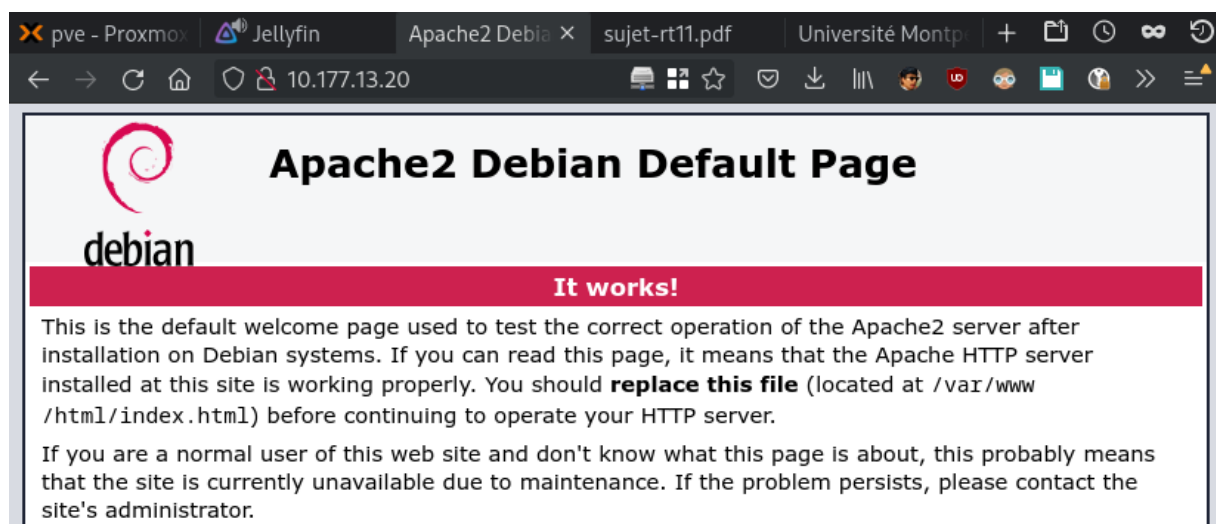


Figure 30 : Page par défaut d'apache

Comme nous pouvons le voir sur cette capture d'écran, la connexion au serveur HTTP fonctionne sans problèmes.

La mise en place du serveur proxy Squid a nécessité de le configurer. Nous avons donc supprimé la configuration originale pour utiliser une configuration simplifiée.

```
root@www-proxy:/etc/squid# cat squid.conf
#Proxy : squid 4/8
# Cette première ligne veut dire que Squid écoute sur le port 3128
http_port 3128
# Cette ligne affichera le nom de machine spécifié lors des messages d'erreurs
visible_hostname squid.lan
# Access List. Ici on crée un groupe qui sera utilisé pour gérer l'IPsource des clients qui
# utiliserons le proxy et un groupe général.
acl localnet src 10.177.13.0/24
acl all src all
# Squid fonctionne un peu comme Iptables. La première règle qui est concordante avec
# le paquet qui arrive sera utilisée et n'ira pas plus loin. Ainsi donc, dans notre cas, l'ACL
# qui regroupe les ip du lan seront autorisées et toutes les autres refusées
http_access allow localnet
http_access deny all
# Spécifie le chemin vers les logs d'accès créé pour chaque page visitée
access_log /var/log/squid/access.log
# Indique à Squid d'attendre 4 secondes avant de se couper quand on essaye de le
# stopper ou de le redémarrer. Par défaut c'est 30 secondes
shutdown_lifetime 4 secondes
```

Figure 31: Fichier squid.conf

Dans cette configuration, nous spécifions que le serveur proxy doit utiliser le port 3128, nous précisons l'adresse de réseau, et nous spécifions l'emplacement des logs d'accès.

Après avoir redémarré le service squid avec systemctl, nous pouvons vérifier que le port 3128 est bien en état de LISTEN.

```
root@www-proxy:~# ss -ltp
State Recv-Q Send-Q Local Address:Port Peer Address:Port Process
LISTEN 0 4096 127.0.0.53%lo:domain 0.0.0.0:* users:(("systemd-resolve",pid=89,fd=18))
LISTEN 0 100 127.0.0.1:smtp 0.0.0.0:* users:(("master",pid=367,fd=13))
LISTEN 0 4096 0.0.0.0:5355 0.0.0.0:* users:(("systemd-resolve",pid=89,fd=12))
LISTEN 0 511 *:http *:~ users:(("apache2",pid=160,fd=4),("apache2",pid=161,fd=5))
LISTEN 0 4096 *:ssh *:~ users:(("systemd",pid=1,fd=23))
LISTEN 0 256 *:3128 *:~ users:(("squid",pid=605,fd=12))
LISTEN 0 100 [:::1]:smtp [:::]:* users:(("master",pid=367,fd=14))
LISTEN 0 4096 [:::]:5355 [:::]:* users:(("systemd-resolve",pid=89,fd=14))
```

Figure 32: Etat des ports de la machine

Après avoir configuré un navigateur pour utiliser le proxy, nous pouvons vérifier le bon fonctionnement du proxy en vérifiant le fichier log.

```

root@www-proxy:/etc/squid# cat /var/log/squid/access.log
1654524970.252 51 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524971.202 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524972.204 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524974.467 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524977.208 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524977.481 96 10.177.13.238 TCP_TUNNEL/200 39 CONNECT ac.duckduckgo.com:443 - HIER_DIRECT/52.142.124.215 -
1654524977.601 62 10.177.13.238 TCP_TUNNEL/200 39 CONNECT ac.duckduckgo.com:443 - HIER_DIRECT/52.142.124.215 -
1654524979.212 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524979.213 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524983.232 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524984.232 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524985.234 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524987.241 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524991.127 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524991.127 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524992.127 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524995.142 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524997.139 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524998.172 290 10.177.13.238 TCP_MISS/200 1151 GET http://example.org/ - HIER_DIRECT/2606:2800:220:1:248:1893:
1654524998.491 125 10.177.13.238 TCP_MISS/404 1142 GET http://example.org/favicon.ico - HIER_DIRECT/2606:2800:220:
1654524999.139 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654524999.139 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654525003.146 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654525004.146 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -
1654525006.147 0 10.177.13.238 NONE/503 0 CONNECT pve:8006 - HIER_NONE/- -

```

Figure 33: Log d'accès du proxy

Après avoir ouvert le site `example.org`, nous pouvons bien le voir apparaître dans les logs. Le serveur proxy fonctionne donc correctement.

V/ Conclusion

Au cours de cette SAÉ, nous avons déployé deux commutateurs, un routeur ainsi que des PC. Nous les avons ensuite configurés comme demandé dans l'énoncé. Nous avons commencé par mettre en place physiquement le réseau local et nous avons ensuite réfléchi à l'attribution des IP. Ensuite, nous avons configuré les commutateurs et nous avons attribué leurs VLANs. Après cela, nous avons mis en place le routage inter-vlan, configuré la mise en miroir des ports pour écouter tout le trafic d'un VLAN, puis nous avons configuré l'accès SSH sur les commutateurs et le routeur. Nous avons mis en place le DHCP sur le routeur pour attribuer une configuration réseau automatique aux PC des VLANs. Puis nous avons mis en place l'OSPF pour annoncer nos réseaux pour nous permettre de communiquer avec les réseaux de nos camarades. Ensuite, nous avons utilisé une machine Kali Linux pour faire de l'empoisonnement ARP avec Ettercap. Ensuite, nous avons mis en place la sécurité sur les deux commutateurs pour bloquer les attaques de ce type. Pour finir, nous avons déployé un serveur FTP utilisant ProFTPD, puis un serveur HTTP ainsi qu'un proxy Squid.

Cette SAÉ nous a permis d'approfondir notre connaissance des réseaux, ainsi que notre maîtrise de l'architecture réseau Cisco. Travailler en autonomie a développé notre capacité à résoudre des problèmes par nous-même. La rédaction de ce rapport, nous a permis d'appliquer nos cours de traitement de texte, notamment avec la création de la bibliographie.

Références

- [1] Clemanet, «Configuration d'un accès ssh - routeur Cisco,» [En ligne]. Available: <https://routeur.clemanet.com/configuration-ssh.php>.
- [2] IETF, «RFC2328: OSPFv2,» Avril 1998. [En ligne]. Available: <https://www.rfc-editor.org/rfc/rfc2328>. [Accès le 2022].
- [3] P. R. Tadimety, OSPF: A Networking Routing Protocol, Apress, 2015.
- [4] Wikipedia, «Algorithme de parcours en largeur,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Algorithme_de_parcours_en_largeur. [Accès le 1 Juin 2022].
- [5] Wikipedia, «Depth-first search,» [En ligne]. Available: https://en.wikipedia.org/wiki/Depth-first_search. [Accès le 1 Juin 2022].
- [6] Wikipedia, «Algorithme de Dijkstra,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Algorithme_de_Dijkstra. [Accès le 01 Juin 2022].
- [7] K. P. Sahoo, «How OSPF protocol implements Dijkstra Algorithm,» 02 Août 2021. [En ligne]. Available: <https://medium.com/@kp-the-great/how-ospf-protocol-implements-dijkstra-algorithm-53c390199ee8>. [Accès le 1 Juin 2022].
- [8] M. Mills, «ARP Poisoning Attack: How to Do It on Kali Linux,» 18 août 2021. [En ligne]. Available: <https://itigic.com/arp-poisoning-attack-how-to-do-it-on-kali-linux/>.