

# Cours réseau

Vigneau FR<sup>1</sup>

Édition 02/2005

<sup>1</sup>Cours de M.Lalitte



# Table des matières

0.1	Historique . . . . .	9
0.1.1	L'idée révolutionnaire . . . . .	9
0.1.2	Le modèle de Baran . . . . .	9
0.1.3	L'ARPANET . . . . .	9
0.1.4	Le courrier électronique . . . . .	9
0.2	Démarche des universitaires . . . . .	10
<b>1</b>	<b>La couche 1</b>	<b>11</b>
1.1	Le câblage . . . . .	11
1.1.1	Les paires torsadées . . . . .	11
1.1.2	Les paires torsadées blindées . . . . .	12
1.1.3	Le câble coaxial avec prise BNC . . . . .	12
1.1.4	La fibre optique . . . . .	13
1.1.5	Les ondes hertziennes . . . . .	13
1.1.6	Les lasers . . . . .	13
1.2	Organisation de la communication . . . . .	13
1.2.1	Le bus . . . . .	14
1.2.2	L'anneau . . . . .	14
1.2.3	L'étoile . . . . .	15
1.3	Les collisions sur un bus . . . . .	15
1.4	Le branchement de deux machines . . . . .	17
<b>2</b>	<b>La couche 2</b>	<b>19</b>
2.1	Le protocole Ethernet et l'adresse MAC . . . . .	19
2.2	La trame Ethernet . . . . .	19
2.2.1	Description de la trame . . . . .	19
2.2.2	Taille de la trame . . . . .	20
2.3	Les collisions sur un HUB . . . . .	20
2.4	Collisions sur un switch . . . . .	22
2.4.1	Présentation du problème . . . . .	22
2.4.2	Conclusion . . . . .	22
2.5	Les VLAN - Virtual Local Area Network -. . . . .	23
2.5.1	Intérêt du VLAN . . . . .	23
2.5.2	Ports Trunk . . . . .	23
2.5.3	Attaque possible . . . . .	23
<b>3</b>	<b>La couche 3</b>	<b>25</b>
3.1	Adresses IP réseau et adresses IP machine . . . . .	25
3.1.1	L'adresse IP . . . . .	25
3.1.2	Le masque . . . . .	25

3.2	Le datagramme ou paquet . . . . .	26
3.2.1	Le datagramme . . . . .	26
3.3	Le routage . . . . .	27
3.3.1	Table de routage . . . . .	27
3.3.2	Exercice . . . . .	28
3.3.3	Évolution des tables de routage . . . . .	29
3.4	Adresses particulières . . . . .	29
3.4.1	Adresses réservée . . . . .	29
3.4.2	Exemple . . . . .	29
3.5	Gestion des adresses IP . . . . .	30
3.5.1	Problématique . . . . .	30
3.5.2	Passage à IPv6 . . . . .	30
3.5.3	L'adressage CIDR . . . . .	30
3.5.4	Regroupement géographique d'adresses . . . . .	30
3.5.5	NAT - Network Address Translation-. . . . .	30
3.5.6	Exercice pratique . . . . .	31
3.6	Exercice . . . . .	32
3.6.1	Énoncé . . . . .	32
3.6.2	Correction . . . . .	32
3.6.3	Obtention des adresses MAC . . . . .	33
3.7	ICMP . . . . .	33
3.7.1	Messages ICMP . . . . .	34
3.7.2	Outils utilisant les messages ICMP . . . . .	34
<b>4</b>	<b>La couche 4</b>	<b>35</b>
4.1	Les protocoles de couche 4 . . . . .	35
4.2	Les adresses de couche 4 . . . . .	35
4.3	Le protocole UDP . . . . .	36
4.3.1	Retour sur la notion de datagramme . . . . .	36
4.3.2	Datagramme UDP . . . . .	36
4.3.3	Établissement et rupture de la communication . . . . .	36
4.4	Protocole TCP . . . . .	36
4.4.1	Établissement et rupture de la communication . . . . .	36
4.4.2	Segment TCP . . . . .	37
4.4.3	Remarque . . . . .	37
<b>5</b>	<b>Le protocole DNS</b>	<b>39</b>
5.1	Qu'est ce que le DNS ? . . . . .	39
5.2	Historique . . . . .	39
5.3	Organisation du système DNS . . . . .	39
5.4	Attaque contre le système DNS . . . . .	41
5.5	Résolution d'un nom . . . . .	41
5.5.1	Requête DNS . . . . .	41
5.5.2	Exemple . . . . .	42
5.5.3	Exemple d'attaque . . . . .	42
5.6	Types de requêtes DNS . . . . .	42
5.6.1	Requête récursive ou requête itérative . . . . .	42
5.6.2	Exemple . . . . .	42
5.7	Types d'enregistrements . . . . .	43
5.8	Types de serveurs . . . . .	44

5.8.1	Deux types de serveurs DNS . . . . .	44
5.8.2	Questions . . . . .	44
5.9	Les outils DNS . . . . .	44
5.9.1	Au niveau de la machine . . . . .	44
5.9.2	Les outils serveur . . . . .	44
<b>6</b>	<b>Retour sur le protocole IP</b>	<b>47</b>
6.1	L'en-tête IP . . . . .	47
6.1.1	Description de l'en-tête . . . . .	47
6.1.2	Remarques sur l'ID et le Fragment offset . . . . .	47
6.2	La fragmentation . . . . .	49
6.2.1	Présentation . . . . .	49
6.2.2	Champs de l'en-tête IP concernant la fragmentation . . . . .	49
6.2.3	Question . . . . .	50
6.3	Exercice d'application . . . . .	50
6.3.1	Énoncé . . . . .	50
6.3.2	Corrigé . . . . .	50
6.4	Gestion du flag DF . . . . .	51
6.5	Exemple d'attaque utilisant la fragmentation . . . . .	52
6.6	Influence de la MTU . . . . .	52
<b>7</b>	<b>Retour sur le protocole TCP</b>	<b>53</b>
7.1	Description de l'en-tête TCP . . . . .	53
7.2	Mécanisme de contrôle d'acheminement . . . . .	53
7.2.1	Exemple . . . . .	54
7.2.2	Description . . . . .	54
7.3	L'attaque de Noël . . . . .	55
7.3.1	Historique . . . . .	55
7.3.2	Stratégie . . . . .	55
7.3.3	Déroulement de l'attaque . . . . .	55
7.3.4	Remarque . . . . .	56
7.3.5	La parade . . . . .	56
<b>8</b>	<b>La traduction d'adresse : NAT</b>	<b>57</b>
8.1	Introduction . . . . .	57
8.2	La NAT statique . . . . .	57
8.2.1	Le principe . . . . .	57
8.2.2	Intérêt de la NAT statique . . . . .	57
8.2.3	Le fonctionnement de la NAT statique. . . . .	57
8.2.4	Avantages et inconvénients de la NAT statique. . . . .	58
8.2.5	Problèmes de routage liés à l'utilisation de la NAT statique -proxy ARP-. . . . .	58
8.3	La NAT dynamique . . . . .	59
8.3.1	Le principe . . . . .	59
8.3.2	Le fonctionnement de la NAT dynamique . . . . .	60
8.3.3	résumé . . . . .	60
8.3.4	Problèmes liés à la NAT dynamique . . . . .	61
8.4	Rendre les machines du réseau local joignables malgré la NAT . . . . .	61
8.4.1	Le port forwarding . . . . .	62
8.4.2	Le port mapping . . . . .	62
8.4.3	Les limites du port forwarding . . . . .	62

8.5	Les proxys . . . . .	62
8.5.1	Présentation . . . . .	62
<b>9</b>	<b>Les Firewalls</b>	<b>63</b>
9.1	Introduction . . . . .	63
9.2	Les différents types de firewall . . . . .	63
9.2.1	Le filtrage simple . . . . .	63
9.2.2	Le filtrage par états . . . . .	63
9.2.3	Les proxys . . . . .	63
9.3	Le filtrage sous Linux . . . . .	63
9.3.1	Fonctionnement de Netfilter . . . . .	64
9.3.2	Le fonctionnement de iptables . . . . .	65
9.3.3	La traduction d'adresses . . . . .	65
9.4	Liste des principaux ports . . . . .	66
9.5	Une configuration standard . . . . .	66
<b>10</b>	<b>Introduction au protocole SSH</b>	<b>67</b>
10.1	Définition . . . . .	67
10.2	Quelle est l'utilité de SSH ? . . . . .	67
10.3	Limite des firewalls . . . . .	68
10.4	Les avantages du protocole SSH . . . . .	68
10.4.1	Les avantages fonctionnels . . . . .	68
10.4.2	Les avantages techniques . . . . .	68
10.5	Fonctionnement d'une authentification par clé RSA . . . . .	68
10.6	Les fichiers importants . . . . .	69
10.7	Réalisation pratique . . . . .	69
10.7.1	Les demandes de passphrases . . . . .	69
10.7.2	Création et copie des clés . . . . .	69
10.7.3	Debugger le serveur . . . . .	69
10.7.4	Durcir le fichier de configuration . . . . .	70
<b>A</b>	<b>Exercices</b>	<b>71</b>
A.1	La couche 3 . . . . .	71
A.1.1	Exercice 1 . . . . .	71
A.1.2	Exercice 2 . . . . .	71
A.1.3	Exercice 3 . . . . .	71
A.1.4	Exercice 4 . . . . .	72
A.1.5	Exercice 5 . . . . .	72
A.1.6	Exercice 6 . . . . .	73
A.1.7	Exercice 7 . . . . .	73
A.2	La couche 4 . . . . .	73
A.2.1	Exercice 1 . . . . .	73
A.2.2	Exercice 2 . . . . .	74
A.3	La NAT . . . . .	76
A.3.1	Exercice 1 . . . . .	76

<b>B</b>	<b>La transmission en bande de base</b>	<b>77</b>
B.1	Introduction . . . . .	77
B.2	Codage des signaux . . . . .	77
B.3	Le codage NRZ . . . . .	77
B.4	Le codage NRZI . . . . .	78
B.5	Le codage Manchester . . . . .	78
B.6	Le codage delay . . . . .	79
B.7	Le codage bipolaire . . . . .	79
<b>C</b>	<b>Fibres optiques</b>	<b>81</b>
C.1	Principe de fonctionnement . . . . .	81
C.2	Les trois types de fibre optique . . . . .	82
C.3	Les connections . . . . .	84





# Introduction

## 0.1 Historique

### 0.1.1 L'idée révolutionnaire

En 1962, en pleine guerre froide, consciente de la vulnérabilité de son organisation centralisée, l'US Air Force demande à un petit groupe de chercheurs de créer un réseau de communication militaire capable de résister à une attaque nucléaire. Le concept de ce réseau reposait sur un système décentralisé, permettant au réseau de fonctionner malgré la destruction de une ou plusieurs machines.

### 0.1.2 Le modèle de Baran

Paul Baran est considéré comme un des acteurs principaux de la création d'Internet. Il eu l'idée, en 1964, de créer un réseau sous forme de grande toile. Il avait réalisé qu'un système centralisé était vulnérable car la destruction de son noyau provoquait l'anéantissement des communications. Il mit donc au point un réseau hybride d'architectures étoilées et maillées dans lequel les données se déplaceraient de façon dynamique, en « cherchant » le chemin le moins encombré, et en « patientant » si toutes les routes étaient encombrées. Cette technologie fut appelé « packet switching ».

### 0.1.3 L'ARPANET

En 1969, le réseau expérimental ARPANET fut créé par l'ARPA - Advanced Research Projects Agency dépendant du DOD, Department of Defense - afin de relier quatre instituts universitaires :

- le Stanford Institute,
- l'université de Californie à Los Angeles,
- l'université de Californie à Santa Barbara,
- l'université d'Utah.

Le réseau ARPANET est aujourd'hui considéré comme le réseau précurseur d'Internet. Il comportait déjà à l'époque certaines caractéristiques fondamentales du réseau actuel :

- Un ou plusieurs noeuds du réseau pouvait être détruits sans perturber son fonctionnement ;
- La communication entre machines se faisait sans machine centralisée intermédiaire ;
- Les protocoles utilisés étaient basiques.

### 0.1.4 Le courrier électronique

En 1972 Ray Tomlinson mit au point un nouveau mode de communication : le courrier électronique. Le contenu de ce premier e-mail était le suivant : QWERTYUIOP Par ailleurs, le caractère « @ » servait déjà à séparer le nom de l'utilisateur du nom de la machine dans les adresses. En juillet 1972, Lawrence G. Roberts améliora les possibilités ouvertes par Ray Tomlinson en développant la première application permettant de lister, de lire de manière sélective, d'archiver, de répondre ou de faire suivre un e-mail. Dès lors, la messagerie électronique n'aura de cesse de croître, pour devenir la principale utilisation du réseau des réseaux au début du XXIe siècle. C'est également en octobre 1972 que le réseau ARPANET

fut présenté pour la première fois au grand public, lors de la conférence ICCC - International Computer Communication Conference -. A cette même époque, l'ARPA devint le DARPA - Defense Advanced Research Projects Agency - et le terme « internetting » est utilisé pour désigner l'ARPANET, devenant alors un embryon d'Internet.

## 0.2 Démarche des universitaires

Pour concevoir ce réseau ils sont partis de l'observation du monde et des moyens de communication de l'époque -téléphone, poste, parole ...- et ils ont fait, pour chacun d'eux, l'inventaire de ce qui était nécessaire à leur mise en oeuvre. Ils ont ensuite fusionné ces modèles pour déterminer des caractéristiques générales :

**le média** : fil, air...

**le protocole** : langage.

**l'émetteur**

**le récepteur**

**le contenant** : enveloppe, facteur, adresse...

**le contenu** : l'information.

Les chercheurs ont alors considéré que, pour communiquer, il y avait autant de tâches à accomplir que d'éléments génériques dans cette liste. Ils ont alors imaginé un modèle en couches, où chacune correspond à une tâche. L'intérêt majeur de cette approche est le suivant : si une couche a été mal implémentée, ou nécessite une modification en raison des évolutions techniques, il n'y a qu'elle à changer. Ceci induit une contrainte forte : les couches doivent être indépendantes. Ce modèle s'appelle le modèle OSI<sup>1</sup> - Open System Interconnexion -. Il comporte sept couches : quatre pour le réseau et trois pour les applications.

7	APPLICATION
6	PRÉSENTATION
5	SESSION
4	TRANSPORT
3	RÉSEAU
2	LIAISON
1	PHYSIQUE

**La couche 1** offre le média, le support de transmission.

**La couche 2** permet de gérer les connexions en réseau local : elle définit le langage nécessaire pour que deux machines inter-connectées puissent communiquer.

**La couche 3** permet le dialogue entre réseaux locaux.

**La couche 4** gère les connexions applicatives, elle permet de faire communiquer deux applications entre elles ce qui est le but du réseau.

En effet le réseau est au service des applications. À l'heure actuelle les couches cinq et six sont intégrées à la couche application. Pour garantir l'indépendance des couches entre elles, une couche ne peut communiquer directement qu'avec une couche contiguë.

---

<sup>1</sup>En fait le modèle OSI date de 1984. Le réseau s'est appuyé sur le modèle TCP/IP dont il est encore aujourd'hui la référence. Ce n'est que par la suite que le modèle OSI est apparu, reprenant les idées du modèle TCP/IP et élargissant les perspectives possibles en terme de réseau

# Chapitre 1

## La couche 1

**Rôle :** Elle est responsable du support de transmission.

### 1.1 Le câblage

#### 1.1.1 Les paires torsadées

C'est un câble constitué de quatre paires torsadées. Il ne faut pas confondre le câble et la prise qui est au bout (prise RJ45).

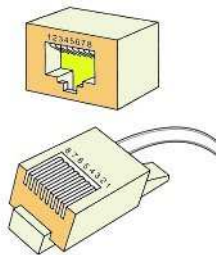


FIG. 1.1 – Prise RJ45

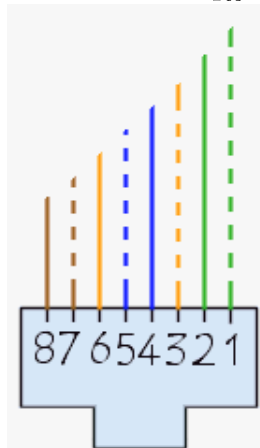


FIG. 1.2 – câblage prise RJ45

- une paire est utilisée en émission (fils 1 et 2).
- une paire est utilisée en réception (fils 3 et 6).
- les fils 4,5,7 et 8 ne sont pas utilisés en standard.

Les paires sont torsadées deux à deux pour diminuer l'influence des perturbations électromagnétiques.

### 1.1.2 Les paires torsadées blindées

Ce sont des paires torsadées dotées d'une gaine métallique entourant le câble. Elles sont nommées :

- 10 BT -lire 10 base T-
- 100 BT
- 100 BTx
- 1000 BT
- 1000 BF

Le chiffre représente le débit maximum supporté en mégabits par seconde. La lettre B désigne la bande de base<sup>1</sup>, c'est la manière dont sont codées les informations qui transitent sur le câble. La dernière lettre désigne le support :

- T : paire torsadée.
- F : fibre optique.

Dans les câbles de type 1000 BT on peut utiliser certains fils non reliés à la base.

### 1.1.3 Le câble coaxial avec prise BNC

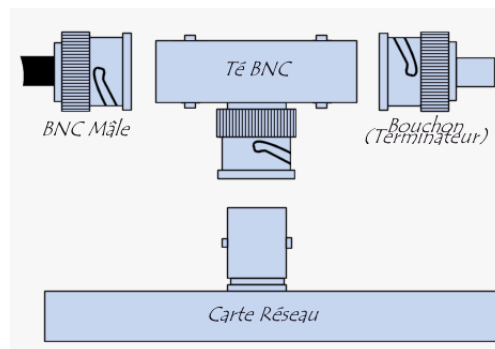


FIG. 1.3 – Prise T BNC

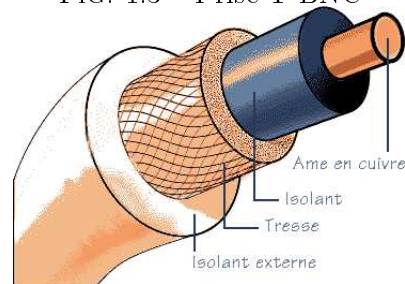


FIG. 1.4 – câble coaxial

Il est nommé :

- 10 B 2
- 10 B 5

Le second est le plus ancien et utilise le phénomène d'onde stationnaire, il faut donc se connecter à certains endroits, au niveau des « ventres » de l'onde, en utilisant des prises vampires. Le dernier chiffre de la dénomination - 2 ou 5 - correspond à la longueur maximale de câble utilisable, exprimée en centaines de mètres - 2 pour 200m -. Le câble coaxial n'est presque plus utilisé de nos jours.

<sup>1</sup>voir B page 77

### 1.1.4 La fibre optique

Elle utilise la propagation de la lumière à l'intérieur de la fibre pour véhiculer l'information. Le principe repose sur la réfraction pour garder le rayon lumineux à l'intérieur de la fibre. Il existe deux méthodes pour y parvenir <sup>2</sup> :

1. Le changement de milieux,
2. la variation continue d'indice.

On distingue deux catégories de Fibre :

1. La fibre monomode : adaptée à une seule longueur d'onde, elle ne peut véhiculer qu'une seule longueur d'onde.
2. La fibre multimode : adaptée à la lumière blanche, qui peut véhiculer des ondes de différentes longueurs d'onde.

L'atténuation est plus faible en monomode car elle est inversement proportionnelle au carré de la longueur d'onde -  $\frac{c}{\lambda^2}$  -. D'autre part l'indice dépend de la longueur d'onde, donc le trajet au sein de la fibre également, on assiste donc à un décalage du signal en fonction de la longueur d'onde sur les fibres multimode.

C'est pourquoi des faisceaux en monomode peuvent être utilisés sur des distances de l'ordre de 60 km alors qu'en multimode la limite se situe à environ 2 km. De plus, le débit peut être augmenté en monomode. En effet, l'atténuation étant plus faible, il est possible de diminuer la longueur du train d'onde - donc l'énergie émise - en gardant un rapport signal sur bruit suffisant en sortie pour continuer à le détecter. Ceci permet de faire passer, à un instant  $t$ , plus de trains d'onde dans une même longueur de fibre. Enfin on peut réaliser du multiplexage de longueur d'onde en faisant passer sur une onde monomode plusieurs ondes de couleur différente - il faut juste vérifier que pour chaque longueur considérée l'indice permet d'obtenir la réflexion totale dans l'épaisseur de la fibre-. La combinaison de ces méthodes permet d'obtenir des débits de l'ordre de 60 Gigabits par seconde.

### 1.1.5 Les ondes hertziennes

### 1.1.6 Les lasers

On utilise un faisceau laser pour propager, en air libre et en portée optique, un train d'onde. Ce vecteur est réservé aux courtes distances car il est très sensible aux conditions atmosphériques.

## 1.2 Organisation de la communication

La couche 1 assure le support de la communication, mais aussi organise cette communication entre les machines sur le réseau - la topologie réseau ou comment connecter les machines-.

On a trois topologies différentes : le bus, l'anneau et l'étoile.

---

<sup>2</sup>voir annexe C page 81

### 1.2.1 Le bus

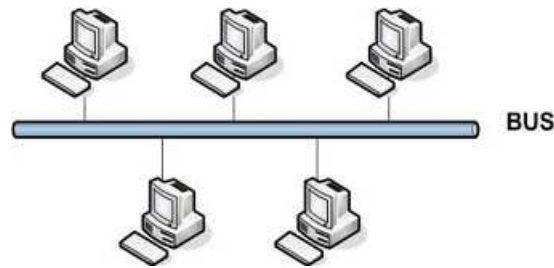


FIG. 1.6 – Topologie bus

C'est un câble sur lequel toutes les machines sont connectées. On place à chaque extrémité du bus un bouchon destiné à prévenir les réflexions en bout de câble. Si deux personnes parlent à la fois sur le réseau il y a interférence entre les deux messages, qui sont donc perdus. Ce phénomène est appelé collision. Pour l'éviter une seule personne doit parler à la fois sur le bus. Donc plus il y a de machines, plus les communications sont difficiles.

#### Limitations

1. Nombre de machines : lié à l'encombrement du réseau,
2. longueur de bus : cette limitation est liée à l'atténuation le long du bus et à la détection des collisions -cf couche 2-.

### 1.2.2 L'anneau

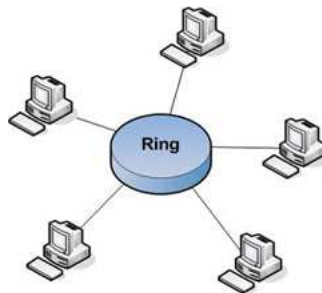


FIG. 1.7 – Topologie anneau

Les machines sont reliées à un câble circulaire -d'où l'appellation d'anneau-. Pour prévenir les collisions un « jeton » circule sur l'anneau. Quand une machine veut parler, elle prend le jeton, le remplit avec les données, et l'envoi sur le réseau. Il est déchargé par la machine destinataire puis remis à disposition du réseau.

#### Limitations

1. Nombre de machines : lié à l'encombrement du réseau -une machine parle à la fois-,
2. longueur de bus : cette limitation est liée à l'atténuation le long de l'anneau et au temps de circulation du jeton.

### conclusion

Bien qu'une seule machine soit capable de parler en même temps sur le réseau le débit obtenu est supérieur à celui du bus.

### 1.2.3 L'étoile

Les machines sont reliées à un noeud central. En fonction de l'équipement réalisant ce noeud on peut faire communiquer les machines deux à deux sans interférence.

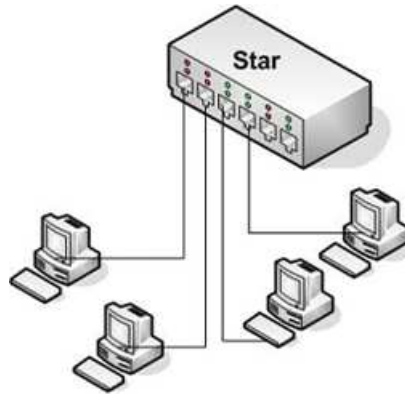


FIG. 1.8 – Topologie étoile

### Limitations

1. Nombre de machines : liée au type d'équipement réalisant la connexion centrale,
2. longueur de bus : cette limitation est liée à l'atténuation le long du câble reliant la machine au noeud central - de l'ordre de 100 m -.

### conclusion

Aujourd'hui la topologie en étoile est la plus développée car elle est la plus souple d'emploi. Il est également possible de réaliser des topologies hybrides.

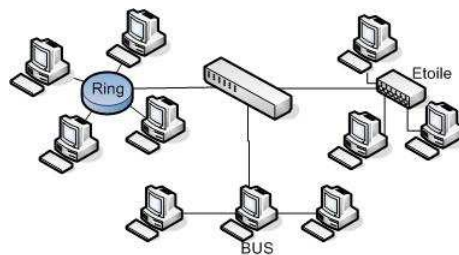


FIG. 1.9 – Topologie hybride

## 1.3 Les collisions sur un bus

Une collision est l'interférence entre deux paquets émis par deux machines différentes. Il en résulte la perte des informations transmises, il faut donc les traiter. Pour gérer les collisions on utilise la méthode CSMA/CD - Carrier Sens Multiple Access / Collision Detection -. Celle-ci consiste à écouter en permanence le bus - pour détecter les collisions et l'occupation du réseau - pour parler quand le bus

est libre. Après l'émission on écoute les éventuelles collisions. Si une collision est détectée le paquet est ré-émis après un temps aléatoire - ce qui permet de séparer les ré-émissions des différentes machines -.

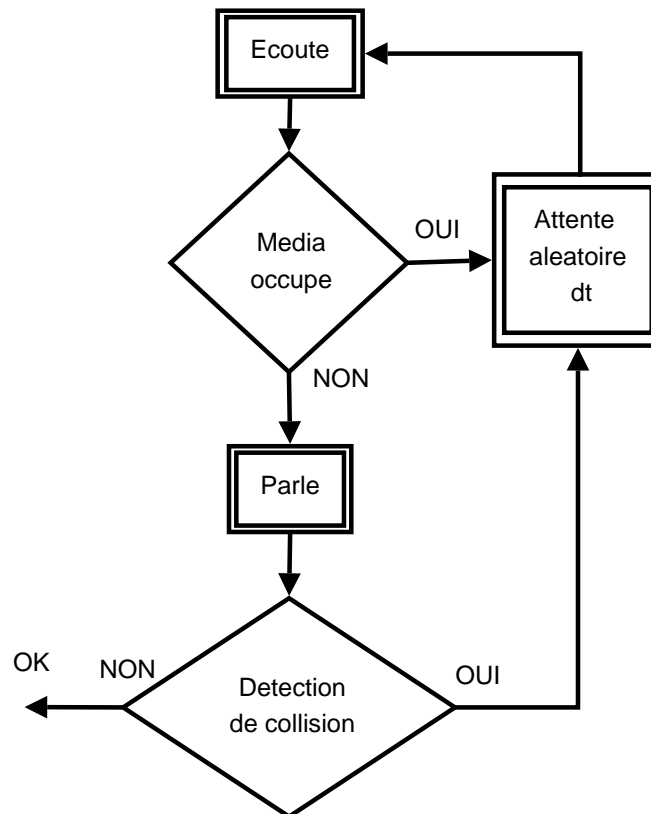


FIG. 1.10 – Protocole CSMA / CD

**Problème :** Plus il y a de machines, plus le temps d'attente risque d'être long, et plus il y a de chances qu'il y ait deux machines qui parlent en même temps après une collision -donc qui aient des  $\delta t$  avant réémission proches -, générant ainsi une nouvelle collision. À partir d'un certain nombre de machines,  $\approx 60$ , il y a plus de temps passé à gérer les collisions qu'à transmettre des informations utiles !



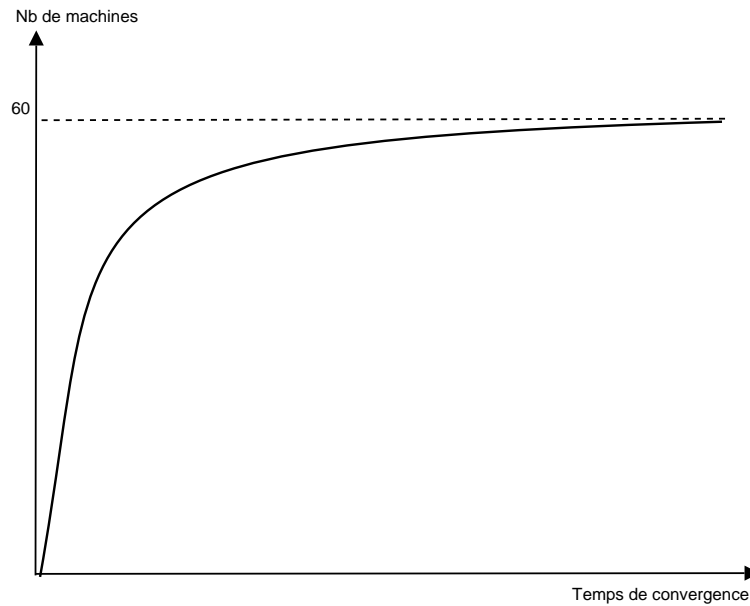


FIG. 1.11 – Collisions sur un bus

## 1.4 Le branchement de deux machines

Le branchement est réalisé à l'aide d'un câble de paires torsadées. Mais si les deux machines sont branchées directement, considérant qu'une paire est utilisée en émission et une paire en réception, il ne faut pas oublier de croiser les paires. Ainsi la machine A émet sur la paire 1 qui est branchée en réception sur la machine B et inversement.



FIG. 1.12 – Câble croisé

En revanche si on est branché à un hub on utilise un câble droit, c'est le hub qui est câblé croisé en interne pour assurer la liaison entre les machines. Il est plus simple d'utiliser un câble droit et il y a plus de connexions entre matériels de types différents qu'entre matériels de même type, c'est pourquoi cette convention a été retenue.

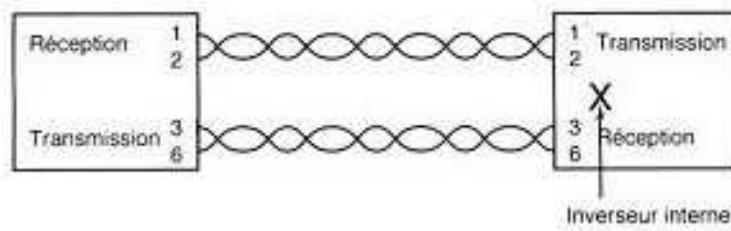


FIG. 1.13 – Utilisation des câbles droits  
Cable croise

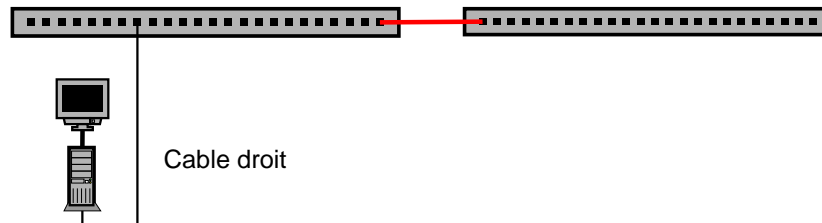


FIG. 1.14 – Utilisation des câbles croisés

Donc j'utilise un câble croisé entre deux matériels de même type :

- pour connecter un hub et un switch,
- une machine à un routeur,
- ...

et un câble droit entre deux matériels de type différent :

- pour connecter une machine à un switch,
- un routeur à un switch,
- deux switches,
- ...

# Chapitre 2

## La couche 2

**Rôle :** Elle est chargée de gérer les connexions sur un réseau local, et de détecter les erreurs de transmission. Il faut définir un protocole, une norme, qui fixe les règles de communication. Les protocoles les plus répandus sont : Ethernet, Token-ring et FDDI.

### 2.1 Le protocole Ethernet et l'adresse MAC

Il existe un identifiant de chaque matériel ou machine : *l'adresse MAC -Medium access control-*. Elle se compose de six octets, et s'écrit en hexadécimal, les octets étant séparés par le caractère « : ». *Exemple : 12 :B4 :C8 :EF :22 :C6* Les trois premiers octets correspondent au constructeur, et les trois derniers au matériel. L'adresse MAC de chaque matériel est unique dans le monde. Pour garantir cette unicité les trois premiers octets identifient le constructeur et celui ci gère les trois derniers qui identifie un matériel donné - chaque machine possède donc un identifiant unique-. Lorsque le constructeur a épuisé toutes les adresses qui lui étaient dévolues un nouveau triplet constructeur lui est attribué.  $2^{6*8} = 2^{48}$  adresses MAC sont ainsi disponibles. En fait il y en a une de moins, l'adresse nulle étant exclue. Une adresse joue est réservée et joue un rôle particulier : l'adresse de broadcast, *FF :FF :FF :FF :FF :FF*, qui permet d'adresser toutes les machines d'un réseau local.

### 2.2 La trame Ethernet

#### 2.2.1 Description de la trame

@MAC dest	@MAC src	Protocole 3	?	DATA	CRC
-----------	----------	-------------	---	------	-----

TAB. 2.1 – Trame Ethernet

Les champs @MAC dest, @MAC src, Protocole 3 et CRC constituent l'en tête Ethernet, elle mesure 18 octets. Description des champs :

**@MAC dest** : est l'adresse MAC de la machine destinataire - 6 octets -,

**@MAC src** : est l'adresse MAC de la machine émettrice - 6 octets -,

**Protocole 3** : indique le protocole utilisé par la couche supérieure pour coder les informations. Cette information est nécessaire pour que la trame DATA soit bien décodée à la réception - 2 octets -,

**?** : il y a eu plusieurs normes Ethernet, et dans certaines il y avait des informations à cet emplacement.

Avec la norme actuelle - norme RFC 802 - ce n'est plus le cas. En fait il y a plusieurs normes désignées RFC 802.xx, où xx désigne la norme particulière adaptée à un type de réseau local donné - wi fi par exemple -.

**CRC** : c'est un code de contrôle d'erreur dont le calcul s'effectue au fur et à mesure que la trame est envoyée. Il n'est donc disponible qu'une fois la trame entièrement réalisée, c'est pourquoi il est placé en queue de trame, après le DATA. À la réception le calcul s'effectue de la même façon au cours de la réception puis le résultat est comparé à la valeur stockée dans le code CRC de la trame. Ce calcul se faisant bit à bit il est nécessaire que le résultat soit à la fin. Ce code permet de détecter les erreurs mais pas de les corriger. Ce champs occupe 4 octets.

### 2.2.2 Taille de la trame

Nous avons déjà vu que l'en tête faisait dix huit octets. La norme définit une taille maximale et une taille minimale. La limite supérieure, définie pour ne pas monopoliser le réseau, est fixée à 1518 octets - 1500 octets de données et 18 octets d'en tête -. Cette taille maximale peut être modifiée. La taille minimale - 64 octets, 46 octets de données et 18 octets d'en tête - a été fixée pour que, sur un bus, le premier octet émis ait atteint la destination la plus lointaine du réseau avant que la trame ne soit entièrement émise. Cette restriction permet de détecter les collisions.

## 2.3 Les collisions sur un HUB

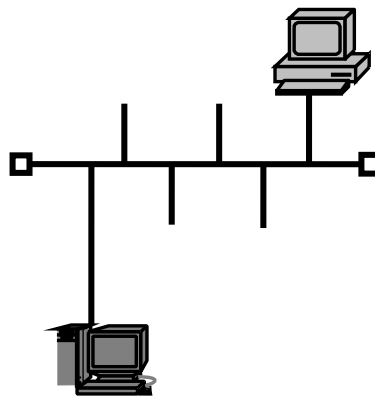


FIG. 2.1 – collision sur un bus

Quand la machine A émet une trame, celle-ci est re-dirigée au niveau du hub vers tous les ordinateurs. Ceux-ci la détectent, regardent si elle les concerne - grâce à l'adresse MAC de destination -, et seule la machine concernée poursuit le décodage.

Une amélioration possible est d'utiliser, pour le coeur du réseau, un matériel capable de lire le protocole de couche deux - le protocole Ethernet -, donc l'adresse MAC de destination. Il est alors capable de distribuer la trame lui même vers le bon destinataire plutôt que de l'envoyer à toutes les machines. Un tel équipement s'appelle un *SWITCH* - ou commutateur -. C'est un matériel de couche 2 - car il lit le protocole de couche 2 -. Le hub - ou concentrateur - est lui un équipement de couche 1 - c'est un câble qui est incapable de lire le protocole de couche 2 et travaille au niveau électrique uniquement -.

Pour re-diriger la trame, le switch dispose en mémoire d'une table, dite table CAM, qui met en correspondance une adresse MAC et un numéro de port du switch. Pour établir cette table il lit l'adresse source des paquets qui lui proviennent et les associe avec le numéro du port sur lequel le paquet est arrivé. C'est pourquoi l'adresse MAC de destination est placée en début de trame, ainsi les équipements de couche 2 ne perdent pas de temps à lire des informations inutiles pour la commutation.

Donc, si un pirate P émet une trame qui comporte en adresse source l'adresse MAC d'une autre machine V, le processus de remplissage de la table CAM étant par défaut dynamique, le switch va mettre à jour cette table en associant @MAC de V avec port du switch de P. Il est donc possible de leurrer un switch. Cependant cette modification ne perdurera que jusqu'à ce que V émette une trame, entraînant à nouveau la mise à jour de la table. Toutefois, en émettant suffisamment souvent avec l'adresse MAC de V en adresse source P peut rafraîchir régulièrement la table et rendre sa modification pérenne. De cette façon le pirate reçoit toutes les trames destinées à V, à sa place. Si la table CAM comporte trop d'entrées elle finit par saturer et le switch se comporte alors comme un hub, ce qui peut être intéressant pour espionner le réseau. Amener la table CAM du switch à saturation en envoyant plein de trame avec des adresses MAC différentes s'appelle *flooder* le switch. Si le switch ne trouve pas l'adresse MAC du destinataire dans sa table CAM il transmet la trame à tous les ports - broadcast -. Il est possible d'avoir plusieurs adresses MAC correspondant à un même port. *Exemple* : Si je relie deux switch entre eux.

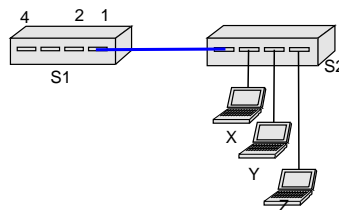


FIG. 2.2 – switch

Dans ce cas la table CAM de S1 comportera :

Port 1	@MAC X
	@MAC Y
	@MAC Z

TAB. 2.2 – Table CAM

Le switch S1 ne sait pas qu'il est connecté à un autre switch, il sait juste qu'il y a plusieurs adresses MAC derrière le port 1. Pour envoyer un message de broadcast il faut utiliser l'adresse de broadcast : FF :FF :FF :FF :FF :FF. Afin d'augmenter la disponibilité du réseau, une idée pourrait être de relier deux switch par deux câbles. Ainsi en cas de défaut sur un câble le second continu d'assurer la connexion.

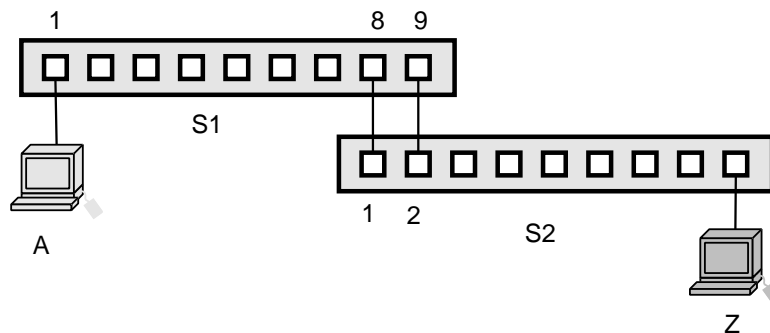


FIG. 2.3 – Boucle de commutation

Cette solution présente un inconvénient majeur :

Dans le cas d'un broadcast de A sur le switch, le message est envoyé via le port 8 du switch1 et le port 1 du switch 2 à tous les ports du switch 2. Celui ci répercute le broadcast à tous ses ports. Donc le port 2 du switch 2 transmet un message de broadcast au switch1 via son port 9. Le switch 1 retransmet alors ce broadcast à tous ses ports. Donc le port 8 le retransmet au switch 2 via son port 1 et ainsi de suite. Le même comportement peut être observé pour le message arrivant sur le port 9 de S1 transmis via le port 2 de S2 à tous les ports de S2 donc au port 1 qui transmet au port 8 de S1 qui broadcast à tous ses ports donc au port 9 etc. Ce phénomène s'appelle le *broadcast storm* - tempête de broadcast - et résulte d'une *boucle de commutation*. La conséquence de ce phénomène est un effondrement du réseau.

## 2.4 Collisions sur un switch

### 2.4.1 Présentation du problème

Le switch répond à la technologie *store and forward*, car il joue le rôle de tampon et les paquets sont traités les uns après les autres et mis en relation directement avec le destinataire. Cependant la mémoire n'est pas infinie et doit être calibrée en fonction de la taille du switch et du débit maximal du réseau. Par exemple, pour un switch de ports à 100 Mbits il faut  $100 * n$  Mo de mémoire. Celle ci est appelée le *fond de panier*.

D'autre part avec des paires torsadées, il y a une paire en émission et une paire en réception, il n'y a donc pas de collision possible.

### 2.4.2 Conclusion

En alliant les paires torsadées et la technologie *store and forward*, il n'y a plus de collisions possibles. Il n'est donc plus nécessaire de les traiter - plus de CSMA/CD -. C'est le *full duplex*. Le taux de transfert est donc amélioré. Avec une carte à 100Mbits le taux de transfert total peut atteindre jusqu'à 200Mbits : 100 en émission et 100 en réception. Le CSMA/CD est lui qualifié de *half duplex*.

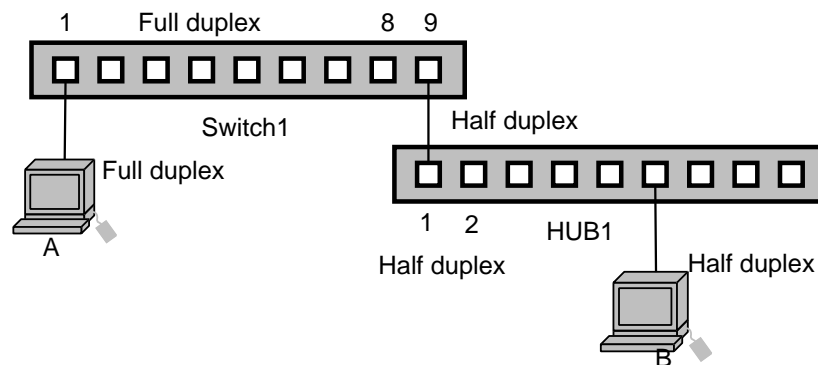


FIG. 2.4 – Half duplex et full duplex

Les ports du switch doivent être configurés un par un, en full ou half duplex, en fonction de l'équipement connecté. Tous les ports d'un hub sont en half duplex, c'est une contrainte matérielle. Pour le switch, seul le port relié au hub voit les collisions, les autres, étant configurés en full duplex, ne les voient pas. Si le débit est faible sur le réseau, c'est souvent un problème de configuration d'un port du switch - mauvaise option entre full et half duplex -. Cette configuration se fait sur le switch lui-même, il est alors qualifié de « switch administrable ».

## 2.5 Les VLAN - Virtual Local Area Network -.

### 2.5.1 Intérêt du VLAN

Aujourd'hui il y a des switches qui comportent 500 ou 1000 ports. Économiquement un gros switch est moins onéreux que plusieurs petits. Une technologie a donc été développée permettant de séparer de façon logique les ports entre eux. Certains ports sont regroupés au sein d'un VLAN et deux VLAN, bien qu'appartenant au même switch, ne peuvent communiquer entre eux.

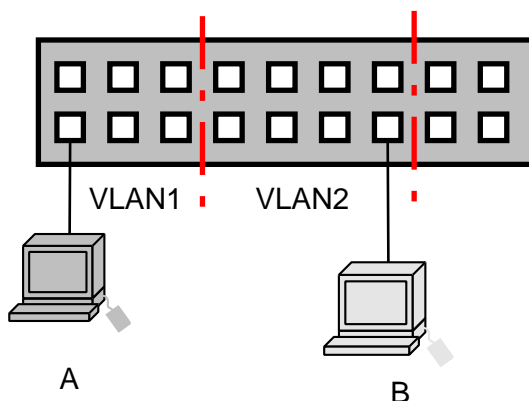


FIG. 2.5 – Les VLAN

Dans ce cas si A veut communiquer avec B c'est impossible. Tout se passe comme si VLAN1 et VLAN2 étaient deux switches différents. De même un broadcast reste confiné sur son VLAN.

### 2.5.2 Ports Trunk

Si, disons une université, dispose d'un côté de deux réseaux nommés Prof1 et élèves1, et dans une autre partie du bâtiment de deux autres réseaux nommés Prof2 et élèves2, il serait intéressant de pouvoir mettre les deux réseaux Prof en communication, ainsi que les réseaux élèves. Il faut donc deux câbles - un par VLAN -, ce qui suppose de tirer ces deux câbles peut être sur de grandes distances. Or nous disposons d'un seul switch de chaque côté il serait donc pratique de pouvoir faire la connexion en utilisant un seul câble reliant les deux switches. Il a donc été créé les *ports Trunk*, il y en a un par switch et tous les VLAN du switch peuvent y accéder, du moins ceux que l'administrateur autorise. Pour garder la confidentialité du VLAN le switch de départ ajoute en tête de trame le numéro du VLAN d'origine : c'est le TAG.

TAG	TRAME ETHERNET
-----	----------------

TAB. 2.3 – VLAN TAG

Intérêt : il n'y a plus qu'un seul câble. Le protocole s'appelle 802.1Q.

Certains constructeurs ont leur propre norme : -CISCO : ISL-, se pose alors le problème de l'interconnexion entre deux switches de marque différentes, si au moins un d'entre eux ne respecte pas la norme 802.1Q.

### 2.5.3 Attaque possible

Si je rajoute à ma trame Ethernet, le TAG de l'autre VLAN, le switch départ rajoute mon TAG à la trame, que le switch arrivé décode, mais comme il ne détecte pas de trame Ethernet il décode le

second TAG - celui que j'ai rajouté -, donc je peux accéder à l'autre VLAN. C'est ce qu'on appelle le saut de VLAN. Cette faille est, à l'heure actuelle, comblée.

Le VLAN peut être configuré soit par les ports qui lui sont rattachés, soit par les adresses MAC, soit encore par une identification - login et password -. Un protocole associé à l'administration des VLAN est le protocole 802.1x. Il permet, suite à une authentification, de placer un port dans un des VLANs du switch, de façon dynamique.



# Chapitre 3

## La couche 3

**Rôle :** Cette couche est responsable de l'interconnexion de réseaux locaux ainsi que de la fragmentation des datagrammes.

Pour réaliser ces tâches il sera nécessaire de :

- définir un protocole,
- définir une adresse, appelée adresse IP (@IP). Il n'est pas possible d'utiliser l'adresse MAC sinon les couches 2 et 3 seraient liées, ce qu'il faut éviter -pour respecter le modèle OSI-.

### 3.1 Adresses IP réseau et adresses IP machine

#### 3.1.1 L'adresse IP

Il serait souhaitable de définir des adresses IP pour les machines -ordinateurs,...- mais aussi pour les réseaux. Ces dernières sont obtenues grâce au masque réseau. Une adresse IP est une suite de quatre octets, écrits en hexadécimal, et séparés par des points -*exemple : 127.24.17.255*-. Il y a donc  $2^{32}$ , soit environ quatre milliards, adresses disponibles. Aujourd'hui, pour Internet ce nombre est tout juste suffisant. Dans l'adresse IP une partie est dédiée au réseau et l'autre aux machines le composant. C'est le masque qui permet de différencier ces deux parties. Une adresse IP n'est donc valable qu'accompagnée de son masque, sinon il n'y a aucun moyen de séparer le réseau des machines dans l'adresse.

#### 3.1.2 Le masque

##### Définition

Dans le masque les bits codant le réseau sont tous positionnés à 1, ceux codant les machines à 0. Par convention, les bits codant le réseau sont toujours les bits de poids fort et les bits réseau et les bits machine ne sont pas mélangés - pour simplifier la gestion du réseau -. Ce nombre de bits est variable d'où la nécessité de connaître le masque avec l'adresse.

*Exemple : si l'adresse est 192.68.0.42 et que le réseau est codé par les trois premiers octets et les machines par le dernier. Le masque vaut alors 255.255.255.0 Cette adresse définit donc la machine 42 du réseau 192.68.0*

Une autre notation, plus synthétique est  $192.68.0.42/24$  - le 24 représente le nombre de bits à un dans le masque -. Dans le réseau de l'exemple précédent il y a donc  $2^8 = 256$  adresses machines disponibles.

De nombreux exemples de définition de réseau sont donnés en exercice corrigé en annexe.

**Exemple**

**Énoncé** Étant donné l'adresse 192.168.4.26 avec le masque 255.255.240.0, trouvez la première et la dernière adresse d'une machine sur ce réseau.

**Correction**

1. Il faut tout d'abord décomposer le masque : Les deux premiers octets étant égaux à 255 ils ne sont composés que de 1. La décomposition binaire de 240 vaut 11110000, le masque comporte donc 20 bits à 1.
2. Ces bits sont fixés pour toutes les machines de ce réseau. Donc pour toutes les machines l'adresse commencera par 192.168.? . Il faut donc déterminer à partir de l'adresse donnée la valeur des bits fixés du troisième octet. La décomposition de 4 en binaire donne 00000100 donc pour toutes les adresses cherchées les quatre premiers bits du troisième octet vaudront 0.
3. L'adressage des machines se fait donc sur douze bits - soit  $2^{12}$  adresses possibles-. La première adresse possible verra donc ces douze bits à 0 et la dernière tous ces bits à 1.

Compte tenu de la valeur des bits réseau déterminée on obtient :

- première adresse possible : 192.168.0.0
- dernière adresse possible : 192.168.15.255

**Remarque**

La définition du masque impose, par convention, d'avoir tous les 1 d'un côté et les 0 de l'autre, ce qui permet :

- d'avoir la contiguïté des bits
- d'avoir des masques avec 0,128,192,224,240,248,252,254 ou 255

**3.2 Le datagramme ou paquet**

Après avoir défini les adresses intéressons nous à la manière dont la couche 3 code les informations. La « trame » de niveau 3 est appelée datagramme ou paquet -le mot trame est réservé à la couche 2-.

**3.2.1 Le datagramme**

	Long. Tot	Protocole couche 4	@IP src	@IP dest	DATA
8 octets	2 octets	2 octets	4 octets	4 octets	

TAB. 3.1 – Datagramme IP

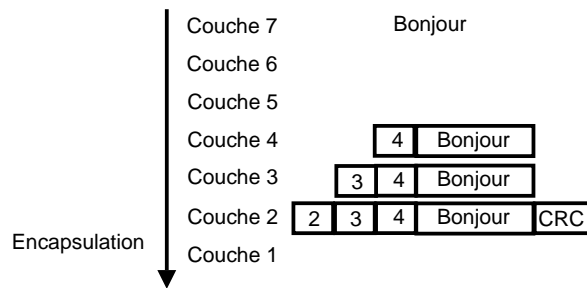


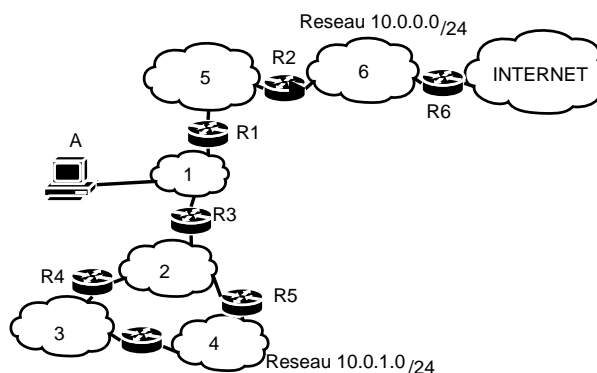
FIG. 3.1 – Encapsulation des trames et paquets.

La taille de l'en-tête est donc de 20 octets. La taille minimale du datagramme est de 0 octets de données et 20 octets d'en-tête. La taille maximale est de 65535 octets. Si il y a moins de 44 octets de données, c'est la couche 2 qui fait du bourrage pour atteindre les 64 octets minimum de couche 2. Dans l'en-tête la longueur totale du datagramme est codé sur deux octets, sa valeur maximale vaut donc  $2^{16} - 1 = 65535$  octets. Si le datagramme a une longueur supérieure à 1500 octets, qui est la limite pour les datagrammes des trames de couche 2, la couche 3 fragmente le datagramme. Le datagramme est celui d'origine - de 65535 octets de taille maximale-, le résultat de la fragmentation est appelé paquet - taille maximale 1518 octets -. C'est lui qui circule sur le réseau. Le datagramme est ensuite reconstitué sur la machine de destination.

### 3.3 Le routage

#### 3.3.1 Table de routage

Pour joindre un réseau donné, les routeurs utilisent une table de routage qui contient les chemins pour joindre un réseau donné. Il faut identifier le réseau - adresse et masque -, et indiquer comment le rejoindre grâce à une passerelle judicieusement choisie. En fait on définit une passerelle par défaut, il est donc inutile de décrire toutes les possibilités. Il ne faut décrire que les réseaux qui ne sont pas accessibles par cette passerelle.



La table de routage sera alors :

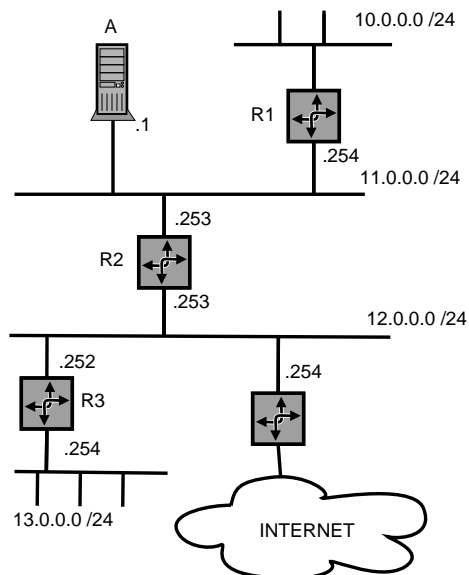
@	MASQUE	PASSERELLE
1	/24	0.0.0.0
2	/24	@ IP de R3
3	/24	@ IP de R3
4	/24	@ IP de R3
0.0.0.0	0.0.0.0	@ IP de R1

TAB. 3.2 – Table de routage

La ligne 0.0.0.0 désigne la passerelle par défaut. Aujourd'hui un gros routeur Internet a de l'ordre de mille routes dans sa table de routage. Le routage dynamique -protocoles BGP, OSPF ...- permet d'auto-configurer les routeurs. Il gère aussi les surcharges du réseau. Pour éviter les aller-retour entre deux routeurs, dans le cas où les tables de routages ne sont pas cohérentes, un TTL - Time to Live - est placé dans l'en-tête du paquet, il est codé sur un octet. Il représente la durée de vie d'un paquet. Initialement il représentait un temps mais de nos jours il représente le nombre de routeurs par lequel le paquet peut passer - il est décrémenté de 1 à chaque routeur -. Quand le TTL atteint 0 le paquet est jeté et on envoie un message à la machine émettrice pour lui indiquer que le paquet a été jeté. Il y a un petit risque que le message d'erreur fasse également du ping-pong entre les deux routeurs, cependant il ne peut pas générer de message d'erreur. Un tel phénomène est appelé boucle de routage.

### 3.3.2 Exercice

#### Énoncé



- Donner la table de routage de la machine A,
- Donner la table de routage du routeur R2.

**Corrigé**

@	MASK	PASSERELLE
10.0.0.0	/24	11.0.0.254
11.0.0.0	/24	0.0.0.0
0.0.0.0	0.0.0.0	11.0.0.253

TAB. 3.3 – Table de routage de la machine A

@	MASK	PASSERELLE
10.0.0.0	/24	11.0.0.254
13.0.0.0	/24	12.0.0.252
11.0.0.0	/24	0.0.0.0
12.0.0.0	/24	0.0.0.0
0.0.0.0	0.0.0.0	12.0.0.254

TAB. 3.4 – Table de routage du routeur R2

**3.3.3 Évolution des tables de routage**

Dans les systèmes modernes -XP et Linux-, il y a deux informations supplémentaires dans les tables de routage.

@	MASK	PASSERELLE	Interface	Métrique

TAB. 3.5 – Table de routage moderne

**Interface** : si il y a plusieurs interfaces sur la machine - plusieurs cartes réseau-, ce champs indique laquelle utiliser.

**sec :Métrique** : c'est l'indice de priorité, codé par un chiffre de 1 à 20. Plus le chiffre est petit, plus la route est prioritaire. Ces priorités sont définies soit à la main par l'administrateur, soit par des protocoles de gestion dynamique type BGP.

**3.4 Adresses particulières****3.4.1 Adresses réservée**

Des adresses ne peuvent être affectées à des machines :

- l'adresse où tous les bits machine sont à 0 : c'est l'adresse du réseau, elle est réservée.
- l'adresse où tous les bits machine sont à 1 : adresse de broadcast du réseau.
- 127.0.0.1/8 : c'est l'interface de communication locale. Elle est utilisée lorsque la carte réseau sert à faire communiquer deux systèmes internes à une machine. Les paquets transitent par l'interface réseau mais aucun d'eux n'est émis sur le réseau. C'est ce qui s'appelle l'*interface de loopback*.

**3.4.2 Exemple**

Pour le réseau 10.0.0.0/16 la première adresse, 10.0.0.0, et la dernière adresse 10.0.255.255, ne sont pas utilisées pour les machines. Elles représentent respectivement l'adresse du réseau et l'adresse de broadcast du réseau.

## 3.5 Gestion des adresses IP

### 3.5.1 Problématique

Le nombre d'adresses IP disponible n'est pas infini et se limite à  $2^{32}$ , soit environ quatre milliards d'adresses. Si je veux définir un réseau de 130 machines, compte tenu du masque il me faut coder les machines sur huit bits  $2^7 \leq 130 \leq 2^8$ . Je bloque donc  $2^8 = 256$  adresses IP. Il faut gâcher un minimum d'adresses d'autant que, aujourd'hui, les limites de cette plage d'adresse ont été atteintes avec le développement d'Internet. Diverses solutions ont été envisagées pour résoudre ce problème.

### 3.5.2 Passage à IPv6

La solution simple, d'un point de vue théorique, consiste à augmenter le nombre de bits disponibles pour coder les adresses IP. C'est la logique qui prévaut pour le nouveau protocole IP -IP version 6-, qui code une adresse sur 16 octets, au lieu de 4. Il pose néanmoins des problèmes de mises en place, au premier rang desquels son coût d'implémentation. En effet modifier la taille de l'adresse suppose le changement de tous les matériels ce qui représente un investissement prohibitif, d'autant que d'autres solutions ont été trouvées, qui repoussent la limite du nombre d'adresse IP disponible et rendent ainsi le changement moins critique. Le passage à IPv6 n'est donc pas à l'ordre du jour. Certains pays, notamment en Chine, devant mettre en place une structure pour Internet ont fait le choix de s'équiper directement en matériel IPv6.

### 3.5.3 L'adressage CIDR

L'adressage CIDR -Classless Internet Domain Routing- permet d'appliquer n'importe quel masque à n'importe quelle adresse. Ce changement remonte à une dizaine d'années. À l'époque, il y avait des classes d'adresses A, B ou C.

- Class A : rassemblait les gros réseaux dont le masque était 255.0.0.0 et dont les adresses allaient de 0.0.0.0 à 126.0.0.0. Ils pouvaient comporter  $2^{24} - 1$  soit environ 16 millions d'adresses chacun.
- Class B : rassemblait les réseaux dont le masque était 255.255.0.0 et dont les adresses allaient de 128.0.0.0 à 191.255.0.0. Ils pouvaient comporter chacun  $2^{16} - 1$ , soit 65535 adresses.
- Class C : rassemblait les réseaux dont le masque était 255.255.255.0 et dont les adresses commençaient à 192.0.0.0 .

Ces classes, avec leurs masques fixés, faisaient perdre beaucoup d'adresses inutilement. Elles ont été abandonnées à la fin des années 1990, et ne sont citées que pour la culture personnelle du lecteur, certains ouvrages y faisant encore référence bien qu'elles soient obsolètes. À l'heure actuelle le masque peut comporter n'importe quel nombre de bits à 1.

### 3.5.4 Regroupement géographique d'adresses

Les plages d'adresses qui se suivent ont été regroupées géographiquement. Ce processus est toujours en cours. Par exemple les plages d'adresses 11.0.0.0/8, 12.0.0.0/8, 13.0.0.0/8, et 14.0.0.0/8 ont été regroupées aux USA. Du point de vue des adresses c'est donc comme si on avait un seul réseau en 8.0.0.0/5. Il y a donc moins de perte d'adresses. Cependant ces solutions ne font que repousser le problème de la famine en adresses IP. Le procédé qui permet de s'affranchir de la limite des quatre milliards d'adresses est la NAT.

### 3.5.5 NAT - Network Address Translation-

La NAT est un mécanisme qui permet de cacher un réseau complet derrière une seule adresse IP.

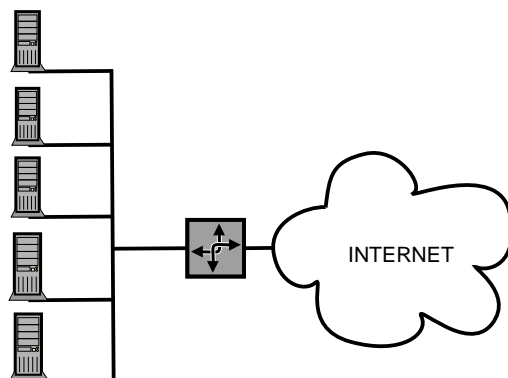


FIG. 3.2 – Principe de la NAT

Toutes les adresses des machines du réseau prennent pour adresse d'expédition l'adresse du routeur. Toutes les adresses du réseau sont donc masquées derrière celle du routeur. Le problème posé est que les adresses que je donne à mes machines dans le réseau ne doivent pas être utilisées ailleurs sur Internet sinon des problèmes de routage vont apparaître. La recommandation **RFC1918** définit des plages d'adresses disponibles pour des réseaux locaux :

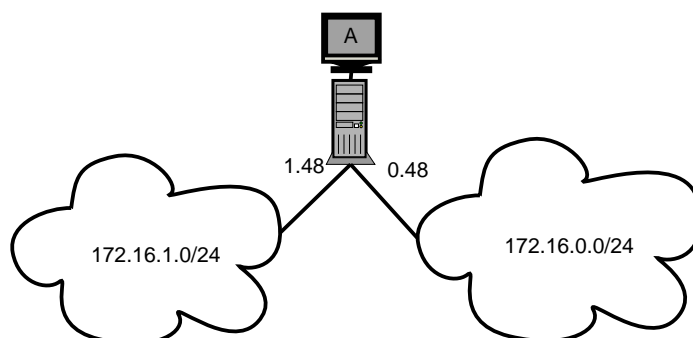
- 10.0.0.0/8
- 192.168.0.0/16
- 172.16.0.0/12

Ces adresses réseaux sont réservées à un usage privé. Il est donc nécessaire d'utiliser la NAT pour que les paquets reviennent au routeur qui, lui, connaît le réseau local. Le routage par le routeur vers le réseau local est explicité dans le chapitre concernant la NAT <sup>1</sup>.

### 3.5.6 Exercice pratique

#### Énoncé

Le but est de configurer un réseau 172.16.0.0/24 et un réseau 172.16.1.0/24. Puis configurer une machine pour qu'elle joue le rôle de bascule entre les deux réseaux.



#### Quelques commandes Linux utiles

**Changer une adresse IP sur une machine :** `ifconfig Interface @IP mask`

```
ifconfig eth0 172.16.0.11 255.255.255.0
```

**Afficher la table de routage :** `netstat -nr` ou encore `route -n`

<sup>1</sup>voir 8 page 57

**Ajouter une entrée à la table de routage pour atteindre un réseau :** `route add -net @reseau netmask mask gw @passerelle - gw : get away  $\approx$  passerelle - route add -net 172.16.1.0 netmask 255.255.255.0 gw 172.16.0.48`

**Forcer une route entre un réseau et une machine :** `route add machine @machine_dest gw @machine_src`, permet de communiquer entre deux machines du même réseau - ajout dans le fichier `host` -  
`route add -host 172.16.1.X gw 172.16.0.45`

**Ajouter l'entrée default de la table :** `route add default gw @IPpasserelle`

## Réalisation

Il faut tout d'abord configurer chaque machine du réseau en :

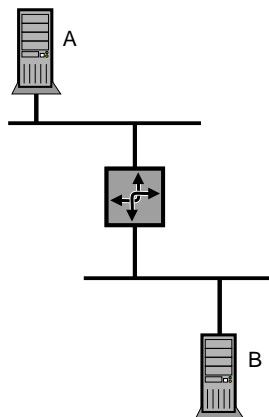
- lui donnant son adresse IP pour le réseau
- remplissant sa table de routage à la main en indiquant une entrée pour le réseau local et une entrée -default- pour l'autre réseau.

Pour pouvoir déclarer deux interfaces, cas de la machine assurant la bascule, il faut déclarer une interface virtuelle `eth0 :0` grâce à la commande `ifconfig eth0 :0 @IP netmask mask` Cependant la machine A n'est pas configurée en routeur, donc, à ce stade, elle jette les paquets qui ne lui sont pas destinés mais sont pour l'autre réseau. Pour transformer une machine en routeur il suffit d'écrire un 1 dans le fichier `/proc/sys/net/ipv4/ip_forward`. Si ce fichier contient un 1 le routage est activé, si il contient un 0 il ne l'est pas.

```
echo 1>/proc/sys/net/ipv4/ip_forward
```

## 3.6 Exercice

### 3.6.1 Énoncé



Écrire la trame d'un ping de la machine B par la machine A.

### 3.6.2 Correction

frames

@MAC R1	@MAC A	Protoc 3	Long	Protoc 4	@IP A	@IP B	DATA	CRC
---------	--------	----------	------	----------	-------	-------	------	-----

TAB. 3.6 – Trame quittant A



Passant R1 seules les adresses MAC changent, ce qui est normal vu que ces adresses sont propres aux réseaux locaux.

### 3.6.3 Obtention des adresses MAC

#### Protocole ARP

Pour obtenir l'adresse MAC de R1, connaissant son adresse IP grâce à la table de routage, il est fait appel au protocole ARP -Address Resolution Protocol-. Dans chaque machine il y a une table ARP mettant en correspondance des adresses IP et des adresses MAC. Chaque entrée de cette table a une durée de validité limitée - un Time to Live -, pour pallier aux changements de matériels, d'adresse IP, etc. Celui-ci est de l'ordre de deux minutes. Si l'information n'est pas disponible dans la table ARP, la machine lance un broadcast demandant l'adresse MAC correspondant à l'adresse IP donnée - soit celle de R1 -. C'est la requête ARP.

#### Commandes sous Linux

- Pour voir la table ARP : `arp - a`
- Pour ajouter une adresse à la table : `arp -s`
- Pour supprimer une adresse : `arp -d`

#### ARP poisoning

Si il y a une machine sur le réseau, P, qui répond à la requête ARP de A, après R1, elle modifie la table ARP de A, liant l'adresse IP de B avec sa propre adresse MAC. Pour ce faire elle envoie la trame suivante :

@MAC A	@MAC P	Protoc 3	Long	Protoc 4	@IP B	@IP A	CRC
--------	--------	----------	------	----------	-------	-------	-----

Tous les paquets de A passeront donc par P avant d'être redirigé, grâce à la table de routage, et à la table ARP, de P vers le routeur R1. P voit donc passer tout le trafic. C'est ce qui s'appelle le *ARP cache poisoning*. Ce type d'attaque est difficilement parable. En fait il n'y a même pas nécessairement besoin d'un broadcast car chaque machine, même si elle n'a rien demandé, recevant une réponse à un broadcast ARP met sa table ARP à jour. La seule manière de se prémunir contre une telle menace est d'inscrire « en dur » la table arp et d'interdire toute modification dynamique. La gestion du réseau s'en trouve passablement alourdie.

#### Conclusion

Le protocole ARP est à mi chemin entre la couche 2 -adresse MAC- et la couche 3 -adresse IP-. Une attaque arp est une attaque locale sur le réseau. Elle ne marche pas dès que l'on sort de son propre réseau, donc pas sur Internet.

## 3.7 ICMP

Si il y un problème sur le réseau il n'y a pas moyen de le déboguer. Il a donc été créé l'ICMP -Internet Control Message Protocol-. Ce protocole dispose d'un certains nombre de messages qui permettent de résoudre les problèmes liés notamment à la couche 3 et au protocole IP.

### 3.7.1 Messages ICMP

Il y a différents messages.

**echo request**

**echo reply**

**redirect** : une machine routeur indique qu'il y a un chemin plus court pour joindre la cible.

**host unreachable**

**network unreachable**

**port unreachable**

**TTL exceeded**

Redirect modifie la table de routage de la machine qui le reçoit. Il y a donc moyen d'envoyer des redirect pour détourner le trafic normal. C'est pourquoi, à l'heure actuelle, les machines n'acceptent pas les redirect. Tous ces paquets ne sont pas transmis par la machine à l'utilisateur, il faut donc « sniffer » le réseau pour les voir.

### 3.7.2 Outils utilisant les messages ICMP

#### Ping

Cette instruction correspond à deux messages ICMP : un echo request et un echo reply. Elle permet de voir si une machine est visible sur le réseau.

#### Traceroute

Cette instruction permet de visualiser les routeurs par lesquels on passe pour aller sur une certaine machine.

**Exemple** : Traceroute 82.230.89.213 Si la réponse est 1/R1 2/R2 3/R3 4/R2 5/R3 6/R2 cela indique clairement l'existence d'une boucle de routage. Donc R2 ou R3 est mal configuré -ou les deux-.

**Fonctionnement de cette instruction** Elle utilise un des messages ICMP et envoie une suite de messages avec des TTL croissants - en commençant par un -. Quand la machine ne reçoit plus de la part du réseau un message « TTL exceeded » c'est que la cible a été atteinte. Cependant le routage étant dynamique, rien ne garantit que le message numéro quatre suivra le même chemin que le numéro cinq. En pratique, il apparaît que le chemin utilisé varie peu. N'importe lequel des messages ICMP peut permettre d'atteindre le but fixé. Sous linux c'est un message « udp », qui renvoie « port unreachable » à destination. Windows, en revanche utilise un message « echo request » qui se traduit par un message « echo reply » à destination.

# Chapitre 4

## La couche 4

**Rôle :** cette couche est chargée de gérer les connexions applicatives. Comme pour les autres couches il va nous falloir définir des protocoles et des adresses.

### 4.1 Les protocoles de couche 4

Il existe principalement deux protocoles :

- TCP en mode connecté,
- UDP en mode non connecté.

Envoyer en mode non connecté c'est envoyer un paquet sans garantie qu'il arrive à destination - à rapprocher du courrier standard-. En revanche si l'envoi a lieu en mode connecté, la machine de destination acquitte la réception des paquets - à rapprocher du téléphone ou du recommandé avec accusé de réception-. En protocole UDP, c'est l'application qui gère la réexpédition. En télévision et en radio on utilise UDP, si la perte d'un paquet n'est pas préjudiciable, en revanche, le renvoi d'un paquet n'a pas d'intérêt vu que l'on affiche ou émet en temps réel. Il y a encore quelques années UDP ne servait qu'au protocole de résolution de nom DNS<sup>1</sup>, mais l'avènement de la radio et de la télévision par Internet a développé son utilisation. Le passage d'un protocole à l'autre est impossible. C'est le concepteur du logiciel qui choisit le protocole utilisé. Le TCP est parfois privilégié car les firewalls bloquent en général l'UDP, à l'exception des requêtes DNS.

### 4.2 Les adresses de couche 4

Les ports jouent le rôle d'adresse de couche 4, et sont donc les adresses des applications. Il sont écrits en décimal, sur deux octets. Il y a donc  $2^{16} - 1$  soit 65535 ports disponibles. Certains ports sont associés à des applications :

**port 80 :** utilisé pour le web (http)

**port 25 :** utilisé pour le mail en envoi de courrier (smtp)

**port 110 :** utilisé pour le mail en réception de courrier (pop)

**ports 20 et 21 :** utilisés pour ftp

...

Les ports numérotés de 0 à 1023 sont réservés, sous UNIX, à root - pas sous Windows - contrairement ceux de 1024 à 65535 qui sont appelés les ports élevés. Les ports réservés sont les adresses destination des applications recherchées. Historiquement, leur réservation permettait de garantir à l'utilisateur que le serveur avait été lancé par un administrateur et non par un utilisateur quelconque. Les ports élevés

---

<sup>1</sup>ce protocole est développé plus tard , chapitre 5, page 39

sont utilisées comme adresses clients - source-. Le port destination est vers une application serveur, le port source est choisi par le client. Si une machine ouvre dix pages web c'est à chaque fois le même port destination, mais il y a un port élevé source ouvert pour chaque page.

## 4.3 Le protocole UDP

### 4.3.1 Retour sur la notion de datagramme

Un datagramme représente un ensemble de données dont il n'y a aucune garantie qu'il soit correctement acheminé. C'est pour cela que le terme datagramme UDP est utilisé.

### 4.3.2 Datagramme UDP

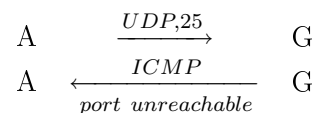
Port SRC 2 octets	Port DST 2 octets	Long Tot 2 octets	CRC 2 octets	DATA
----------------------	----------------------	----------------------	-----------------	------

TAB. 4.1 – Datagramme UDP

Le CRC porte sur tout le datagramme car il peut y avoir des erreurs générées lors du passage d'une couche à l'autre. Cette en-tête est la plus petite de toutes les en-têtes.

### 4.3.3 Établissement et rupture de la communication

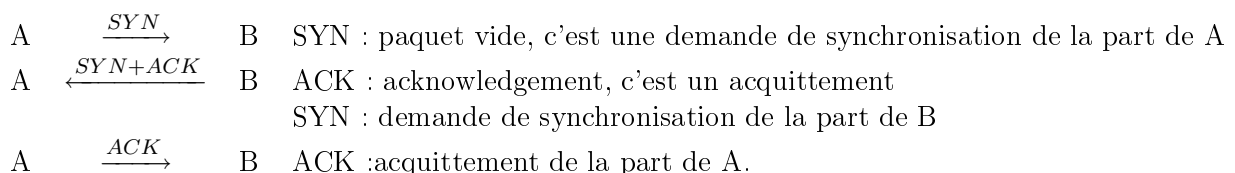
Le serveur étant en attente sur un port bas, le système d'exploitation ouvre un port - source - pour l'application et garde en mémoire l'application qui y est rattaché. Il peut ainsi renvoyer les informations en provenance du serveur. Si A cherche à joindre GOOGLE, avec en port source X, en UDP, sur le port 25, alors que google utilise le port 80



## 4.4 Protocole TCP

### 4.4.1 Établissement et rupture de la communication

L'établissement de la connexion repose sur le three ways handshake - la triple poignée de main -. Pour qu'une machine A établisse une connexion avec une machine B il faut :



Tous ces paquets sont vides et ne contiennent que l'en-tête. La fin de la communication se déroule de la même manière. Si A n'a plus de données à transmettre elle envoie un signal de fin de connexion :

Note : Si, à la réception du signal de *FIN* envoyé par A, B a encore des données à envoyer elle continue à transmettre et n'enverra son signal *FIN* qu'une fois que toutes les données seront transmises.

Le dernier *FIN* n'étant pas acquitté la machine qui l'envoie reste en écoute pendant un temps défini par le système d'exploitation au cas où il serait nécessaire de le ré-émettre.

Si A cherche à joindre GOOGLE, avec en port source X, en TCP, sur le port 25, alors que google utilise le port 80

$$\begin{array}{l} A \xrightarrow{SYN,25} G \\ A \xleftarrow{RST,X} G \end{array}$$

#### 4.4.2 Segment TCP

Port SRC 2 octets	Port DST 2 octets	FLAGS 6 bits	URG POINTER 2 octets		DATA
----------------------	----------------------	-----------------	-------------------------	--	------

TAB. 4.2 – Segment TCP

Il y a différents flags possibles, tous codés sur un bit :

**SYN** : flag de synchronisation,

**ACK** : flag d'acquittement,

**PSH** : PUSH demande à ce que les paquets envoyés soient traités en priorité,

**URG** : URGENT comparable à PUSH mais ne porte que sur certaines informations du paquet.

**RST** : RESET « Je ne veux plus parler avec toi », ou alors « il y a un problème dans la communication recommençons la synchronisation ».

Le URG POINTER - pointeur urgent- est utilisé avec le flag URG pour indiquer la position des informations à traiter en priorité dans le paquet. Il pointe sur la fin des données urgentes. Selon les systèmes d'exploitation il désigne soit le dernier octet urgent, soit le premier non urgent.

L'en-tête TCP comporte 20 octets, plus éventuellement des options. Nous en avons identifié un peu plus de six jusqu'à présent. Les autres informations contenues dans cette en-tête seront détaillées au chapitre 7.<sup>2</sup>

#### 4.4.3 Remarque

Le protocole TCP est destiné à faire communiquer une machine avec une autre machine. Donc la notion de broadcast n'a pas grand sens - elle suppose la communication entre au moins trois machines-. Donc le broadcast n'est pas supporté par TCP. Si nécessaire, il faut avoir recours à UDP.<sup>3</sup>

<sup>2</sup>page 53

<sup>3</sup>voir chapitresA page 73



## Chapitre 5

# Le protocole DNS

**Rôle :** Le but du DNS est de transformer un nom en adresse IP. C'est une information de niveau mondial.

### 5.1 Qu'est ce que le DNS ?

Le DNS, ou Domain Name System, représente un protocole ainsi que l'ensemble des machines et logiciels qui participent à la résolution des noms. C'est avant tout une base de données partagée, répandue partout dans le monde. L'information est distribuée entre différents serveurs de part le monde. Le protocole DNS associe un FQDN - Fully Qualified Domain Name -, qui comprend un nom de machine et un nom de domaine, et une adresse IP.

Exemple de FQDN :  $\underbrace{www}_{machine}.\underbrace{google.com}_{domaine}$

### 5.2 Historique

À la création d'ARPANET il y avait une centaine de machines, soit une centaine d'adresses IP. Le nombre de machines augmentant, l'idée d'associer un nom à chaque adresse IP est apparue dans le but de simplifier les adressages. C'est la création des fichiers Host qui font l'association entre le FQDN et les adresses IP. Les utilisateurs faisaient ces associations « à la main » afin de créer ces fichiers. Avec l'expansion d'Internet cette façon de faire n'est plus possible. C'est pourquoi le système DNS a été créé. Il représente un système hiérarchisé de noms de machine. Le système nommé Domain Name System, a été mis au point en novembre 1983 -RFC 882 et RFC 883-, puis révisé en 1987 dans les RFCs 1034 et 1035. Le DNS a fait l'objet depuis de nombreuses RFCs.

### 5.3 Organisation du système DNS

Le système DNS est organisé en arborescence. Une fonctionnalité dynamique étant recherchée, des serveurs ont été dédiés à cette tâche.

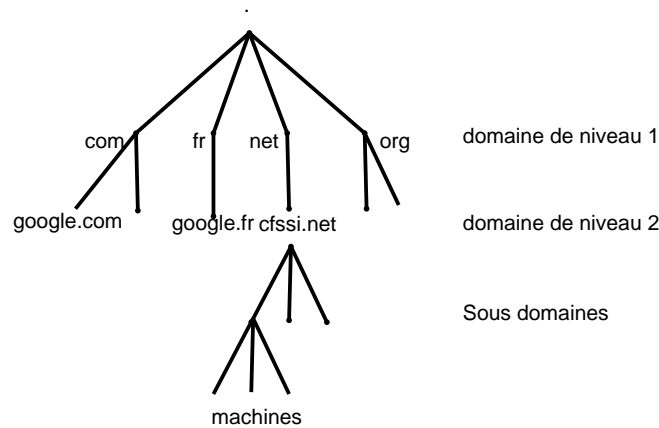


FIG. 5.1 – Arborescence DNS

Cette arborescence est de type unique et se lit de droite à gauche : l'origine de l'arbre est à droite puis sont indiqués les domaines de niveau un puis deux, le sous domaines et enfin à l'extrême gauche la machine. Il est courant d'oublier le point final, l'adresse indiquée -sans le point- est donc une adresse relative. Cependant, le mécanisme rajoute systématiquement le point à la fin de l'expression rétablissant l'adressage absolu. Chaque noeud de l'arborescence porte une étiquette d'une longueur maximale de 63 caractères, appelée **nom de domaine**. L'extrémité d'une branche de l'arbre est appelée **hôte**, et représente une machine ou une entité du réseau. Le FQDN est le nom complet de l'hôte comportant tous les domaines et sous domaines séparés par des points, par exemple « www.besssi.cfssi.net. ». La profondeur maximale de l'arborescence est de 127 niveaux et la longueur maximale d'un FQDN est de 255 caractères. Les domaines sont gérés par des entités différentes, d'où la distribution de la base de données de part le monde. Chaque entité dispose de deux serveurs DNS - un serveur primaire et un serveur secondaire - pour assurer la continuité du service. Sinon, en cas de crash du serveur il n'y aurait aucun moyen d'accéder au domaine sinon en utilisant directement les adresses IP. Une entité unique doit gérer les serveurs racine - le « . » de l'arborescence -. Il y en a seize au total -treize aux USA, un au Japon, un au Royaume Uni et un aux Pays bas-. Ils portent les noms de « a.root-servers.net » à « p.root-servers.net ». Les trois derniers ont été mis en place pour casser le monopole américain qui avait la main mise sur Internet. En faisant tomber les serveurs racine la majorité du réseau internet tombe, seuls quelques initiés pourraient continuer à l'utiliser en se servant des adresses IP directement - à condition de les connaître-. Il n'en reste pas moins que l'influence américaine est toujours prépondérante compte tenu du nombre de serveurs DNS hébergés. Les différents serveurs racine sont tous des miroirs les uns des autres. Certains serveurs sont exploités par des entreprises commerciales. Ces entreprises peuvent donc re-router les paquets à leur guise, vers des sites commerciaux de leur choix par exemple - ce type de dérive a été observé à de rares occasions-. Il existe deux catégories de TLD -Top Level Domain, soit domaines de plus haut niveau- :

- Les domaines dits « génériques », appelés gTLD -generic TLD-. Les gTLD sont des noms de domaines génériques de niveau supérieur proposant une classification selon le secteur d'activité. Ainsi chaque gTLD possède ses propres règles d'accès :
  - gTLD historiques :
    - **.arpa** correspond aux machines issues du réseau originel ;
    - **.com** correspondait initialement aux entreprises à vocation commerciale. Désormais ce TLD est devenu le « TLD par défaut » et l'acquisition de domaines possédant cette extension est possible, y compris par des particuliers.
    - **.edu** correspond aux organismes éducatifs ;
    - **.gov** correspond aux organismes gouvernementaux ;



- .int** correspond aux organisations internationales ;
- .mil** correspond aux organismes militaires ;
- .net** correspondait initialement aux organismes ayant trait aux réseaux. Ce TLD est devenu depuis quelques années un TLD courant. L'acquisition de domaines possédant cette extension est possible, y compris par des particuliers.
- .org** correspond habituellement aux entreprises à but non lucratif.
- nouveaux gTLD introduits en novembre 2000 par l'ICANN :
  - .aero** correspond à l'industrie aéronautique ;
  - .biz** -business- correspondant aux entreprises commerciales ;
  - .museum** correspond aux musées ;
  - .name** correspond aux noms de personnes ou aux noms de personnages imaginaires ;
  - .info** correspond aux organisations ayant trait à l'information ;
  - .coop** correspondant aux coopératives ;
  - .pro** correspondant aux professions libérales.
- gTLD spéciaux :
  - .arpa** correspond aux infrastructures de gestion du réseau. Le gTLD arpa sert ainsi à la résolution inverse des machines du réseau, permettant de trouver le nom correspondant à une adresse IP.
- Les domaines dits « nationaux », appelés ccTLD -country code TLD-. Les ccTLD correspondent aux différents pays et leurs noms correspondent aux abréviations des noms de pays définies par la norme ISO 3166.

## 5.4 Attaque contre le système DNS

Il est donc établi que, à l'heure actuelle, faire tomber le système DNS revient à faire tomber Internet. Une attaque visant à démontrer la faisabilité d'une telle attaque a déjà été réalisée. Cette attaque a réussi à faire tomber dix des treize serveurs américains pendant une courte durée. L'impact de l'attaque a été faible car le serveur DNS, en même temps que la résolution de nom, donne une durée de validité à cette résolution - en général de l'ordre d'une journée -. Donc pour réaliser une attaque efficace il faut provoquer un déni de service pendant 24 heures sur les serveurs.

## 5.5 Résolution d'un nom

### 5.5.1 Requête DNS

`www.google.com`

Le « `www` » représente donc la machine - par convention les machines serveur de web sont appelées `www` - et `www.google.com` est le FQDN. Je peux appeler une machine `www.google.com.lalitte.com`. si je dispose du domaine « `lalitte.com`. » . C'est le point final qui fait la différence. Il n'y a aucun moyen de savoir où finit le nom de domaine et où commence le nom de machine. La résolution est faite par un logiciel appelé « `resolver` ». La question posée n'est en général pas complète - pas de point final -. Ma machine appartient à un domaine - disons `iti.esiea.fr` - et son `resolver` dispose d'un fichier de configuration : `resolv.conf`. Ce fichier contient le nom de domaine auquel ma machine est rattachée ainsi que les adresses IP des serveurs DNS auxquels je dois m'adresser. Le `resolver` fait d'abord la requête dans son domaine « `www.google.com.iti.esiea.fr`. » et interroge un serveur DNS référencé dans le fichier de configuration. En cas d'échec le `resolver` recommence la requête en enlevant un domaine à chaque fois - au moins un point -. Il interroge donc dans l'ordre :

- www.google.com.esiea.fr
- www.google.com.fr
- www.google.com

Cette dernière tentative aboutit enfin.

A la configuration des serveurs DNS cache il suffit d'indiquer l'adresse IP du serveur DNS racine. Les autres adresses IP seront apprises au gré des requêtes.

### 5.5.2 Exemple

« Viaris.cfssi.local »

Domaine d'appartenance de la machine : cfssi.local

Si on demande viaris il trouve, en revanche si on demande viaris. il cherche à la racine DNS, donc ne trouve pas.

### 5.5.3 Exemple d'attaque

Si je gère un serveur DNS du domaine je peux créer un site bancaire fantôme - www.bmp.fr.cfssi.local -. Les machines du réseau local cherchant à accéder au site www.bmp.fr vont accéder au site fantôme.

## 5.6 Types de requêtes DNS

### 5.6.1 Requête récursive ou requête itérative

Quand un resolver fait une requête - www.inria.fr.- il y a deux types de requête : Soit le resolver s'adresse à un serveur de cache qui interroge les différents serveurs sur internet. C'est une requête récursive : je pose la question et j'attends la réponse. Si le serveur cache ne connaît pas la réponse il interroge le serveur racine. Celui-ci répond qu'il ne connaît pas la résolution - la machine www.inria.fr n'est pas référencée -, mais indique au cache « interroge fr dont l'adresse du serveur DNS est @IP-DNS. ». Le serveur cache interroge alors fr. qui lui répond « inconnu mais interroge inria.fr dont l'adresse du serveur DNS est @IP-DNS. ». Le serveur de cache fait alors la requête et le serveur de domaine inria.fr répond « je connais et l'adresse IP est 192.83.2.3 ». Le serveur cache renvoie alors l'information à la machine. Donc le resolver de ma machine procède à une requête récursive mais le serveur de cache émet des requêtes itératives. Le serveur cache - appelé aussi serveur DNS local - possède son propre resolver qui se charge des requêtes itératives.

### 5.6.2 Exemple

#### Énoncé

Déterminer les requêtes DNS pour les commandes suivantes, exécutées l'une à la suite de l'autre :

1. ping www.google.com.
2. ping www.yahoo.com.

#### Solution

**www.google.com** Le serveur DNS local est interrogé Le resolver DNS local interroge le serveur . : réponse : inconnu mais interroge le serveur DNS de com. dont l'adresse IP est @IP. Le resolver DNS local interroge le serveur com. : réponse : inconnu mais interroge le serveur DNS de google.com. dont l'adresse IP est @IP. Le resolver DNS local interroge le serveur google.com. qui répond : connu l'adresse IP est @IPgoogle. Le resolver DNS local transmet l'adresse IP au resolver de la machine

**www.yahoo.com** Le resolver DNS local se souvient de la requête précédente et interroge directement le serveur DNS de .com. dont il connaît l'adresse IP. Celui-ci répond : inconnu mais interroge le serveur DNS de yahoo.com. dont l'adresse IP est @IP. Le resolver DNS local interroge le serveur yahoo.com. : réponse : connu l'adresse IP est @IPyahoo. Le resolver DNS local transmet l'adresse IP au resolver de la machine

C'est le serveur DNS qui fournit l'information, qui fixe sa durée de validité.

## 5.7 Types d'enregistrements

Un DNS est donc une base de données répartie contenant des enregistrements, appelés RR -Resource Records-, associants un FQDN à une adresse IP. En raison du système de cache permettant au système DNS d'être réparti, les enregistrements de chaque domaine possèdent une durée de vie -Time To Live-, qui permet aux serveurs intermédiaires savoir s'il est nécessaire de les revérifier. Un enregistrement DNS comporte généralement les informations suivantes :

Nom de domaine -FQDN-	TTL	Type	Classe	RData
www.cfssi.net.	3600	A	IN	185.115.255.215

TAB. 5.1 – Exemple d'enregistrement DNS

Il y a différents types d'enregistrement en fonction de la machine visée ou de la fonction recherchée.

**SOA** - Start of authority - définit le serveur : nom du serveur, numéro de série du fichier de configuration, time to live.

**NS** - nom de serveur - qui indique le serveur DNS de mon domaine. Il associe donc un nom de serveur DNS à une adresse IP.

**A** - address - c'est l'enregistrement de base. Il assure la correspondance entre le FQDN et l'adresse IP : j'envoie le nom de domaine et je récupère l'adresse IP.

**MX** - Mail exchanger - correspond au serveur de gestion du courrier. Lorsqu'un utilisateur envoie un courrier électronique à une adresse « utilisateur@domaine », le serveur de courrier sortant interroge le serveur de nom ayant autorité sur le domaine afin d'obtenir l'enregistrement MX. Il peut exister plusieurs MX par domaine, afin de fournir une redondance en cas de panne du serveur de messagerie principal. Ainsi l'enregistrement MX permet de définir une priorité avec une valeur pouvant aller de 0 à 65 535. Cet enregistrement associe donc un nom de domaine avec soit le FQDN, soit l'adresse IP du serveur de messagerie du domaine. Si je rentre en plus le nom, j'ai directement l'enregistrement A en même temps.

**PTR** Il associe l'adresse IP et le FQDN : j'envoie l'adresse IP et je récupère le nom de domaine.

**HINFO** champ uniquement descriptif permettant de décrire notamment le matériel (CPU) et le système d'exploitation (OS) d'un hôte. Il est généralement conseillé de ne pas le renseigner afin de ne pas fournir d'éléments d'informations pouvant se révéler utiles pour des pirates informatiques.

**CNAME** - canonical name - Il associe deux FQDN, c'est un alias. Il est particulièrement intéressant pour fournir des alias aux différents services d'une même machine.

Les autres champs de l'enregistrement sont :

**Classe** : la classe peut être :

- **IN** pour les protocoles d'internet, il s'agit donc du système utilisé
- **CH** -pour le système chaotique-;

**RDATA** : il s'agit des données correspondant à l'enregistrement. Les informations attendues selon le type d'enregistrement sont :

- A : une adresse IP sur 32 bits ;
- CNAME : un nom de domaine ;
- MX : une valeur de priorité sur 16 bits, suivi d'un nom d'hôte ;
- NS : un nom d'hôte ;
- PTR : un nom de domaine ;
- SOA : plusieurs champs.

## 5.8 Types de serveurs

Sur un même serveur DNS je peux héberger plusieurs domaines - soit plusieurs zones -. Il faut alors un fichier de configuration de zone par domaine.

### 5.8.1 Deux types de serveurs DNS

Il y a les serveurs de domaine et les serveurs relais - ou de cache -.

- serveur de domaine : il répond aux requêtes pour un domaine particulier - exemple cfssi.local -
  - serveur de relais ou de cache : répond à toutes les requêtes - de tous les domaines - et peut répondre à une requête sur un domaine particulier mais en la relayant à d'autres serveurs DNS.
- Il est aussi possible de définir quelles adresses sont autorisées à interroger un serveur DNS.

### 5.8.2 Questions

*Le serveur fr. doit il accepter les requêtes récursives ?* Non, c'est un serveur de domaine, il serait rapidement surchargé. C'est pourquoi il se limite aux requêtes itératives.

*Un serveur de domaine doit il accepter les requêtes récursives ?* Non, voir la réponse précédente.

*Qui fait les requêtes itératives ?* Les serveurs DNS de relais ou de cache. Ils acceptent les requêtes récursives.

## 5.9 Les outils DNS

### 5.9.1 Au niveau de la machine

**nslookup** Il peut s'utiliser de deux manières :

- nslookup FQDN
- nslookup puis au prompt
  - >server @IP. Il sert à modifier le serveur DNS associé à la machine.
  - >set query = MX
  - >FQDN

**Host, dig** plus récents

**online nslookup** - controlops.net - qui permet de faire des requêtes en ligne, et permet de voir sa zone DNS de l'extérieur.

**Protocoles utilisés** : DNS utilise à la fois l'UDP et le TCP. Le TCP est utilisé pour les grosses requêtes - taille supérieure à 512 octets -. Donc il utilise aussi TCP53.

### 5.9.2 Les outils serveur

**Bind** C'est l'outil historique, la joie des pirates. En effet, tout le monde doit avoir du Bind pour profiter de DNS, donc dès qu'une faille est trouvée tout le monde est susceptible d'être attaqué. Bind n'a pas été pensé, au départ, avec un objectif de sécurité. Il est en train d'être entièrement refondu pour garantir une certaine sécurité.

**DjbDNS** C'est un outils créé par Bernstein, très sécurisé - Bernstein offre US\$500 au premier qui trouvera une faille -. DjbDNS dispose de sa propre licence qui interdit toute modification sans accord du propriétaire. Bernstein a également développé qmail qui est le serveur de messagerie le plus utilisé.



## Chapitre 6

# Retour sur le protocole IP

### 6.1 L'en-tête IP

#### 6.1.1 Description de l'en-tête

4 bits	4 bits	1 octet	3 bits	5 bits	1 octets
version	IHL	TOS	LONG TOT		
IP ID			FLAGS	Frag offset	
TTL	Protocole		CRC en-tête		
adresse IP source					
adresse IP destination					
Options				Padding	

TAB. 6.1 – En-tête IP

**version** IPv4

**IHL** IP header length : longueur de l'en-tête.

**TOS** Type of service : définit la priorité des informations.

**LONG TOT** Longueur totale du datagramme.- sur deux octets donc 65535 au maximum -.

**ID** c'est le numéro d'ordre du datagramme, permet donc de l'identifier.

**FLAGS** Ils sont important pour la fragmentation, il y en a trois :

- X
- MF - More fragments - : Si le datagramme est fragmenté tous les fragments ont le flag MF à 1 sauf le dernier. Il indique donc que le datagramme n'a pas encore été entièrement reçu.
- DF - Don't fragment - : Est placé pour être sûr que le datagramme ne soit pas fragmenté - ce qui suppose de vérifier qu'il peut être acheminé avec sa taille actuelle.

**Fragment offset** Cet offset indique la position du début de ce fragment dans le datagramme d'origine.

**TTL** c'est la durée de vie du paquet

**Protocole** indique quel est le protocole de couche supérieure.

**CRC** Uniquement calculé sur l'en-tête IP

#### 6.1.2 Remarques sur l'ID et le Fragment offset

##### Fragment offset

Si le datagramme a une longueur de 1500 octets et est fragmenté en morceaux de 500 octets :

- Pour le premier fragment l'offset vaut 0
- Pour le second il vaut 500
- Pour le dernier il vaut 1000

En fait cet offset est codé sur 13 bits ce qui permet de coder des nombres de 0 à  $2^{13} - 1$ , ce qui est largement insuffisant, compte tenu que la taille maximale du datagramme est 65535 octets - soit  $2^{16} - 1$  -. Il est immédiat qu'il y a un facteur  $8 - 2^3$  - entre les deux. C'est pourquoi le fragment offset code en fait la position du début du fragment dans le datagramme d'origine mais en multiple de huit octets. Il n'est donc possible de fragmenter que sur un multiple de huit octets. L'exemple précédent est donc inexact et la vraie valeur du fragment offset est donc :

- Pour le premier fragment 0
- Pour le second il vaut 62
- Pour le troisième il vaut 124
- Pour le dernier il vaut 186

Il est donc nécessaire de faire un fragment supplémentaire, car  $500/8 = 62,5$  donc sur chaque fragment quatre  $0,5 * 8$  octets sont perdus.

## L'IP ID

L'IP ID est initialisé de manière arbitraire au boot de la machine. Il est ensuite incrémenté de un à chaque datagramme émis, indépendamment du message du port utilisé... Malheureusement cette définition de l'IP ID crée une faille de sécurité. Celle-ci peut être exploitée grâce à l'attaque nommée « idle port scan ».

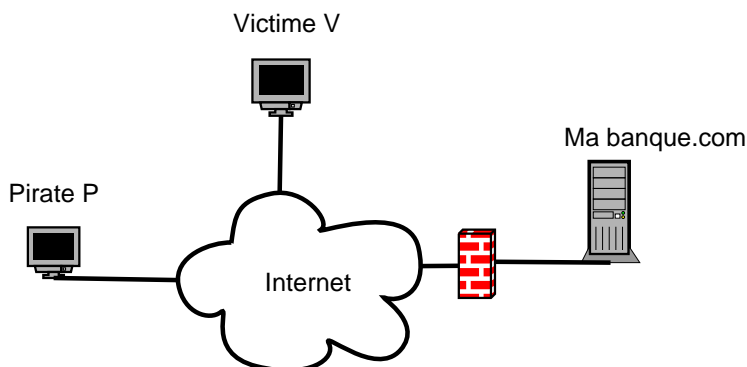


FIG. 6.1 – Schéma de l'attaque

Le pirate P veut attaquer le site de Ma banque.com. Cependant ce site enregistre tous les trafics donc une attaque directe se solde par une détection et le pirate est pris. L'idée est donc de se faire passer pour un tiers, la victime V pour réaliser l'attaque.

### Phase de découverte

si P envoie un message SYN vers le site Ma banque.com en ayant forgé un paquet où son adresse IP source est celle de V.

Si le port destination est valide le site renvoie un paquet SYN+ACK à V. Revenant une réponse à un paquet qu'il n'a pas envoyé V renvoie alors un RST. Si son IP ID vaut X au départ de la connexion et qu'il n'y a pas d'autres paquets envoyés, à l'issue de l'échange l'IP ID de V vaut donc X+1 - il n'a envoyé qu'un datagramme : le RST -. Si le port destination n'était pas ouvert le site Ma banque.com renvoie un datagramme RST à V. Celui-ci, qui n'a pas de raison particulière d'en attendre un l'ignore et n'y répond pas. Son IP ID vaut donc toujours X. Donc si l'IP ID de V augmente c'est que le port choisi est ouvert. Il reste un problème : P n'a pas d'accès direct à l'IP ID de v!

### Déroulement de l'attaque



1. Afin de connaître l'IP ID de V P envoie un datagramme à V. Le paquet retour contient l'IP ID, l'IPID, de V et met fin à la connexion si ce n'est pas déjà le cas.
2. P envoie un paquet SYN à Ma banque en se faisant passer pour V - adresse IP source : celle de V -.
3. P envoie un nouveau paquet à V et note l'IP ID contenu dans le paquet réponse :
  - soit il vaut IPID + 1 si le port utilisé pour Ma banque.com est fermé.
  - soit il vaut IPID + 2 si il est ouvert.
4. il est alors possible d'attaquer le site de Ma banque.com en blind à condition de flood V pour l'empêcher de lui envoyer un RST après réception d'un paquet.

Cette attaque a une limitation : il ne faut pas que V envoie de paquets entre temps ce qui ferait augmenter son IPID. Il n'est donc possible d'attaquer que des sites pas trop fréquentés. Néanmoins cette attaque est très simple à mettre en oeuvre. Le logiciel *Nmap* avec l'option *-sI* permet de la réaliser.

#### Parade

Il suffit d'incrémenter l'IPID de façon aléatoire. Par exemple le firewall Open BSD, *pf*, augmente de manière aléatoire la valeur de l'IP ID.

## 6.2 La fragmentation

La taille maximale d'un datagramme IP est de 65535 octets or la taille maximale d'une trame Ethernet est de 1518 octets. Il va donc être nécessaire de fragmenter les datagrammes avant de les passer à la couche 2.

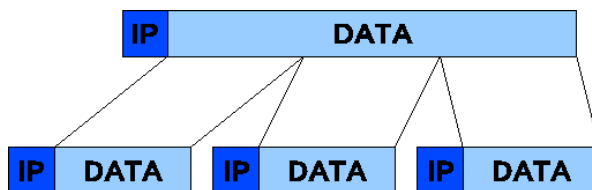


FIG. 6.2 – La fragmentation

### 6.2.1 Présentation

La taille maximale des paquets pouvant circuler sur un réseau donné, donc supportée par la couche 2, est donnée par la MTU - *Most Transportable Unit* -. Celle-ci est une caractéristique propre à chaque réseau et est définie pour le réseau local dans les paramètres de la carte réseau. C'est donc grâce à la MTU que la couche 3 sait comment fragmenter les datagrammes. Si la MTU évolue au cours du routage d'un réseau à l'autre c'est l'OS du routeur considéré qui va être responsable de refragmenter les paquets. Le ré-assemblage s'effectue au niveau de la machine destination. Le chemin suivi par chacun des paquets n'a donc aucune importance. L'OS de la machine origine fragmente pour se conformer au MTU de son réseau local ensuite chacun des paquets émis peut être refragmenté ou non en fonction de son chemin vers la machine destination qui sera chargé de ré-assembler tous les paquets pour retrouver le datagramme d'origine.

### 6.2.2 Champs de l'en-tête IP concernant la fragmentation

**L'IP ID :** il identifie le numéro du datagramme d'origine. Tous les fragments d'un même datagramme auront le même IP ID pendant tout le trajet. Si un paquet est refragmenté tous les nouveaux fragments conservent l'IPID d'origine - puisqu'ils sont tous issus du même datagramme.

**Les flags :** Il y en a deux qui concernent la fragmentation :

- MF : - more fragments - indique que le paquet n'est pas le dernier du datagramme d'origine. Il est donc positionné à 0 uniquement dans le dernier fragment du datagramme.
- DF : - don't fragment - indique que le paquet ne doit pas être fragmenté. Si il n'est pas compatible avec la MTU au cours de son trajet il est jeté.

**Le fragment offset :** il représente la position du début du fragment dans le datagramme d'origine. Or il est codé sur treize bits donc il peut prendre des valeurs allant jusqu'à 8191 ce qui est trop peu pour rendre compte de la taille possible des datagrammes. Il compte donc la position de la fragmentation en multiples de 8 octets - et  $8 * 8191 = 65528$  -, ce qui permet de décrire entièrement un datagramme.

Par exemple un paquet dont le fragment offset vaut 4 contient le fragment du datagramme d'origine qui commence à l'octet 32. Dans le cas où un routeur doit redécouper un paquet il garde l'IP ID mais il recalcule le fragment offset pour tous les sous paquets. À l'arrivée la machine de destination ré-assemble les différents paquets. Si elle constate un trou dans les offsets, à l'expiration d'un timeout elle jette tous les fragments concernant le datagramme et demande sa réémission.

### 6.2.3 Question

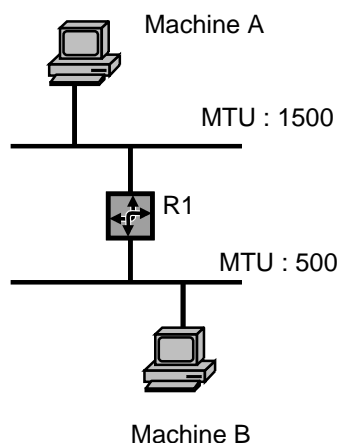
En utilisant un sniffer je vois passer sur le réseau des paquets de 1500 octets, or 1500 n'est pas un multiple de 8 !

Il est nécessaire d'avoir recours à des multiples de huit uniquement si il y a fragmentation. Donc ces paquets sont parfaitement valides, ils n'ont pas été fragmentés. Donc le fragment offset set nul et le bit MF est à zéro.

## 6.3 Exercice d'application

### 6.3.1 Énoncé

La machine A envoie un datagramme de 2000 octets. Écrire les informations de l'en-tête IP relative à la fragmentation.



### 6.3.2 Corrigé

NB : Il ne faut pas oublier les en-têtes IP. Le datagramme d'origine en comporte une mais tous les fragments doivent aussi en avoir une.

**sortie de A**

La MTU sur le réseau local vaut 1500. Compte tenu de l'en-tête IP de 20 octets il est possible de véhiculer 1480 octets maximum par paquet. Or 1480 est un multiple de huit et représente 185 blocs de huit octets. De plus le datagramme contient une en-tête IP - 20 octets -, donc 1980 octets de DATA. La machine A envoie donc :

**Paquet P1** : IPID, MF : 1, fragment offset : 0, envoi de 185 blocs de 8 octets de DATA - 1480 octets -, reste 500 octets de DATA ;

**Paquet P2** : IPID, MF : 0, fragment offset : 185, le paquet contient 500 octets de DATA.

**en R1**

Il faut refragmenter puisque la nouvelle MTU est de 500. Il est possible de véhiculer  $500 - 20 = 480$  octets de DATA par paquet, soit 60 blocs de 8 octets. R1 envoie donc sur le réseau :

– Pour le paquet P1 :

**paquet P'1** : IPID, MF : 1, FO : 0, envoi de 480/480 octets ;

**paquet P'2** : IPID, MF : 1, FO : 60, envoi de 480/960 octets ;

**paquet P'3** : IPID, MF : 1, FO : 120, envoi de 480/1440 octets ;

**paquet P'4** : IPID, MF : 1, FO : 180, envoi de 40/1480 octets ;

– Pour le paquet P2 :

**paquet P'5** : IPID, MF : 1, FO : 185, envoi de 480/1960 octets ;

**paquet P'6** : IPID, MF : 0, FO : 245, envoi de 20/1980 octets ;

Pour le paquet P'4 il n'est pas nécessaire de procéder à un padding puisque la taille des DATA dans le paquet est un multiple de huit, ce qui est garanti par la fragmentation réalisée par la machine d'origine. Elle découpe le datagramme en multiples de huit octets.

**Remarque :**

La MTU est définie par la formule  $x * 8 + 20$  octets, sinon la taille maximale des paquets de la couche IP ne pourra être atteinte que pour des paquets tels que MF = 0.

**6.4 Gestion du flag DF**

Si un routeur reçoit un paquet dont le flag DF est positionné à 1, et que la MTU du réseau suivant est inférieure à la taille du paquet, le paquet est jeté et un message ICMP est envoyé à l'adresse IP source pour l'en informer. Cette propriété peut être utilisée pour déterminer la MTU optimale, c'est à dire la plus grande pour laquelle les paquets ne seront plus refragmentés jusqu'à destination. C'est ce qui s'appelle le *path MTU discovery*.

**Principe du path MTU discovery**

La méthode consiste à envoyer des paquets, en activant le flag DF, de taille importante et de faire décroître cette taille. Tant que la taille est supérieure à une des MTU sur le trajet le paquet est rejeté et un message ICMP est envoyé à la machine d'origine. Donc le premier paquet qui atteint la destination, indiqué par une réponse de la machine destination, donne la MTU sur le parcours.

## 6.5 Exemple d'attaque utilisant la fragmentation

La taille minimale des paquets pour Ethernet est de 64 octets. C'est cette taille minimale qui va être mise à profit.

Les ports source et destination sont placés en tête d'en-tête TCP. Donc si la fragmentation est forcée juste après l'en-tête IP, quitte à rajouter un peu de padding pour atteindre les 64 octets minimum, les numéros de port seront situés dans le second paquet. Considérons que le firewall du réseau bloque le port 80 en sortie, donc il n'est pas possible de se connecter sur un site web. Compte tenu de la fragmentation adoptée, le firewall voit passer le premier paquet et n'a aucune raison de le bloquer. Cependant, si il accepte le premier paquet, il doit laisser passer tous les paquets liés à cette connexion<sup>1</sup>. Donc il laisse s'établir une connexion sur le port 80, qu'il est censé empêcher. Avec cette attaque tous les ports de toutes les machines sont accessibles.

### Parade

Il suffit de fixer comme règle, pour le firewall, de refuser tous les paquets qui ne contiennent pas toutes les en-têtes de niveau 2, 3 et 4.

Une autre approche consiste à faire ré-assembler les datagrammes par le firewall.

## 6.6 Influence de la MTU

Le choix de la MTU influe sur le débit du réseau. Aujourd'hui, la plupart des réseaux sur Internet utilisent Ethernet, donc laissent passer les paquets de 1500 octets, sinon c'est à l'utilisateur de s'adapter.

---

<sup>1</sup>voir le chapitre 9 sur les firewalls 63

## Chapitre 7

# Retour sur le protocole TCP

### 7.1 Description de l'en-tête TCP

4 bits	6 bits	6 bits	2 octets
port SRC			port dest
N° de séquence			
N° d'acquittement			
data offset	Réservés	FLAGS	Window
CRC			URGENT POINTER
OPTIONS		Bourrage	

TAB. 7.1 – En tête TCP

Les différents champs de cette en-tête sont :

**PORT SRC** : désigne le port source, c'est l'adresse utilisée par l'application de la machine.

**PORT DEST** : désigne le port destination, c'est l'adresse utilisée par l'application sur la machine destinataire.

**DATA OFFSET** : c'est un pointeur qui indique le début des données, en fonction du nombre d'options indiquées dans le paquet.

**FLAGS** : c'est là que sont indiqués les différents flags de couche 4 : FIN, SYN, ACK, PUSH, URG...

**CRC** : c'est un code de correction d'erreur qui ne porte que sur l'en-tête de couche 4.

**URGENT POINTER** : si le flag URG est indiqué, ce pointeur indique la fin des données urgentes. La norme n'est pas claire, il existe donc deux conventions : soit il indique le premier octet non urgent, soit il indique le dernier octet urgent.

**Réservé** : ces octets sont inutilisés.

**N° de séquence** : correspond à la quantité de données envoyée à l'autre machine - nombre d'octets -.

**N° d'acquittement** : correspond au nombre d'octets de données reçus de l'autre machine, permet donc de gérer les réémissions par comparaison avec le numéro de séquence.

### 7.2 Mécanisme de contrôle d'acheminement

Ce mécanisme s'appuie sur l'échange des numéros de séquence et d'acquittement entre les deux machines.

### 7.2.1 Exemple

Si une machine A envoie un segment contenant 500 octets de données à une machine B avec un numéro de séquence à 500 - nombre d'octets envoyés depuis le début de la connexion- et un nombre d'acquittement à 0 - nombre d'octets de données reçus de la machine B depuis le début de la connexion -. Elle reçoit en réponse 300 octets de données de la machine B, avec un numéro séquence à 300 - A a donc reçu tous les octets depuis le début de la transaction -, et un numéro d'acquittement à 0. Il apparaît donc que la machine B a manqué un segment ou que celui-ci n'a pas encore trouvé sa destination.

$$\begin{array}{l} A \xrightarrow{500 \text{ octets}} B \\ A \xleftarrow{300 \text{ octets}} B \end{array}$$

### 7.2.2 Description

Il y a des segments de service en permanence sur le réseau, échangées par les deux machines pour rester synchronisées - chaque machine peut ainsi savoir ou en est l'autre machine de la réception des infos envoyées -. Donc si une machine, disons A, n'a pas d'informations à envoyer, elle envoie des segments ACK. Le numéro de séquence est codé sur quatre octets, ce qui permet d'éviter d'avoir, pour deux machines, le même doublet - numéro de port, numéro de séquence - identifiant une connexion. En effet, contrairement à l'exemple précédent, le numéro de séquence n'est pas initialisé à un en début de communication, afin d'éviter les interférences. Au moment de la connexion, il faut donc choisir un numéro de séquence initial - ISN : Initial sequence number - pour le premier paquet envoyé. Au boot de la machine, l'ISN vaut zéro, puis il est incrémenté de 64000 par demi seconde et de 64000 de plus pour chaque connexion TCP. Donc si la machine a booté depuis une seconde et a établi une connexion TCP  $ISN = 64000 * 2 + 64000 = 192000$ . Il y a donc peu de chances que deux connexions utilisent les mêmes numéros de séquence. Le bouclage complet de l'ISN prend environ neuf heures. A chaque fois que je veux ouvrir une connexion TCP je choisis donc l'ISN pour le premier numéro de séquence.

$$\begin{array}{l} A \xrightarrow[N^{\circ}_{seq}=ISN_A]{N^{\circ}_{ACK}=0} B \quad \text{FLAG SYN pas de données} \\ \xleftarrow[N^{\circ}_{seq}=ISN_B]{N^{\circ}_{ACK}=ISN_A+1} \quad \text{FLAGS SYN + ACK} \\ \xrightarrow[N^{\circ}_{seq}=ISN_A+1]{N^{\circ}_{ACK}=ISN_B+1} \quad \text{FLAGS ACK} \end{array}$$

Dans le segment contenant le flag SYN il n'y a pas de données, cependant, pour indiquer que le SYN a bien été reçu, le numéro d'acquittement est incrémenté de un - idem pour FIN-. A ce moment la connexion est établie entre A et B.

$$\begin{array}{l} A \xrightarrow{50 \text{ octets}} B \quad NS = ISN_A + 51 \quad NA = ISN_B + 1 \\ \xleftarrow{300 \text{ octets}} \quad NS = ISN_B + 301 \quad NA = ISN_A + 51 \\ \xrightarrow{50 \text{ octets}} \quad NS = ISN_A + 101 \quad NA = ISN_A + 301 \end{array}$$

On assiste ensuite à l'échange simultané de deux segments entre A et B

A	$\xleftarrow[200 \text{ octets}]{simult}$ $\xrightarrow[100 \text{ octets}]{simult}$ $\xrightarrow{20 \text{ octets}}$ $\xrightarrow{FIN}$ $\xrightarrow{ACK+FIN}$ $\xrightarrow{ACK}$	B	$NS = ISN_B + 501$ $NS = ISN_A + 121$ $NS = ISN_B + 601$ $NS = ISN_A + 121$ $NS = ISN_B + 601$ $NS = ISN_A + 122$	$NA = ISN_A + 101$ $NA = ISN_B + 301$ $NA = ISN_A + 121$ $NA = ISN_B + 601$ $NA = ISN_A + 122$ $NA = ISN_B + 602$	   FLAG FIN FLAGS ACK + FIN FLAG ACK
---	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---	----------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------

## 7.3 L'attaque de Noël

### 7.3.1 Historique

C'est un défi entre deux informaticiens : Mitnick contre Shimomura. Ce dernier lance un challenge : je dispose d'informations protégées sur une machine dont l'adresse IP est @IPc. Si tu peux t'en emparer alors tu es plus fort que moi. Le détail omis par Shimomura, était qu'il travaillait pour le gouvernement américain et que ces informations étaient classifiées. Mitnick releva le défi et gagna, cependant il fut poursuivi par le gouvernement qui lui donna le choix suivant : aller en prison ou travailler pour lui. Mitnick choisit la prison. Depuis il lui est interdit de s'approcher d'un ordinateur.

### 7.3.2 Stratégie

Mitnick détermine d'abord que Shimomura dispose d'un ordinateur chez lui -@IPSm- et que cette machine est autorisée à se connecter à la machine à @IPc. L'idée ensuite est une attaque le soir de Noël, pour que les administrateurs soient avec leurs familles et non pas devant leur console. L'attaque est une attaque en spoofing - qui consiste à se faire passer pour une autre machine, en l'occurrence celle du domicile de Shimomura-, et plus précisément en blind IP spoofing. En effet la réponse de la machine cible -@IPc- répond toujours à @IPSm et non pas à l'attaquant qui ne la voit donc jamais.

### 7.3.3 Déroulement de l'attaque

1. Mitnick envoie un SYN vers @IPc ce qui lui permet de récupérer l'ISN de la machine cible - en utilisant le temps écoulé et la règle d'évolution de l'ISN : + 64000 par demi seconde et par connexion TCP-.
2. Mitnick envoie un SYN à la machine cible en forgeant un segment avec pour adresse IP source l'adresse IP @IPSm. La machine cible répond à @IPSm SYN + ACK
3. à partir du temps écoulé depuis le premier envoi, Mitnick calcule l'ISN de @IPc et envoie un segment ACK, toujours avec pour adresse source @IPSm.

En fait il envoie une série de segments avec des numéros de séquence décalés de 64000 pour être sûr de tomber sur le bon avec un des segments. Le three ways handshake a donc été réalisé, la connexion est établie. Mitnick envoie alors une unique commande, toujours avec l'adresse source @IPSm, qui réalise un `echo ++` dans `etc/rhosts` ce qui a pour conséquence d'autoriser n'importe quelle machine à se connecter sur la machine cible. Mitnick peut donc se connecter directement sur la machine cible pour récupérer les fichiers classifiés, objets du défi.

En fait l'attaque ne marche pas telle quelle. En effet, quand la machine du domicile de Shimomura -@IPSm- reçoit le SYN + ACK, alors qu'elle n'a pas envoyé de SYN, elle doit renvoyer RST, interrompant la communication. Pour contourner le problème, Mitnick a eu recours au SYN flood. Le but est de « floder » la machine à @IPSm avec des messages SYN pour la saturer et l'empêcher de traiter les messages en provenance de la machine cible. Elle n'est donc pas en mesure d'envoyer le RST sa pile TCP/IP étant pleine.

### 7.3.4 Remarque

La connexion ainsi établie par Mitnick avec la machine cible lui permet d'envoyer des segments mais pas de prendre des infos, la réponse étant systématiquement envoyée à la machine à @IPSm.

### 7.3.5 La parade

L'attaque ainsi décrite est structurelle, elle est liée à la conception même du protocole TCP. Cependant pour prévenir ce type d'attaques les ISN sont maintenant générés aléatoirement - les décalages ne sont plus fixés -. La génération des ISN sont en fait le résultat d'un processus pseudo aléatoire. Donc en envoyant beaucoup de paquets on peut remonter au générateur d'aléa. Toutefois, aujourd'hui, le spoofing sur Internet est très délicat à mettre en oeuvre.



# Chapitre 8

## La traduction d'adresse : NAT

### 8.1 Introduction

La NAT (*Network Address Translation*) est un mécanisme destiné à modifier les adresses IP dans l'en-tête d'un datagramme IP. Il recouvre en fait différents mécanismes : NAT statique, NAT dynamique, port forwarding. Si le premier est un mécanisme de couche 3, en revanche les suivants utilisent également des informations de couche 4.

### 8.2 La NAT statique

#### 8.2.1 Le principe

La NAT statique consiste à substituer  $n$  adresses IP externes à  $n$  adresses IP internes. A chaque adresse IP interne est associée une adresse IP externe. Au passage du paquet, la seule action effectuée par le routeur est de remplacer l'adresse source ou destination par l'adresse correspondante.

#### 8.2.2 Intérêt de la NAT statique

Dans un réseau configuré en adresses privées si une machine veut envoyer un paquet sur Internet, vers `www.google.com`. Dans l'en-tête IP, l'adresse en destination est celle de `www.google.com`, et l'adresse IP source est, par exemple, 10.0.0.1. Lorsque google veut répondre à la requête il utilise donc pour adresse destination 10.0.0.1. Or les adresses privées ne sont pas routées sur Internet. En conséquence notre machine ne recevra jamais la réponse de Google. Pour résoudre le problème une des solutions est d'utiliser la NAT statique en associant  $n$  adresses privées à  $n$  adresses publiques.

#### 8.2.3 Le fonctionnement de la NAT statique.

Pour que notre machine puisse dialoguer sur Internet avec d'autres machines nous allons donc associer une adresse publique virtuelle à son adresse privée. Le routeur va donc être responsable de modifier, dans l'en-tête IP du paquet, l'adresse source privée, pour les paquets sortants, par l'adresse publique virtuelle, et pour les paquets entrants, l'adresse destination publique virtuelle par l'adresse privée. Ainsi, notre machine est vue de l'Internet avec l'adresse publique. La NAT statique nous a donc permis de rendre une machine accessible sur Internet alors qu'elle possédait une adresse privée. Les associations adresse publique↔adresse privée sont répertoriées dans la table NAT du routeur.

### 8.2.4 Avantages et inconvénients de la NAT statique.

L'association d'une adresse IP publique à une adresse IP privée, nous a permis de rendre une machine accessible sur Internet. En revanche nous avons besoin d'une adresse publique par adresse privée ce qui ne nous aide pas à résoudre le problème de la pénurie des adresses IP. D'autre part pourquoi ne pas utiliser directement les adresses publiques sans passer par les adresses privées? En fait la NAT statique apporte plusieurs avantages : tout d'abord il est souvent préférable de garder un adressage uniforme en interne et de ne pas mêler les adresses publiques aux adresses privées. Ainsi, si il faut faire des modifications, changements, interventions sur le réseau local, il suffit de changer la correspondance entre les adresse privées et les adresses publiques dans la table NAT du routeur pour rediriger les requêtes vers un serveur en état de marche. D'autre part, on gâche un certain nombre d'adresses lorsqu'on découpe un réseau en sous-réseaux -adresse de réseau, adresse de broadcast...-, comme pour une DMZ afin de rendre ses serveurs publics disponibles. Avec la NAT statique, on évite de perdre ces adresses.

Malgré ces quelques avantages, le problème de pénurie d'adresses n'a toujours pas été réglé. Pour cela, il va falloir avoir recours à un autre mécanisme : la NAT dynamique.

### 8.2.5 Problèmes de routage liés à l'utilisation de la NAT statique -proxy ARP- Proxy ARP

Ces problèmes ne sont pas toujours rencontrés lors de l'implémentation de la NAT statique. Si celle-ci est bien faite, tous les mécanismes décrits devraient être implémentés de façon transparente pour l'utilisateur. Le premier problème rencontré est celui de la redirection d'un paquet vers l'adresse virtuelle de la NAT statique.

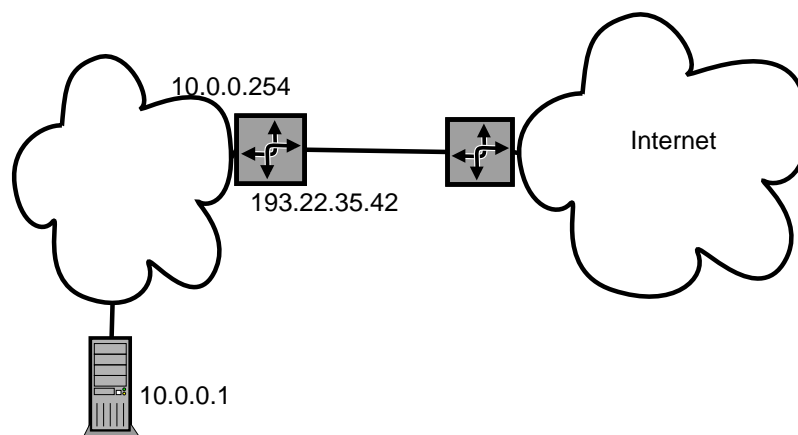


FIG. 8.1 – NAT

Considérons une machine 1, dans un réseau privé, faisant une requête vers le site `www.google.com`. Le paquet est NATé au niveau du routeur avec comme adresse source l'adresse publique, disons 193.22.35.43, ainsi, le site `www.google.com` renvoie sa réponse vers cette adresse. Le paquet est routé sur Internet et arrive sur le routeur Internet précédant le routeur de notre réseau. Celui-ci regarde l'adresse de destination et observe qu'elle se situe sur le même réseau qu'une de ses interfaces. Ainsi, elle a maintenant besoin de l'adresse MAC de la machine 193.22.35.43 pour lui envoyer le paquet. Elle fait donc une requête ARP en demandant "Quelle est l'adresse MAC de la machine ayant 193.22.35.43 comme adresse IP ?" Or, sur ce réseau, aucune machine n'a cette adresse puisqu'il s'agit d'une adresse virtuelle. Il faut donc que le routeur -193.22.35.42- réponde lui-même à cette requête ARP. C'est ce

que l'on appelle faire proxy ARP. Quand la NAT statique est implémentée, le proxy ARP l'est souvent automatiquement, cependant, il est bon de connaître ce mécanisme si ce n'est pas le cas. Il y a plusieurs façon de pallier à ce problème.

- mettre en place soit même un mécanisme de proxy ARP sur la machine faisant la NAT statique.
- ajouter une entrée statique dans la table ARP du routeur Internet -pas le routeur faisant la NAT, mais le premier routeur rencontré après celui-ci sur Internet-.
- ajouter une route host statique pour chacune des adresses virtuelles.

### Routage sur la passerelle

Un second problème peut concerner l'équipement réalisant la NAT. Revenons à l'exemple précédent.

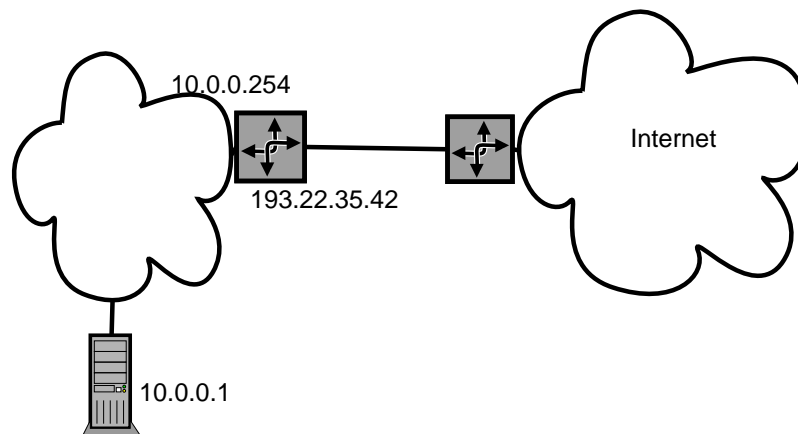


FIG. 8.2 – NAT

Le routeur Internet envoie le paquet au routeur de l'entreprise. Celui-ci reçoit la trame Ethernet, lit son adresse MAC en destination, il envoie donc le contenu des données à la couche IP. Celle-ci lit l'adresse 192.22.35.43 -l'adresse virtuelle de notre machine- en destination. Il va lire sa table de routage, et :

- soit une route spécifique existe pour cette adresse pour rediriger le paquet vers le réseau interne,
- soit ce n'est pas le cas, et il sera renvoyé sur l'interface externe du routeur, vu que l'adresse de destination appartient au même réseau que son interface externe 193.22.35.42!

Donc pour que la NAT fonctionne, il faut donc qu'il y ait une route spécifique vers le réseau interne.

Pour définir cette route il faut utiliser la commande suivante :

```
route add -p 193.22.35.43 mask 255.255.255.255 10.0.0.1
```

Ainsi, quand le routeur recevra un paquet à destination de l'adresse virtuelle 193.22.35.43, il le redirigera bien vers l'adresse réelle de la machine, 10.0.0.1.

## 8.3 La NAT dynamique

### 8.3.1 Le principe

La NAT dynamique, aussi appelée IP masquerading, associe une seule adresse publique à n adresses privées. Ainsi, un grand nombre de machines ayant des adresses privées peuvent accéder à Internet! En revanche, cette méthode possède quelques inconvénients. Contrairement à la NAT statique, le routeur qui effectue la NAT devra à la fois modifier les adresses IP mais aussi les ports . C'est ce qui s'appelle le PAT - Port Address Translation -

### 8.3.2 Le fonctionnement de la NAT dynamique

Le fonctionnement est un peu différent de celui de la NAT statique. Reprenons l'exemple précédent : Cette fois, c'est l'adresse publique de l'interface externe du routeur 193.22.35.42 qui va être utilisée pour sortir. Ainsi, lorsque le paquet arrive à la machine de destination, `www.google.com` par exemple, celle-ci le renvoie vers l'adresse 193.22.35.42. Le routeur reçoit donc ce paquet et lit que sa propre adresse de destination. Comment faire la différence entre un paquet qui lui est réellement adressé et un paquet destiné à une machine du réseau privé ? C'est grâce aux ports qu'il va pouvoir faire la différence. Ainsi, si une machine en interne fait une requête avec comme port source 2356, le routeur pourra savoir que lorsqu'il recevra un paquet avec comme port destination 2356, il faut le rediriger vers la machine en interne qui a initialisé la connexion. Il est donc clair que deux machines ne doivent pas utiliser le même port source sinon le routeur ne sera plus capable de savoir à qui il doit renvoyer le paquet. Or le port source est choisit par le système d'exploitation donc comment s'assurer que deux OS ne choisiront pas le même port source ? En fait, pour un paquet sortant, le routeur va remplacer à la fois l'adresse IP source et le port source. Il choisit un de ses port source libre et conserve dans sa table NAT le binôme adresse IP interne ↔ port interne en correspondance avec le numéro de port qu'il a choisi.

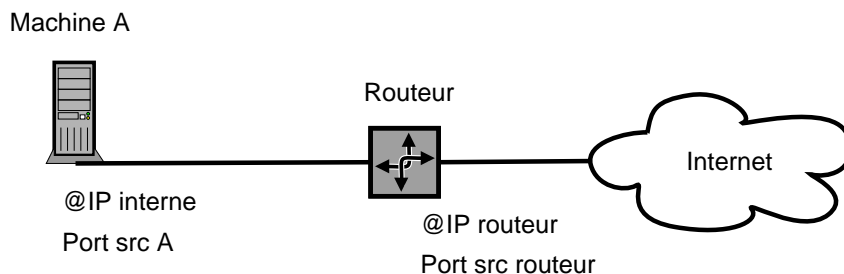


FIG. 8.3 – Nat dynamique en sortie du réseau local

Ainsi tout paquet revenant sur ce port sera renvoyé avec le binôme adresse IP interne ↔ port interne correspondant de la table NAT vers la machine voulue et les paquets sortant du réseau local auront après le passage du routeur un numéro de port correspondant à la connexion. Ces numéros de port seront différents pour deux connexions différentes - deux machines différentes en interne - puisque c'est le routeur qui les choisi parmi ses ports disponibles.

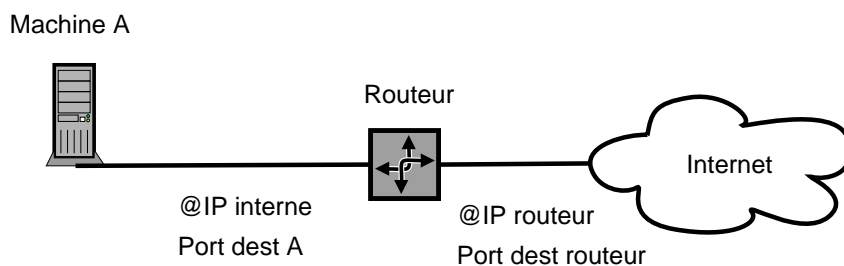


FIG. 8.4 – NAT dynamique en entrée du réseau local

### 8.3.3 résumé

La machine 10.0.0.1 veut se connecter au site `www.google.com`, elle envoie donc un paquet avec comme adresse source la sienne, 10.0.0.1, et comme port source un port quelconque supérieur à 1024 choisit par son système d'exploitation, soit par exemple 5987. Le paquet arrive au routeur réalisant la NAT qui remplace donc l'adresse IP source par la sienne 193.22.35.42, et le port source 5987 par un de

ses ports libres, 10000 par exemple. Il garde ces informations de correspondance dans une table NAT. Le paquet arrive à `www.google.com` qui le renvoie à 193.22.35.42. Le paquet arrive au routeur, il voit que l'adresse destination est lui-même, il regarde donc le port destination qui est 10000 puis consulte la table NAT pour avoir la correspondance. Il envoie donc ce paquet à 10.0.0.1, tout en ayant modifié le port destination 10000 en 5987 qui est le port sur lequel 10.0.0.1 a initialisé la connexion.

Il est ainsi possible de masquer autant de machines que nécessaire derrière une seule adresse publique! Inversement la NAT permet à des machines ayant des adresses privées d'accéder à Internet. En revanche elle ne permet pas à ces machines d'être joint depuis Internet. En effet pour une trame sortante le routeur qui fait la NAT reçoit les informations adresse privée et port privé avec la première trame sortante et c'est ces informations qui lui permettent de rediriger les trames réponse. En revanche pour une trame entrante il a pour seule information l'adresse IP destination - la sienne -, le port destination - celui de l'application - mais il n'a aucun moyen de retrouver à quelle machine interne la trame est destinée. La NAT ne permet donc pas de rendre un serveur accessible de l'extérieur. D'autre part, même si ce n'est pas le but de la NAT, le fait que les machines internes ne soient pas accessibles de l'extérieur permet un petit gain de sécurité.

### 8.3.4 Problèmes liés à la NAT dynamique

#### Avec le protocole ICMP

La Nat dynamique repose donc sur l'utilisation des ports de couche 4, or certains protocoles utilisés sur le réseau n'y ont pas recours - par exemple ICMP, Netbios... -. Considérons le protocole ICMP, c'est un protocole de couche 3, il n'utilise donc pas les ports de couche 4. Il n'est donc pas possible d'implémenter la NAT. Or le protocole ICMP est largement implémenté. Il faut donc adapter la NAT pour lui permettre d'agir sur le trafic ICMP. Les ports n'étant pas disponibles, c'est l'identifiant ICMP, situé dans l'en-tête ICMP, qui va être utilisé à leur place. Le reste du mécanisme est inchangé. Enfin certains paquets ICMP contiennent dans leurs data des informations sur le datagramme IP ayant causé l'erreur. Pour que l'information apportée à la machine émettrice soit exploitable il est donc nécessaire que le routeur qui fait la NAT les modifie.

#### Avec le protocole FTP

Le protocole FTP utilise deux connexions en parallèle. La première sert au contrôle de la connexion et la seconde au transfert des données. Ce protocole peut fonctionner en mode actif ou en mode passif. En mode passif c'est le client qui initialise les deux connexions sur les ports 20 et 21. Il n'y a donc pas de problème avec la NAT. En revanche, en mode actif, le client initialise la connexion de contrôle, mais, lorsque des données sont demandées, c'est le serveur qui initialise la connexion de données. Or comme nous l'avons déjà remarqué précédemment, il n'est pas possible d'initialiser une connexion à partir de l'extérieur du réseau local avec la NAT dynamique. Enfin, le protocole FTP contient des données relatives aux adresses des machines, ce qui n'est pas sans incidence sur la NAT. Donc pour utiliser le protocole FTP avec de la NAT il faut pouvoir lire les informations contenues dans les données FTP, ce qui nécessite l'utilisation d'un proxy <sup>1</sup> qui sera capable de suivre la connexion et de modifier les données du paquet ftp.

## 8.4 Rendre les machines du réseau local joignables malgré la NAT

Dans le cas de la NAT statique ça ne pose pas de problèmes particulier en revanche avec la NAT dynamique nous avons vu que ce n'était pas possible directement. Il existe deux mécanismes, que nous allons étudier permettant de résoudre ce problème : le **port forwarding** et le **port mapping**.

---

<sup>1</sup>voir §8.5 page 62

### 8.4.1 Le port forwarding

Le port forwarding permet de rediriger un paquet vers une machine précise en fonction du port destination contenu dans le paquet. C'est associer un port à une adresse IP en entrée du réseau. Tous les ayant ce port en destination seront redirigés vers la même machine. Il est ainsi possible d'initialiser une connexion de l'extérieur du réseau local vers une ses machines - une seule connexion est possible par port -. Ainsi si le une machine du réseau héberge un serveur FTP, il suffit de configurer le routeur pour qu'il redirige toutes les connexions vers le port 21 sur cette machine. Elle devient ainsi joignable de l'extérieur. Le routeur associe le couple -adresse IP interne, port 21- à port 21. Le port forwarding nous permet donc de rendre des machines du réseau local visibles depuis Internet en ayant une unique adresse publique.

### 8.4.2 Le port mapping

C'est un procédé comparable au port forwarding, sinon qu'il ne modifie également le port de destination. Il redirige une requête sur un port X vers un port Y. Par exemple si une des machines du réseau local héberge un serveur web accessible via le port 8080 depuis le réseau local, et qu'il doit être visible depuis Internet, il suffit de rediriger le port 80 vers l'adresse du serveur et le port 8080. Ainsi les internautes accéderont au serveur web en utilisant le port standard.

### 8.4.3 Les limites du port forwarding

Nous avons donc vu qu'il était possible d'associer un port avec une machine en interne pour une adresse publique. Donc si le réseau local héberge plusieurs serveurs du même type - disons FTP - il faudra autant d'adresses publiques que de serveurs FTP, chacun étant rattaché à une adresse publique.

## 8.5 Les proxys

### 8.5.1 Présentation

Un proxy est un intermédiaire dans une connexion entre le client et le serveur. Le client s'adresse toujours au proxy et c'est lui qui s'adresse ensuite au serveur. Le proxy fonctionne pour une application donnée - http, ftp, smtp, etc. -. C'est donc un élément de couche applicative 7. Il est donc capable de modifier les informations contenues dans les trames - il est familier avec l'application utilisée -. En contrepartie, il faut un proxy par application. Cependant beaucoup de produits sous l'appellation proxy sont en fait des multi-proxys, capables de comprendre la plupart des applications courantes. En revanche, il faut différencier le proxy des fonctionnalités complémentaires qui lui sont souvent associées. Faire de la NAT, ou réaliser un serveur cache ne sont que des fonctionnalités additionnelles, même si elles sont souvent utiles. En effet, comme il centralise l'accès à l'Internet il représente un étranglement des flux entrants et sortant. Le fait de garder en cache les pages web permet d'améliorer les performance si plusieurs utilisateurs cherchent à accéder au même site web. D'autre part tous les paquets devant passer par lui la NAT est souvent nécessaire et il modifiera les adresses des paquets. Cependant ce n'est pas systématiquement le processus employé.

## Chapitre 9

# Les Firewalls

### 9.1 Introduction

Un firewall, aussi appelé pare-feux ou garde-barrières, est un programme, ou un matériel, chargé de vous protéger du monde extérieur et de certains programmes malveillants placés à votre insu sur votre ordinateur. Placé entre vous et Internet, le firewall contrôle tout ce qui passe, et surtout tout ce qui ne doit pas passer de l'un vers l'autre.

### 9.2 Les différents types de firewall

Il y a trois types de firewalls : ceux qui réalisent un filtrage simple, un filtrage à états et les proxys.

#### 9.2.1 Le filtrage simple

Le firewall ne fait qu'une vérification sur les différents sur les différentes en-têtes présentes dans les trames Ethernet. Il faut parfaitement maîtriser les principes TCP/IP et mettre en place plusieurs règles pour identifier un flux.

#### 9.2.2 Le filtrage par états

Le firewall possède une table dans laquelle il conserve l'état des connexions actives. Ainsi, il peut savoir si un paquet appartient à une connexion active grâce au quadruplet adresse IP source et destination et ports source et destination. Pour une connexion TCP le premier message SYN est filtré et plus les suivants, associés à la même connexion. Ces firewalls opèrent donc en couches trois et quatre et ils représentent 95% des firewalls déployés.

#### 9.2.3 Les proxys

Le firewall joue le rôle d'un proxy, c'est à dire d'un mandataire applicatif qui sait interpréter les données applicatives. Ainsi, le filtrage peut remonter aux données de couche 7.

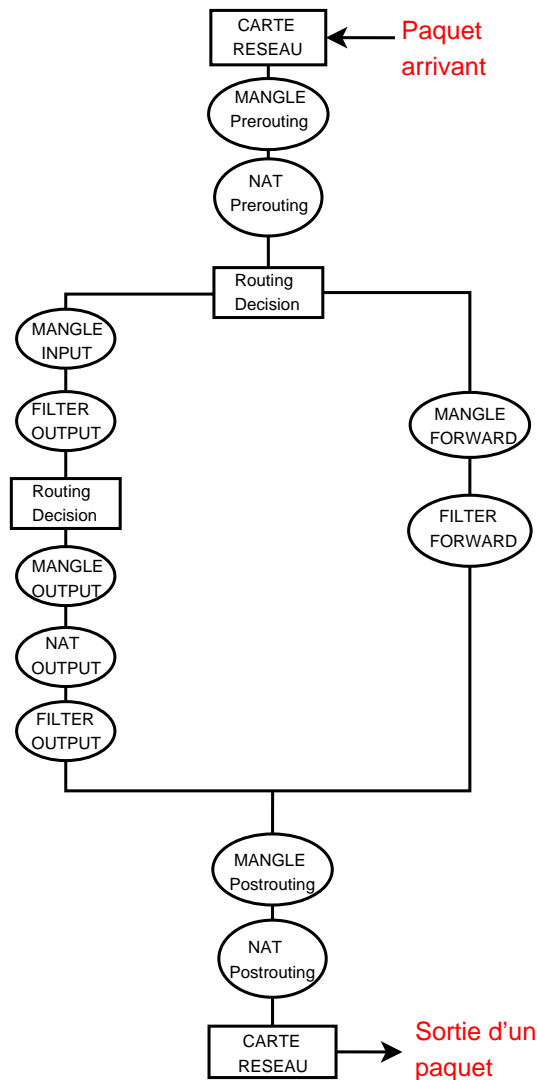
### 9.3 Le filtrage sous Linux

Linux possède un moteur de filtrage intégré au noyau - depuis le 2.4 -, **Netfilter** qui est un firewall à états. Le précédent moteur de filtrage, jusqu'au noyau 2.2, **ipchain** réalisait un filtrage simple. Netfilter se configure à travers une interface en ligne de commande qui est l'utilitaire **iptables**. Netfilter permet le filtrage, mais aussi la translation d'adresses - port forwarding -, le taggage des paquets,...

### 9.3.1 Fonctionnement de Netfilter

#### Architecture générale

Le filtrage est réalisé à différents niveaux : avant, après le routage et avant ou après la NAT. Les paquets reçus suivent un certain nombre d'étapes avant d'être traités par la machine ou d'en ressortir. Netfilter possède trois tables principales - MANGLE, NAT et FILTER - qui ont chacune un rôle différent. La table MANGLE sert à modifier les paquets, la table NAT à effectuer la traduction d'adresses et la table FILTER à filtrer. Chacune de ces tables possède des règles principales - par exemple pour FILTER les chaînes INPUT, OUTPUT et FORWARD -, mais il est également possible de créer des chaînes personnalisées.<sup>1</sup>



L'endroit où est réalisé la NAT n'est pas sans incidence sur le filtrage. Nous allons donc mettre en place des règles avec des critères au sein de ces chaînes et les paquets seront filtrés si ils coïncident avec les critères des règles. Chaque règle contient donc des critères - flag SYN, flag ACK, port X... - ainsi que des actions à effectuer.

<sup>1</sup>voir le tutoriel d'OSKAR ANDREASSON



### Le filtrage à états

Netfilter possède un moteur à états qui permet de dire si un paquet appartient à une connexion déjà établie. Un paquet peut avoir les états suivants :

**NEW** : premier paquet de la connexion

**ESTABLISHED** : à partir du second paquet



**RELATED** : Par exemple sur une connexion en UDP, si le serveur tombe, il répond par un message ICMP « host unreachable ». Netfilter regarde alors dans le paquet ICMP qui contient une partie du paquet d'origine. Il est donc classé RELATED par Netfilter qui le laisse passer - RELATED signifie donc lié à une connexion déjà établie -. Il en est de même avec une connexion FTP qui comprend en fait deux connexions sur les ports 20 et 21 mais sont liées.

### INVALID

Ces critères pourront être pris en compte dans une règle.

#### 9.3.2 Le fonctionnement de iptables

Iptables permet d'écrire les règles qui seront interprétées par Netfilter. Les règles sont du type :

```
iptables -t TABLE -A CHAINE -xxx CRITERE -j ACTION
```

**TABLE** : désigne la table dans laquelle la règle sera insérée - NAT, MANGLE, FILTER -par défaut-

**CRITERE** : permet de renseigner les critères de choix - adresse IP source, port destination...-

**ACTION** : peut prendre les valeurs ACCEPT, DROP, LOG, envoi dans une chaîne.

Par exemple pour autoriser les débuts de connexion http vers la machine il faut utiliser :

```
iptables -t filter -A INPUT -m state --state NEW -p tcp --dport 80 --syn -j ACCEPT
```

Dès que Netfilter rencontre une règle applicable au paquet il arrête la recherche et exécute les actions correspondantes. Il n'y a qu'une exception à cette règle c'est si l'action est « LOG » Il est donc possible d'effectuer deux actions si l'une d'elle est LOG - exemple LOG et DROP -.

L'idée de base est que tout ce qui n'est pas explicitement autorisé doit être interdit. Donc il faut écrire les règles pour les paquets utilisés et la dernière règle doit être un DROP de tous les paquets qui ont réussi à passer.

Configuration par défaut Le traitement des règles se fait donc dans l'ordre d'apparition de celles-ci. Dès qu'un paquet coïncide avec une règle celle-ci lui est appliquée et le processus de filtrage est interrompu pour le paquet. Si aucune règle ne s'applique au paquet c'est la politique par défaut qui est appliquée. C'est l'option -P qui définit la politique par défaut.

Par exemple, pour définir une politique de rejet de tous paquets il faut :

```
Iptables -P INPUT DROP
```

```
Iptables -P OUTPUT DROP
```

```
Iptables -P FORWARD DROP
```

#### 9.3.3 La traduction d'adresses

Pour la traduction d'adresses il y a trois cibles supplémentaires possibles : SNAT, DNAT, MASQUERADE. Par exemple pour faire de la NAT dynamique derrière une adresse IP Internet 194.2.232.1 il faut écrire :

```
Iptables -t nat -A POSTROUTING -o eth0 -j SNAT -to -source 194.2.232.1
```

ou encore :

```
Iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE -to -source 194.2.232.1
```

Cette règle crée implicitement deux règles : une pour les paquets sortant et une pour les paquets entrant. Un firewall est donc en quelques sortes un routeur amélioré.

Pour définir une chaîne personnalisée il faut écrire :

```
Iptables -N CHAINE
```

## 9.4 Liste des principaux ports

- Port 20 Ce port est utilisé lors des connexions FTP dynamiques.
- Port 21 Ce port est utilisé pour les connexions FTP - téléchargement de logiciels -.
- Port 25 Ce port est utilisé pour les connexions SMTP - envoi de votre courrier vers le serveur de votre FAI -.
- Port 53 Ce port est utilisé pour les requêtes DNS. Celles ci permettent, entre autre, de trouver l'adresse correspondant au site que vous cherchez à atteindre.
- Port 80 Ce port est utilisé pour les connexions HTTP, autrement dit à chaque fois que vous surfez.
- Port 110 Ce port est utilisé pour les connexions POP3 - téléchargement de votre courrier depuis le serveur de votre FAI -.
- Port 119 Ce port est utilisé pour les connexions NNTP. C'est-à-dire la lecture des forums Usenet.
- Port 443 Ce port est utilisé pour les connexions HTTPS, à savoir les connexions vers des sites web "sécurisés".

## 9.5 Une configuration standard

- Fermer tous les ports, en entrée comme en sortie, et en TCP comme en UDP ;
- Interdire tous les paquets ICMP à entrer ou sortir de l'ordinateur ;
- Autoriser les paquets ICMP "Echo Request" à sortir de l'ordinateur ;
- Autoriser les paquets ICMP "Echo Reply", "Destination Unreachable" et "Time Exceeded for a Datagram" à rentrer ;
- Ouvrir le port 53 dans les deux sens ;
- Autoriser le navigateur Internet à sortir vers les ports 80 et 443 ;
- Autoriser le lecteur de courrier à sortir vers les port 25 et 110 ;
- Autoriser le lecteur de news à sortir vers le port 119 ;
- Autoriser le lecteur de news à sortir vers le port 25 ;
- Autoriser le logiciel de FTP à sortir vers le port 21 ;
- Autoriser le logiciel de FTP à recevoir des données en provenance du port 20 ;

A noter que selon le firewall, les deux premières règles - celle interdisant toute connexion en TCP et UDP, et celle bloquant tous les paquets ICMP - peuvent être à placer après toutes les autres règles.

## Chapitre 10

# Introduction au protocole SSH

### Introduction

Le protocole SSH - *Secure SHell* - permet d'obtenir un shell sur une machine distante

#### 10.1 Définition

SSH est une application permettant d'accéder à une machine distante de façon sécurisée. Elle se place dans la continuité des applications telnet, RSH ou Rlogin qui bien que toujours utilisées par des applications - notamment telnet -, ne sont pas sécurisées. Par exemple avec telnet lors de la connexion distante, le mot de passe circule en clair sur le réseau. Donc, les mots de passe connexion distante et système n'étant pas décorélés il est possible de voir le mot de passe root en clair sur le réseau, avec tout ce que cela implique. SSH est donc une application mais le terme recouvre également le protocole associé. L'application la plus utilisée est *OpenSSH* qui s'appuie sur le protocole SSH1 ou SSH2.

#### 10.2 Quelle est l'utilité de SSH ?

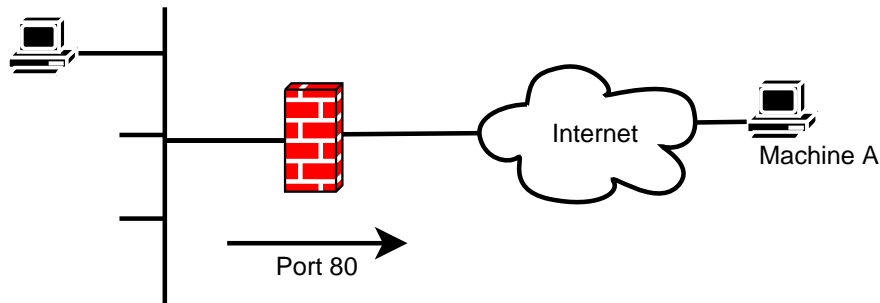
Cette application permet :

- permet d'obtenir un accès distant à une machine avec authentification, intégrité et confidentialité. Dans certains pays à ces qualités de base sont ajoutées aux trois fonctions de sécurité : la non répudiation<sup>1</sup> et l'auditabilité.
- permet de faire du *X11 forwarding* - donc d'avoir une interface graphique sur une machine distante -.
- permet de faire du transfert de fichiers sécurisé - SCP ou SFTP -. Avec FTP se pose le problème du routage et de la circulation du mot de passe en clair. SFTP simule du FTP et SCP fait du transfert fichier par fichier.
- permet d'encapsuler un autre protocole. Par exemple pour dialoguer en telnet, qui est généralement bloqué par les firewalls, il suffit d'utiliser SSH en sortie de machine pour encapsuler telnet, les firewalls laissent alors passer les paquets qui sont vus comme SSH.

---

<sup>1</sup>qui consiste à empêcher quelqu'un de dire « je ne l'ai pas fait » si c'est le cas

### 10.3 Limite des firewalls



Si le firewall ne laisse passer que les paquets à destination du port 80 en sortie il suffit d'envoyer les informations à la machine A sur le port 80, à l'aide du protocole SSH, mais en rajoutant dans la trame un autre protocole :

Donc pour filtrer au niveau du firewall il faut monter en couche applicative pour vérifier que le protocole utilisé avec le port 80 est bien http. C'est le rôle du proxy en complément du firewall. Cependant même un proxy présente des failles. En effet, le port https est le port 443, mais sur le port https les paquets sont chiffrés, donc le firewall ne peut pas lire les données de la couche applicative. Il devient donc possible de faire passer SSH malgré le proxy.

## 10.4 Les avantages du protocole SSH

### 10.4.1 Les avantages fonctionnels

Il y a deux avantages fonctionnels principaux :

- pouvoir travailler sur une machine distante comme si l'opérateur était devant.
- pouvoir facilement transférer des fichiers.

### 10.4.2 Les avantages techniques

- la décorrélation des mots de passe SSH et UNIX, ce qui permet d'éviter de corrompre le mot de passe UNIX.
- l'authentification forte par clé RSA. L'authentification consiste à prouver quelle est notre identité. L'authentification forte consiste à associer un secret - mot de passe - avec un objet possédé - la clé -. Pour être authentifié il faut fournir à la fois la clé et le mot de passe. Cette méthode permet d'obtenir des connexions multiples avec un seul mot de passe.

## 10.5 Fonctionnement d'une authentification par clé RSA

Tout le monde possède la clé publique. Comment savoir si un serveur SSH tourne sur une machine ? `netstat -anpe` donne la liste des ports ouverts. En utilisant `netstat -anpe | grep SSH` il est possible d'obtenir la liste des connexions SSH ouvertes. Pour se connecter sur une machine distante, à l'adresse @IP, en tant qu'utilisateur toto il faut utiliser :

```
ssh toto@IP
```

Il est possible de garantir son identité en prouvant la possession de la clé privée. Par exemple, une personne chiffre un défi avec la clé publique, seul le possesseur de la clé privée est capable de le lire en le déchiffrant avec cette clé privée. Application à une connexion :

1. Le serveur envoie, lors de la première connexion, sa clé publique.

2. La clé publique du serveur permet de déchiffrer la signature qu'il envoie en début de toute nouvelle connexion, donc de l'authentifier.
3. Le client envoie sa signature codée avec sa clé privée.
4. Le serveur vérifie alors la signature grâce à sa clé privée.
5. Si les deux vérifications sont valides, une clé de session est créée pour commencer les échanges de données. La clé de session est une clé symétrique de type DES.

## 10.6 Les fichiers importants

- fichier de configuration du serveur :  
`/etc/ssh/sshd_config`
- fichier contenant les clés publiques des utilisateurs voulant se connecter à la machine :  
`/.ssh/authorized_keys`  
 ou `/.ssh/identity.pub` sous fedora.
- fichier contenant les clés privées des utilisateurs voulant se connecter à d'autres machines :  
`/.ssh/id_rsa`  
 ou `/.ssh/id_rsa2` - utilisation du chiffage RSA2 -  
 ou encore `/.ssh/authorized_keys` sous fedora.

## 10.7 Réalisation pratique

### 10.7.1 Les demandes de passphrases

Il faut impérativement en donner une relativement complexe lors de la première connexion sur la machine distante. Elle représente le secret pour les connexions suivantes. La passphrase est le mot de passe qui permet de chiffrer la clé sur le disque, donc pas de passphrase implique pas de cryptage.

### 10.7.2 Création et copie des clés

Attention il y a un risque d'écraser les anciennes clés lors de la création de la nouvelle clé. La solution consiste donc à créer les clés dans un autre répertoire grâce à l'option `-f path`, path représentant le chemin du répertoire où seront créées les clés. Ensuite il suffit de recopier la clé dans le bon fichier en la concaténant au fichier déjà existant. Les permissions sur le répertoire contenant les clés - `/.ssh/` - doivent être 755. La génération des clés est simple :

```
ssh-keygen -t rsa -b 1024
```

Il est ensuite nécessaire de copier la clé publique sur le serveur - dans le répertoire de copie - :

```
scp /home/toto/.ssh/id_rsa.pub /toto@serveur :/home/toto/authorized_keys.cop
```

Ensuite, la clé est recopiée dans le bon répertoire du serveur :

```
cat /home/toto/.ssh/authorized_keys.cop » /home/toto/.ssh/authorized_keys
```

Pour préciser l'emplacement de la clé privée il faut utiliser :

```
ssh toto@192.168.101.47 -i /home/.ssh/id_rsa
```

Il ne reste plus qu'à se connecter depuis le client :

```
ssh toto@serveur
```

Il reste à indiquer la passphrase pour l'utilisateur toto.

### 10.7.3 Debugger le serveur

Pour lancer le serveur ssh en mode debug il faut tout d'abord stopper le serveur avant de le relancer en mode debug :

```
/etc/init.d/ssh stop  
/usr/sbin/sshd -d
```

#### 10.7.4 Durcir le fichier de configuration

Le fichier de configuration du serveur est `/etc/ssh/sshd_config`.

Il faut désactiver certaines options :

**PermitRootLogin : no** interdit de se connecter directement en tant que root.

**UsePAMAuthentication : no** l'identification n'est pas déléguée à PAM

**PasswordAuthentication : no** interdit l'authentification par mot de passe

# Annexe A

## Exercices

### A.1 La couche 3

#### A.1.1 Exercice 1

Trouvez la première et la dernière adresse disponibles pour le réseau associé à l'adresse suivante : 127.24.99.132 masque 255.255.255.224

1. Exploitation du masque :  $224 = 11100000$ . Donc le masque est :  
11111111.11111111.11111111.11100000
2. Définition des bits réseau : une machine de ce réseau est 127.24.99.132. Les machines sont codées uniquement dans le dernier bit -cf masque-.  $132 = 10000100$  donc les bits réseau sont : 127.24.99.100xxxxx.
3. La première adresse vaut donc 127.24.99.128 -bits machines à 0-, et la dernière est 124.24.99.159 -bits machine à 1-.
4. Le nombre d'adresses de ce réseau est donc de  $2^5$ , les machines étant codées sur cinq bits.

#### A.1.2 Exercice 2

Trouvez la première et la dernière adresse disponibles pour le réseau associé à l'adresse suivante : 192.168.84.27 masque 255.255.192.0

1. Exploitation du masque :  $192 = 11000000$ .
2. Définition des bits réseau : Les machines sont codées uniquement dans les troisième et quatrième bits -cf masque-.  $84 = 01010100$  donc les bits réseau sont : 127.24.01xxxxxx.xxxxxxxx.
3. La première adresse vaut donc 192.168.64.0 -bits machines à 0-, et la dernière est 192.168.127.255 -bits machine à 1-.
4. Le nombre d'adresses de ce réseau est donc de  $2^{14}$ .

#### A.1.3 Exercice 3

Un fournisseur d'accès internet fournit 256 adresses à une école : plage 194.2.232.0 masque 255.255.255.0. Celle ci comprend 100 élèves, 20 professeurs et 5 administrateurs. Je veux découper la plage en 3 réseaux différents. Donner la première et la dernière adresse de chaque sous réseau

1. Définition des masques :
  - Pour les élèves il faut 100 adresses, ce qui nécessite sept bits pour les machines -  $2^6 \leq 100 \leq 2^7$ -, le masque vaut donc 255.255.255.10000000.

- Pour les professeurs il faut vingt adresses, ce qui nécessite cinq bits pour les machines - $2^4 \leq 20 \leq 2^5$ -, le masque vaut donc 255.255.255.11100000.
  - Pour les administratifs il faut cinq adresses ce qui nécessite trois bits pour les machines. Cependant si on se limite à trois les possibilités d'extension du réseau seront limitées, c'est pourquoi il est plus opportun de coder les machines sur quatre bits dans ce cas. Le masque vaut donc 255.255.255.11110000.
2. Répartition des bits de réseau pour les sous-réseaux. Seul le dernier octet est indiqué les trois autres étant fixés à 194.2.232.
    - élèves : 1xxxxxxx
    - professeurs : 000xxxxx
    - administratifs : 0100xxxx
  3. Les premières et dernières adresses de chaque sous-réseau sont donc :
    - pour les élèves : 194.2.232.128 à 192.2.232.255
    - pour les professeurs : 194.2.232.0 à 194.2.232.31
    - pour les administratifs : 194.2.232.64 à 194.2.232.79
  4. Il est facile de vérifier que les différentes plages d'adresse ne se recoupent pas.

#### A.1.4 Exercice 4

**Un fournisseur d'accès internet fournit à une école : plage 10.24.192.0 masque 255.255.240.0. Celle ci comprend 1200 élèves, 150 professeurs et 10 administratifs. Je veux découper la plage en 3 réseaux différents. Donner la première et la dernière adresse de chaque sous réseau**

1. Définition des masques :  $240 = 11110000$   $192 = 11000000$  On dispose donc de  $2^{12}$  adresses
  - Pour les élèves il faut 1500 adresses, ce qui nécessite onze bits pour les machines - $2^{10} \leq 1500 \leq 2^{11}$ -, le masque vaut donc 255.255.11111000.0 soit 255.255.248.0.
  - Pour les professeurs il faut cent cinquante adresses, ce qui nécessite huit bits pour les machines - $2^7 \leq 150 \leq 2^8$ -, le masque vaut donc 255.255.255.0.
  - Pour les administratifs il faut dix adresses ce qui nécessite quatre bits pour les machines. Le masque vaut donc 255.255.255.240.
2. Répartition des bits de réseau pour les sous-réseaux. Seul les troisième et quatrième octets sont indiqués :
  - élèves : 11000xxx xxxxxxxx
  - professeurs : 11001000.xxxxxxxx
  - administratifs : 11001001.0000xxxx
3. Les premières et dernières adresses de chaque sous-réseau sont donc :
  - pour les élèves : 10.24.192.0 à 10.24.199.255
  - pour les professeurs : 10.24.200.0 à 10.24.200.255
  - pour les administratifs : 10.24.201.0 à 10.24.201.15
4. Il est facile de vérifier que les différentes plages d'adresse ne se recoupent pas.

#### A.1.5 Exercice 5

**Même exercice que le numéro 4 mais en plaçant d'abord les petites plages**

1. Définition des masques : inchangé.
2. Répartition des bits de réseau pour les sous-réseaux. Seul les troisième et quatrième octets sont indiqués :
  - élèves : 11001xxx xxxxxxxx



- professeurs : 11000100.xxxxxxxx
  - administratifs : 11000000.0000xxxx
3. Les premières et dernières adresses de chaque sous-réseau sont donc :
- pour les élèves : 10.24.200.0 à 10.24.207.255
  - pour les professeurs : 10.24.196.0 à 10.24.196.255
  - pour les administratifs : 10.24.192.0 à 10.24.192.15
4. Il est facile de vérifier que les différentes plages d'adresse ne se recoupent pas.

### A.1.6 Exercice 6

Étant donné le réseau 10.0.0.0 masque 255.255.255.0, les adresses machines de 1 à 63 étant déjà utilisées, choisir une plage de 110 adresses.

- Pour coder 110 adresses il faut 7 bits - $2^6 \leq 110 \leq 2^7$ -.
- Les adresses déjà utilisées correspondent au sous-réseau 10.0.0.0 masque 255.255.255.192 avec les bits de réseau à 10.0.0.00xxxxxx
- Le masque de notre nouveau sous réseau est donc 255.255.255.128 avec les bits de réseau à 10.0.0.1xxxxxxx
- La nouvelle plage d'adresses s'étend donc de 10.0.0.128 à 10.0.0.255

### A.1.7 Exercice 7

Je dispose du réseau 193.168.47.0 masque 255.255.255.0 et je souhaite ajouter 250 adresses à ce réseau quel réseau choisir ? Il me faut donc 505 adresses au total -les 255 initiales et les 250 nouvelles-. Or  $2^8 \leq 505 \leq 2^9$  donc le nouveau masque sera 255.255.254.0. La première adresse de mon nouveau sous-réseau sera 193.168.46.0 et la dernière 193.168.46.255. La première adresse de mon réseau total sera donc 193.168.46.0 et la dernière 193.168.47.255.

## A.2 La couche 4

### A.2.1 Exercice 1

Énoncé

2	@IP dest	@IP src	Port src X	Port dest 80	
---	----------	---------	---------------	-----------------	--

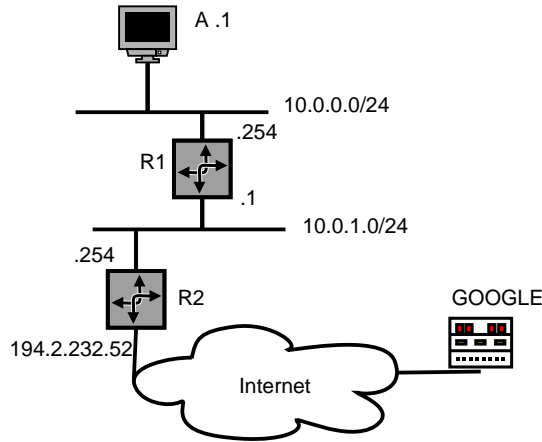
Le champs @IP dest vaut 255.255.255.255. Est ce normal ?

Corrigé

Nous sommes donc, à priori, en présence d'un broadcast. Ce pourrait être une recherche des serveurs sur le réseau, mais il risque alors d'y avoir plusieurs machines qui répondent en SYN + ACK ce que je ne sais pas traiter. Donc TCP n'a pas de broadcast, car ça n'a pas de sens : TCP établit une connexion avec une seule autre machine par session. Donc si il est nécessaire de faire un broadcast il faut utiliser UDP

**A.2.2 Exercice 2**

**Énoncé**



1. Donner la table de routage de R1
2. Ecrire la trame, provenant d'une requête web vers Google, de A en sortie de R1
3. Même question en sortie de R2
4. Quel est le premier paquet qui sort de A lors de cette connexion ?
5. Donner toutes les étapes de la communication entre A et Google.

**Corrigé**

Réseau	masque	Passerelle
10.0.0.0	/24	0.0.0.0
10.0.1.0	/24	0.0.0.0
0.0.0.0	/24	10.0.1.254

TAB. A.1 – Table de routage de R1

**question 1** Rappel : Les adresses 10.0.0.0/8 ne sont pas routées sur internet, elles sont réservées aux réseaux locaux donc elles ne doivent pas être routées sur internet.

@MAC R2	@MAC R1	IP	Long tot 3	TCP	@IP A	@IP Google	Port $X_A$	Port 80	Flg	Urg Ptr	DATA	CRC
------------	------------	----	---------------	-----	----------	---------------	---------------	------------	-----	------------	------	-----

**question 2**

@MAC R3	@MAC R2	IP	Long tot 3	TCP	@IP R2	@IP Google	Port $X_{R2}$	Port 80	Flg	Urg Ptr	DATA	CRC
------------	------------	----	---------------	-----	-----------	---------------	------------------	------------	-----	------------	------	-----

**question 3** Les adresses 10.0.0.0/8 n'étant pas routées sur internet il est nécessaire d'avoir recours à la NAT.

**question 4** Il peut y avoir deux situations :

- Si la table ARP de A est renseignée avec l'adresse MAC de R1 la première trame qui circulera la demande de synchronisation - SYN -.
- Sinon la première trame sera la requête ARP.

**question 5** L'application envoie une requête en donnant : les DATAS, le protocole de couche 4 -TCP-, la priorité...

**couche 4** Le port destination est connu -80-, le système d'exploitation fournit un port source libre puis forme le segment TCP. Pour obtenir l'adresse IP de Google, il fait appel au resolver pour la résolution de nom et envoie le segment et les adresses IP à la couche 3.

**couche 3** Elle va former le datagramme puis le segmenter si nécessaire et l'envoyer à la couche 2.

**couche 2** Pour former la trame Ethernet il faut connaître l'adresse MAC destination. Il faut donc aller voir dans la table de routage pour obtenir l'adresse IP de la passerelle, ou celle de la machine de destination, si elle est dans le réseau local. Ici il faut rechercher la passerelle. Avec cette adresse IP on consulte la table ARP, soit elle est renseignée et on obtient l'adresse MAC soit il faut la renseigner auquel cas une requête ARP est envoyée sur le réseau pour obtenir l'adresse MAC recherchée. Ensuite il est possible de former la trame Ethernet en calculant au passage le CRC de couche 2. La trame est ensuite envoyée sur le réseau.

**couche 1/2** Le paquet arrive au switch qui aiguille le paquet vers le routeur après avoir lues informations de couche 2.

**Le(s) routeur(s)** Il lit la couche 2, l'adresse MAC correspond à la sienne donc il poursuit la lecture en vérifiant le CRC puis envoie le résultat à la couche 3. Celle-ci lit les informations de couche 3. Consulte éventuellement - pour le dernier routeur du réseau local- la table NAT, puis consulte la table de routage pour déterminer la prochaine étape - passerelle ou destination - au vu de l'adresse IP destination et grâce au même processus que A elle reconstitue la trame de niveau 3, détermine l'adresse MAC destination comme précédemment -requête ARP-. La trame est ensuite transmise à la couche 2 qui forme la trame Ethernet. Puis celle-ci est renvoyée sur le réseau.

**Google** L'adresse MAC correspond à la sienne, il lit donc les informations de couche 3 en recalculant le CRC. L'adresse IP correspond bien à celle référencée pour 127.0.0.1 dans sa table de routage donc il garde la trame et envoie les DATAS à la couche 4. Le port correspond bien à l'application donc les informations sont exploitées par l'application.

**Remarque** Sur le réseau initial il y a en plus un serveur DNS de résolution de nom et éventuellement un proxy. La page google n'est pas envoyée immédiatement : Le premier message de A comporte le SYN, la réponse de google le SYN + ACK et enfin le ACK de A. Google ne commence à envoyer la page qu'à partir de la seconde trame.

## A.3 La NAT

### A.3.1 Exercice 1

#### Énoncé

Une entreprise TOTO s'adresse à COLT pour obtenir des adresses IP. En interne il y a :

- 600 collaborateurs
- 4 serveurs web
- 5 serveurs FTP
- 5 serveurs mail
- 2 serveurs SSH

Combien d'adresses web faut-il demander ?

#### Réponse

Je ne peux placer que un type de serveur par adresse, car avec la NAT je dois associer au niveau du routeur desservant cette adresse l'adresse du serveur sur le réseau local avec le numéro de port destination affecté à ce type d'application. Il faut donc demander cinq adresses web.

#### Complément

Si COLT me donne 194.2.232.0/24, je demande donc les adresses allant de 194.2.232.0/29 à 194.2.232.7/29. Je dispose donc de huit adresses au total, dont celle de réseau et celle de broadcast. Il reste donc six adresses disponibles.

- 194.2.232.6 réseau collaborateur
- 194.2.232.1 : WEB 1, FTP 1, Mail 1, SSH 1
- 194.2.232.2 : WEB 2, FTP 2, Mail 2, SSH 2
- 194.2.232.3 : WEB 3, FTP 3, Mail 3
- 194.2.232.4 : WEB 4, FTP 4, Mail 4
- 194.2.232.5 : FTP 5, Mail 5

Cette répartition est acceptable, mais je peux garder une adresse en réserve si je regroupe les collaborateurs sur une autre adresse. En règle générale, il est préférable de demander un masque sur 28 bits pour avoir 16 adresses et se garder des possibilités d'évolution.

# Annexe B

## La transmission en bande de base

### B.1 Introduction

La transmission numérique consiste à faire transiter les informations sur le support physique de communication sous forme de signaux numériques. Ainsi, des données analogiques devront préalablement être numérisées avant d'être transmises.

Toutefois, les informations numériques ne peuvent pas circuler sous forme de 0 et de 1 directement, il s'agit donc de les coder sous forme d'un signal possédant deux états, par exemple :

- deux niveaux de tension par rapport à la masse
- la différence de tension entre deux fils
- la présence/absence de courant dans un fil
- la présence/absence de lumière
- ...

Cette transformation de l'information binaire sous forme d'un signal à deux états est réalisée par l'ETCD, appelé aussi codeur bande de base, d'où l'appellation de transmission en bande de base pour désigner la transmission numérique...

### B.2 Codage des signaux

Pour que la transmission soit optimale, il est nécessaire que le signal soit codé de façon à faciliter sa transmission sur le support physique. Il existe pour cela différents systèmes de codage pouvant se classer en deux catégories :

- Le codage à deux niveaux : le signal peut prendre uniquement une valeur strictement négative ou strictement positive ( $-X$  ou  $+X$ ,  $X$  représentant une valeur de la grandeur physique permettant de transporter le signal)
- Le codage à trois niveaux : le signal peut prendre une valeur strictement négative, nulle ou strictement positive ( $-X$ ,  $0$  ou  $+X$ )

### B.3 Le codage NRZ

Le codage NRZ (signifiant No Return to Zero, soit Non Retour à Zéro) est le premier système de codage, car le plus simple. Il consiste tout simplement à transformer les 0 en  $-X$  et les 1 en  $+X$ , de cette façon on a un codage bipolaire dans lequel le signal n'est jamais nul. Par conséquent, le récepteur peut déterminer la présence ou non d'un signal.

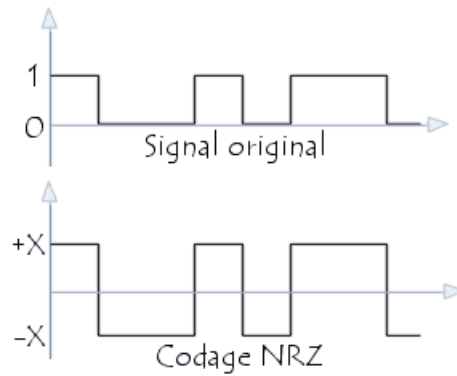


FIG. B.1 – Codage NRZ

## B.4 Le codage NRZI

Le codage NRZI est sensiblement différent du codage NRZ. Avec ce codage, lorsque le bit est à 1, le signal change d'état après le top d'horloge. Lorsque le bit est à 0, le signal ne subit aucun changement d'état. Le codage NRZI possède de nombreux avantages, dont :

- la détection de la présence ou non du signal
- la nécessité d'un faible courant de transmission du signal

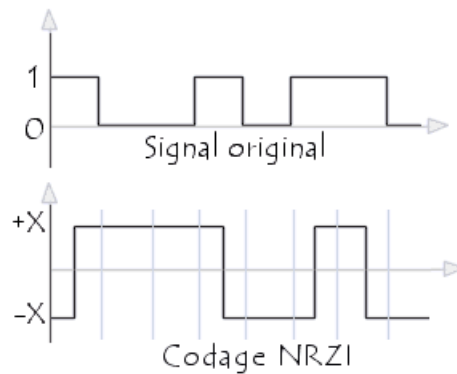


FIG. B.2 – Codage NRZI

En revanche, il possède un défaut : la présence d'un courant continu lors d'une suite de zéro, gênant la synchronisation entre émetteur et récepteur.

## B.5 Le codage Manchester

Le codage Manchester, également appelé codage biphasé ou PE (pour Phase Encode), introduit une transition au milieu de chaque intervalle. Il consiste en fait à faire un OU exclusif (XOR) entre le signal et le signal d'horloge, ce qui se traduit par un front montant lorsque le bit est à zéro, un front descendant dans le cas contraire. Le codage Manchester possède de nombreux avantages, dont :

- le non passage par zéro, rendant possible par le récepteur la détection d'un signal
- un spectre occupant une large bande

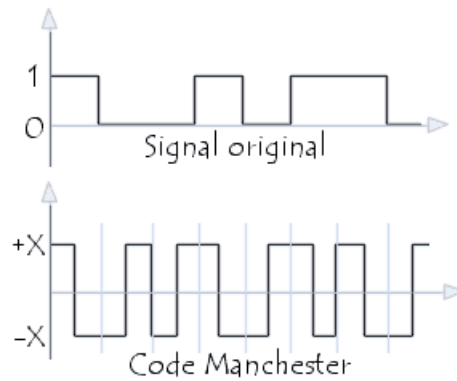


FIG. B.3 – Codage Manchester

## B.6 Le codage delay

Le codage Delay Mode, aussi appelé code de Miller, est proche du codage de Manchester, à la différence près qu'une transition apparaît au milieu de l'intervalle uniquement lorsque le bit est à 1, cela permet de plus grands débits.

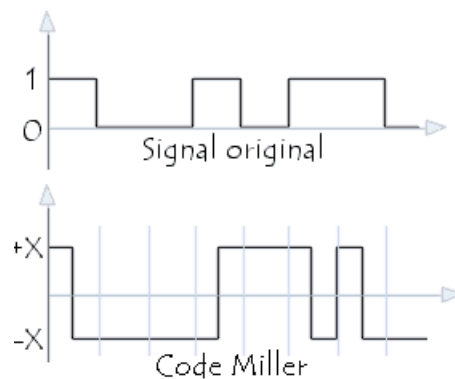


FIG. B.4 – Codage Miller

## B.7 Le codage bipolaire

Le codage bipolaire simple est un codage sur trois niveaux. Il propose donc trois états de la grandeur transportée sur le support physique :

- La valeur 0 lorsque le bit est à 0
- Alternativement X et -X lorsque le bit est à 1

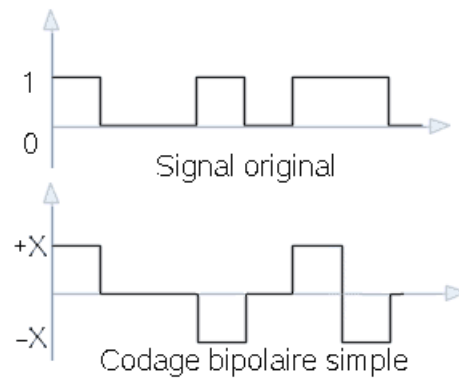


FIG. B.5 – Codage bipolaire



## Annexe C

# Fibres optiques

### C.1 Principe de fonctionnement

Le principe de fonctionnement des fibres optiques repose sur les relations de Fresnel. L'indice de réfraction de la fibre varie le long du rayon de la fibre. Considérons le passage d'un milieu d'indice  $\lambda_1$  à un milieu d'indice  $\lambda_2$ . Si l'onde lumineuse incidente arrive sur la surface de contact avec un angle  $\theta_1$  par rapport à la normale à la surface alors la relation de Fresnel indique que l'onde réfractée présentera un angle  $\theta_2$  à la normale.

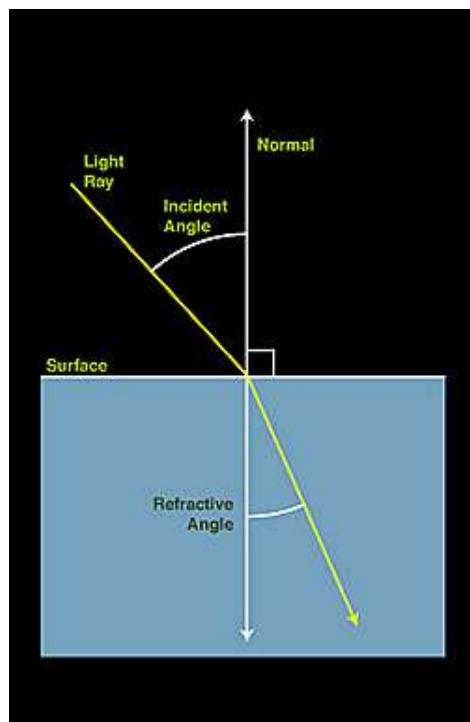


FIG. C.1 – La relation de Fresnel

La valeur  $\theta_2$  est donnée par  $\lambda_1 * \sin \theta_1 = \lambda_2 * \sin \theta_2$ . Le résultat visé étant que l'onde lumineuse reste dans l'épaisseur de la fibre, la propriété utilisée va être différente en fonction du type de fibre. Pour une fibre à saut d'indice, c'est la réflexion totale qui va être recherchée lors du changement de milieu. En revanche pour une fibre à variation d'indice c'est la réfraction qui va être utilisée pour que l'onde se rapproche de l'axe de la fibre jusqu'à atteindre l'angle de réflexion totale. Donc, il faut que l'angle  $\theta$  augmente donc que l'indice diminue.

## C.2 Les trois types de fibre optique

Les fibres optiques sont divisée en deux grandes familles : les fibres multimodes et les monomodes. Au sein des multimodes existent les fibres à saut d'indice et celles à gradient d'indice. Les fibres monomodes sont elles essentiellement à saut d'indice.

**La fibre à saut d'indice 200/380** constituée d'un coeur et d'une gaine optique en verre de différents indices de réfraction. Cette fibre provoque de par l'importante section du coeur, une grande dispersion des signaux la traversant, ce qui génère une déformation du signal reçu.

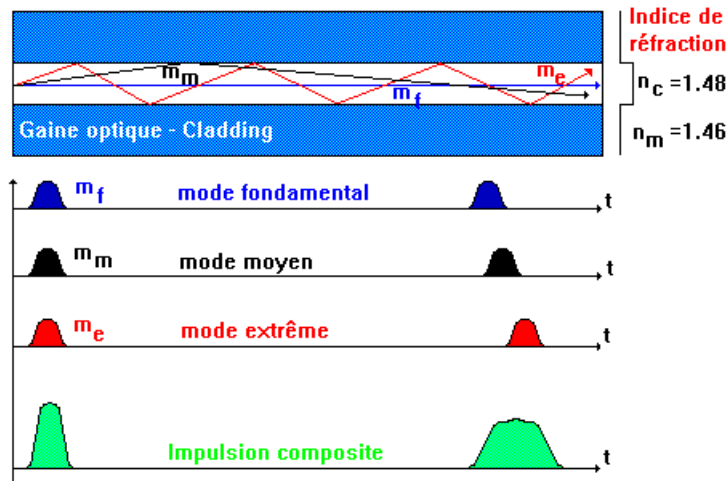


FIG. C.2 – Propagation de la lumière dans une fibre à saut d'indice

**La fibre à gradient d'indice** dont le coeur est constitué de couches de verre successives ayant un indice de réfraction proche. On s'approche ainsi d'une égalisation des temps de propagation, ce qui veut dire que l'on a réduit la dispersion nodale. Bande passante typique 200-1500Mhz par km. C'est ce type de fibre qui est utilisé entre certains sites desservis par les PTT (50/125).

**La fibre monomode** dont le coeur est si fin que le chemin de propagation des différents modes est pratiquement direct. La dispersion nodale devient quasiment nulle. La bande passante transmise est presque infinie ( $> 10\text{GHz/km}$ ). Cette fibre est utilisée essentiellement pour les transmissions à grande distance.

Le petit diamètre du coeur (10 $\mu\text{m}$ ) nécessite une grande puissance d'émission, donc des diodes au laser, qui sont relativement onéreuses.

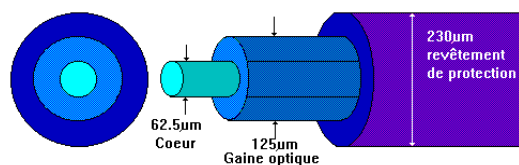


FIG. C.3 – Constitution d'une fibre optique multimode

Le schéma ci dessous retrace pour les trois types de fibre la propagation de l'onde lumineuse en fonction de leur angle d'émission. Il illustre le principe de fonctionnement décrit en début de chapitre.

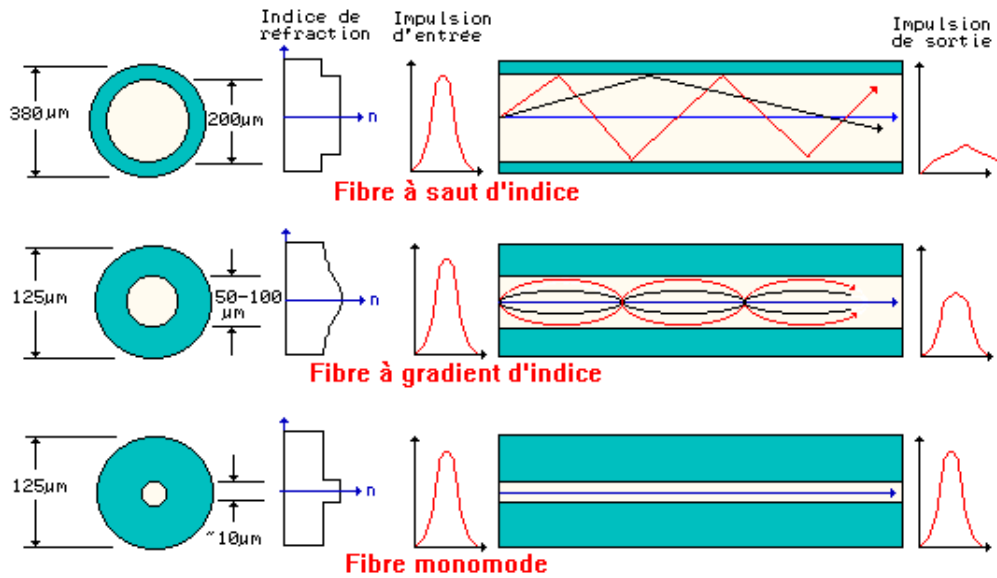


FIG. C.4 – Propagation de la lumière dans les trois types de fibres

Le chemin parcouru n'a pas la même longueur pour tous les rayons. C'est ce que l'on appelle la dispersion nodale.

L'atténuation est constante quelle que soit la fréquence. Seule la dispersion lumineuse limite la largeur de la bande passante.

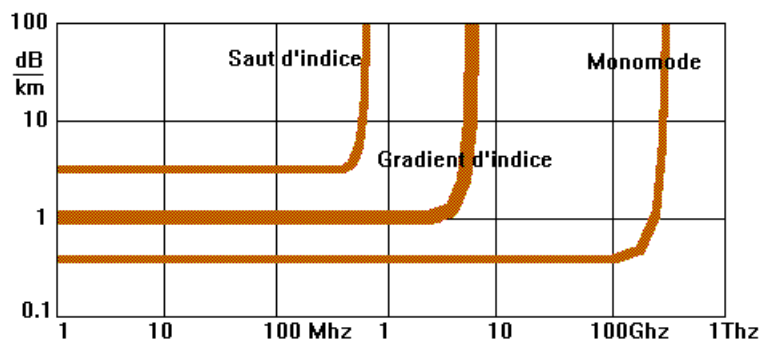


FIG. C.5 – Atténuation dans une fibre optique

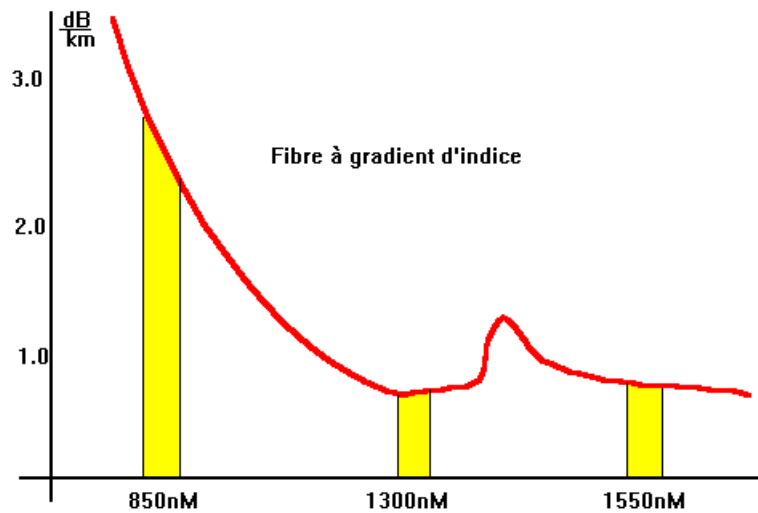


FIG. C.6 – Affaiblissement de la lumière en fonction de la longueur d'onde

L'affaiblissement de la lumière dans la fibre est fonction de la longueur d'onde de la source. Elle est constante pour toutes les fréquences du signal utile transmis. Le dessin ci-dessus montre que l'affaiblissement est plus important dans le rouge (850nm) que dans l'infrarouge (1300-1550nm).

### C.3 Les connections

Il existe nombre de connecteurs pour la fibre optique. Les plus répandus sont les connecteurs **ST** et **SC**. Pour les réseaux FDDI, on utilise les connecteurs doubles **MIC**.

Il faut encore citer les connecteurs **SMA** -à visser- et les connecteurs **FCPC** utilisés pour la fibre monomode.



FIG. C.7 – Connecteur ST



FIG. C.8 – Connecteur SC



FIG. C.9 – Connecteur FDDI ou MIC

Il y a plusieurs manières pour coupler de la fibre optique :

- Le couplage mécanique de deux connecteurs mis bout à bout au moyen d'une pièce de précision. Le dessin ci-dessous montre l'union de deux connecteurs ST, mais il existe des coupleurs ST/SC ou ST/MIC.
- Le raccordement par *Splice* mécanique qui est utilisé pour les réparations à la suite de rupture ou pour raccorder une fibre et un connecteur déjà équipé de quelques centimètres de fibre que l'on peut acquérir dans le commerce -*Pig tail*-.
- La fusion au moyen d'un appareil à arc électrique appelé fusionneuse.

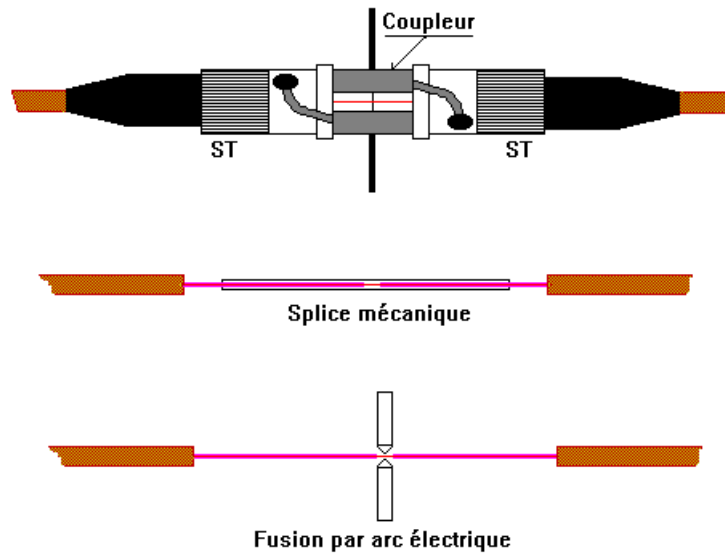


FIG. C.10 – Trois exemples de connexions en fibre optique



# Bibliographie

- [1] Stéphane CATTEAU. Faq sur la nat.
- [2] Stéphane CATTEAU. Faq sur les firewalls.
- [3] IP-FRAME. <http://www.frameip.com>.
- [4] Comment Ca Marche. <http://www.commentcamarche.net/internet/internet.php3>.