0 — introduction

Nicolas Ollinger, *Université d'Orléans*

M2 SIR Sécurité des réseaux — **S4** 2017/2018

Organisation

7x séances thématiques de 4h CM/TP
dont 24h avec le prof

Généralement le vendredi,
en salles E09 (netkit) + E11 (Cisco).

http://tinyurl.com/srsir

Évaluation

$$\textbf{Note} = (2\textbf{CT} + \textbf{CC})\ /3$$

CT examen terminal de 2h sur table

Seconde session CT de 2h sur table

Documents autorisés : notes de séances.

Objectifs du cours

Acquérir une première culture technique sur les équipements (*middleboxes*) et technologies réseau assurant des fonctions de sécurité.

Énoncer quelques bonnes pratiques lors de la mise en œuvre de ces technologies.

Visualiser clairement les atouts et les limites de ces technologies.

Pourquoi sécuriser un réseau IP ?

Interconnexion des réseaux d'entreprise aux réseaux IP privés (*intranet*) et publics (*internet*).

Compromission très rapide des serveurs exposés à l'internet sans protection (quelques heures à quelques semaines).

L'utilisation des réseaux IP et l'interconnexion à l'internet pour des usages sensibles (e-commerce, informations bancaires, *etc*) implique une sécurité optimale en termes :

- de sécurité d'accès (filtrage) ;
- de sécurité des données (chiffrement) ;
- de sécurité de fonctionnement (haute disponibilité).

Déroulement des séances

0. Introduction + rappels.

1. Réseaux locaux virtuels (VLAN) et problématiques de sécurité associées.

2. Filtrage des communications IP sur switches, routeurs, firewalls (**pf**) et serveurs.

3. Réseaux privés virtuels (VPN) et chiffrement des communications.

Déroulement des séances

4. Attaques, malveillance et intrusion par l'exemple.

5. Prévention des intrusions réseau, dispositifs NIDS-NIPS (**snort**).

6. Sécurité de fonctionnement et haute disponibilité, load-balancing.

Organisation pratique

En salle de TP classique (E09) : **Netkit**, simulation de postes Debian GNU/Linux.

En salle dédiée réseaux (E11) :

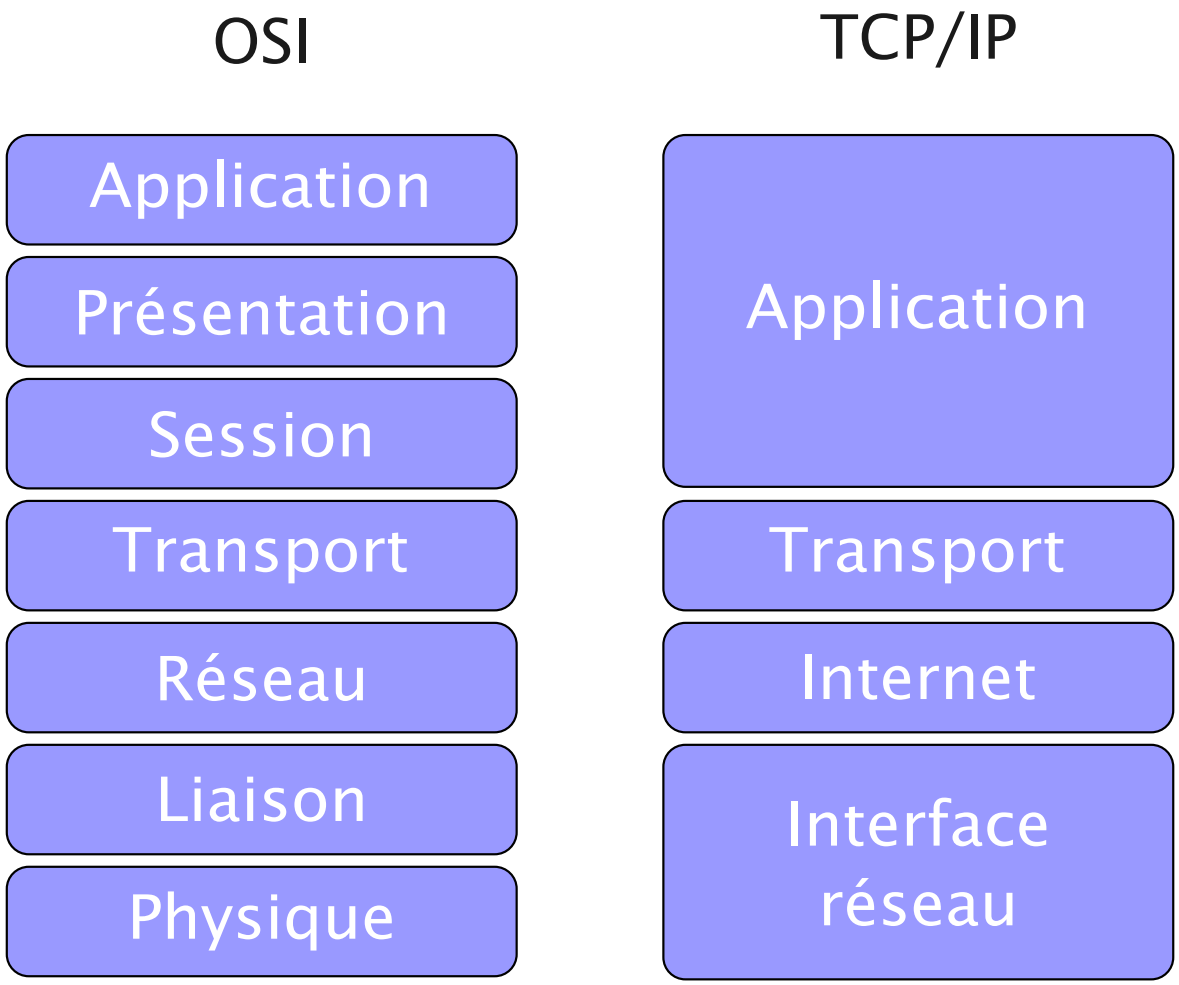
- appliances CISCO (switches Catalyst, routeurs, ASA) ;
- routeurs OpenBSD Soekris ;
- postes en dual boot Linux/OpenBSD.

Les supports sont sur Celene :
http://tinyurl.com/srsir

Un wiki collaboratif à développer sur Confluences :
https://orleans.miage.fr/confluence/display/REZO

Rappels

Modèles OSI et TCP/IP



Couche Liaison

Data Link Layer

Transmission de **trames** d'un point A à un point B reliés par un medium :

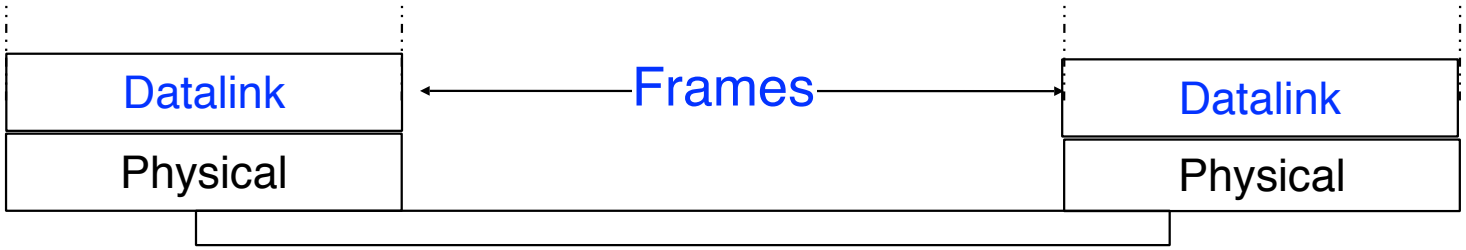
- sans erreurs ;
- sans perte de messages ;
- sans duplication.

Toujours **point-à-point** ou **réseau de diffusion** avec gestion d'**accès au canal**.

Ethernet

Couche Liaison

Data Link Layer



Couche Réseau

Network Layer

Transmission de données d'un point A à un point B en utilisant des points **intermédiaires** comme relais.

Notion d'**adresse**, comprise par tous les points intermédiaires.

Notion de **routage**, indépendant ou non des données.

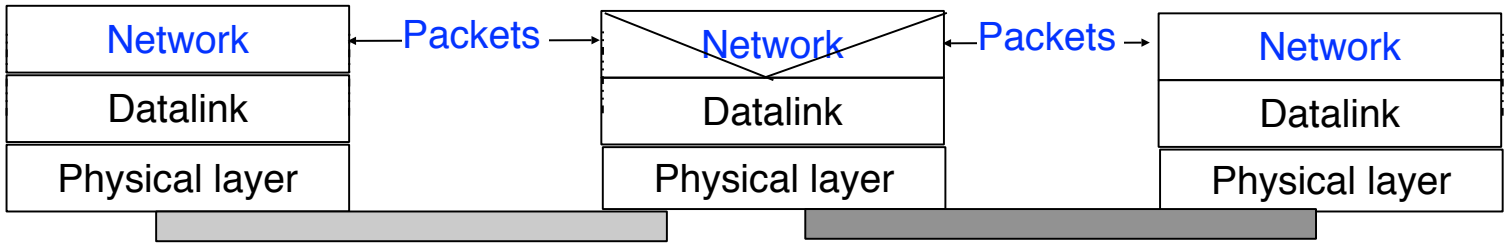
Le message peut être perdu.

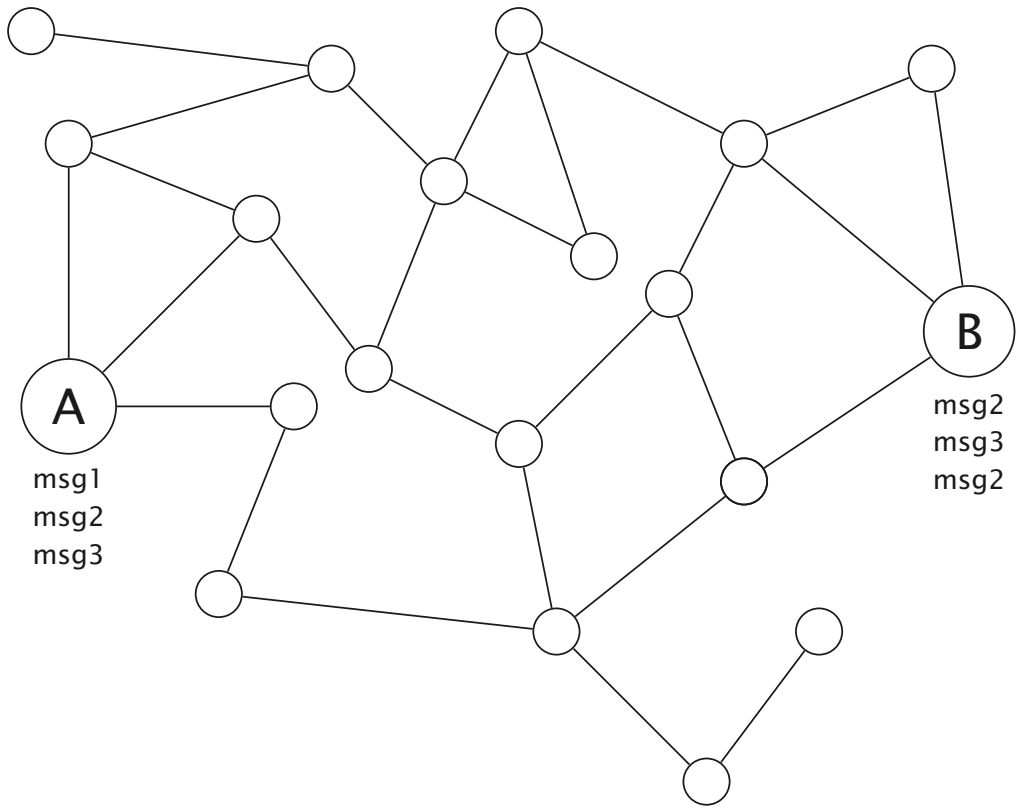
Les morceaux d'un message peuvent arriver dans le désordre.

IP

Couche Réseau

Network Layer





Couche Transport

Transport Layer

Transmission **fiable** de données d'un point A à un point B en utilisant des points intermédiaires comme relais.

Notion de **port** pour distinguer les applications sur un même hôte.

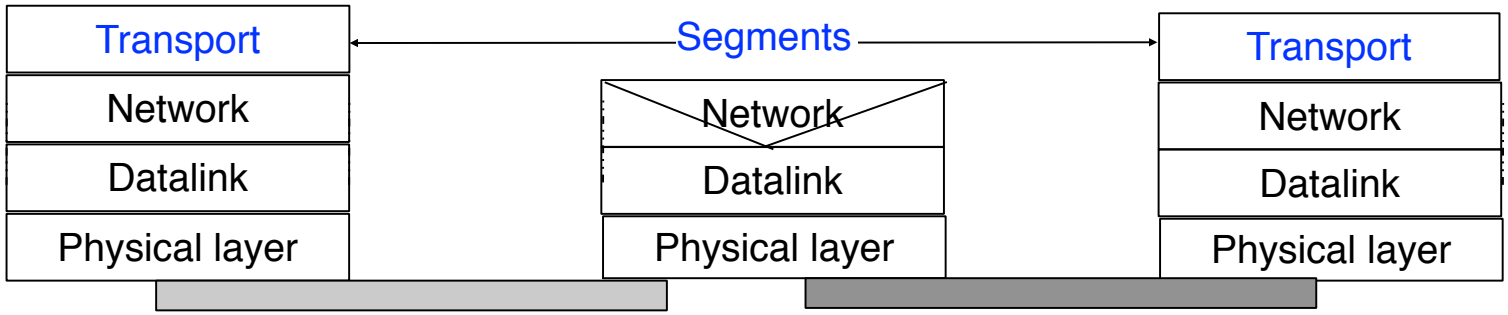
Transmission de **datagrammes**, en mode non-connecté.

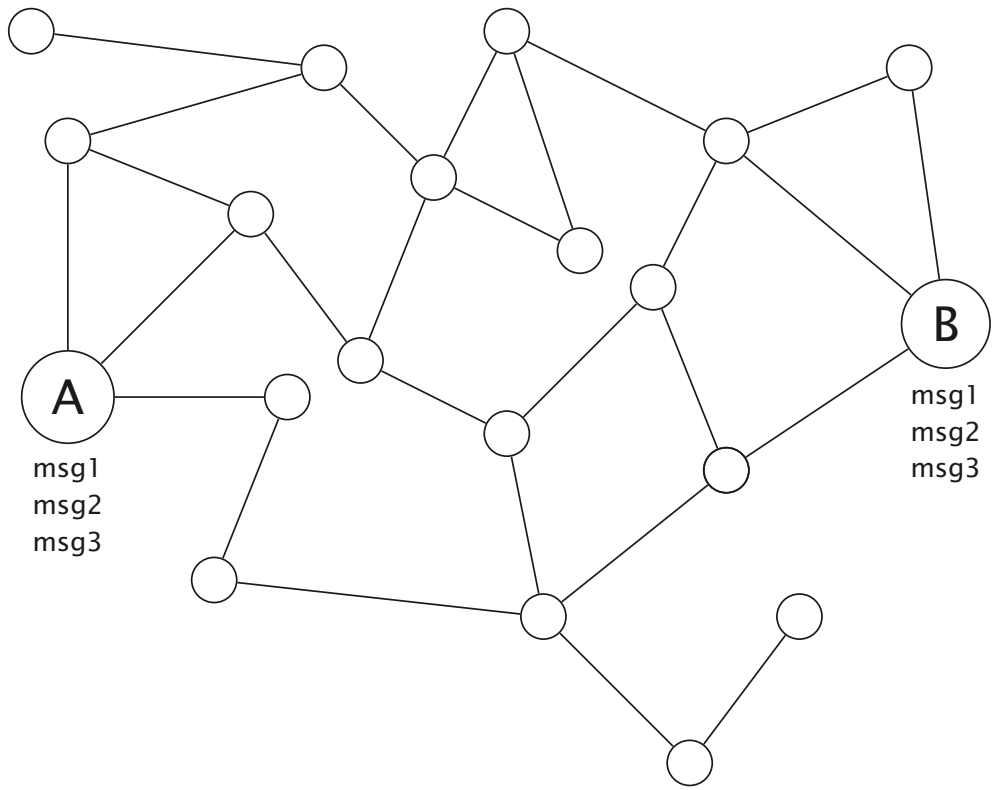
Transmission de **flux de données**, en mode connecté.

TCP/UDP

Couche Transport

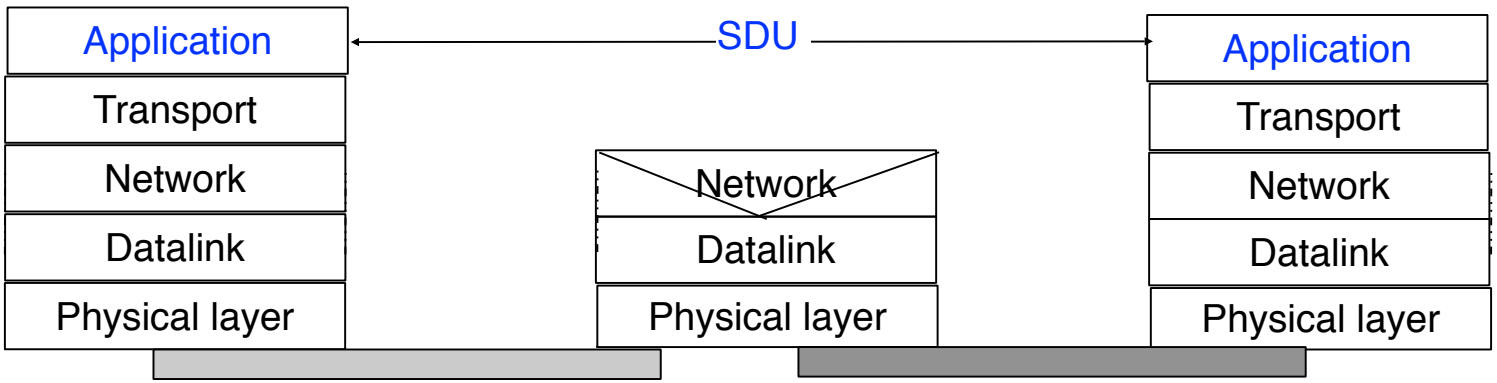
Transport Layer





Couche Application

Application Layer



Protocoles applicatifs, aussi nombreux que variés.

SMTP, HTTP, FTP, XMPP, ...

Interface Ethernet

```
$ ifconfig eth0

eth0  Link encap:Ethernet  HWaddr 00:11:D8:8D:51:F9
      [snip]
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:94686301 errors:0 dropped:0 overruns:0 frame:0
      TX packets:82880491 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:400469386 (381.9 MiB)  TX bytes:3207666956 (2.9 GiB)
```

Adresse

- Adresse sur 2^{48} bits
- Dans l'exemple `00:11:D8:8D:51:F9`
- Mais seulement 2^{46} adresses potentielles
  - Un bit pour adresses globales/locales
  - Autre bit pour le multicast

Certaines adresses ont des significations précises :

- `FF:FF:FF:FF:FF:FF` Broadcast (destiné à toutes les machines)

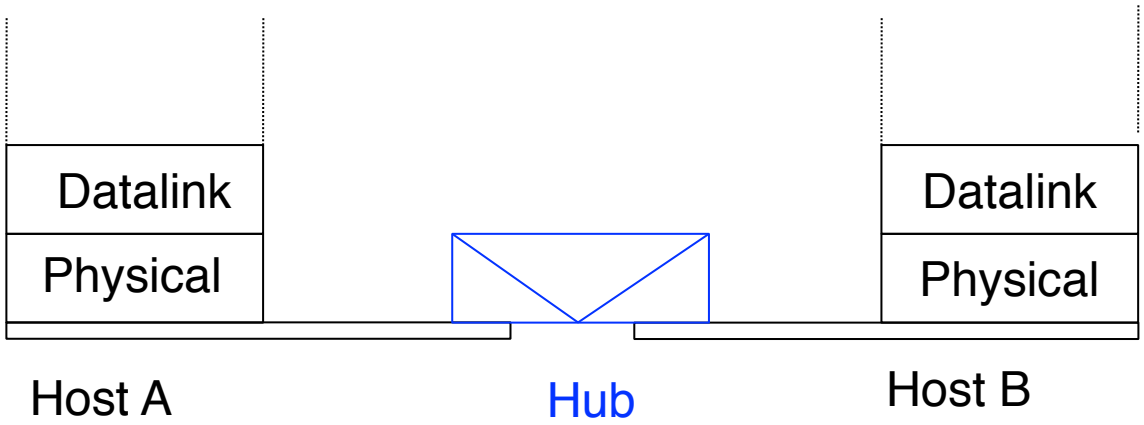
Trame

IEEE 802.3

Flag	Destination	Source	Type	Données	CRC
8	6	6	2	46-1500	4

- Une trame Ethernet transporte au maximum 1500 octets (Maximum Transfer Unit (MTU))
- Au minimum 46 caractères
- Taille max de 1518 (sans compter le Flag)
- CRC 0x04C11DB7

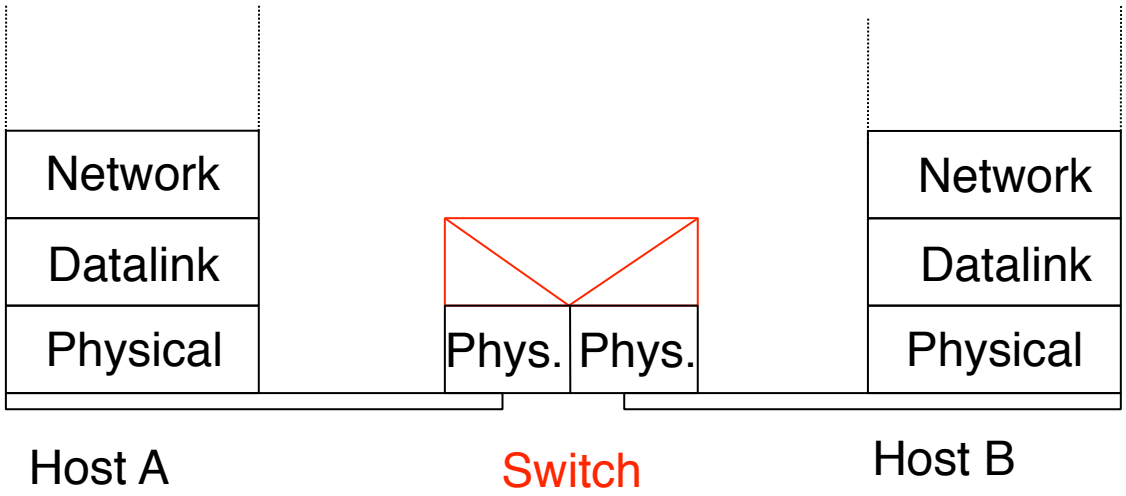
Hub



- Un hub se contente de transmettre un message reçu sur un port vers chacun des autres ports (souvent en l'amplifiant)
- Si une machine envoie un message, toutes les autres le reçoivent.
- Si le hub reçoit deux messages en même temps, il y a collision

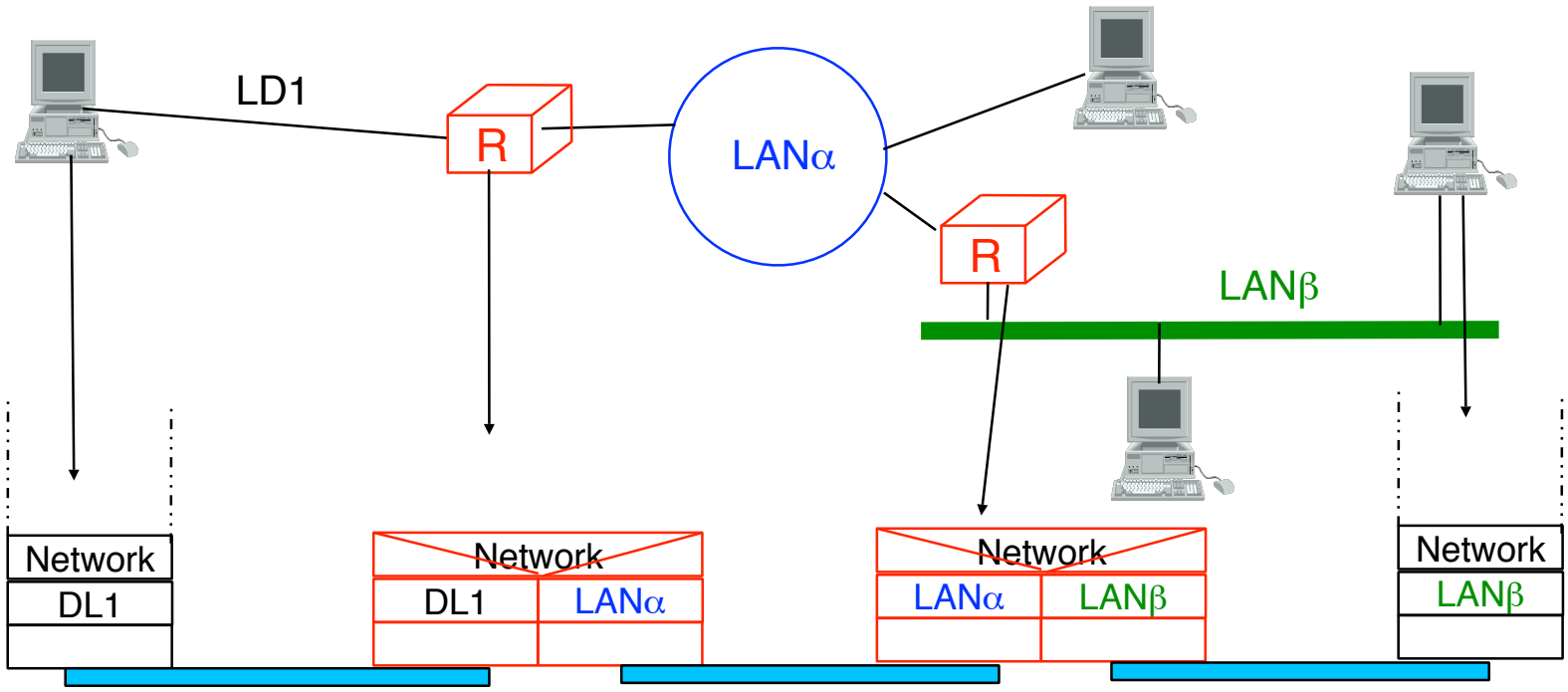
Commutateurs/Ponts

Switch, Bridge



- Lorsqu'il reçoit un paquet, le switch ne l'envoie que là où se trouve la machine destinataire
- Le switch possède une table indiquant, pour chaque adresse MAC, sur quel port rediriger le paquet
- Le switch utilise CSMA/CD pour envoyer les paquets
- Attention, un switch n'a pas d'adresse MAC.

Routeur



Un **routeur** est une machine qui possède des interfaces réseau dans **plusieurs réseaux locaux**.

Problématique

Interconnexion de réseaux locaux sans hypothèse sur la topologie du réseau.

Si une machine veut communiquer avec une autre machine, elle doit pouvoir :

- la nommer $\longrightarrow$ **adressage** ;
- la contacter $\longrightarrow$ **routage**.

La couche réseau spécifie le **format des paquets** qui permet de réaliser ces objectifs.

Adressage IPv4

Une **adresse IP** est codée sur **32 bits** (4 octets) et représentée en notation pointée.

Exemple :

10001100		01011110		00010000		01000000
140	.	94	.	16	.	64

Comment identifier l'ensemble des adresses des machines appartenant au même réseau ?

Voir l'adresse comme un **identifiant de réseau** + un **identifiant d'hôte** sur ce réseau.

Adressage avec classes

Pour une adresse `x.y.z.w` :

classe A $1 \leq x < 128$ (x commence par 0)
Réseau de 2^{24} machines : toutes les machines `x.*.*.*`

classe B $128 \leq x < 192$ (x commence par 10)
Réseau de 2^{16} machines : toutes les machines `x.y.*.*`

classe C $192 \leq x < 224$ (x commence par 110)
Réseau de 2^{8} machines : toutes les machines `x.y.z.*`

classe D $224 \leq x < 240$ (x commence par 1110)
multicast (voir plus loin)

classe E $x \geq 240$ réservé

Broadcast

L'**adresse du réseau local** s'obtient en remplaçant les * par des 0

On envoie un paquet à tout le monde en remplaçant les * par des 255 (ex : `196.168.1.255`)

En général les routeurs ne relaient pas les **broadcast** aux autres réseaux.

Plus généralement, l'adresse `255.255.255.255` permet d'envoyer un paquet à tout le réseau local (sans avoir à connaître le réseau).

Adressage CIDR

Classless Inter-Domain Routing

Remplace l'adressage avec classes et permet une **organisation hiérarchique**.

Un bloc **CIDR** est donnée par une **adresse** et un **masque**

Ex : `192.168.1.0/255.255.255.0`

Une adresse appartient au réseau si le **ET bit à bit** avec le masque est égal à l'adresse :

`192.168.1.58` ?

`192.168.2.0` ?

Notation CIDR

Classless Inter-Domain Routing

Spécifie le **nombre de bits à 1** du masque.

`192.168.1.0/255.255.255.0` = `192.168.1.0/24`

Ex : une machine appartient au réseau si ses 24 premiers bits sont égaux à `192.168.1`

classe A : /8

classe B : /16

classe C : /24

Blocs réservés

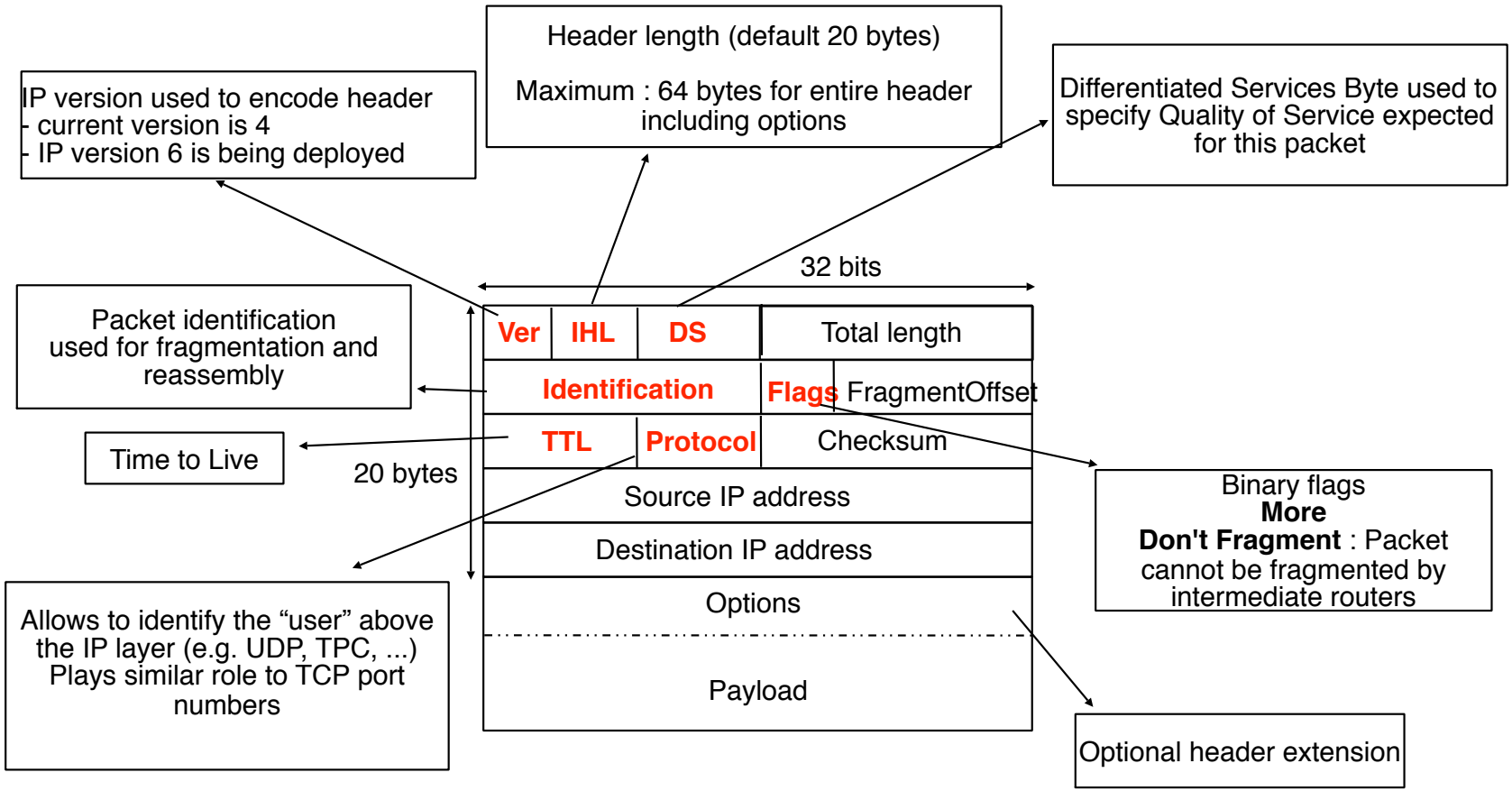
Machine locale (ne se voit jamais sur un réseau) :

- 127.0.0.0/8

Réseaux privés (ne se voit jamais sur l'internet) :

- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16
- 169.254.0.0/16

Format de paquet



Fiabilité et erreurs

La **charge utile** peut être transmise avec erreur mais l'**entête IP** est munie d'un **code détecteur d'erreur**.

Tout paquet comportant un code incorrect est ignoré.

Pour éviter les cycles infinis, le champ **TTL** est décrémenté à chaque saut.

Lorsque **TTL** atteint 0, le paquet est détruit et un **message d'erreur** est envoyé à l'émetteur.

ICMP

Internet Control Message Protocol

Protocole de la couche réseau.

Erreurs, **Diagnostics**.

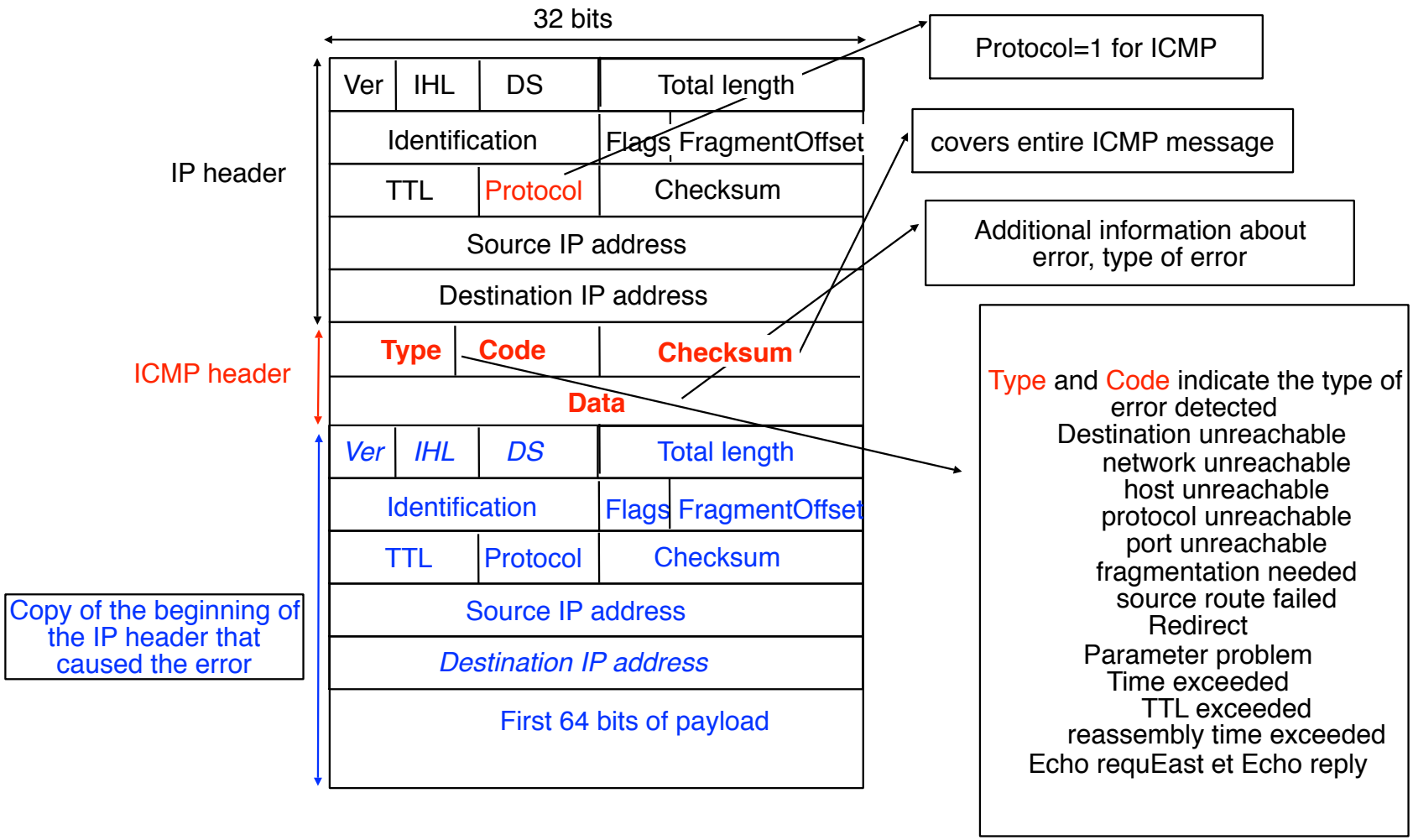
over IP

Exemples :

- Echo request/Echo reply (ping)
- Destination Unreachable
- TTL expired in transit

Comment fonctionne l'utilitaire **traceroute** ?

Messages ICMP



Un hôte

Un **hôte** en bord du réseau possède une **adresse IP** et un **masque de sous-réseau** pour son interface.

Les paquets à destination des machines sur le même réseau sont acheminés directement.

Tous les autres paquets sont transmis à un **routeur** qui joue le rôle de passerelle **par défaut**.

Un routeur

Un **routeur** relie **plusieurs réseaux** entre eux.

Il décide pour chaque paquet entrant, sur quel port il va sortir.

Deux tampons (buffer) :

- Sur le port entrant (avant la décision de routage) ;
- Sur le port sortant.

Table

Chaque routeur possède une **table de routage**.

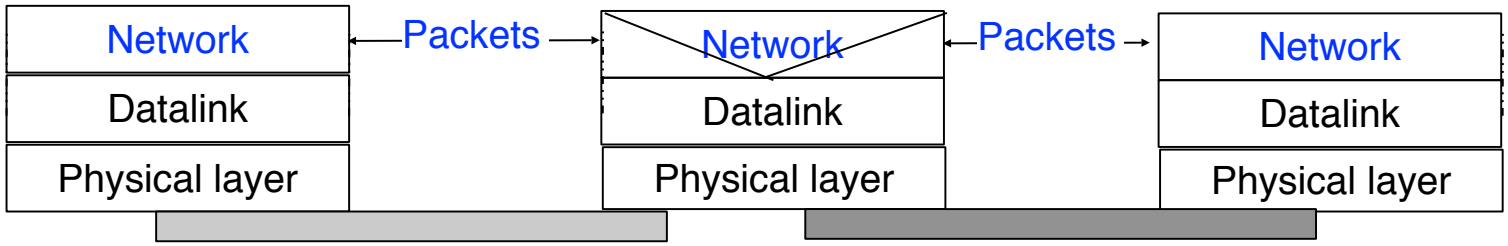
Le routeur **relaie** les paquets en utilisant uniquement cette table.

Toute la subtilité est dans le remplissage (et la mise à jour !) de la table.

Entrée **par défaut** dans la table pour les destinations que le routeur ne connait pas explicitement.

Remplissage les routes **les plus spécifiques d'abord**.

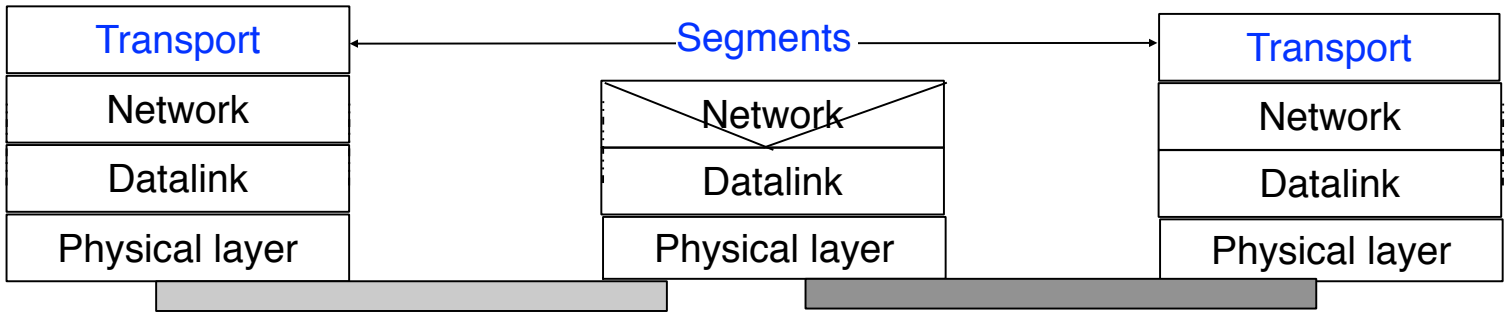
Couche réseau



Réseau **routable** de bout en bout, **non fiable** et **non-connecté** :

- perte de paquets ;
- arrivée dans le désordre ;
- duplication de paquets ;
- longueur maximale limitée (de l'ordre de 64Ko)

Couche transport



Enrichit la couche réseau en ajoutant **fiabilité** et **multiplexage**.

Service **connecté fiable**.
Service **non-connecté non-fiable**.

Ports

La couche transport ajoute une notion de **ports**.

Pour **UDP** et **TCP** un entier codé sur 16 bits.

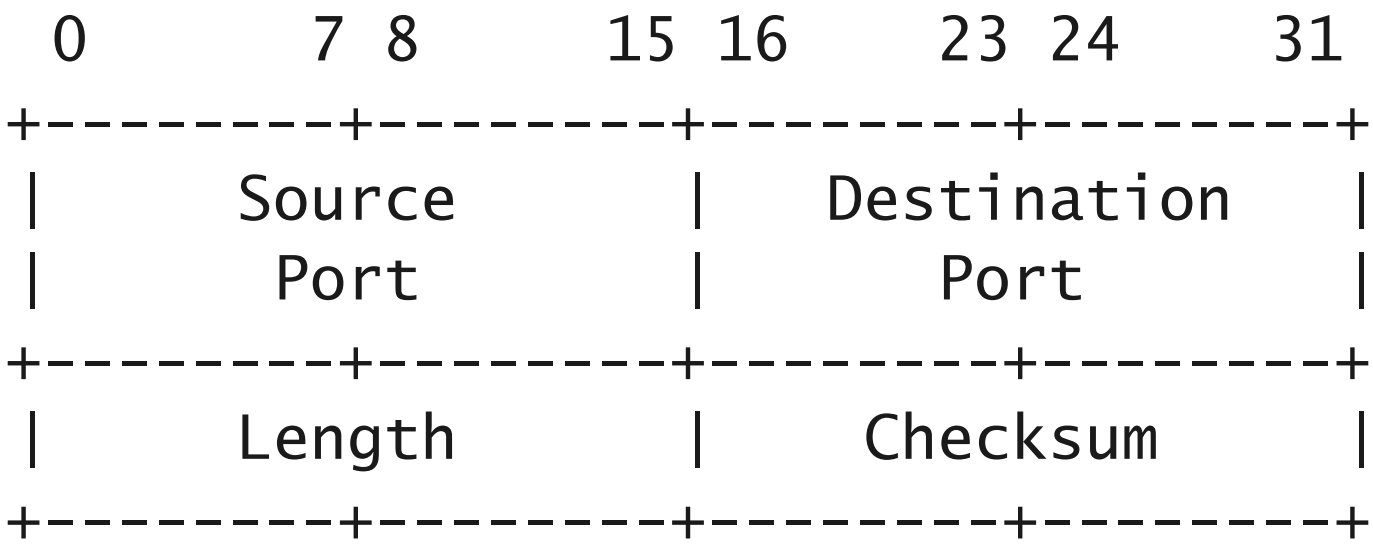
Communication identifiée par un **quadruplet**

(Adresse Réseau A, Port A, Adresse Réseau B, Port B)

UDP

Ajoute les **ports** à IP, et c'est tout

Encapsulé dans IP



Checksum : parité sur 16 bits de UDP+ header IP

TCP

Protocole **connecté** bâti au-dessus d'**IP**.

A la création, on alloue des structures de chaque côté pour s'occuper spécifiquement de la connexion.

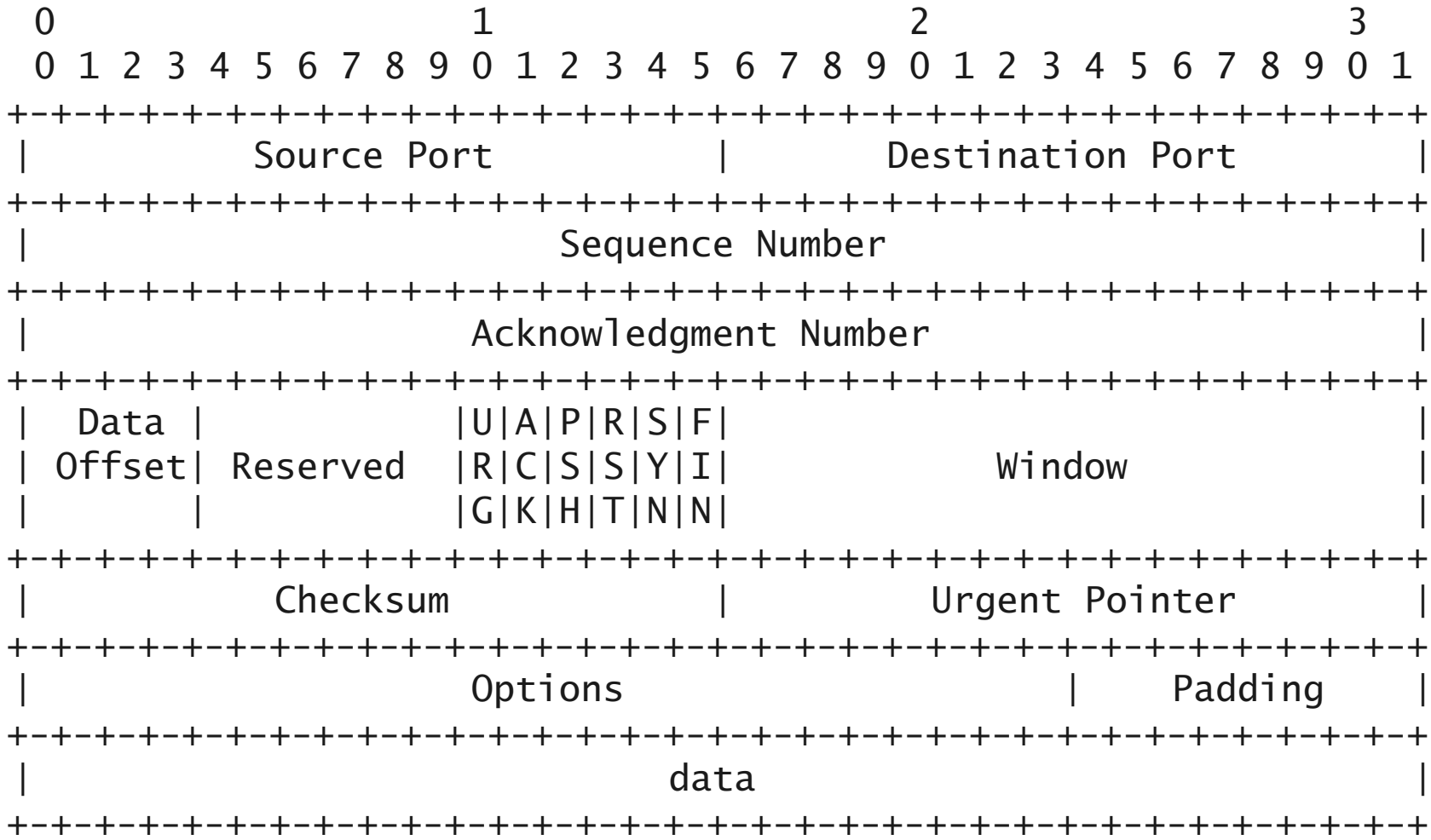
Ajout de la **fiabilité**.

Contrôle de la **congestion**.

L'unité de base en **TCP** s'appelle le **segment**

MSS : Max Segment Size.

Entête TCP


0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Source Port
Destination Port
Sequence Number
Acknowledgment Number
Data Offset
Reserved
U R G
A C K
P S H
R S T
S Y N
F I N
Window
Checksum
Urgent Pointer
Options
Padding
data

Transfert fiable

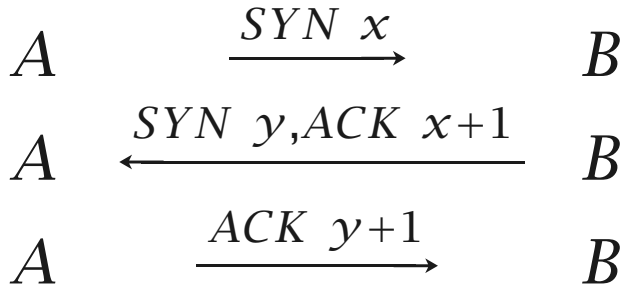
TCP utilise une variante de **GoBackN**.

Contrôle au niveau des **octets** et non pas des segments.

ACK n : j'ai tout reçu jusqu'au n-ième octet non compris.

Pas de **NACK**.

Création de la connexion



Three-way handshake : poignée de main en 3 temps.