SYR – Réseau Licence 3 Info - MIAGE

Couches transport et réseau

Adlen Ksentini

istic Informatique Électronique

UNIVERSITÉ DE RENNES 1

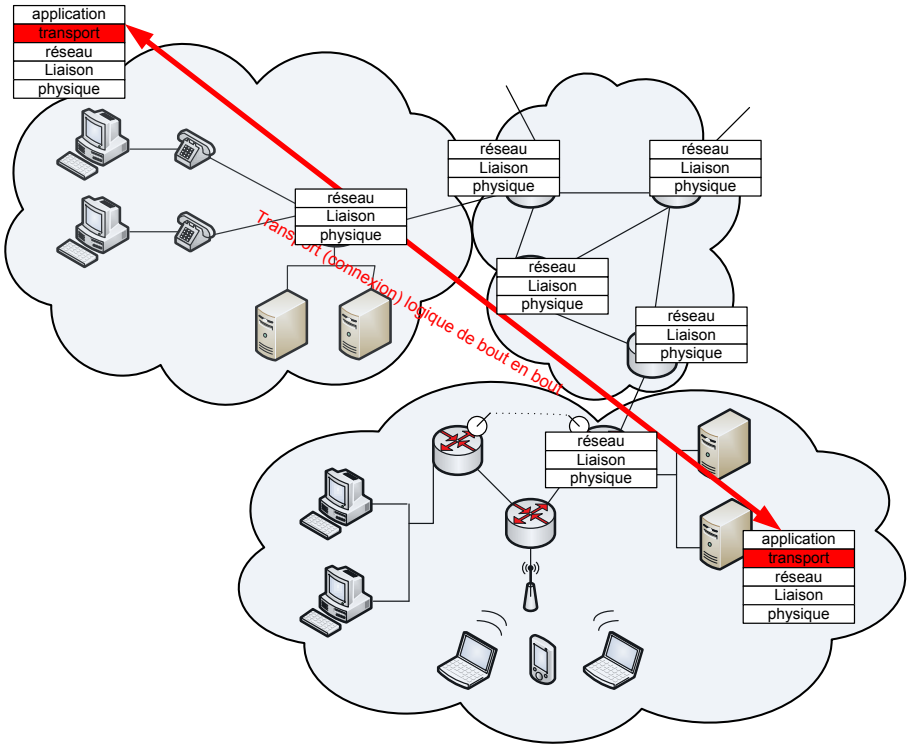
Couche Transport : TCP et UDP

Introduction

- But :
  - Comprendre les principes de fonctionnement de la couche transport Internet
    - Multiplexage/Démultiplexage
    - Transport fiable des données
  - Comprendre :
    - TCP : mode connecté
    - UDP : mode non-connecté

Les services de transport et les protocoles

- Assure un transport logique entre processus fonctionnant sur différents hôtes
- Les protocoles de transport fonctionnent sur les systèmes terminaux
  - Coté émetteur : découper les messages en segments et les passer à la couche réseau
  - Coté récepteur : rassembler les segments en messages, et les passer à la couche application



Couche transport vs. Couche réseau

- Couche réseau : communications logiques entre hôtes
- Couche transport : communication logique entre processus
  - Se base sur, améliore, les services de la couche réseau
  - Ex. Sécurise les échange par l'ajout de la fiabilité

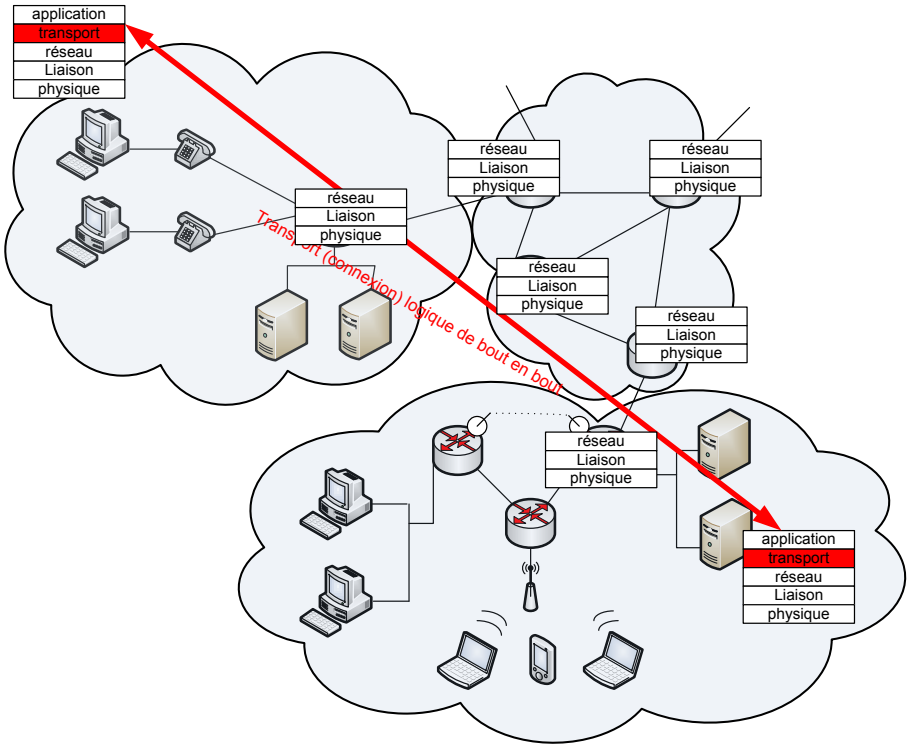
Analogie avec une famille :

12 enfants envoyant des lettres à 12 autres enfants

- Processus = enfants
- Messages couche appl. = les enveloppes contenant les lettres
- Hôtes = maisons
- Protocole de transport = Toto et Titi
- Couche réseau = les services de la poste

Les services de transport Internet

- Fiable, assure l'ordre d'arrivée des segments (TCP)
  - Contrôle de congestion
  - Contrôle de flux
  - Établissement d' une connexion
- Non fiable (UDP)
- Services non disponibles
  - Garantie des délais de bout-en-bout
  - Garantie de la bande passante (débit)

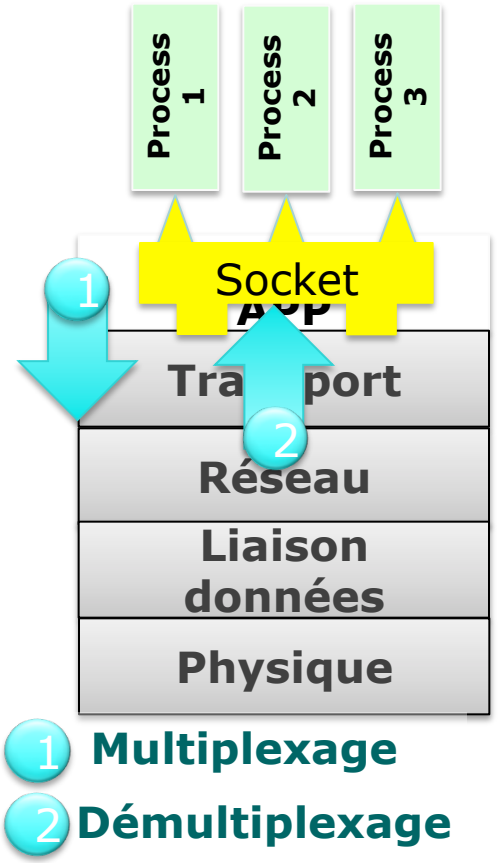


Ports

- **Couche réseau:** les adresses IP désignent les machines entre lesquelles les communications sont établies.
  - On doit adresser le processus s'exécutant sur cette machine.
- **Couche transport:** L'adressage de processus est effectué selon un concept abstrait: **les ports**
  - Les processus sont créés/détruits dynamiquement sur les machines
  - Il faut pouvoir remplacer un processus par un autre (exemple reboot) sans que l'application distante ne s'en aperçoive
  - Il faut identifier les destinations selon les services offerts, sans connaître les processus qui les mettent en œuvre

Ports (suite)

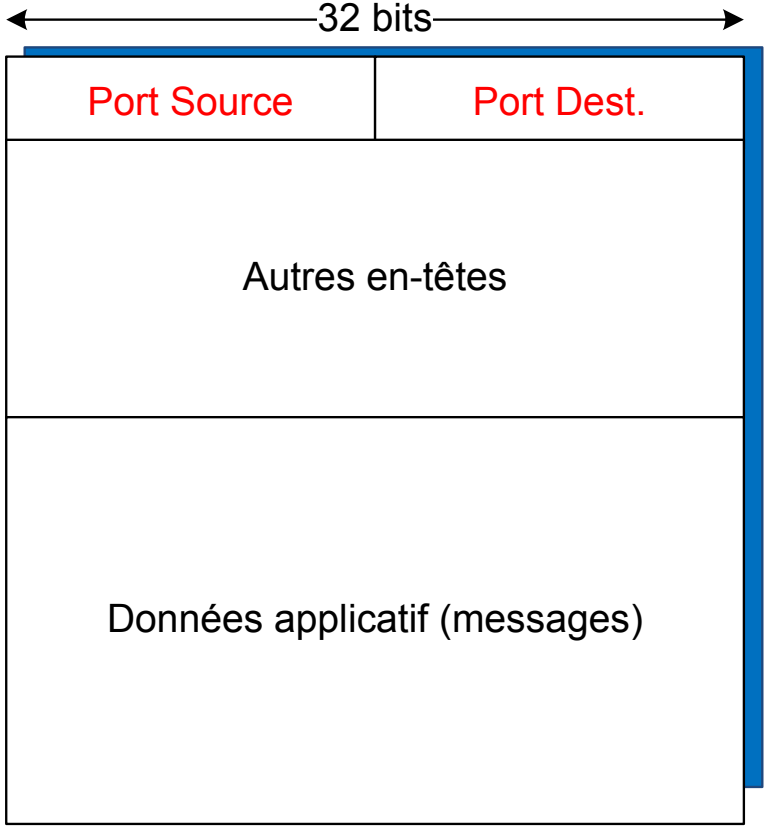
- L'émission d'un message se fait sur la base d'un port SRC (Id Processus SRC) et un port DEST (Id Processus DEST) … Multiplexage
  - La réception se base sur le port DEST (port SRC de la machine distante) … Démultiplexage
- Interface système pour spécifier un port destination (**socket**, ...).
  - Port source spécifier par le système.
- Les accès aux ports sont généralement synchrones, les opérations sur les ports sont tamponnés (files d'attente).



Fonctionnement du démultiplexage

- Un hôte reçoit un datagramme IP
  - Chaque datagramme contient une adresse IP source et une adresse IP destination
  - Chaque datagramme transporte un seul segment de la couche transport
  - Chaque segment contient le port source et le port destination (well-known port pour les applis. connues)
- Un hôte, utilisera alors les @IP et les numéros de port pour diriger le segment vers la bonne socket

Format d'un segment TCP/UDP



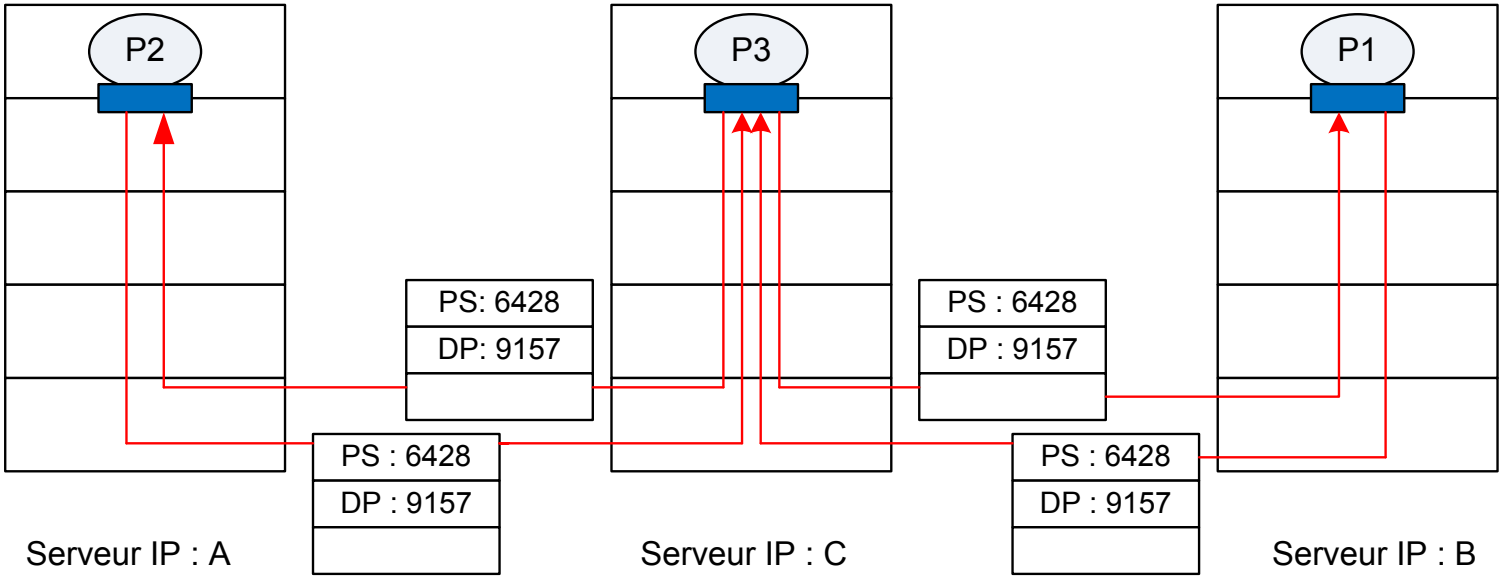
Démultiplexage en mode non connecté

- Création de la socket avec les numéros de ports :

```
DatagramSocket socket1 = new DatagramSocket(9001);
DatagramSocket socket1 = new DatagramSocket(9002);
```

- La socket UDP est identifiée par : (@IP dest., numéro de port dest.)
- Lorsqu'un hôte reçoit un segment UDP
  - Vérifier le port de destination spécifié dans le segment
  - Passer le segment UDP à la socket qui est associée avec ce port
- Les datagrammes IP avec différents @IP sources ou/et numéro de port source, sont passés à la même socket

Démultiplexage en mode non connecté

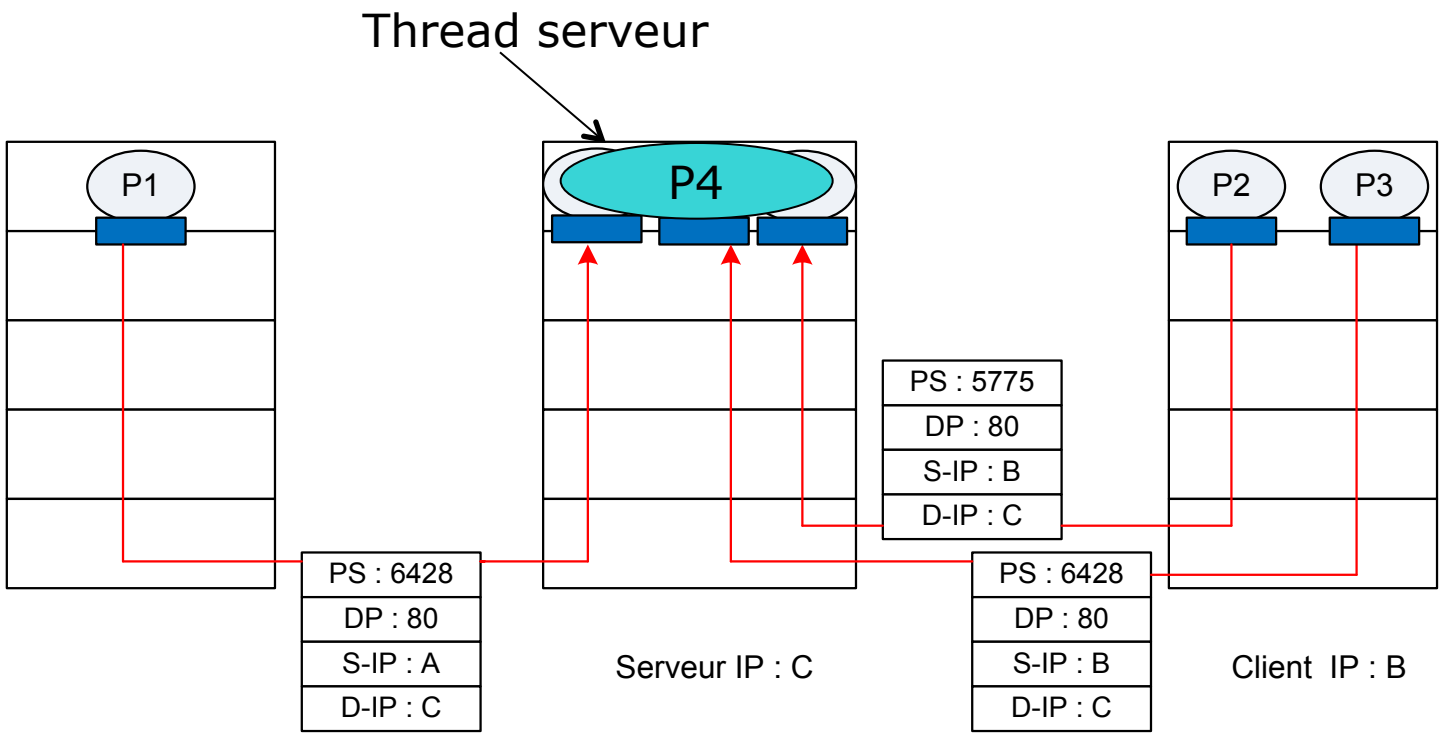


Port Source => d'@ de retour

Démultiplexage en mode connecté

- Une socket TCP est identifiée par :
  - L'@IP source
  - Le numéro de port source
  - L'@IP destination
  - Le numéro de port destination
- L'hôte récepteur utilise ces 4 composants pour passer le trafic vers la bonne socket
- L'hôte serveur peut supporter plusieurs connexions TCP en parallèle
  - Chaque socket est identifiée par ses 4-composants
- Un serveur web a plusieurs sockets pour chaque client connecté
  - Dans HTTP non persistant, une socket pour chaque requête

Démultiplexage en mode connecté (suite)

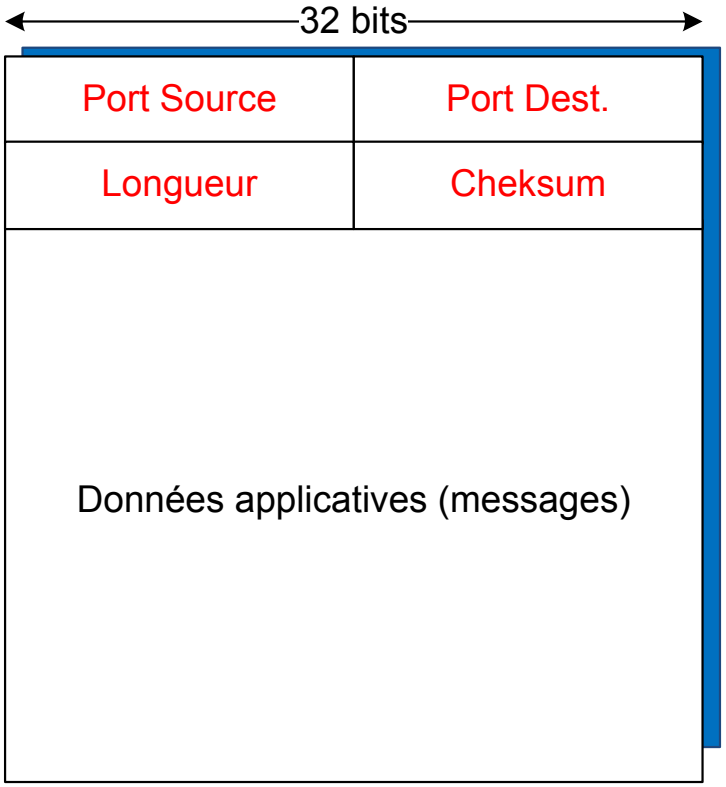


UDP – User Datagram Protocol

- Service Best-effort (au meilleur), les segments UDP peuvent :
  - Se perdre
  - Arriver dans le désordre
- Pas d'établissement de connexion entre l'émetteur UDP et le récepteur
- Chaque segment UDP est traité indépendamment des autres
- Pourquoi utiliser UDP ?
  - Pas d'établissement de connexion (peut diminuer les délais)
  - Segment allégé (faible taille des en-têtes)
  - Pas de contrôle de congestion, l'émetteur UDP peut utiliser toute la bande passante disponible

UDP (suite)

- Utilisé pour les applications de streaming multimédia
  - Tolérance aux pertes
  - Gourmande en bande passante
- Autres applications
  - DNS
- Pour assurer la fiabilité des transferts sur UDP
  - Rajouter du contrôle au niveau de la couche application



Checksum UDP

Transmetteur

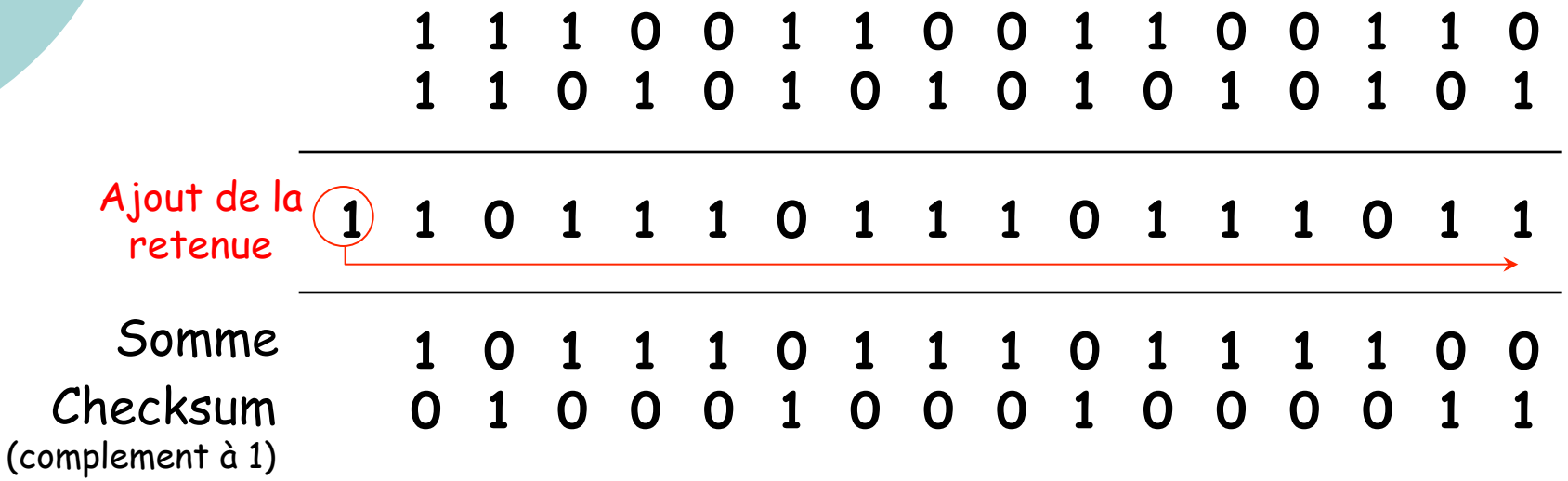
- Le segment, y compris les champs d'en-tête, est traité comme une séquence de nombres entiers de 16 bits
- checksum: addition (la somme de complément à un) du contenu du segment
- Mettre la valeur obtenue dans le champ Checksum

Récepteur

- Calcul le checksum du segment reçu
- Vérifier si les deux checksum sont égaux
  - Non: détection d'une erreur
  - Oui: pas d'erreur détectée (n'élimine pas complètement l'existence d'erreur …)

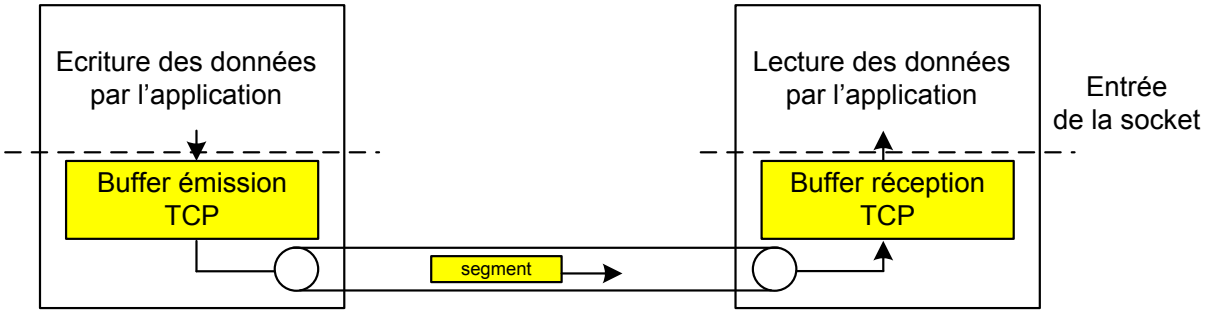
Checksum UDP: exemple

Ex.: addition de deux entiers de 16-bit

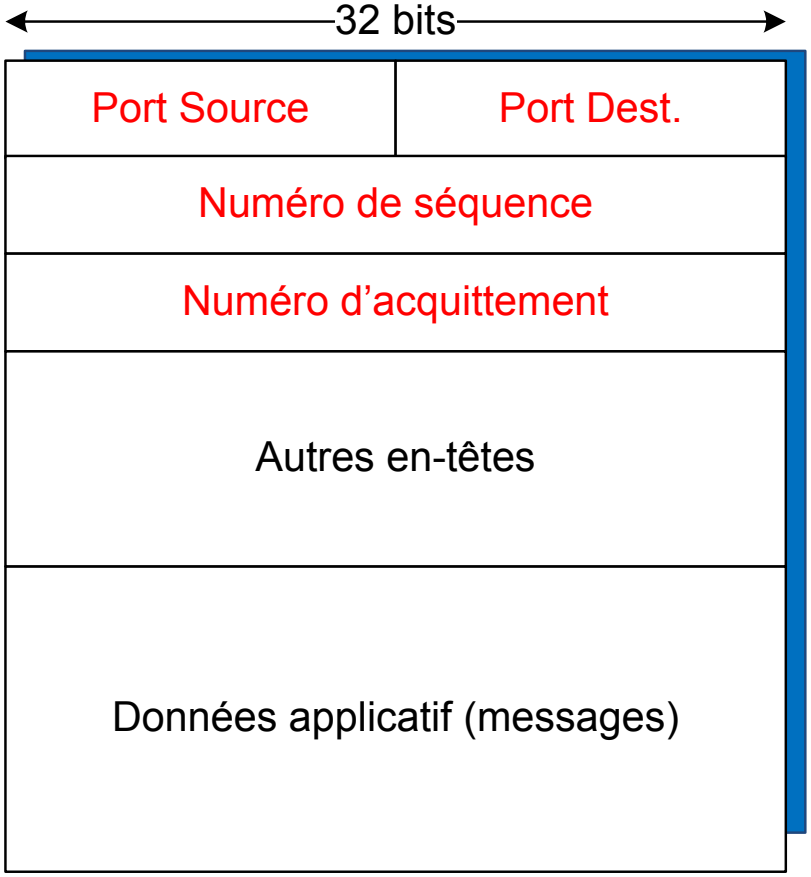


TCP

- RFCs : 793, 1122, 1323, 2018, 2581
- Fiable, réception dans l'ordre
- Etablissement d'une connexion
  - Echange de messages de contrôle pour initialiser les états du récepteur et de l'émetteur
  - Avant l'envoie des messages de données
- Contrôle de congestion
  - Fenêtre d'émission
- Contrôle de flux
  - MSS (Maximum Segment Size) : taille maximum des segments négociée à l'établissement de la connexion
- Utilise des buffers au niveau de l'émetteur et le récepteur

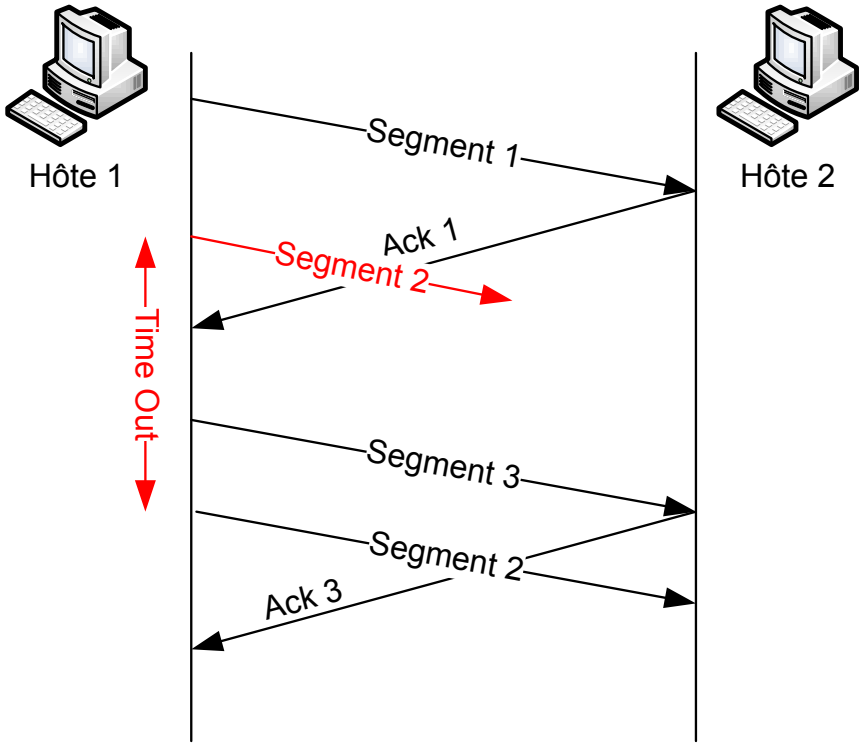


TCP – Format du segment



TCP - Fiabilité

- Segment bien arrivé ?
  - Accusé de réception (Acquittement)



Timeout ?

- Comment le Timeout est défini ?
  - Supérieur à un RTT, mais les RTTs varient
  - Trop petit, résultera de retransmission inutile
  - Trop grand, une réaction tardive aux pertes
- Comment estimer le RTT ?
  - Mesurer ce RTT en temps réel
  - Mesures peuvent varier,
    - Utiliser une moyenne

RTT et Timeout

RTT_estimé = (1-α)*RTT_estimé_avant + α*RTT_mesuré

- Influence des mesures antérieures
- Valeur typique de α = 0.125

RTT et Timeout (suite)

- Définition du Timeout
  - RTT_estimé plus une valeure de garde
    - Variation du RTT estimé
  - Calculer la déviation du RTT_estimé par rapport au RTT_mesuré

```
DevRTT = (1-β)*DevRTT + β (RTT_estimé –
  RTT_mesuré)
```

- Alors le Timeout est défini comme suit :

```
Timeout = RTT_estimé + 4*DevRTT
```

Réception dans l'ordre – N° Séquence

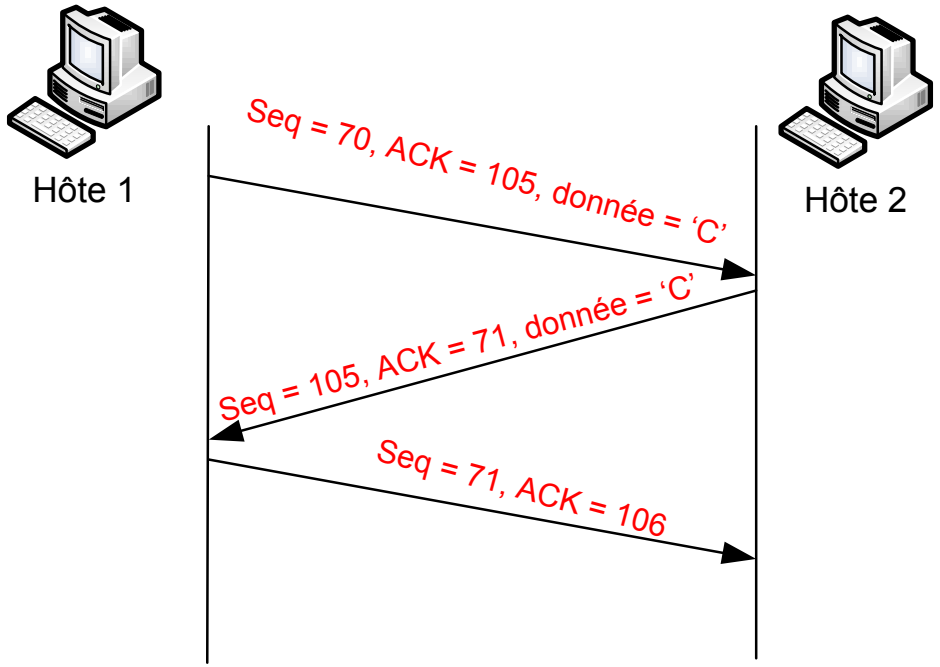
- Ordre nécessaire pour recomposer le message
- Acheminement Internet => commutation par paquet
  - Ne garantit pas l'ordre
    - Chaque segment est acheminé séparément
  - Chemins différents pour segments successifs
- Retransmission

Numéro de séquence

- Introduction d'un « numéro de segment »
  - Introduit un ordre sur les segments
  - Permet d'accuser réception d'un segment
  - Permet de détecter les segments perdus
- Un même numéro ne doit pas être réutilisé

Numéro de séquence - exemple

- Num. seq
  - Le numéro d'emplacement du premier octet du segment dans le flux (stream)
- ACK
  - Le numéro de séquence attendu pour le prochain segment



TCP établissement de la connexion

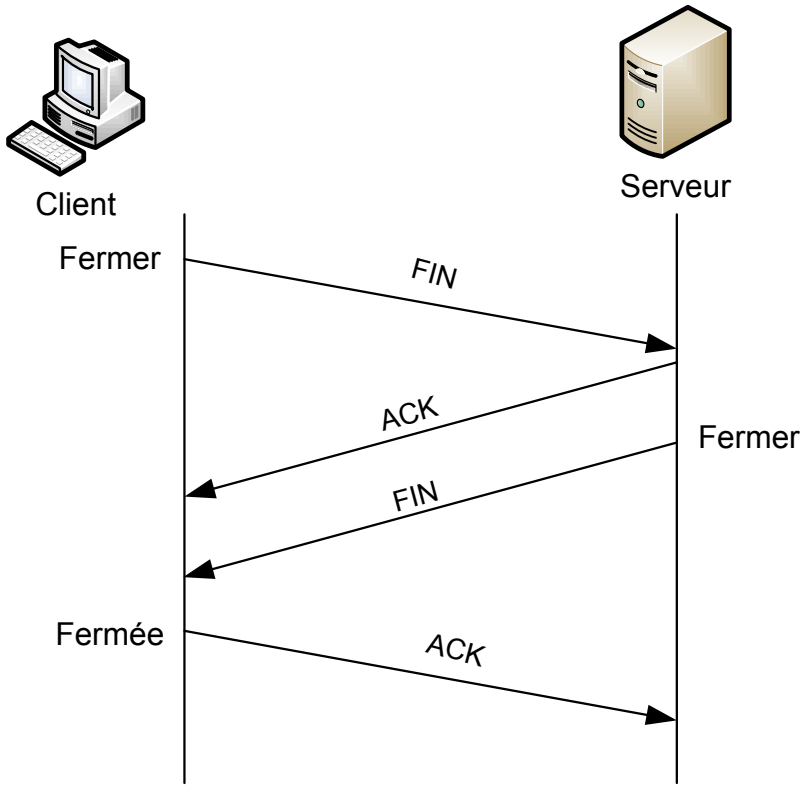
- Etape 1 : le client envoie un segment TCP SYN au serveur
  - Spécifie le numéro de séquence initial
  - Pas de données
- Etape 2 : le serveur répond avec un segment TCP SYNACK
  - Le serveur alloue des buffers à la connexion
  - Spécifie le numéro de séquence initial du serveur
- Etape 3 : le client reçoit le SYNACK et répond par un segment ACK, qui peut contenir des données

TCP fermeture de la connexion

- Le client ferme la connexion

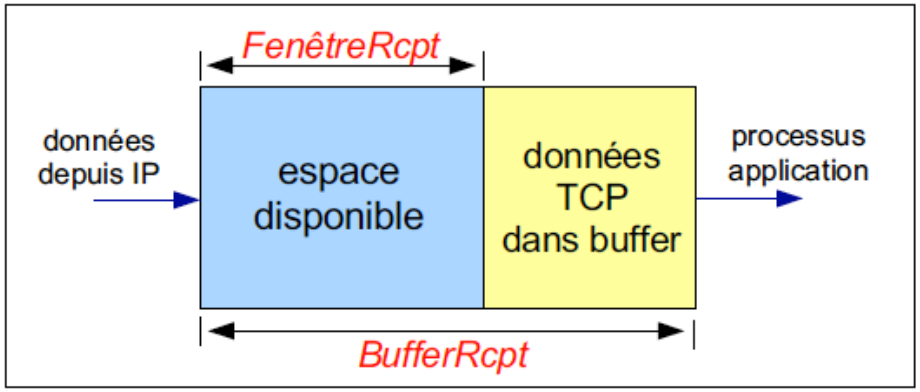
```
ClientSocket.close ();
```

- Etape 1 : Le système du client envoie un segment TCP FIN au serveur
- Etape 2 : Le serveur reçoit le segment FIN et répond par un ACK. Ferme la connexion et envoie un ACK



TCP : Contrôle de flux

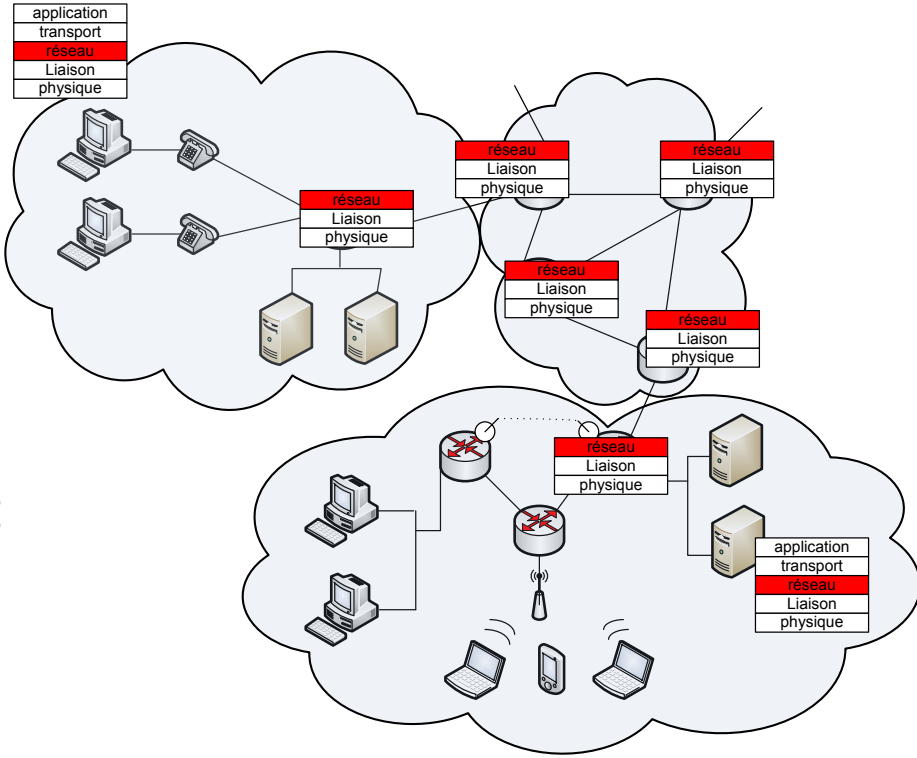
- Le coté récepteur d'une connexion TCP gère un buffer
- Les processus (coté application) peuvent être lents à lire dans le buffer
- Contrôle de flux => l'émetteur ne doit pas faire déborder le buffer récepteur
- Le récepteur montre l'espace disponible en mettant la valeur de la taille actuelle du Buffer dans les segments



Couche Réseau : IP

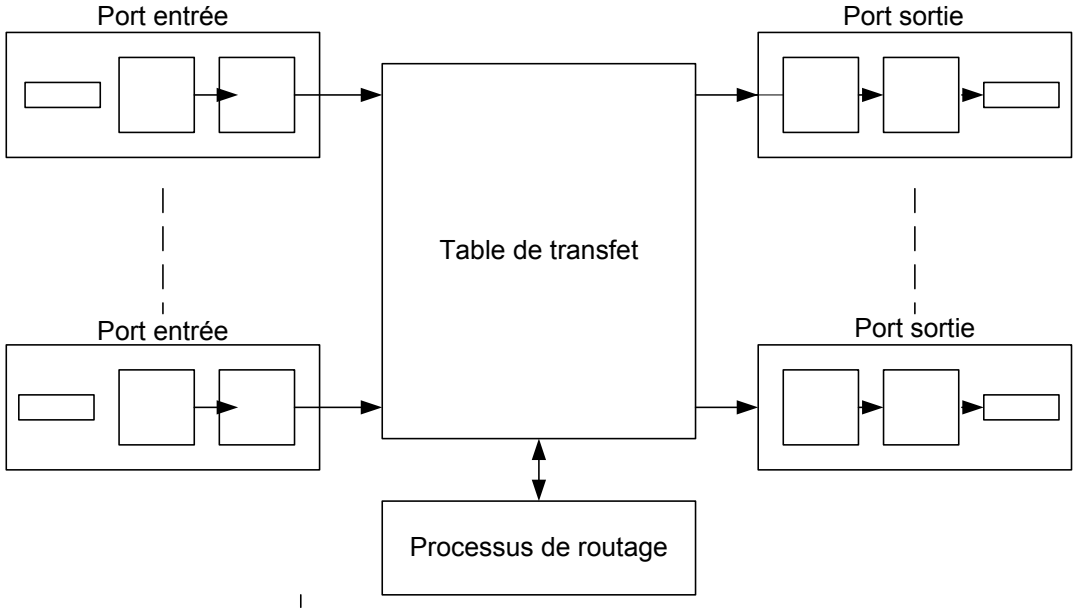
La couche réseau

- Assure le transport des segments entre l'émetteur et le récepteur
- Utilise des adresses logiques : IP
- Du coté émetteur, un segment TCP est encapsulé dans un datagramme
- Du coté récepteur, le segment est délivré à la couche transport
- Le protocole de la couche réseau est exécuté sur tout les hôtes, et routeurs
- Le routeur examine l'en-tête de chaque datagramme
  - Pour le choix du chemin

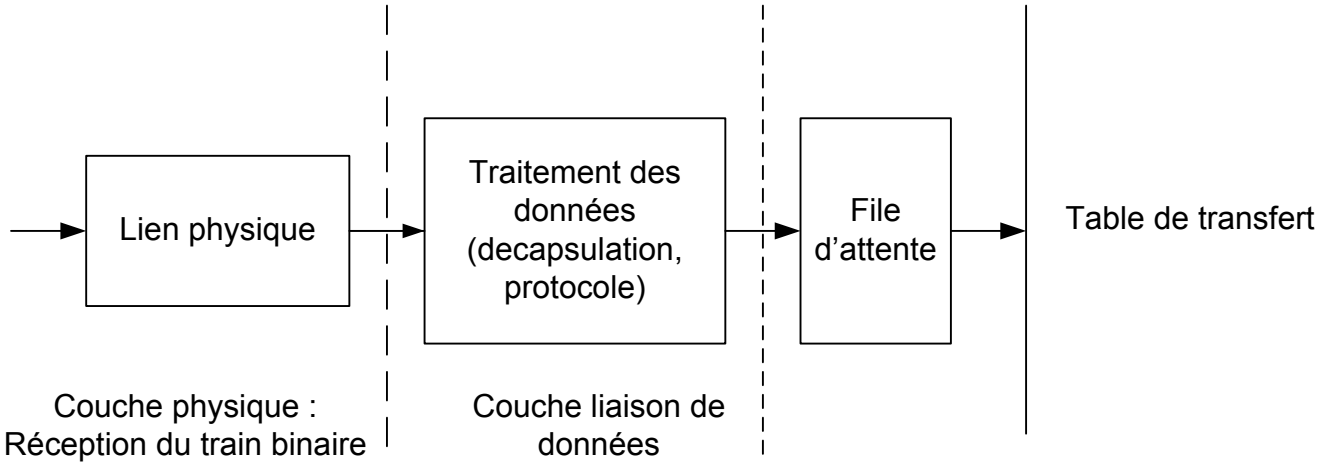


Architecture d'un routeur

- Deux fonctions principales
  - Exécuter un algorithme de routage (RIP, OSPF, BGP)
  - Transférer les paquets d'une interface vers une autre.

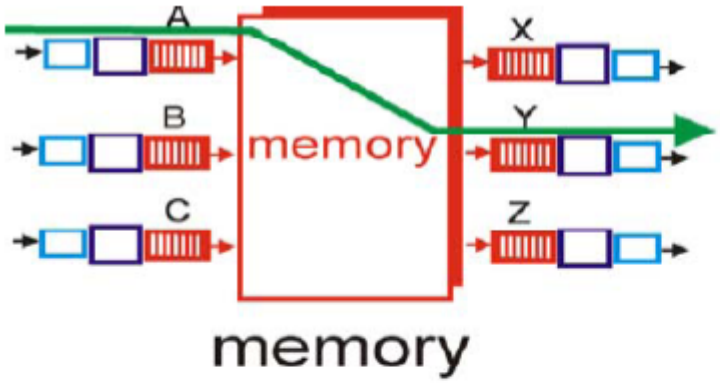


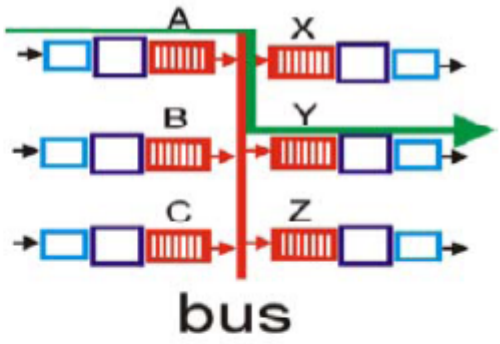
Fonctionnalités du port d'entrée

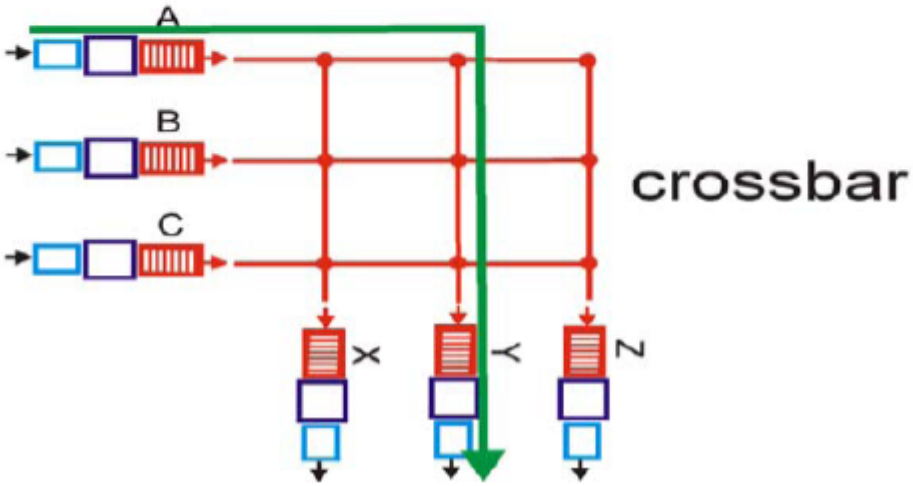


- Traitement décentralisé sur chaque port
- Mise en file : si le datagramme IP arrive plus rapidement que le transfert de données

Table de transfert

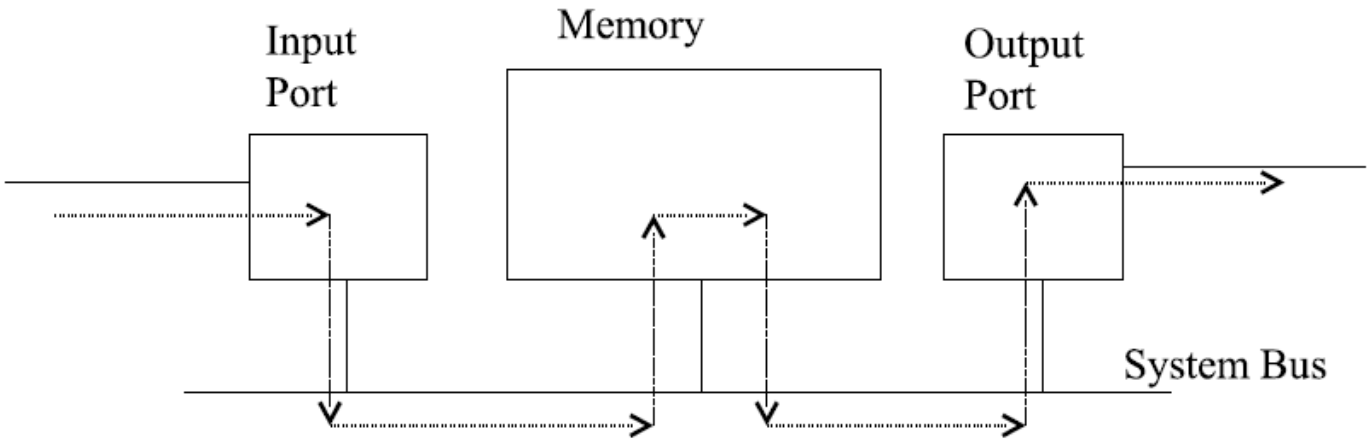






Transfert via la mémoire

- Routeurs de première génération
  - Fonctionne comme un ordinateur faisant office de routeur -> sous le contrôle de la CPU
  - Paquet copié dans la mémoire du système
  - Vitesse limitée par la bande passante de la mémoire.



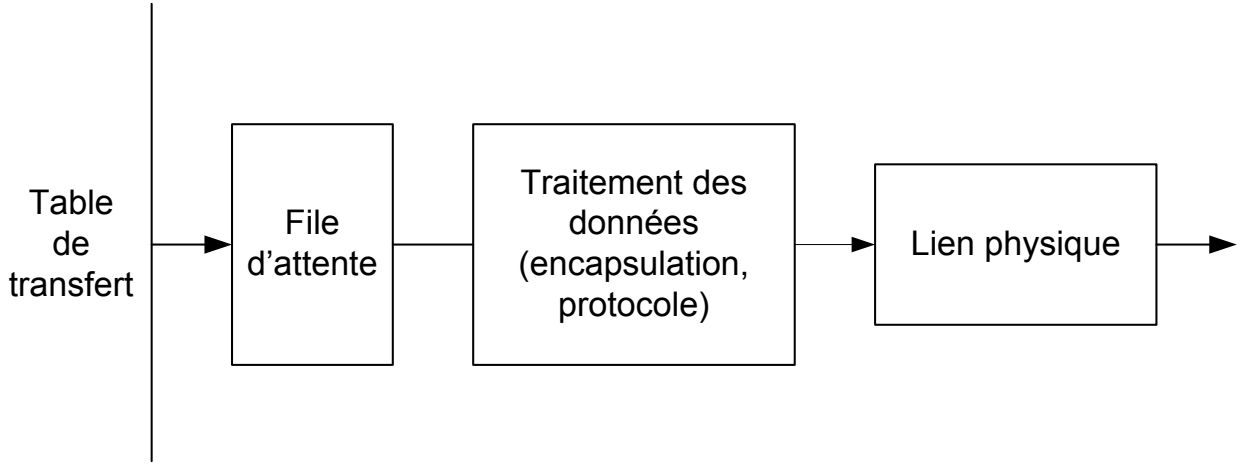
Transfert via un bus

- Datagramme de la mémoire du port d'entrée vers la mémoire du port de sortie à travers un bus partagé
- La vitesse de transfert limitée par la bande passante du bus
- 1 Gbps est suffisant pour un transfert dans le cadre d'un réseau d'accès (pas régionale ni de cœur)

Transfert via réseau d'interconnexion

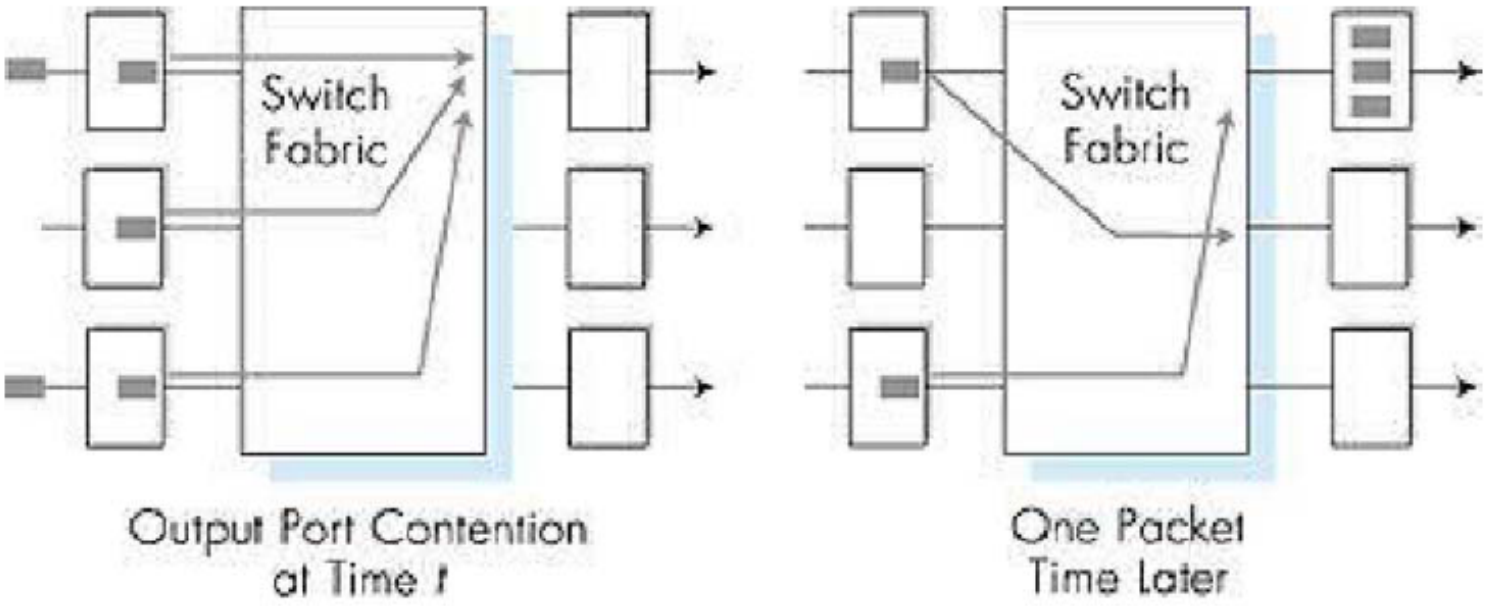
- N'a pas les limitations de la bande passante d'un bus
- Utilisation d'un réseau d'interconnexion initialement dessiné pour connecter les processeur dans le cas de systèmes multiprocesseurs
- Transfert en Gbps.

Ports de sorties



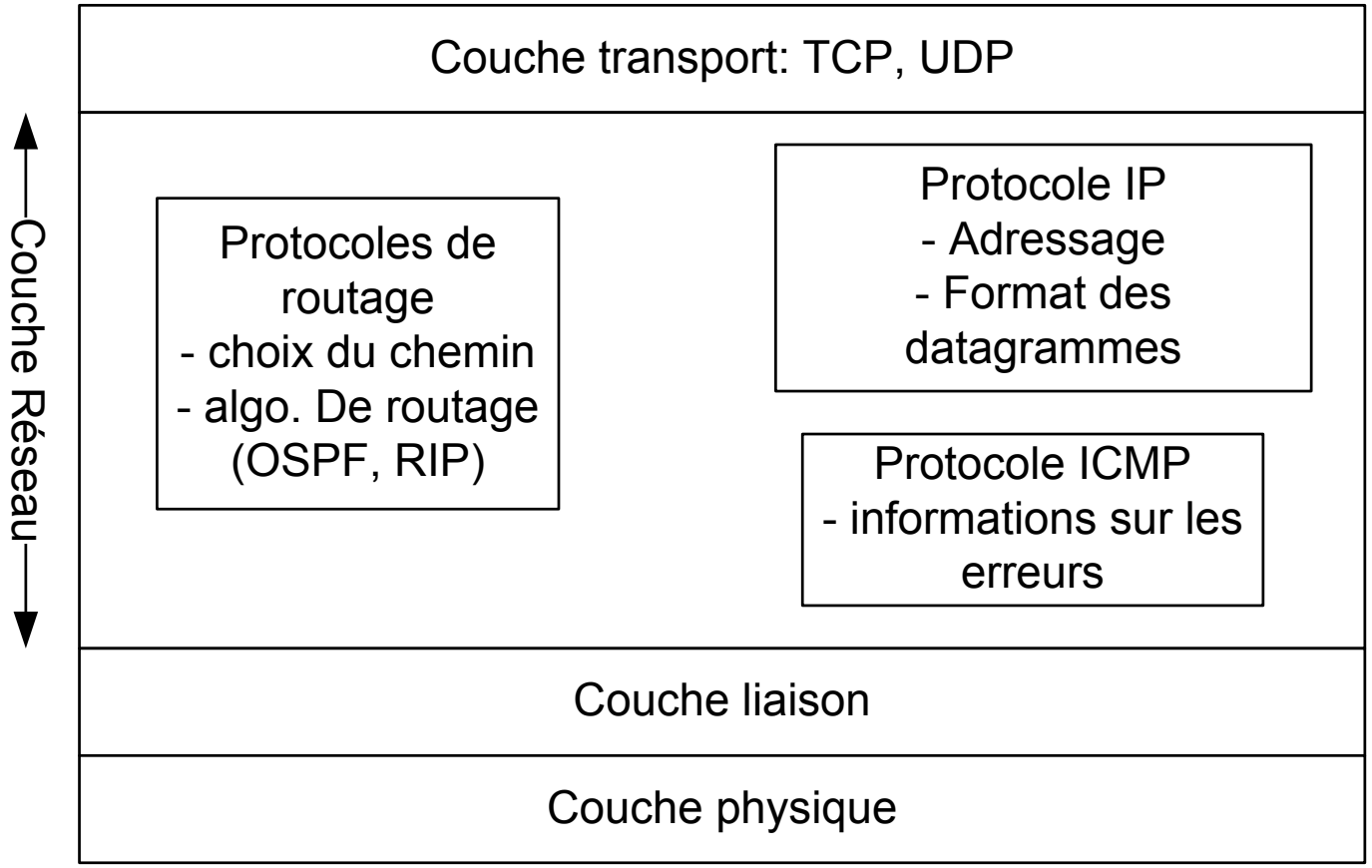
- Mise en file d'attente lorsque les datagrammes IP arrivent avec une vitesse plus élevée que la vitesse de la ligne de sortie
- Discipline de scheduling

Port de sortie : file d'attente



- Délais dans la file d'attente et pertes causés par le dépassement de la taille du buffer de sortie

Couche réseau : Internet

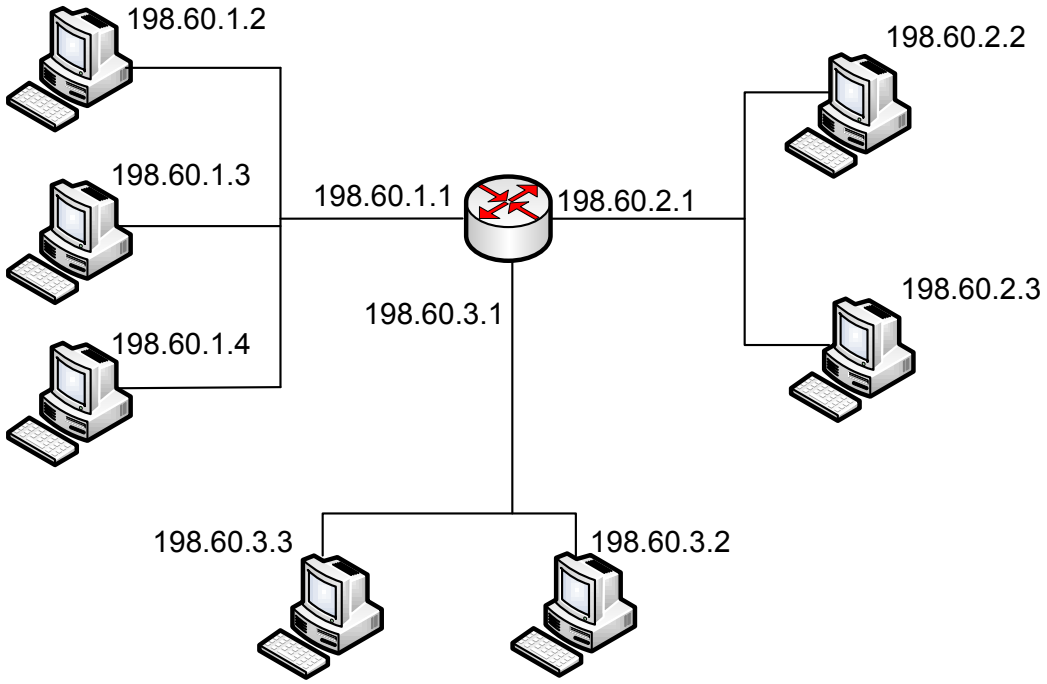


- Hôte, routeur : fonctionnalité de la couche réseau

Adressage IP

- Une adresse IP : 32 bits qui identifie un hôte, ou une *interface* d'un routeur
  - Notée en décimale
- Interface : connexion entre un routeur/ hôte et le lien physique
  - Un routeur contient plusieurs interfaces
  - Un hôte peut avoir plusieurs interfaces
  - Chaque interface a une adresse IP

Adressage IP

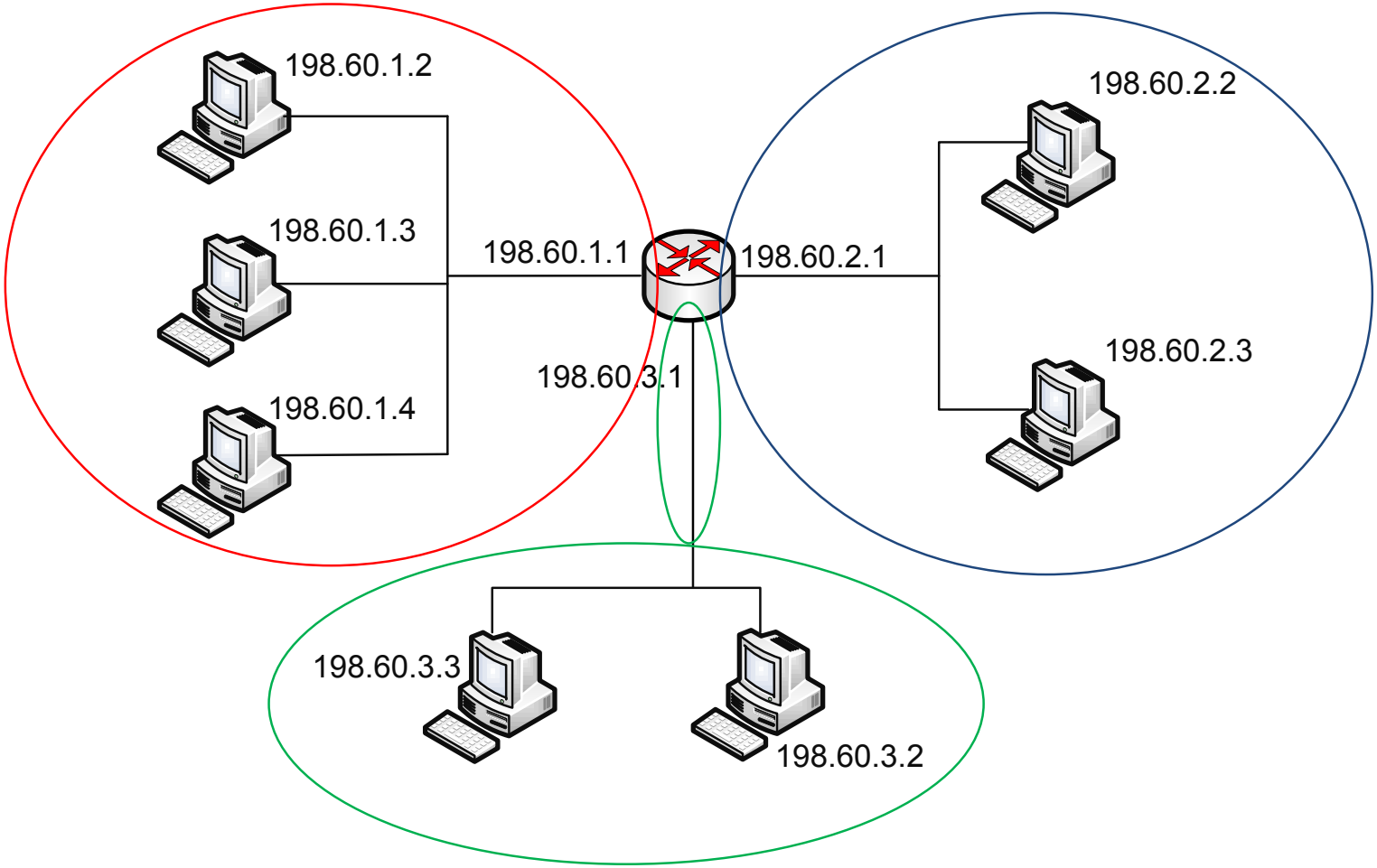


148.168.1.1 = 11000000 10101000 00000001 00000001

Sous-réseaux

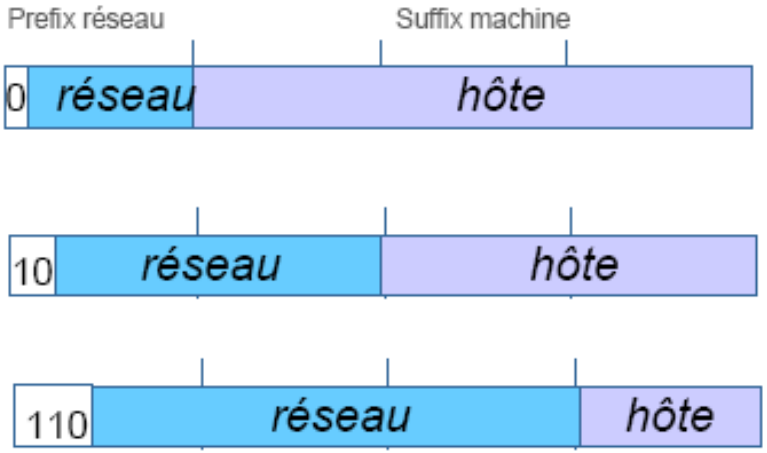
- Adresse IP :
  - Une partie pour identifier le sous-réseau
  - Une partie pour identifier le hôte
- Sous-réseaux ?
  - Les interfaces ayant le même identifiant du sous-réseau (dans l'@IP)
  - Deux hôtes sur le même sous réseau communiquent directement (sans passer par un routeur

Sous-réseaux (suite)



Un réseau avec trois sous-réseaux

Sous-réseaux et classes d'adresses



- Comment connaître les sous-réseaux ?
  - Classe d'adresses IP et masque de sous réseaux
- Classe A : 1.x.x.x à 127.x.x.x
  - 127 réseaux, 16777216 machines
- Classe B : 128.0.x.x à 191.255.x.x
  - 16384 réseaux, 65536 machines
- Classe C : 192.0.0.x à 223.255.255.x
  - 2097152 réseaux, 256 machines

Masque de sous-réseaux

- Masque du réseau : adresse IP particulière servant à identifier l'adresse du réseau à partir d'une adresse IP de machine.
  - Le masque d'un réseau de classe A = 255.0.0
  - Le masque d'un réseau de classe B = 255.255.0.0
  - Le masque d'un réseau de classe C = 255.255.255.0
- Adresses réseau : adresse IP dont la partie « hostid » ne comprend que des zéros;
  - la valeur zéro ne peut être attribuée à une machine réelle : 192.20.0.0 désigne le réseau de classe C 192.20.0

Masque de sous-réseau (suite)

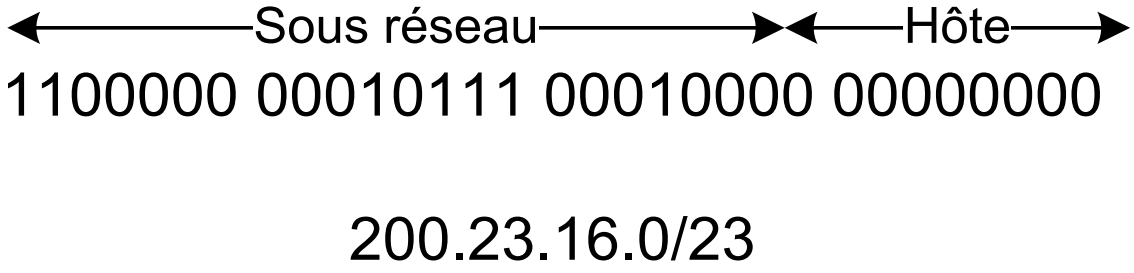
- Permet à une station de savoir si la station destination est dans le même réseau qu'elle ou s'il lui faut envoyer son paquet au routeur qui l'acheminera
- Exemple station A veut envoyer un paquet à une station B :
  - @IP de A : 198.60.12.2
  - @IP de B : 198.60.12.5
  - @ netmask A : 255.255.0.0
- La station A doit réaliser 3 opérations
  - @A AND @netmask = Res1
  - @B AND @netmask = Res2
  - Comparer Res1 et Res2
    - Si Res1 = Res2 alors stations sur le même réseau
    - Sinon stations sur réseaux distants

Adresses particulières

- 0.0.0.0 : utilisée au démarrage pour la configuration automatique d'une @IP
- 127.0.0.1 : réservée pour la désignation de la machine locale, c'est à dire la communication intra-machine.
  - Associée nom « localhost »
- Adresses IP privées
  - Classe A : 10.0.0.0 -> 10.255.255.255
  - Classe B : 172.16.0.0 -> 172.16.255.255
  - Classe C : 192.168.0.0 -> 192.168.255.255

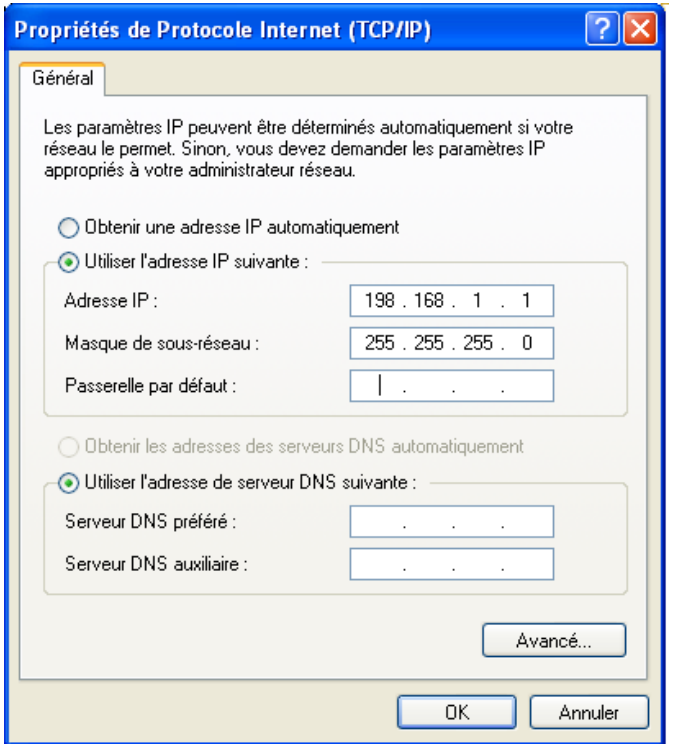
Adressage IP : CIDR

- CIDR : Classless InterDomain Routing
  - Format de l'adresse : a.b.c.d/x, où x est le nombre de bits égale à 1 dans le masque



Comment définir une adresse IP

- Manuellement
  - Windows : Panneau de configuration-> connexions réseau -> nom_connexion -> protpriété (bouton droit de la souris) -> tcp/ip -> propriétés



  - UNIX : /etc/rc.conf

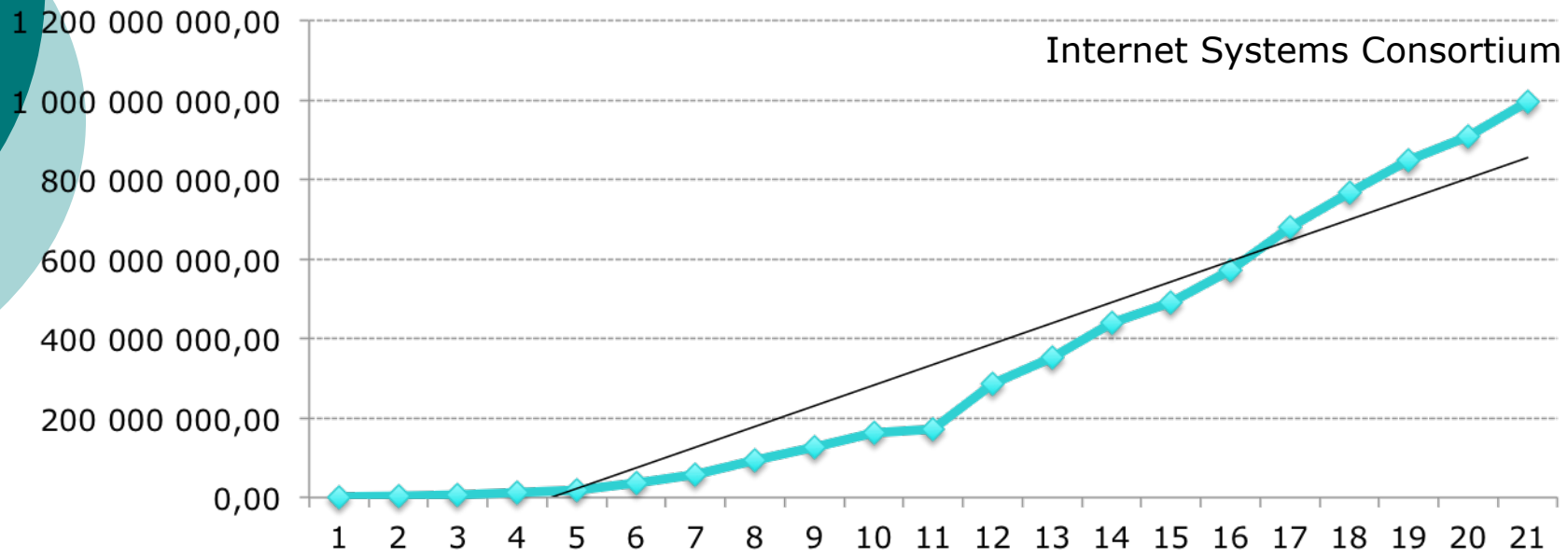
Comment définir une adresse IP

- Dynamiquement
  - Utiliser un serveur DHCP (Dynamic Host Configuration Protocols)
    - Le client DHCP (hôte) demande une @IP
    - Le serveur DHCP, lui envoie une @IP avec une durée de vie limité
  - Plug-and-play

Problème des adresses IPv4

- L'assignation d'une classe par bit, signifie que: la classe A prend 1/2 des adresses, la classe B 1/4, la classe C 1/8 etc.
- Problèmes avec une telle assignation :
  - Saturation dans les routeurs
  - Manque d'adresses IPv4
    1. Gaspillage
    3. Pénurie des adresses encore libres

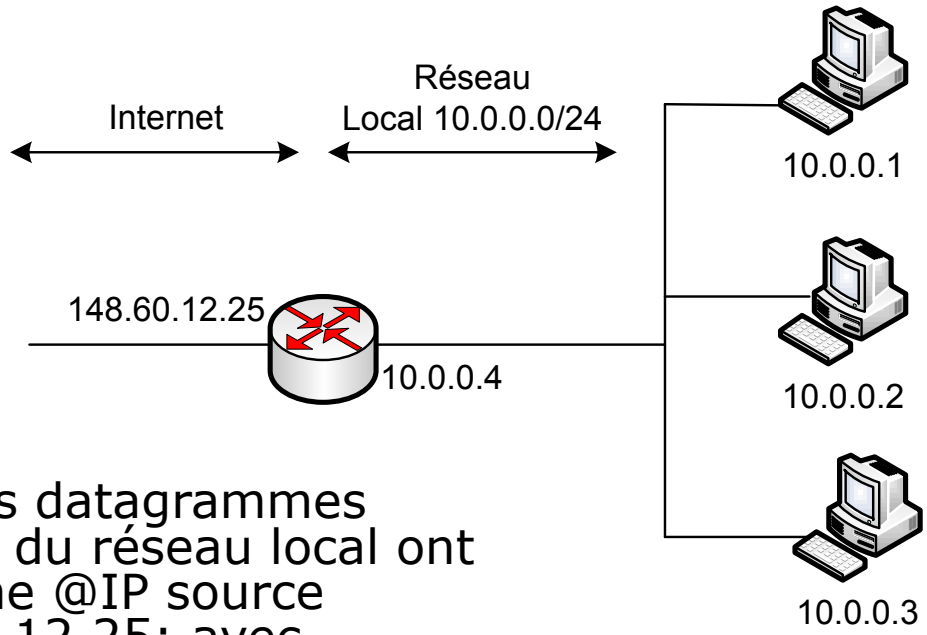
Internet aujourd'hui



- Croissance phénoménale - 1 million d'utilisateurs/mois
- L'accroissement aujourd'hui est quasi linéaire (expo. Jusqu'à 1998)
- Prévision de pénurie d'adresses IPv4 vers 2010-2015

- **L'IANA a distribué ses derniers bloques d'adresses IP (5 bloques de 16 M d'adresses) le 03/02/2011, laissant aux registres régionaux d'internet (RIRs) la distributions des derniers bloques, qui devraient s'épuiser dans quelques mois.**
- **RIPE NCC (registre régional de référence pour la zone Europe) n'a plus d'adresse à fournir (17/09/2012)**
- **) …**

Translation d'adresse ou NAT

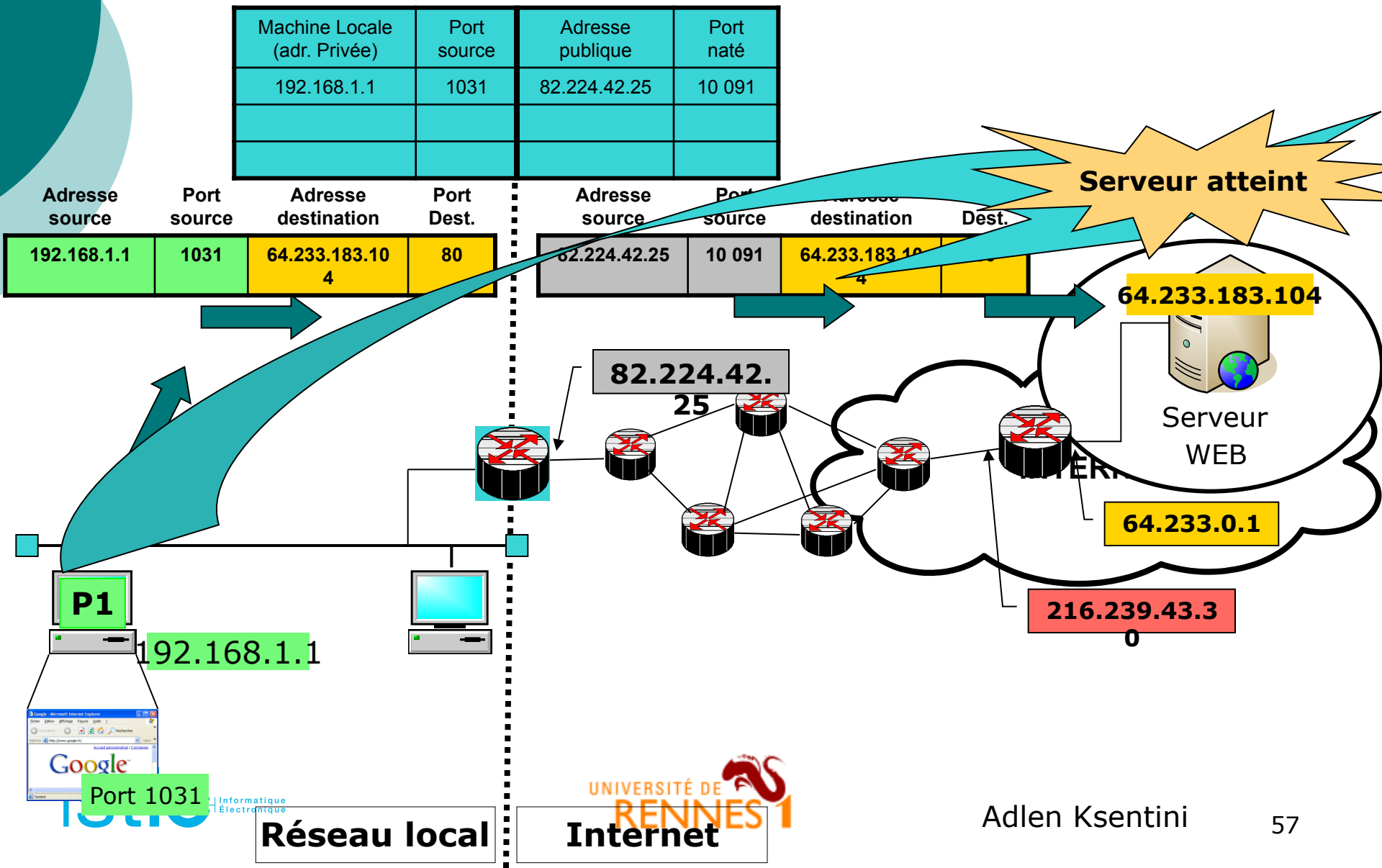


- Tous les datagrammes sortant du réseau local ont la même @IP source 148.60.12.25; avec différent numéros ports sources
- Datagrammes avec @ source et destination dans ce réseau sont sous la forme 10.0.0.X/24

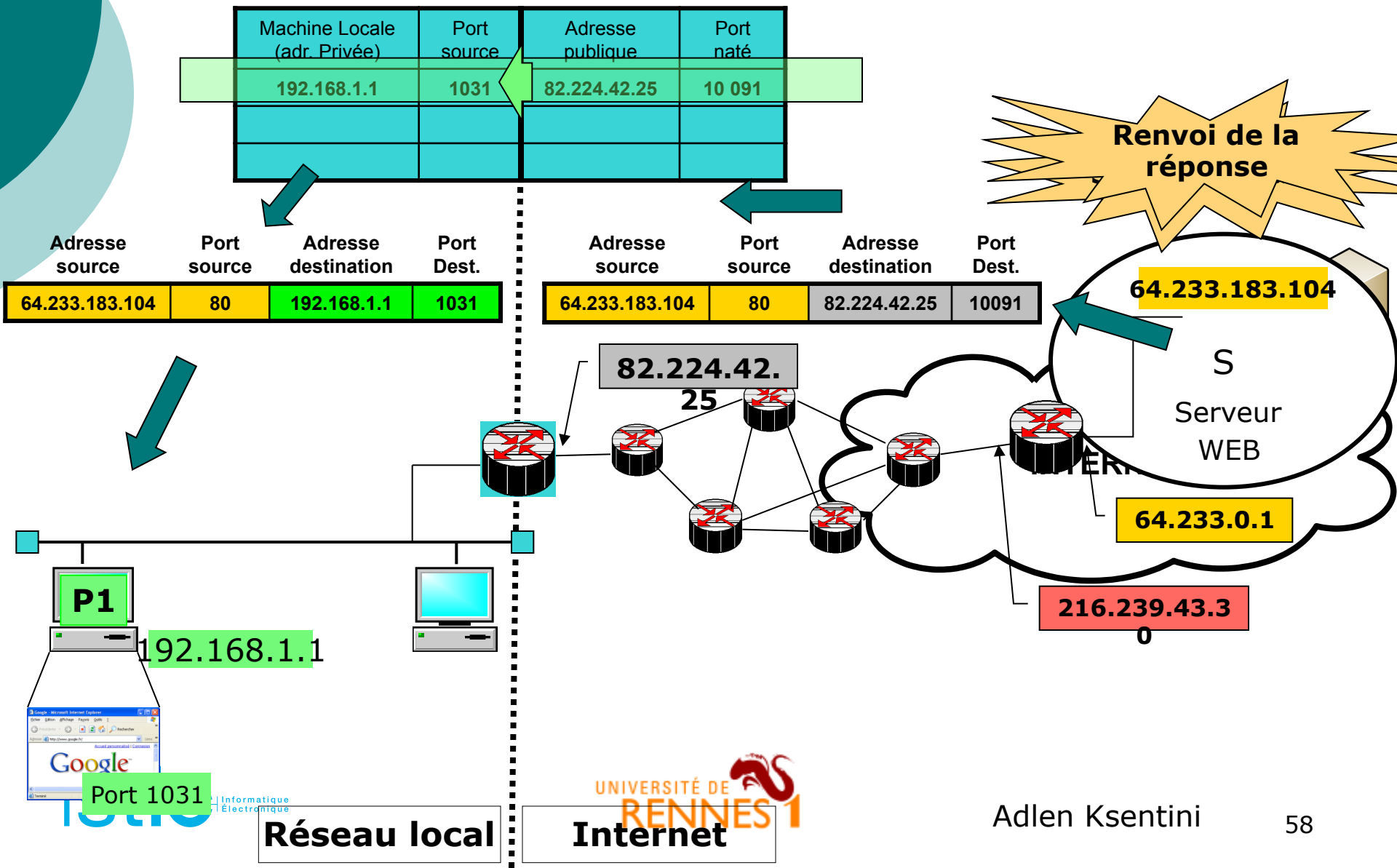
Translation d'adresse ou NAT (2)

- Motivation : Le réseau local aura besoin d'une seule @IP publique
  - Pas besoin d'une plage d'@IP publique; une seule @ pour tous les hôtes
  - Possibilité de changer les @IP des hôtes du réseau local sans aucune notification pour le FAI
  - Possibilité de changer de FAI sans affecter l'adressage du réseau local
  - Les hôtes du réseau local ne sont pas visibles par l'extérieur

Principe du NAT/PAT



Principe du NAT

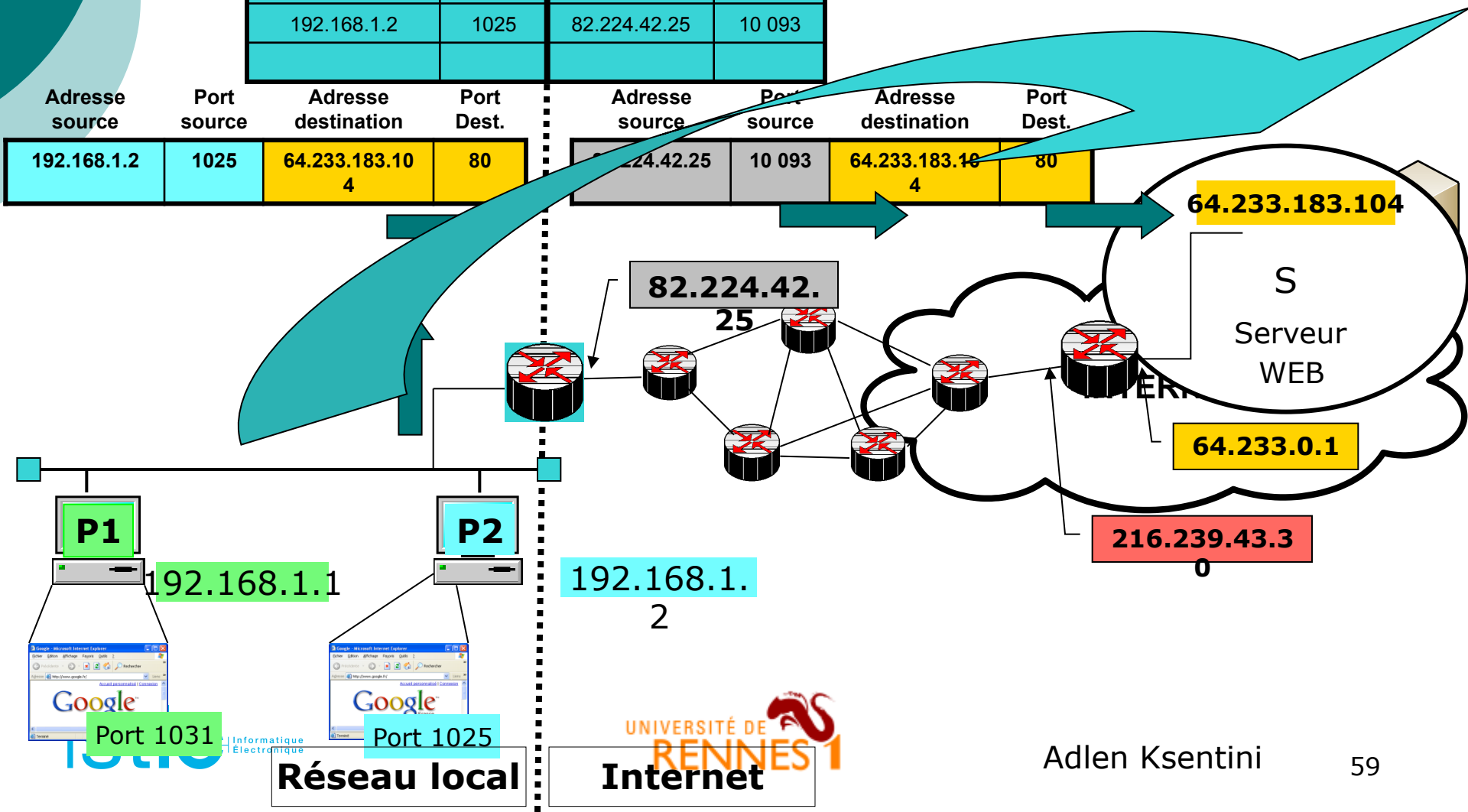


Principe du NAT

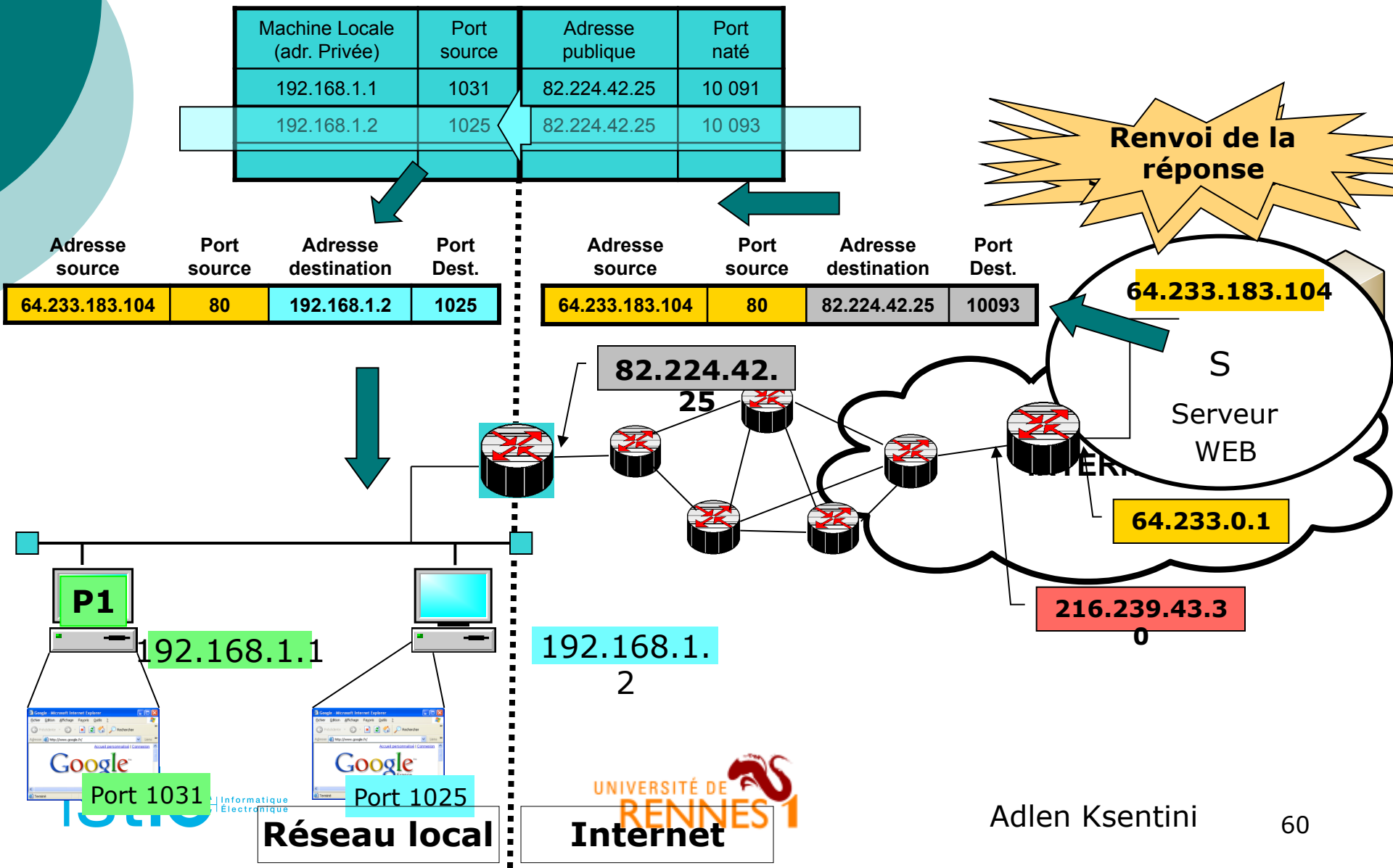
Machine Locale (adr. Privée)	Port source	Adresse publique	Port naté
192.168.1.1	1031	82.224.42.25	10 091
192.168.1.2	1025	82.224.42.25	10 093

Adresse source	Port source	Adresse destination	Port Dest.
192.168.1.2	1025	64.233.183.104	80

Adresse source	Port source	Adresse destination	Port Dest.
82.224.42.25	10 093	64.233.183.104	80



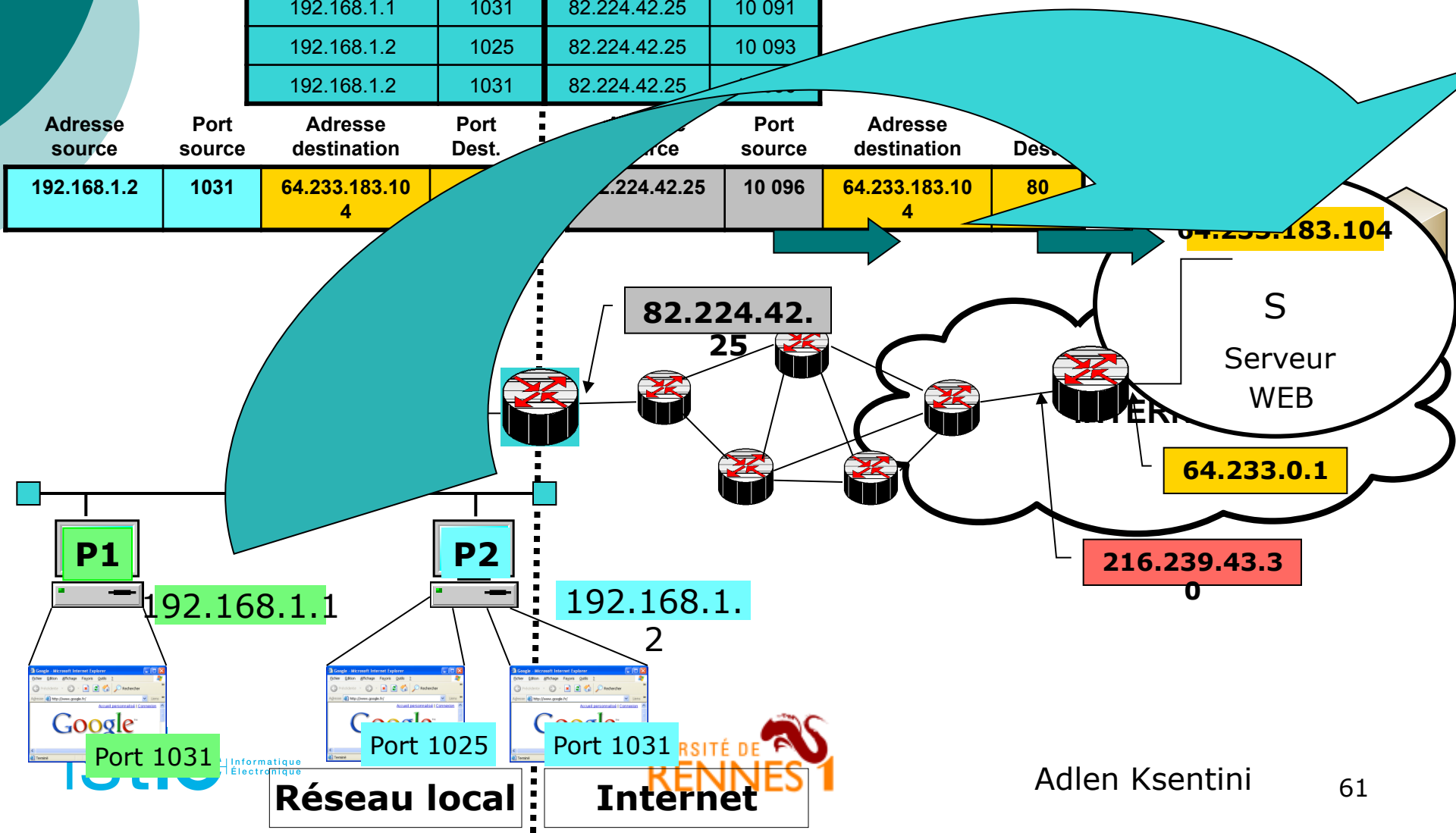
Principe du NAT



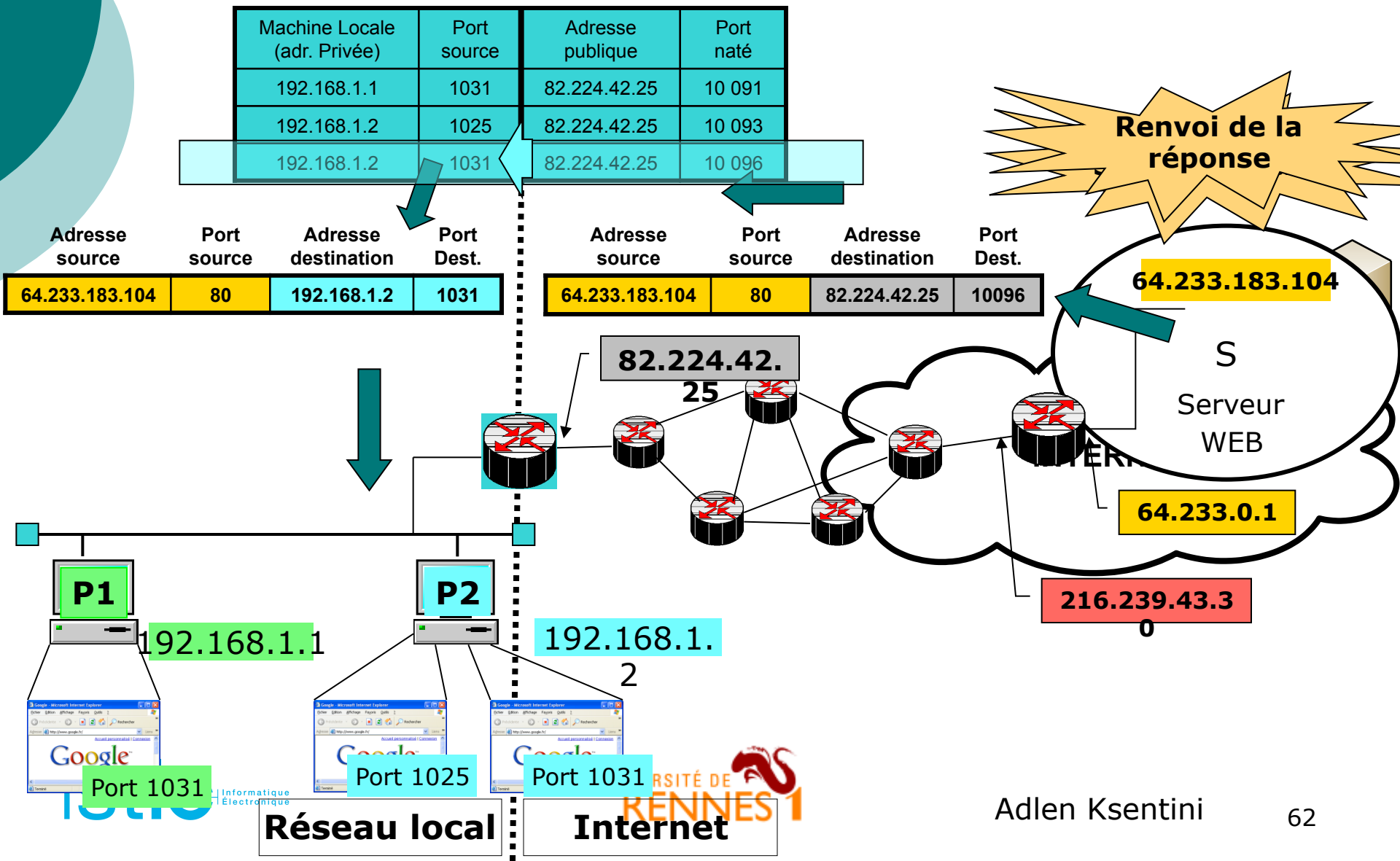
Principe du NAT

Machine Locale (adr. Privée)	Port source	Adresse publique	Port naté
192.168.1.1	1031	82.224.42.25	10 091
192.168.1.2	1025	82.224.42.25	10 093
192.168.1.2	1031	82.224.42.25	

Adresse source	Port source	Adresse destination	Port Dest.		Port source	Adresse destination	Dest.
192.168.1.2	1031	64.233.183.104		.224.42.25	10 096	64.233.183.104	80



Principe du NAT



ICMP : Internet Control Message Protocol

- Utilisé par les hôtes et routeurs pour communiquer les informations de niveau réseau
  - Rapports d'erreurs (hôte ne répond pas, erreurs de routage …)
  - Echo request/reply (utilisé par l'application ping)
  - Les messages ICMP sont transportés par des datagrammes IP