0 - Au programme

- Cours : rappels, IP, NAT, TCP, UDP, DNS, routage
- TP : approfondissement de certains points vus en cours

- Évaluation avec 2 notes :
  - 1 écrit « surprise », 50% de la moyenne
  - 1 écrit en fin de module, 50% de la moyenne

I - Rappels

Vues en couches

- Composants réseau d'un équipement « standard »

Application engendre / contrôle échanges

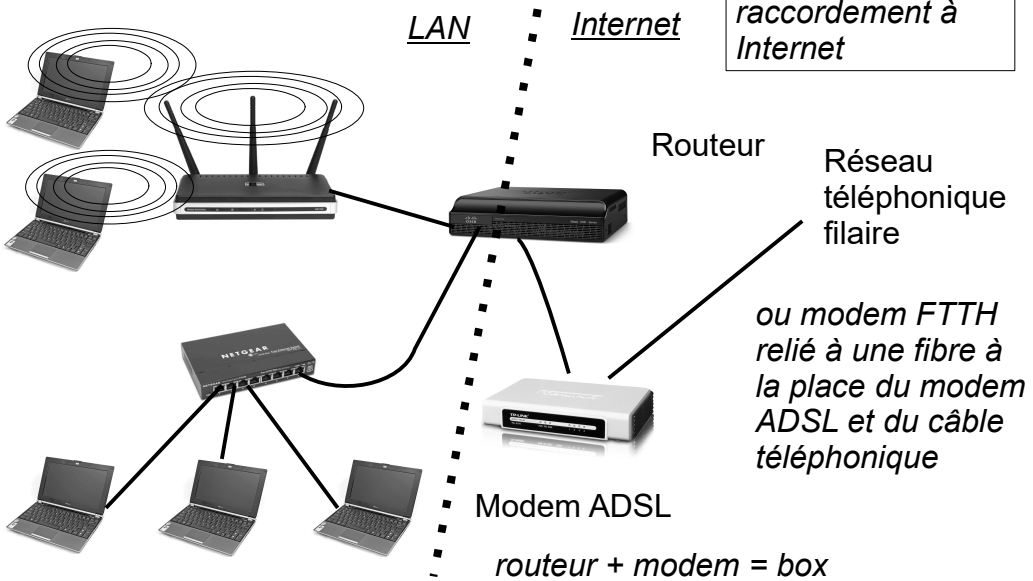
Application utilise TCP ou UDP, qui utilise IP, qui utilise une interface réseau (WiFi, ...)

	OSI	TCP/IP	pratique
7	Application	Application	Application
6	Présentation		
5	Session		
4	Transport	Transport (TCP, UDP)	« protocoles »
3	Réseau	Réseau (IP)	
2	Liaison de données	Interface réseau	Interface réseau
1	Physique		

émission *réception*

I - Rappels

Exemple typique de LAN avec raccordement à Internet

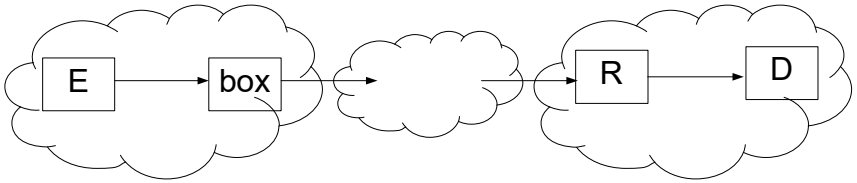


II – Couche réseau – Objectifs

- Exemple introductif :
  - soit 2 machines, E (expéditeur) et D (destinataire)
  - E est utilisée par une personne cliquant sur un lien Web
  - D correspond au serveur Web à contacter
- Le clic va engendrer un paquet
- Trajet du paquet : E vers box (par WiFi par ex), box vers FAI (par ADSL par ex), FAI vers opérateur gérant D et enfin arriver en D
- A chaque étape, une trame différente (car dépend de la technologie), sauf pour la partie « data » qui correspond au paquet

II – Couche réseau – Objectifs

- L'interface réseau gère les échanges de proximité (de E vers la box en WiFi par exemple)
- Tandis que la couche réseau gère le transfert longue distance de E jusqu'à D, en dirigeant le paquet d'un réseau à un autre, mais sans se soucier des échanges de proximité



II – Couche réseau – Objectifs

- Chaque tronçon (E à box, box à FAI, etc) est un réseau indépendant (point de vue admin)
- Ces réseaux sont reliés entre eux par des machines spéciales appelées routeur (R sur le schéma)

- Note : une box est aussi un routeur (entre autres choses)

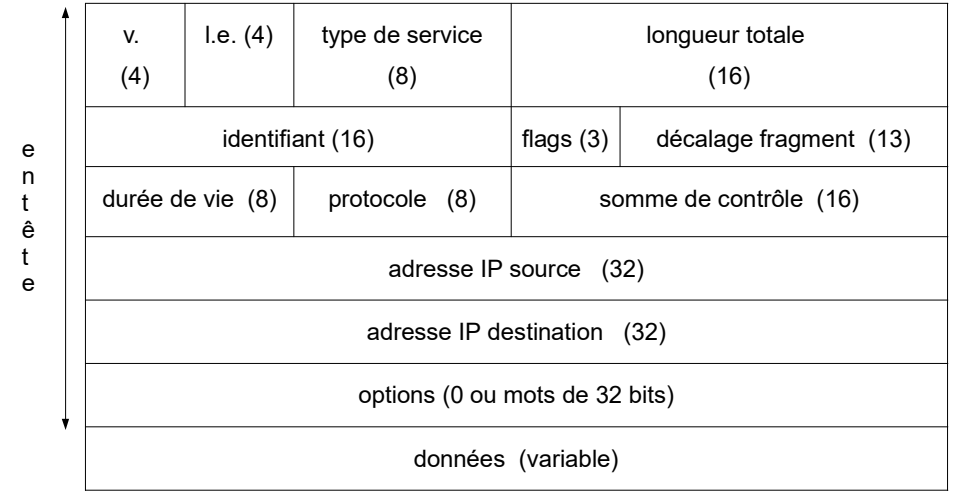
II – Couche réseau – Objectifs

- Au final, la couche réseau :
  - identifie toutes les machines (interfaces réseaux) indépendamment des aspects matériels, des technologies. Cet identifiant est une adresse logique (car non définitive, donnée a posteriori en fonction de choix d'administration)
  - place les données à traiter (issues de la couche transport et donc indirectement des applications) dans des paquets
  - fait circuler les paquets de E jusqu'à D en passant par tous les routeurs intermédiaires nécessaires (= fonction routage)

II – Couche réseau – IP

- IP (Internet Protocol) est THE protocole réseau dans Internet (RFC 791, ...)
- Philosophie : « best effort delivery » = faire de son mieux pour transmettre ; aucune garantie, fiabilité faible
- Principales caractéristiques :
  - Adressage logique = adresses IP
  - Gère la fragmentation (découpage) : chaque segment ou datagramme est inséré dans un ou plusieurs paquets IP suivant la quantité de données à transmettre
  - Gère le routage statique (cf plus loin)

II – Couche réseau – IP - format d'un paquet



(..) : taille du champ en bit

II – Couche réseau – IP – format d'un paquet

- v. : version d'IP. Ici v. = 4
- l. e. : longueur de l'entête seule, en multiples de 4 octets. Sans option, l'entête contient 20 octets => l. e. = 20 / 4 = 5
- type de service (TOS) : options de qualité de service (QoS) et d'acheminement. Par défaut, TOS = 0
- longueur totale : taille du paquet (quantité de données + entête), en octets => la taille max. d'un paquet est 2^16 octets
- identifiant : n° du paquet donné par l'expéditeur (compteur de paquets)

II – Couche réseau – IP – format d'un paquet

- Pour la fragmentation : flags et décalage fragment (cf plus loin)
- flags (drapeaux) de 1 bit chacun : 0 | DF | MF, avec :
  - DF (Don't Fragment) : DF = 0 pour autoriser la fragmentation (cas le plus courant)
  - MF (More Fragment) : MF = 1 si la fragmentation est nécessaire et il reste encore des fragments. MF = 0 pour le dernier fragment ou si la fragmentation n'est pas nécessaire
- décalage fragment : localise le fragment courant par rapport à l'ensemble ; multiples de 8
- durée de vie (TTL = Time-To-Live) : champ initialisé par l'émetteur et décrémenté par chaque routeur rencontré et à chaque seconde passée dans un routeur

II – Couche réseau – IP – format d'un paquet

- protocole : identifiant (RFC 1700) permettant de reconnaître le contenu du champ données. Exemples : 6 pour TCP, 17 pour UDP
- somme de contrôle : pour le contrôle de l'entête. Ce contrôle est effectué à chaque passage par un routeur et par le destinataire final
- adresse IP source : adresse qui identifie la machine (l'interface) émettrice du paquet
- adresse IP destination : adresse qui identifie la machine (l'interface) destinataire du paquet
- Ces adresses peuvent être modifiées en cours de route

II – Couche réseau – IP – format d'un paquet

- options : chaque option doit occuper un multiple de 4 octets. Exemples (non vus en détail) :
  - routage imposé, enregistrement de la route (pour BGP)
  - estampille horaire
- données : nature des données transportées, du type indiqué dans le champ « protocole »

II – Couche réseau – IP – Fragmentation et MTU

- Soit D les données à transporter (contenu du champ données), représentées sous forme de tableau D[i], i = 0 → N-1
- La fragmentation consiste à découper D, si besoin, en plusieurs morceaux afin de transporter une quantité supérieure aux limites d'IP (et même des trames)
- IP fait de la fragmentation non transparente => le découpage est fait par l'expéditeur et D n'est reconstitué que par le destinataire
- MTU (Maximum Transmission Unit) = la taille max d'un paquet (en octet)
- IP prend le plus petit MTU proposé par chaque machine du réseau, celui-ci étant basé sur les caractéristiques des interfaces réseau (MTU = 1500 pour Ethernet, 2300 pour WiFi)

II – Couche réseau – IP – Fragmentation et MTU

- Si (N + entête IP) ≤ MTU, pas de fragmentation => D est mis dans un seul paquet et MF = 0, « décalage fragment » = 0
- Si (N + entête IP) > MTU, la fragmentation est nécessaire. Il y aura plusieurs paquets IP, qu'on appelle fragments, construits avec les règles suivantes :
  - chaque fragment ne peut contenir que Dmax octets

    Dmax = E[(MTU - entête IP)/8]x8 (E[x] = partie entière de X)
  - nombre de paquets = N/Dmax (arrondi à l'entier supérieur)

    (suite après)

II – Couche réseau – IP – Fragmentation et MTU

- Règles pour la fragmentation (suite) :
  - Construction du j-ième fragment (j commence à 0) :
    - contient (sauf le dernier) D[j*Dmax] à D[(j+1)*Dmax-1]
    - le dernier fragment contient le reste des données
    - « décalage fragment » = j*Dmax/8
  - MF = 1 , sauf pour le dernier fragment (MF = 0)
  - Tous les autres champs se comportent identiquement avec ou sans fragmentation. Ils sont constants pour tous les fragments (sauf « somme de contrôle »)
- Si un fragment n'est pas reçu, il faut renvoyer tous les fragments !

II – Couche réseau – IP – Fragmentation et MTU

- Exemple
  - Hypothèses : D contient 2000 octets (N) ; MTU = 1495 ; pas d'option (entête IP de 20 octets)
  - (N + entête IP) > MTU => fragmentation nécessaire
  - Dmax = E[(1495 – 20)/8]x8 = 1472
  - 2000 / 1472 = 1,36 => 2 fragments
  - 1er fragment : contient D[0] à D[1471] dans le champ données ; « décalage fragment » = 0 ; MF = 1 (reste encore des fragments)
  - 2ème fragment : contient D[1472] à D[1999] dans le champ données ; « décalage fragment » = 1472/8 = 184 ; MF = 0 (dernier fragment)

II – Couche réseau – IP – échanges

- IP est basique de ce point de vue : pas de préambule, d'acquittement, etc
- => le diagramme de séquences est constitué d'une flèche par paquet IP (ou par fragment si la fragmentation a été nécessaire)

II – Quelques compléments à IP

- Quelques défauts de IP :
  - IP est non fiable et en plus sans avertissement. Solution : ICMP
  - Habituellement, le FAI ne fournit qu'une adresse IP valide pour tout le LAN, adresse donnée au routeur (box) => comment les autres machines peuvent échanger avec Internet ? Solution : NAT

II – ICMP

- ICMP (Internet Control Message Protocol, RFC 792)
- Protocole pour échanger des messages codés entre différentes machines (routeurs ou standards) afin de contrôler IP, avertir en cas de problème, etc
- Chaque message ICMP est intégré dans le champ « données » de IP (champ « protocole » dans l'en-tête IP = 1)

II – ICMP

- Un message ICMP est constitué de 2 nombres : le « type » et le « code »
- Ex. :
  - 3 | 1 <=> « dest. inaccessible »

    Message envoyé à l'exp. par un routeur lorsque l'adresse IP fournie n'est pas trouvée
- « ping » est une commande qui utilise ICMP. Elle envoie (4 fois pour Windows, indéfiniment pour Linux) un message ICMP « 8 | 0 » et reçoit « 0 | 0 » en réponse

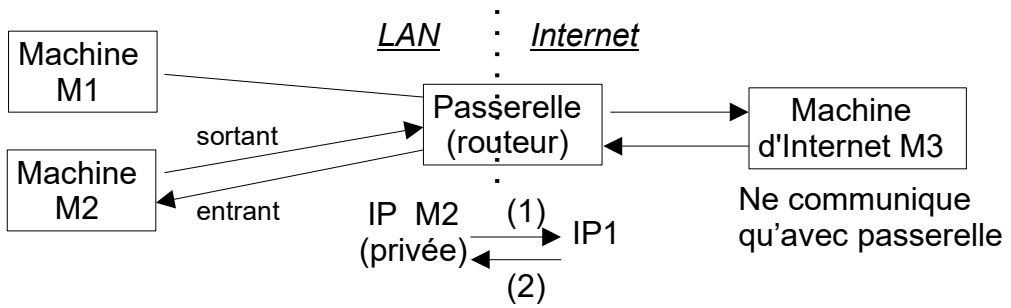
II – NAT

- NAT (Network Address Translation) = Translation d'adresses
- aussi appelé IP masquerading
- RFC 2663 et 3022
- NAT n'est pas un protocole (aucun échange entre machines), mais un mécanisme implanté dans les routeurs (passerelles)
- NAT a été créé pour solutionner le problème des adresses IP « privées » utilisées dans les LAN : adresses IP privées => pas valides sur Internet => normalement, pas de communication possible avec les autres machines d'Internet en dehors du LAN

II – NAT

- La passerelle du LAN, côté FAI, a une adresse IP valide, fournie par le FAI = IP1
- Lorsqu'une machine du LAN envoie un paquet vers l'ext (paquet sortant), NAT change dans le paquet l'IP présente (privée) par IP1
- Donc, vu de l'extérieur, le LAN ne contient qu'une seule machine, qu'une seule adresse IP : IP1
- Pour le sens inverse (paquet entrant), NAT se sert d'une table
- Cette table contient, pour chaque paquet sortant, l'IP privée ET le n° de port
- Les paquets entrants reçus par la passerelle contiennent tous IP1 => distinction du vrai destinataire uniquement par n° port
- Exemple diapo suivante

II – NAT



Infos enregistrées par passerelle en (1)

Paquet reçu de M3 : port dest. = 50012 => pour M2 => (2)

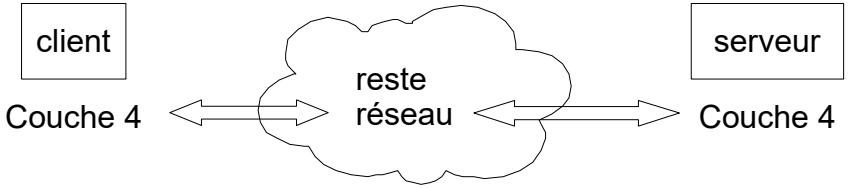
Id paquet	IP source	Port source	IP dest	Port dest
#1210	M2	50012	M3	80
etc				

II – NAT

- Principaux problèmes liés aux adresses IP privées et pour lesquels NAT ne peut rien :
  - applications utilisant une adresse IP comme identifiant, car IP non valide
  - machines utilisant IPSec (version sécurisée de IP), car aucun changement dans les paquets autorisé
  - serveurs dans un LAN, car IP non valide
  - …

III – Couche 4 - Intro

- Rôle : fournir aux applications un mécanisme de gestion du transport de leurs données :
  - de bout-en-bout, c.-à-d. directement entre l'émetteur et le destinataire final (client ou serveur par ex)
  - transparent, c.-à-d. indépendant du réseau physique
  - et qui respecte une qualité de service prédéfinie



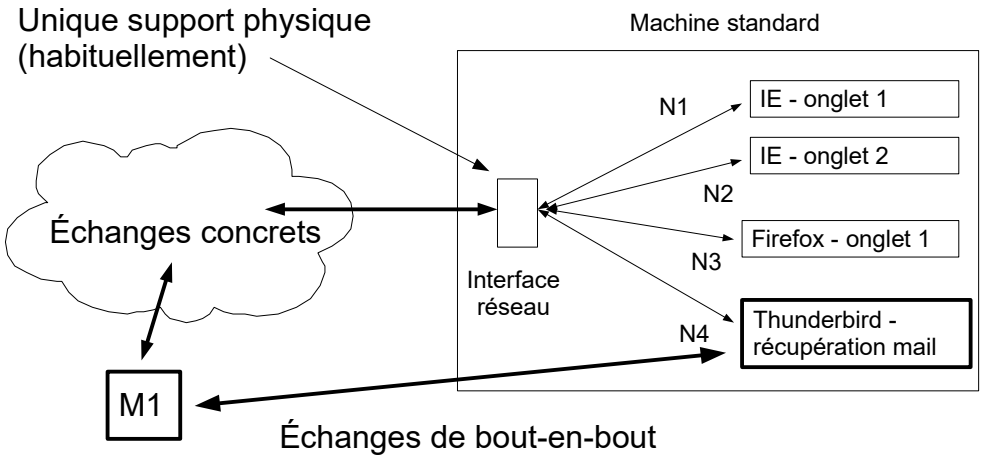
III – Couche 4 - Intro

- La qualité de service (en anglais, QoS : Quality of Service) peut s'exprimer à partir de nombreux critères. Les principaux :
  - la rapidité : débit, temps d'acheminement (= vitesse de transfert), ...
  - la fiabilité : contrôle des échanges, détection / correction des erreurs, ...
- Remarque : ces 2 critères sont antinomiques. Améliorer la fiabilité ralentit les échanges

III – Numéro de port

- Les machines utilisant les applications sont généralement des machines « standards » => 1 seule interface réseau
- => tous les flux de données passent par le même support
- La couche 4 est chargée de gérer un identifiant permettant de les différencier
- Plus précisément, cet identifiant distingue chaque processus réseau
- Dans Internet, il est appelé numéro de port (un entier de 16 bits), un port étant une zone mémoire utilisée comme tampon d'E/S

III – Numéro de port



III – Numéro de port

- Sur Internet, chaque protocole de couche 4 dispose sur chaque machine de 2^16 (env. 65000) ports différents
- Ils se divisent en 3 catégories (voir IANA ; http://www.iana.org/assignments/port-numbers) :
  - « bien connus » (well-known) : de 0 à 1023
  - Répertoriés (registered) : de 1024 à 49151
  - Dynamiques et/ou privés : de 49152 à 65535
- Les n° de port « bien connus » sont réservés aux serveurs des applications classiques d'Internet : FTP (20 et 21), TELNET (23), SMTP (25), HTTP (80), POP3 (110), ...

III – Numéro de port

- Les n° de port répertoriés sont réservés aux serveurs d'applications d'Internet standard mais soit moins classiques, soit plus récentes : NFS (2049), XMPP (Jabber ; 5269), VNC (Virtual Network Computing, prise contrôle à distance ; 5900), HPJET (9100), …
- Les n° de port privés sont laissés aux serveurs des applications non standards
- Les n° de port dynamiques (les mêmes que privés) sont attribués automatiquement par les O.S. (parmi ceux disponibles) au démarrage d'un client, quelque soit l'application
- ! chaque instance d'application a un numéro de port différent. Par ex., 2 onglets de navigateur = 2 numéros de port différents

III – Numéro de port

- Pourquoi seuls les serveurs ont un numéro de port fixés ?
- D'après le modèle d'application C/S (le plus utilisé), les clients font le 1er envoi. Ils doivent connaître à l'avance les coordonnées (@ IP + n° de port) des serveurs pour les contacter. Ces coordonnées doivent donc être fixes
- Les serveurs, eux, connaissent les coordonnées des clients dès la réception de la 1ère requête (infos dans les paquets IP)

III – TCP

- TCP = Transmission Control Protocol (RFC 793, 1122 et 1323, ...)
- TCP est le principal protocole de couche 4 dans Internet
- Il offre, aux applications l'utilisant, un service avec connexion et accusé de réception, donc fiable (mais pas à 100%)

- Un accusé de réception ou acquittement (= ACK pour « acknowledge » en anglais) permet de valider un envoi
- Il est émis par le récepteur vers l'émetteur, en faisant référence à ce qu'il valide
- Il n'y a pas d'ACK d'ACK

III – TCP

- Il existe 2 types de connexion : physique et logique
- Connexion physique : indispensable ; relie physiquement les machines entre elles = interfaces réseaux + support physique
- Connexion logique : non obligatoire ; est un lien virtuel (de durée finie) entre plusieurs entités échangeant des informations
- Connexion logique sert à fiabiliser les échanges en :
  - préparant les échanges : s'assurer que les processus sont prêts avant le début des échanges, entente sur divers paramètres (débit, taille, …), etc
  - assurant le suivi des échanges : identification des interlocuteurs, numérotation des échanges, etc

III – TCP

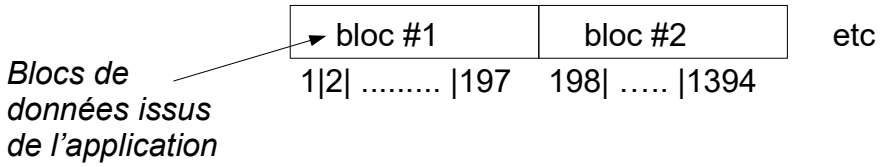
- Une connexion TCP est bidirectionnelle et point-à-point :
  - La connexion permet des échanges dans les 2 sens (sinon, il faudrait 2 connexions différentes)
  - Les échanges ne sont possibles qu'entre les 2 mêmes processus réseaux durant toute la durée de la connexion
  - Rappel : TCP utilise le numéro de port pour identifier le processus sur une machine donnée (identifiée par IP) => numéro de port + adresse IP permet d'identifier de manière unique un processus réseau sur une machine particulière (numéro de port + adresse IP = coordonnées réseau)

III – TCP

- Format d'un segment TCP (un segment par envoi) *en-tête*

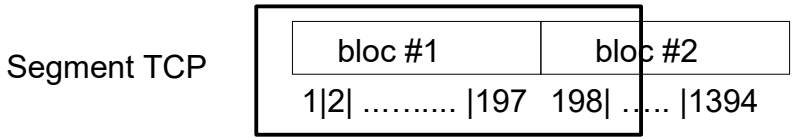
<table>
<tr><td colspan="8">Port source (16 bits)</td><td colspan="8">Port destination (16 bits)</td></tr>
<tr><td colspan="16">Numéro de séquence (SEQ) (32 bits)</td></tr>
<tr><td colspan="16">Numéro d'acquittement (ACK) (32 bits)</td></tr>
<tr><td colspan="4">Longueur en-tête (4 bits)</td><td colspan="2">Rés. (6 bits)</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="8">Taille de fenêtre (WIN) (16 bits)</td></tr>
<tr><td colspan="8">Total de contrôle (16 bits)</td><td colspan="8">Pointeur d'urgence (16 bits)</td></tr>
<tr><td colspan="16">Options (n x 32 bits)</td></tr>
<tr><td colspan="16">Données (taille variable)</td></tr>
</table>

- Port source : numéro de port de l'expéditeur du segment
- Port destination : numéro de port du destinataire du segment
- SEQ : contient le n° du 1er octet dans « données »
- En effet, TCP ne voit qu'un flux indifférencié d'octets (octet stream) => TCP « dissous » les blocs et numérote chaque octet provenant du même n° port depuis le début des échanges



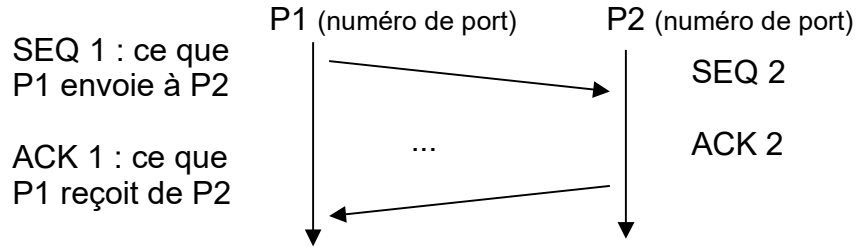
- Si aucune donnée envoyée (segment vide), SEQ est incrémenté de 1 quand même
- SEQ ne correspond pas nécessairement au n° du 1er octet d'un bloc ! Cf. diapo suivante
- Optimisations à l'envoi :
  - TCP attend un peu (200 ms) avant de passer à IP
  - les acquittements sont envoyés en même temps que des données (très souvent). Aucun impact sur SEQ mais sur le nombre d'échanges (cf. plus loin)

- Optimisations à l'envoi (suite) :
  - Si la taille des blocs est faible, TCP concatène i.e. mets plusieurs blocs disponibles (entiers ou en parties) dans « données » du même segment (extrêmement rare)



  - Si au contraire la taille d'un bloc est trop importante => fragmentation i.e. le bloc est réparti sur plusieurs segments

- ACK : indique le n° du prochain octet à recevoir en provenance de l'interlocuteur. Autrement dit, valide / acquitte tous les octets reçus avec numéro < ACK
- ATTENTION ! Chaque entité TCP gère son SEQ et son ACK



- ACK 1 est lié à SEQ 2 et ACK 2 est lié à SEQ 1

  ACK i = SEQ j + qté données j -> i

III – TCP

- Longueur en-tête : taille totale de l'en-tête, y compris les options, en multiple de 4 octets. Si pas d'option, taille = 20 octets => Longueur en-tête = 5
- Drapeaux :
  - ACK = 1 : valide le champ « ACK » ; ACK = 0 => pas d'acquittement à faire et « ACK » contient 0
  - SYN = 1 pendant l'ouverture de connexion ; 0 sinon
  - FIN = 1 enclenche la fermeture de connexion ; 0 sinon
  - Autres drapeaux pas vus

III – TCP

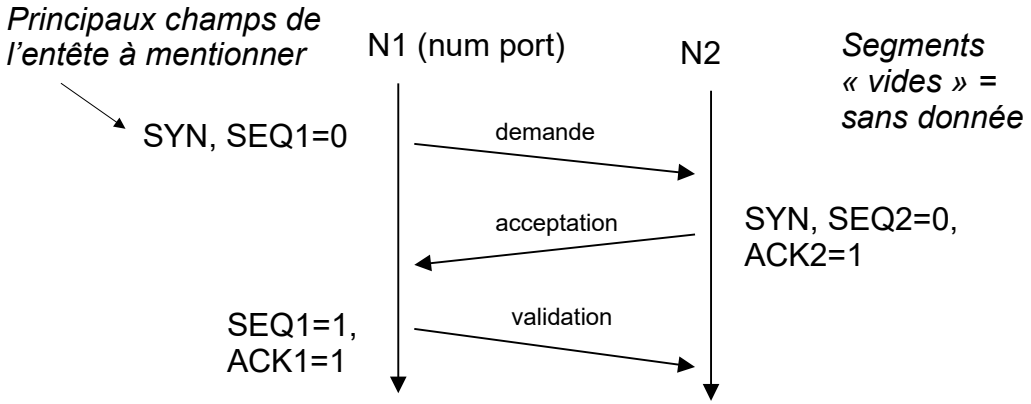
- WIN : quantité max de données dans un segment, en octets = seuil de fragmentation. Pour simplifier, constante. But : éviter des saturations (machines « lentes »)
- Total de contrôle : pour détecter une erreur dans le segment. Mode opératoire pas vu
- Pointeur d'urgence : pour le mode urgent (pas vu)
- Options : on verra MSS et NACK (voir plus loin)

III – TCP

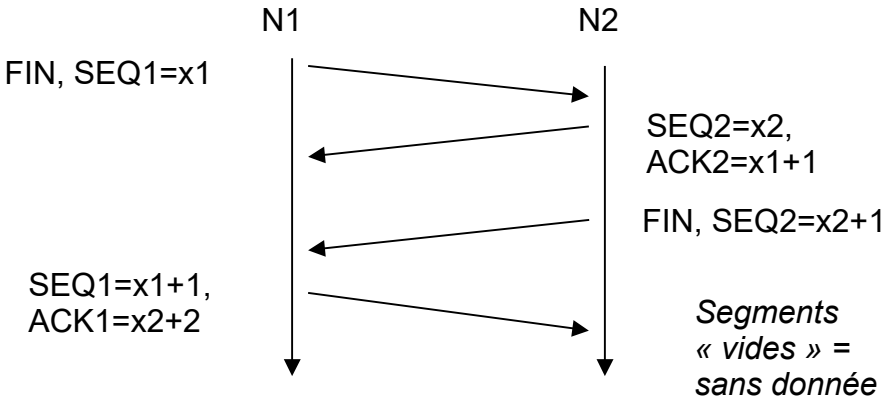
- Échanges en 3 phases (à cause de la connexion logique) :
  - ouverture (établissement) de la connexion. Négociations (de WIN, de l'utilisation d'options, …) pas vues
  - puis transfert de données
  - puis fermeture de la connexion, après la fin du transfert de données

III – TCP

- Ouverture de la connexion (diagramme de séquences)



- Fermeture de connexion : 2 semi-fermetures (pas d'ordre imposé)



- Règles normales (sans option) pour la phase « transfert de données » :
  - chaque segment envoyé contient au maximum WIN octets
  - pour chaque segment reçu par D (en provenance de E), D envoie un segment à E avec drapeau ACK = 1 et champ ACK rempli ; par abus de langage, le segment entier est appelé ACK
  - si possible, ACK contient des données D → E (cf optimisations) sinon ACK = segment sans données (vide)

- Avec option MSS (Maximum Segment Size), les règles pour la phase « transfert de données » changent ainsi :
  - MSS (constante) est toujours un sous multiple de WIN (par ex. MSS = WIN / 2)
  - chaque segment contient jusqu'à MSS octets (donc < WIN)
  - un ACK est envoyé après réception de WIN octets donc tous les WIN / MSS segments
  - même optimisation que règle normale

- Traitements normaux des problèmes (toute phase, sans option) :
  - Si non réception d'un segment : TCP gère une temporisation (appelée TIMEOUT) pour chaque segment envoyé. Si la tempo est dépassée avant réception du « ACK » correspondant, l'expéditeur (N1 ici) retransmet le même segment (petit nombre d'essais)