Services et transport de paquets vers les applications : les modèles de référence OSI et TCP/IP

Description du thème

Propriétés	Description
Intitulé long	Services et transport de paquets vers les applications : les modèles de référence OSI et TCP/IP
Formation(s) concernée(s)	BTS Services Informatiques aux Organisations
Matière(s)	SI2
Présentation	Les mécanismes abordés dans les séances précédentes (protocole Ethernet, résolution ARP, adressage IP, routage…) permettent à deux postes de communiquer sur un même réseau ou deux réseaux éloignés. Les applications utilisent l'infrastructure réseau pour rendre service à un utilisateur. Mais entre le réseau et les applications, il existe d'autres protocoles facilitant l'utilisation du réseau par les applications et prenant en charge les problèmes survenant sur les réseaux. **L'objectif de ce thème est de comprendre :** • comment s'interfacent le réseau et les applications et quels sont les protocoles en jeu ; • comment est géré un problème sur le réseau entraînant une perte de paquets ou de trames. Cette séance décrit la suite des protocoles en jeu dans une relation client/serveur et explique comment les modèles TCP/IP et OSI sont utilisés pour faciliter d'une part la normalisation dans le processus de communication et d'autre part le processus de résolution de pannes. C'est aussi l'occasion d'approfondir les notions de trame, de paquet, de segment, d'en-tête et d'encapsulation.
Savoirs	**Savoir-faire** • Exploiter un service de base • Analyser des unités de données de protocole **Savoirs associés** • Modèles de référence associés aux architectures réseaux • Services de base et unités de données de protocole associées
Compétences	
Transversalité	
Prérequis	Adressage IP, principes généraux de la commutation, notions de client/serveur
Outils	PC avec lecture de vidéos possibles. *Les vidéos peuvent aussi être consultables à la maison dans le cadre d'une pédagogie de classe inversée.*
Mots-clés	TCP IP OSI applications client serveur service port http ftp smtp pop dhcp dns
Durée	6/8 heures
Auteur.e(s)	Apollonie Raffalli et David Duron
Version	v 1.0
Date de publication	Décembre 2017

Indications à destination des enseignants

Cette ressource comprend un cours réalisable en classe entière, d'un TP analysant la communication entre un client et un serveur FTP et un QCM de validation.

I La modélisation en couche

1 La nécessité et l'utilité d'un modèle ouvert

Les éditeurs informatiques avaient depuis longtemps leurs propres réseaux pour interconnecter leurs équipements, comme l'architecture SNA (System Network Architecture) chez IBM, ou DNA (Digital Network Architecture) chez Digital Equipment Corporation. Mais ces architectures ne permettaient pas d'interconnecter des matériels hétérogènes.

Aussi, afin d'éviter la multiplication des solutions d'interconnexion, l'ISO (International Standards Organisation) a développé dans les années 70/80 **un modèle de référence à 7 couches appelé modèle OSI** (Open Systems Interconnection). L'objectif est de fournir un cadre dans lequel concevoir **une suite de protocoles ouverts.** L'idée était que cet ensemble de protocoles serait utilisé pour développer un réseau international qui ne dépendrait pas de systèmes propriétaires.

En parallèle des travaux de l'OSI, le département de la défense américain, la DARPA (United States Department of Defense Advanced Research Projects Agency), créait le réseau ARPANET (Advanced Research Projects Agency Network) qui jetait les bases de l'Internet et du **modèle TCP/IP à 4 couches**. Du fait de la rapidité avec laquelle Internet basé sur TCP/IP a été adopté et de sa vitesse de développement, l'élaboration et l'acceptation de la suite de protocoles OSI sont restées à la traîne. Mais ce dernier reste le modèle de référence.

Le modèle de protocole TCP/IP (appelé aussi modèle Internet), largement utilisé dans les réseaux, définit quatre catégories de fonctions qui doivent intervenir pour que les communications aboutissent. L'architecture de la suite de protocoles TCP/IP suit la structure de ce modèle.

Les règles de fonctionnement sont issues de normes ouvertes définies au niveau mondial. Le groupe de travail IETF (Internet Engineering Task Force) est ainsi en charge de définir les spécifications techniques qui déterminent le fonctionnement du réseau.
En ce qui concerne l'IETF, ce travail technique, accompli dans une centaine de groupes, consiste à rédiger des **Request for comments (RFC[1])** disponibles au public qui contiennent notamment les spécifications formelles des protocoles de communication de données ainsi que des ressources qui décrivent l'utilisation des protocoles..

Dans les thèmes précédents, nous avons vu comment un réseau efficace peut être construit et quels protocoles de bas niveau sont employés pour transporter des données « brutes » sur le réseau local (**protocole Ethernet** par exemple) ou entre plusieurs réseaux locaux (**protocole IP** par exemple).

Mais ces protocoles ne sont pas suffisants pour assurer une bonne communication, par exemple :

- le protocole IP ne permet pas de savoir si le paquet envoyé est bien reçu ;
- il n'identifie qu'un hôte sur le réseau : utilisé seul, il n'identifie pas le service voulu au niveau de l'hôte ;
- il ne se préoccupe pas de savoir comment les différents services (comme le service Web) fonctionnent.

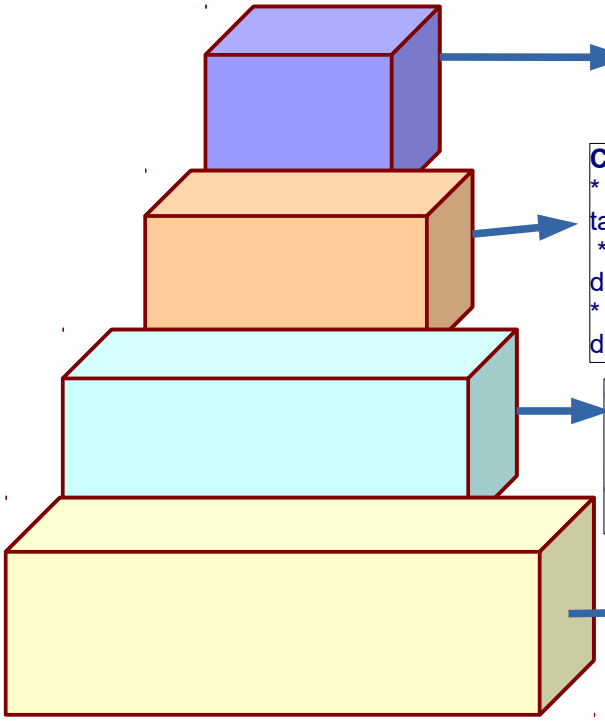
La suite de cette séance se propose de décrire et d'expliquer l'utilité de la suite de protocoles nécessaires pour qu'une application cliente puisse dialoguer avec l'application serveur correspondante.

1 **Les RFC**, littéralement « demande de commentaires », sont une série numérotée de documents officiels décrivant les aspects techniques d'Internet, ou de différents matériels informatiques (routeurs, serveur DHCP).

2 Le modèle TCP/IP

Regardez la vidéo https://www.youtube.com/watch?v=Y0W8aX_Ih78 (8mn 41).

Q1. Inscrivez dans les "cubes" quelques protocoles connus correspondants à la définition de chaque couche.



Couche 4 appelée **Application :**
* Construit les **données.**
* Définit le format des requêtes et des réponses échangées selon l'application.

Couche 3 appelée **Transport** :
* Découpe les données à envoyer en **segments** de petite taille.
* Permet de repérer précisément le service utilisé par le biais du numéro de port.
* Permet de transporter les données de la couche application de façon fiable ou non (selon le protocole utilisé).

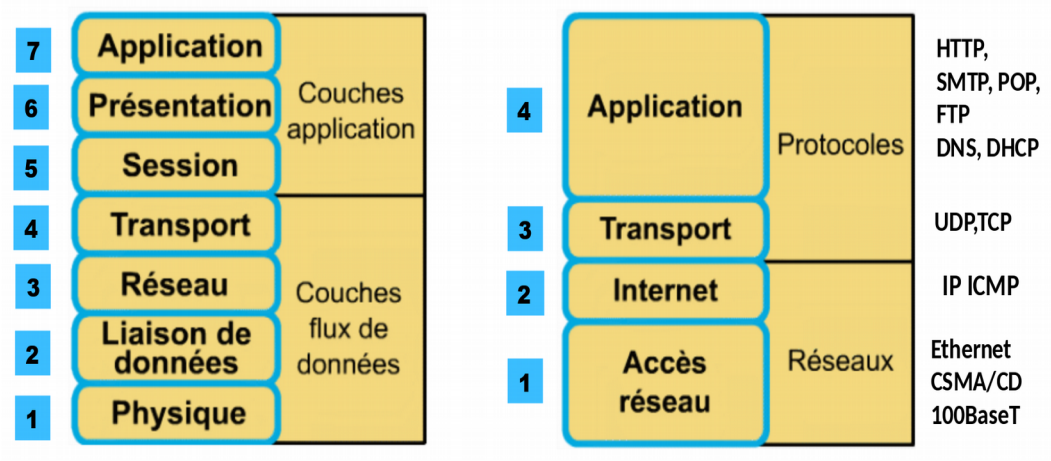
Couche 2 appelée **Interréseau ou Internet** :
* Permet de repérer l'hôte source et l'hôte destination par le biais d'adresses IP et de les ajouter au segment afin de construire le **paquet**.
* Permet de router les paquets entre les réseaux.

Couche 1 appelée **Accès réseau :**
* Permet de connaître les adresses physiques et de fabriquer la **trame**.
* Assure la bonne gestion du média.
* Permet l'acheminement des informations sous forme binaire entre émetteur et destinataire.

Q2. Quels sont les éléments matériels que l'on peut associer respectivement aux couches 1 et 2 du modèle TCP/IP ?

3 Modèle OSI versus modèle TCP/IP

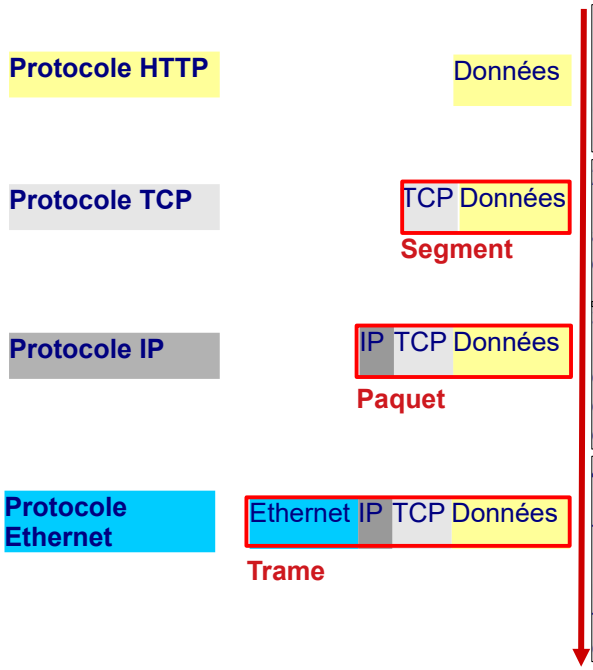
Contrairement au modèle TCP/IP, le modèle OSI ne spécifie l'interaction d'aucun protocole particulier. Il a été conçu comme l'architecture de référence pour l'élaboration de protocoles de communications réseau. Il se décline en 7 couches.



II Modèle et protocoles en couches : le processus d'encapsulation des données

Lorsque les **données de la couche application** descendent la pile de protocoles en vue de leur transmission sur le support réseau, les différents protocoles ajoutent des informations à chaque niveau.

Exemple : interaction entre un navigateur Web (client) et un serveur Web



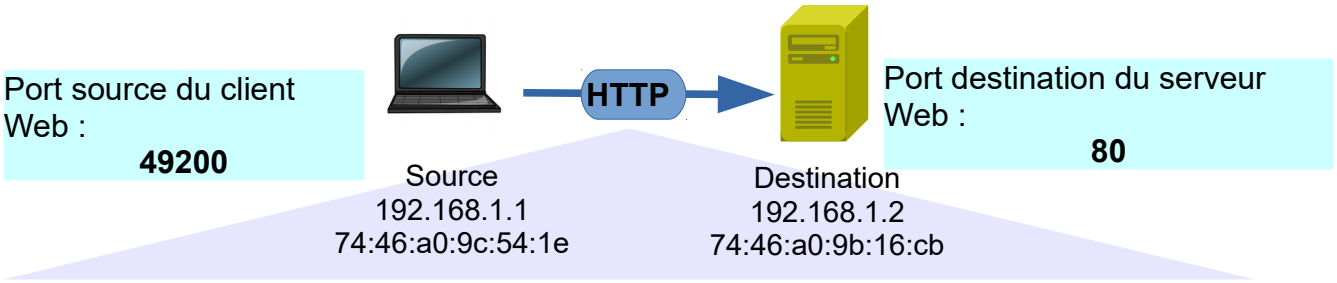
1. L'utilisateur lance son navigateur et saisit une URL : http://192.168.1.2/coucou.html. **Le navigateur passe l'ordre d'envoi d'une requête sur l'adresse IP à TCP** (voir les données envoyées dans l'analyse de trame en bas de la page).

2. Le protocole TCP divise les **données** en parties de plus petite taille, appelées **segments**, pour les envoyer au serveur. Il ajoute également aux données transmises des informations essentielles comme les **numéros de ports**.

3. Le **protocole IP** récupère les segments formatés à partir du protocole TCP, s'occupe de leur encapsulation en **paquets**, de l'affectation des adresses IP appropriées et de la sélection du meilleur chemin vers l'hôte de destination.

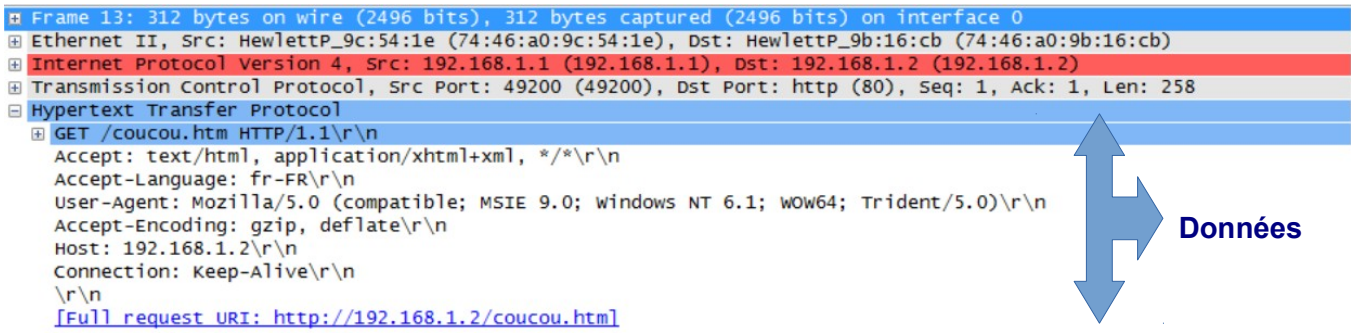
4. Les protocoles en jeu ici prennent les paquets depuis le protocole IP et les formatent pour les transmettre à travers les supports sous forme de **trames**. De même, les normes et les protocoles des supports physiques régissent la manière dont les signaux sont envoyés à travers les supports, ainsi que leur interprétation par les clients destinataires.

La **trame** arrive donc en bout de processus et intègre l'ensemble des données y compris les numéros de port ajoutés par la couche transport, les adresses IP ajoutées par la couche IP et les adresses MAC :

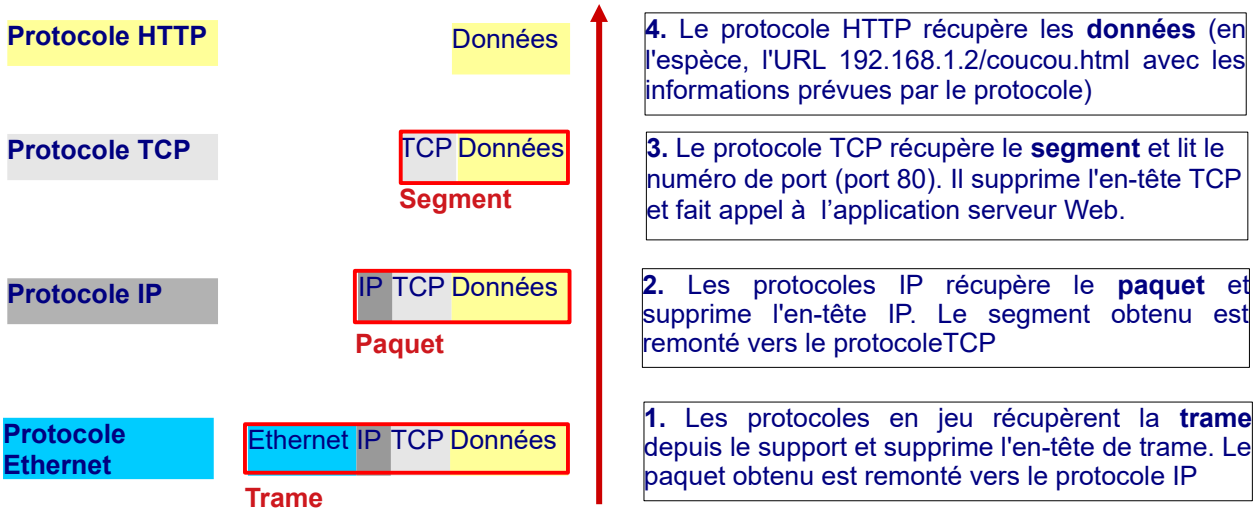


@MAC destination	@MAC source	IP source	IP destination	Port source	Port Dest
74:46:a0:9b:16:cb	74:46:a0:9c:54:1e	192.168.1.1	192.168.1.2	49200	80

On visualise, dans l'analyse de trame (Frame 13), les 4 couches (Ethernet, IP, TCP et HTTP) :



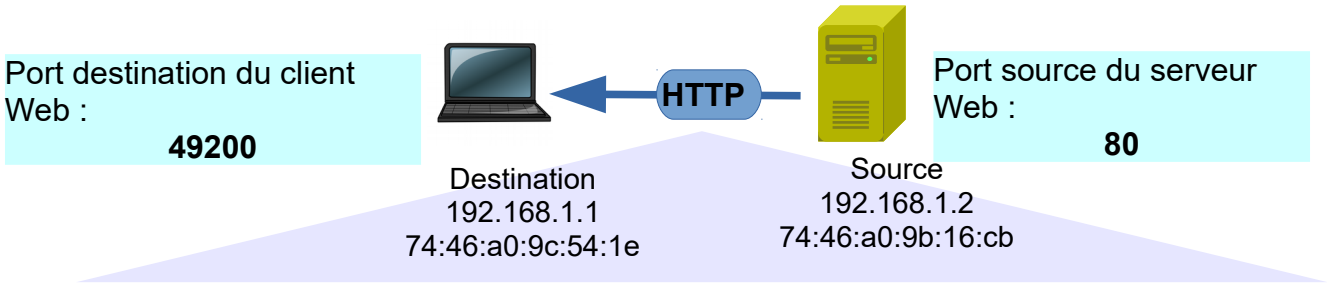
Lorsque la trame finale est récupérée par l'hôte de destination (après qu'éventuellement le paquet IP soit passé par quelques routeurs) un **processus inverse** est mis en œuvre.



Sur les 2 machines nous avons la même pile de protocoles, chacun jouant un rôle précis (simplifié dans les explications précédentes). Chaque protocole de niveau supérieur demande un service au protocole de niveau inférieur et fournit des informations au protocole de même niveau sur la machine distante.
L'ensemble de ce dialogue à distance entre protocoles est véhiculé par les trames Ethernet dans son champ données, plus précisément ce champ données contient les données IP, les données IP contiennent les données TCP, les données TCP contiennent les données HTTP, et les données HTTP contiennent les données (en l'espèce, l'URL). **Ce mécanisme de poupée russe s'appelle l'encapsulation.**

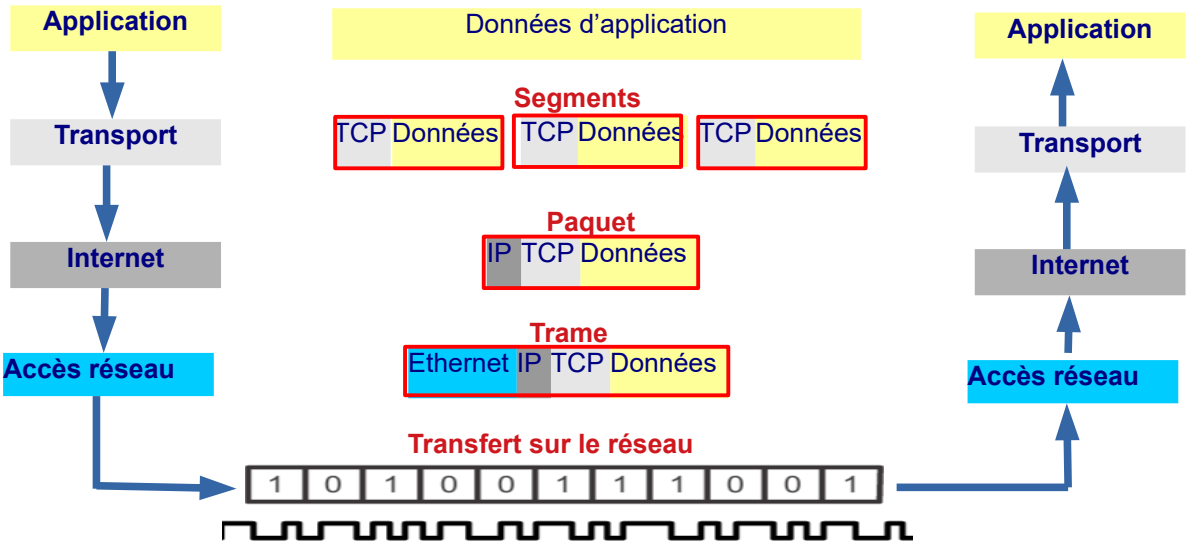
Une fois la requête reçue et interprétée, le serveur répond en envoyant la page demandée (ou un message d'erreur si, par exemple, la page n'existe pas).

Q3. Reconstituez la trame simplifiée de la réponse du serveur Web au client.



@MAC destination	@MAC source	IP source	IP destination	Port source	Port Dest

Récapitulatif : encapsulation/dés-encapsulation



Exercice

Observez l'analyse de trame ci-dessous.

```
    11 2.7782977… 192.168.0.90    80.12.242.10    SMTP          87 C: EHLO [192.168.0.90]

▸ Frame 11: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface 0
▸ Ethernet II, Src: RivetNet_d2:d4:13 (9c:b6:d0:d2:d4:13), Dst: Sagemcom_32:44:30 (68:15:90:32:44:30)
▸ Internet Protocol Version 4, Src: 192.168.0.90, Dst: 80.12.242.10
▸ Transmission Control Protocol, Src Port: 55990, Dst Port: 587, Seq: 1, Ack: 38, Len: 21
▾ Simple Mail Transfer Protocol
  ▸ Command Line: EHLO [192.168.0.90]\r\n
```

Q1. Quel est l'objet de cette communication ?

Q2. Complétez le tableau ci-dessous qui associe à chaque couche du modèle TCP/IP le protocole correspondant à la trame émise.

Couches du modèle TCP/IP	Protocole associé
4 - Application	
3 - Transport	
2 - Internet	
1 – Accès réseau	

Q3. Compléter la trame simplifiée correspondante à cette analyse de trame.

@MAC destination	@MAC source	IP source	IP destination	Port source	Port dest

Q4. En déduire la trame simplifiée de réponse à la trame précédente.

@MAC destination	@MAC source	IP source	IP destination	Port source	Port Dest

III Approfondissement de la couche transport

1 Limites du protocole IP (couche 2 – Internet du modèle TCP/IP)

IP est un protocole de bas niveau permettant l'interconnexion de réseau qui s'appuie sur la technologie des réseaux locaux (Ethernet). Mais le protocole IP n'offre aucune garantie sur la remise des paquets au destinataire et n'est pas utilisable directement par les applications orientées utilisateurs **==> on vient de le voir, il faut d'autres protocoles pour offrir des services utilisateurs et garantir la fiabilité du réseau.**

Les protocoles de transport définissent comment transmettre les messages entre les hôtes. Les deux protocoles de transport les plus courants sont **TCP (Transmission Control Protocol)** assimilé à une lettre recommandée avec accusé de réception et **UDP (User Datagram Protocol)** assimilé à une lettre ordinaire, qui n'utilise pas d'accusé de réception et achemine au mieux les datagrammes.

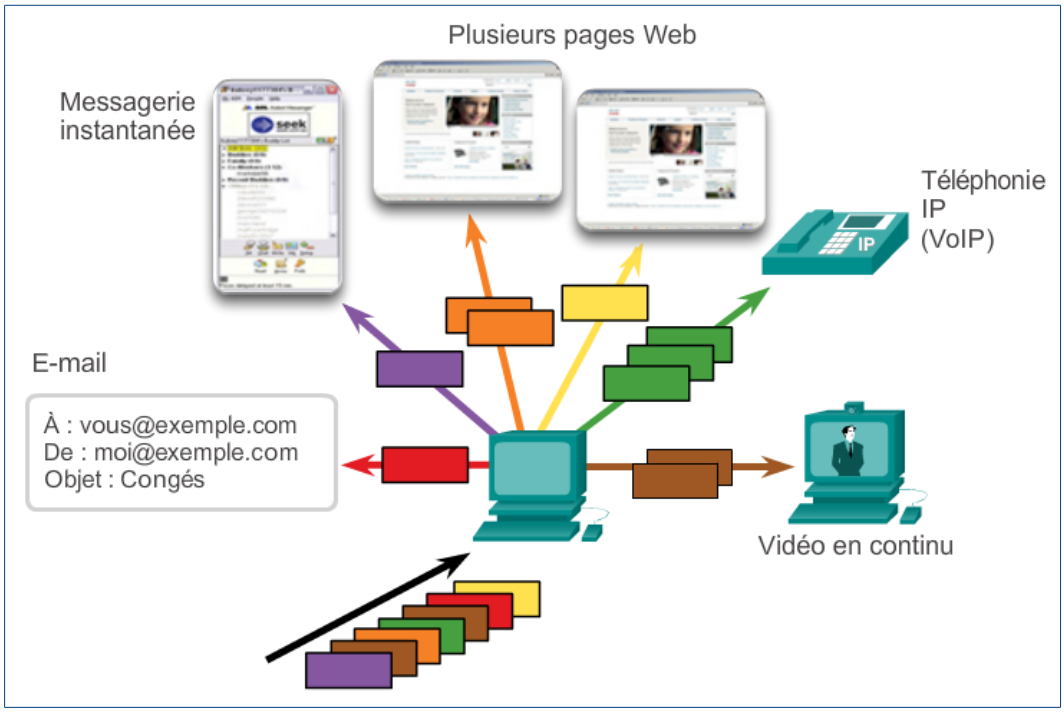
On peut assimiler IP à l'adresse postale écrite sur une enveloppe qui va déterminer non seulement le destinataire final mais aussi les différents relais postaux et TCP au fonctionnement de la poste qui est chargée de la transporter (remise avec ou sans accusé de réception, lettre suivie, Chronoposte pour les messages prioritaire, etc.).

2 Le protocole TCP (couche 3 – Transport du modèle TCP/IP)

Une application qui a besoin d'un accusé de réception, pour s'assurer que le message est bien transmis, utilise TCP. Ce processus est similaire à l'envoi d'une lettre recommandée par la poste, dont le destinataire accuse réception par sa signature.
FTP et HTTP sont des exemples d'applications utilisant TCP pour assurer la transmission des données.

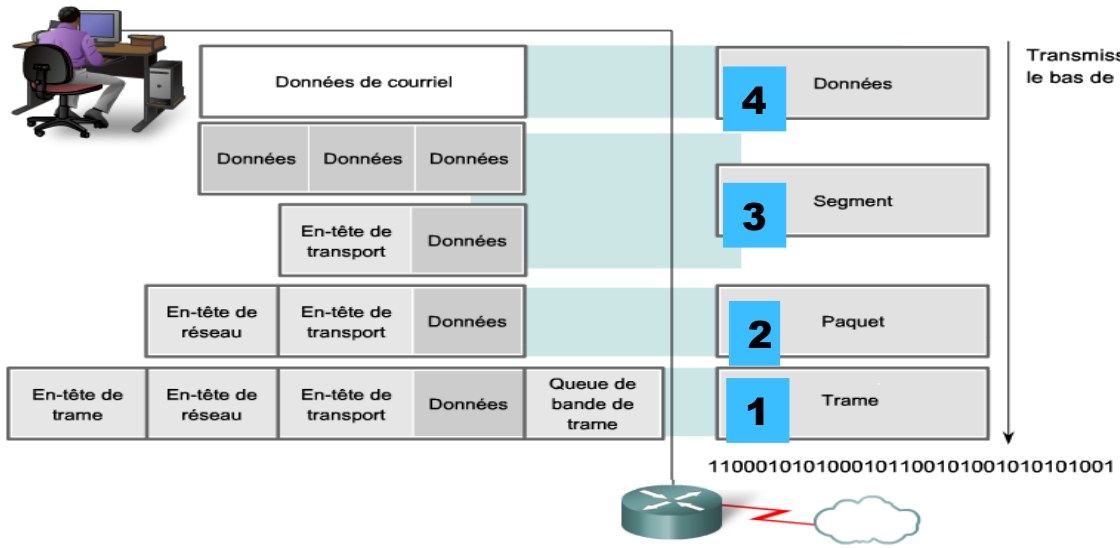
2.1 Fonctionnement

TCP découpe un message en petits morceaux appelés segments à partir des données fournies par les protocoles clients ou serveurs (HTTP par exemple). La taille de ces segments est déterminée par le MTU (Maximum Transfert Unit), taille maximum d'une trame échangée qui est de 1518 octets.



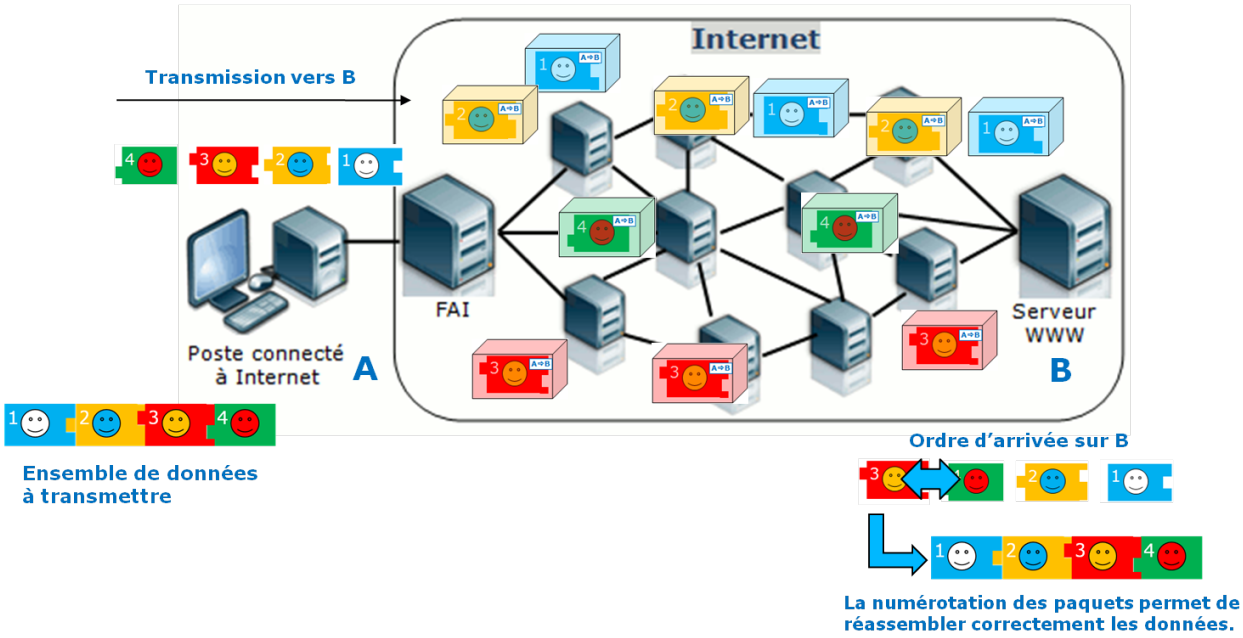
Un hôte peut héberger plusieurs applications qui communiquent sur le réseau simultanément. Chacune de ces applications communique avec une ou plusieurs applications sur un ou plusieurs hôtes distants. **La couche transport est aussi chargée d'identifier les différentes applications (via les numéros de port), de garantir ces multiples conversations et d'en effectuer le suivi.**

Les **segments** (couche 3) qui comprennent les numéros de port sont numérotés en séquence puis passés au processus IP (couche 2) où ces derniers sont encapsulés pour former des **paquets.** Ces derniers sont ensuite transmis à la couche Accès réseau pour constituer les **trames** qui seront transmises sur le réseau.



2.2 Transmission correcte avec TCP

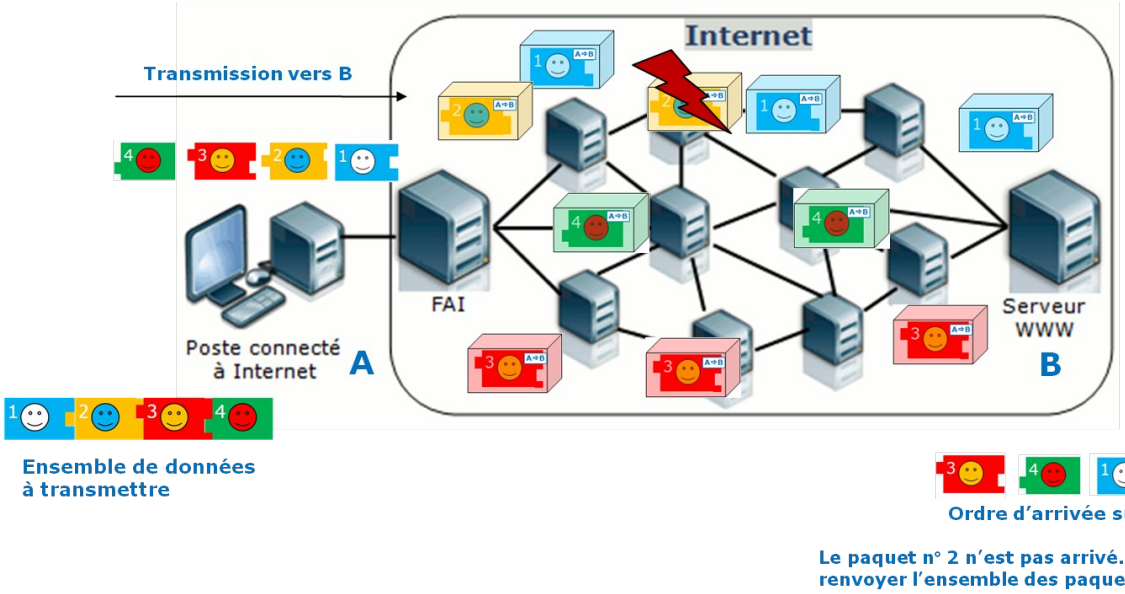
Les paquets n'empruntent pas forcément le même chemin, n'arrivent pas forcément dans le bon ordre, mais le réassemblage du « puzzle » est possible.



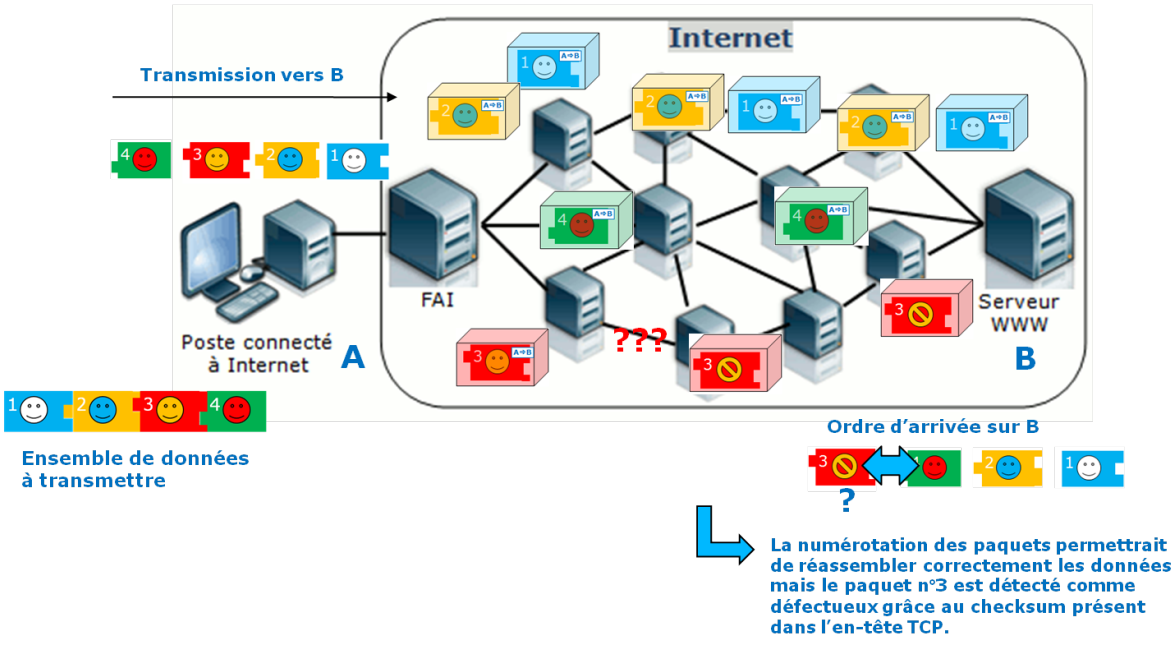
TCP conserve une trace du nombre de segments qui ont été envoyés à un hôte donné. **Si l'expéditeur ne reçoit pas d'accusé de réception au bout d'un certain temps, il suppose que certains segments ont été perdus, et il retransmet l'ensemble des segments depuis le dernier accusé de réception**. Seule la partie du message qui a été perdue est renvoyée, pas l'intégralité.

2.3 Transmissions incorrectes avec TCP

Si un paquet se perd, l'accusé de réception n'est pas envoyé par le destinataire, l'expéditeur renvoie les segments qu'il avait émis après le dernier accusé réception.



C'est le même processus si un paquet est corrompu.



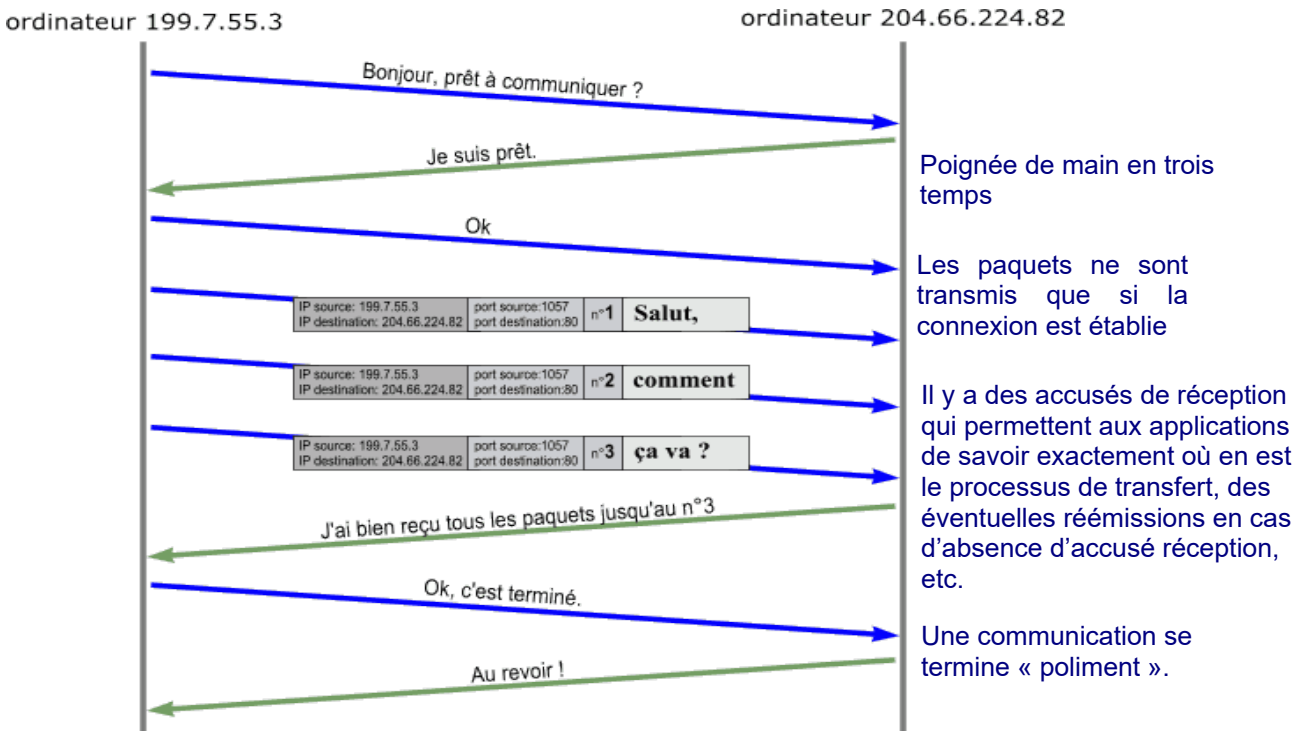
NB : Les accusés de réception ne sont pas envoyés après chaque paquet, mais tous les n paquets. La « fenêtre de réception » est négociée avec le client et peut varier d'une communication à l'autre, et même au cours d'un transfert (par exemple s'il y a beaucoup d'erreurs).
La « fenêtre » correspond au nombre d'octets que le récepteur souhaite recevoir sans accusé de réception.

Ci-dessus, on suppose une fenêtre correspondant à 4 paquets (≈ 6000 octets). Dans le dernier cas étudié, si la fenêtre était divisée par 2, la corruption du paquet 3 n'aurait entraîné la réémission que des paquets 3 et 4.

2.4 Protocole TCP et mode connecté : ouverture, utilisation et fermeture de la connexion

Dans ce mode, il se met en place un processus de poignée de main (handshake) entre le client et le serveur. Ce processus permet d'établir un dialogue à propos du transfert de données.

Ci-dessous, le schéma de principe :



Poignée de main en trois temps

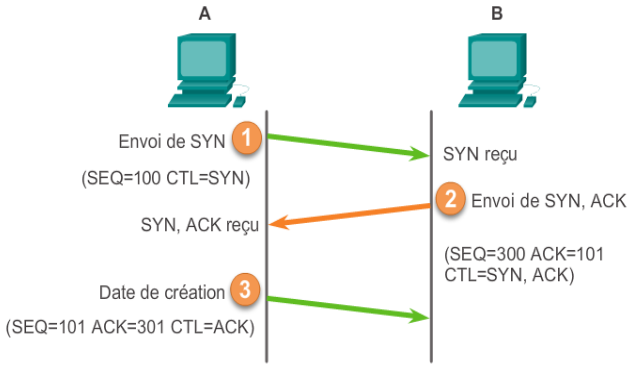
Les paquets ne sont transmis que si la connexion est établie

Il y a des accusés de réception qui permettent aux applications de savoir exactement où en est le processus de transfert, des éventuelles réémissions en cas d'absence d'accusé réception, etc.

Une communication se termine « poliment ».

Remarque : en cas de rupture de connexion, il y a abandon de la remise des paquets et signalement de l'abandon de la part du service TCP à l'application utilisatrice.

Utilisation des drapeaux (*flags*) lors de la poignée de main en trois temps.



Un flag (SYN, ACK, etc.) est généralement une valeur binaire informant sur l'état d'un objet.

6 bits sont consacrés aux flags dans l'en-tête TCP :

URG	ACK	PSH	RST	SYN	FIN

Exemples
ACK: si ce drapeau est à 1 le paquet est un accusé de réception.
SYN: Le Flag TCP SYN positionné à "1" indique une demande d'établissement de connexion.

```
14 6.202100   10.1.1.1        192.168.254.254   TCP   1069 > http [SYN] Seq=0 Len=0 MSS=
15 6.202513   192.168.254.254 10.1.1.1          TCP   http > 1069 [SYN, ACK] Seq=0 Ack=1
16 6.202543   10.1.1.1        192.168.254.254   TCP   1069 > http [ACK] Seq=1 Ack=1 Win=
```

Description du processus de « handshake » (poignée de main) :

Le client envoie une séquence de synchronisation avec un numéro de séquence (**X = 0** dans l'analyse de trame). Le flag "SYN" est positionné à 1.	Le serveur répond par une acceptation dans laquelle il renvoie : - un n° d'acquittement égal à X+1 - un n° de séquence (**Y= 0** dans l'analyse de trame) Les flag "SYN" et "ACK" sont positionnés à 1.	Le client acquitte la réponse en envoyant : - un n° d'acquittement égal à Y+1 - un n° de séquence égal au numéro d'acquittement envoyé par le serveur Le flag "ACK" est positionnéà1.

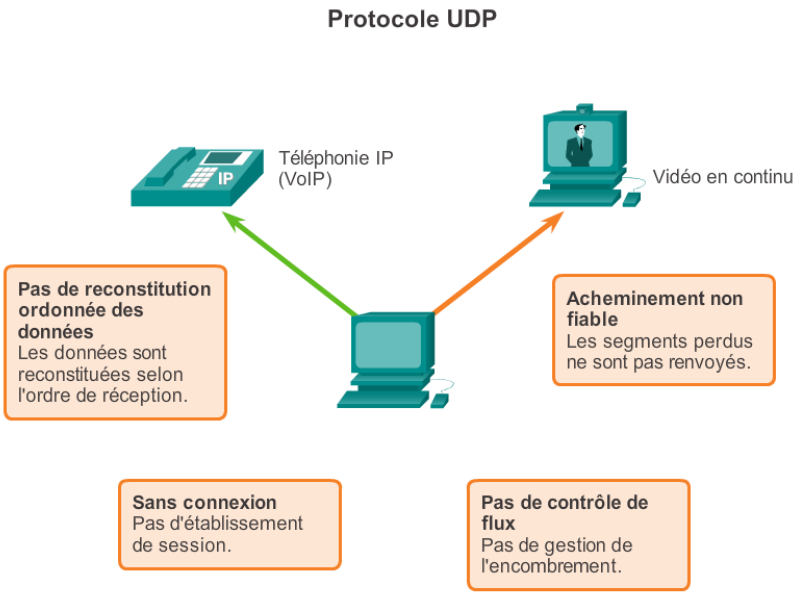
3 Le protocole UDP

Le protocole TCP est indispensable quand la communication ne peut se permettre de perdre des paquets. On en peut par exemple pas imaginer une page Web où il manquerait une lettre de temps à autre.

Mais dans certains cas, le protocole d'accusé de réception TCP n'est pas nécessaire. Il ralentit même le transfert des informations du fait notamment des accusés réceptions et des réémissions en cas de non accusé de réception. Dans un échange audio la perte de paquets ne se manifestera que par une altération de l'écoute. De toute façon, même si on renvoie le paquet, l'altération est perçue.

Dans ce cas là, UDP s'avère être un protocole de transport plus approprié.

UDP est un système d'acheminement « au mieux » qui ne nécessite pas d'accusé de réception. Ce processus est similaire à l'envoi d'une lettre ordinaire par la poste. La réception de la lettre n'est pas garantie, mais il y a de bonnes chances pour qu'elle parvienne à destination.



UDP fabrique les segments et les transmet à IP. Par contre **il n'établit pas de connexion préalable** et ne peut donc garantir une remise fiable ==> **c'est donc au niveau application que doivent être détectés les problèmes de transmission.**

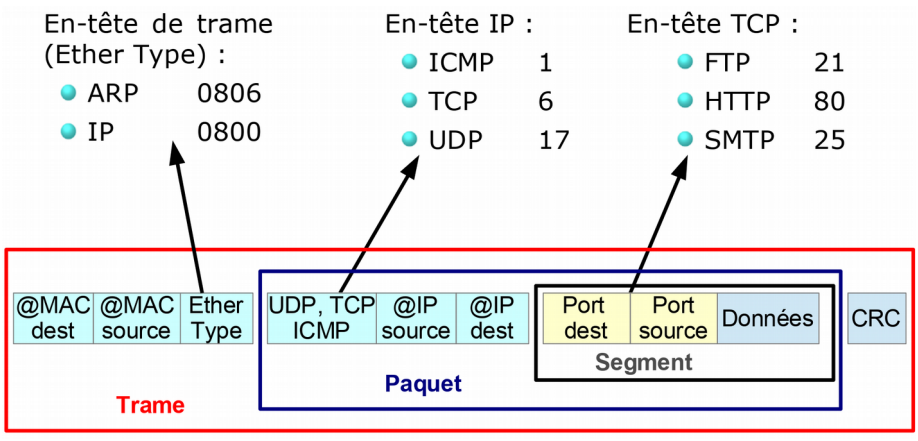
UDP est donc à préférer, notamment pour la lecture audio en continu, la vidéo et la voix sur IP (VoIP). Les accusés de réception ralentiraient la livraison, et les retransmissions ne sont pas souhaitables.

La webradio est un exemple d'application utilisant UDP. Si une partie du message est perdue pendant sa transmission via le réseau, elle n'est pas retransmise. Si certains paquets manquent, il se peut que la personne qui écoute entende de légères interruptions dans le son. Si TCP était utilisé et si les paquets perdus étaient renvoyés, la transmission serait interrompue pour recevoir ces paquets, et l'interruption se remarquerait davantage.

NB : les applications prennent souvent en charge ces problèmes (ou une partie de ces problèmes) pour qu'ils soient quasiment imperceptibles pour l'utilisateur.

4 Les identifiants de protocole

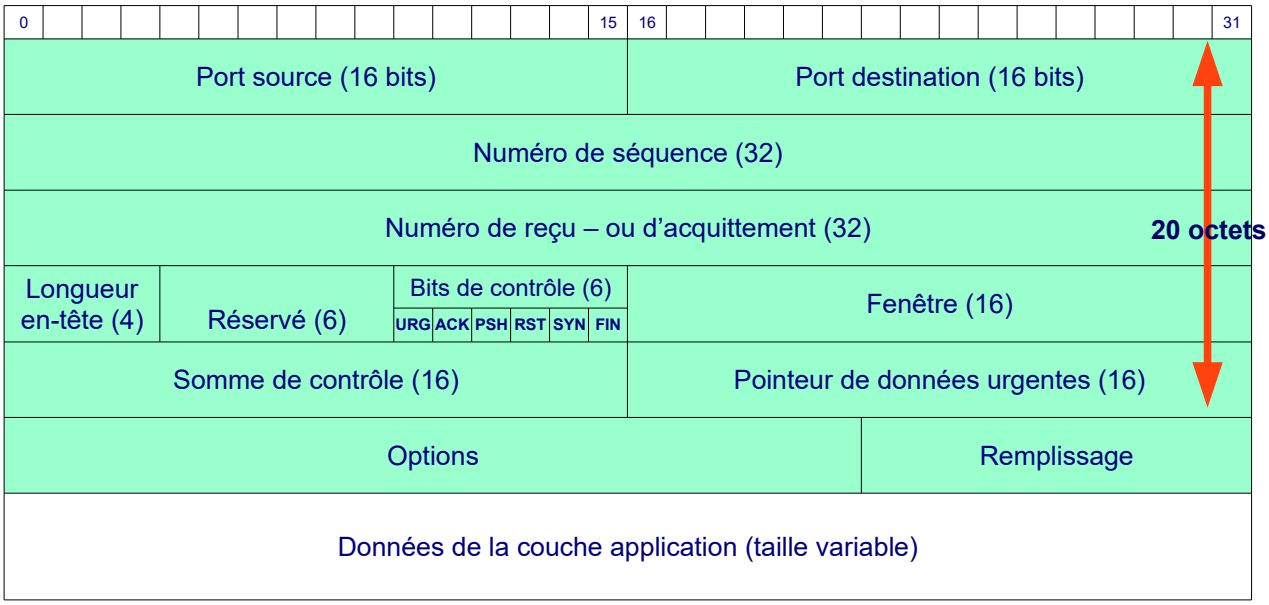
Certains champs d'une trame permettent d'identifier les protocoles présents dans la trame. Les valeurs contenues dans ces champs déterminent ainsi, à chaque niveau, le type de message qui sera encapsulé (une trame ARP n'a pas les mêmes champs qu'une trame de type IP, un paquet TCP n'a pas les mêmes champs qu'un paquet UDP, etc).



5 Les en-têtes de protocole

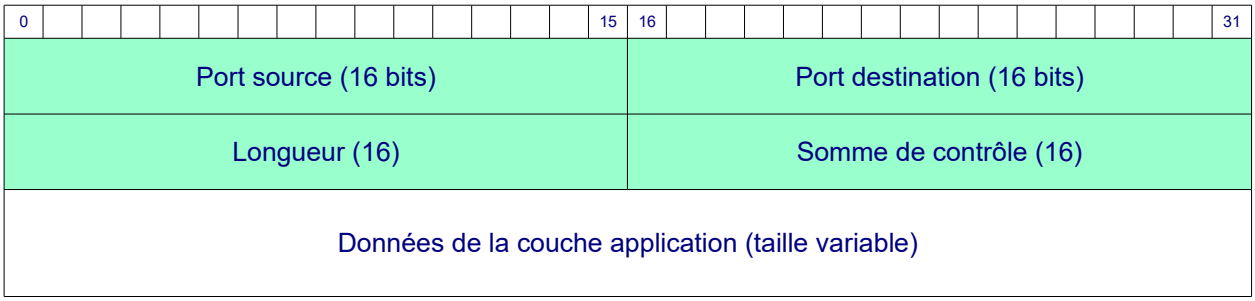
L'en-tête ne comporte pas uniquement les ports source et destination. Du fait du caractère fiable du protocole TCP, l'en-tête est beaucoup plus important que pour le protocole UDP.

En-tête de protocole TCP



L'en-tête TCP comporte au minimum 20 octets (sans les options éventuelles).

En-tête de protocole UDP

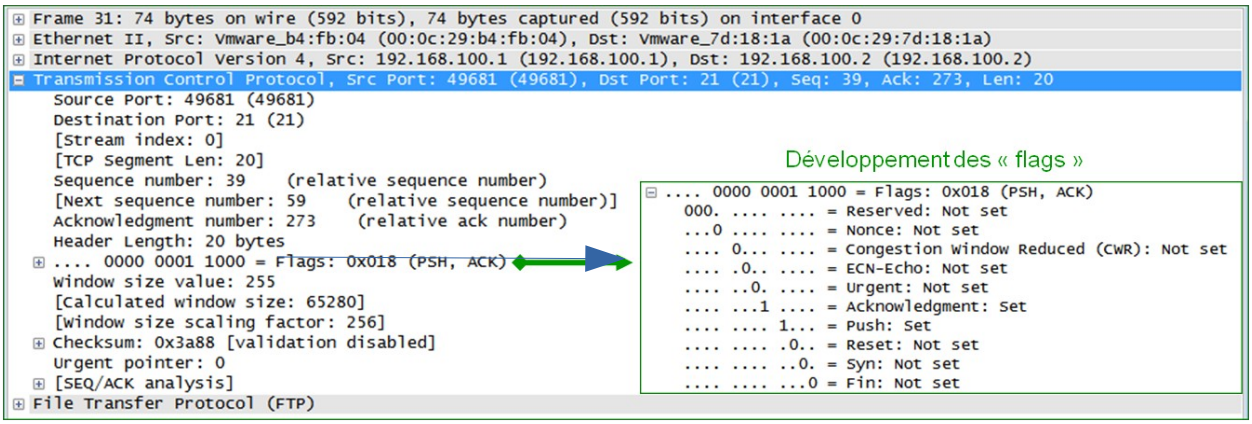


L'en-tête UDP ne comporte que 8 octets.

Exercice

On peut facilement repérer les parties d'un en-tête TCP ou UDP dans une capture de trames.

Trame n°1



Q1. Quel est le service applicatif concerné dans cette trame n°1 ?

Q2. Que est le protocole de transport utilisé ?

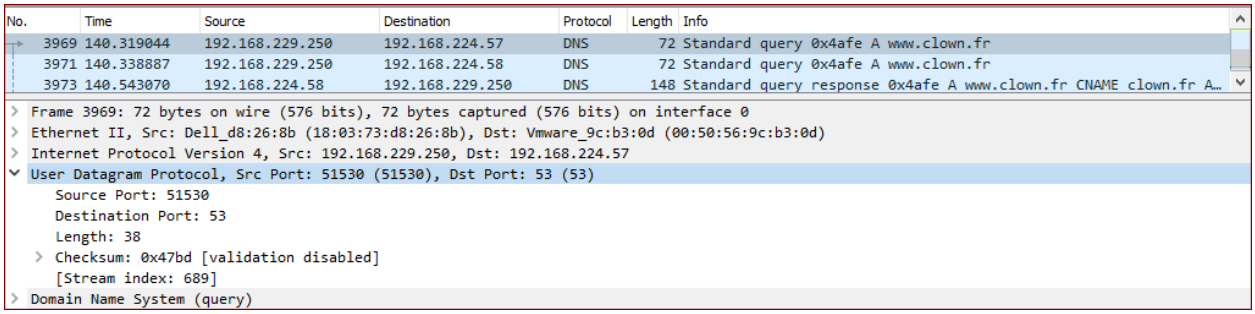
Q3. Quel est le port ouvert sur le client ?

Q4. Quelle est la longueur de l'en-tête TCP ?

Q5. Quelle est la « taille de la « fenêtre » ?

Q6. Ce segment comporte-t-il un accusé réception ?

Trame n°2



Q7. Quel est le service applicatif concerné ? Quel est le protocole de transport associé ?

Q8. Pourquoi d'après vous ce service utilise-t-il (le plus souvent) ce protocole de transport ?

Q9. Quel est le port source ouvert sur le client ?

Q10. À quoi correspond la longueur (Length) indiquée dans l'en-tête UDP