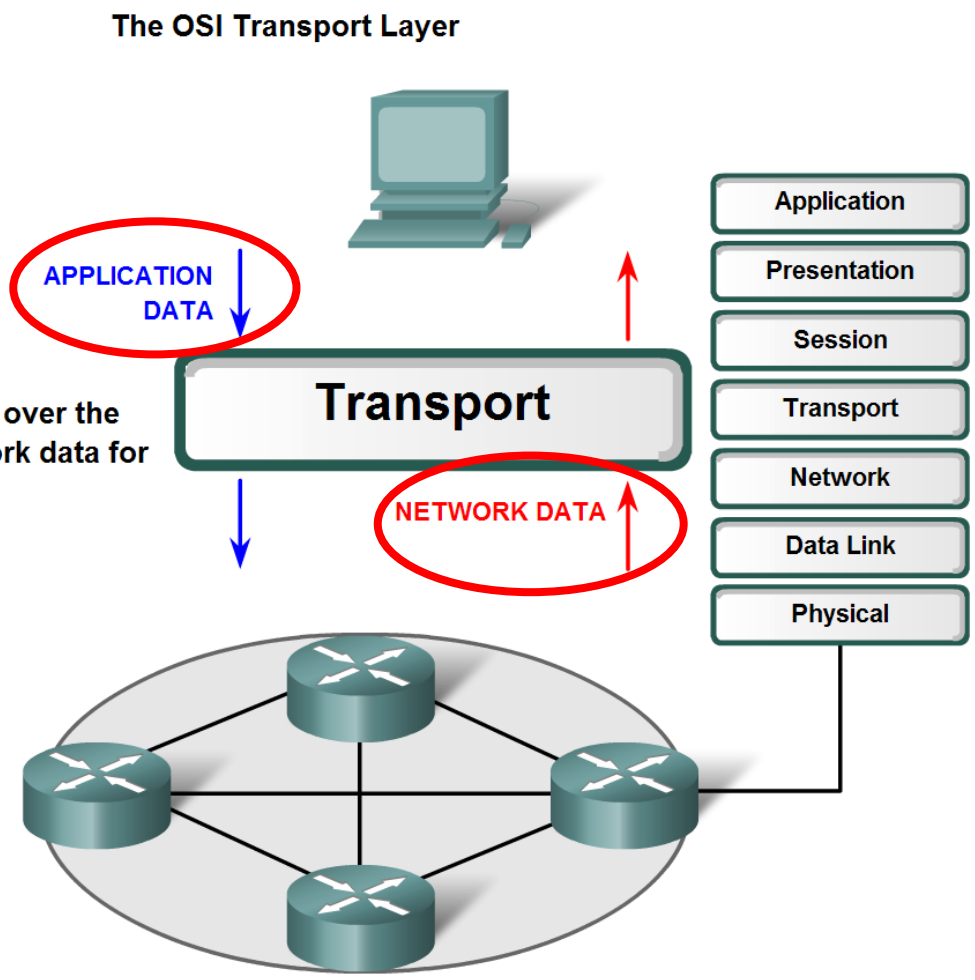
La couche 4: Objectifs

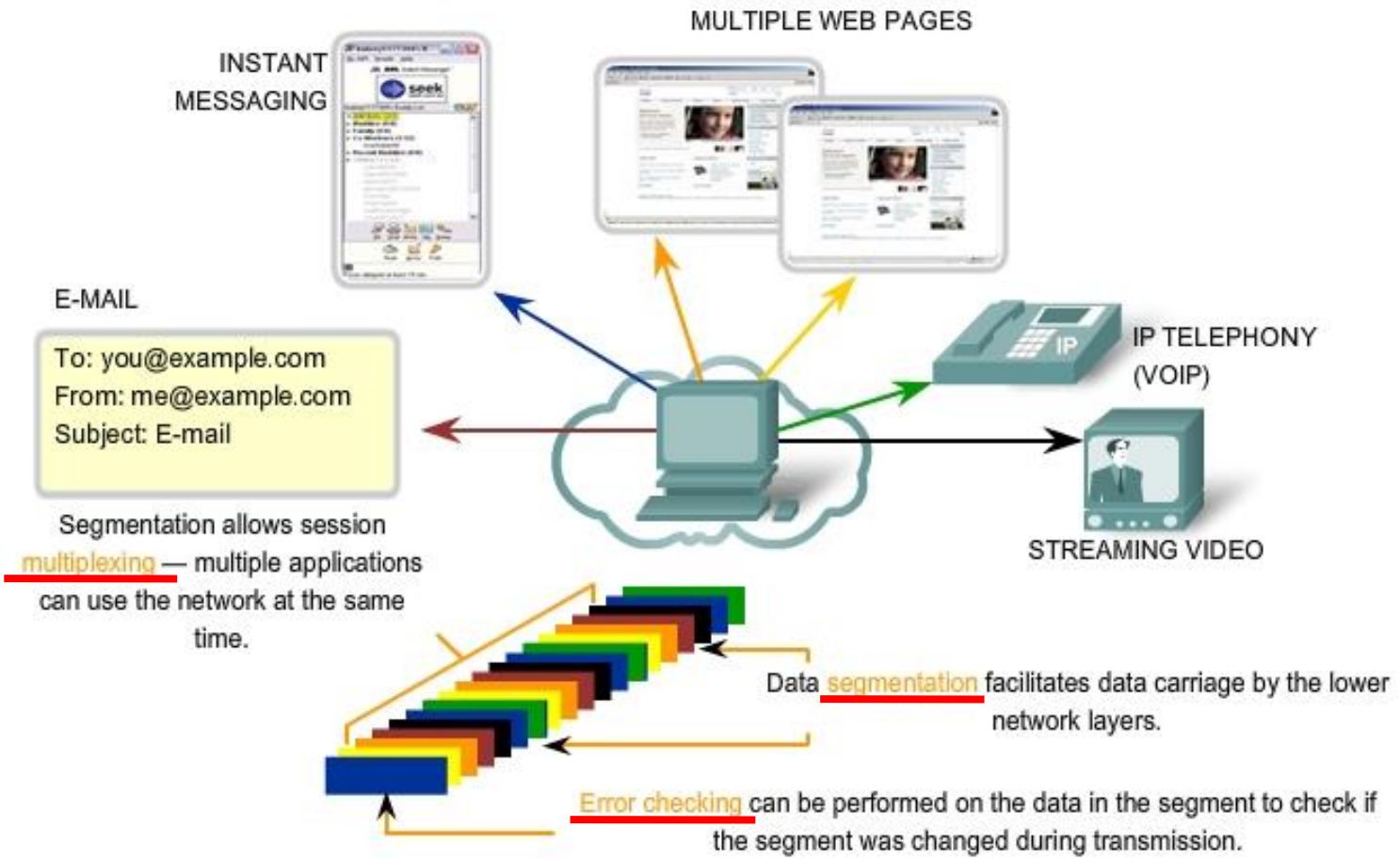
- Expliquer les rôles de la couche Transport et développez les protocoles et services rencontrés.
- Analyser les mécanismes mis en place par TCP pour assurer la fiabilité du transport.
- Différencier UDP de TCP. Avantages et inconvénients de chacun.

Couche 4: Transport

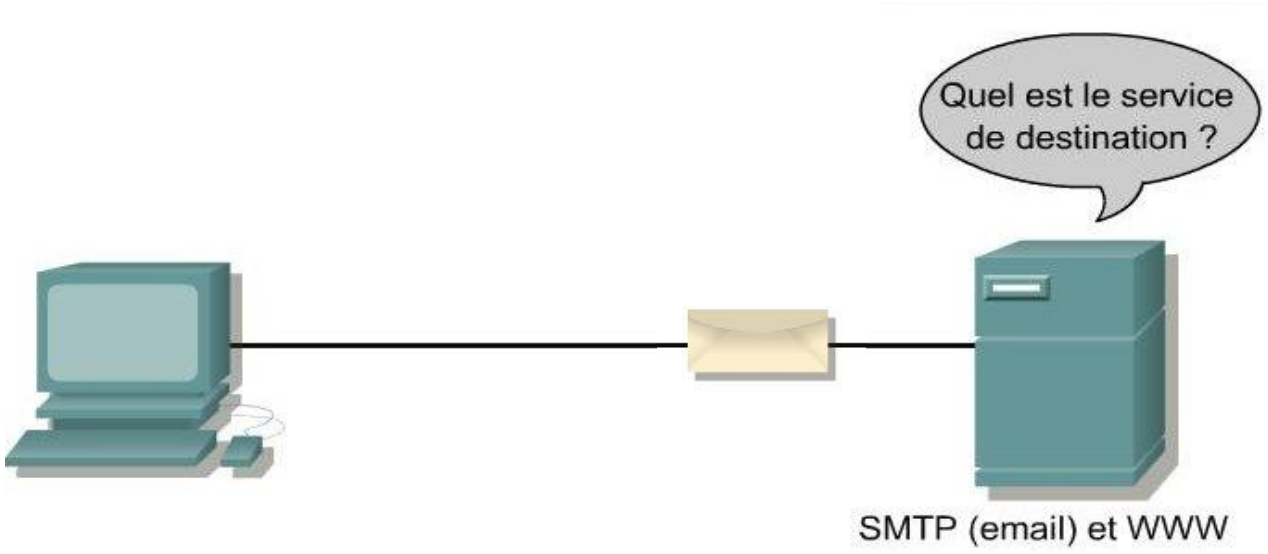
The Transport layer prepares application data for transport over the network and processes network data for use by applications.



TCP-UDP – Fonctionnalités communes: segmentation, multiplexage et contrôle d'erreurs

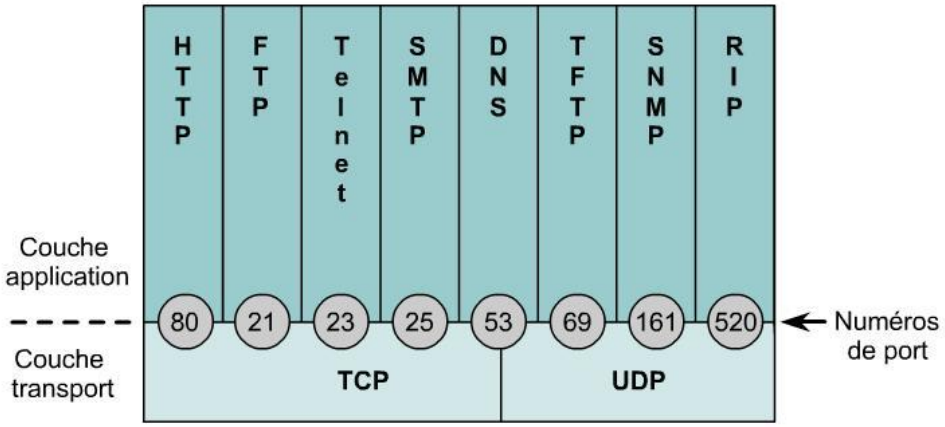


Numéros de ports



Si une machine envoie des paquets à un serveur qui exécute plusieurs services, comment le serveur sait-il à quelle application les données sont destinées?

➔ Port de destination



Registered TCP Ports:
1863 MSN Messenger
2000 Cisco SCCP (VoIP)
8008 Alternate HTTP
8080 Alternate HTTP

Well Known TCP Ports:
21 FTP
23 Telnet
25 SMTP
80 HTTP
110 POP3
194 Internet Relay Chat (IRC)
443 Secure HTTP (HTTPS)

Registered UDP Ports:
1812 RADIUS Authentication Protocol
5004 RTP (Voice and Video Transport Protocol)
5040 SIP (VoIP)

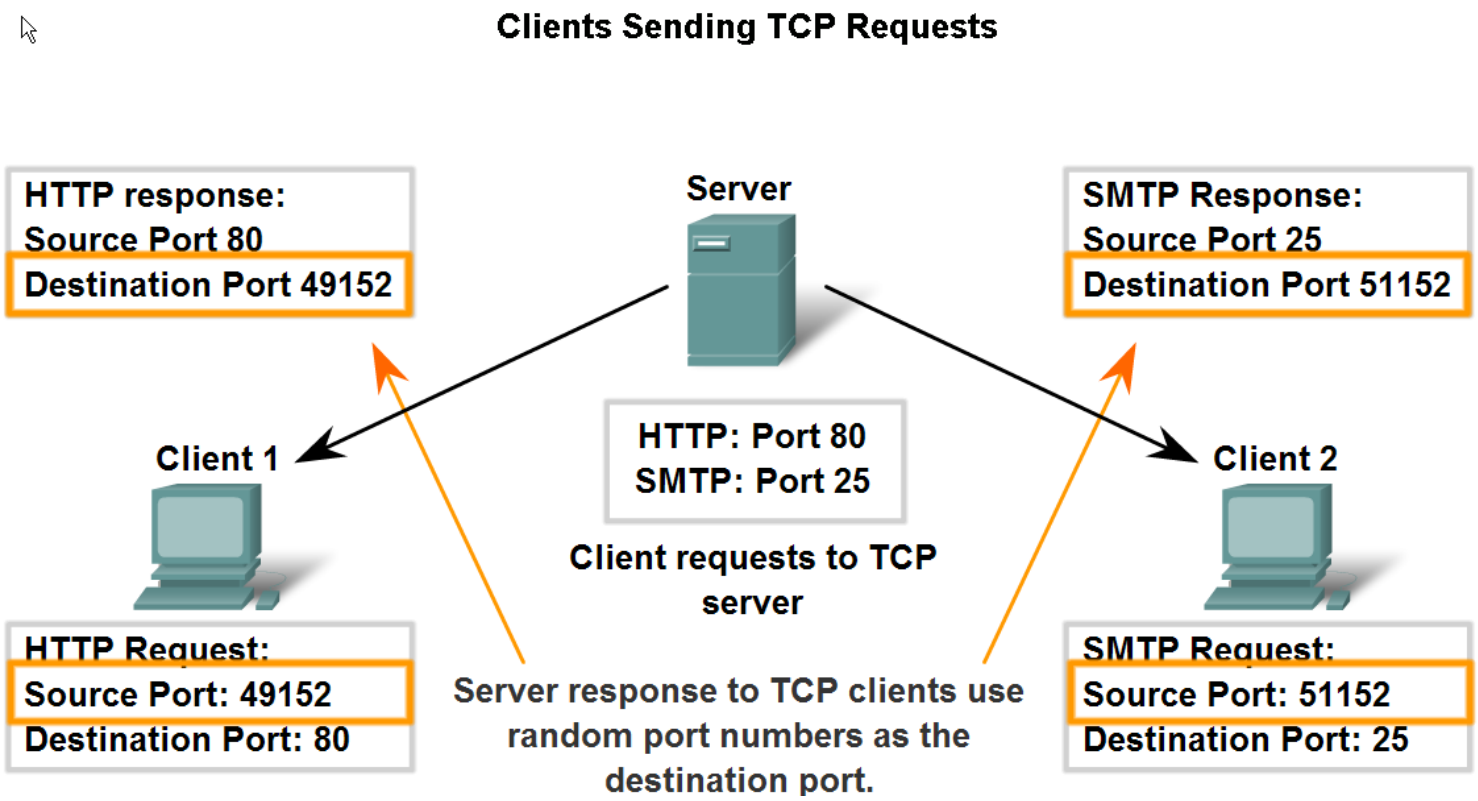
Well Known UDP Ports:
69 TFTP
520 RIP

Registered TCP/UDP Common Ports:
1433 MS SQL
2948 WAP (MMS)

Well Known TCP/UDP Common Ports:
53 DNS
161 SNMP
531 AOL Instant Messenger, IRC

Les numéros de port compris entre 0 et 1023 sont considérés publics et sont contrôlés par l'IANA (Internet Assigned Numbers Authority).

Numéros de port dans une session TCP-UDP



Netstat

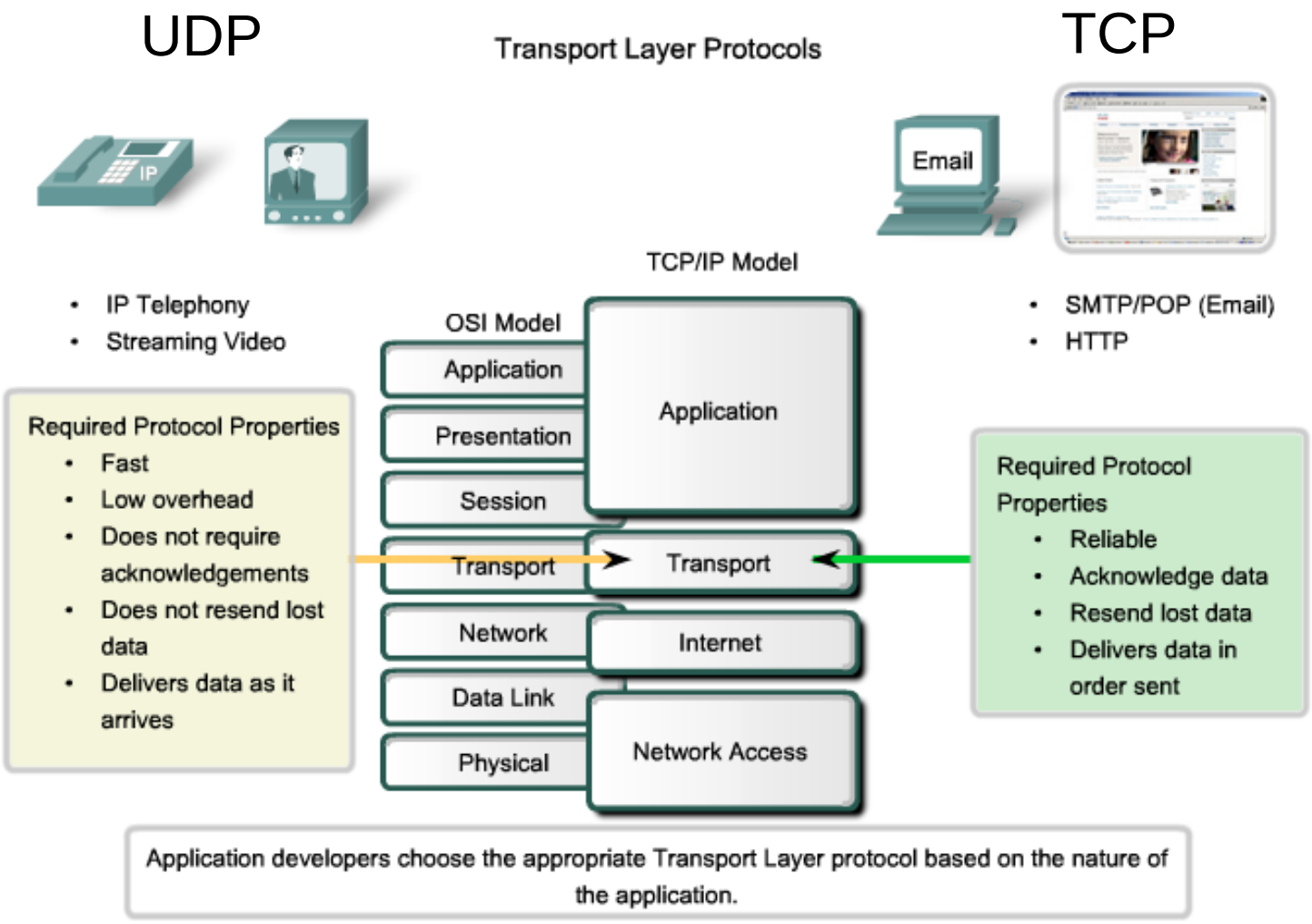
- Utilisé pour examiner les connexions TCP qui sont ouvertes sur un hôte réseau

```
C:\>netstat

Active Connections

Proto   Local Address   Foreign Address           State
TCP     kenpc:3126      192.168.0.2:netbios-ssn   ESTABLISHED
TCP     kenpc:3158      207.138.126.152:http      ESTABLISHED
TCP     kenpc:3159      207.138.126.169:http      ESTABLISHED
TCP     kenpc:3160      207.138.126.169:http      ESTABLISHED
TCP     kenpc:3161      sc.msn.com:http           ESTABLISHED
TCP     kenpc:3166      www.cisco.com:http        ESTABLISHED

C:\>
```

Un client et plusieurs sessions

- Que se passe-t-il si vous ouvrez depuis le même client plusieurs sessions vers un même serveur.
- Exemple: vous lancez deux recherches sur *google*. Comment le serveur distingue-t-il vos recherches et ne mélange pas les réponses ?
- Solution: attribuer un port différent, côté client, pour chacune des deux recherches (sessions)

TCP ou UDP

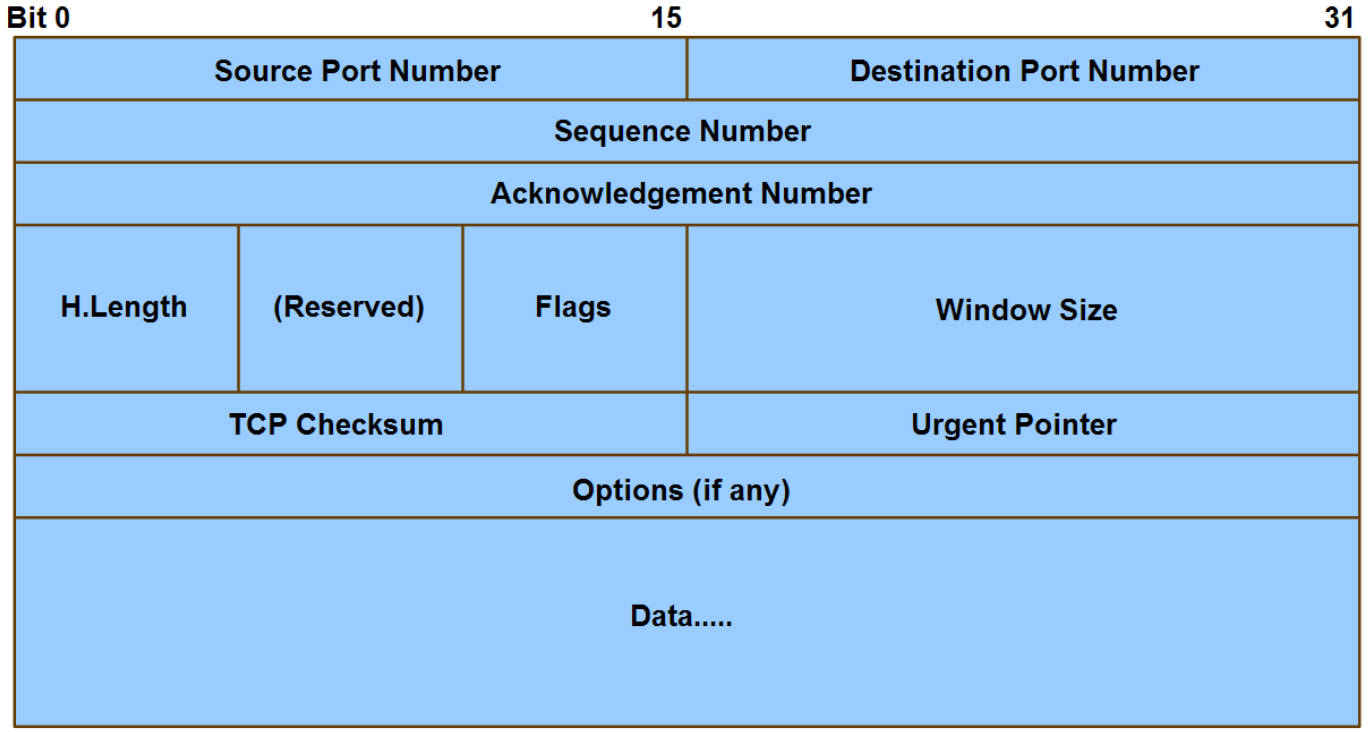

UDP
Transport Layer Protocols
TCP
IP
Email
TCP/IP Model
OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Transport
Internet
Network Access
IP Telephony
Streaming Video
SMTP/POP (Email)
HTTP
Required Protocol Properties
Fast
Low overhead
Does not require acknowledgements
Does not resend lost data
Delivers data as it arrives
Required Protocol Properties
Reliable
Acknowledge data
Resend lost data
Delivers data in order sent
Application developers choose the appropriate Transport Layer protocol based on the nature of the application.

Fonctionnalité de TCP (rfc 793)

- Fiabilité
  - Accusés de réception et Stateful protocol
  - Retransmission des segments perdus
- Délivrer les données dans le bon ordre
  - Segmenter
  - Numéroter
  - Réassembler
- Connexion,
  - Session TCP « Three Ways Handshake »
- Contrôle de flux
  - Fenêtre glissante
  - Garder une trace de la session

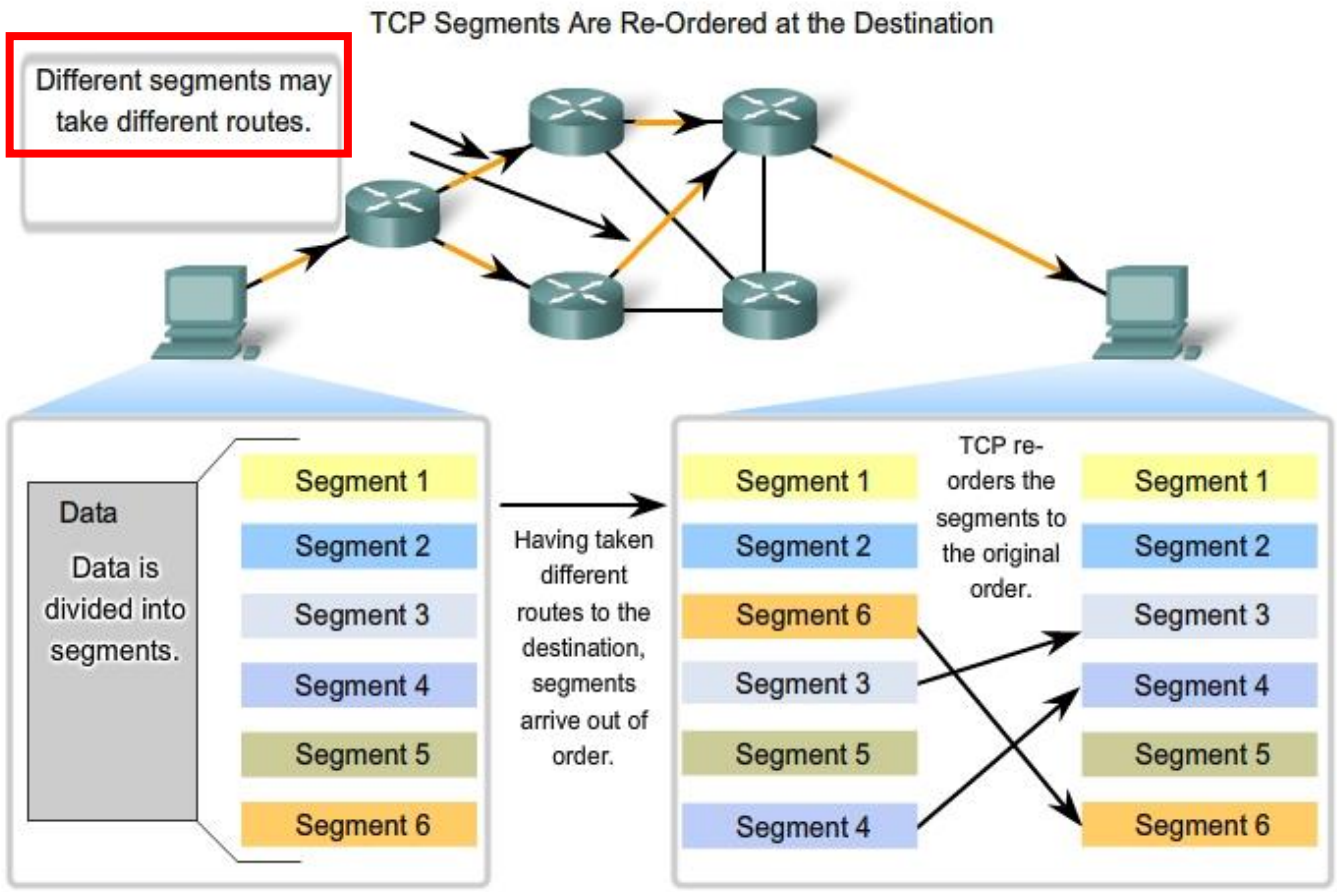
En-tête TCP

TCP Segment Header Fields



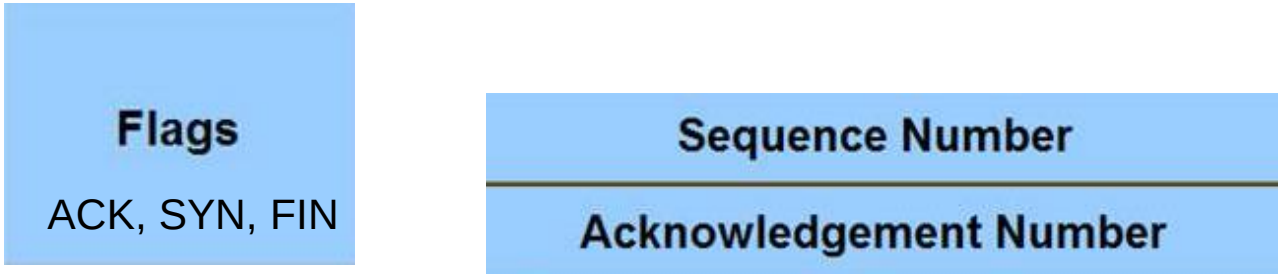
The fields of the TCP header enable TCP to provide connection-oriented, reliable data communications.

Numéros de séquence: Délivrer les données dans le bon ordre



Étapes d’une session TCP

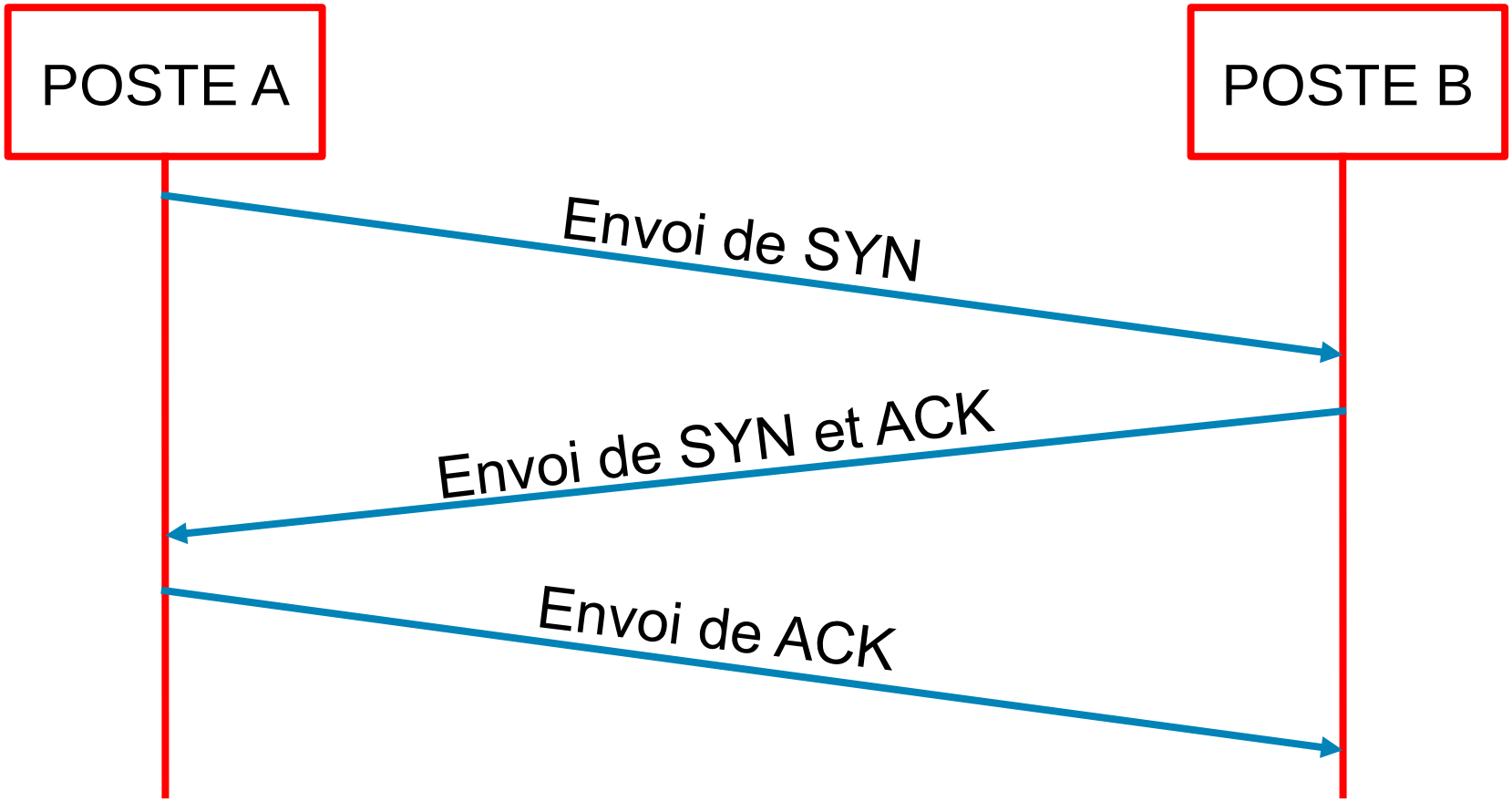
- Établissement de la session
- Session
- Fin de la session
- Etapes utilisant les éléments suivants:



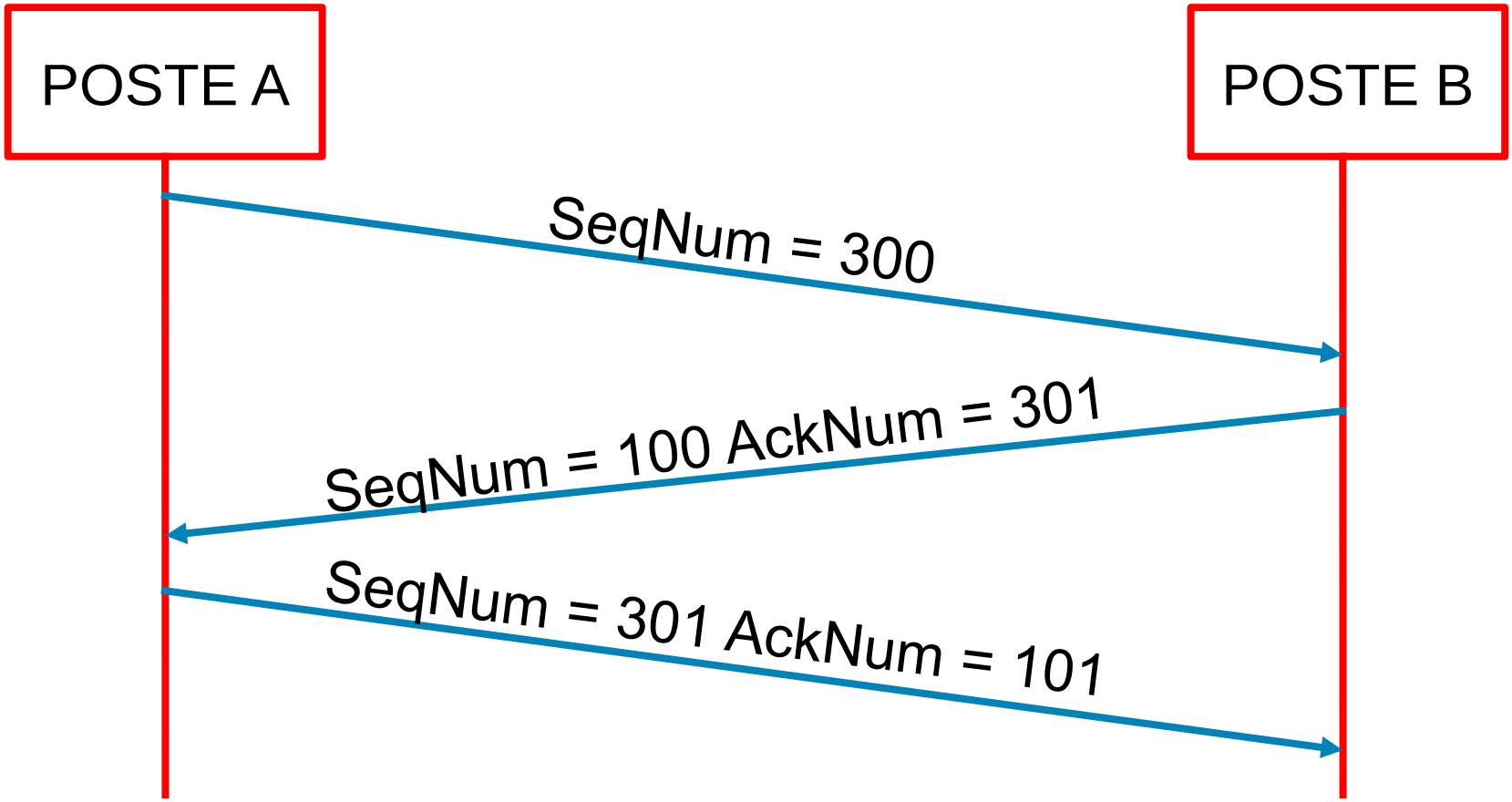
Étapes d'une session TCP

- ACK → le champ acknowledge number est valide
- SYN → ouverture de session
- FIN → fermeture de session
- Sequence number = le numéro du premier octet de données du segment → identifie le segment envoyé par la source.
- Acknowledgement number = le numéro du prochain octet (segment) attendu par le récepteur.
- Segment envoyé → copié dans une file d'attente pendant une tempo donnée.
  Si pas d'aknowledgement avant la fin de tempo → segment renvoyé
- !!! Full duplex → un périphérique = émetteur et récepteur

Établissement d'une session TCP: observation des flags dans Wireshark



Etablissement d'une session TCP: observation des numéros de séquence et d'acquittement



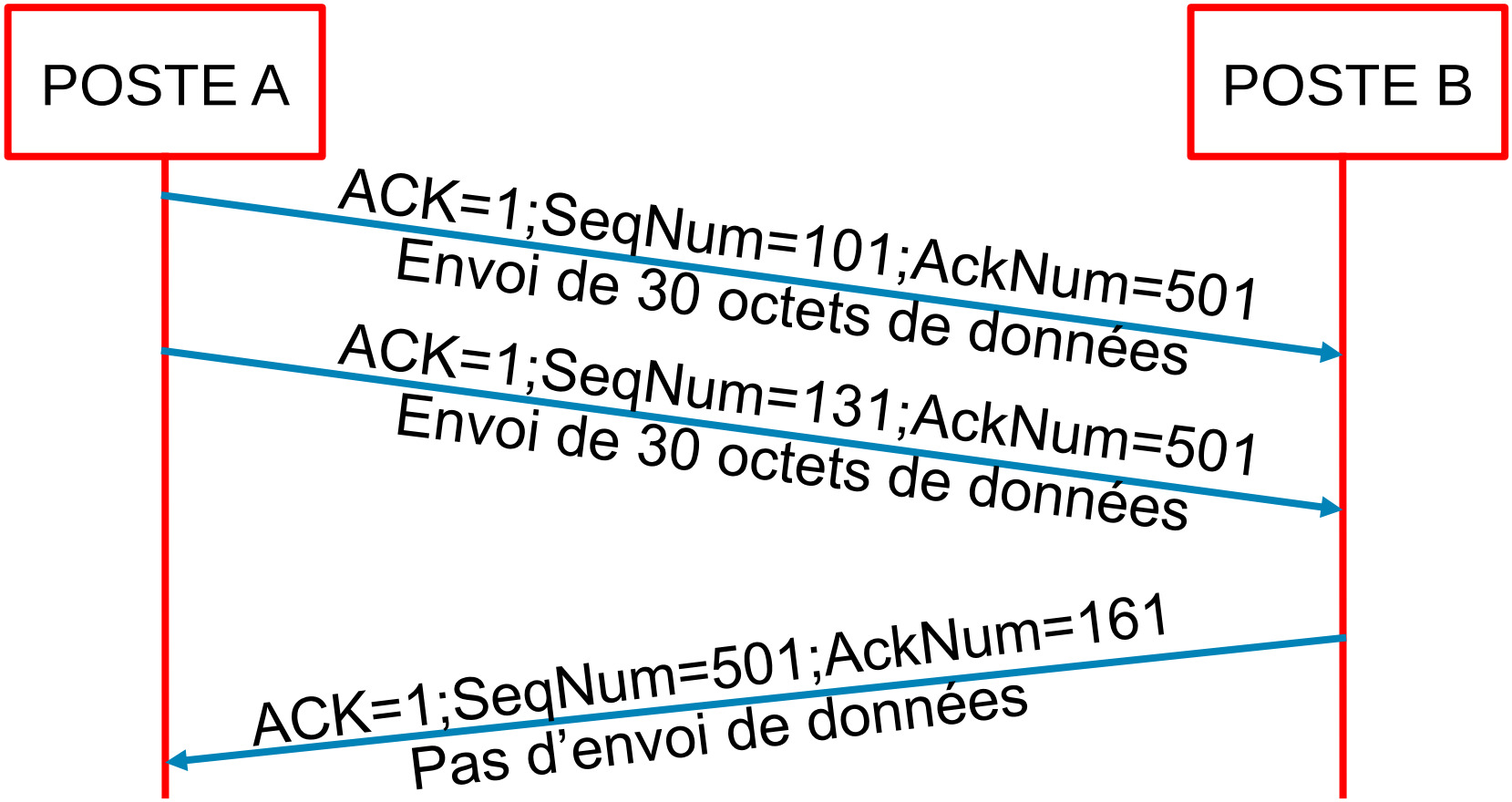
Etablissement de la session: en conclusion

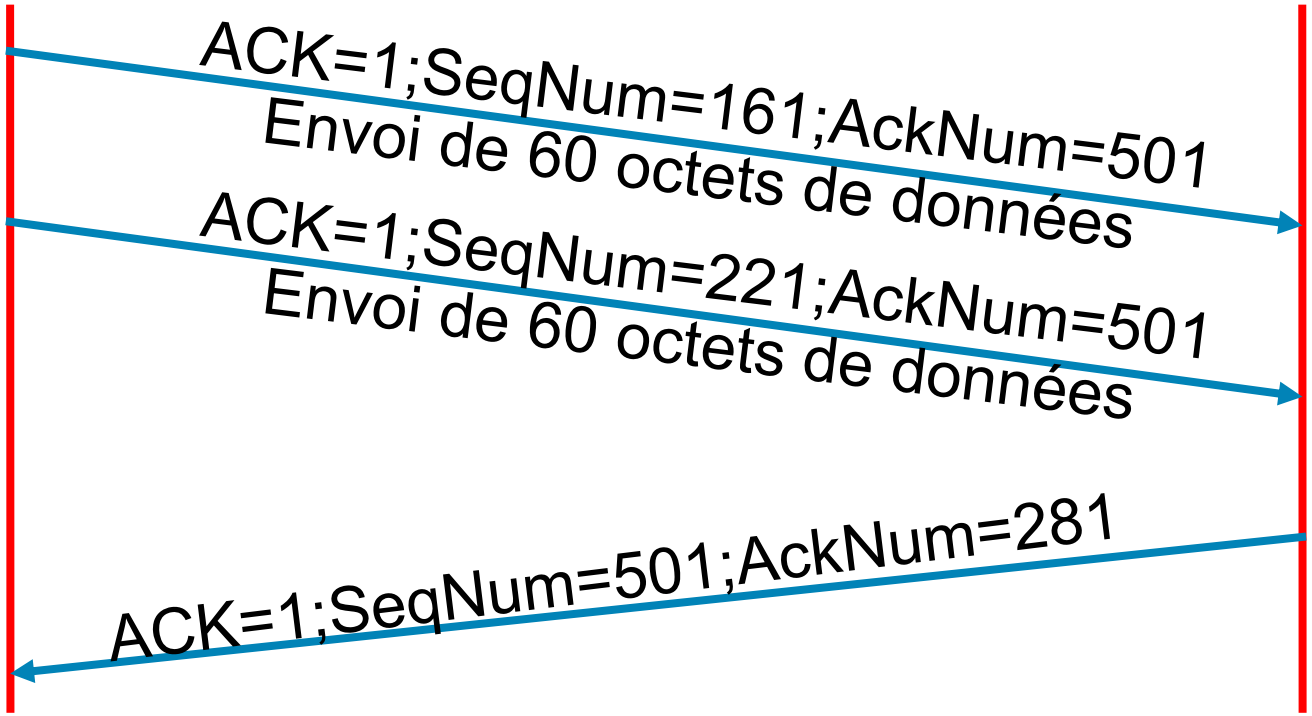
==> SYN=1 - ACK=0 - SeqNum=300

<== SYN=1 - ACK=1 - SeqNum=100 - AckNum=301

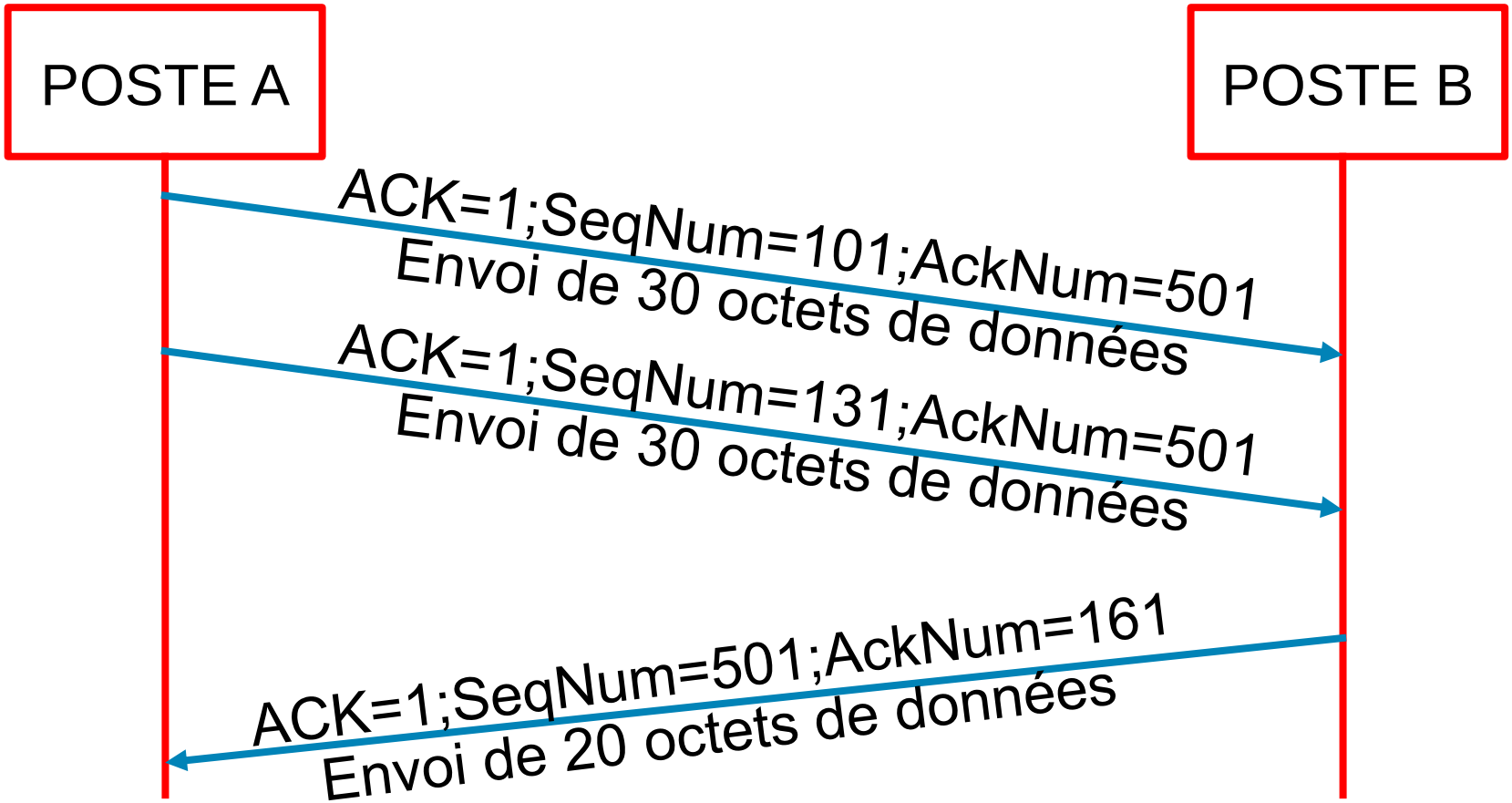
==> SYN=0 - ACK=1 - SeqNum=301 - AckNum=101

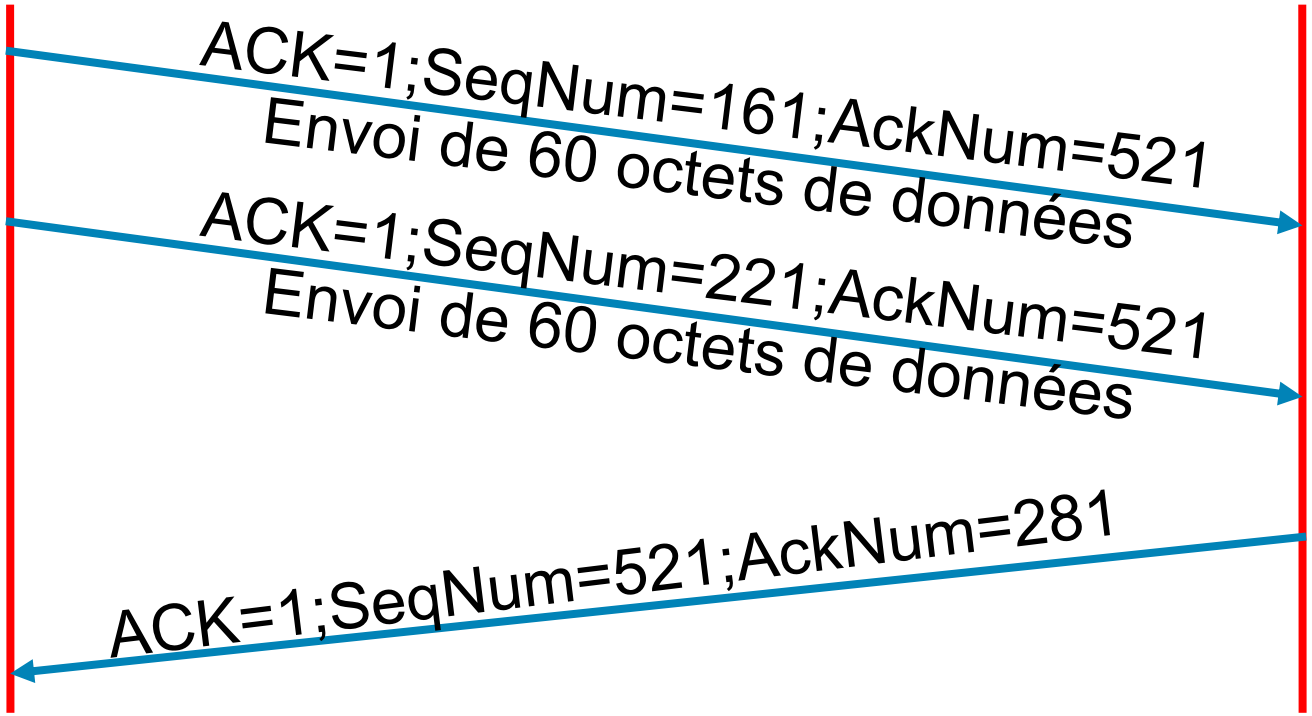
Session TCP – Exemple 1: données dans un seul sens



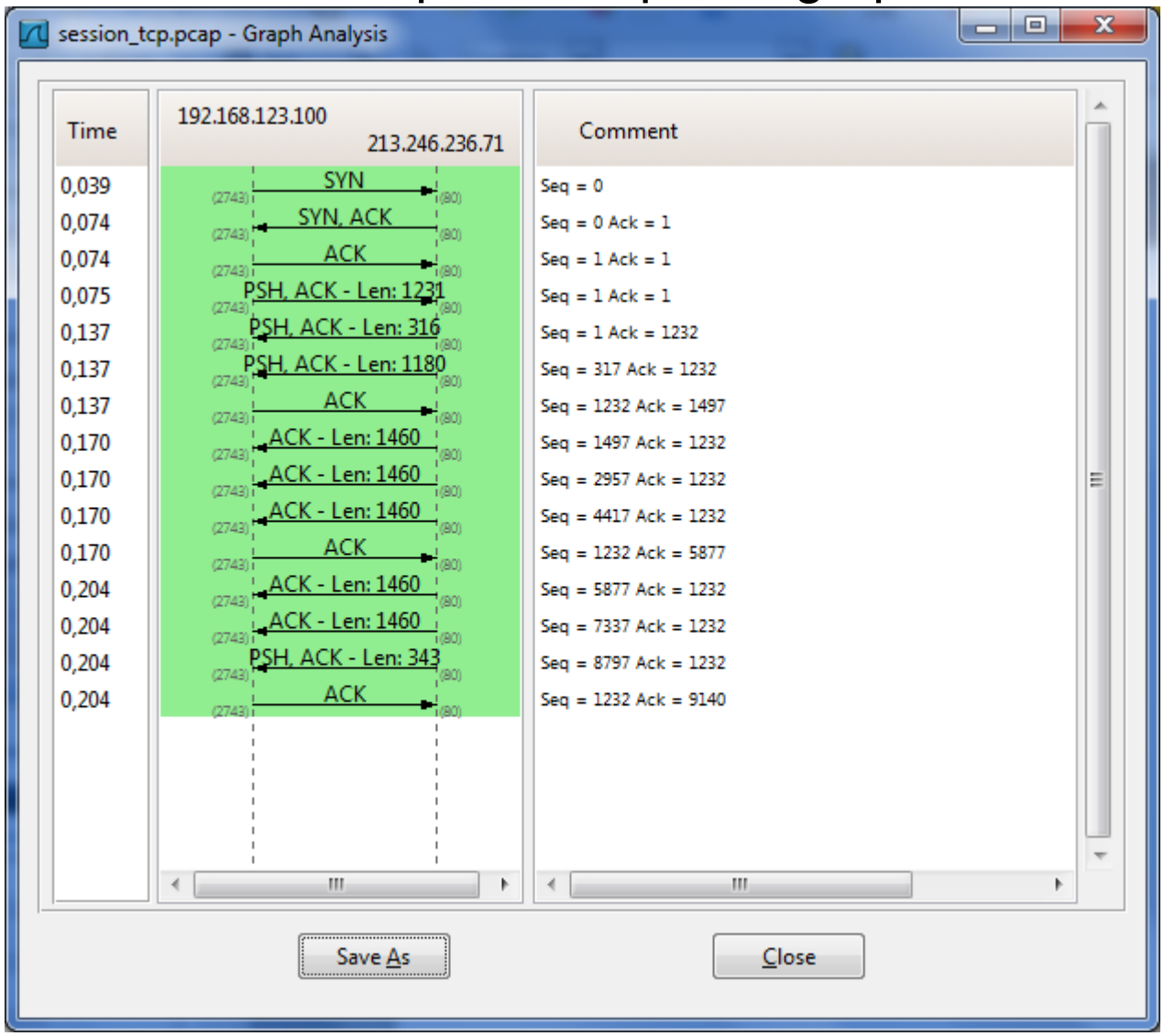

ACK=1;SeqNum=161;AckNum=501
Envoi de 60 octets de données
ACK=1;SeqNum=221;AckNum=501
Envoi de 60 octets de données
ACK=1;SeqNum=501;AckNum=281

Session TCP – Exemple 2: données dans les deux sens

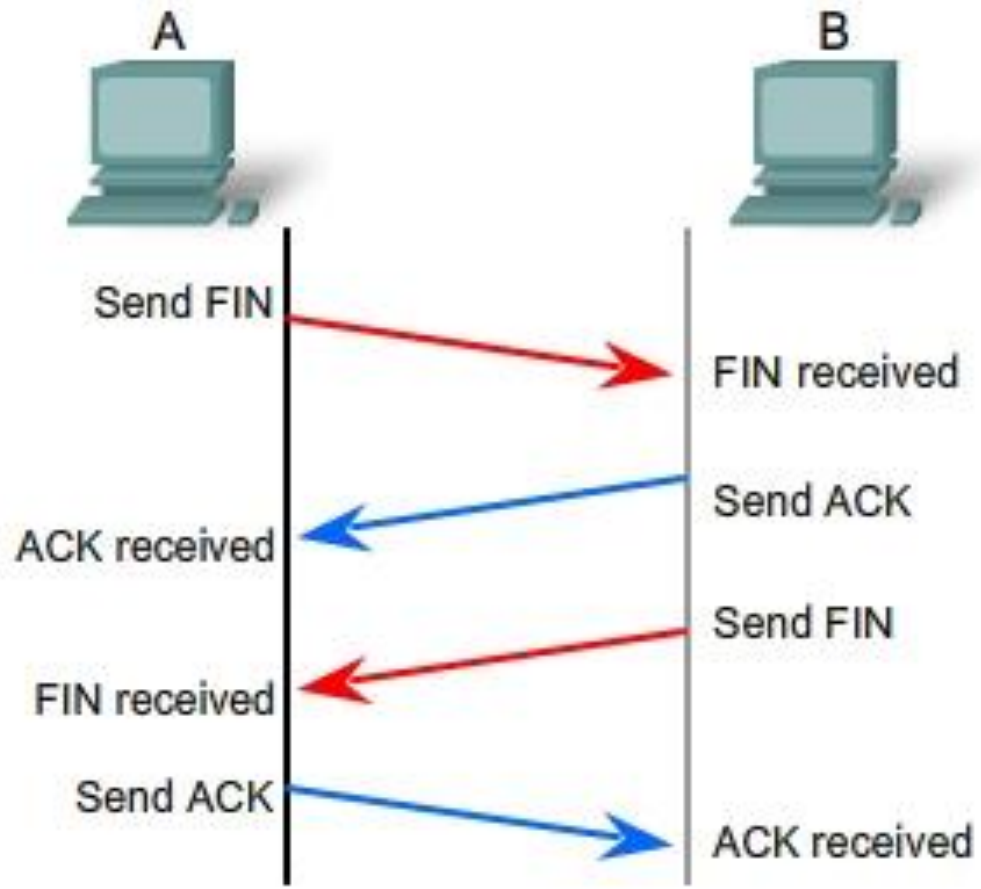


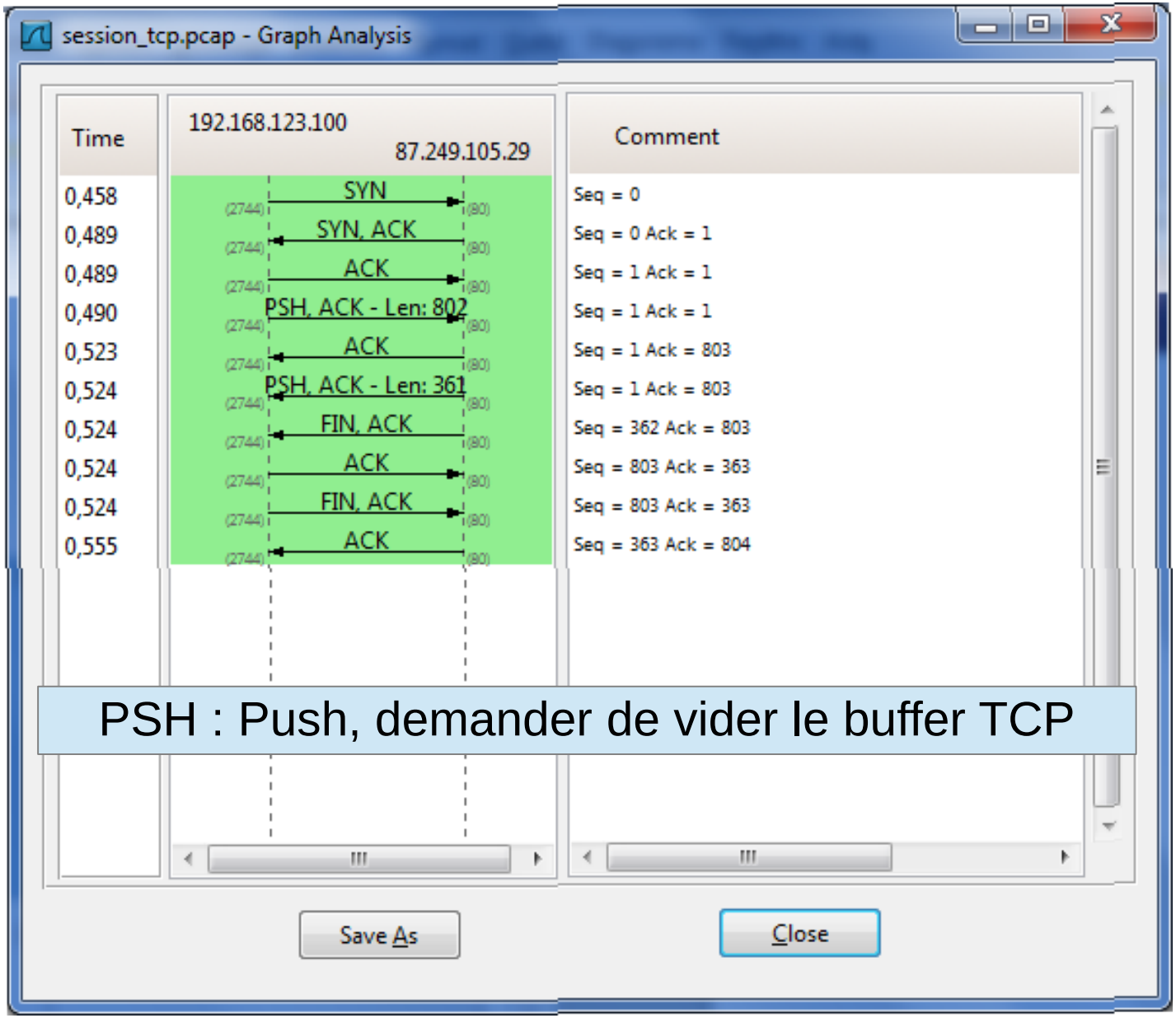

ACK=1;SeqNum=161;AckNum=521
Envoi de 60 octets de données
ACK=1;SeqNum=221;AckNum=521
Envoi de 60 octets de données
ACK=1;SeqNum=521;AckNum=281

Wireshark : statistiques → tcp flow graph



Fin de la session TCP




session_tcp.pcap - Graph Analysis
Time
192.168.123.100
87.249.105.29
Comment
0,458
SYN
(2744)
(80)
Seq = 0
0,489
SYN, ACK
Seq = 0 Ack = 1
0,489
ACK
Seq = 1 Ack = 1
0,490
PSH, ACK - Len: 802
Seq = 1 Ack = 1
0,523
ACK
Seq = 1 Ack = 803
0,524
PSH, ACK - Len: 361
Seq = 1 Ack = 803
0,524
FIN, ACK
Seq = 362 Ack = 803
0,524
ACK
Seq = 803 Ack = 363
0,524
FIN, ACK
Seq = 803 Ack = 363
0,555
ACK
Seq = 363 Ack = 804
PSH : Push, demander de vider le buffer TCP
Save As
Close

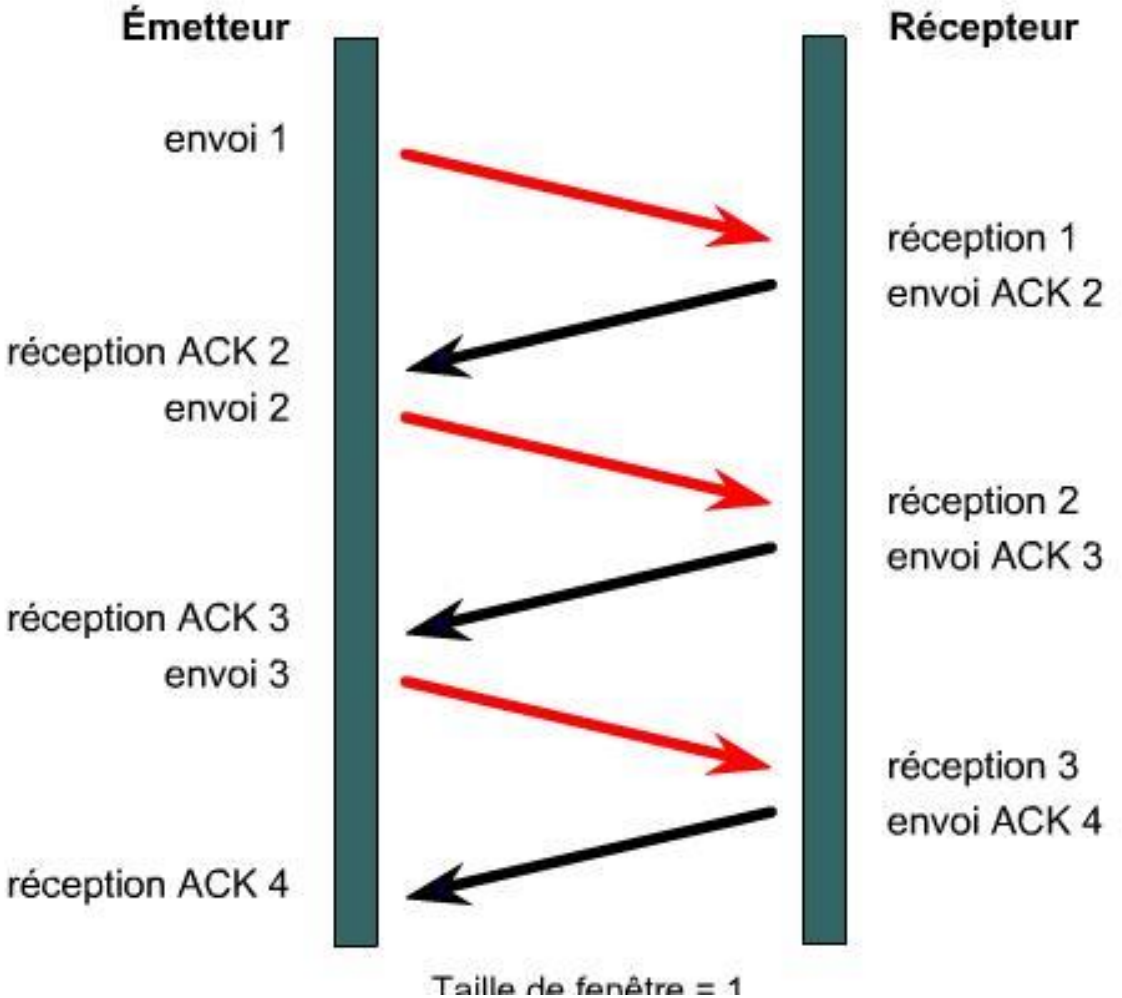
Reset de la session

==> ACK=1 - RST=0 - SeqNum=200 - AckNum=400
<== ACK=0 - RST=1 - SeqNum=400

OR

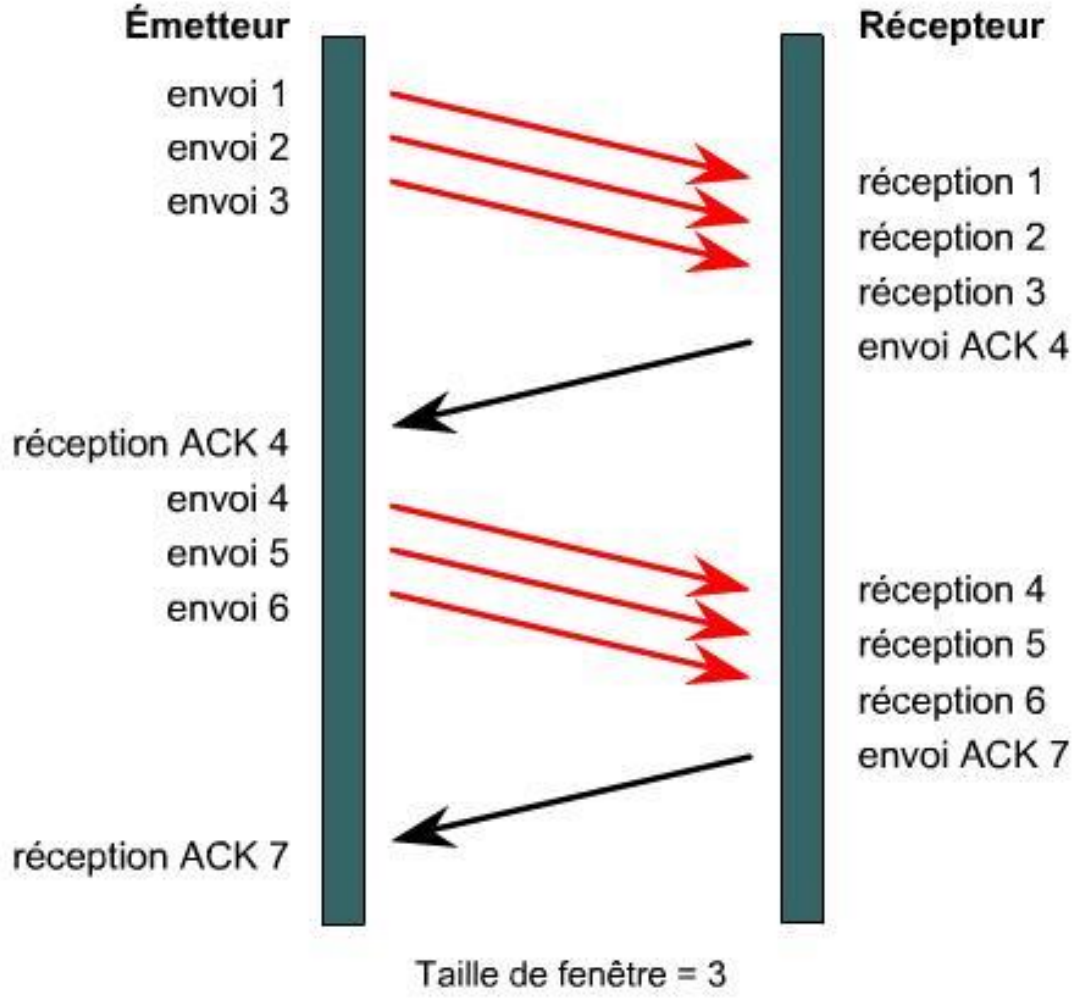
<== ACK=0 - RST=0 - SeqNum=200 - Data=30 octets
==> ACK=0 - RST=1 - SeqNum=230

Contrôle du flux et fenêtre



Si l'émetteur doit attendre un ACK après chaque segment

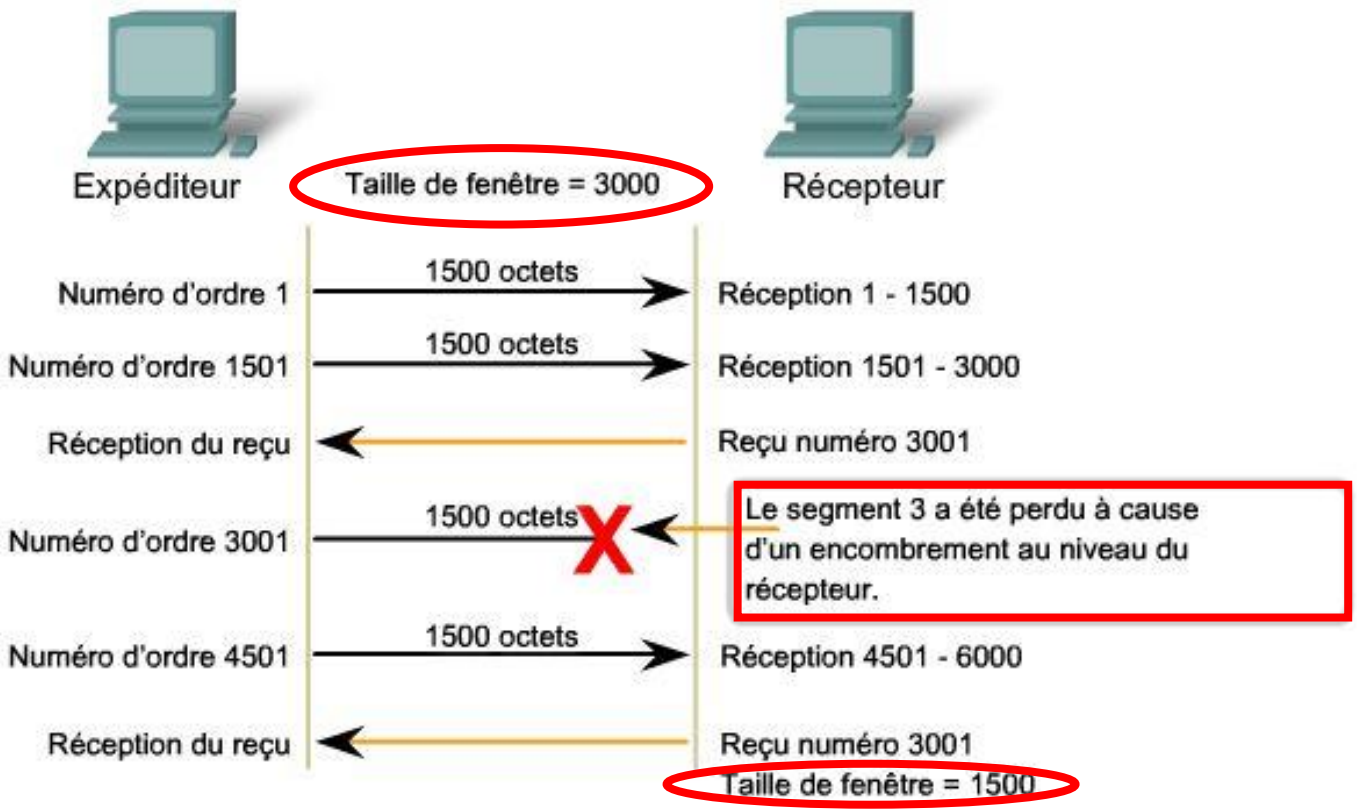
→ lent.



On peut accroître le débit en augmentant la taille de la fenêtre.

C’est en contrôlant cette fenêtre que le flux est régulé.

Taille de la fenêtre en cas de perte



Si des segments sont perdus du fait d'un encombrement, le récepteur enverra un reçu pour le dernier segment séquentiel reçu et répondra en utilisant une taille de fenêtre réduite.

Variante TCP: Selective Acknowledgements (SACK)

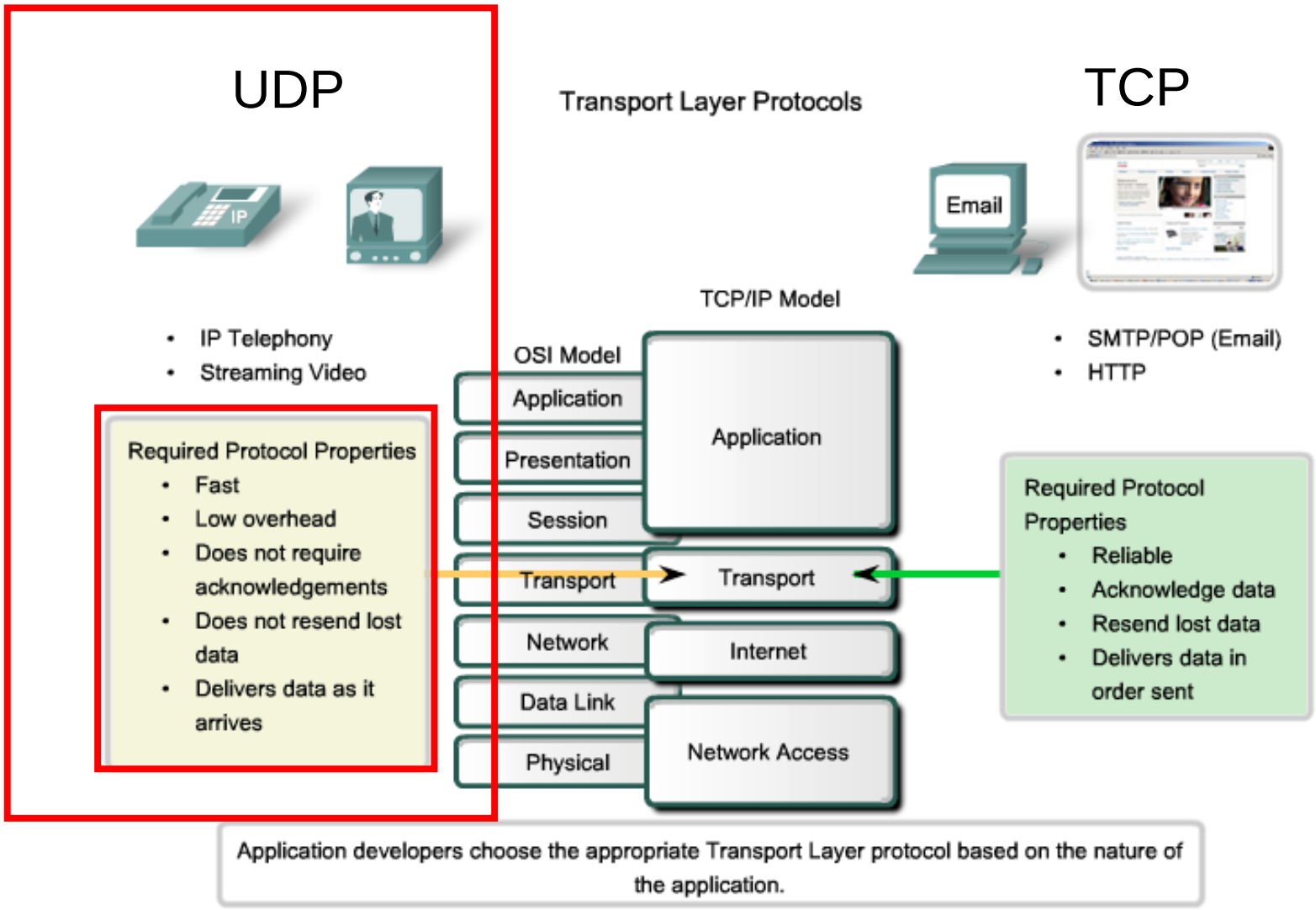
Implémentation d'un mécanisme dans TCP pour éviter un défaut du mécanisme d'accusé de réception :

- Si le client ne reçoit pas un segment
- SACK permet au client de ne demander de transmettre que les segments qui n'ont pas été réçu

Lecture de l'article :

http://packetlife.net/blog/2010/jun/17/tcp-selective-acknowledgments-sack/

UDP


UDP
Transport Layer Protocols
TCP
IP
Email
TCP/IP Model
OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Transport
Internet
Network Access
IP Telephony
Streaming Video
SMTP/POP (Email)
HTTP
Required Protocol Properties
Fast
Low overhead
Does not require acknowledgements
Does not resend lost data
Delivers data as it arrives
Required Protocol Properties
Reliable
Acknowledge data
Resend lost data
Delivers data in order sent
Application developers choose the appropriate Transport Layer protocol based on the nature of the application.

Comparaison TCP - UDP

TCP and UDP Headers

TCP Segment

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Sequence Number (32)		
Acknowledgement Number (32)		
Header Length (4) Reserved (6) Code Bits (6)	Window (16)	
Checksum (16)	Urgent (16)	
Options (0 or 32 if any)		
APPLICATION LAYER DATA (Size varies)		

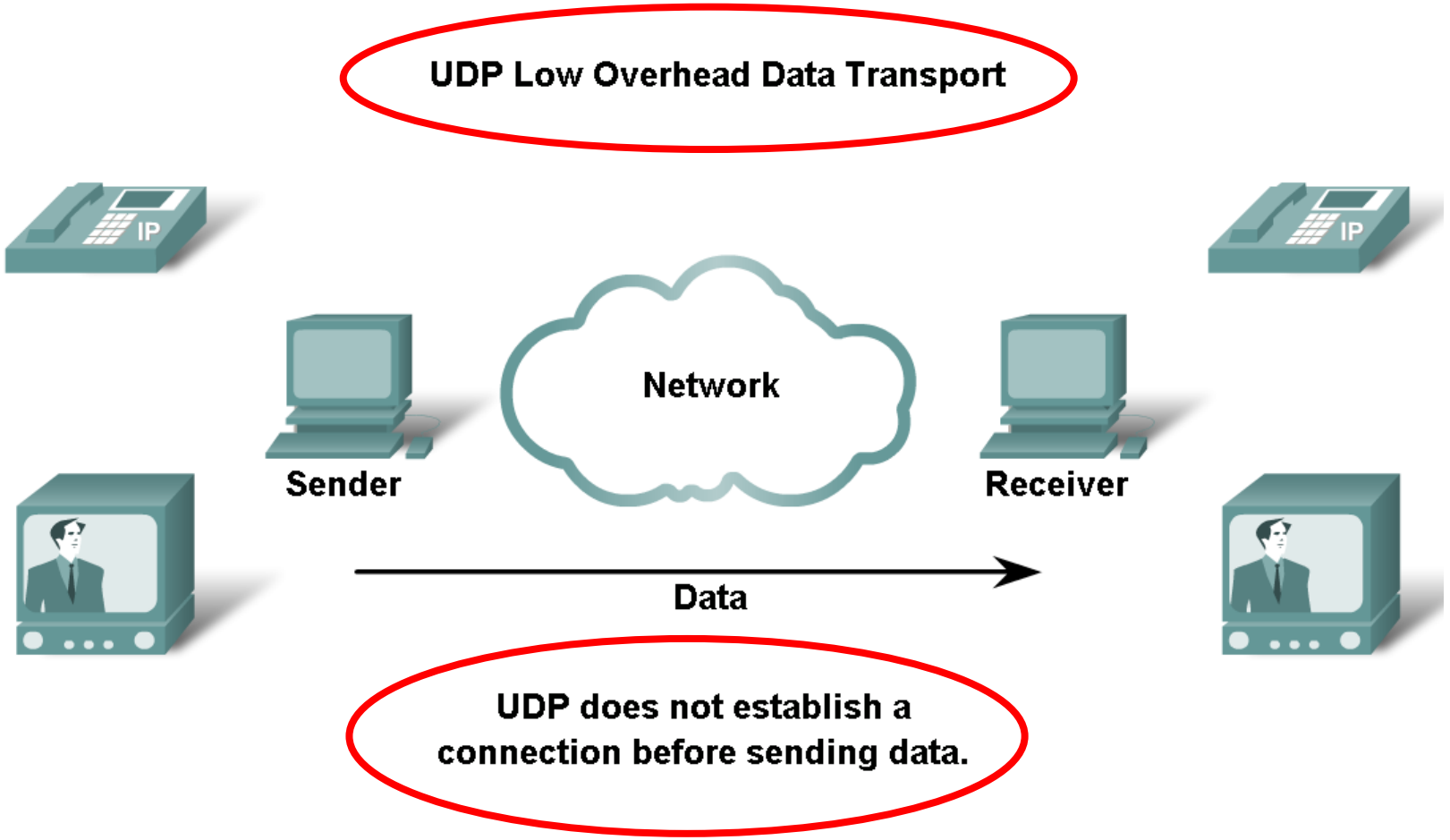
20 Bytes

UDP Datagram

Bit (0)	Bit (15) Bit (16)	Bit (31)
Source Port (16)	Destination Port (16)	
Length (16)	Checksum (16)	
APPLICATION LAYER DATA (Size varies)		

8 Bytes

UDP: moins de surcharge et mode non connecté



Réassemblage des données avec UDP