

**INTRODUCTION AUX RESEAUX
INFORMATIQUES
PROTOCOLES (partie 2)**

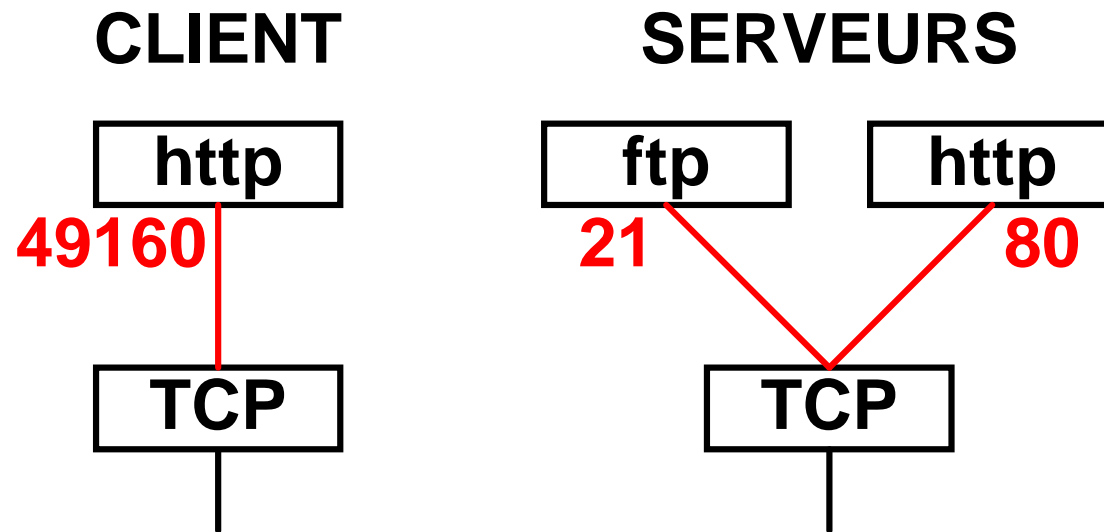
20 décembre 2007

Gérald Litzistorf

- ***Transmission Control Protocol (TCP)*** → slides 3-17
- ***User Datagram Protocol (UDP)*** → slides 18-19 (rappel)
- ***Applications web*** → slides 20
- ***HyperText Transfer Protocol (http)*** → slides 21-27
- ***Labo*** → slide 28
- ***Loopback*** → slide 29

- Le service offert par cette couche est **orienté connexion**
- Tout échange va donc commencer par une **phase d'établissement**
- Le flux de données sécurisé dispose donc de fonctions de **contrôles d'erreur et de flux** (numéro de séquence, accusé de réception,...)
- L'unité d'échange est appelé segment

- Ces champs identifient le protocole de couche supérieure
Source port + Destination port



- La norme définit le *socket* = adresse IP + port

- Des valeurs normalisées (*Well Known Ports*) comprises entre 0 et 1023 donnent **accès aux serveurs** :

23 → telnet

21 → ftp

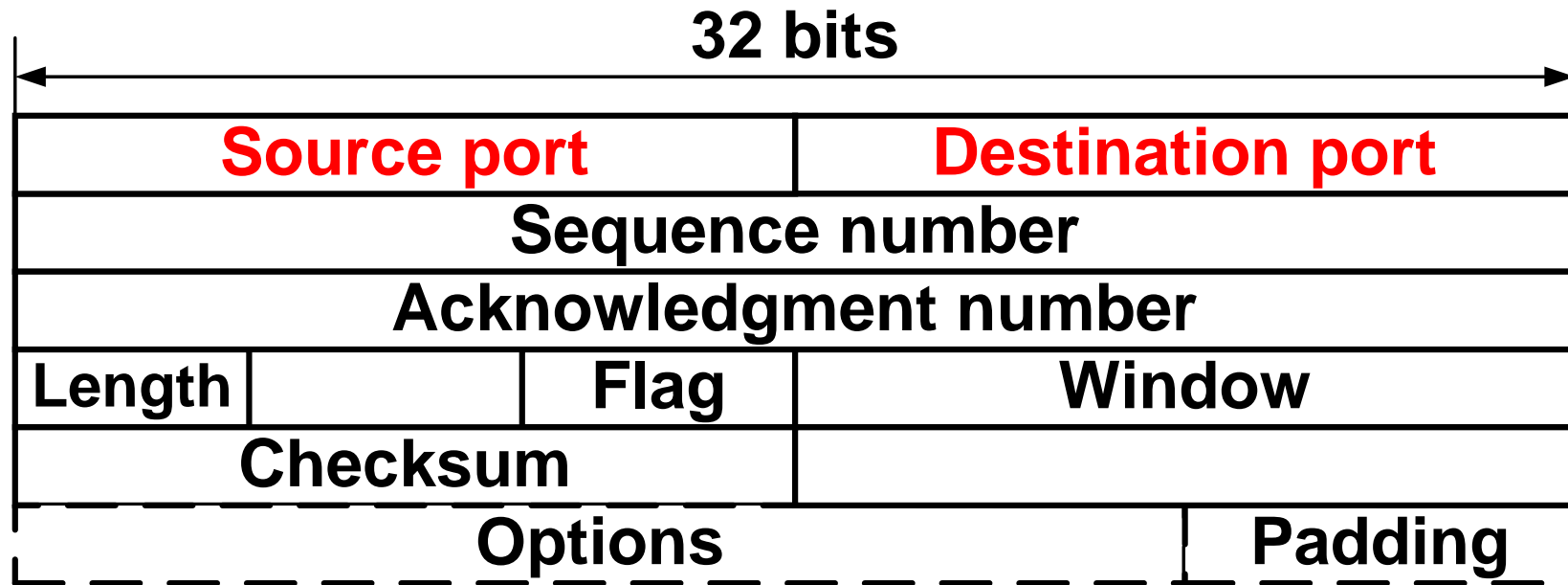
25 → smtp

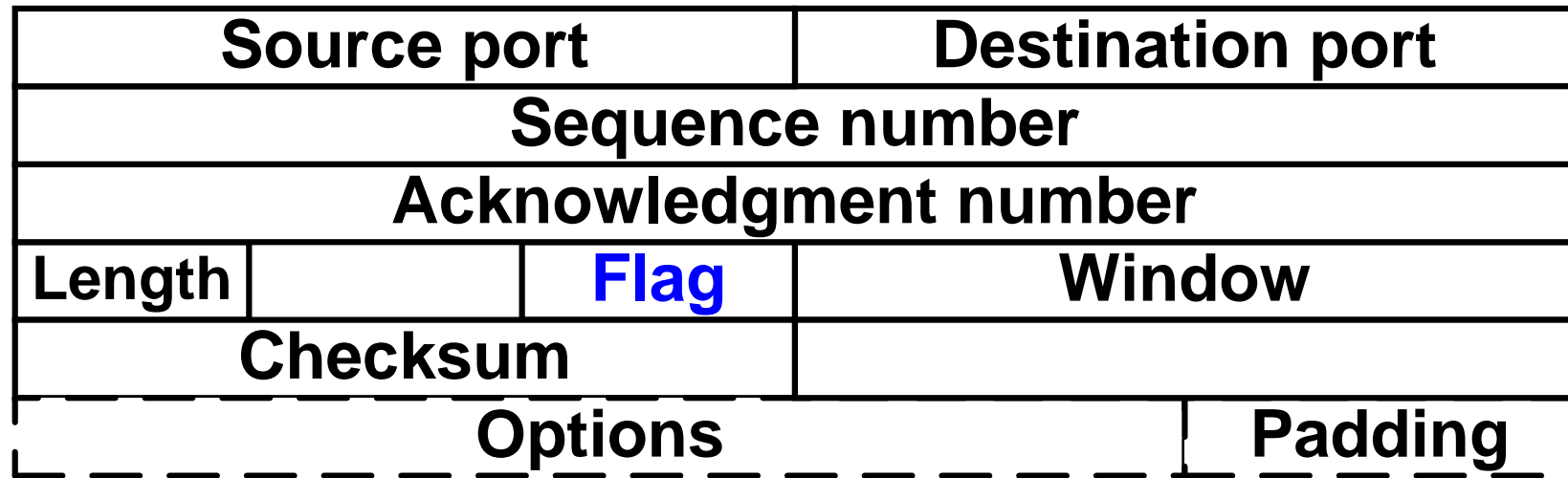
80 → http

<C:\WINDOWS\system32\drivers\etc\services>

- Le **client attribue dynamiquement** les numéros de port dans l'espace 49152 et 65535 (*Dynamic Ports*)
- Autre espace défini : *Registered Ports* = 1024 .. 49151

- La longueur habituelle (sans option) est de 20 octets :





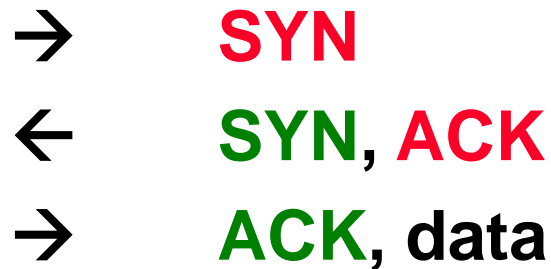
- **SYN** synchronise les numéros de séquence lors de l'établissement
- **FIN** libération
- **PSH** (*push*) envoi des données utiles
- **ACK** accusé de réception
- **RST** *reset* de la connexion

- Ce protocole orienté connexion comporte 3 phases :



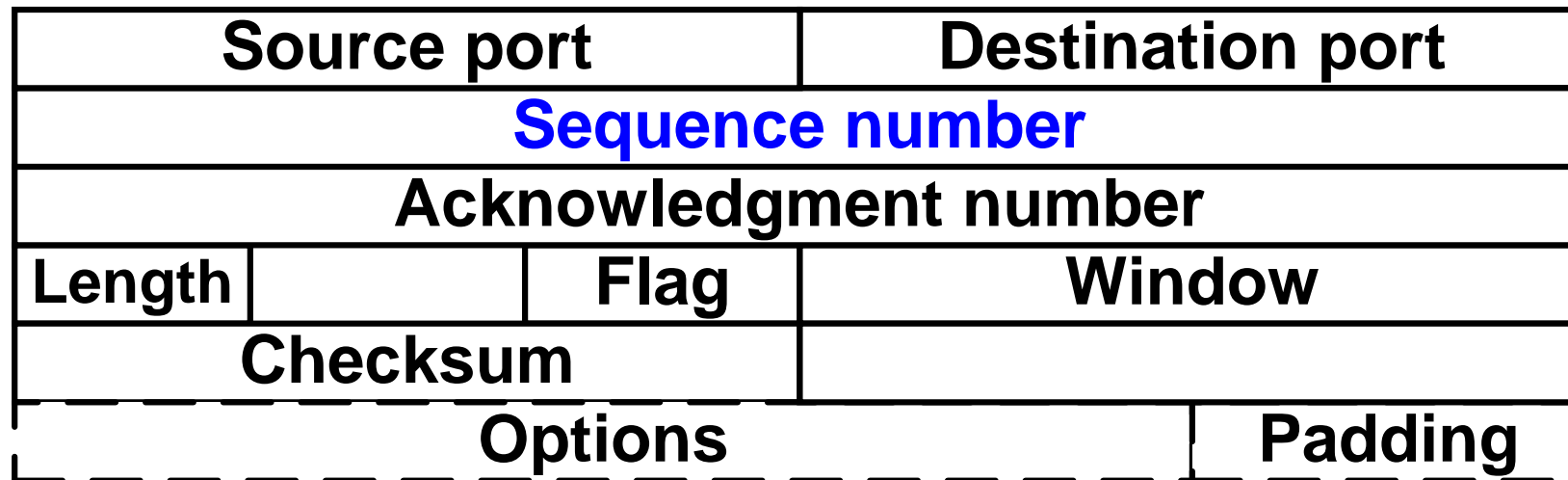
Client

Serveur



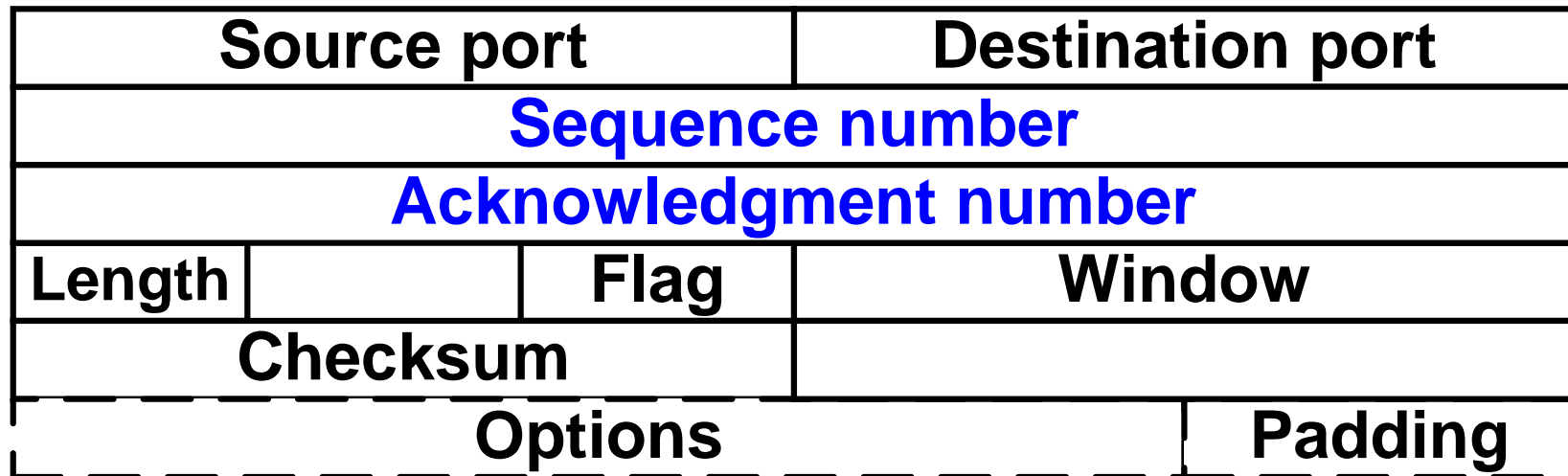
- Etablissement symétrique

- Il est orienté octet (*byte stream service*)
- Le numéro de séquence (*sequence number*) marque la position du premier octet du segment envoyé



- Valeur aléatoire fixée à l'établissement

- Numéro de séquence à l'émission (**Producteur**)
- Accusé de réception retourné par le **Consommateur**



Producteur



Consommateur

1000 bytes → PSH seq = 1
← ACK ack = 1000

1000 bytes → PSH seq = 1001

1000 bytes → PSH seq = 2001
← ACK ack = 3000

← PSH

Producteur

Producteur



Consommateur

Buffer vide

→ PSH seq = 1

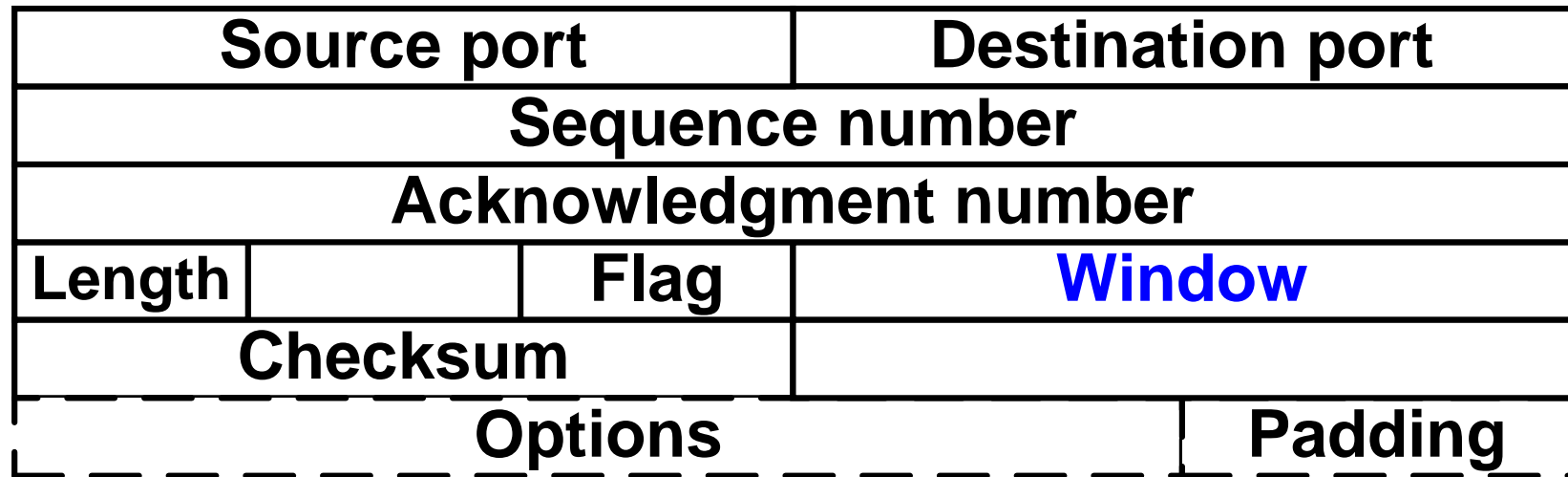
1000 bytes

← ACK ack = 1000

Buffer vide

- Le **Producteur** (TCP) doit conserver une copie dans l'attente d'une éventuelle retransmission
- **Buffers de contrôle d'erreur** situés du côté **Producteur**

- Champ **Window** indique le nombre d'octets que la station peut recevoir



Producteur



Consommateur

Buffer de 6000 bytes

← ACK win = 6000

1000 bytes → PSH seq = 1001

1000 bytes → PSH seq = 2001

- **Le Consommateur dispose d'un buffer de 6000 bytes**
- **Buffers de contrôle de flux situés du côté Consommateur**

Producteur



Consommateur

→ PSH seq=1 ack=1

win=2000

→ PSH seq=1001 ack=1

win=2000

→ PSH seq=2001 ack=1

win=2000

→ PSH seq=3001 ack=1

win=2000

← ACK ack=3000

win=5000

→ PSH seq=4001 ack=1

win=2000

octets déjà envoyés par A



octets reçus
par B

octets autorisés à être envoyés
par A

Transmission Control Protocol:

Source Port : 1308 Client

Destination Port : 80 Server web

Sequence number: 611644602

Acknowledgement: 3855596529

Header Length: 20 bytes

TCP flags:

..... ..0..... : Urgent pointer field not significant

..... ...1.... : Acknowledgment field significant

.....1... : Push function ON

.....0.. : Reset the connection OFF

.....0. : Synchronize sequence numbers OFF

.....0 : Expect more data from sender

Window: 65535

Checksum: 0xC20E (Good)

Transmission Control Protocol:

Source Port : 80 Server web

Destination Port : 1308 Client

Sequence number: 3855596529

Acknowledgement: 611644953

Header length: 20 bytes

TCP flags:

..... ..0..... : Urgent pointer field not significant

..... ...1.... : Acknowledgment field significant

.....1... : Push function ON

.....0.. : Reset the connection OFF

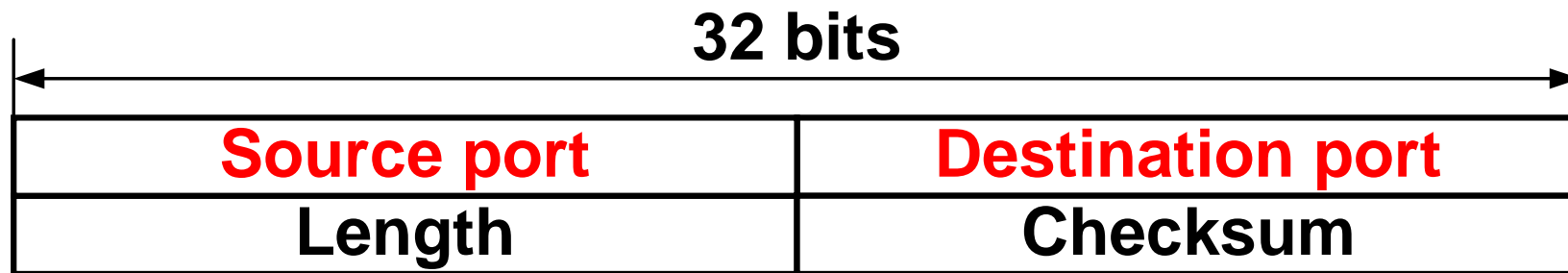
.....0. : Synchronize sequence numbers OFF

.....0 : Expect more data from sender

Window: 17520

Checksum: 0x6E6B (Good)

- UDP est un protocole simple **orienté datagramme**
Il fait ainsi partie des protocoles **sans connexion**
A l'inverse de TCP, il n'offre **aucune garantie de fiabilité**
- Champs de l'en-tête UDP :



- UDP utilise les mêmes valeurs de ports que TCP

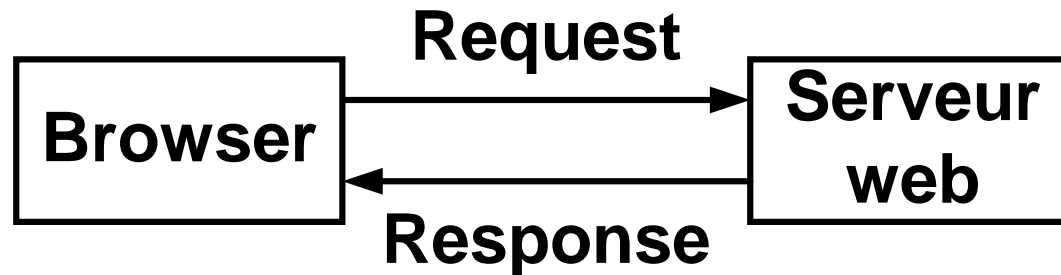
User Datagram Protocol:

Source port: 1311
Destination port: 53 DNS
UDP length: 40
Checksum: 0xEFD5 (Good)

User Datagram Protocol:

Source port: 53 DNS
Destination port: 1311
UDP length: 115
Checksum: 0xD12C (Good)

- Relation de type client – serveur
- Protocole HTTP (*HyperText Transfer Protocol*) utilisé entre navigateur (*browser*) et serveur web :



→ Get <http://www.td.unige.ch/default.html>

← Response

Request-Line =

Method SP Request-URI SP HTTP-Version CRLF

- **Example 1 with absolute URI**

```
GET http://www.google.ch/default.htm HTTP/1.1\r\n
```

```
Accept: */*\r\n
```

```
User-agent: ... (compatible; MSIE 6.0)\r\n
```

```
\r\n
```

Method	GET
Request-URI	http://www.google.ch/default.htm
HTTP-Version	HTTP/1.1

- **Example 2 with host header**

```
GET image1.gif HTTP/1.1\r\n
```

```
Host: www.td.unige.ch\r\n
```

```
Accept: */*\r\n
```

```
User-agent: ... (compatible; MSIE 6.0)\r\n
```

- **Exemple 3**

```
POST image1.gif HTTP/1.1\r\n
```

```
Host: www.td.unige.ch\r\n
```

```
Content-Length: XX
```

- **Exemple 4**

```
HEAD www.unige.ch HTTP/1.1\r\n
```

- **GET** Retrieve information identified by the URI
Used to retrieve an HTML document
- **HEAD** Retrieve meta-information about the URI
Used to find out if a document has changed
- **POST** Send information to a URI and retrieve result
Used to submit a form
- **PUT** Store information in location named by URI
- **DELETE** Remove *entity* identified by URI

Status-Line =

HTTP-Version SP Status-Code SP Reason-Phrase
CRLF

HTTP/1.1 200 OK

Date: Wed, 12 Nov 2003 14:44:36 GMT

Server: Apache/1.3.23

Last-Modified: Fri, 07 Nov 2003 17:15:03 GMT

Content-Type: text/html

Content-Length: 1145

<html><head><title>..</html>

Accept :	indique la liste des types de données supportées par le client
User-agent :	identifie le logiciel de navigation
Host :	indique le nom du serveur + numéro de port éventuel
Server :	identifie le logiciel serveur
Date :	date de génération de la réponse
Last-Modified :	date de la dernière modification

- **1xx** messages d'information (pas utilisé)

- **2xx** **Commande reçue et traitée par le serveur**
 - 200** Requête s'est déroulée correctement
 - 201** Requête OK et création d'un nouveau document
 - 202** Requête acceptée, traitement en cours
 - 204** Requête OK, mais aucune information à envoyer

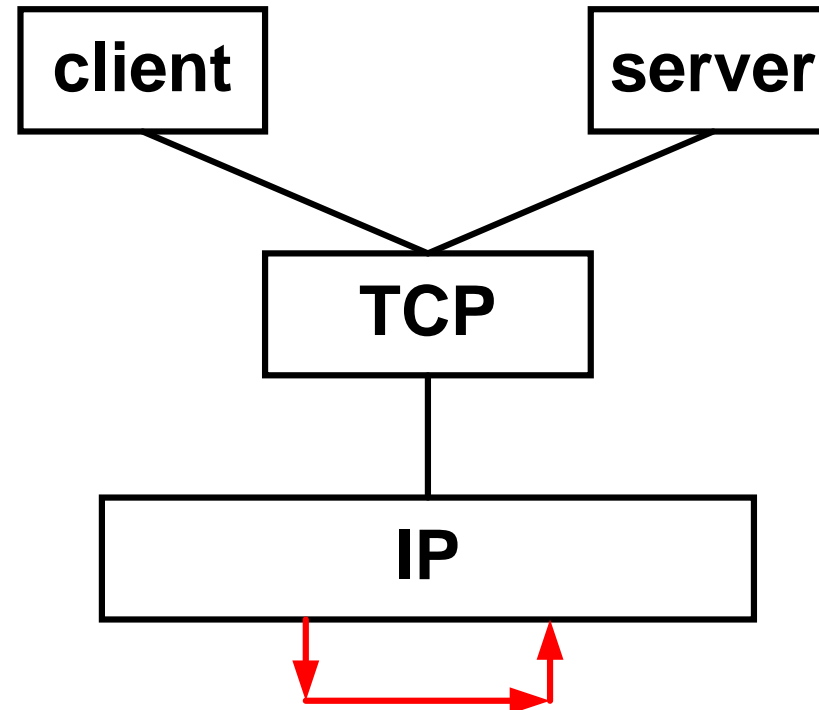
- **3xx** **Redirection, nouvelle requête nécessaire pour accéder au document demandé**
 - 301** Ressource demandée a été déplacée de façon permanente
 - 302** Ressource demandée a été déplacée de façon temporaire

- **4xx** **Erreur due au client**
 - 400** **Erreur de syntaxe**
 - 401** **Accès à la ressource exige une authentification**
 - 403** **Accès à la ressource interdit**
 - 404** **Ressource n'existe pas**

- **5xx** **Erreur due au serveur**
 - 500** **Erreur interne**
 - 503** **Serveur indisponible**

- §3 **Protocoles TCP et http** **40 min**
- §4 **Wireshark en mode statistique** **10 min**
Apprentissage
- §5 **Wireshark en mode statistique** **20 min**
Trafic EIG
- §6 **Travail personnel** **20 min**

- Interface de bouclage permettant à client et serveur de communiquer



```
C:\WINDOWS\System32\drivers\etc\hosts  
127.0.0.1 localhost
```