La couche transport dans Internet

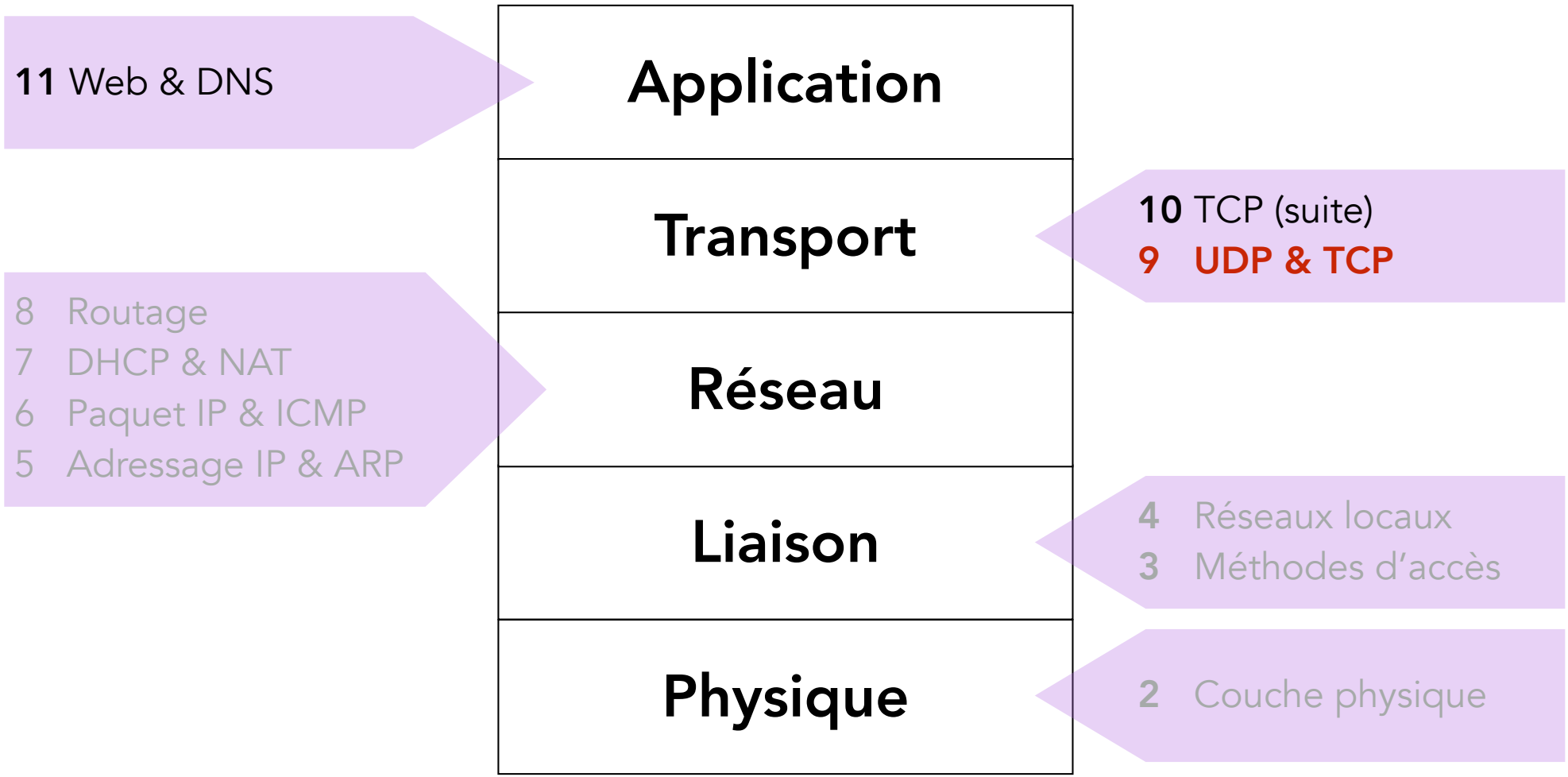
UE LU3IN033 Réseaux

2023-2024

Bruno Baynat

Bruno.Baynat@sorbonne-universite.fr

Programme de l'UE LU3IN033



Plan du cours

- Rôle de la couche transport
- Conception des applications réseau
  - Modèle client-serveur vs modèle P2P
- Classification des besoins des applications
  - Fiabilité, bande passante, délai, sécurité
- Les deux protocoles de la couche transport dans Internet
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)
- Les principes sous-jacent aux services de la couche transport
  - (Dé-)multiplexage
  - Détection des erreurs et des pertes
  - Livraison fiable
  - Contrôle de flux
  - Contrôle de congestion

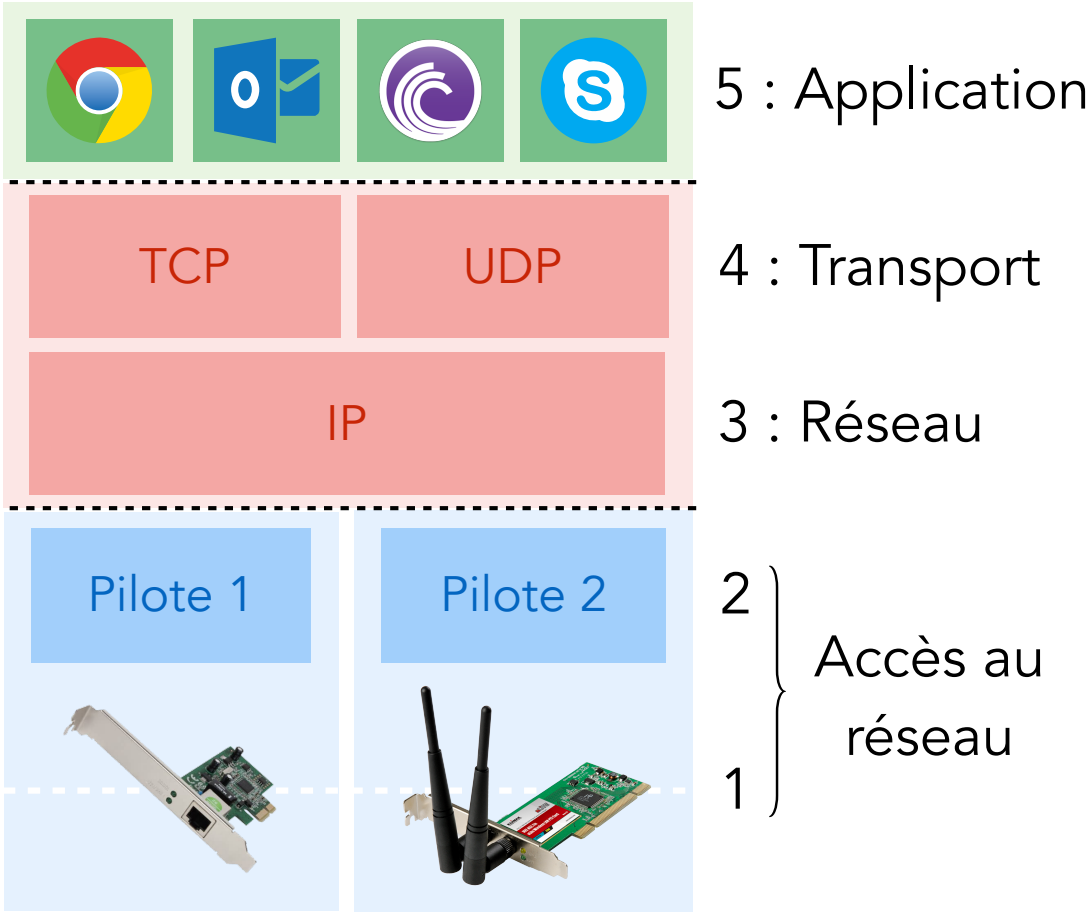
Deux protocoles de transport : UDP et TCP

Pile protocolaire TCP/IP

- Les machines hôtes exécutent des applications
  - qui communiquent selon un protocole de couche 5
  - les données qu'elles génèrent sont passées aux couches inférieures
- Les routeurs acheminent les données pour le compte des applications
  - les données sont transportées dans des paquets IP selon un protocole de couche 3

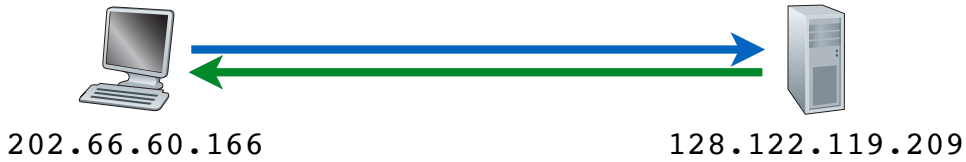
Pourquoi n'est-ce pas suffisant ?

A quoi sert la couche 4 ?

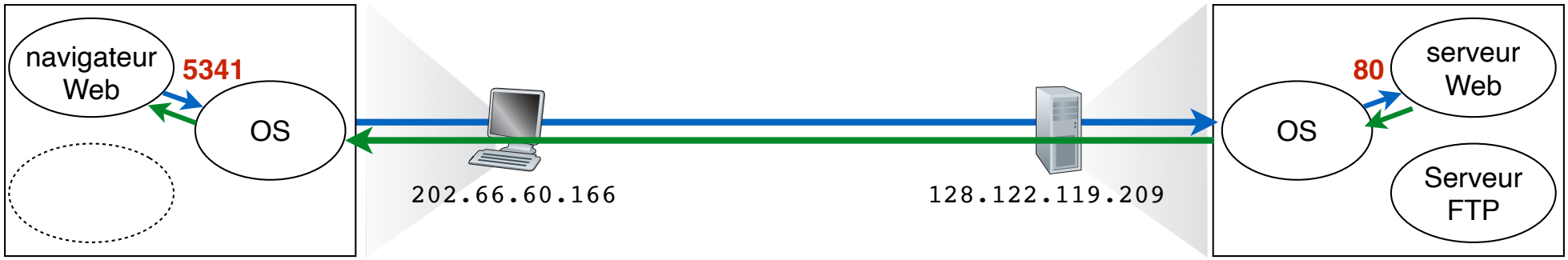


Première raison

- La couche IP (réseau) a pour rôle de transmettre des paquets de données d'une machine source vers une machine destination
  - de proche en proche
  - en suivant le « meilleur » chemin
- Elle offre donc un service de remise « de machine à machine »



- La couche transport étend le service offert par la couche réseau en un service de remise « d'application à application »



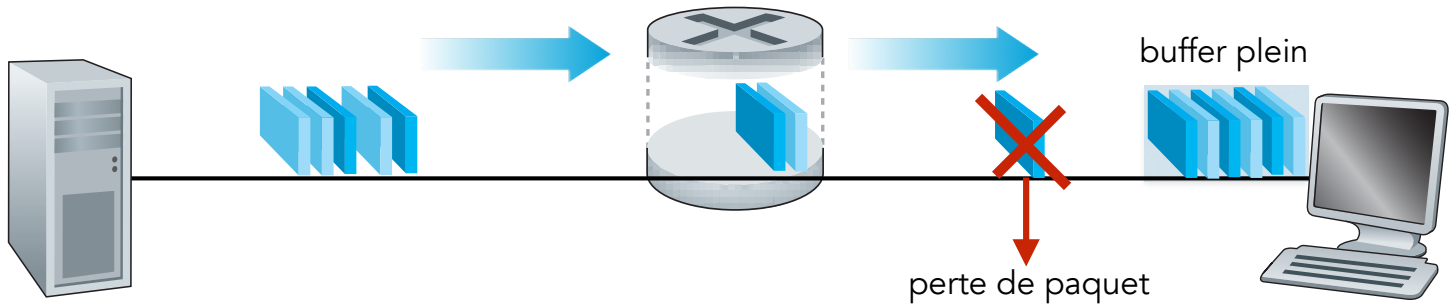
- C'est le rôle des ports

Deuxième raison

- La couche IP (réseau) n'est pas fiable
- Lors de son acheminement de la source vers la destination, un paquet IP peut être
  - perdu
    - buffers saturés
    - problème de routage
  - en erreur
    - trame encapsulant le paquet en erreur
  - déséquencé
    - changement de routes
  - dupliqué
    - erreur de la couche liaison
- Certaines applications ont besoin de fiabilité
- Pour ces applications, la couche transport propose un service fiable de transfert de données

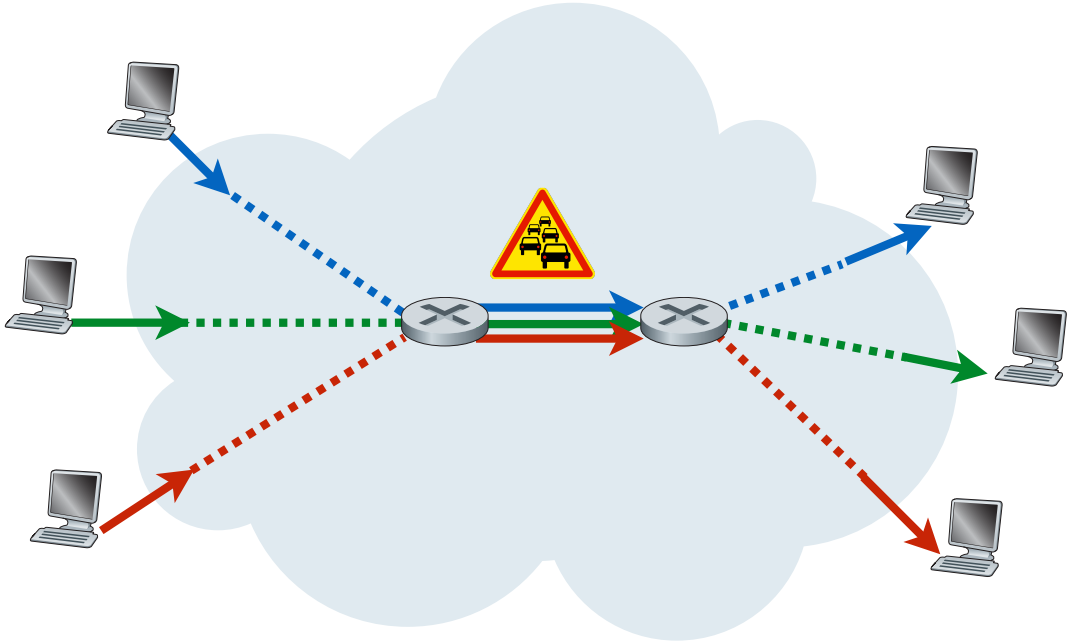
Troisième raison

- La couche IP (réseau) ne fait pas de contrôle de flux
- Elle ne permet pas à l'émetteur de s'adapter au rythme du récepteur
  - risque d'engorgement des buffers du récepteur
  - perte de données possible
- Pour les applications qui en ont besoin, la couche transport ajoute à la couche IP un mécanisme de contrôle de flux



Quatrième raison

- La couche IP (réseau) ne fait pas de contrôle de congestion
- Elle ne permet pas d'équilibrer les flux entre différents émetteurs
  - congestion possible d'une partie du réseau
  - lenteur du réseau (ou perte de données dans le cas extrême)
- Pour les applications qui en ont besoin, la couche transport ajoute à la couche IP un mécanisme de contrôle de congestion



Protocoles de transport

- Il y a 2 protocoles de transport au dessus d'IP : UDP (User Datagram Protocol) et TCP (Transmission Control Protocol)
- Pourquoi ?
  - Toute les applications n'ont pas besoin de fiabilité
    - certaines ont au contraire besoin de rapidité
    - Ex : Streaming vidéo
  - Le contrôle de flux est incompatible avec certaines applications
    - applications temps réel
    - Ex : Voix sur IP, Jeux en réseau
  - Le contrôle de congestion ne peut pas être imposé à certaines applications
    - applications qui génèrent des échanges sporadiques nécessitant une réponse rapide
    - Ex : DNS

Classification des applications

Application	Pertes	Bande passante	Délai
Transfert de fichiers	pas de tolérance	élastique	non sensible
Email	pas de tolérance	élastique	non sensible
Documents Web	pas de tolérance	élastique	non sensible
Audio/video temps réel	tolérance	audio: 5kb/s-1Mb/s video:10kb/s-5Mb/s	100aines msec
Streaming audio/video préenregistré	tolérance	audio: 5kb/s-1Mb/s video:10kb/s-5Mb/s	qques sec
Jeux interactifs	tolérance	qques kbps	100aines msec

TCP

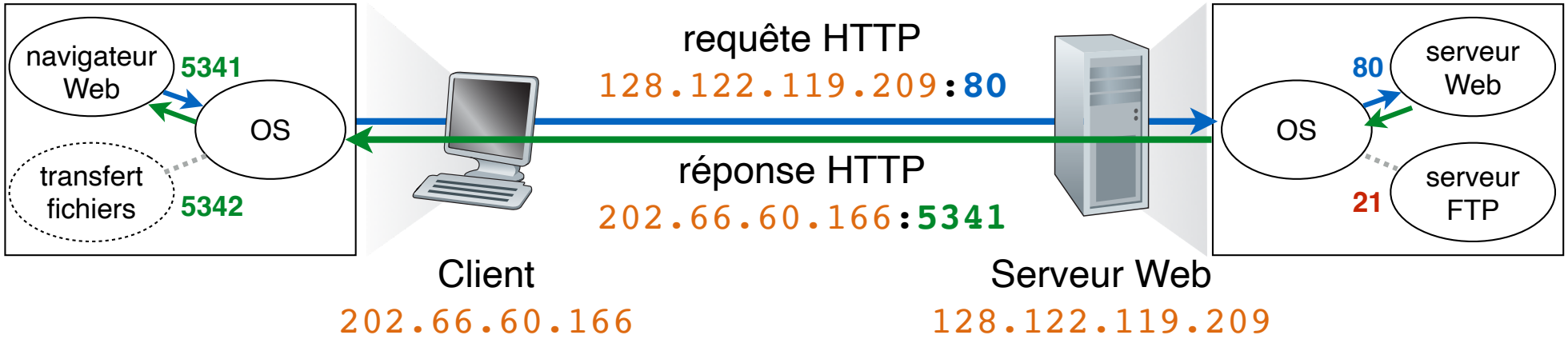
UDP

UDP vs TCP

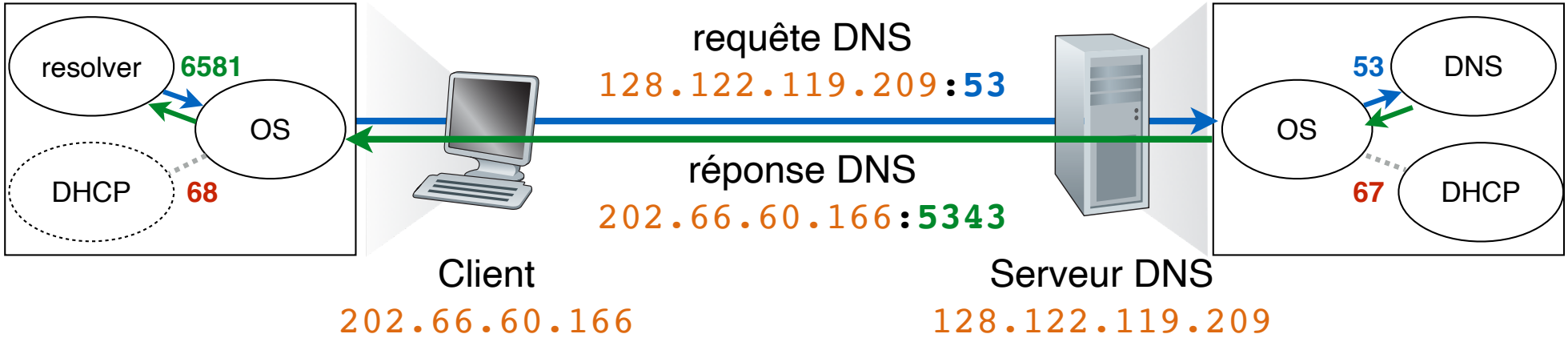
- Deux protocoles de transport : UDP et TCP
  - TCP offre un service de livraison fiable / UDP est sans garantie
  - TCP fonctionne en mode flux d'octet / UDP en mode datagramme
  - TCP est orienté connexion / UDP est sans connexion
- Services de base communs à UDP et TCP
  - Multiplexage / Démultiplexage
    - identification des processus exécutés sur les machines d'extrémités
    - numéros de port source et destination
  - Détection d'erreurs
    - somme de contrôle sur l'entête et les données transportées
- Services spécifiques à TCP
  - Retransmission des octets en erreur ou perdus
  - Remise en séquence des octets reçus dans le désordre
  - Suppression des octets dupliqués
  - Contrôle de flux
  - Contrôle de congestion

Service commun : (Dé-)Multiplexage

TCP



UDP



Modèle Client-Serveur vs P2P

Client-Serveur

- Conception asymétrique
  - l'application réseau résulte de l'exécution de deux programmes différents
- Le serveur
  - s'exécute en permanence
  - attend les requêtes des clients
  - écoute sur un numéro de port connu des clients
- Les clients
  - se connectent par intermittence
  - initient la communication en envoyant des requêtes au serveur
  - doivent connaitre l'adresse IP et le numéro de port du serveur

Pair-à-Pair

- Conception symétrique
  - tous les noeuds exécutent le même programme
  - les noeuds sont des clients qui peuvent agir comme des serveurs pour d'autres pairs
- Service de découverte des pairs
  - les applications P2P nécessitent un mécanisme de découverte
    - de l'adresse IP des pairs
    - du numéro de port qu'ils utilisent
  - un serveur (!), une base de données répartie sur les noeuds, …

Modèle Client-Serveur

Client

- Les clients
  - ont une adresse IP dynamique
    - qui change en fonction du temps
    - et de leur localisation
  - utilisent un numéro de port arbitraire (> 1024)
    - codé sur 16 bits (2 octets)
    - dont le choix est laissé au système d'exploitation
- Les clients découvrent
  - le nom (URL) d'un serveur
    - moteurs de recherche
  - l'adresse IP d'un serveur
    - DNS (*Domain Name Server*)
    - pages blanches de l'Internet

Serveur

- Les serveurs
  - ont une adresse IP statique
  - ont un nom connu des clients
    - moteurs de recherche
  - écoutent sur un numéro de port connu des clients (≤ 1024)
    - well-known ports
      - 80 pour HTTP (web)
      - 53 pour DNS
      - 67 pour DHCP
- Les serveurs sont contactés par les clients
  - et découvrent ainsi
    - leur adresse IP
    - le numéro de port qu'ils ont choisi

Service commun : détection d'erreur

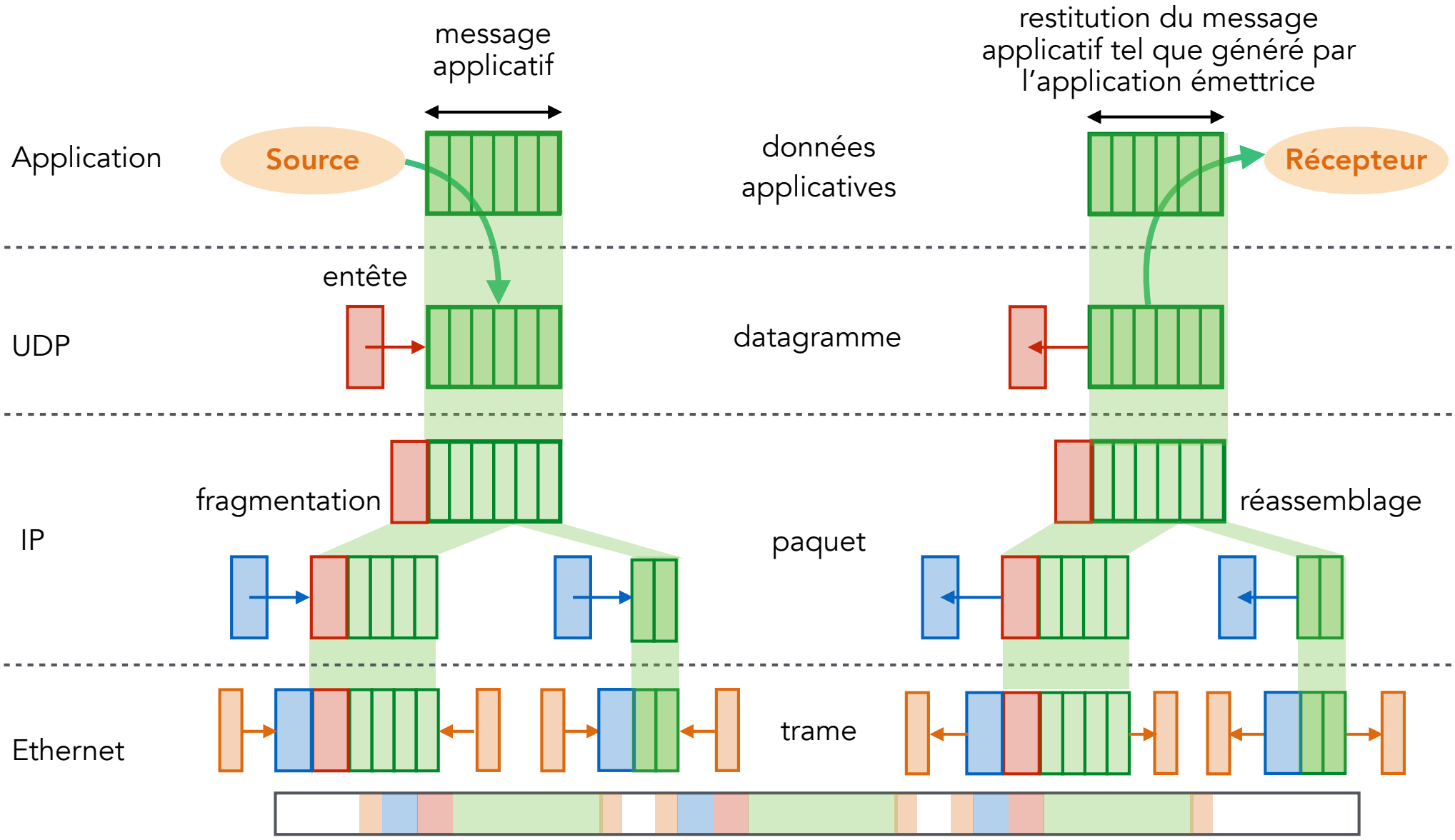
- UDP et TCP proposent tous les deux un mécanisme de détection d'erreur
  - obligatoire pour TCP
  - optionnel pour UDP
- Champ Checksum dans l'en-tête (du segment TCP ou du datagramme UDP)
  - somme de contrôle calculée par l'émetteur
  - vérifiée par le récepteur
  - portant
    - sur l'en-tête (du segment ou du datagramme)
    - sur les données transportées
    - sur un « pseudo-entête » reprenant des données d'IP (essentiellement @IP source et @IP destination)
- UDP ne propose aucun mécanisme pour réparer les erreurs détectées
  - sur détection d'erreur les données sont ignorées
- TCP répare les erreurs
  - retransmission des données en erreur

UDP vs TCP

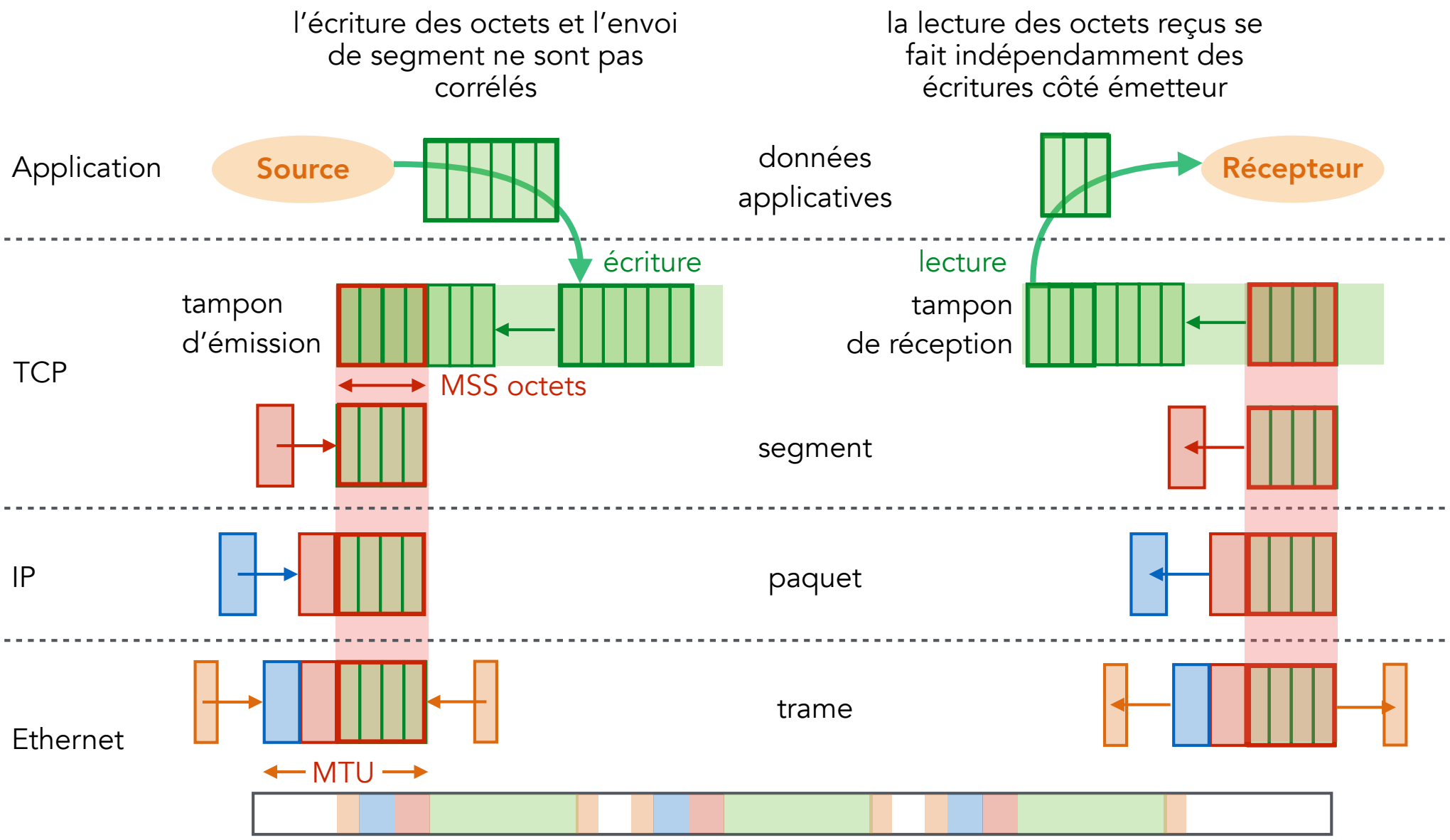
Mode datagramme vs Mode flux d'octets

- UDP : mode datagramme
  - Les messages applicatifs sont encapsulés tels quels dans des datagrammes
  - Les datagrammes sont envoyés immédiatement
  - IP fragmente éventuellement les datagrammes en fonction de la MTU locale
  - L'application côté récepteur reçoit les messages applicatifs tel que générés côté émetteur
- TCP : mode flux d'octets
  - Côté émetteur, TCP attend
    - que l'application génère MSS octets pour remplir une trame complète (MTU)
    - ou l'expiration d'un temporisateur pour former un « petit » segment
  - TCP encapsule ces octets dans un segment et attend le moment opportun pour l'envoyer
    - sans engorger le récepteur
    - sans congestionner le réseau
  - IP n'a pas de raison de fragmenter les paquets encapsulant des segments TCP
  - Côté récepteur, TCP écrit les octets reçus dans un tampon le temps que l'application les lise

UDP : Mode Datagramme



TCP : Mode Flux d'octets

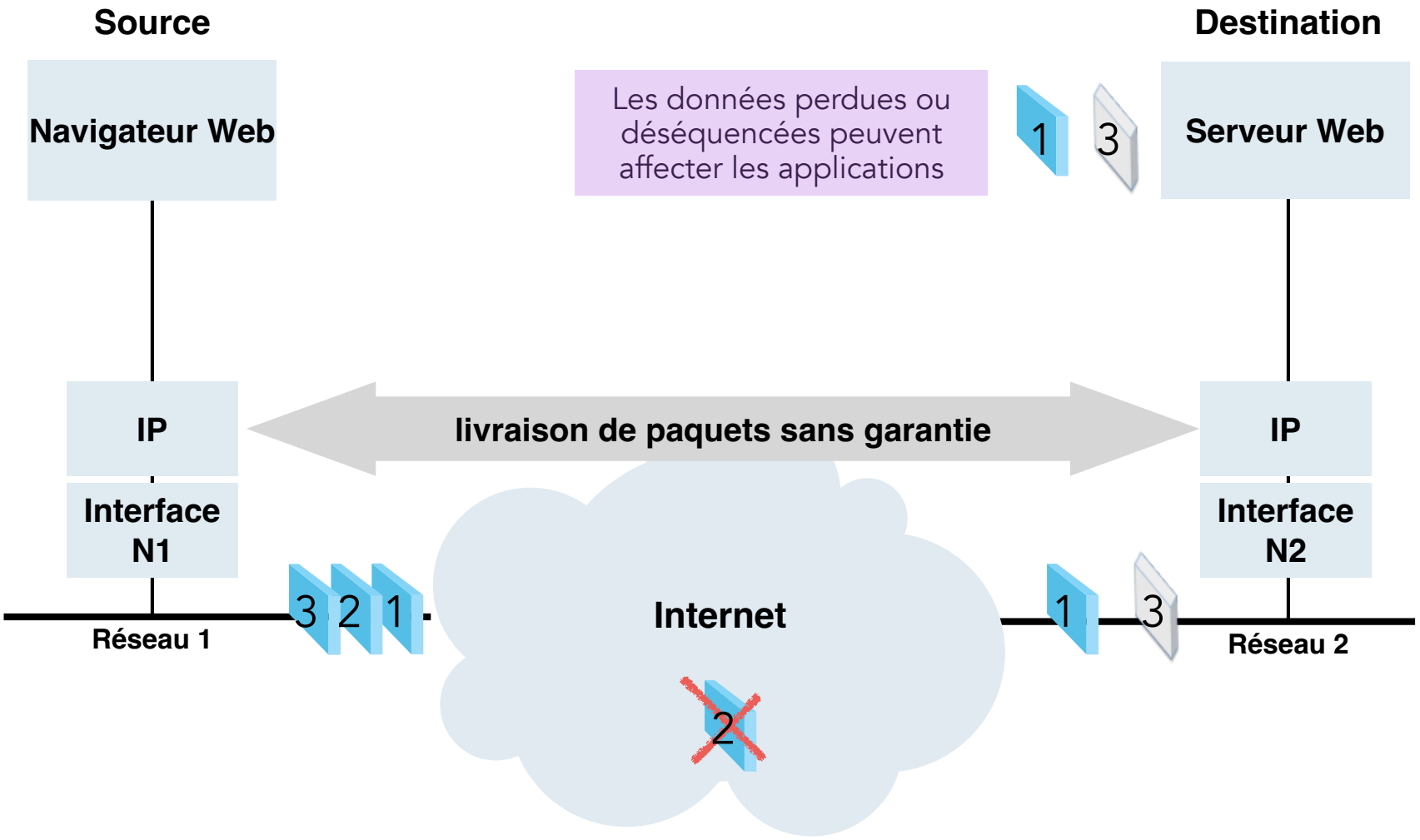


UDP vs TCP

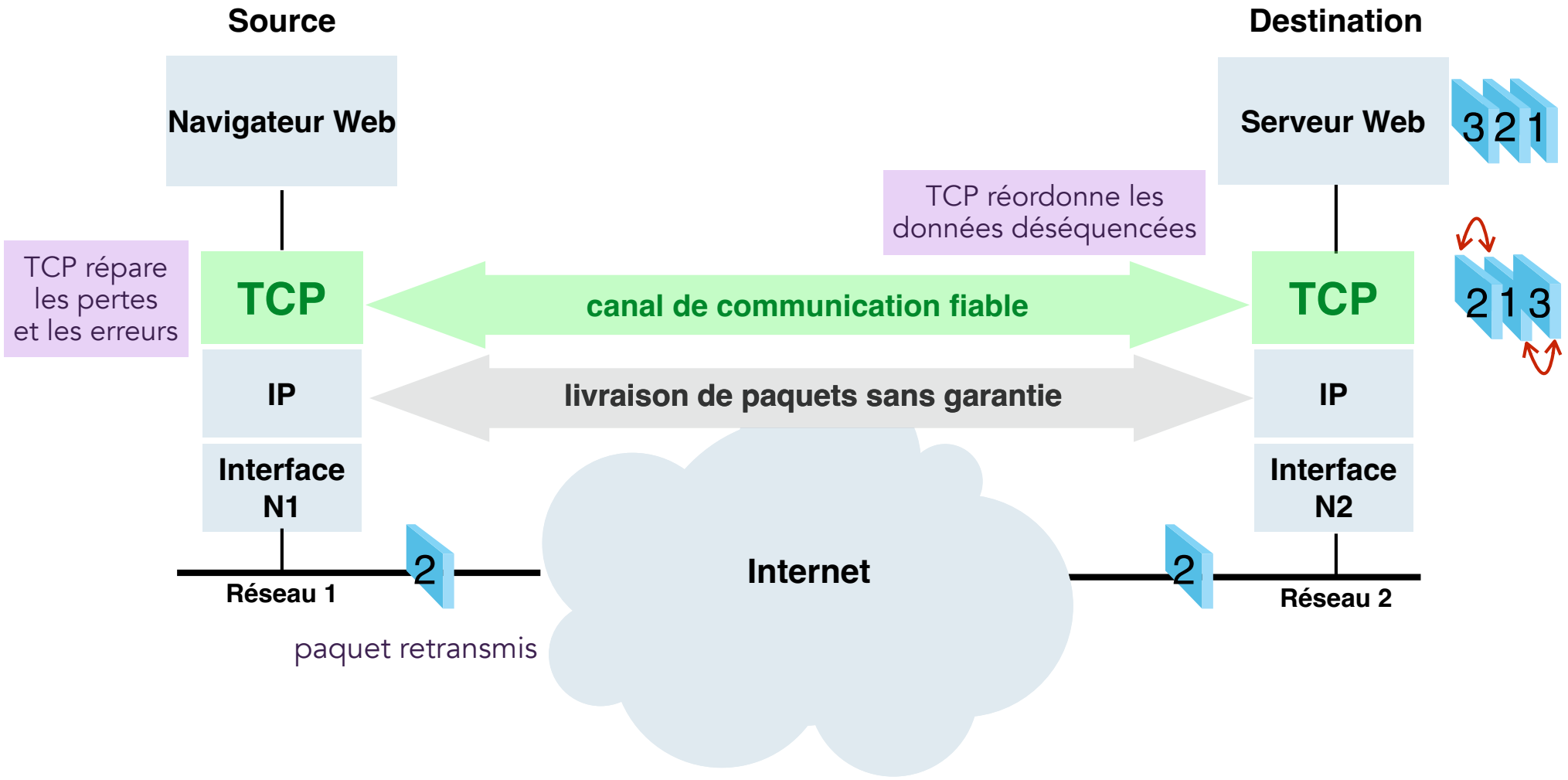
Mode non connecté vs Mode connecté

- UDP : mode non connecté
  - Envoi des données sans attendre l'établissement d'une connexion
    - adapté pour des échanges simples de type « une requête-une réponse »
  - Sans installer d'états
    - pas de tampons, pas de numéros de séquence, pas de fenêtres, pas de temporisateurs
- TCP : mode connecté
  - 3 phases
    1. Etablissement de la connexion
    2. Transfert de données
    3. Libération de la connexion
  - Installation et maintien d'états
    - tampons, numéros de séquence, fenêtres, temporisateurs, …

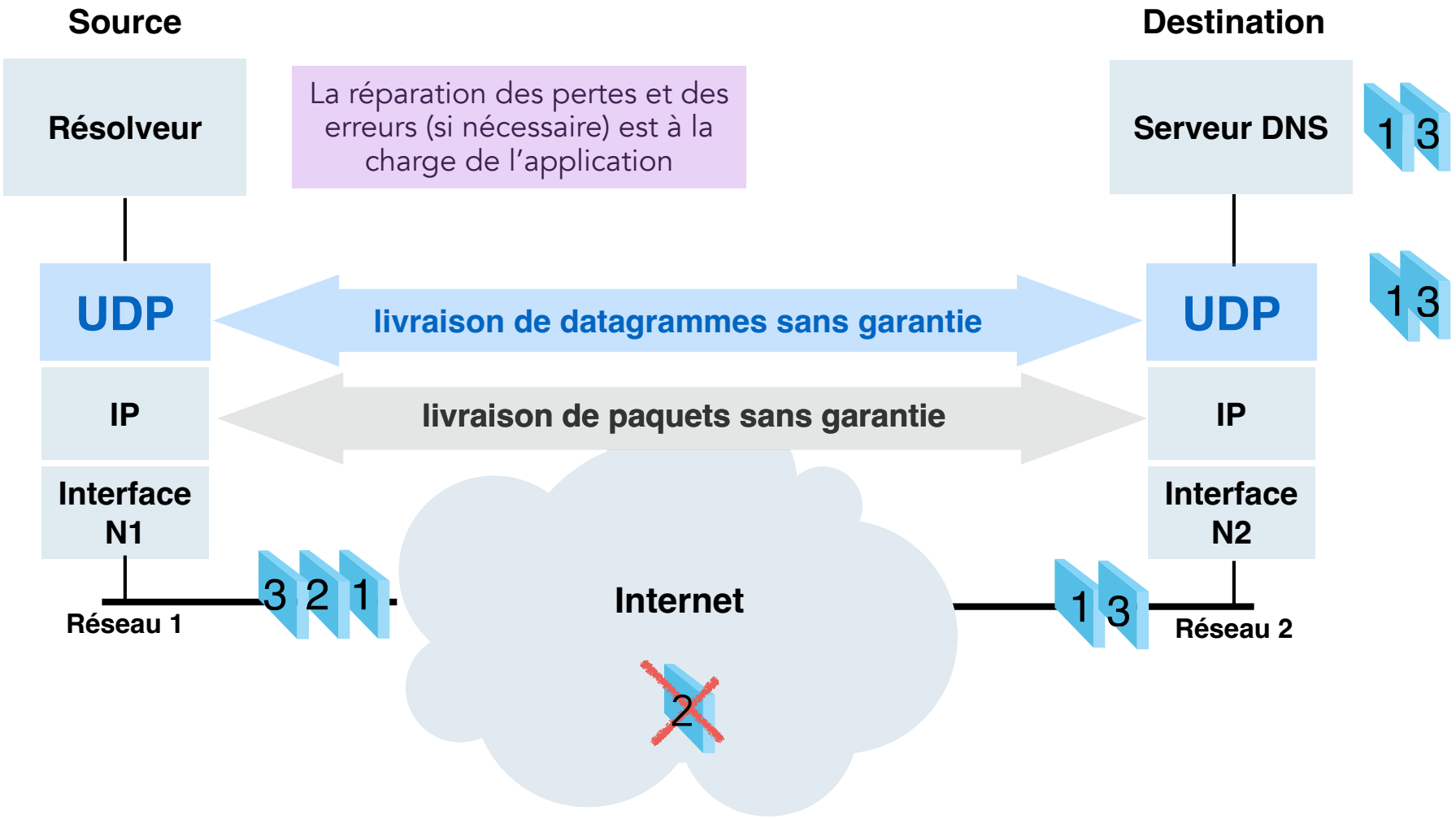
IP : Acheminement de proche en proche sans garantie



TCP : livraison fiable de bout en bout entre processus distants



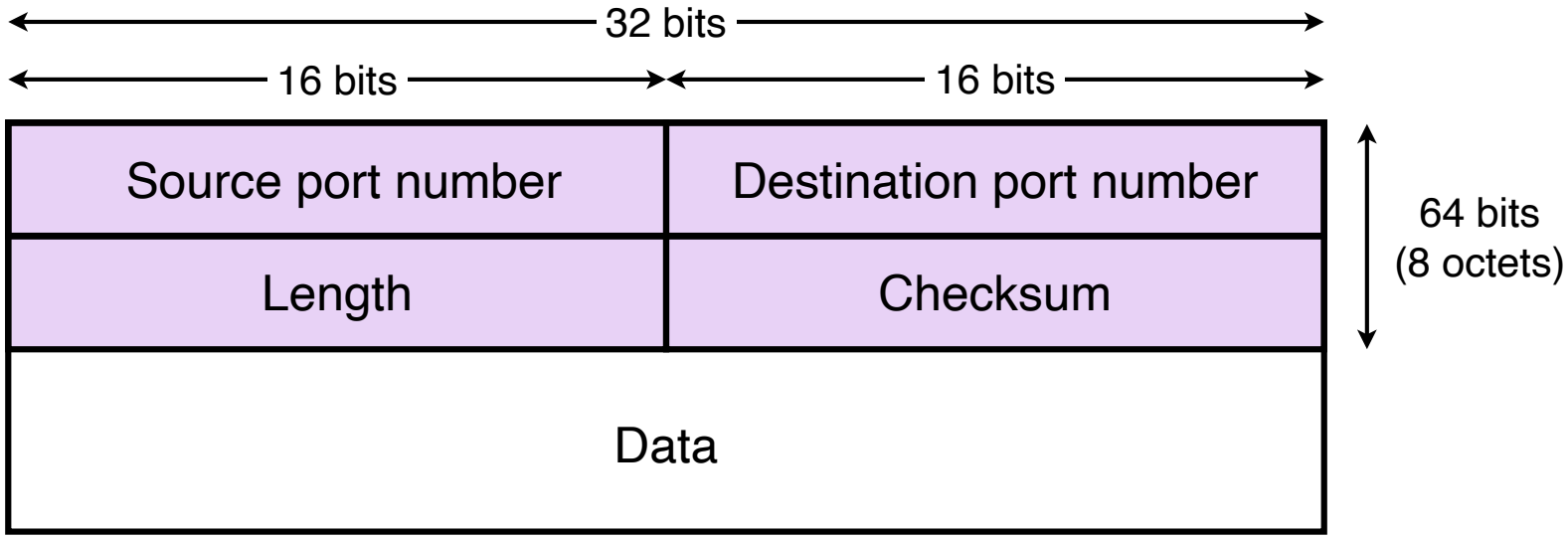
UDP : livraison de bout en bout sans garantie entre processus distants



UDP
User Datagram Protocol

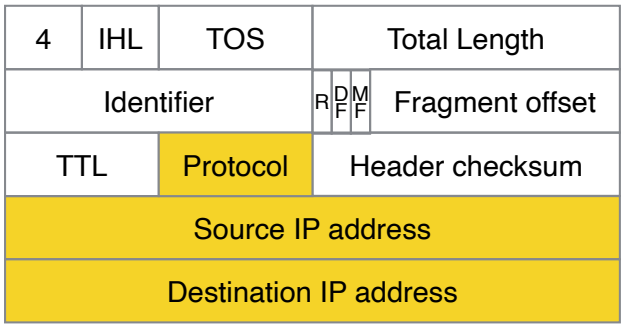
UDP : User Datagram Protocol

- Service de livraison de datagrammes simples
  - Numéros de port : identification des applications en communication
  - Somme de contrôle : détection des datagrammes en erreur
  - Longueur : pour accélérer la remise des messages à l'application destinataire

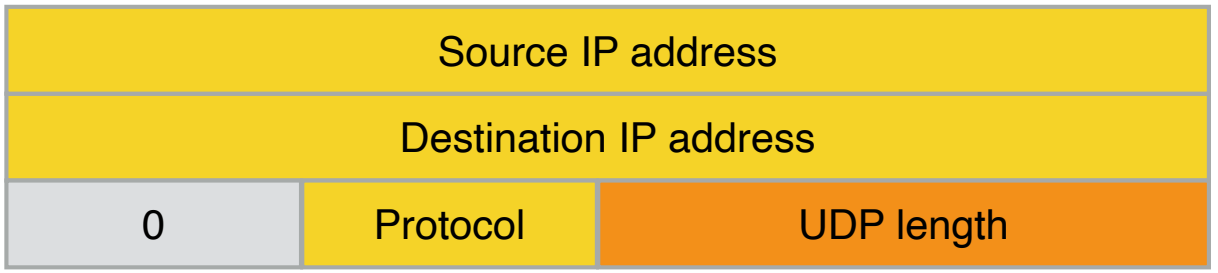


Pseudo entête et somme de contrôle

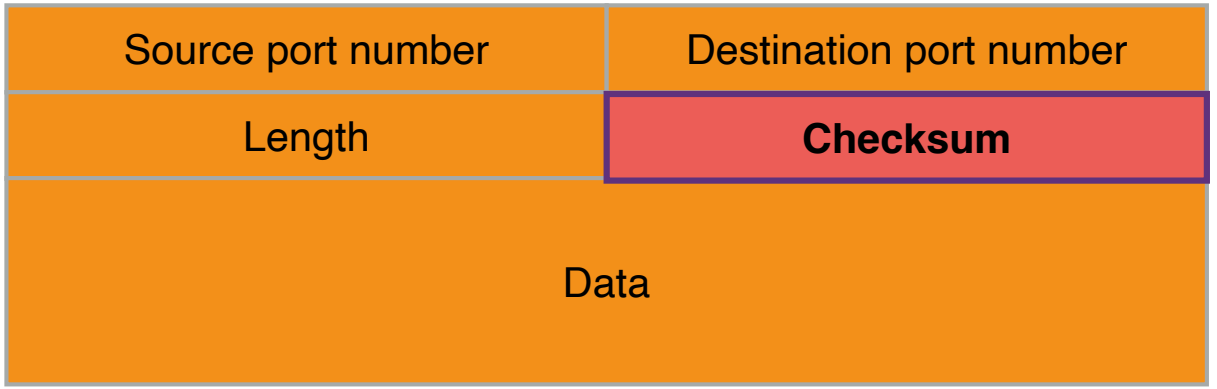
paquet IP



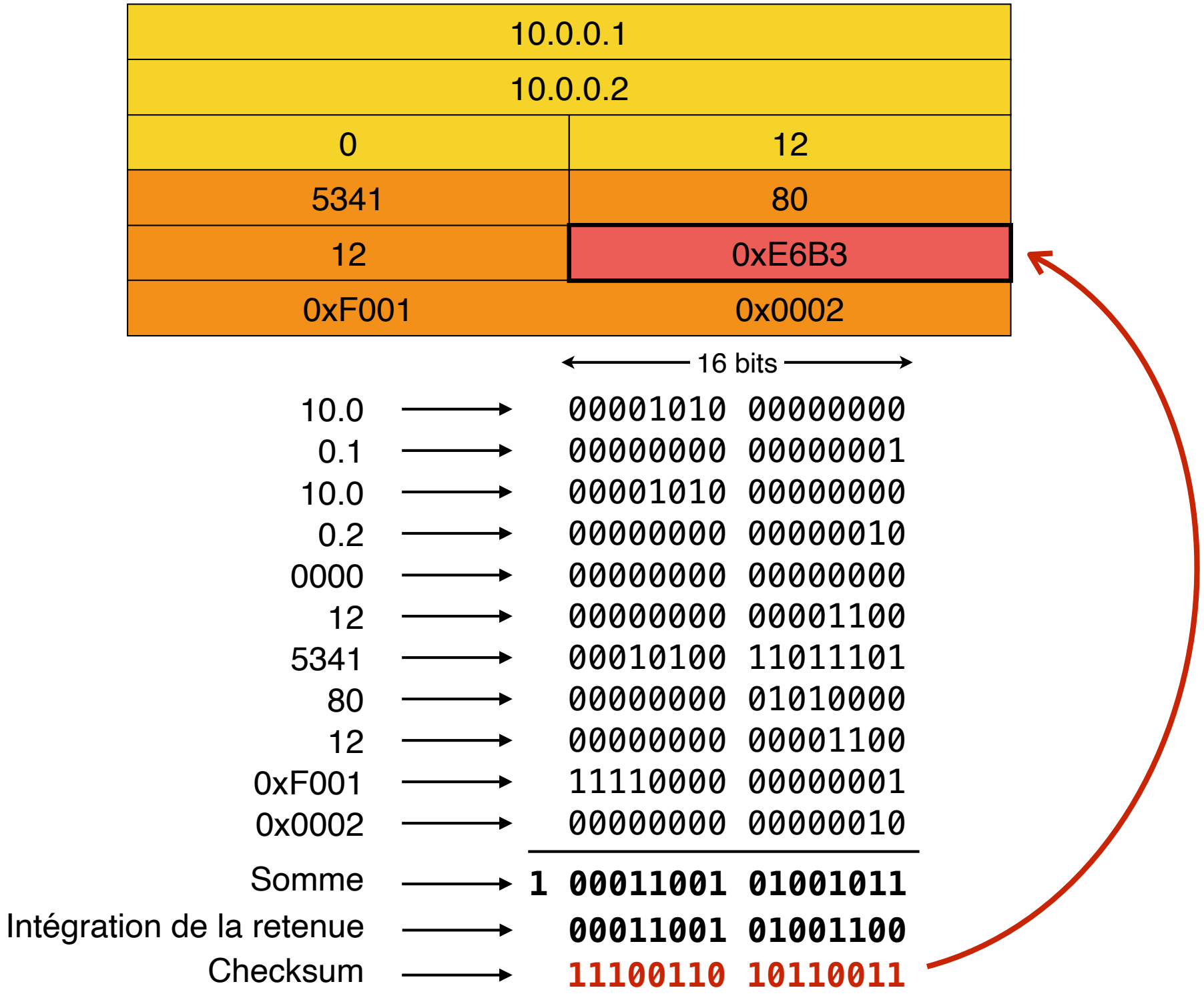
pseudo entête



datagramme UDP



Le pseudo-entête UDP (et TCP) permet la double-vérification des informations IP


10.0.0.1
10.0.0.2
0
12
5341
80
12
0xE6B3
0xF001
0x0002
16 bits
10.0 00001010 00000000
0.1 00000000 00000001
10.0 00001010 00000000
0.2 00000000 00000010
0000 00000000 00000000
12 00000000 00001100
5341 00010100 11011101
80 00000000 01010000
12 00000000 00001100
0xF001 11110000 00000001
0x0002 00000000 00000010
Somme 1 00011001 01001011
Intégration de la retenue 00011001 01001100
Checksum 11100110 10110011

UDP : avantages et inconvénients

- UDP offre un service simple de livraison de datagrammes
  - entre applications : (dé-)multiplexage des datagrammes grâce aux numéros de port
  - avec détection (optionnelle) des datagrammes en erreur : somme de contrôle
- Inconvénients
  - UDP ne résout pas les problèmes de fiabilité d'IP
    - les pertes et les erreurs ne sont pas rattrapées
    - les datagrammes (si reçus) peuvent l'être dans le désordre
  - UDP ne se soucie pas de l'engorgement éventuel du récepteur
    - pas de contrôle de flux
  - UDP ne se soucie pas de la congestion éventuelle du réseau
    - pas de contrôle de congestion
- Avantages
  - UDP envoie immédiatement les données de l'application (après ajout de l'en-tête)
  - UDP évite la complexité et les délais nécessaires à la fiabilisation des échanges
    - mode non connecté : pas de phase d'établissement de la connexion
    - pas d'états, pas de mémoire nécessaire
- UDP est destiné aux applications qui privilégie la rapidité de remise des données à la fiabilité
  - « Mieux vaut jamais que tard »

Applications populaires qui utilisent UDP

- Applications multimédia
  - La fiabilité n'est pas compatible avec les applications interactives
    - les retransmissions retardent la réception des données
  - Ces applications sont peu sensibles aux pertes
    - Ex : appels téléphoniques, visioconférences, jeux en ligne, IPTV, …
- Applications simples de type « une requête-une réponse »
  - Les états nécessaires par connexion ne sont soit pas possibles soit trop coûteux à mettre en place
    - si l'adresse des clients n'est pas encore connue
    - si les clients se connectent en grand nombre au même serveur
  - Ex : DHCP, DNS, …
- Applications nécessitant une réponse très rapide
  - Ex : serveurs de localisation, serveur de temps (NTP)

A faire

- Cours 9
  - à relire attentivement
- Devoir 9 sur Moodle
  - date de rendu : dimanche 19 novembre