Option Informatique et Science du Numérique

Réseau - niveau 1

Alexandre Guitton

UNIVERSITE BLAISE PASCAL CLERMONT-FERRAND UBP

Conventions

- Magenta : exemples de la vie quotidienne
- Vert : liens avec d'autres parties du cours
- Étoile () : schéma hors support
- Turquoise : commandes et outils

Plan

- 1. Définitions
- 2. Réseaux familiaux
- 3. Réseaux locaux
- 4. Réseaux mondiaux
- 5. Web
- 6. Internet

1. Définitions

1. Définitions

- Réseau
  - interconnexion de plusieurs entités
  - maths : graphe orienté valué
- Exemples
  - réseau d'électricité
  - réseau téléphonique
  - réseau d'amis
  - réseau de distribution d'eau

1. Définitions

- Communication
  - *unicast* = un à un
  - *broadcast* = un à tous
  - *multicast* = un à plusieurs, plusieurs à plusieurs
- Exemples
  - conversations téléphoniques
  - télévision, affiches publicitaires, alarmes

1. Définitions

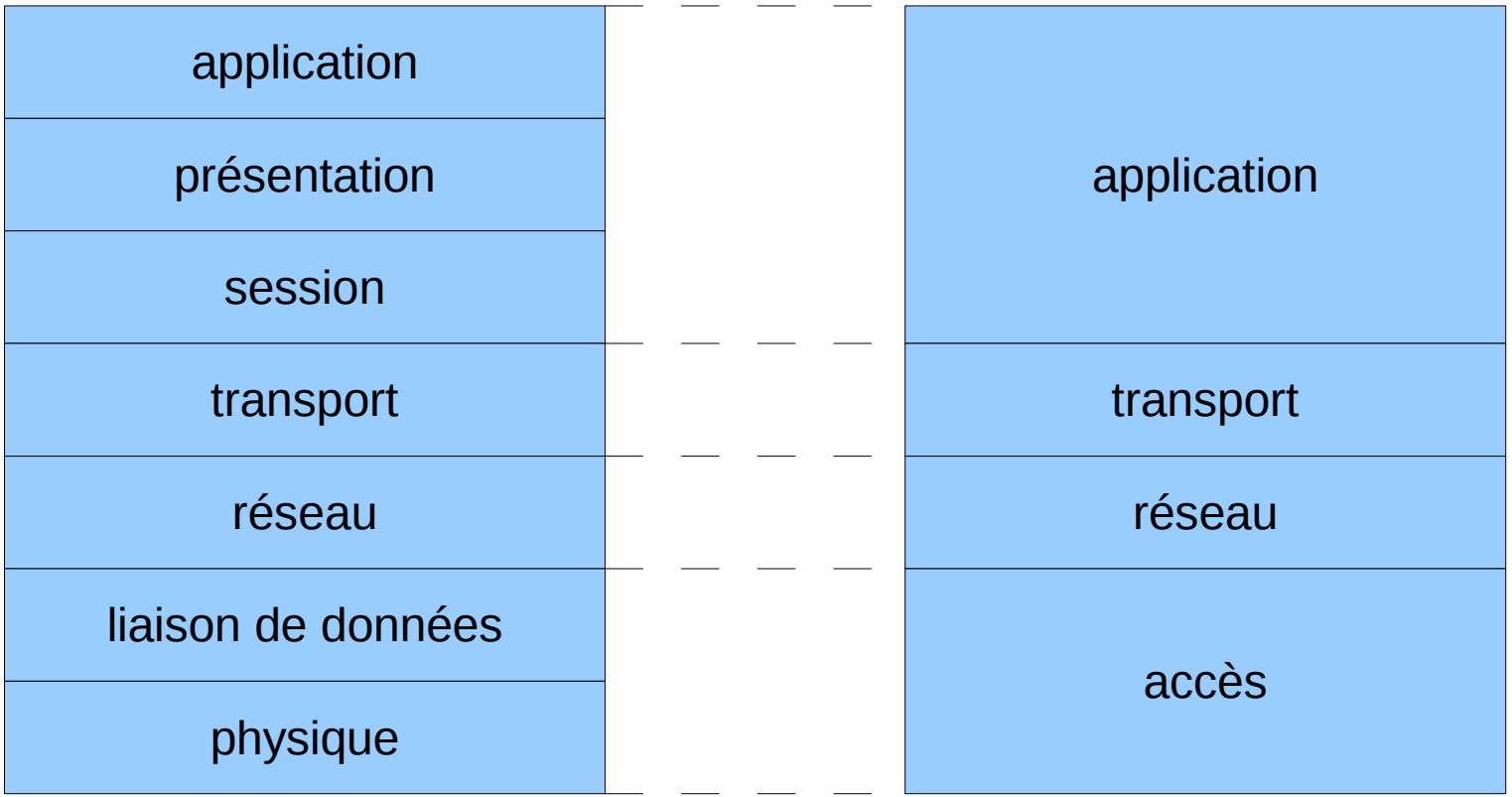
- Protocole
  - règles communes à plusieurs machines
  - algorithmique : algorithme distribué
  - vie quotidienne : langue (français, anglais, etc.)
- Types de protocoles
  - communication : temps-réel (textuelle, audio, vidéo), asynchrone (emails, SMS, courriers)
  - gestion : découverte des voisins, notification d'erreurs

1. Définitions

- Couche protocolaire
  - ensemble de protocoles fournissant un service donné
- Pile protocolaire
  - ensemble de couches protocolaires nécessaires pour communiquer

1. Définitions

- Pile protocolaire OSI (historique) vs pile protocolaire Internet



1. Définitions

- Couche physique (couche 1)
  - objectif : si l'émetteur envoie un 0 ou un 1, le récepteur reçoit le 0 ou le 1
  - exemples : optique (visible, infra-rouge), acoustique, électrique, électromagnétique
  - techniques : modulation, correction d'erreurs

1. Définitions

- Couche liaison de données (couche 2)
  - objectif : quand un émetteur parle, le récepteur l'écoute + transfert de plusieurs 0 et 1
  - exemples : conversation téléphonique, conversation en groupe
  - techniques : découpage temporel, droit de parole, détection de collisions

1. Définitions

- Couche réseau (couche 3)
  - objectif : interconnexion indirecte
  - exemples : conduite en voiture, prise de transports en commun
  - techniques : transmission de proche en proche, calcul du plus court chemin

1. Définitions

- Couche transport (couche 4)
  - objectif : contrôle de la communication de bout en bout
  - exemples : « Pourriez-vous parler moins vite ? »
  - techniques : retransmissions, acquittements

1. Définitions

- Couche session (couche 5)
  - objectif : gestion des connexions et déconnexions
  - exemples : reconnexion automatique à certains sites
  - techniques : temps d'expiration, point de reprise

1. Définitions

- Couche présentation (couche 6)
  - objectif : définition de la manière dont les données sont représentées pour être transférées
  - exemples : guillemets autour des chaînes de caractère, caractère terminal '\0', taille de la chaîne précédant la chaîne
  - techniques : encodage, compression, chiffrement, déchiffrement

1. Définitions

- Couche application (couche 7)
  - objectif : fournir un service particulier
  - exemples : réseaux sociaux, courses en ligne
  - techniques : programmation client-serveur

1. Définitions

- Exemple 
  - Alice veut aller chez Bob pour réviser : couche 7
  - Alice doit marcher, prendre le bus numéro 3, puis marcher à nouveau : couche 3
  - en marchant, Alice doit regarder de chaque côté de la rue avant de traverser : couche 2
  - en marchant, Alice doit mettre un pied devant l'autre : couche 1
  - avant de prendre le bus, Alice doit attendre qu'il arrive : couche 2
  - etc.

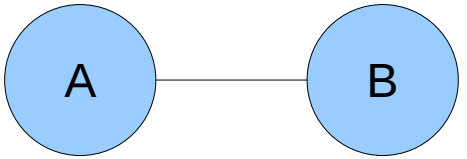
2. Réseaux familiaux

2. Réseaux familiaux

- Topologie point à point
- Topologie en bus
- Topologie en étoile
- Approfondissements

2.1 Topologie point à point

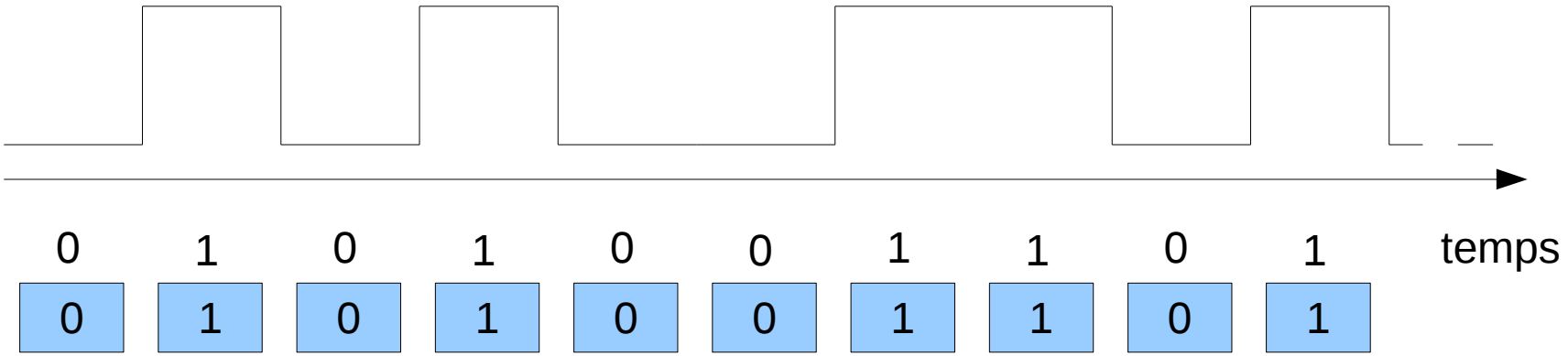
- Topologie point à point



- Canal
  - filaire : électrique, (optique)
  - sans fil : optique visible, optique infrarouge, (acoustique)
- Exemple
  - télévision et télécommande

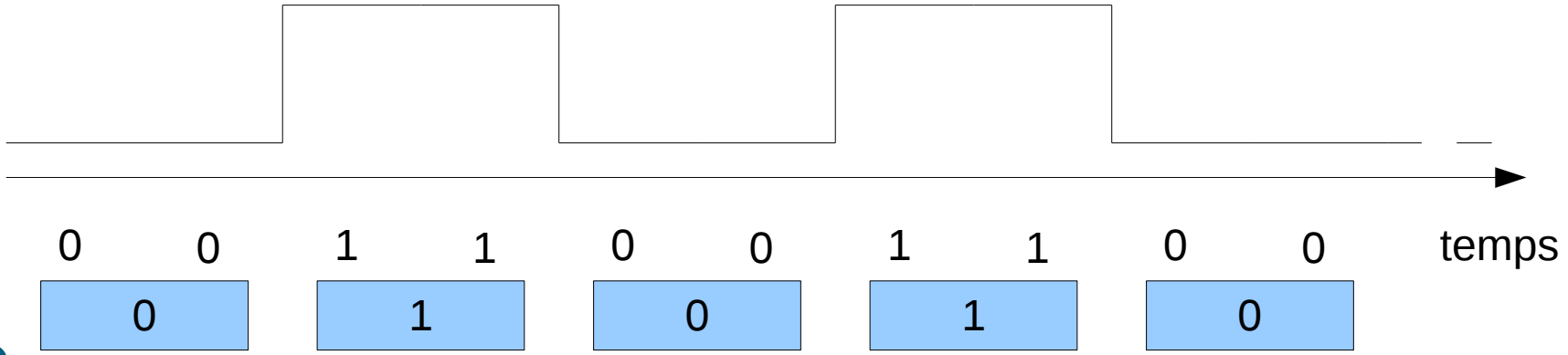
2.1 Topologie point à point

- Couche physique
  - codage des 0 et des 1 par modulation
  - modulation = variation d'intensité, de longueur d'onde, etc.
- Exemple



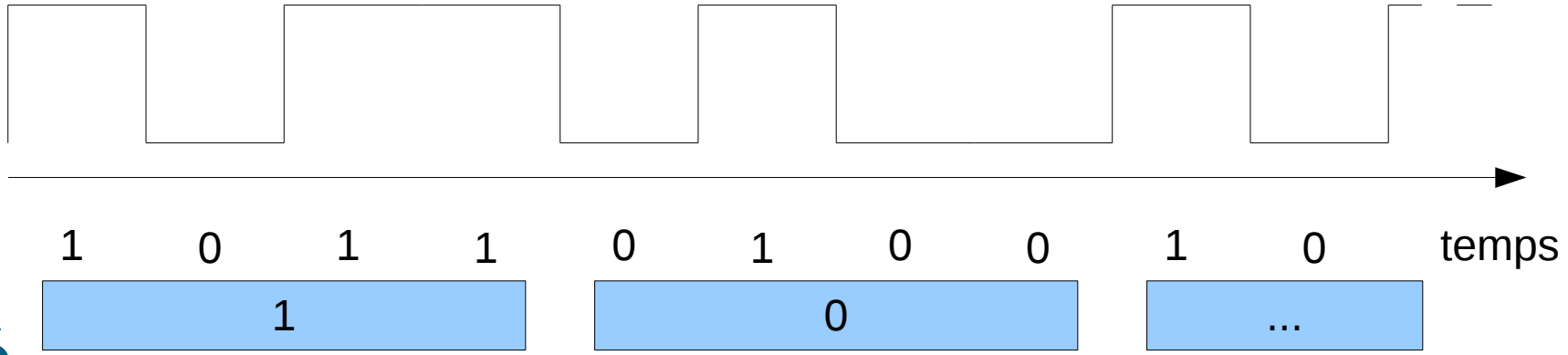
2.1 Topologie point à point

- Problème
  - atténuations sur le canal, interférences
- Solution
  - redondance, augmentation des seuils (durée, intensité)
- Exemple



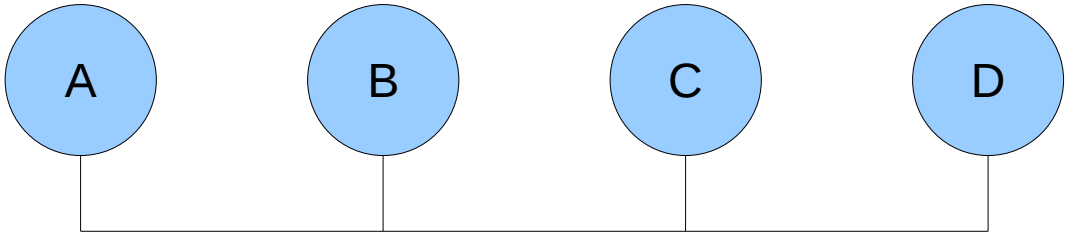
2.1 Topologie point à point

- Problème
  - atténuation dépend de l'intensité, changements d'intensité plus faciles à détecter
- Solution
  - modulations complexes avec correction d'erreur
- Exemple



2.2 Topologie en bus

- Topologie en bus



- Canal
  - filaire
- Exemple
  - BNC et câble coaxial, radio CB, (audioconférence)



2.2 Topologie en bus

- Problème
  - au cours du temps, plusieurs émetteurs parlent à plusieurs récepteurs
- Solution
  - la suite de 0 et de 1 (du niveau physique) est découpée en trames
  - les trames sont délimitées (équivalent de « Allô » et « Aurevoir » au téléphone)
  - chaque trame concerne une communication, mais une communication nécessite plusieurs trames

2.2 Topologie en bus

- Problème
  - identification = quand un émetteur parle, il doit identifier le(s) récepteur(s) voulu(s)
- Solution
  - adressage : chaque entité possède une adresse (dite physique ou MAC)
  - adressage unique mondial (48 bits) : numéro de constructeur, numéro de série, numéro de carte
  - remarques : adresse de diffusion, adressage non unique, trames contenant des adresses

2.2 Topologie en bus

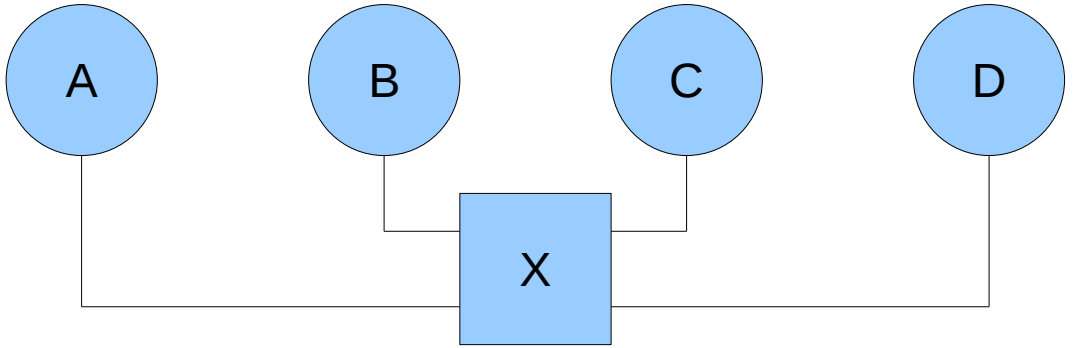
- Problème
  - collision = si deux émetteurs parlent en même temps, les communications sont brouillées
- Solution
  - besoin d'un protocole d'accès au canal (MAC = *medium access control*)
  - exemples : discussions en groupe

2.2 Topologie en bus

- Protocole Ethernet 
  - CSMA/CD = *Carrier Sense Multiple Access with Collision Detection*
  - avant de parler, un émetteur s'assure que personne ne parle
  - quand une collision est détectée, chaque émetteur retransmet après un temps aléatoire
  - nécessite un temps minimum de transmission pour détecter les collisions
  - la fenêtre de tirage des temps aléatoires augmente exponentiellement

2.3 Topologie en étoile

- Topologie en étoile (avec concentrateur)



- Canal
  - filaire, sans fil
- Exemple
  - RJ45 et paires torsadées, WiFi, discussions en classe (présence d'un maître)

2.3 Topologie en étoile

- Avantage de l'étoile sur le bus
  - facilité de mise à jour de la topologie (ajout ou suppression)
- Inconvénients de l'étoile sur le bus
  - nécessite un concentrateur
  - nombre de ports limités sur le concentrateur
- Protocole MAC : Ethernet

2.3 Topologie en étoile

- Concentrateur
  - ce qui est reçu sur un port quelconque est envoyé sur tous les autres ports
- Commutateur
  - ce qui est reçu sur un port est envoyé au port amenant à la destination (nécessite une écoute des communications)
  - fonctionne comme un concentrateur pour les adresses inconnues, oublie périodiquement les adresses
  - améliore les performances

2.4 Approfondissements

- Modulations physiques
- Allocation des adresses MAC
- Limitation de la longueur d'une topologie en bus ou en étoile
- Mécanisme CSMA/CA en WiFi (principe similaire à CSMA/CD)
- Topologie en anneau et protocole Token Ring (accès au canal par droit de parole)
- Mécanisme RTS/CTS pour réseaux sans fil

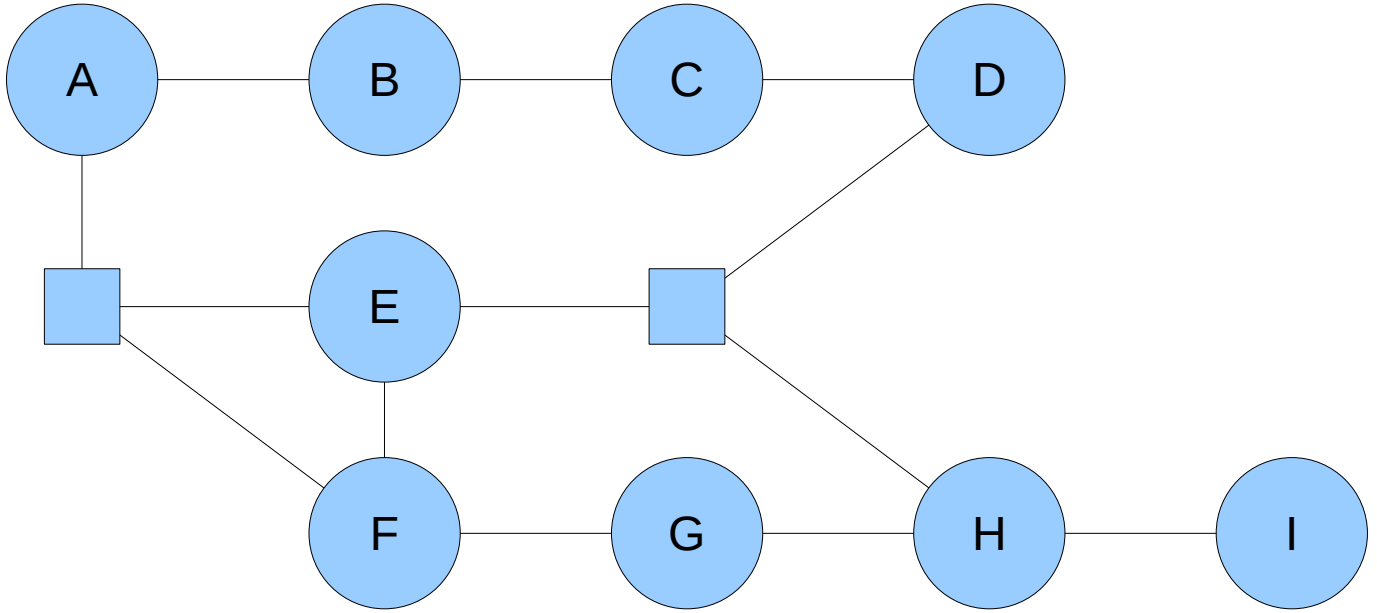
3. Réseaux locaux

3. Réseaux locaux

- IP
- Routage
- Approfondissements

3.1 IP

- IP = *Internet Protocol*
  - fonctionnalités : encapsulation, adressage, acheminement, fragmentation / réassemblage



3.1 IP

- Encapsulation
  - un paquet IP est encapsulé dans une trame
  - le paquet IP contient des informations de couche 3 (source initiale, destination finale), la trame contient des informations de couche 2 (noeud émetteur sur le lien, noeud récepteur sur le lien)

3.1 IP

- Adressage logique 
  - adresses à points
  - l'adresse logique dépend de la topologie
  - adressage en classes (A, B, C), masques de sous-réseaux
  - identifiant de classe + identifiant de réseau + identifiant de machine
  - exemple : numéros de téléphone, adresses postales
  - plus pratique que l'adressage physique
  - ipconfig (Windows), ifconfig (Linux)

3.1 IP

- Acheminement
  - utilise les tables de routage pour trouver la route à suivre par les paquets
  - algorithme : *longest common prefix match*
  - permet l'agrégation d'entrées dans les tables de routage
  - suivi des panneaux lors de la conduite
  - route -n (Linux), route print (Windows)

```
~/isn# route -n
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic   Metric  Iface
193.49.118.0    0.0.0.0         255.255.255.0   U       0       eth0
0.0.0.0         193.49.118.1    0.0.0.0         UG      0       eth0
```

3.1 IP

- Fragmentation / réassemblage
  - les paquets trop gros sont découpés en fragments autonomes, lorsque c'est nécessaire
  - les fragments sont réassemblés à la destination

3.2 Routage

- Routage = calcul du chemin à suivre
  - recherche de l'itinéraire sur une carte routière
  - calcul d'un plus court chemin de la source à la destination
  - construction de la table de routage
- Métriques
  - plus courte distance
  - plus petite durée
  - moins cher

3.2 Routage

- Algorithme de Dijkstra pour le calcul des plus courts chemins depuis une source 
  - tous les noeuds sont étiquettés avec $+\infty$, la source est étiquettée avec *0*
  - les noeuds sont considérés (une seule fois) à tour de rôle, par ordre croissant d'étiquette
  - quand un noeud *n* d'étiquette e_n est considéré, on met à jour l'étiquette e_v de chaque voisin *v* : si $e_v > e_n + d(n,v)$ alors $e_v = e_n + d(n,v)$

3.3 Approfondissements

- Protocole de routage OSPF (utilise Dijkstra)
- Protocole de routage RIP (n'utilise pas Dijkstra)
- Protocole de routage inter-domaines BGP
- Protocole ARP (traduction d'adresses)
- Traduction d'adresses et de ports (NAT, PAT)
- Protocole IPv6 (limites de IPv4, réseaux privés, nouveautés de IPv6)
- Allocation d'adresses locales avec DHCP

4. Réseaux mondiaux

4. Réseaux mondiaux

- UDP
- TCP
- Approfondissements

4.1 UDP

- UDP = *User Datagram Protocol*
  - ajoute à IP : multiplexage
  - protocole non fiable (dit « non connecté »), rapide
- Multiplexage
  - plusieurs applications différentes pour un même couple (source, destination)
  - ports
- Exemple d'utilisation
  - téléphonie, télévision, jeux temps-réel

4.2 TCP

- TCP = Transfer Control Protocol
  - ajoute à IP : multiplexage, fiabilité, communication bidirectionnelle, contrôle de flux, contrôle de congestion
  - protocole fiable (dit « connecté »), lent
- Fiabilité
  - cf problème des généraux
  - suppression des doublons (par numérotation), réordonnancement des paquets (par mise en attente et numérotation), garantie de livraison (par acquittements positifs et retransmissions)

4.2 TCP

- Communication bidirectionnelle
  - deux sens : A → B et B → A
  - A → B : données de A pour B + informations concernant B → A
  - serveur : entité attendant la connexion
  - client : entité demandant la connexion
  - client-serveur : type de communication *unicast*, entités fonctionnellement différentes, identiques au niveau du protocole (sauf à l'initialisation)
  - initialisation : poignée de main

4.2 TCP

- Contrôle de flux
  - un receveur lent peut ralentir l'émetteur
  - le receveur transmet la taille de sa mémoire de réception
- Contrôle de congestion
  - TCP estime la bande passante disponible sur le réseau, et adapte le débit au niveau de l'émetteur
  - deux modes : démarrage lent + évitement de congestion

4.2 TCP

- Contrôle de congestion (suite) 
  - CW = nombre maximum de paquets en transit (non acquittés)
  - démarrage lent : CW démarre à 1, CW augmente de 1 à chaque ACK reçu, sort du mode à la première perte
  - évitement de congestion : CW augmente de 1/CW à chaque ACK reçu, CW est divisé par 2 à chaque perte

4.2 TCP

- Retransmission rapide
  - retransmission après réception de trois ACK identiques
  - durée entre deux retransmissions augmente exponentiellement
- Acquittements retardés 
  - pas plus de 200 ms
  - sauf quand des données hors séquence sont reçues

4.2 TCP

- Exemple d'utilisation
  - pages web, transactions bancaires, jeux (hors temps-réel), streaming, etc.
- Internet vit grâce à TCP
  - contrôle de congestion
  - fiabilité

4.2 TCP

- Exemple : 
  - dans le terminal 1 (serveur) : nc -l -p 3000
  - dans le terminal 2 (client) : telnet localhost 3000
  - dans le terminal 2 (client) : bonjour ?
  - dans le terminal 1 (serveur) : adieu !
  - dans le terminal 2 (client) : Ctrl-], quit

4.2 TCP

- Capture du traffic : telnet localhost 3000

```
~/isn# tcpdump -i lo -s 100 -x
15:34:42.662582 IP localhost.54470 > localhost.3000:
        S 427552453:427552453(0) win 32792
        <mss 16396,sackOK,timestamp 3890034 0,nop,wscale 6>
        4510 003c b5b7 4000 4006 86f2 7f00 0001 7f00 0001 d4c6 0bb8 197b eec5 0000 0000
        a002 8018 4b21 0000 0204 400c 0402 080a 003b 5b72 0000 0000 0103 0306
15:34:42.663315 IP localhost.3000 > localhost.54470:
        S 207531451:207531451(0) ack 427552454 win 32768
        <mss 16396,sackOK,timestamp 3890034 3890034,nop,wscale 6>
        4500 003c 0000 4000 4006 3cba 7f00 0001 7f00 0001 0bb8 d4c6 0c5e adbb 197b eec6
        a012 8000 3561 0000 0204 400c 0402 080a 003b 5b72 003b 5b72 0103 0306
15:34:42.663439 IP localhost.54470 > localhost.3000:
        . ack 1 win 513 <nop,nop,timestamp 3890034 3890034>
        4510 0034 b5b8 4000 4006 86f9 7f00 0001 7f00 0001 d4c6 0bb8 197b eec6 0c5e adbc
        8010 0201 1c84 0000 0101 080a 003b 5b72 003b 5b72
```

4.2 TCP

- Capture du traffic : bonjour ?

```
15:34:51.100250 IP localhost.54470 > localhost.3000:
    P 1:12(11) ack 1 win 513 <nop,nop,timestamp 3892143 3890034>
    4510 003f b5b9 4000 4006 86ed 7f00 0001 7f00 0001 d4c6 0bb8 197b eec6 0c5e adbc
    8018 0201 fe33 0000 0101 080a 003b 63af 003b 5b72 626f 6e6a 6f75 7220 3f0d 0a
15:34:51.100320 IP localhost.3000 > localhost.54470:
    . ack 12 win 512 <nop,nop,timestamp 3892143 3892143>
    4500 0034 f6ce 4000 4006 45f3 7f00 0001 7f00 0001 0bb8 d4c6 0c5e adbc 197b eed1
    8010 0200 0c00 0000 0101 080a 003b 63af 003b 63af
```

  - ASCII : "bonjour ?\r\n" = 626F 6E6A 6F75 7220 3D0D 0A

4.2 TCP

- Capture du traffic : adieu !

```
15:34:53.196610 IP localhost.3000 > localhost.54470:
    P 1:9(8) ack 12 win 512 <nop,nop,timestamp 3892667 3892143>
    4500 003c f6cf 4000 4006 45ea 7f00 0001 7f00 0001 0bb8 d4c6 0c5e adbc 197b eed1
    8018 0200 fe30 0000 0101 080a 003b 65bb 003b 63af 6164 6965 7520 210a
15:34:53.196667 IP localhost.54470 > localhost.3000:
    . ack 9 win 513 <nop,nop,timestamp 3892667 3892667>
    4510 0034 b5ba 4000 4006 86f7 7f00 0001 7f00 0001 d4c6 0bb8 197b eed1 0c5e adc4
    8010 0201 07df 0000 0101 080a 003b 65bb 003b 65bb
```

  - ASCII : "adieu !\n" = 6164 6965 7520 210A

4.2 TCP

- Capture du traffic : Ctrl-], quit

```
15:34:56.176315 IP localhost.54470 > localhost.3000:
    F 12:12(0) ack 9 win 513 <nop,nop,timestamp 3893412 3892667>
    4510 0034 b5bb 4000 4006 86f6 7f00 0001 7f00 0001 d4c6 0bb8 197b eed1 0c5e adc4
    8011 0201 04f5 0000 0101 080a 003b 68a4 003b 65bb
15:34:56.176687 IP localhost.3000 > localhost.54470:
    F 9:9(0) ack 13 win 512 <nop,nop,timestamp 3893412 3893412>
    4500 0034 f6d0 4000 4006 45f1 7f00 0001 7f00 0001 0bb8 d4c6 0c5e adc4 197b eed2
    8011 0200 020c 0000 0101 080a 003b 68a4 003b 68a4
15:34:56.176750 IP localhost.54470 > localhost.3000:
    . ack 10 win 513 <nop,nop,timestamp 3893412 3893412>
    4510 0034 b5bc 4000 4006 86f5 7f00 0001 7f00 0001 d4c6 0bb8 197b eed2 0c5e adc5
    8010 0201 020b 0000 0101 080a 003b 68a4 003b 68a4
```

4.3 Approfondissements

- Drapeaux TCP (SYN, ACK, FIN, PSH, URG, RST)
- Algorithme de Nagle
- Accélérateurs de connexion

5. Web

5. Web

- HTTP
- FTP
- Approfondissements

5.1 HTTP

- Web = *World Wide Web* = toile mondiale
  - le Web est une (grande) partie d'Internet
  - partie navigable d'Internet *via* des liens hypertextes
- HTTP = *HyperText Transfer Protocol*
  - protocole de transfert de fichiers hypertextes (pages HTML, images)
  - protocole basé sur des requêtes-réponses

5.1 HTTP

- Définitions
  - URL = *Uniform Resource Location* = adresse HTTP
  - URI = *Uniform Resource Identifier* = adresse d'une ressource sur un serveur
  - *http://serveur/uri*
- Requêtes
  - GET URI HTTP/version
  - Serveur hôte, encodages acceptés, langues, etc.

5.1 HTTP

- Réponses
  - version HTTP et code de retour de la requête
  - informations sur le serveur, la ressource ou la réponse
  - éventuellement des *cookies*
  - le fichier demandé (généralement)

5.1 HTTP

```
~/isn# telnet www.google.fr 80
Trying 209.85.148.99...
Connected to www.l.google.com.
Escape character is '^]'.
GET / HTTP/1.0
Host: www.google.fr

HTTP/1.0 200 OK
Date: Sun, 25 Sep 2011 13:07:28 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
Set-Cookie: PREF=ID=3359...; expires=...; path=/; domain=.google.fr
Set-Cookie: NID=51...; expires=...; path=/; domain=.google.fr; HttpOnly
Server: gws

<html>(...)</html>
```

5.2 FTP

- FTP = *File Transfer Protocol*
  - transfert de fichiers
  - les navigateurs web récents sont aussi des clients FTP, et affichent les fichiers comme des liens
  - nécessite un serveur FTP

5.2 FTP

- Commandes FTP (du protocole)
  - USER (envoie le login), PASS (envoie le mot de passe), PWD (affiche le répertoire), CWD (change de répertoire), LIST (liste les fichiers), DELE (efface un fichier), RETR (récupère un fichier), PORT (prépare un transfert), QUIT (quitte), STOR (envoie un fichier)
- Commandes FTP (des clients)
  - CD, LS, GET, PUT, QUIT

5.2 FTP

- Réponses FTP (du protocole)
  - trois chiffres : *xyz*
  - *x* : 1 = réponse préliminaire positive, 2 = réponse terminale positive, 3 = réponse intermédiaire positive, 4 = réponse temporaire négative, 5 = réponse permanente négative
  - *y* : 0 = syntaxe, 1 = information, 2 = connexions, 3 = authentification, 4 = système de fichiers
  - exemples : 200 = commande ok, 230 = utilisateur connecté, 331 = login correct mais attend mot de passe, 500 = erreur de syntaxe

5.2 FTP

```
~/isn# ftp localhost
Connected to localhost.
220 ProFTPD 1.3.2c Server (ftp)
Name: anonymous
331 Anonymous login ok, send your complete email address as your password
Password: ********
230-Bienvenue sur le serveur ftp.
230 Anonymous access granted, restrictions apply
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
-rw-r--r--   1 ftp      ftp           521 Jan 1  2011 welcome.msg
226 Transfer complete
ftp> get welcome.msg
local: welcome.msg remote: welcome.msg
200 PORT command successful
150 Opening BINARY mode data connection for welcome.msg (521 bytes)
226 Transfer complete
521 bytes received in 0.00 secs (574.3 kB/s)
ftp> quit
221 Goodbye.
```

5.3 Approfondissements

- Intranet
- Extranet
- VPN (*Virtual Private Network)*

6. Internet

6. Internet

- DNS
- Emails
- Réseaux *overlays*
- Aspects sociétaux
- Approfondissements

6.1 DNS

- DNS = *Domain Name System*
  - traduit les noms de domaine en adresses IP
  - architecture hiérarchique 
- Exemple : *www.google.com.*
  - serveur racine => adresse de *com.*
  - serveur *.com.* => adresse de *google.com.*
  - serveur *.google.com.* => adresse de *www.google.com.*

6.1 DNS

- Autres utilisations
  - traductions d'adresses IP en noms de domaine
  - informations concernant un nom de domaine (redirections, clés publiques, serveur d'emails, coordonnées du responsable, etc.)

6.2 Emails

- Emails
  - protocole d'envoi : SMTP (*Simple Mail Transfer Protocol)*
  - protocole de transfert : SMTP
  - protocole de récupération : POP3 *(Post-Office Protocol v3)*

6.2 Emails

- Exemple
  - *alice@a.com* envoie un email à *bob@b.com*
  - Alice se connecte à son serveur SMTP (par exemple *smtp.a.com*) et crée son email
  - *smtp.a.com* cherche (via le DNS) le serveur d'emails associé à *b.com*, qui est *smtp.b.com*
  - *smtp.a.com* envoie l'email à *smtp.b.com*
  - *smtp.b.com* détecte que l'utilisateur est local, et transfère l'email dans la boîte de l'utilisateur
  - Bob se connecte en POP3 ou IMAP

6.3 Réseaux *overlays*

- Réseau *overlay* = réseau dans le réseau
  - réseaux sociaux (amis, professionnels)
  - réseaux pair à pair
- Réseau pair à pair
  - architecture décentralisée : pas de serveur central, tout les utilisateurs sont serveurs
  - topologie logique simple des adresses permettant le routage
  - hachage pour l'identification des fichiers
  - recherche de fichiers par inondation contrôlée

6.3 Réseaux *overlays*

- Problématique de l'identification des fichiers dans les réseaux pair à pair
  - mise à disposition d'un fichier *F* de 1 Go par l'utilisateur *A*
  - mise à disposition d'un fichier *G* de 1 Go par l'utilisateur *B*
  - est-ce que *F=G* ?

6.3 Réseaux *overlays*

- Solution
  - associer à chaque fichier un (petit) numéro *via* une fonction de hachage *H*
  - si *F* est proche de *G*, *H(F)* est très différent de *H(G)*
  - si *F* et *G* sont différents, il est peu probable que *H(F)=H(G)*
- En pratique
  - la recherche d'un fichier se fait sur son nom (et sa description), puis par hachage (pour trouver les fichiers identiques de noms différents)

6.3 Réseaux *overlays*

- Problématique de l'envoi des fichiers dans les réseaux pair à pair
  - comment envoyer un fichier de 1 Go alors qu'il n'y a pas de serveur permanent, et que les utilisateurs se connectent et se déconnectent arbitrairement ?
- Solution
  - découpage du fichier en blocs
  - envoi des blocs indépendemment
  - dès qu'un utilisateur a téléchargé un bloc, il peut le retransmettre à d'autres utilisateurs, même si le fichier est encore incomplet

6.4 Aspects sociétaux

- Internet est un support de communication ouvert (par principe initial)
  - liberté d'expression ?
  - anonymat et propos diffamatoires ?
  - véracité des informations sur Internet ?
- Piratage
  - impact économique ?
  - impact juridique ?
  - Hadopi en France ? À l'étranger ?

6.4 Aspects sociétaux

- Vie privée sur Internet
  - réseaux sociaux non anonymes
  - longue conservation des informations
- Guerre électronique ?
  - enjeux ?
  - impact ?
  - attaques possibles à l'échelle mondiale
  - unifier les justices locales
  - unifier les informations locales

6.5 Approfondissements

- Tables de hachage distribuées (routage dans les réseaux pair à pair)
- *Spiders* et *Crawlers* (wget)
- IMAP *(Internet Message Access Protocol)*