ANNÉE PASSERELLE EN INGÉNIERIE

Introduction à la Téléinformatique

Notions fondamentales, théorie, exemples et exercices

v0.10

HEIA-FR

Vincent Audergon

Année académique 2026-2027

ANNÉE PASSERELLE EN INGÉNIERIE

Introduction à la Téléinformatique

Notions fondamentales, théorie, exemples et exercices

Vincent Audergon

HEIA-FR

Année académique 2026-2027

Édition 0.10 .
Contact : vincent.audergon@hefr.ch

PRÉFACE

ATTENTION
Cette édition est publiée **progressivement** au fil du semestre : elle contient les chapitres des semaines 1 à **9** (voir le plan du cours ci-après). Les chapitres suivants seront ajoutés au fur et à mesure — consultez le site du cours pour la dernière version.

Ce livre accompagne le cours de base de téléinformatique. Il n'a pas vocation à remplacer les cours. Il rassemble, dans un même volume, la théorie essentielle, des exemples résolus et des exercices pour s'entraîner. La présence aux cours reste obligatoire.

L'ouvrage est pensé comme un **ouvrage de référence** pour le cours. Chaque chapitre s'ouvre sur ses objectifs et se lit de façon autonome : vous pouvez le parcourir dans l'ordre ou y revenir ponctuellement, lorsqu'une notion vous échappe ou qu'un exercice réclame un rappel. Les chapitres sont ordonnés dans le même ordre que les slides du cours.

Les concepts théoriques présentés dans cet ouvrage sont parfois simplifiés pour faciliter la compréhension. Des subtilités et des pans théoriques sont volontairement omis, car ils ne sont pas nécessaires pour atteindre les objectifs du cours. L'ouvrage est donc un **outil d'apprentissage**, et non un manuel de référence exhaustif pour une introduction à la téléinformatique. Il est important de compléter votre apprentissage avec les cours, les travaux pratiques et d'autres ressources pour obtenir une compréhension complète du sujet.

Les **définitions**, **exemples** et **exercices** sont numérotés par chapitre et signalés par des encadrés de couleur, afin de repérer d'un coup d'œil ce que l'on cherche. Les solutions figurent à la fin du livre : essayez d'abord, consultez ensuite.

Toute suggestion d'amélioration ou de correction est la bienvenue. N'hésitez pas à me contacter par mail à l'adresse vincent.audergon@hefr.ch ou à passer dans mon bureau pour en discuter (C10.08).

Transparence sur l'usage de l'IA: cet ouvrage est classé **DALIA D2**[1] (Augmenté par IA) : une intelligence artificielle est intervenue dans la création de petits éléments spécifiques, sans changer le sens de l'oeuvre. L'idéation, la rédaction et la validation finale du contenu restent humaines.

A travers le livre, divers blocs colorés sont utilisés pour signaler des éléments particuliers. Voici leur signification :

REMARQUE
Un point particulier, important ou délicat.

ATTENTION
Un point auquel il faut faire particulièrement attention.

ASTUCE
Une astuce qui permet de mieux assimiler une notion

DÉFINITION 0.1 · EXEMPLE

Une définition est un encadré de couleur, destiné à signaler une notion ou un concept à connaître.

EXEMPLE 0.1

Un exemple est un encadré de couleur, destiné à illustrer une notion ou un concept de manière concrète.

EXERCICE 0.1

Un exercice à résoudre. Les solutions sont disponibles à la fin du livre.

APPROFONDISSEMENT

[1] https://daliascale.org/fr/

Informations complémentaires, mais en dehors des critères d'évaluation du cours, c'est pour votre propre intérêt.

ANECDOTE

Complément d'information sous forme d'anecdote, également en dehors des critères d'évaluation du cours.

PLAN DU COURS

SEM.	CHAPITRE
1	Introduction à la téléinformatique
1	Signaux, topologies et classifications des réseaux
1	Binaire, hexadécimal et opérations logiques
2	Protocoles et modèles en couches
2	Matériel
2	Accès au médium
2	Ethernet et IP
2	ARP et ICMP
3	Routage
4	TCP et UDP
4	Linux
5	*Révisions TE1*
6	*Semaine de relâche*
7	DHCP et DNS
8	HTTP
8	Cryptographie, Telnet et SSH
8	Mail et FTP
9	NAT
10	CISCO IOS *(à venir)*
10	RIP et OSPF *(à venir)*
11	Sécurité *(à venir)*

SEM.	CHAPITRE
11	Wireless *(à venir)*
11	Automatisation *(à venir)*
12	*TE2 (à venir)*
13	*Révisions TE2 (à venir)*

Table des matières

CHAPITRE 1

Introduction à la téléinformatique

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir la téléinformatique
- situer l'état de l'art et les enjeux de la téléinformatique
- décrire les éléments d'une chaîne de transmission
- citer des exemples d'applications et de services de téléinformatique

1.1 Qu'est-ce que la téléinformatique ?

Avant de pouvoir définir ce qu'est la téléinformatique, il est nécessaire de rappeler ce que sont l'informatique et les télécommunications.

DÉFINITION 1.1 · INFORMATIQUE

Science du traitement automatique et rationnel de l'information considérée comme le support des connaissances et des communications. (larousse.fr)

DÉFINITION 1.2 · TÉLÉCOMMUNICATIONS

Transmission, émission ou réception d'informations par fil, radioélectricité, optique, ou d'autres systèmes électromagnétiques. (larousse.fr)

DÉFINITION 1.3 · TÉLÉINFORMATIQUE

Association de techniques des télécommunications et de l'informatique en vue de réaliser, à distance, l'échange de données et la commande des traitements automatiques. (larousse.fr)

1.2 Historique

Il y a fort longtemps

Transmission de signaux

Les Hommes communiquaient à distance par des signaux visuels (fumée, feu, drapeaux, etc.) ou sonores (trompes, tambours, etc.).

1837

Télégraphe électrique

Samuel Morse et Alfred Vail mettent au point un code de points et de traits : la première transmission numérique de masse.

1876

Téléphone

Graham Bell dépose le brevet du téléphone : la voix devient un signal électrique analogique transporté sur fil.

1969

ARPANET

Premier réseau à commutation de paquets reliant quatre universités américaines : l'ancêtre d'Internet.

1983

TCP/IP

ARPANET adopte la pile TCP/IP comme protocole unique : naissance de l'Internet moderne.

1991

Le Web

Tim Berners-Lee publie le premier site web au CERN et rend le protocole HTTP public.

2000

Internet devient grand public

L'Internet devient accessible au grand public. On dénombre environ 361 millions d'utilisateurs en 2000.

2007

iPhone

Apple lance l'iPhone, un smartphone qui révolutionne la téléinformatique en combinant téléphone, internet et applications mobiles. D'autres smartphones suivent rapidement, rendant l'accès à l'information et aux services en ligne omniprésent.

2010

2 milliards d'utilisateurs

Le nombre d'utilisateurs d'Internet atteint 2 milliards, marquant une adoption mondiale massive de la téléinformatique.

2012

Généralisation de la 4G

Le déploiement de la technologie 4G LTE permet des vitesses de transmission de données plus rapides et une meilleure connectivité mobile, facilitant l'utilisation d'applications en temps réel et le streaming vidéo sur les smartphones.

2013

Les smartphones en Suisse

En Suisse, le nombre de smartphones dépasse celui des téléphones mobiles traditionnels, avec environ 50 à 60% de la population équipée.

2019

5G

Le déploiement de la 5G commence, offrant des vitesses de transmission encore plus rapides, une latence réduite et la possibilité de connecter un grand nombre d'appareils simultanément, ouvrant la voie à de nouvelles applications de téléinformatique.

2020

Télétravail et pandémie

La pandémie de COVID-19 entraîne une adoption massive du télétravail et des outils de communication en ligne, mettant en évidence l'importance de la téléinformatique dans la vie professionnelle et personnelle.

2022

ChatGPT et IA générative

L'essor de l'intelligence artificielle générative, avec des modèles comme ChatGPT, transforme la manière dont les utilisateurs interagissent avec les systèmes informatiques, facilitant l'accès à l'information et la création de contenu.

2025

IA Omniprésente

L'intelligence artificielle devient omniprésente dans les applications de téléinformatique, offrant des assistants

virtuels, des recommandations personnalisées et des capacités d'automatisation avancées.

EXERCICE 1.1 · CHRONOLOGIE DE LA TÉLÉINFORMATIQUE

Sans regarder la frise ci-dessus, classez ces événements du plus ancien au plus récent : le Web, ARPANET, l'iPhone, le télégraphe électrique, l'adoption de TCP/IP.

1.3 Enjeux sociétaux, éthiques et environnementaux

Aujourd'hui, l'ITU (International Telecommunication Union) dénombre environ 6 milliards d'utilisateurs d'internet, soit près de 75% de la population mondiale [1]. La téléinformatique est devenue un élément central de la vie quotidienne, influençant la communication, l'éducation, le commerce, le divertissement et bien d'autres aspects de la société. Elle a transformé la manière dont les individus interagissent, accèdent à l'information et participent à l'économie mondiale. Selon l'étude JAMES 2024 [2], les jeunes Suisses passent en moyenne 3 à 4 heures par jour sur leur smartphone et 34% utilisent des outils à base d'IA générative au moins une fois par semaine.

Les retombées positives de cette révolution sont concrètes et mesurables. L'éducation s'est ouverte : le nombre d'apprenants inscrits à des cours en ligne ouverts à tous (MOOC) est passé de zéro en 2012 à environ 220 millions en 2021 [3]. En médecine, selon Flodgren et al. [4], la télémédecine améliore le contrôle de la glycémie chez les patients diabétiques, avec des résultats par ailleurs équivalents à ceux d'une prise en charge en face à face : un accès aux soins facilité, sans perte de qualité. À cela s'ajoutent des bénéfices quotidiens plus diffus : rester en contact avec ses proches, accéder à l'information en quelques secondes ou encore utiliser des robots pour effectuer des tâches dangereuses pour un humain à distance.

Les revers de la médaille sont également nombreux : cybercriminalité, désinformation, dépendance aux écrans, atteintes à la vie privée et à la sécurité des données, impact environnemental des data centers et de la blockchain, etc. La téléinformatique soulève donc des questions éthiques, sociales et environnementales importantes, nécessitant une réflexion sur son utilisation responsable et durable.

D'après l'étude JAMES 2024 [2], plus d'un quart des jeunes suisses déclarent avoir déjà reçu plusieurs fois des insultes en ligne, 20% ont été

victimes de diffusions de photos ou vidéos embarrassantes à leur sujet, et 7% ont déjà été victimes de menaces ou de chantage en ligne.

Du côté de la désinformation, l'OFS (Office fédéral de la statistique) indique qu'en 2025, 58% de la population suisse déclare avoir déjà rencontré du contenu faux ou douteux sur des sites ou des réseaux sociaux [5].

La consommation d'énergie des data centers est également un enjeu majeur. Selon l'Agence internationale de l'énergie (IEA), les data centers à travers le monde consomment environ 1.5% de l'électricité mondiale en 2024 pour un total de 415 TWh. Cette consommation devrait plus que doubler, pour atteindre 945 TWh d'ici 2030, soit la consommation électrique annuelle du Japon en 2025 [6]. Cette augmentation est due à la croissance exponentielle des données générées par les utilisateurs, l'essor de l'intelligence artificielle et le développement de services en ligne. Selon Freitag et al. [7], l'empreinte carbone du numérique en 2020 représente 2.1% à 3.9% des émissions mondiales de gaz à effet de serre, soit davantage que l'aviation civile.

Les datacenters ne sont pas les seuls responsables de l'empreinte carbone de la téléinformatique. Les blockchains et par extension les cryptomonnaies représentent une part significative de la consommation électrique mondiale. Selon le Cambridge Bitcoin Electricity Consumption Index (CBECI), le réseau du Bitcoin à lui seul consomme environ 201 TWh par an (état mi-juillet 2026), soit environ 0.7% de la consommation électrique mondiale [8].

En résumé, la révolution de la téléinformatique a apporté des avantages considérables à la société, mais elle pose également des défis importants en matière d'éthique, de sécurité et d'environnement. Il est crucial de continuer à développer des technologies et des pratiques responsables et durables pour maximiser les bénéfices tout en minimisant les impacts négatifs.

1.4 Qu'est-ce qu'un réseau ?

Au sens large, un réseau désigne un ensemble d'éléments (noeuds) interconnectés par des liens (arêtes), permettant la circulation d'une ressource ou d'une information.

L'un des réseaux les plus connus est probablement le réseau routier, qui relie des villes et des villages par des routes et autoroutes, permettant la circulation de véhicules et de personnes. Les autoroutes sont des liens à

grande capacité, tandis que les routes secondaires sont des liens à plus faible capacité. Les villes et villages sont les noeuds du réseau.

En téléinformatique, l'analogie avec le réseau routier est pertinente : les ordinateurs et autres équipements informatiques sont les noeuds, tandis que les câbles, fibres optiques et ondes radio sont les liens qui permettent la circulation des données. Certains liens sont à grande capacité (fibre optique, 5G), tandis que d'autres sont à plus faible capacité (ADSL, 3G). Les protocoles de communication jouent un rôle similaire aux règles de circulation, régissant la manière dont les données circulent dans le réseau.

DÉFINITION 1.4 · RÉSEAU INFORMATIQUE

Mise en relation d'au moins deux systèmes informatiques (ordinateurs, serveurs, périphériques) au moyen d'un médium de communication.

Un réseau informatique peut être de différentes tailles, formes et complexités, allant d'un simple réseau local dans une maison ou un bureau, à un vaste réseau mondial comme Internet.

EXEMPLE 1.1 · EXEMPLES DE RÉSEAUX INFORMATIQUES

- Internet
- Réseau d'entreprise
- Réseau de l'HEIA-FR
- Des écouteurs Bluetooth connectés à un smartphone
- Deux ordinateurs reliés par un câble Ethernet
- etc.

1.5 Éléments d'un réseau informatique

Un réseau informatique est composé de différents éléments, chacun ayant un rôle spécifique dans le système global. Dans le cadre du cours de téléinformatique, la nomenclature suivante est utilisée :

- **Périphérique / Hosts** : Un noeud du réseau, que ce soit un noeud dit « terminal » (ordinateur, smartphone, tablette, etc.) ou un noeud dit « intermédiaire » (routeur, switch, etc.).
- **Medium / Support de transmission** : Le lien physique ou logique qui permet la transmission des données entre les périphériques. Il peut s'agir de câbles (cuivre, fibre optique), d'ondes radio (Wi-Fi, 4G/5G) ou d'autres technologies de communication.
- **Messages / Données** : L'information qui transite à travers le réseau.
- **Protocoles / Règles de communication** : Les règles et conventions qui régissent la manière dont les périphériques communiquent entre eux, assurant que les messages sont correctement transmis, reçus et interprétés.

La Fig. 1 montre un exemple de réseau informatique utilisé pour envoyer un message d'un périphérique à un autre.

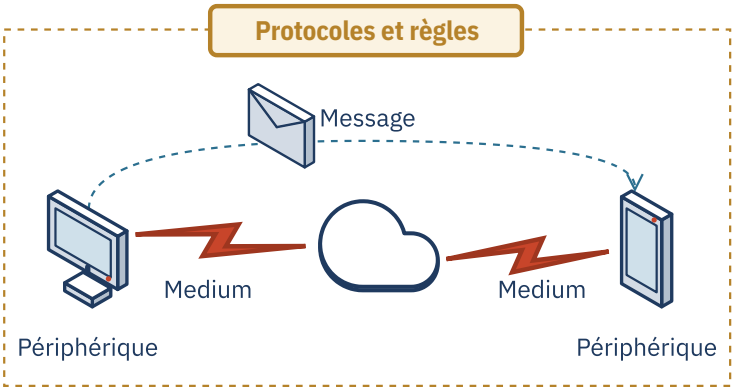


Fig. 1. – Illustration des différents éléments d'un réseau informatique. Dans cet exemple, l'ordinateur envoie un message au smartphone. La communication est régie par des règles.

EXERCICE 1.2 · NOMMER DES PÉRIPHÉRIQUES

Nommer des périphériques (terminaux et intermédiaires).

1.6 Médium de transmission

Les médiums (ou supports) de transmission sont les canaux par lesquels les données circulent dans un réseau informatique. Ils peuvent être classés en deux grandes catégories : les médiums filaires et les médiums sans fil. Pour transmettre une information d'un point A à un point B, des médiums de différents types peuvent être utilisés, chacun ayant ses propres caractéristiques, avantages et inconvénients.

EXEMPLE 1.2 · TRANSMISSION DE DONNÉES SUR DIFFÉRENTS MÉDIUMS

Alice envoie un message à Bob depuis son smartphone. Tout d'abord, les données sont transmises par les ondes radio entre le smartphone d'Alice et l'antenne relais la plus proche. Ensuite, les données voyagent à travers le réseau de l'opérateur mobile, composé de fibre optique. Enfin, elles atteignent l'ordinateur de Bob par le câble cuivre du réseau local.

EXERCICE 1.3 · IDENTIFIER LES ÉLÉMENTS D'UN RÉSEAU

Alice regarde une vidéo en streaming sur sa tablette, connectée au Wi-Fi de la maison. Le routeur de la maison est relié à Internet par une fibre optique. Identifiez dans ce scénario : les périphériques terminaux, les périphériques intermédiaires, les médiums de transmission et le message.

1.7 Applications et services

La téléinformatique permet la mise en place de nombreux services et applications qui facilitent la communication, l'accès à l'information et la collaboration à distance. Parmi les applications les plus courantes, on peut citer :

- La communication par messagerie
  - Email
  - Messagerie instantanée (WhatsApp, Telegram, Signal)
  - Réseaux sociaux (Facebook, Twitter, Instagram)
- Applications en temps réel
  - Jeux en ligne
  - Streaming vidéo (Netflix, YouTube, Twitch)

  - Streaming audio (Spotify, Apple Music)
  - Communication en temps réel (Discord, Slack, MS Teams)
- Applications partagées
  - Stockage en ligne (Google Drive, Dropbox, OneDrive)
  - Outils de collaboration (Google Docs, Microsoft Office 365)
  - Plateformes de gestion de projets (Trello, Asana, Jira)
- Partage de ressources
  - Impression à distance
  - Partage de fichiers
  - Accès à des bases de données distantes
- Gestion centralisée
  - Administration de systèmes et réseaux
  - Supervision et monitoring d'infrastructures
  - Gestion de serveurs et services cloud
- Cloud
  - Infrastructure as a Service (IaaS)
  - Platform as a Service (PaaS)
  - Software as a Service (SaaS)
- Etc.

Chaque application ou service possède des besoins spécifiques en termes de débit, latence et fiabilité. Le débit est la quantité de données pouvant transiter par seconde (ou autre unité de temps), la latence est le délai entre l'envoi et la réception d'un message, et la fiabilité est la capacité à transmettre les données sans perte ni corruption (le message arrive intact et complet).

EXERCICE 1.4 · FIABILITÉ, DÉBIT ET LATENCE SELON L'APPLICATION

Parmi les points suivants: débit, latence et fiabilité, quel est l'aspect le plus important pour l'envoi d'un email ? Pour le streaming vidéo (Netflix) ? Et pour les jeux en ligne ?

CHAPITRE 2

Signaux, topologies et classifications des réseaux

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir les notions de signal, de bruit et de canal
- distinguer un signal analogique d'un signal numérique
- citer les modes de transmission unicast, multicast et broadcast
- citer les principales topologies de réseau et leurs caractéristiques
- citer les classifications des réseaux de téléinformatique

2.1 Chaîne de transmission

Une chaîne de transmission est un ensemble d'éléments qui permettent de transmettre des données d'un émetteur à un récepteur. Elle se compose de trois éléments principaux : l'émetteur, le canal et le récepteur. Le signal est émis par l'émetteur, transmis par le canal et reçu par le récepteur. Le bruit peut altérer le signal lors de sa transmission, ce qui peut entraîner des erreurs de communication. Ce modèle fonctionne autant pour la téléinformatique que pour tout autre type de communication (orale, écrite, radio, etc.). Par exemple, dans le cadre d'une communication orale, l'émetteur est la personne qui parle, le canal est l'air et le récepteur est la personne qui écoute. Le bruit peut être un bruit de fond ou une mauvaise prononciation qui altère le message.

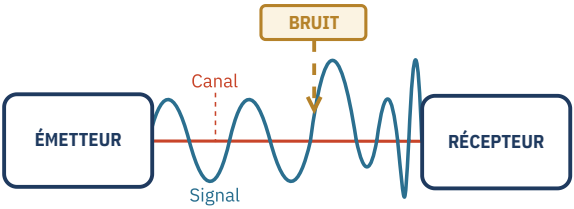


Fig. 2. – Illustration d'une chaîne de transmission. Le signal est émis par un émetteur, transmis par un canal et reçu par un récepteur. Le bruit peut altérer le signal lors de sa transmission, ce qui peut entraîner des erreurs de communication.

Dans le cadre de la téléinformatique, des mécanismes de correction d'erreurs sont mis en place pour détecter et corriger les erreurs de transmission. Ces mécanismes permettent d'assurer la fiabilité des communications, même en présence de bruit. Les différents types de médiums de transmission (câbles, fibres optiques, ondes radio, etc.) ont une sensibilité différente au bruit et aux interférences, ce qui peut influencer la qualité de la transmission.

EXERCICE 2.1 · IDENTIFIER LES ÉLÉMENTS D'UNE CHAÎNE DE TRANSMISSION

Pour le contexte ci-dessous, identifiez l'émetteur, le canal, le récepteur et les sources potentielles de bruit qui pourraient altérer le signal.

Alice envoie une lettre manuscrite à Bob. Le facteur transporte la lettre de la maison d'Alice à la maison de Bob. Pendant le transport, la lettre est exposée à des conditions météorologiques défavorables, comme la pluie et le vent, qui peuvent endommager le papier et rendre le texte illisible. De plus, il est possible que la lettre soit perdue ou volée pendant le transport.

2.2 Signal numérique et signal analogique

Il existe deux types de signaux : les signaux analogiques et les signaux numériques. Un signal analogique est un signal continu qui peut prendre une infinité de valeurs (Fig. 3), tandis qu'un signal numérique est un signal discret qui ne peut prendre qu'un nombre limité de valeurs (Fig. 4).

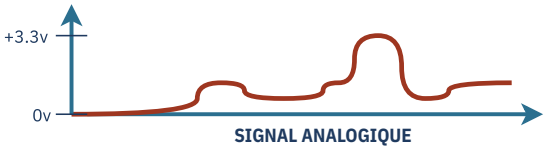


Fig. 3. – Illustration d'un signal analogique. Le signal analogique est continu et peut prendre une infinité de valeurs.

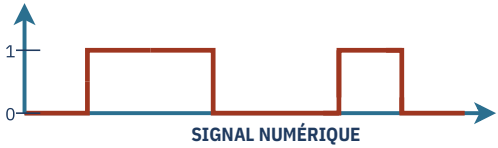


Fig. 4. – Illustration d'un signal numérique. Le signal numérique est discret et ne peut prendre qu'un nombre limité de valeurs.

Historiquement, les signaux analogiques étaient utilisés pour transmettre des informations, mais avec l'avènement de l'informatique et des technologies numériques, les signaux numériques sont devenus prédominants. Les signaux numériques présentent plusieurs avantages par rapport aux signaux analogiques, notamment une meilleure résistance au bruit et aux interférences.

ANECDOTE

Par abus de langage, le terme « numérique » est souvent remplacé par le terme « digital » en français. Cependant, ce terme est erroné car il signifie « relatif aux doigts » et n'a pas de rapport avec le traitement de l'information. Cette confusion est due à l'anglais, où le terme digital signifie « relatif aux chiffres » et est utilisé pour désigner les systèmes numériques.

EXERCICE 2.2 · ANALOGIQUE OU NUMÉRIQUE ?

Pour chaque élément ci-dessous, indiquez s'il s'agit d'un signal (ou support) analogique ou numérique.

A. La voix humaine se propageant dans l'air

B. Un fichier MP3

C. Un disque vinyle

D. Un SMS

E. La température affichée par un thermomètre à mercure

2.3 Modes de transmission

En télécommunication, la transmission peut se faire entre un émetteur et un ou plusieurs récepteurs. On distingue trois modes de transmission : le unicast, le multicast et le broadcast.

Le mode de transmission unicast (illustré par la Fig. 5) consiste à envoyer un signal d'un émetteur vers un récepteur unique. Ce mode est utilisé pour les communications directes entre deux appareils, comme dans le cas d'une conversation téléphonique ou d'une connexion Internet entre un ordinateur et un serveur.

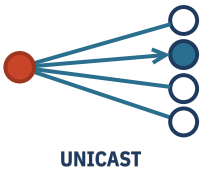


Fig. 5. – Illustration d'une transmission unicast. Le signal est envoyé d'un émetteur vers un récepteur unique.

Le mode de transmission multicast (illustré par la Fig. 6) consiste à envoyer un signal d'un émetteur vers un groupe de récepteurs. Ce mode est utilisé pour les communications de groupe, comme dans le cas d'une diffusion de vidéo en direct ou d'une conférence en ligne.

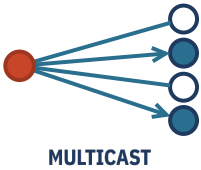


Fig. 6. – Illustration d'une transmission multicast. Le signal est envoyé d'un émetteur vers un groupe de récepteurs.

Le mode de transmission broadcast (illustré par la Fig. 7) consiste à envoyer un signal d'un émetteur vers tous les hôtes du domaine de diffusion (voir le chapitre *Accès au médium*). Il est utilisé pour transmettre des informations à tous les appareils d'un réseau, comme dans le cas d'une diffusion de messages ou d'annonces.

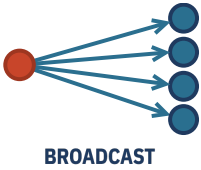


Fig. 7. – Illustration d'une transmission broadcast. Le signal est envoyé d'un émetteur vers tous les récepteurs du réseau.

EXERCICE 2.3 · IDENTIFIER LE MODE DE TRANSMISSION

Pour chaque situation, indiquez le mode de transmission : unicast, multicast ou broadcast.

A. Un appel vidéo entre deux personnes.

B. La transmission d'un flux vidéo en direct à un groupe d'abonnés.

C. Une annonce envoyée à tous les hôtes d'un réseau local.

D. Le téléchargement d'un fichier depuis un serveur.

2.4 Topologies de réseau

Un réseau peut prendre de nombreuses formes différentes, appelées topologies. Un certain nombre de topologies sont couramment utilisées dans les réseaux de téléinformatique, chacune ayant ses propres caractéristiques et avantages. Les principales topologies sont la topologie en bus, la topologie en anneau, la topologie en étoile, la topologie maillée et la topologie en arbre (ou hiérarchique). Chacune de ces topologies est décrite ci-dessous.

2.4.1 Topologie en bus

La topologie en bus est une topologie dans laquelle tous les périphériques du réseau sont connectés à un seul câble de transmission, comme illustré par la Fig. 8. Les données sont transmises le long du bus et chaque périphérique peut écouter les données qui passent. Cette topologie est simple à mettre en place et peu coûteuse, mais elle présente des limitations en termes de performance et de fiabilité. En effet, si le lien est coupé ou si le bus est endommagé, l'ensemble du réseau peut être affecté.

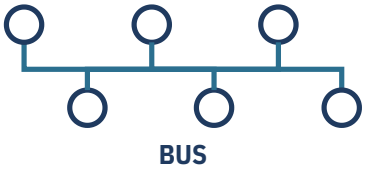


Fig. 8. – Illustration d'une topologie en bus. Chaque point représente un périphérique du réseau.

ANECDOTE

De nos jours, la topologie en bus n'est plus beaucoup utilisée à cause des nombreux défauts qu'elle présente. Dans les années 1980, une norme Ethernet appelée 10BASE5 avait pour principe d'utiliser une topologie en bus. Les périphériques se branchaient directement sur le câble du bus à l'aide de prises appelées « prises vampire » (Fig. 9). Ces prises étaient appelées ainsi car elles venaient s'accrocher au câble du bus comme un vampire s'accroche à sa victime.

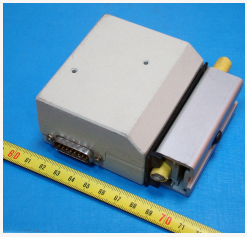


Fig. 9. – Illustration d'une prise vampire. Ces périphériques se branchaient directement sur le câble du bus.

2.4.2 Topologie en anneau

Dans la topologie en anneau, chaque périphérique est connecté à deux autres périphériques, formant ainsi un cercle. Les données circulent dans un seul sens autour de l'anneau, et chaque périphérique agit comme un répéteur, transmettant les données au périphérique suivant. Certaines implémentations avancées utilisent un double anneau avec circulation inverse pour assurer une redondance, mais cela augmente significativement la complexité et le coût. Dans un tel cas, la topologie supporte jusqu'à un lien coupé, mais pas deux.

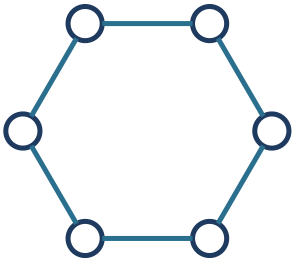


Fig. 10. – Illustration d'une topologie en anneau. Chaque point représente un périphérique du réseau.

La topologie en anneau fonctionne souvent avec un protocole de jeton, qui permet de réguler l'accès au réseau et d'éviter les collisions. Dans ce protocole, un jeton circule autour de l'anneau et seul le périphérique qui possède le jeton peut transmettre des données. Cela garantit que les données sont transmises de manière ordonnée et évite les conflits entre les périphériques.

2.4.3 Topologie en étoile

La topologie en étoile est la topologie la plus couramment utilisée dans les réseaux modernes. Dans cette topologie, tous les périphériques sont connectés à un périphérique d'interconnexion central. Si un lien tombe en panne, seul le périphérique concerné est affecté, et non l'ensemble du réseau. En revanche, le périphérique central représente un point de défaillance unique (« SPOF » pour « Single Point of Failure » en anglais). Si ce périphérique tombe en panne, l'ensemble du réseau est affecté. Cette topologie est facile à mettre en place et à maintenir, et elle offre de bonnes performances. Elle est parfois doublée (ou redondée) pour éviter le problème du SPOF, ce qui donne une étoile double (voir Fig. 11).

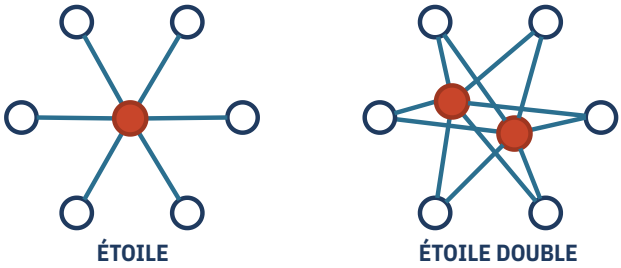


Fig. 11. – Illustration d'une topologie en étoile et d'une topologie en étoile double. Chaque point représente un périphérique du réseau.

2.4.4 Topologie maillée

La topologie maillée est la plus robuste de toutes les topologies, mais également la plus coûteuse. Dans cette topologie, chaque périphérique est connecté à tous les autres périphériques du réseau. Cela permet une redondance maximale, car si un périphérique tombe en panne, les données peuvent toujours circuler par d'autres chemins. Elle est généralement utilisée dans les réseaux de grande taille ou critiques, où la fiabilité est primordiale. Elle peut être complète (chaque périphérique est connecté à tous les autres) ou partielle (certains périphériques sont connectés à plusieurs autres, mais pas à tous), comme illustré par la Fig. 12. Internet est un exemple de réseau maillé partiel.

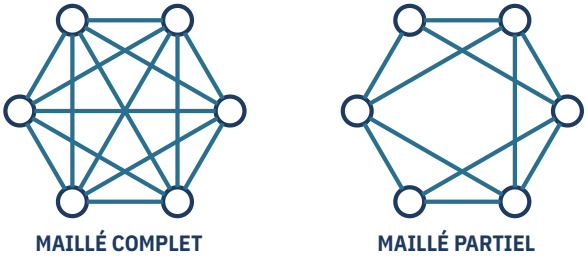


Fig. 12. – Illustration d'une topologie maillée. Chaque point représente un périphérique du réseau.

2.4.5 Topologie en arbre

La topologie en arbre (ou hiérarchique) est essentiellement un ensemble de topologies en étoile connectées entre elles, sous la forme d'une structure arborescente. Elle présente l'avantage de permettre une hiérarchisation des périphériques, ce qui facilite la gestion du réseau. Cependant, elle reste vulnérable aux pannes, car si un périphérique central tombe en panne, tous

les périphériques connectés à lui sont affectés. Cette topologie est souvent utilisée dans les réseaux d'entreprise ou les réseaux de campus.

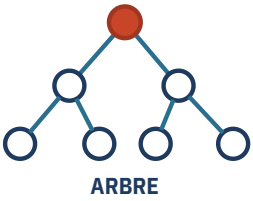


Fig. 13. – Illustration d'une topologie en arbre. Chaque point représente un périphérique du réseau.

2.4.6 Comparaison des topologies

Toutes les topologies ont leurs avantages et leurs inconvénients. Le Tableau 1 résume les principales caractéristiques de chaque topologie, en termes de coût, de fiabilité, de scalabilité (à quel point ce réseau peut être étendu facilement) et de difficulté de gestion.

Topo.	Avantages	Inconvénients	Exemples d'utilisation
Bus	Simple à mettre en place, peu coûteuse	Peu fiable, difficile à étendre	Anciens réseaux Ethernet (années 80-90)
Anneau	Accès déterministe, pas de collision (jeton)	Vulnérable aux pannes, complexe	Anciens réseaux Token Ring
Étoile	Facile à mettre en place et à maintenir, bonne performance	SPOF, nécessite un périphérique central	Réseaux locaux modernes
Maillée	Très robuste, redondance maximale	Coûteuse, complexe à mettre en place	Internet, réseaux critiques
Arbre	Hiérarchisation, facilité de gestion	La panne d'un nœud isole toute sa branche	Réseaux d'entreprise, réseaux de campus

Tableau 1. – Comparaison rapide des topologies principales

EXERCICE 2.4 · NOMBRE DE LIENS D'UN MAILLAGE COMPLET

A. Dans une topologie maillée complète de 6 périphériques, combien de liens sont nécessaires ?

B. Donnez la formule générale pour n périphériques.

C. Ce résultat explique l'un des inconvénients listés dans le Tableau 1. Lequel ?

2.5 Classification des réseaux

Un réseau est classé en fonction de sa taille. Les principales classes sont les réseaux locaux (LAN), les réseaux métropolitains (MAN) et les réseaux étendus (WAN). Le Tableau 2 présente les principales caractéristiques de chaque classe de réseau.

Classe	Nom complet	Portée (m)	Exemples d'utilisation
PAN	Personal Area Network	10^1	Bluetooth, casques audio, télécommandes
LAN	Local Area Network	10^3	Réseaux domestiques, réseaux d'entreprise
MAN	Metropolitan Area Network	10^4	Réseaux d'entreprise, réseaux de campus
WAN	Wide Area Network	10^6	Internet, réseaux interurbains
GAN	Global Area Network	10^7	Réseaux intercontinentaux, réseaux satellitaires

Tableau 2. – Classification des réseaux en fonction de leur taille

Un réseau local (LAN) est un réseau à l'échelle d'un bâtiment ou d'un site. Il couvre une zone géographique limitée et est généralement utilisé pour connecter des ordinateurs, des imprimantes et d'autres périphériques au sein d'une entreprise ou d'une maison. La gestion d'un LAN est souvent

privée et le débit de transmission est généralement élevé. Il existe une variante de LAN appelée WLAN (Wireless Local Area Network), qui utilise des technologies sans fil pour connecter les périphériques.

Un WAN est quant à lui à grande échelle, couvrant des zones géographiques étendues, comme des villes, des pays ou même des continents. Il est utilisé pour connecter plusieurs LAN entre eux et permet la communication entre des sites distants.

La Fig. 14 illustre les différences entre un réseau local (LAN) et un réseau étendu (WAN).

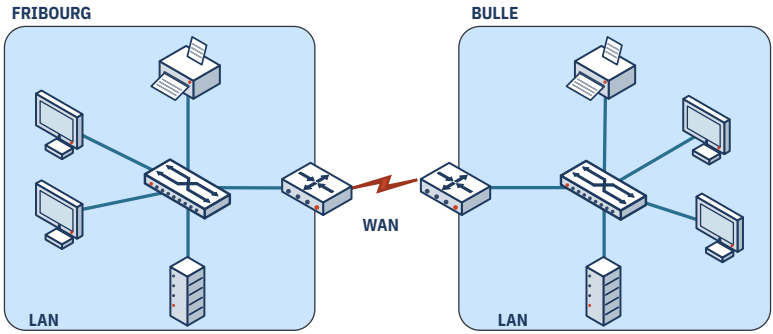


Fig. 14. – Illustration des différences entre un réseau local (LAN) et un réseau étendu (WAN). Dans cet exemple, une entreprise a un réseau local sur son site principal à Fribourg, et un autre réseau local sur son site secondaire à Bulle. Les deux sites sont connectés par un réseau étendu (WAN).

EXERCICE 2.5 · IDENTIFIER LA CLASSE DE RÉSEAU

Pour chaque exemple ci-dessous, identifiez la classe de réseau correspondante.

A. Un réseau domestique qui connecte plusieurs ordinateurs, smartphones et imprimantes dans une maison.

B. Un réseau d'entreprise qui connecte plusieurs bureaux dans une ville.

C. Une manette de jeu vidéo qui se connecte à la console de jeu via Bluetooth.

D. Le réseau de satellites de géolocalisation Galileo (« GPS » européen) qui permet la communication entre les satellites et les récepteurs sur Terre.

CHAPITRE 3

Binaire, hexadécimal et opérations logiques

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- Définir les unités de mesure de l'information (bit, octet, kilo-octet, méga-octet, etc.)
- définir les systèmes de numération binaire et hexadécimal
- effectuer des conversions entre ces systèmes
- appliquer les opérations logiques de base (AND, OR, NOT, XOR)

Ce chapitre introduit les concepts fondamentaux de la logique binaire et des systèmes de numération utilisés en informatique et en téléinformatique. Il couvre les unités de mesure de l'information, les bases de la numération binaire et hexadécimale, ainsi que les opérations logiques de base qui sont essentielles pour la manipulation des données et la conception des circuits numériques. Il est important d'aborder ces concepts avant de plonger dans les protocoles de communication et les architectures réseau, car ils constituent le fondement de la compréhension des systèmes informatiques et des réseaux modernes.

3.1 Unités de mesure de l'information

Ce segment présente les unités de mesure de l'information, qui sont essentielles pour quantifier la taille des données et pour représenter une taille de stockage ou la quantité d'information transmise sur un réseau. Ces unités sont basées sur le bit, l'octet et leurs multiples.

3.1.1 Bit et octet

En informatique, l'information est mesurée en **bit** (pour binary digit), qui est l'unité de base. Un bit peut prendre deux valeurs : 0 ou 1. Cela est

lié à la nature binaire des systèmes informatiques, qui utilisent des états électriques (Off/On) pour représenter l'information.

Un ensemble de 8 bits constitue un **octet** (aussi appelé **byte** en anglais), qui est l'unité de mesure la plus couramment utilisée pour quantifier la taille des données.

> **APPROFONDISSEMENT** · Pourquoi 8 bits ?
>
> Pourquoi 8 bits ? Pourquoi pas 10 ? ou 100 ? La taille de 8 bits a été choisie pour des raisons historiques et pratiques. Elle permet de représenter 256 valeurs différentes (de 0 à 255), ce qui permet de coder un large éventail de caractères et de symboles, notamment dans les systèmes de codage comme ASCII; tout en restant suffisamment compacte pour être efficace en termes de stockage et de traitement.

Formellement, un byte n'est pas forcément égal à un octet, mais dans la pratique, ces deux termes sont souvent utilisés de manière interchangeable.

3.1.2 Ordres de grandeur

Les bits et les octets sont souvent regroupés en multiples pour représenter de plus grandes quantités d'information. Les multiples les plus courants sont le kilo-octet (Ko), le méga-octet (Mo), le giga-octet (Go), et ainsi de suite. Il est important de noter que ces multiples peuvent être basés sur des puissances de 10 ou de 2, ce qui peut prêter à confusion.

En suivant le système international d'unités (SI), les préfixes sont basés sur des puissances de 10 :

- 1 kilo-octet (Ko) = 1 000 octets
- 1 méga-octet (Mo) = 1 000 Ko = 1 000 000 octets
- 1 giga-octet (Go) = 1 000 Mo = 1 000 000 000 octets
- etc.

En revanche, on retrouve aussi des préfixes binaires, basés sur des puissances de 2 :

- 1 kibi-octet (Kio) = 1 024 octets
- 1 mébi-octet (Mio) = 1 024 Kio
- 1 gibi-octet (Gio) = 1 024 Mio

etc.

Il n'est pas rare de trouver les notations utilisant le **B** de **byte** pour les octets, et le **b** pour les bits, afin de distinguer clairement les deux unités.

3.1.3 Notations

Le Tableau 3 ci-dessous résume les principales unités de mesure de l'information, leurs symboles et leurs ordres de grandeur. Une « unité » peut être remplacée par un bit (**b**), un octet (**o**), ou un byte (**B**).

Unité	Symbole	Description	Ordre de grandeur
bit	**b**	unité fondamentale de l'information (0 ou 1)	
byte	**B**	8 bits	
octet	**o**	8 bits	
kilo	**k**	1 000 unités	10^3
méga	**M**	1 000 kilo	10^6
giga	**G**	1 000 méga	10^9
tera	**T**	1 000 giga	10^{12}
kibi	**Ki**	1 024 unités	2^{10}
mébi	**Mi**	1 024 kibi	2^{20}
gibi	**Gi**	1 024 mébi	2^{30}
tébi	**Ti**	1 024 gibi	2^{40}

Tableau 3. – Tableau des unités de mesure de l'information

EXERCICE 3.1 · CONVERSIONS

Convertir les quantités suivantes d'une unité à une autre:

A. 10 MB en Mb

B. 1000 kB en GB

C. 250 o en b

D. 1 Gb en GB

E. 500 ko en KiB

3.1.4 Débit

Un débit est la quantité d'information transmise par unité de temps. Il est généralement exprimé en bits par seconde (bit/s ou bps) ou en bytes par

seconde (B/s). Le débit peut varier en fonction de la technologie utilisée, de la qualité du canal de transmission et des protocoles de communication.

EXERCICE 3.2 · CALCUL DU TEMPS DE TÉLÉCHARGEMENT: MODEM 56K

Vous avez peut-être déjà entendu parler du modem 56k. Le « 56k » fait référence à un débit de 56 kilobits par seconde (kbps). En théorie, et avec une telle vitesse, combien de temps faudrait-il pour télécharger un PDF de 5 Mo ?

EXERCICE 3.3 · CALCUL DU TEMPS DE TÉLÉCHARGEMENT: FIBRE OPTIQUE

Une fibre optique peut offrir un débit de 1 Gbps. En théorie, combien de temps faudrait-il pour télécharger un PDF de 5 Mo ?

3.1.5 Latence

La latence est le temps nécessaire pour qu'un signal parcoure un canal de communication, de l'émetteur au récepteur. Elle est généralement mesurée en millisecondes (ms) et peut être influencée par divers facteurs tels que la distance, la qualité du canal, les protocoles utilisés et la congestion du réseau. Elle n'est pas à confondre avec le débit, qui mesure la quantité d'information transmise par unité de temps.

3.1.6 Résumé

Une quantité de données est mesurée en bits ou en octets, et peut être exprimée en multiples tels que kilo, méga, giga, etc. Le débit mesure la quantité d'information transmise par unité de temps, tandis que la latence mesure le temps nécessaire pour qu'un signal parcoure un canal de communication.

3.2 Représentation des nombres

Au cours de l'histoire, l'humanité a appris à compter et à représenter les nombres de différentes manières. Le système décimal est celui que nous utilisons le plus souvent dans la vie quotidienne. Pourquoi « décimal » ? Parce qu'il est basé sur 10 chiffres (0 à 9), probablement parce que nous avons 10 doigts.

Si on généralise, on parle de symbole à la place de chiffre, et de base à la place de dix. Ainsi, le système décimal est un système de base 10, utilisant

10 symboles (0 à 9). La position du symbole dans un nombre représente son poids, par rapport à la base. Le symbole le plus à droite a un poids de 1 (10^0), le suivant a un poids de 10 (10^1), puis 100 (10^2), et ainsi de suite. Par exemple, le nombre 305 en base 10 peut être décomposé comme ceci :

$$305 = 3 \times 10^2 + 0 \times 10^1 + 5 \times 10^0 = 300 + 0 + 5$$

ASTUCE

Si nous avons n positions à disposition pour représenter un nombre en base b, le nombre maximum que nous pouvons représenter est $b^n - 1$. Par exemple, avec 3 positions en base 10, le nombre maximum est $10^3 - 1 = 999$.

3.2.1 Binaire

Le binaire fonctionne de la même manière que le décimal, mais la base est **2**, et le nombre de symboles est également 2 : `0` et `1`. Chaque position dans un nombre binaire représente une puissance de 2, à partir de la droite.

Par exemple, le nombre binaire `1011` peut être décomposé comme ceci :

$$1011_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1 = 11_{10}$$

REMARQUE

Comment savoir si 101 est écrit en binaire ou en décimal ? 101 pourrait représenter le nombre cent un en décimal, ou le nombre cinq en binaire. Pour lever l'ambiguïté, on peut ajouter un indice à la base : 101_{10} pour le décimal, 101_2 pour le binaire ou encore 101_{16} pour l'hexadécimal. On peut aussi utiliser des préfixes (typiquement utilisé dans les langages de programmation) : 0b101 pour le binaire, 0x101 pour l'hexadécimal, et rien pour le décimal.

EXERCICE 3.4 · CONVERSION DE LA BASE 2 À LA BASE 10

Convertir les nombres suivants de la base 2 à la base 10:

A. 1011_2

B. 1101_2

C. 10010_2

D. 111111_2

ASTUCE
Apprenez par coeur les puissances de 2 de 2^0 à 2^{16}, elles vous serviront tout au long de votre vie professionnelle ! La suite est 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536.

EXERCICE 3.5 · VALEURS MAXIMALES SUR n BITS

A. Quelle est la plus grande valeur représentable sur 8 bits ? Sur 16 bits ?

B. Combien de bits faut-il au minimum pour représenter le nombre 1000 ?

3.2.2 Hexadécimal

La base 16, appelée **hexadécimal**, utilise 16 symboles : les chiffres de 0 à 9 et les lettres de A à F (ou a à f) pour représenter les valeurs de 10 à 15. Elle est souvent utilisée en informatique pour représenter des valeurs binaires de manière plus compacte, car chaque chiffre hexadécimal correspond à 4 bits (un **nibble**). Voici un exemple de nombre hexadécimal et sa décomposition mathématique :

$$2F_{16} = 2 \times 16^1 + F \times 16^0 = 2 \times 16 + 15 \times 1 = 32 + 15 = 47_{10}$$

La Tableau 4 montre les symboles hexadécimaux et leurs équivalents décimaux et binaires.

ASTUCE
Pour convertir un nombre hexadécimal en binaire, il suffit de remplacer chaque chiffre hexadécimal par son équivalent binaire à 4 bits. Par exemple, pour convertir le nombre hexadécimal `2F` en binaire, on remplace `2` par `0010` et `F` par `1111`, ce qui donne `00101111`.

Valeur hexadécimale	Valeur décimale	Valeur binaire
0	0	0000
1	1	0001
2	2	0010
3	3	0011

Valeur hexadécimale	Valeur décimale	Valeur binaire
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Tableau 4. – Tableau des symboles hexadécimaux et leurs équivalents décimaux et binaires

EXERCICE 3.6 · CONVERSION DE LA BASE 16 AUX BASES 10 ET 2

Convertir les nombres suivants de la base 16 aux bases 10 et 2:

A. $1\mathrm{A}_{16}$

B. FF_{16}

C. $2\mathrm{B}_{16}$

D. $\mathrm{C3}_{16}$

3.2.3 Conversion de la base 10 à la base 2 et 16

Convertir un nombre de base 2 ou 16 à la base 10 est assez intuitif, on fait la somme des produits de chaque chiffre par sa puissance de base. La conversion inverse, de la base 10 à la base 2 ou 16, est un peu plus complexe et nécessite l'utilisation de divisions successives.

La méthode générale pour convertir un nombre décimal en binaire ou hexadécimal est la suivante :

1. Trouver la plus grande puissance de la base cible (2 pour binaire, 16 pour hexadécimal) qui est inférieure ou égale au nombre décimal.
2. Soustraire cette puissance du nombre décimal et noter le chiffre correspondant (1 pour binaire, le chiffre hexadécimal pour hexadécimal).
3. Répéter le processus avec le reste jusqu'à ce que le reste soit zéro.

Comme en notation décimale, des zéros peuvent être ajoutés à gauche pour compléter le nombre à un certain nombre de bits ou de chiffres hexadécimaux, si nécessaire. Ils n'ont aucune valeur et ne changent pas le nombre représenté (ce qui n'est pas le cas pour les zéros à droite, qui eux, changent la valeur du nombre).

$00000010_2 = 10_2 = 2_{10} = 00000002_{10} = 2_{16} = 002_{16}$

ATTENTION

La représentation octale (base 8) est rarement utilisée dans la programmation moderne, bien qu'on la retrouve encore dans certaines configurations (comme les permissions Unix/Linux : chmod 755) et quelques langages hérités. Historiquement, plusieurs langages utilisaient le préfixe « 0 » seul pour désigner un nombre octal — ainsi, 010 correspondrait à 8 en décimal ($1 \times 8 + 0$), non 10. Cependant, cette notation pose des risques d'erreur : Python 3 exige désormais 0o10, JavaScript (ES6+) préfère aussi 0o au lieu du simple 0. Pour éviter toute ambigüité, privilégiez donc 0o quand vous écrivez littéraux octaux, et évitez de remplir arbitrairement des valeurs numériques de zéros à gauche, sauf si requis par un formatage spécifique.

EXEMPLE 3.1 · CONVERSION DE LA BASE 10 À LA BASE 2

Nous souhaitons convertir le nombre décimal 45 en binaire.

Premièrement on se pose la question « Quelle est la plus grande puissance de 2 qui entre entièrement dans 45 ? 64 est trop grand, 32 est parfait. Donc on note un 1 pour la puissance de 2^5 (32).

Pour l'instant nous avons : 1xxxx_2

Ensuite, on soustrait 32 de 45, ce qui nous laisse avec 13. La plus grande puissance de 2 qui entre dans 13 est 8 (2^3), donc on note un 1 pour la puissance de 2^3. La puissance entre 32 et 8 est 16 (2^4), qui ne rentre pas dans 13, donc on note un 0 pour la puissance de 2^4.

Nous avons maintenant : 101xx_2

On soustrait 8 de 13, ce qui nous laisse avec 5. La plus grande puissance de 2 qui entre dans 5 est 4 (2^2), donc on note un 1 pour la puissance de 2^2.

Nous avons maintenant : 1011x_2

On soustrait 4 de 5, ce qui nous laisse avec 1. La plus grande puissance de 2 qui entre dans 1 est 1 (2^0), donc on note un 1 pour la puissance de 2^0.

Il n'y a pas de reste, donc on arrête ici. Ainsi, le nombre binaire correspondant à 45 est 101101_2. On peut vérifier cela en calculant la valeur décimale de 101101_2 : $1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 0 + 8 + 4 + 0 + 1 = 45$.

EXERCICE 3.7 · CONVERSION DE LA BASE 10 AUX BASES 2 ET 16

Convertir les nombres décimaux suivants :

A. 100_{10} en binaire

B. 173_{10} en binaire

C. 255_{10} en binaire

D. 300_{10} en hexadécimal

E. 4096_{10} en hexadécimal

3.3 Opérations logiques

Un opérateur logique (ou porte logique) est un dispositif qui prend une ou plusieurs entrées binaires (0 ou 1) et produit une sortie binaire en fonction d'une règle logique spécifique. Les opérateurs logiques sont fondamentaux en informatique et en électronique numérique, car ils permettent de manipuler des données binaires et de prendre des décisions basées sur des conditions.

ANECDOTE · Porte logique dans Minecraft

Vous avez peut-être déjà joué à Minecraft, auquel cas vous connaissez sûrement la Redstone. Cet élément du jeu permet de créer des circuits logiques. La Fig. 15 montre un exemple de porte AND réalisée avec de la Redstone dans Minecraft. Les portes logiques dans le jeu fonctionnent de manière similaire à leurs homologues dans le monde réel, permettant aux joueurs de créer des mécanismes complexes et des systèmes automatisés.

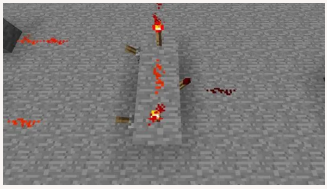


Fig. 15. – Porte AND dans Minecraft

L'ensemble des règles logiques qu'applique un opérateur logique peut être représenté sous forme de table de vérité. Une table de vérité énumère toutes les combinaisons possibles d'entrées et les sorties correspondantes pour un opérateur logique donné. Cela permet de visualiser et de comprendre le comportement de l'opérateur dans différentes situations. Dans ce document et dans le cours, nous nous concentrerons sur les quatre opérateurs logiques de base : AND, OR, NOT et XOR. Nous considérons le cas de deux entrées binaires, mais ces opérateurs peuvent être étendus à plus d'entrées si nécessaire (sauf pour NOT, qui n'a qu'une seule entrée, et le XOR, qui est souvent défini pour deux entrées).

3.3.1 Opérateur ET (AND)

L'opérateur logique ET (AND) nécessite que les deux entrées soient vraies (1) pour que la sortie soit vraie (1). Si l'une ou les deux entrées sont fausses

(0), la sortie sera fausse (0). La table de vérité pour l'opérateur AND est la suivante :

Entrée A	Entrée B	Sortie A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Tableau 5. – Table de vérité pour l'opérateur AND

L'opérateur AND est aussi appelé l'intersection logique, ou le produit logique. Il est noté `AND`, `·`, `&`, ou encore `∧` selon le contexte.

3.3.2 Opérateur OU (OR)

L'opérateur logique OU (OR) nécessite qu'au moins une des deux entrées soit vraie (1) pour que la sortie soit vraie (1). Si les deux entrées sont fausses (0), la sortie sera fausse (0). La table de vérité pour l'opérateur OR est la suivante :

Entrée A	Entrée B	Sortie A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Tableau 6. – Table de vérité pour l'opérateur OR

L'opérateur OR est aussi appelé l'union logique, ou la somme logique. Il est noté `OR`, `+`, ou encore `∨` selon le contexte.

3.3.3 Opérateur NON (NOT)

L'opérateur logique NON (NOT) prend une seule entrée et inverse sa valeur. Si l'entrée est vraie (1), la sortie sera fausse (0), et si l'entrée est fausse (0), la sortie sera vraie (1). La table de vérité pour l'opérateur NOT est la suivante :

Entrée A	Sortie NOT A
0	1

Entrée A	Sortie NOT A
1	0

Tableau 7. – Table de vérité pour l'opérateur NOT

L'opérateur NOT est aussi appelé la négation logique, ou l'inverse logique. Il est noté `NOT`, `!`, ou encore `¬` selon le contexte. En algèbre booléenne, il est souvent représenté par une barre au-dessus de la variable (par exemple, $\overline{A}$ pour NOT A).

3.3.4 Opérateur OU exclusif (XOR)

L'opérateur logique OU exclusif (XOR) nécessite que les deux entrées soient différentes pour que la sortie soit vraie (1). Si les deux entrées sont identiques (les deux vraies ou les deux fausses), la sortie sera fausse (0). La table de vérité pour l'opérateur XOR est la suivante :

Entrée A	Entrée B	Sortie A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Tableau 8. – Table de vérité pour l'opérateur XOR

Le OU exclusif est noté `XOR`, `^`, ou encore `⊕` selon le contexte. Il est souvent utilisé dans les opérations de cryptographie.

APPROFONDISSEMENT · Autres portes logiques

D'autres portes logiques existent, comme NAND, NOR, XNOR, etc. Elles sont souvent utilisées dans la conception de circuits numériques et peuvent être combinées pour créer des fonctions logiques plus complexes. Elles ne sont pas abordées dans ce cours, mais vous pouvez les explorer si vous le souhaitez.

3.3.5 Opérations sur plusieurs bits

Il est possible d'appliquer un opérateur logique sur un ensemble de bits en appliquant l'opérateur sur chaque paire de bits correspondants. Par exemple, si nous avons deux mots binaires de 4 bits chacun :

$A = 1010_2, B = 1100_2$

$A \text{ AND } B = 1000_2$

On appelle souvent cet ensemble de bits un **mot binaire** (**machine word** ou simplement **word** en anglais). Les opérations logiques sur des mots binaires sont courantes en programmation basse niveau, notamment pour la manipulation de masques, le chiffrement simple, ou la compression de données.

REMARQUE

Opération bitwise: Ce terme technique décrit les opérations qui agissent bit par bit sur des mots entiers, plutôt que sur des nombres considérés comme des valeurs arithmétiques globales.

REMARQUE

La notion de **mot** dépend de l'architecture : sur un processeur 32 bits, un **word** correspond généralement à 32 bits et un **half-word** à 16 bits. Dans ces exemples, nous utiliserons des mots de 4 bits pour simplifier les calculs et les représentations.

EXERCICE 3.8 · OPÉRATEURS LOGIQUES FONDAMENTAUX

Appliquez les opérateurs logiques suivants. Calculez le résultat final pour chaque opération ci-dessous:

A. $0 \text{ AND } 1$

B. $1 \text{ XOR } 0$

C. $(0 \text{ AND } 1) \text{ OR } (1 \text{ XOR } 0)$

D. $1 \text{ AND } (0 \text{ OR } 1)$

E. $(1 \text{ XOR } 1) \text{ OR } (0 \text{ AND } 1)$

EXERCICE 3.9 · OPÉRATEURS LOGIQUES SUR DES MOTS BINAIRES

Appliquez les opérateurs logiques sur les mots binaires suivants. Calculez le résultat final pour chaque opération ci-dessous, avec $A = 1010_2$, $B = 1100_2$, et $C = 0110_2$:

A. A AND B

B. A XOR C

C. $(A$ OR $B)$ AND C

D. $(A$ XOR $B)$ OR $(B$ AND $C)$

E. $(A$ AND $C)$ XOR $(B$ OR $C)$

ASTUCE

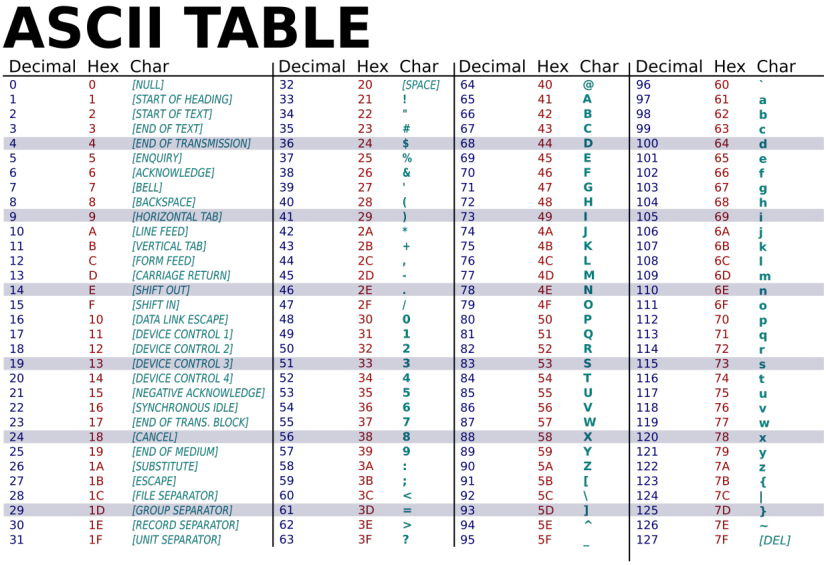
Pour mémoriser rapidement les tables de vérité, associez les opérateurs à leur nom français :

- ET (AND) demande que TOUT soit vrai: sortie 1 seulement si tout est 1
- OU (OR) accepte qu'UN SEUL soit vrai: sortie 0 seulement si tout est 0
- NON (NOT) est un simple inverseur
- XOR signifie « l'un ou l'autre mais pas les deux »: sortie 1 si différents

3.4 Encodage ASCII

L'informatique ne traite que des nombres binaires, mais nous avons besoin de représenter des caractères lisibles par l'homme. Comment faire cela ? C'est là qu'intervient l'encodage de caractères. L'un des encodages les plus connus est l'ASCII (American Standard Code for Information Interchange), qui a été développé dans les années 1960 pour standardiser la représentation des caractères dans les systèmes informatiques. Il utilise 7 bits pour représenter chaque caractère, permettant de coder 128 caractères différents, incluant les lettres majuscules et minuscules, les chiffres, la ponctuation et certains caractères de contrôle. Etant destiné à l'anglais, il ne contient pas de caractères accentués. L'ASCII étendu utilise 8 bits pour représenter 256 caractères, incluant des symboles supplémentaires et des caractères accentués pour certaines langues européennes. De nos jours, les encodages plus modernes comme UTF-8 sont largement utilisés, car ils permettent de représenter un plus grand nombre de caractères, y compris ceux des langues non latines.

En ASCII, chaque caractère est associé à un code numérique unique. Par exemple, le caractère “A” est représenté par le code 65 en décimal, ou 01000001 en binaire. La Fig. 16 montre la table ASCII sur 7 bits, avec les codes décimaux correspondants pour chaque caractère.



ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Fig. 16. – Table ASCII, https://commons.wikimedia.org/wiki/File:ASCII-Table-wide.svg

EXERCICE 3.10 · DÉCODAGE D’UN MESSAGE BINAIRE EN ASCII

Décoder ce mystérieux message binaire, à l’aide de la table ASCII :

01000010 01110010 01100001 01110110 01101111 00100001

EXERCICE 3.11 · ENCODAGE D’UN MESSAGE EN ASCII

À l’aide de la table ASCII, encodez le message « Ok! » en binaire.

CHAPITRE 4

Protocoles et modèles en couches

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir la notion de protocole et citer des exemples de protocoles
- définir le modèle OSI et citer ses couches
- définir le modèle TCP/IP et citer ses couches
- décrire le rôle des couches dans un modèle en couches
- citer les principaux protocoles et normes associés à chaque couche
- citer les principaux organismes de normalisation et leurs rôles
- décrire le principe d'encapsulation et de décapsulation des données

Le monde de la téléinformatique, et de l'informatique en général, est régi par des protocoles. Un protocole est **un ensemble de règles et de conventions** qui définissent comment les données sont transmises et reçues entre les différents périphériques d'un réseau. Les protocoles permettent aux périphériques de communiquer efficacement et de manière fiable, quel que soit le fabricant ou le système d'exploitation utilisé. Il existe de nombreux protocoles, chacun ayant des fonctions spécifiques et étant utilisé dans différents contextes. Par exemple, le protocole HTTP est utilisé pour la communication entre les navigateurs web et les serveurs web, tandis que le protocole FTP est utilisé pour le transfert de fichiers entre les périphériques.

DÉFINITION 4.1 · PROTOCOLE

Ensemble de règles définissant le mode de communication entre deux ordinateurs.

- Larousse.fr

Un réseau informatique est souvent représenté sous forme de couches, chaque couche ayant des fonctions spécifiques et communiquant avec les couches adjacentes. Cette approche en couches permet de simplifier la conception et la mise en oeuvre des réseaux, tout en assurant la compatibilité entre les différents périphériques et protocoles. Les deux modèles de référence les plus connus sont le modèle OSI (Open Systems Interconnection) et le modèle TCP/IP (Transmission Control Protocol/Internet Protocol). Ces modèles définissent les différentes couches d'un réseau et les protocoles associés à chaque couche.

4.1 Modèle OSI

Le modèle OSI (pour « Open Systems Interconnection ») est le modèle de référence, défini par l'ISO (International Organization for Standardization) en 1984. Il est composé de sept couches, chacune ayant des fonctions spécifiques et communiquant avec les couches adjacentes. Les sept couches du modèle OSI sont, de la couche la plus basse à la couche la plus haute : la couche physique, la couche liaison de données, la couche réseau, la couche transport, la couche session, la couche présentation et la couche application.

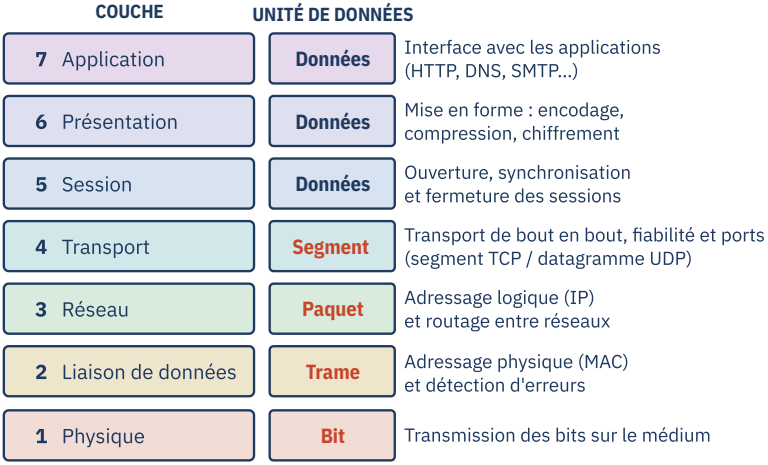


Fig. 17. – Les sept couches du modèle OSI, de la couche physique à la couche application. Chaque couche est responsable d'une partie spécifique du processus de communication et utilise des protocoles spécifiques pour accomplir ses fonctions.

Chaque couche est responsable d'une partie spécifique du processus et est conçue pour dialoguer avec son homologue sur l'autre périphérique, comme si une liaison virtuelle était établie entre elles. Une couche fournit des services à la couche supérieure et utilise les services de la couche inférieure.

Un tel modèle a pour but de promouvoir des normes universelles, d'assurer la compatibilité entre les différents constructeurs, de réaliser des systèmes ouverts et de faciliter la compréhension des réseaux. De plus, il permet de décomposer les problèmes complexes en sous-problèmes plus simples et de garantir l'interconnexion grâce à des interfaces normalisées et standardisées entre les couches, ce qui assure la cohérence globale du système.

Les quatre premières couches du modèle OSI (couches 1 à 4) sont appelées les couches basses, tandis que les trois dernières couches (couches 5 à 7) sont appelées les couches hautes. Les couches basses sont responsables de la transmission des données sur le réseau, tandis que les couches hautes sont responsables de la gestion des applications et des services utilisés par les utilisateurs finaux.

> **ASTUCE**
> Il existe des phrases mnémotechniques pour se souvenir de l'ordre des couches du modèle OSI. Par exemple, la phrase « Petit Lapin Rose Trouvé à la SPA » permet de se souvenir de l'ordre des couches : Physique, Liaison de données, Réseau, Transport, Session, Présentation et Application. Une autre phrase mnémotechnique est « Please Do Not Throw Sausage Pizza Away », qui correspond aux initiales des couches en anglais : Physical, Data Link, Network, Transport, Session, Presentation et Application.

Chaque couche du modèle OSI a sa propre PDU (Protocol Data Unit), qui est l'unité de données utilisée par cette couche pour communiquer avec les autres couches. La PDU de la couche physique est le bit, celle de la couche liaison de données est la trame, celle de la couche réseau est le paquet, celle de la couche transport est le segment, et les PDU des couches session, présentation et application sont les messages (ou « Data »). Cette distinction est importante car elle permet de comprendre et de mettre en lumière les principes d'encapsulation et de décapsulation des données, qui sont essentiels pour la communication entre les périphériques d'un réseau (voir Chapitre 4.3).

REMARQUE
Pour parler des couches du modèle OSI, on utilise souvent l'acronyme «Lx» pour «Layer» + le numéro de la couche. Par exemple «L1» désigne la couche physique, L4 la couche transport, etc.

4.1.1 Couche 1 : Physique

La couche physique est responsable de la transmission des bits sur le support physique. Elle transmet les bits bruts et ne s'occupe ni de la signification des données ni de la gestion des erreurs de transmission (c'est le rôle des couches supérieures). Son rôle est de s'assurer de suivre les spécifications du support physique utilisé pour la transmission des données, comme le type de câble, la vitesse de transmission, le codage des signaux, etc.

Sur la couche 1, on peut citer des normes comme Ethernet (IEEE 802.3), Wi-Fi (IEEE 802.11), Bluetooth (IEEE 802.15.1), USB (Universal Serial Bus), et les normes de câblage comme Cat5e, Cat6, et fibre optique. La PDU de cette couche est le bit.

4.1.2 Couche 2 : Liaison de données

La couche liaison de données fournit aux couches supérieures un service de communication sur un lien physique. Elle est responsable de la détection des erreurs de transmission (mais aucune correction n'est effectuée, une trame erronée est simplement rejetée), ainsi que de la gestion des adresses physiques (adresses MAC) pour identifier les périphériques sur le réseau. Elle organise les bits reçus de la couche physique en trames et assure leur livraison correcte à la couche réseau.

Cette couche permet également de gérer l'accès au support physique partagé, en utilisant des protocoles d'accès au média comme CSMA/CD (Carrier Sense Multiple Access with Collision Detection) pour Ethernet ou CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) pour les réseaux sans fil. Le PDU de cette couche est la trame.

4.1.3 Couche 3 : Réseau

La couche réseau, comme son nom l'indique, ajoute la notion de réseau à la communication. Lorsque des machines deviennent distantes, il n'est plus possible de se contenter d'envoyer des trames sur un lien physique. Il faut désormais savoir comment atteindre la machine distante, et c'est le rôle de la couche réseau. Elle est responsable de l'adressage logique et du routage des paquets de données entre les périphériques sur différents réseaux. La notion de routage apparaît à ce niveau, qui désigne le processus de déter-

mination du chemin optimal pour acheminer les paquets de données d'un périphérique source à un périphérique de destination à travers un réseau.

En plus de ces notions, elle gère la fragmentation et le réassemblage des paquets de données. Les protocoles les plus connus de cette couche sont IPv4, IPv6, ICMP (sert à faire des « ping ») ou encore RIP et OSPF qui sont des protocoles de routage. La PDU de cette couche est le paquet.

4.1.4 Couche 4 : Transport

La couche transport assure la communication de bout en bout entre les applications des périphériques source et destination. Elle introduit la notion de port, qui permet d'identifier les applications sur les périphériques. Elle est responsable de la segmentation et du réassemblage des données, ainsi que de la gestion de la fiabilité et du contrôle de flux. Les protocoles les plus connus de cette couche sont TCP (Transmission Control Protocol) et UDP (User Datagram Protocol). La PDU de cette couche est le segment.

TCP est le protocole qui permet d'assurer une communication fiable, orientée connexion, avec contrôle de flux et retransmission des segments perdus. Il est utilisé pour les applications nécessitant une transmission fiable des données, comme le web (HTTP/HTTPS), le courrier électronique (SMTP, IMAP, POP3) et le transfert de fichiers (FTP).

UDP, quant à lui, est un protocole qui permet d'assurer une communication non fiable, mais rapide (faible overhead et faible latence). Il est utilisé pour les applications nécessitant une transmission rapide des données, comme le streaming vidéo et audio, les jeux en ligne et la VoIP (Voice over IP). L'unité de données est le segment lorsque l'on parle de TCP, et le datagramme lorsque l'on parle d'UDP. Cependant, dans le contexte du modèle OSI, on parle de segment pour les deux protocoles.

4.1.5 Couche 5 : Session

La couche session est responsable de l'établissement, de la gestion et de la terminaison des sessions de communication entre les applications des périphériques source et destination. Elle est aujourd'hui la plus floue des couches du modèle OSI, car elle est souvent implémentée dans les couches supérieures. Elle gère la synchronisation des échanges de données, la reprise après une interruption de communication et la gestion des dialogues entre les applications. Les protocoles de cette couche sont des protocoles comme NetBIOS, RPC ou encore SOCKS. La PDU de cette couche est le message (ou « Data »).

4.1.6 Couche 6 : Présentation

La couche présentation est responsable de l'encodage / décodage des données, de la compression et du chiffrement. Elle assure que les données transmises par la couche application sont compréhensibles par la couche application du périphérique de destination. Comme la couche 5, elle est souvent implémentée dans la couche supérieure. Les couches présentation et session sont souvent regroupées dans la couche application, car elles sont étroitement liées aux applications et aux services utilisés par les utilisateurs finaux. Les protocoles de cette couche sont des protocoles comme SSL/TLS, JPEG, MPEG ou encore ASCII. La PDU de cette couche est le message (ou « Data »).

4.1.7 Couche 7 : Application

La couche application est la couche la plus proche de l'utilisateur final. Elle fournit des services et des interfaces pour les applications et les utilisateurs, permettant l'accès aux ressources du réseau. Elle est responsable de la communication entre les applications des périphériques source et destination, ainsi que de la gestion des protocoles de communication utilisés par ces applications. Les protocoles de cette couche sont des protocoles comme HTTP, FTP, SMTP, DNS ou encore SSH. La PDU de cette couche est le message (ou « Data »).

4.1.8 Résumé des couches du modèle OSI

Le Tableau 9 résume les principales caractéristiques de chaque couche du modèle OSI, y compris son rôle, les protocoles associés et la PDU correspondante. Les exemples de protocoles et de normes mentionnés dans le tableau ne sont pas exhaustifs, mais illustrent les protocoles les plus couramment utilisés pour chaque couche.

Couche	Rôle	Exemples de protocoles / normes	PDU
Physique	Transmission des bits sur le support physique	Ethernet (802.3), Wi-Fi (802.11)	Bit

Couche	Rôle	Exemples de protocoles / normes	PDU
Liaison de données	Communication sur un lien physique, gestion des adresses MAC et de l'accès au support physique partagé	Ethernet (802.3), Wi-Fi (802.11), CSMA/CD	Trame
Réseau	Adressage logique et routage des paquets de données entre les périphériques sur différents réseaux	IPv4, IPv6, ICMP, RIP, OSPF	Paquet
Transport	Communication de bout en bout entre les applications des périphériques source et destination, segmentation et réassemblage des données, gestion de la fiabilité et du contrôle de flux	TCP, UDP	Segment
Session	Établissement, gestion et terminaison des sessions de communication entre les applications des périphériques source et destination	NetBIOS, RPC, SOCKS	Message

Couche	Rôle	Exemples de protocoles / normes	PDU
Présentation	Encodage / décodage des données, compression et chiffrement	SSL/TLS, JPEG, MPEG, ASCII	Message
Application	Fourniture de services et d'interfaces pour les applications et les utilisateurs finaux, communication entre les applications des périphériques source et destination	HTTP, FTP, SMTP, DNS, SSH	Message

Tableau 9. – Résumé des couches du modèle OSI, de la couche physique à la couche application. Chaque couche est décrite avec son rôle, les protocoles associés et la PDU correspondante.

EXERCICE 4.1 · ASSOCIER LES ÉLÉMENTS AUX COUCHES OSI

Pour chaque élément ci-dessous, indiquez la couche du modèle OSI à laquelle il appartient.

A. Le protocole HTTP

B. Une adresse MAC

C. Le protocole TCP

D. Un câble Cat 6

E. Le protocole IP

F. La compression d'une image JPEG

G. La notion de port

4.1.9 Exemple concret : Obtenir une page web

Que se passe-t-il lorsque vous tapez l'adresse d'un site web dans votre navigateur et appuyez sur « Entrée » ? Le processus de communication entre

votre navigateur web et le serveur web implique toutes les couches du modèle OSI, comme illustré par la Fig. 18.

Dans cet exemple, un utilisateur demande une page web en tapant l'URL dans son navigateur. Le navigateur utilise le protocole HTTP (couche application) pour envoyer une requête au serveur web. La requête est ensuite segmentée par la couche transport (TCP), encapsulée dans un paquet par la couche réseau (IP), et enfin encapsulée dans une trame par la couche liaison de données (Ethernet). La trame est transmise sur le support physique par la couche physique (câble Ethernet ou Wi-Fi). De l'autre côté de la chaîne, la requête est décapsulée couche par couche jusqu'à ce que le serveur web reçoive la requête HTTP et renvoie la page web demandée. Le processus inverse se produit pour la réponse du serveur, qui est transmise de la même manière jusqu'à ce que le navigateur affiche la page web à l'utilisateur.

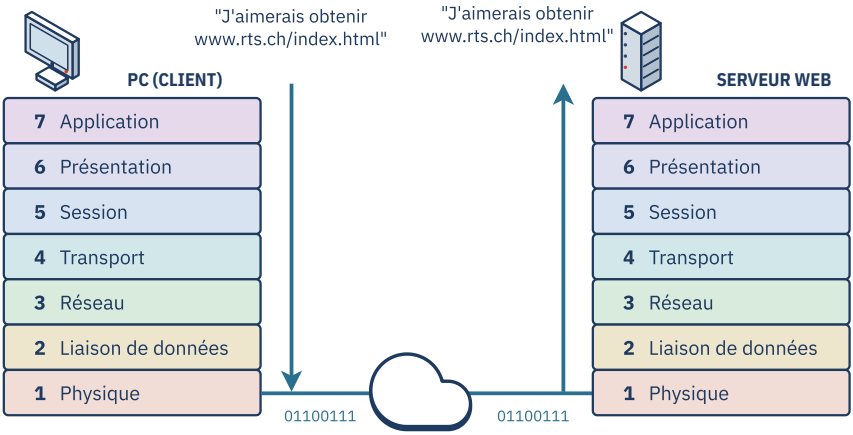


Fig. 18. – Exemple concret d'utilisation du modèle OSI pour obtenir une page web. Chaque couche du modèle OSI est impliquée dans le processus de communication entre le navigateur web et le serveur web, en utilisant les protocoles appropriés à chaque couche. Des bits sont transmis sur le médium physique.

4.2 Modèle TCP/IP

Le modèle TCP/IP (pour « Transmission Control Protocol/Internet Protocol ») est le modèle de référence utilisé pour Internet. Il s'agit d'un modèle simplifié en quatre couches à la place de sept couches. Les quatre

couches du modèle TCP/IP sont, de la plus basse à la plus haute, la couche d'accès réseau, la couche Internet, la couche transport et la couche application. Chaque couche du modèle TCP/IP correspond à une ou plusieurs couches du modèle OSI, comme illustré par la Fig. 19.

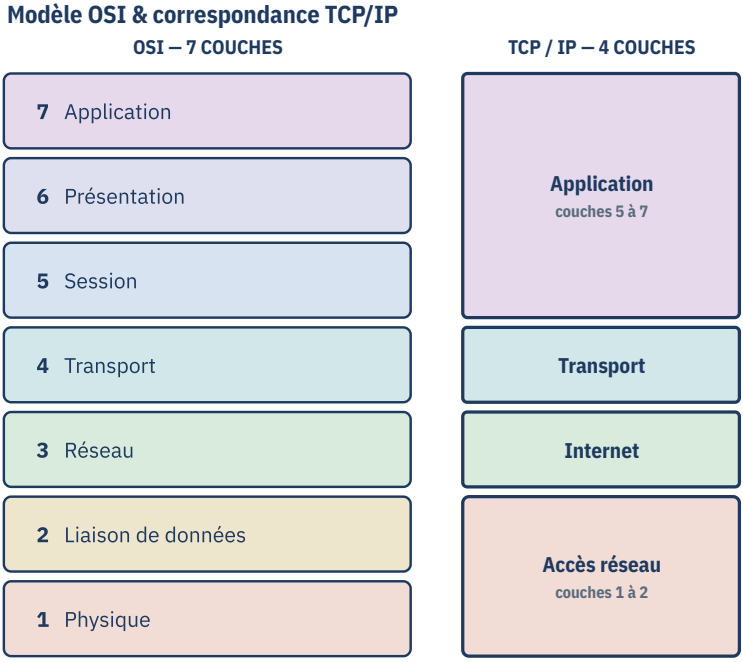


Fig. 19. – Comparaison entre le modèle OSI et le modèle TCP/IP. Le modèle OSI comporte sept couches, tandis que le modèle TCP/IP en comporte quatre. Chaque couche du modèle TCP/IP correspond à une ou plusieurs couches du modèle OSI.

Le Tableau 10 résume les principaux protocoles et normes associés à chaque couche du modèle TCP/IP.

Couche	Rôle	Exemples de protocoles / normes
Accès réseau	Transmission des bits sur le support physique et communication sur un lien physique	Ethernet (802.3), Wi-Fi (802.11), CSMA/CD

Couche	Rôle	Exemples de protocoles / normes
Internet	Adressage logique et routage des paquets de données entre les périphériques sur différents réseaux	IPv4, IPv6, ICMP, RIP, OSPF
Transport	Communication de bout en bout entre les applications des périphériques source et destination, segmentation et réassemblage des données, gestion de la fiabilité et du contrôle de flux	TCP, UDP
Application	Fourniture de services et d'interfaces pour les applications et les utilisateurs finaux, communication entre les applications des périphériques source et destination	HTTP, FTP, SMTP, DNS, SSH

4.2.1 Différences fondamentales entre les modèles OSI et TCP/IP

OSI est un modèle **théorique** utilisé pour comprendre le fonctionnement des réseaux et la communication entre les périphériques. Il est utilisé comme référence pour la conception et l'implémentation des protocoles de communication. TCP/IP, quant à lui, est un modèle **pratique** utilisé pour la communication sur Internet. Il est basé sur un ensemble de protocoles standardisés qui permettent aux périphériques de communiquer efficacement et de manière fiable sur Internet.

Attention toutefois, il ne faut pas voir TCP/IP comme une « implémentation » du modèle OSI. Il s'agit de deux modèles différents, chacun ayant ses propres caractéristiques et objectifs. Les deux ont été développés indépendamment et ont évolué séparément au fil du temps. Cependant, il est possible de faire des correspondances entre les couches des deux modèles, comme illustré par la Fig. 19.

4.2.2 Normes IEEE (802.x)

Les normes IEEE (Institute of Electrical and Electronics Engineers) sont des normes techniques qui définissent les spécifications et les protocoles utili-

sés dans les réseaux informatiques. Elles sont développées par l'IEEE, une organisation internationale à but non lucratif qui regroupe des professionnels de l'ingénierie électrique et électronique. Les normes IEEE sont notées 802.x dans le cadre des réseaux locaux et métropolitains (IEEE 802 LMSC pour « Local and Metropolitan Area Networks Standards Committee »). Elles couvrent un large éventail de technologies, y compris Ethernet (IEEE 802.3), Wi-Fi (IEEE 802.11), Bluetooth (IEEE 802.15) et d'autres technologies de communication sans fil.

On retrouve beaucoup ces normes dans les couches 1 et 2 du modèle OSI, ainsi que dans la couche d'accès réseau du modèle TCP/IP, comme illustré par la Fig. 20.

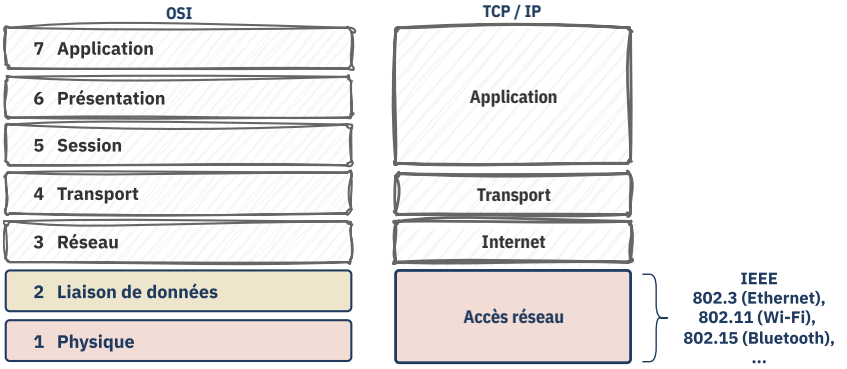


Fig. 20. – Illustration des normes IEEE 802.x et de leur correspondance avec les couches du modèle OSI et du modèle TCP/IP.

4.2.3 Normes IETF (RFC)

Les RFC (Request for Comments) sont des documents publiés par l'IETF (Internet Engineering Task Force) qui définissent les protocoles, les normes et les meilleures pratiques pour Internet. Chaque RFC est identifié par un numéro unique et peut être consulté librement sur le site de l'IETF[2]. Les RFC sont complémentaires aux normes IEEE et sont souvent utilisés pour définir les protocoles de communication utilisés dans les couches supérieures du modèle OSI et du modèle TCP/IP, comme HTTP, FTP, SMTP, DNS ou encore SSH, tel qu'illustré par la Fig. 21.

Voici une liste non exhaustive de quelques RFC importantes et leurs numéros correspondants :

[2] https://www.ietf.org/standards/rfcs/

- RFC 791 : Internet Protocol (IP), définit le protocole IP utilisé pour l'adressage logique et le routage des paquets de données sur Internet.
- RFC 8200 : Internet Protocol version 6 (IPv6), définit la version 6 du protocole IP, qui utilise des adresses sur 128 bits et permet d'adresser un nombre quasi infini d'hôtes.
- RFC 9293 : Transmission Control Protocol (TCP), définit le protocole TCP utilisé pour assurer une communication fiable, orientée connexion, avec contrôle de flux et retransmission des segments perdus.
- RFC 768 : User Datagram Protocol (UDP), définit le protocole UDP utilisé pour assurer une communication non fiable, sans connexion, avec un faible overhead et une faible latence.
- RFC 9110-9112 : Hypertext Transfer Protocol (HTTP), définit le protocole HTTP utilisé pour la communication entre les navigateurs web et les serveurs web.
- RFC 959 : File Transfer Protocol (FTP), définit le protocole FTP utilisé pour le transfert de fichiers entre les périphériques.
- RFC 5321 : Simple Mail Transfer Protocol (SMTP), définit le protocole SMTP utilisé pour l'envoi de courriers électroniques.

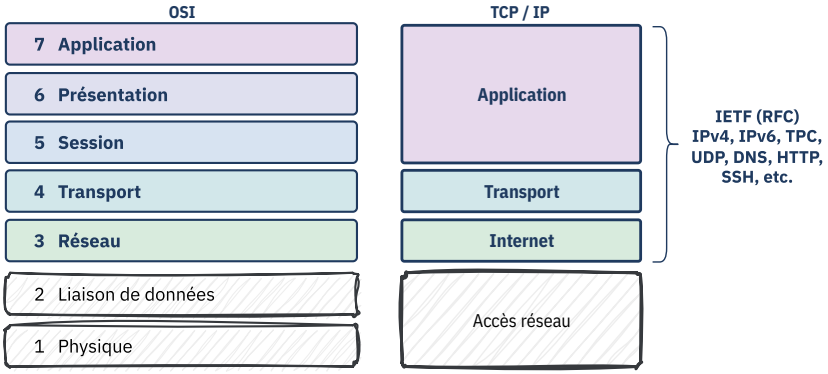


Fig. 21. – Illustration des RFC (Request for Comments) publiées par l'IETF (Internet Engineering Task Force) et de leur correspondance avec les couches du modèle OSI et du modèle TCP/IP.

REMARQUE

Naturellement, il n'est pas nécessaire de connaître les normes par coeur, et leur contenu est parfois assez indigeste. Mais savoir où aller chercher l'information est essentiel pour un ingénieur réseau. Pour

trouver une RFC, il suffit de taper « RFC » suivi du numéro de la RFC dans un moteur de recherche, ou de se rendre directement sur le site de l'IETF. Les normes 802.x de l'IEEE sont également disponibles sur le site de l'IEEE, mais elles sont souvent payantes ou difficiles d'accès.

APPROFONDISSEMENT · Autres organismes de normalisation

IEEE et IETF ne sont pas les seuls organismes de normalisation dans le domaine de l'informatique. On peut citer par exemple l'ISO (International Organization for Standardization), qui est responsable de la définition du modèle OSI, ou encore le W3C qui est responsable de la définition des standards du web, comme HTML, CSS et JavaScript. Dans le cadre du cours, nous nous concentrerons principalement sur les normes IEEE et IETF, car elles sont les plus pertinentes pour la compréhension des réseaux informatiques.

4.3 Encapsulation et décapsulation des données

L'encapsulation et la décapsulation des données sont des concepts fondamentaux dans les modèles en couches. Le principe est que chaque couche du modèle OSI ou TCP/IP ajoute des informations supplémentaires aux données qu'elle reçoit de la couche supérieure avant de les transmettre à la couche inférieure. Ces informations supplémentaires sont appelées en-têtes (ou « headers »).

Lorsqu'une donnée descend dans les couches du modèle, les en-têtes sont ajoutés au fur et à mesure que la donnée est transmise vers la couche inférieure. Ce processus est appelé **encapsulation**. À l'inverse, lorsqu'une donnée remonte dans les couches du modèle, les en-têtes sont retirés au fur et à mesure que la donnée est transmise vers la couche supérieure. Ce processus est appelé **décapsulation**.

La Fig. 22 illustre les entêtes ajoutés à chaque couche du modèle OSI lors de l'encapsulation des données. La couche liaison de données ajoute également une queue (ou « trailer ») à la fin de la trame, qui contient des informations supplémentaires pour la détection des erreurs de transmission. La couche physique ne modifie pas les données, mais se contente de les transmettre sur le support physique.

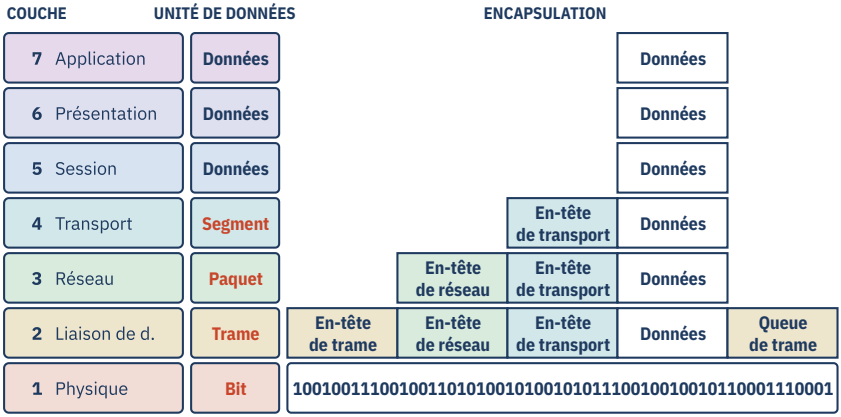


Fig. 22. – Illustration de l'encapsulation des données dans le modèle OSI. Chaque couche ajoute ses propres en-têtes et la couche liaison de données ajoute également une queue.

Prenons une analogie pour illustrer le principe d'encapsulation et de décapsulation des données. Imaginez que vous recevez une lettre par la poste. Cette lettre est transportée par le camion de livraison (L1), qui contient une enveloppe (L2). Sur cette enveloppe, il y a une adresse de destination (L3) et la mention « Urgent » (L4). À l'intérieur de l'enveloppe, il y a une lettre contenant l'objet de la communication (L5). Cette lettre commence par « Chère Madame / Cher Monsieur » (L6). Le contenu de la lettre vous indique que votre inscription à un cours en ligne a été acceptée, il s'agit de la « charge utile » (L7). La Fig. 23 illustre cette analogie.

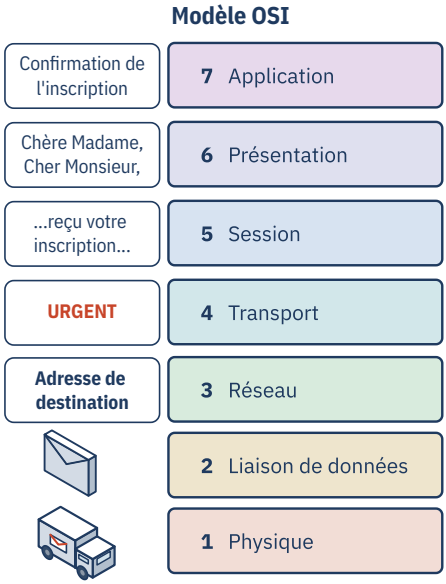


Fig. 23. – Illustration du principe d'encapsulation et de décapsulation des données à travers une analogie avec une lettre envoyée par la poste.

EXERCICE 4.2 · ORDRE D'ENCAPSULATION

Lors de l'envoi d'une requête HTTP, remettez ces PDU dans l'ordre dans lequel elles apparaissent en descendant les couches : trame, message, bits, segment, paquet.

CHAPITRE 5

Matériel

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- identifier les principaux équipements réseau et leur rôle dans un réseau
- identifier les équipements réseaux associés aux différentes couches du modèle OSI, ainsi que leur symbole dans un schéma réseau
- décrire les différents médiums physiques
- décrire les différents types de câbles et leurs caractéristiques
- identifier le connecteur RJ45
- citer les médiums de transmission sans fil et leurs caractéristiques

Afin de faire fonctionner un réseau informatique, différents équipements sont nécessaires. Du switch au pare-feu, en passant par le routeur, chaque équipement a un rôle précis dans le fonctionnement du réseau. Ce chapitre décrit les principaux équipements réseau, ainsi que les différents médiums physiques et sans fil.

5.1 Équipements réseau

Un équipement réseau agit sur un certain nombre de couches du modèle OSI. La Fig. 24 illustre les principaux équipements réseau et les couches du modèle OSI sur lesquelles ils agissent.

ATTENTION

Bien qu'un équipement réseau n'agisse que sur un certain nombre de couches, certains peuvent être configurés à distance avec des protocoles comme SSH (p. ex. switch, routeur, pare-feu). Dans ce cas-là, ils utilisent les 7 couches du côté « management », mais uniquement les couches logiques associées côté **opérationnel**. Il s'agit bien du côté opérationnel qui est représenté sur la Fig. 24.

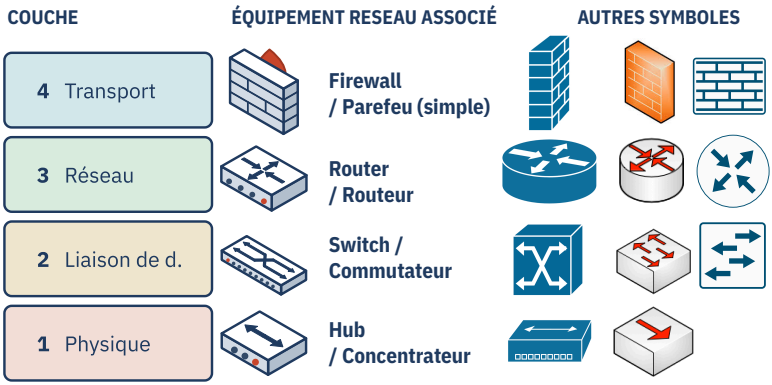


Fig. 24. – Principaux équipements réseau et couches du modèle OSI sur lesquelles ils agissent. Les symboles utilisés dans ce livre sont listés en annexe. La liste des symboles alternatifs est non-exhaustive et peut varier selon les auteurs et les logiciels de dessin.

5.1.1 Hub (Concentrateur)

Le hub (ou « concentrateur » en français) permet de relier plusieurs périphériques réseau entre eux. Il agit uniquement sur la couche physique du modèle OSI. Il ne fait pas de distinction entre les périphériques connectés : lorsqu'il reçoit un signal d'un périphérique, il le retransmet à tous les autres périphériques connectés. Il s'agit du type d'équipement le plus simple et le moins cher, mais il est rarement utilisé dans les réseaux modernes en raison de ses limitations :

- Il ne peut pas gérer le trafic réseau efficacement, ce qui peut entraîner des collisions de données et une baisse des performances.
- Un seul périphérique peut communiquer à la fois avec un autre périphérique, ce qui limite la bande passante disponible pour chaque périphérique.
- Il n'a aucune connaissance des couches supérieures du modèle OSI, ce qui signifie qu'il ne peut pas filtrer ou acheminer les données de manière intelligente. Résultat : tout le monde voit les transmissions de tout le monde.

Le symbole utilisé pour représenter un concentrateur est généralement un rectangle avec une flèche simple ou double à l'intérieur. La Fig. 25 illustre le symbole d'un concentrateur.

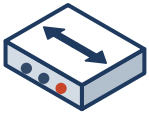


Fig. 25. – Symbole d'un concentrateur (hub).

5.1.2 Switch (Commutateur)

Le switch est la version intelligente du hub. Il agit sur les couches 1 et 2 du modèle OSI. Contrairement au hub, le switch a connaissance des adresses MAC des périphériques connectés (voir le chapitre *Ethernet et IP*) et peut commuter les données uniquement vers le périphérique de destination. Cela permet aux périphériques de communiquer simultanément sans interférer les uns avec les autres, ce qui améliore considérablement les performances du réseau. D'autre part, un périphérique ne reçoit que son propre trafic.

Pour pouvoir savoir à quel périphérique envoyer les données, le switch utilise une table d'adresses MAC, qui est construite dynamiquement en apprenant les adresses MAC des périphériques connectés.

APPROFONDISSEMENT · Switch de niveau 3

Certains switches peuvent également agir sur la couche 3 du modèle OSI, en utilisant des protocoles de routage pour acheminer les données entre différents réseaux. Ces switches sont appelés « switches de niveau 3 » ou « L3 switches ». Ils ne sont pas abordés dans ce cours.

REMARQUE

Un switch à deux ports est appelé un bridge (bien qu'historiquement ce soit l'inverse, un switch est un bridge à plus de 2 ports). À l'instar des hubs, ils ne sont pas beaucoup utilisés, du moins dans leur forme physique. Les bridges logiciels restent en revanche très utilisés dans la virtualisation, par exemple pour relier des machines virtuelles au réseau de l'hôte.

Le symbole du switch est généralement un rectangle avec deux flèches doubles qui s'entrecroisent (ou non) à l'intérieur. La Fig. 26 illustre le symbole d'un switch.

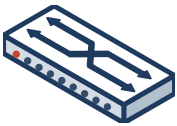


Fig. 26. – Symbole d'un commutateur (switch).

5.1.3 Routeur

Le routeur agit sur les couches 1, 2 et 3 du modèle OSI. Il s'occupe de faire « l'aiguillage » des paquets et de les acheminer vers le réseau de destination. Il utilise une table de routage pour déterminer le meilleur chemin pour chaque paquet. Un routeur est souvent appelé « gateway » (passerelle).

Le symbole du routeur est généralement un cercle (ou un rectangle) avec deux flèches qui pointent à l'intérieur, et deux flèches qui pointent à l'extérieur. La Fig. 27 illustre le symbole d'un routeur.

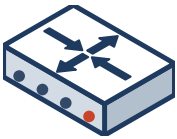


Fig. 27. – Symbole d'un routeur.

> **ATTENTION**
> Une confusion courante est de confondre une « Application Gateway » et un routeur. Une Application Gateway est un type de pare-feu qui agit sur les couches 5 à 7 du modèle OSI, tandis qu'un routeur agit sur les couches 1 à 3. Les deux équipements ont des fonctions différentes et ne doivent pas être confondus.

5.1.4 Pare-feu (Firewall)

Le pare-feu, dans sa version la plus simple, agit sur les couches 1 à 4 du modèle OSI. Il filtre le trafic réseau en fonction de règles prédéfinies, permettant ou bloquant le passage des paquets en fonction de leur adresse IP ou du port utilisé (UDP / TCP). Un pare-feu peut être matériel ou logiciel, et il est souvent utilisé pour protéger les réseaux contre les intrusions et les attaques. Bien souvent, un pare-feu monte jusqu'à la couche 7 du modèle OSI, et ne se limite pas à la couche 4. Il peut alors filtrer le trafic en fonction de l'application utilisée (HTTP, FTP, etc.) ou du contenu des paquets. Pour ce chapitre, la version la plus simple du pare-feu est abordée, c'est-à-dire celle qui agit sur les couches 1 à 4 du modèle OSI.

Le symbole du pare-feu est généralement un mur de brique avec (ou non) une flamme. La Fig. 28 illustre le symbole d'un pare-feu.

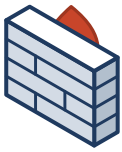


Fig. 28. – Symbole d'un pare-feu (firewall).

5.1.5 Résumé

Chaque équipement réseau agit sur un certain nombre de couches du modèle OSI. Le hub est peu utilisé, contrairement au switch et au routeur. Le pare-feu est un équipement de sécurité qui filtre le trafic réseau en fonction de règles prédéfinies. La Fig. 29 illustre les principaux équipements réseau et les couches du modèle OSI sur lesquelles ils agissent, avec un exemple simple de transmission de données entre un PC et un serveur.

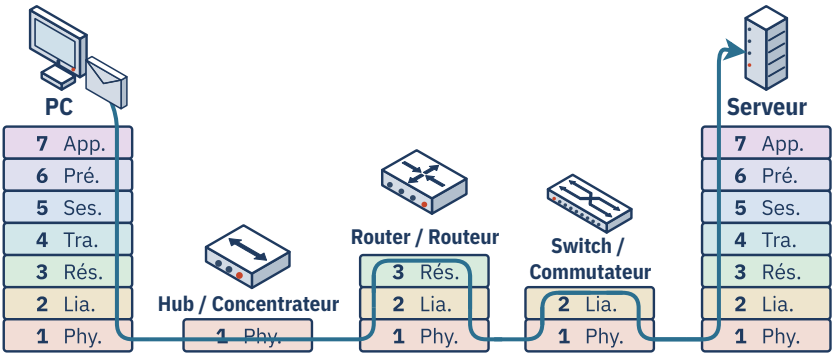


Fig. 29. – Exemple de transmission de données entre un PC et un serveur, avec les principaux équipements réseau et les couches du modèle OSI sur lesquelles ils agissent.

EXERCICE 5.1 · CHOISIR LE BON ÉQUIPEMENT

Pour chaque besoin ci-dessous, indiquez l'équipement approprié : hub, switch, routeur ou pare-feu.

A. Relier les 8 PC d'un bureau au sein d'un même réseau local.

B. Interconnecter le réseau local de l'entreprise avec Internet.

C. Bloquer le trafic entrant sur certains ports.

D. Relier quelques périphériques à moindre coût, en acceptant que chacun voie le trafic de tous les autres.

EXERCICE 5.2 · ÉQUIPEMENTS ET COUCHES OSI

Pour chaque équipement, citez les couches du modèle OSI sur lesquelles il agit (côté opérationnel) : hub, switch, routeur, pare-feu simple.

5.2 Médiums physiques et sans fil

Les médiums physiques sont les supports de transmission des signaux électriques ou optiques dans un réseau. Ils peuvent être classés en deux grandes catégories : les câbles et les médiums sans fil. Chaque support a ses propres caractéristiques, avantages et inconvénients, qui influencent la performance et la fiabilité du réseau. Les principaux médiums physiques sont le cuivre, la fibre optique et les médiums sans fil.

Choisir le bon médium en fonction des besoins et des contraintes fait partie du travail d'un ingénieur réseau.

5.2.1 Cuivre

Les supports de transmission en cuivre sont les plus couramment utilisés dans les réseaux locaux (LAN). Il en existe deux types principaux : les paires torsadées et les câbles coaxiaux. Le cuivre est un bon conducteur électrique, ce qui permet une transmission efficace des signaux électriques.

L'avantage du cuivre est qu'il est économique et facile à installer. Cependant, il est sensible aux interférences électromagnétiques (EMI) et à l'atténuation du signal sur de longues distances. Le cuivre est donc plus adapté aux réseaux locaux (LAN) qu'aux réseaux étendus (WAN).

5.2.1.1 Paires torsadées

Les paires torsadées sont les câbles les plus utilisés dans les réseaux locaux. On les nomme souvent « câble Ethernet » par abus de langage, car ce sont eux qui sont utilisés pour les réseaux Ethernet. Ils sont composés de 4 paires de fils de cuivre torsadés entre eux, ce qui permet de réduire les interférences électromagnétiques. Généralement, la connectique se fait avec des connecteurs RJ45, comme illustré sur la Fig. 30. Les câbles à

paires torsadées peuvent être blindés (STP) ou non blindés (UTP). Le blindage permet de réduire les interférences électromagnétiques, mais il est plus coûteux et moins flexible que les câbles non blindés.

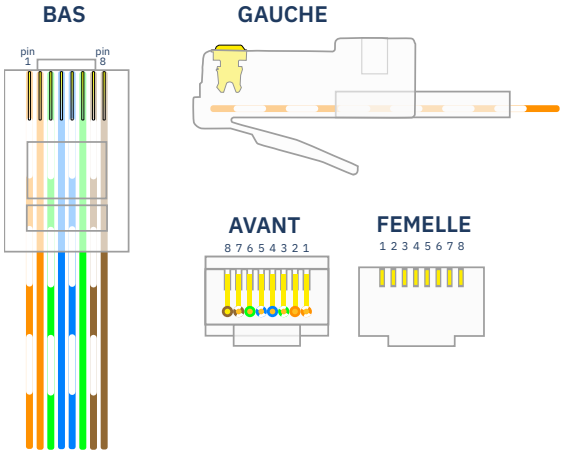


Fig. 30. – Connecteur RJ45 pour câble à paires torsadées.

Il existe différents types de câbles à paires torsadées, classés en fonction de leur catégorie (Cat 5e, Cat 6, Cat 6a, Cat 7, etc.). Chaque catégorie a ses propres caractéristiques de performance, notamment la bande passante et la vitesse de transmission maximale. Le Tableau 11 résume les principales caractéristiques des différentes catégories de câbles à paires torsadées.

Catégorie	Fréquence caractérisée	Débit max (100 m)	Normes Ethernet associées
Cat 5	100 MHz	1 Gbit/s[3]	100BASE-TX, 1000BASE-T
Cat 5e	100 MHz	2.5 Gbit/s	1000BASE-T, 2.5GBASE-T
Cat 6	250 MHz	5 Gbit/s[4]	5GBASE-T
Cat 6a	500 MHz	10 Gbit/s	10GBASE-T
Cat 7	600 MHz	10 Gbit/s	aucune[5]

[3]1000BASE-T a été conçu pour fonctionner sur le parc de câbles Cat 5 existant. En pratique, on installe aujourd'hui du Cat 5e au minimum.

[4]10GBASE-T possible sur Cat 6, mais limité à environ 55 m.

Tableau 11. – Caractéristiques des différentes catégories de câbles à paires torsadées.

APPROFONDISSEMENT · Normes Ethernet associées aux câbles à paires torsadées

La norme Ethernet 802.3 définit les spécifications en termes de débit et de distance pour chaque catégorie de câble. La nomenclature est la suivante:

`xBASE-y`, où `x` est le débit en Mbit/s ou Gbit/s, et `y` est le type de médium utilisé (T pour paires torsadées, S ou L pour fibre optique, etc.). Par exemple, 1000BASE-T correspond à un débit de 1000 Mbit/s (1 Gbit/s) sur un câble à paires torsadées. Les normes Ethernet associées aux câbles à paires torsadées sont listées dans le Tableau 11. Certaines catégories supportent plusieurs normes, cela dépend également du codage utilisé pour la transmission des données.

`BASE` signifie que le signal est transmis en bande de base, c'est-à-dire sans modulation sur une fréquence porteuse. Historiquement on retrouvait aussi le terme `broadband` pour les transmissions modulées sur une fréquence porteuse, mais ce n'est plus utilisé dans les réseaux locaux modernes.

De plus, les anciennes normes comme 10BASE2 et 10BASE5 avaient un digit de fin indiquant la distance maximale en centaines de mètres (2 pour ~200 m, 5 pour ~500 m). Ces normes sont obsolètes et ne sont plus utilisées dans les réseaux locaux modernes.

Deux types de câbles à paires torsadées existent : les câbles droits et les câbles croisés. Entre l'émetteur et le récepteur, les paires doivent être croisées. Il existe deux façons de câbler une interface : MDI (PC, serveurs, routeurs) et MDIX (switches, hubs), qui est câblée de manière inversée. En découle la règle suivante : entre deux équipements câblés de la même manière (deux PC, deux switches, mais aussi un PC et un routeur), on utilise un câble croisé, tandis qu'entre un équipement MDI et un équipement MDIX (par exemple, un ordinateur et un switch), on utilise un câble droit.

[5]Catégorie ISO (classe F) sans norme IEEE dédiée ; supporte 10GBASE-T comme le Cat 6a.

ATTENTION
On entend souvent la règle simplifiée « équipements de même type = câble croisé ». Elle est trompeuse : un PC et un routeur sont des équipements de types différents, mais leurs interfaces sont toutes deux câblées en MDI. Il faut donc un câble croisé entre eux.

La Fig. 31 illustre la différence entre un câble droit et un câble croisé.

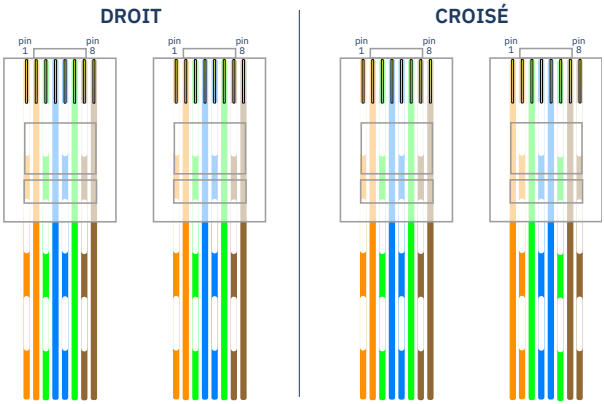


Fig. 31. – Différence entre un câble droit et un câble croisé. Les paires sont croisées entre elles.

Aujourd'hui, l'auto-MDI/MDIX permet de détecter automatiquement le type de câble utilisé et d'adapter la configuration des ports en conséquence. Cependant, il est toujours important de connaître la différence entre un câble droit et un câble croisé, car certains équipements plus anciens ne supportent pas l'auto-MDI/MDIX.

5.2.1.2 Câbles coaxiaux

Les câbles coaxiaux sont composés d'un conducteur central en cuivre, entouré d'un isolant, d'une tresse métallique et d'une gaine extérieure. Ils sont moins utilisés dans les réseaux locaux modernes, mais ils étaient couramment utilisés pour la télévision par câble et les réseaux Ethernet 10BASE2 et 10BASE5. La Fig. 32 illustre un câble coaxial.

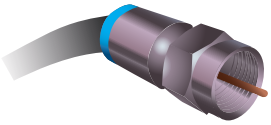


Fig. 32. – Câble coaxial.

5.2.2 Fibre optique

La fibre optique est un médium de transmission qui utilise la lumière pour transmettre les données. Elle est composée d'un coeur en verre ou en plastique, entouré d'une gaine qui réfléchit la lumière et d'une gaine extérieure protectrice. La fibre optique offre des débits très élevés et une grande distance de transmission avec une perte de signal très faible, ce qui la rend idéale pour les réseaux longue distance et les réseaux à haut débit. Elle est également insensible aux interférences électromagnétiques. En revanche, son coût est plus élevé que celui des câbles en cuivre, sa manipulation est plus délicate et elle nécessite des équipements spécifiques pour sa maintenance. La Fig. 33 illustre un câble à fibre optique.

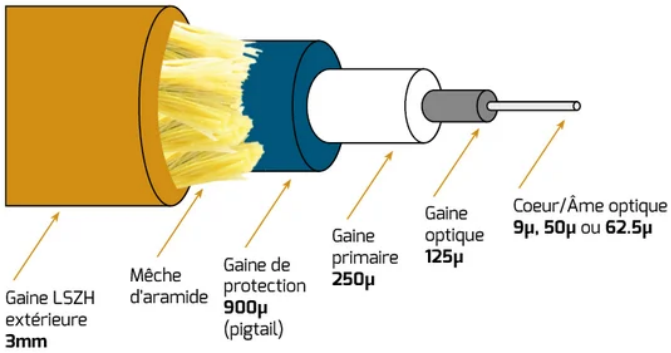


Fig. 33. – Composition d'une fibre optique.[6]

Le débit sur fibre optique peut atteindre plusieurs dizaines de Gbit/s (10, 40, 100 Gbit/s et plus), avec des distances de transmission allant jusqu'à plusieurs dizaines de kilomètres sans répéteur. La fibre optique est donc particulièrement adaptée aux réseaux longue distance et aux réseaux à haut débit. C'est en quelque sorte « l'autoroute de l'information » par rapport aux câbles en cuivre, qui sont plutôt des routes locales.

[6]Source de l'image: https://blog.uniformatic.fr/cable-optique

APPROFONDISSEMENT

Il existe deux types de fibre optique : la fibre monomode et la fibre multimode. La fibre monomode permet une transmission sur de longues distances avec un seul mode de propagation de la lumière, tandis que la fibre multimode permet une transmission sur de courtes distances avec plusieurs modes de propagation. La fibre monomode est généralement utilisée pour les réseaux longue distance, tandis que la fibre multimode est utilisée pour les réseaux locaux et les centres de données. Cette différence n'est pas abordée dans ce cours.

5.2.3 Médiums sans fil

Les médiums sans fil englobent des technologies comme le Wi-Fi, le Bluetooth, la 4G/5G ou encore le LoRa. Toutes ces technologies utilisent des ondes électromagnétiques pour transmettre les données, sur des fréquences radio spécifiques. Les avantages des médiums sans fil sont la mobilité et la facilité d'installation, tandis que les inconvénients incluent la sensibilité aux interférences, aux obstacles physiques ou à la distance, mais aussi la sécurité. En effet, tout périphérique à portée peut potentiellement intercepter les communications, ce qui nécessite des protocoles de sécurité adaptés (WPA2/WPA3 pour le Wi-Fi, chiffrement pour le Bluetooth, etc.).

5.2.3.1 Wi-Fi

Le Wi-Fi est une technologie de réseau local sans fil qui permet aux périphériques de se connecter à un réseau et à Internet sans utiliser de câbles. Il utilise des ondes radio pour transmettre les données entre les périphériques et un point d'accès (AP). Le Wi-Fi est largement utilisé dans les environnements domestiques, les entreprises et les lieux publics. Il existe différentes normes Wi-Fi (802.11a/b/g/n/ac/ax), chacune offrant des débits et des portées différents.

5.2.3.2 Bluetooth

Le Bluetooth est une technologie de communication sans fil à courte portée, utilisée pour connecter des périphériques comme des casques audio, des claviers, des souris ou des smartphones. Il fonctionne sur la bande de fréquence 2,4 GHz et permet des échanges de données à des débits modérés sur des distances généralement inférieures à 100 mètres.

5.2.3.3 4G/5G

La 4G (LTE) et la 5G sont des technologies de communication sans fil à large couverture, utilisées pour connecter des périphériques mobiles à Internet. La 4G offre des débits allant jusqu'à plusieurs centaines de Mbit/s, tandis que la 5G peut atteindre plusieurs Gbit/s et offre une latence très faible. Ces technologies utilisent des bandes de fréquences spécifiques et nécessitent des infrastructures réseau étendues (antennes relais).

5.2.3.4 LoRa

Le LoRa (Long Range) est une technologie de communication sans fil à longue portée et à faible consommation d'énergie, utilisée principalement pour les applications IoT (Internet of Things). Elle permet de connecter des capteurs et des dispositifs sur de grandes distances (jusqu'à plusieurs kilomètres) avec un débit relativement faible. Le LoRa fonctionne sur des bandes de fréquences spécifiques et est particulièrement adapté aux environnements ruraux ou aux applications nécessitant une longue autonomie des dispositifs.

5.2.3.5 Li-Fi

Le Li-Fi est une technologie de communication sans fil qui utilise la lumière visible pour transmettre des données. Elle offre des débits élevés et une faible latence, mais sa portée est limitée et elle nécessite une ligne de vue directe entre l'émetteur et le récepteur. Le Li-Fi est encore en développement et n'est pas largement utilisé dans les réseaux commerciaux.

5.2.4 Exemples d'applications

Un ingénieur réseau doit être capable de choisir le bon médium en fonction des besoins et des contraintes. Ci-dessous, quelques exemples d'applications et de médiums adaptés :

- Réseau local d'entreprise : câbles à paires torsadées
- Réseau longue distance entre deux sites : fibre optique
- Réseau domestique : câbles à paires torsadées ou Wi-Fi

EXERCICE 5.3 · CHOISIR LE BON MÉDIUM

Pour chaque situation, proposez le médium le plus adapté (paires torsadées, fibre optique, Wi-Fi, 4G/5G, LoRa) et justifiez brièvement.

A. Relier deux bâtiments d'un campus distants de 2 km.

B. Connecter le poste de travail fixe d'un développeur.

C. Des capteurs de température répartis dans des champs, envoyant une mesure par heure.

D. L’accès à Internet d’un smartphone en déplacement.

CHAPITRE 6

Accès au médium

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- décrire les domaines de diffusion et de collision dans un réseau
- décrire les types de transmission (half-duplex, full-duplex) et leurs implications sur l'accès au médium
- expliquer le rôle des commutateurs et des routeurs dans la segmentation des domaines de diffusion et de collision
- expliquer le principe de base du Token Ring et son fonctionnement pour gérer l'accès au médium partagé
- expliquer le protocole CSMA/CD et son rôle dans la gestion de l'accès au médium partagé

Ce chapitre traite de la problématique de l'accès à un médium partagé dans un réseau. Il aborde les concepts de domaines de diffusion et de collision, ainsi que les protocoles et techniques utilisés pour gérer l'accès au médium et ainsi éviter les collisions. L'accès au médium est un aspect de la couche 2 (liaison de données) du modèle OSI.

6.1 Domaines de diffusion et de collision

Avant de traiter les techniques qui permettent d'éviter les collisions sur un médium partagé, il est important de comprendre les concepts de domaines de diffusion et de collision.

Un domaine de diffusion est un segment de réseau dans lequel une trame de diffusion (broadcast) est reçue par tous les périphériques.

Un domaine de collision est un segment de réseau dans lequel les trames de données peuvent entrer en collision lorsqu'ils sont envoyés simultanément par plusieurs périphériques. Dans un domaine de collision, une méthode d'accès au médium (Chapitre 6.3) est nécessaire pour éviter les collisions et garantir que les périphériques puissent communiquer efficacement. La

Fig. 34 illustre une collision sur un médium partagé, au sein d'un domaine de collision.

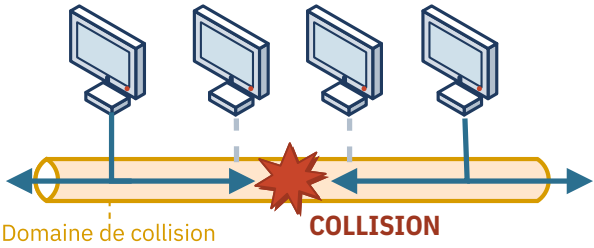


Fig. 34. – Illustration d'une collision sur un médium partagé, au sein d'un domaine de collision. Dans cet exemple, deux périphériques tentent d'envoyer des données en même temps, ce qui entraîne une collision et la perte des trames.

Les domaines de diffusion et de collision sont définis par la topologie du réseau et les équipements utilisés. Les switches séparent les domaines de collision : un domaine de collision par port du switch. Les routeurs séparent les domaines de diffusion : un domaine de diffusion par interface du routeur. Un hub quant à lui ne sépare aucun des domaines. La Fig. 35 illustre les domaines de diffusion et de collision dans un réseau simple.

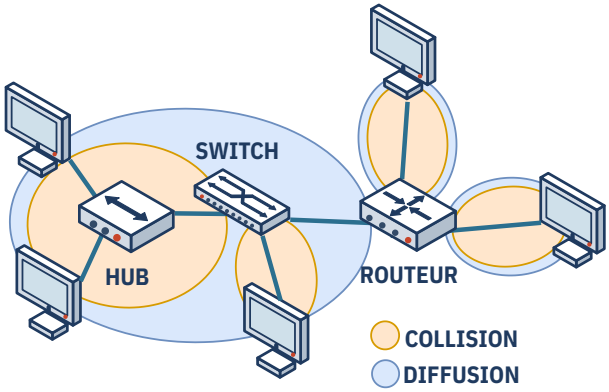


Fig. 35. – Illustration des domaines de diffusion et de collision dans un réseau.

EXERCICE 6.1 · COMPTER LES DOMAINES DE COLLISION ET DE DIFFUSION

Un réseau est construit comme illustré sur la Fig. 36.

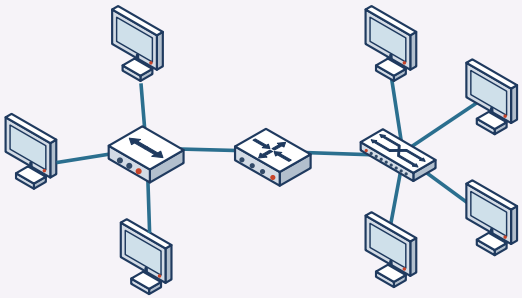


Fig. 36. – Réseau pour l'exercice sur les domaines de diffusion et de collision.

Combien ce réseau compte-t-il de domaines de collision et de domaines de diffusion ?

EXERCICE 6.2 · VRAI OU FAUX : HUB, SWITCH ET ROUTEUR

Pour chaque affirmation, indiquez si elle est vraie ou fausse. Justifiez.

A. Un switch sépare les domaines de diffusion.

B. Un hub sépare les domaines de collision.

C. Chaque interface d'un routeur constitue un domaine de diffusion distinct.

D. Tous les périphériques connectés à un hub appartiennent au même domaine de collision.

6.2 Types de transmission

Dans un réseau, on distingue trois types de transmission : simplex, half-duplex et full-duplex. La Fig. 37 illustre la transmission simplex. Dans ce cas-ci, un seul périphérique peut émettre des données, tandis que l'autre périphérique ne peut que recevoir. Un exemple typique de transmission

simplex est la communication unidirectionnelle, comme la diffusion radio ou la télévision.

Fig. 37. – Transmission simplex : un périphérique émet, l'autre reçoit.

La transmission half-duplex permet aux périphériques d'émettre et de recevoir des données, mais pas simultanément. La Fig. 38 illustre la transmission half-duplex. Un exemple typique de transmission half-duplex est la communication par talkie-walkie.

Fig. 38. – Transmission half-duplex : les périphériques peuvent émettre et recevoir, mais pas simultanément.

La transmission full-duplex permet aux périphériques d'émettre et de recevoir des données simultanément. La Fig. 39 illustre la transmission full-duplex. Un exemple typique de transmission full-duplex est la communication téléphonique moderne, où les deux parties peuvent parler et écouter en même temps. C'est le type de transmission qui est utilisé dans les réseaux Ethernet modernes.

Fig. 39. – Transmission full-duplex : les périphériques peuvent émettre et recevoir simultanément.

Le type de transmission a une conséquence directe sur l'accès au médium : en half-duplex, les deux sens partagent le même canal, des collisions sont possibles et une méthode d'accès au médium (Chapitre 6.3) est indispensable. En full-duplex, chaque sens de transmission dispose de son propre canal (par exemple des paires de cuivre distinctes pour l'émission et la

réception en Ethernet) : il n'y a plus de médium partagé, donc plus de collisions possibles. C'est pourquoi l'Ethernet commuté moderne, full-duplex, n'a plus besoin de CSMA/CD, alors que le Wi-Fi, dont le canal radio reste half-duplex et partagé, utilise toujours CSMA/CA.

ATTENTION
Il existe parfois la nécessité d'utiliser des méthodes d'accès au médium partagé en full-duplex. Cela est typiquement nécessaire pour des réseaux point-à-multipoint (fibre optique partagée par exemple). On peut résumer la nécessité à la phrase suivante : « si le médium possède plusieurs émetteurs potentiels, il est partagé et nécessite une méthode d'accès au médium ».

EXERCICE 6.3 · DUPLEX ET MÉTHODES D'ACCÈS

Expliquez pourquoi l'Ethernet commuté moderne n'a plus besoin de CSMA/CD, alors que le Wi-Fi utilise toujours une méthode d'accès au médium (CSMA/CA).

6.3 Méthodes d'accès au médium

Pour pouvoir gérer l'accès au médium partagé dans les domaines de collision, il existe plusieurs types de méthodes d'accès au médium : par multiplexage, par compétition et par consultation. Ces méthodes permettent de réguler l'accès au médium et d'éviter les collisions entre les périphériques.

6.3.1 Par consultation

Avec les méthodes par consultation, les périphériques se mettent d'accord sur qui utilise le médium à un moment donné. Un exemple de méthode par consultation est le Token Ring, où un jeton circule dans le réseau et permet au périphérique qui le possède d'émettre des données. La Fig. 40 illustre le principe du Token Ring.

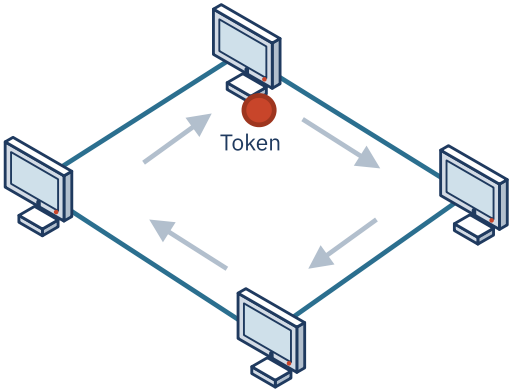


Fig. 40. – Illustration du principe du Token Ring : un jeton circule dans le réseau et permet au périphérique qui le possède d'émettre des données.

Les méthodes par consultation ont l'avantage d'éviter les collisions et d'être déterministes (le temps d'attente maximum pour émettre est connu). Il y a également une équité dans l'accès au médium, car chaque périphérique a la possibilité d'émettre à tour de rôle. Lors de charges très élevées, les méthodes par consultation permettent de garder une performance stable. En revanche, de telles méthodes créent un surcoût en termes de complexité et de latence, car il faut gérer le jeton et la circulation dans le réseau. De plus, si un périphérique tombe en panne ou ne libère pas le jeton, cela peut bloquer l'accès au médium pour tous les autres périphériques.

6.3.2 Par compétition

Les méthodes par compétition permettent aux périphériques de tenter d'accéder au médium en même temps, ce qui peut entraîner des collisions. Le protocole CSMA/CD (Carrier Sense Multiple Access with Collision Detection) est un exemple de méthode par compétition. Il existe également d'autres algorithmes comme « ALOHA » ou CSMA/CA (« équivalent » CSMA/CD pour le Wi-Fi, avec un fonctionnement légèrement différent) qui ne sont pas abordés dans le cours. CSMA/CD était historiquement utilisé pour Ethernet, et fonctionne en détectant si le médium est libre avant d'émettre et en détectant les collisions lorsqu'elles se produisent. La Fig. 41 illustre le principe du CSMA/CD.

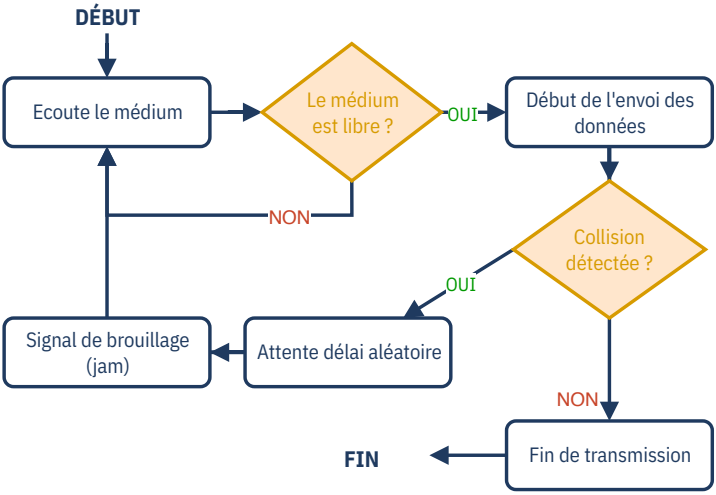


Fig. 41. – Illustration du principe du CSMA/CD : les périphériques écoutent le médium avant d'émettre et détectent les collisions lorsqu'elles se produisent.

Lorsqu'une collision est détectée, les périphériques impliqués attendent un temps aléatoire avant de réessayer d'émettre, ce qui réduit la probabilité de collisions répétées. L'attente aléatoire est gérée par un algorithme appelé « Binary Exponential Backoff », qui augmente le temps d'attente après chaque collision successive.

Les méthodes d'accès par compétition sont simples à mettre en oeuvre, aucun système de gestion centralisé n'est nécessaire. La latence pour émettre est quasi-nulle lorsque le médium est libre. En revanche, ces méthodes sont non déterministes : le temps d'attente pour émettre peut varier considérablement en fonction de la charge du réseau et du nombre de collisions. De plus, elles peuvent entraîner une baisse de performance lors de charges élevées, car les collisions deviennent plus fréquentes et le temps d'attente augmente.

EXERCICE 6.4 · DÉROULER CSMA/CD PAS À PAS

Trois stations A, B et C partagent un médium géré par CSMA/CD. A est en train d'émettre. B et C veulent chacune envoyer une trame. Décrivez, étape par étape, ce qui se passe entre le moment où B et C veulent émettre et le moment où leurs trames sont transmises avec succès.

6.3.3 Par multiplexage

Les méthodes par multiplexage permettent de diviser le médium en plusieurs canaux, chacun pouvant être utilisé par un périphérique différent. Il existe deux types principaux de multiplexage : le multiplexage temporel (TDMA) et le multiplexage fréquentiel (FDMA). La Fig. 42 illustre la différence entre TDMA et FDMA.

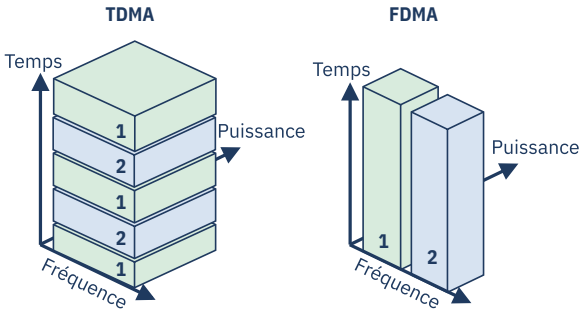


Fig. 42. – Illustration de la différence entre le multiplexage temporel (TDMA) et le multiplexage fréquentiel (FDMA) : dans TDMA, chaque périphérique utilise le médium à tour de rôle dans des intervalles de temps distincts, tandis que dans FDMA, chaque périphérique utilise une bande de fréquence distincte pour transmettre ses données.

Les méthodes par multiplexage ont l'avantage d'éviter les collisions, d'avoir un débit et une latence déterministes, et de permettre une utilisation efficace du médium. Cependant, elles nécessitent une coordination et une planification plus complexes. Si un canal est défini mais jamais utilisé, il est perdu. C'est un type de méthode d'accès au médium qui n'est pas adapté aux communications sporadiques.

6.3.4 Comparaison des méthodes d'accès au médium

Le Tableau 12 compare les différentes méthodes d'accès au médium en termes de complexité, de latence, de débit et de tolérance aux collisions.

Méthode d'accès	Complexité	Latence	Débit	Tolérance aux collisions
Consultation	Élevée	Modérée, bornée (déterministe)	Stable, même en forte charge	Aucune collision
Compétition	Faible	Quasi-nulle à faible charge, imprévisible en forte charge	Se dégrade en forte charge (non déterministe)	Collisions possibles, gérées par backoff
Multiplexage	Moyenne à élevée	Faible, bornée (déterministe)	Garanti par canal, mais capacité perdue si un canal est inutilisé	Aucune collision

Tableau 12. – Comparaison des différentes méthodes d'accès au médium en termes de complexité, de latence, de débit et de tolérance aux collisions.

CHAPITRE 7

Ethernet et IP

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir les adresses IP et MAC, ainsi que leur rôle dans le modèle en couches
- expliquer le rôle des adresses IP et MAC dans la communication entre périphériques sur un réseau
- décrire les différents types d'adresses IP (IPv4, IPv6) et leurs caractéristiques
- expliquer le rôle des masques de sous-réseau et des plages d'adresses dans la segmentation des réseaux
- décrire le rôle des adresses MAC dans la communication sur un lien physique
- expliquer les champs d'une trame Ethernet et d'un paquet IP, ainsi que leur rôle dans la communication entre périphériques sur un réseau

7.1 Introduction aux adresses MAC et IP

Les adresses MAC et IP sont deux éléments essentiels pour l'identification des périphériques sur un réseau. Elles sont utilisées par les couches de bas niveau du modèle OSI et du modèle TCP/IP pour acheminer les données entre les périphériques.

7.1.1 Adresses MAC

Une adresse MAC (pour « Media Access Control ») est une adresse **physique** unique attribuée à chaque périphérique réseau. Elle est utilisée par la couche liaison de données du modèle OSI (L2) et la couche d'accès réseau du modèle TCP/IP pour identifier de manière unique un périphérique dans un réseau **local**.

Les adresses MAC sont non-hiérarchisées et inscrites dans le matériel du périphérique (généralement dans la carte réseau) par le fabricant. Elles

sont généralement représentées sous forme hexadécimale, par exemple `00:1A:2B:3C:4D:5E` (Les `:` sont parfois substitués par des `-`. De plus les lettres peuvent être en majuscules ou en minuscules). Les adresses MAC sont utilisées pour acheminer les trames de données entre les périphériques sur un réseau local, comme un réseau Ethernet ou Wi-Fi.

Dans une adresse MAC, une partie des octets est allouée au fabricant du périphérique, et l'autre partie est un identifiant unique pour le périphérique. Les trois premiers octets (24 bits) de l'adresse MAC sont appelés OUI (Organizationally Unique Identifier) et identifient le fabricant du périphérique. Les trois derniers octets (24 bits) sont un identifiant unique pour le périphérique, attribué par le fabricant. Cela permet de garantir que chaque périphérique réseau dans le monde a une adresse MAC unique.

L'adresse MAC spéciale `FF:FF:FF:FF:FF:FF` est une adresse de diffusion (broadcast) qui permet d'envoyer un message à tous les périphériques d'un même réseau de diffusion.

EXERCICE 7.1

Identifier le constructeur de l'adresse MAC suivante: `78:ac:44:64:7e:4c`. Pour ce faire, vous pouvez utiliser un outil en ligne de recherche d'adresse MAC.

7.1.2 Adresses IP

Une adresse IP (pour adresse « Internet Protocol ») est une adresse **logique** utilisée pour identifier de manière unique un périphérique sur un réseau. Elle est utilisée par la couche réseau du modèle OSI (L3) et la couche Internet du modèle TCP/IP pour acheminer les paquets de données entre les périphériques sur différents réseaux. Il existe deux versions d'adresses IP : IPv4, qui utilise des adresses sur 32 bits, et IPv6, qui utilise des adresses sur 128 bits. Dans le cadre du cours, nous nous concentrerons principalement sur les adresses IPv4, qui sont encore largement utilisées aujourd'hui, bien que l'adoption d'IPv6 soit progressivement en train de se faire. Lorsque nous parlons d'adresses IP dans le cadre du cours, nous faisons référence aux adresses IPv4, sauf indication contraire.

Une adresse IP est hiérarchisée et se compose de deux parties : l'identifiant du réseau et l'identifiant de l'hôte. L'identifiant du réseau permet de déterminer à quel réseau appartient le périphérique, tandis que l'identifiant de l'hôte permet d'identifier de manière unique le périphérique au sein de ce

réseau. Une adresse IP est généralement représentée sous forme décimale pointée, par exemple `192.168.1.24`. Les quatre octets de l'adresse IP sont séparés par des points, et chaque octet est représenté par un nombre compris entre 0 et 255. Contrairement aux adresses MAC, les adresses IP ne sont pas inscrites dans le matériel du périphérique, mais sont attribuées au niveau logique d'un réseau. Il existe plusieurs classes d'adresses IP, qui peuvent être publiques ou privées, en fonction de leur utilisation et de leur portée.

7.1.3 Résumé

Le Tableau 13 résume les principales caractéristiques des adresses MAC et IP, y compris leur rôle, leur format et leur portée.

Caractéristique	Adresse MAC	Adresse IP
Description	Adresse physique	Adresse logique
Couche du modèle OSI	Liaison de données (L2)	Réseau (L3)
Couche du modèle TCP/IP	Accès réseau	Internet
Nature	Non-hiérarchisée, unique pour chaque périphérique, inscrite dans le matériel par le fabricant	Hiérarchisée, classifiée, pas forcément liée à un hôte précis
Format	Hexadécimal, 6 octets (48 bits), séparés par des `:` ou des `-`	Décimale pointée, 4 octets (32 bits), séparés par des `.`
Portée	Utilisée pour identifier les périphériques sur un réseau local	Utilisée pour identifier les périphériques sur différents réseaux, y compris Internet

EXERCICE 7.2 · ADRESSE MAC, IP OU INVALIDE ?

Pour les adresses suivantes, indiquez si elles sont des adresses MAC, IP, ou invalides.

A. `00:1A:2B:3C:4D:5E`

B. `192.168.1.1`

C. `00-1a-2b-3c-4d-5e`

D. `123.456.789.0`

E. `0Z:1A:2B:3C:4D:5E`

7.2 Masque de sous-réseau et plages d'adresses

Pour qu'une adresse IP fasse partie d'un réseau, elle doit être complétée d'un masque de sous-réseau. Le masque de sous-réseau est une séquence de bits qui détermine quelle partie de l'adresse IP correspond à l'identifiant du réseau et quelle partie correspond à l'identifiant de l'hôte. En d'autres termes, le masque permet de dire « de tel bit à tel bit, c'est l'identifiant du réseau, et le reste, c'est l'identifiant de l'hôte ». Le masque de sous-réseau est généralement représenté sous forme décimale pointée, comme une adresse IP, par exemple `255.255.255.0`.

Pour comprendre comment fonctionne un masque de sous-réseau, il est utile de le représenter en binaire. Par exemple, le masque `255.255.255.0` peut être représenté en binaire comme suit : `11111111.11111111.11111111.00000000`. Les bits à 1 indiquent la partie de l'adresse IP qui correspond à l'identifiant du réseau, tandis que les bits à 0 indiquent la partie qui correspond à l'identifiant de l'hôte. Un masque est une suite **continue** de n bits à 1 suivie de m bits à 0, avec $n + m = 32$ (où n peut aller de 0 à 32). Le nombre de bits à 1 dans le masque détermine la taille du réseau et le nombre d'hôtes possibles dans ce réseau. Le masque mentionné ci-dessus contient 24 bits à `1`, ce qui signifie que c'est un « masque à 24 », noté `/24`. Il reste donc 8 bits pour l'identifiant de l'hôte, ce qui permet d'avoir jusqu'à 2^8 = 256 adresses IP dans ce réseau. Cependant, deux adresses sont réservées : l'adresse du réseau (où tous les bits de l'identifiant de l'hôte sont à 0) et l'adresse de diffusion (où tous les bits de l'identifiant de l'hôte sont à 1). Ainsi, dans un réseau avec un masque `/24`, il y a 256 - 2 = 254 adresses IP utilisables pour les périphériques.

En résumé, à partir d'un masque on compte le nombre d'hôtes disponibles avec la formule

$$2^{32-n} - 2$$

où `n` est le nombre de bits à 1 dans le masque. Le nombre d'hôtes disponibles est donc directement lié au masque de sous-réseau utilisé.

DÉFINITION 7.1 · SOUS-RÉSEAU

Un sous-réseau est une subdivision d'un réseau IP plus grand. Il s'agit de l'association d'une adresse IP et d'un masque de sous-réseau, qui permet de déterminer quelles adresses IP appartiennent à ce sous-réseau.

EXERCICE 7.3 · TAILLE DU MASQUE DE SOUS-RÉSEAU

Parmi les masques de sous-réseau suivants, noter la taille de celui-ci ou s'il est invalide.

A. 255.255.254.0

B. 255.255.255.255

C. 255.0.255.0

D. 0.0.0.0

E. 255.255.255.251

EXERCICE 7.4 · NOMBRE D'HÔTES DISPONIBLES

Pour chacun des masques de sous-réseau suivants, calculer le nombre d'hôtes disponibles dans le réseau.

A. /24

B. /26

C. 255.255.240.0

D. /30

E. 255.255.0.0

7.2.1 Adresse de sous-réseau

Pour calculer l'adresse de sous-réseau à partir d'une adresse IP et d'un masque de sous-réseau, on effectue un « AND » binaire entre l'adresse IP et le masque de sous-réseau. Cela permet de déterminer l'identifiant du réseau auquel appartient l'adresse IP. Par exemple, si l'adresse IP est `192.168.0.158` et le masque de sous-réseau est un `/25` (ce qu'on note généralement comme `192.168.0.158/25`), on effectue l'opération suivante :

```
   1 1 0 0 0 0 0 0 . 1 0 1 0 1 0 0 0 . 0 0 0 0 0 0 0 0 . 1 0 0 1 1 1 1 0
AND 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 1 0 0 0 0 0 0 0
    1 1 0 0 0 0 0 0 . 1 0 1 0 1 0 0 0 . 0 0 0 0 0 0 0 0 . 1 0 0 0 0 0 0 0
```

Ce qui, en binaire, équivaut à « garder les bits de l'adresse IP qui correspondent aux bits à 1 du masque et mettre à 0 les bits qui correspondent aux bits à 0 du masque ». Le résultat est l'adresse de sous-réseau, qui est `192.168.0.128`.

EXERCICE 7.5 · CALCUL DE L'ADRESSE DE SOUS-RÉSEAU

Calculer l'adresse de sous-réseau pour les adresses IP et masques de sous-réseau suivants.

A. Adresse IP : `192.168.0.158`, Masque : `255.255.255.128`

B. Adresse IP : `10.0.64.23`, Masque : `255.255.254.0`

C. Adresse IP : `172.16.5.10`, Masque : `255.255.252.0`

D. Adresse IP : `160.98.31.23`, Masque : `255.255.255.0`

7.2.2 Adresse de diffusion (broadcast)

Comme mentionné précédemment, deux adresses sont réservées dans un sous-réseau : l'adresse de sous-réseau et l'adresse de diffusion (broadcast). L'adresse de diffusion est utilisée pour envoyer des messages à tous les périphériques d'un sous-réseau. Pour calculer l'adresse de diffusion, on effectue un « OR » binaire entre l'adresse de sous-réseau et l'inverse du masque de sous-réseau, qu'on appelle aussi « wildcard mask ». Cela revient à mettre tous les bits de l'identifiant de l'hôte à 1, ce qui permet d'adresser tous les périphériques du sous-réseau. Par exemple, si l'adresse de sous-réseau est `192.168.0.0/24`, l'adresse de diffusion est `192.168.0.255`.

Wildcard mask pour un masque de sous-réseau `/24` :

```
NOT 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0 . 1 1 1 1 1 1 1 1
```

Calcul de l'adresse de diffusion :

```
  1 1 0 0 0 0 0 0 . 1 0 1 0 1 0 0 0 . 0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0
OR 0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0 . 1 1 1 1 1 1 1 1
  ---------------------------------------------------------------------
  1 1 0 0 0 0 0 0 . 1 0 1 0 1 0 0 0 . 0 0 0 0 0 0 0 0 . 1 1 1 1 1 1 1 1
```

EXERCICE 7.6 · CALCUL DE L'ADRESSE DE DIFFUSION

Calculer l'adresse de diffusion pour les adresses IP et masques de sous-réseau suivants.

A. Adresse IP : `192.168.0.158` , Masque : `255.255.255.128`

B. Adresse IP : `10.0.64.23` , Masque : `255.255.254.0`

C. Adresse IP : `172.16.5.10` , Masque : `255.255.252.0`

D. Adresse IP : `160.98.31.23` , Masque : `255.255.255.0`

7.2.3 Adresses de multicast

Certaines adresses IP sont réservées pour le multicast, qui permet d'envoyer des messages à un groupe spécifique de périphériques sur un réseau. Les adresses de multicast IPv4 se situent dans la plage `224.0.0.0` à `239.255.255.255` . OSPF (protocole de routage), par exemple, utilise des adresses de la plage `224.0.0.0/24` pour communiquer avec les routeurs OSPF voisins.

APPROFONDISSEMENT

Il n'est pas nécessaire de connaître les plages d'IP multicast, mais simplement savoir que ça existe. Pour information, la liste des plages d'adresses multicast est disponible ici: https://www.iana.org/assignments/multicast-addresses/multicast-addresses.xhtml.

7.2.4 Autres adresses réservées

D'autres adresses IP sont réservées pour des usages spécifiques. Le Tableau 14 liste certaines de ces adresses réservées, ainsi que leur usage (non exhaustif).

Adresse	Usage	Description
127.0.0.1	Loopback	Représente l'hôte local (localhost) d'un périphérique. Les périphériques peuvent s'envoyer des messages à eux-mêmes en utilisant cette adresse.
0.0.0.0	This network	Adresse utilisée pour désigner le réseau lui-même. Par exemple, si un périphérique expose un service sur toutes ses interfaces réseau, il peut utiliser l'adresse `0.0.0.0`
169.254.0.0/16	APIPA	Adresse IP automatique privée (Automatic Private IP Addressing). Si un périphérique ne peut pas obtenir une adresse IP via DHCP, il peut s'auto-attribuer une adresse dans cette plage.
255.255.255.255	Broadcast	Adresse de diffusion limitée. Utilisée pour envoyer des messages à tous les périphériques d'un réseau local, sans regarder le sous-réseau. N'est jamais routée.

7.3 Adresses privées et publiques

Etant donné que les adresses IP sont limitées à 32 bits pour IPv4, il existe un nombre limité d'adresses IP disponibles (environ 4,3 milliards). Pour gérer cette limitation, certaines plages d'adresses IP sont réservées pour un usage privé, tandis que d'autres sont utilisées pour un usage public sur Internet. Les adresses IP privées ne sont pas routables sur Internet et sont utilisées pour les réseaux internes, tandis que les adresses IP publiques sont utilisées pour identifier les périphériques sur Internet.

Les plages d'adresses IP privées sont définies par la RFC 1918 et sont telles que listées dans le Tableau 15. Ces plages sont utilisées pour les réseaux internes, tels que les réseaux domestiques ou d'entreprise, et permettent aux périphériques de communiquer entre eux sans utiliser d'adresses IP publiques.

Classe historique	Bloc CIDR	Plage d'adresses	Nombre d'hôtes possibles
A	10.0.0.0/8	10.0.0.0 – 10.255.255.255	16′777′214
B	172.16.0.0/12	172.16.0.0 – 172.31.255.255	1′048′574
C	192.168.0.0/16	192.168.0.0 – 192.168.255.255	65′534

REMARQUE
Historiquement, les adresses IP étaient divisées en classes de taille fixe : une classe A était un réseau /8, une classe B un réseau /16 et une classe C un réseau /24. Les noms de classes du Tableau 15 viennent de là : la plage privée « classe B » regroupe en réalité 16 réseaux de classe B contigus (172.16.0.0 à 172.31.0.0), et la plage « classe C » regroupe 256 réseaux de classe C (192.168.0.0 à 192.168.255.0). Aujourd'hui, ce découpage en classes n'est plus utilisé : on représente ces plages sous forme de blocs CIDR (Classless Inter-Domain Routing), et l'on peut librement choisir la taille du masque de sous-réseau à l'intérieur de chaque plage, en fonction des besoins du réseau. Le nombre d'hôtes indiqué dans le tableau correspond à cette lecture CIDR, c'est-à-dire au cas où l'on utilise le bloc entier comme un seul réseau.

Toutes les adresses qui ne sont pas dans les plages d'adresses privées sont considérées comme des adresses IP publiques (excepté les adresses réservées mentionnées précédemment). Les adresses IP publiques sont utilisées pour identifier les périphériques sur Internet et sont attribuées par des organismes de gestion des adresses IP, tels que l'IANA (Internet Assigned Numbers Authority) et les RIR (Regional Internet Registries).

EXERCICE 7.7

Déterminer l'adresse IP de votre ordinateur. Pour cela, si vous êtes sur Windows, ouvrez l'invite de commandes et tapez `ipconfig`. Si vous êtes sur macOS ou Linux, ouvrez le terminal et tapez `ifconfig` ou `ip addr`.

EXERCICE 7.8 · PLAGE PRIVÉE OU PUBLIQUE ?

Pour les CIDR suivants, indiquer s'il s'agit d'une plage privée ou publique. Calculer l'adresse de broadcast et le nombre d'hôtes disponibles.

A. `192.168.10.0/24`

B. `10.130.64.0/18`

C. `160.98.20.0/23`

D. `172.20.0.0/14`

E. `172.32.16.0/22`

7.4 Trame Ethernet

Les trames Ethernet telles que définies par la norme IEEE 802.3 sont le format de trame utilisé avec le modèle TCP/IP (couche d'accès réseau) pour la communication sur un réseau local (équivalent couche liaison de données du modèle OSI). Une trame Ethernet est composée de plusieurs champs, chacun ayant un rôle spécifique dans la communication entre périphériques sur un réseau local. Il existe historiquement la trame de type I (Ethernet 802.3) et la trame de type II (Ethernet II). La trame de type II est celle qui est utilisée avec le modèle TCP/IP, et c'est celle qui est détaillée dans ce chapitre.

Les principaux champs d'une trame Ethernet sont les suivants (et illustrés dans la Fig. 43) :

- Préambule et SFD (Start Frame Delimiter) : Il s'agit d'une séquence de 7 octets `0x55` suivie d'un octet `0xD5` (SFD). Le préambule est utilisé pour signaler le début de la trame et permettre aux périphériques de se synchroniser avant la réception des données. Le préambule n'est pas un champ de la trame en soi, mais il est transmis avant la trame pour permettre aux périphériques de se synchroniser.

- En-tête (MAC Header) : contient les informations nécessaires pour acheminer la trame vers le périphérique de destination. L'en-tête comprend :
  - Adresse MAC de destination : l'adresse MAC du périphérique auquel la trame est destinée.
  - Adresse MAC source : l'adresse MAC du périphérique qui envoie la trame.
  - EtherType : un champ de 2 octets qui indique le protocole de la couche supérieure (L3, réseau) utilisé pour traiter les données contenues dans la trame. Par exemple, l'EtherType `0x0800` indique que le protocole de la couche supérieure est IPv4, tandis que `0x86DD` indique IPv6. Ce champ est propre à la trame de type II. Dans la trame de type I, ce champ est remplacé par un champ « Longueur » qui indique la taille de la charge utile.
- Données (Payload) : la charge utile de la trame, qui contient les données à transmettre. La taille minimale de la charge utile est de 46 octets, et la taille maximale est de 1500 octets. Si la charge utile est inférieure à 46 octets, des octets de « remplissage » (padding) sont ajoutés pour atteindre la taille minimale.
- FCS (Frame Check Sequence), ou Checksum : un champ de 4 octets utilisé pour détecter les erreurs de transmission. Le périphérique récepteur calcule la somme de contrôle à partir des données reçues avec l'algorithme CRC (Cyclic Redundancy Check) et la compare à la valeur contenue dans le champ FCS. Si les valeurs ne correspondent pas, cela indique qu'une erreur s'est produite lors de la transmission.

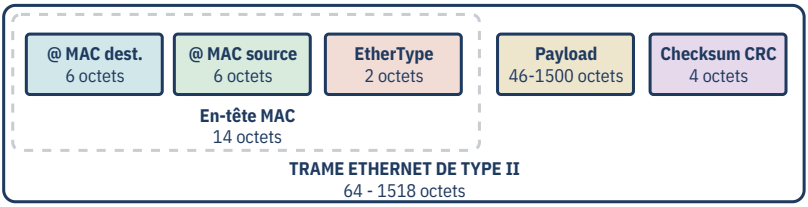


Fig. 43. – Structure d'une trame Ethernet de type II

7.5 Paquet IPv4

Comme mentionné précédemment, le protocole IPv4 est largement utilisé comme protocole de L3, aux côtés d'IPv6. Un paquet IPv4 est le format de paquet utilisé pour la communication sur un réseau utilisant le protocole IPv4. Il est encapsulé dans une trame Ethernet pour être transmis sur un réseau local (EtherType `0x0800`).

Les différents champs d'un paquet IPv4 sont illustrés dans la Fig. 44 et sont les suivants :

- Version : un champ de 4 bits qui indique la version du protocole IP utilisé. Pour IPv4, ce champ est toujours `4`.
- IHL (Internet Header Length) : un champ de 4 bits qui indique la longueur de l'en-tête du paquet IPv4 en mots de 32 bits. La valeur minimale est `5`, ce qui correspond à un en-tête de 20 octets (5×4 octets). Si des options sont présentes dans l'en-tête, la valeur de l'IHL sera supérieure à `5`. Dans la très grande majorité des cas, l'IHL est `5`.
- DSCP (Differentiated Services Code Point) : un champ de 6 bits qui est utilisé pour la qualité de service (QoS) et la priorisation des paquets. Il permet aux périphériques réseau de traiter les paquets en fonction de leur priorité. Particulièrement utile pour les applications sensibles à la latence, comme la voix sur IP (VoIP) ou le streaming vidéo.
- ECN (Explicit Congestion Notification) : un champ de 2 bits qui est utilisé pour signaler la congestion du réseau.
- Total Length : un champ de 16 bits qui indique la longueur totale du paquet IPv4, y compris l'en-tête et les données. La valeur maximale est `65535` ($2^{16} - 1$) octets, ce qui correspond à la taille maximale d'un paquet IPv4.
- IPID (Identification) : un champ de 16 bits qui est utilisé pour identifier de manière unique un paquet IPv4. Il est utilisé pour la fragmentation et le réassemblage des paquets.
- Flags : un champ de 3 bits qui contient des indicateurs pour la fragmentation du paquet.
- Offset de fragmentation (Fragment Offset) : un champ de 13 bits qui indique la position du fragment dans le paquet original. Il est utilisé pour la fragmentation et le réassemblage des paquets.
- TTL (Time to Live) : un champ de 8 bits qui indique le nombre maximum de sauts (hops) qu'un paquet peut effectuer avant d'être rejeté. Chaque fois qu'un paquet traverse un routeur, la valeur du TTL est décrémentée de 1. Si le TTL atteint 0, le paquet est rejeté et un message ICMP « Time Exceeded » est envoyé à l'expéditeur. Ce mécanisme permet d'éviter que

les paquets ne circulent indéfiniment sur le réseau en cas de boucle de routage.

- Protocole : un champ de 8 bits qui indique le protocole de la couche supérieure (L4, transport) utilisé pour traiter les données contenues dans le paquet. Par exemple, le protocole `0x06` indique TCP, tandis que le protocole `0x11` indique UDP.
- Checksum : un champ de 16 bits qui contient la somme de contrôle de l'en-tête IPv4. Il est utilisé pour détecter les erreurs dans l'en-tête du paquet.
- Adresse source : un champ de 32 bits qui contient l'adresse IP source du périphérique qui envoie le paquet.
- Adresse de destination : un champ de 32 bits qui contient l'adresse IP de destination du périphérique auquel le paquet est destiné.
- Options : un champ optionnel qui peut contenir des informations supplémentaires pour le traitement du paquet. La longueur de ce champ est variable et dépend de la valeur de l'IHL. Dans la très grande majorité des cas, ce champ est vide.
- Données (Payload) : la charge utile du paquet, qui contient les données à transmettre. La taille maximale **théorique** est de 65515 octets (65535 - 20 octets d'en-tête), mais en pratique, la taille maximale d'un **paquet** IPv4 est limitée par la MTU (Maximum Transmission Unit) du réseau sur lequel il est transmis. Pour Ethernet, la MTU est généralement de 1500 octets, ce qui signifie que la taille maximale d'un paquet IPv4 transmis sur un réseau Ethernet est de 1500 octets. La taille maximale de la charge utile est de 1480 octets (1500 - 20 octets d'en-tête). Si un paquet IPv4 dépasse la MTU, il doit être fragmenté en plusieurs paquets plus petits pour être transmis sur le réseau.

> **REMARQUE**
> Historiquement, les champs DSCP et ECN étaient regroupés en tant que « ToS » (Type of Service) dans les premières versions du protocole IPv4. Cependant, avec l'évolution des besoins en matière de qualité de service et de gestion de la congestion, ces champs ont été séparés pour permettre une meilleure flexibilité et un contrôle plus précis sur le traitement des paquets.

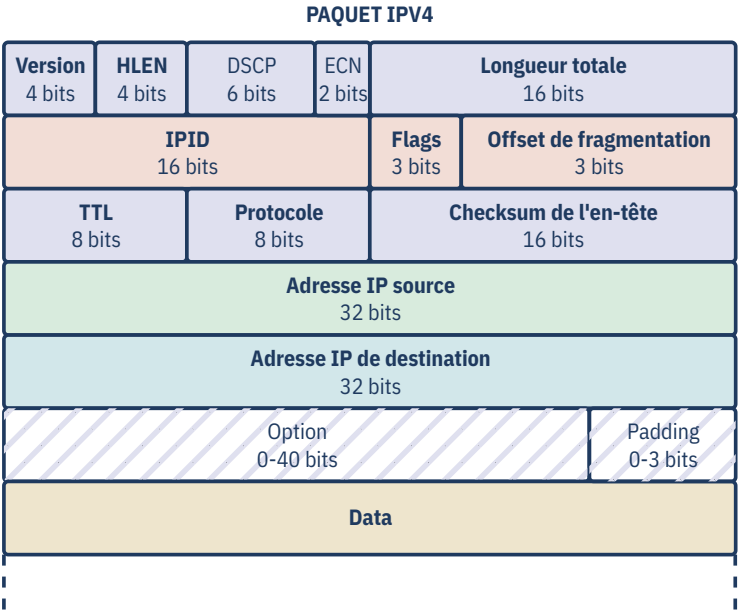


Fig. 44. – Structure d'un paquet IPv4.

Comme mentionné dans la description des champs, un mécanisme de fragmentation est utilisé pour permettre la transmission de paquets plus grands que la MTU d'un réseau. Lorsqu'un paquet IPv4 dépasse la MTU, il est divisé en plusieurs fragments, chacun étant encapsulé dans un paquet IPv4 distinct. Chaque fragment contient une partie des données originales et est identifié par le même IPID. Les fragments sont ensuite réassemblés par le périphérique de destination pour reconstruire le paquet original.

En pratique, on évite autant que possible la fragmentation, car elle peut entraîner une perte de performance et une complexité supplémentaire dans le traitement des paquets. Pour éviter la fragmentation, on utilise généralement le mécanisme de « Path MTU Discovery », qui permet de déterminer la taille maximale de paquet qui peut être transmise sur un chemin réseau sans fragmentation. La séparation d'une donnée en plusieurs paquets est alors déléguée à la couche transport (L4), qui peut segmenter les données

en plusieurs segments plus petits, chacun étant encapsulé dans un paquet IPv4 distinct.

7.6 IPv6

IPv4 présente un problème de taille. Avec seulement 32 bits, le nombre d'adresses IP disponibles est limité à environ 4,3 milliards. Avec l'augmentation du nombre de périphériques connectés à Internet, il est devenu évident que le protocole IPv4 ne pouvait pas répondre aux besoins futurs en matière d'adressage IP. IoT Analytics [9] estime qu'en 2024 18,5 milliards de périphériques IoT (smartphones et ordinateurs exclus) étaient connectés à Internet, et que ce nombre devrait atteindre 39 milliards d'ici 2030.

Pour résoudre ce problème, le protocole IPv6 a été développé. IPv6 utilise des adresses sur 128 bits, ce qui permet d'avoir un nombre d'adresses IP pratiquement illimité (environ 3.4×10^{38}, soit 340 sextillions d'adresses). Cela est plus que le nombre estimé de bactéries sur Terre (de l'ordre de 10^{30}), ce qui permet de garantir que chaque périphérique connecté à Internet peut avoir une adresse IP unique.

En plus du nombre d'adresses disponibles, IPv6 apporte également d'autres améliorations par rapport à IPv4, notamment une simplification de l'en-tête du paquet, une meilleure prise en charge de la qualité de service (QoS), et un mécanisme d'autoconfiguration des adresses IP.

Dans les faits, IPv4 est encore largement utilisé aujourd'hui, mais l'adoption d'IPv6 est en cours et devrait se généraliser dans les années à venir. Les périphériques modernes sont généralement compatibles avec les deux protocoles, et de nombreux réseaux utilisent une combinaison d'IPv4 et d'IPv6 pour assurer la compatibilité avec les périphériques plus anciens tout en tirant parti des avantages d'IPv6. Entre 2012 et 2026, le nombre d'utilisateurs utilisant IPv6 pour atteindre Google est passé d'environ 1% à 50% [10]. Ce chiffre est toutefois à nuancer, car IPv4 reste encore largement utilisé, et par conséquent les 50% utilisent également IPv4 en « dual stack » (c'est-à-dire que les périphériques sont capables de communiquer à la fois en IPv4 et en IPv6). De plus, Google ne représente qu'une partie du trafic Internet mondial, et l'adoption d'IPv6 peut varier selon les études. L'APNIC Labs [11] estime au 13 juillet 2026 que 42% des périphériques sont capables de communiquer en IPv6, avec une grande disparité selon les pays : 83% des périphériques en France contre 10,3% en Espagne.

7.6.1 Structure d'une adresse IPv6

Une adresse IPv6 est composée de 128 bits, soit 16 octets. Elle est généralement représentée sous forme hexadécimale, avec huit groupes de quatre chiffres hexadécimaux séparés par des deux-points. Par exemple, une adresse IPv6 typique pourrait ressembler à ceci : `2001:0db8:85a3:0000:0000:8a2e:0370:7334`. Pour simplifier la notation, les zéros initiaux dans chaque groupe peuvent être omis, et des groupes consécutifs de zéros peuvent être remplacés par `::`.

> **ATTENTION**
> Le raccourci `::` ne peut apparaître qu'une seule fois dans une adresse : si elle contient plusieurs suites distinctes de blocs nuls, une seule peut être compressée, sinon le nombre de blocs omis par chaque `::` serait ambigu.

EXERCICE 7.9 · RACCOURCISSEMENT DES ADRESSES IPV6

Raccourcir la notation des adresses IPv6 suivantes:

A. `2001:0db8:0000:0000:0000:0000:0000:0001`

B. `fe80:0000:0000:0000:0202:b3ff:fe1e:8329`

C. `2001:0db8:0000:0000:0000:0000:0000:0000`

D. `2001:0db8:0000:0000:0001:0000:0000:0001`

7.6.2 Format d'un paquet IPv6

L'en-tête d'un paquet IPv6 est plus grand que celui d'un paquet IPv4 (40 octets contre 20 octets), mais il est également plus simple et plus efficace. Il est composé de plusieurs champs, chacun ayant un rôle spécifique dans la communication entre périphériques sur un réseau utilisant le protocole IPv6. Les principaux champs d'un paquet IPv6 sont les suivants :

- Version : un champ de 4 bits qui indique la version du protocole IP utilisé. Pour IPv6, ce champ est toujours `6`.
- Traffic Class : un champ de 8 bits qui est utilisé pour la qualité de service (QoS). Il est similaire aux champs DSCP et ECN dans un paquet IPv4 et permet aux périphériques réseau de traiter les paquets en fonction de leur priorité.

- Flow Label : un champ de 20 bits qui est utilisé pour identifier les flux de paquets, si on souhaite appliquer des traitements spécifiques à certains flux de données.
- Payload Length : un champ de 16 bits qui indique la longueur de la charge utile du paquet IPv6, c'est-à-dire la taille des données contenues dans le paquet, sans inclure l'en-tête. La valeur maximale est `65535` ($2^{16} - 1$) octets. Comme pour IPv4, la taille maximale d'un paquet IPv6 est limitée par la MTU du réseau sur lequel il est transmis. Pour des trames Ethernet II, la taille maximale d'un paquet IPv6 transmis sur un réseau Ethernet est de 1500 octets. La taille maximale de la charge utile est donc de 1460 octets (1500 - 40 octets d'en-tête).
- Next Header : un champ de 8 bits qui indique le type de l'en-tête suivant. C'est l'équivalent du champ « Protocole » dans un paquet IPv4. Il permet de déterminer le protocole de la couche supérieure (L4, transport) utilisé pour traiter les données contenues dans le paquet. Par exemple, le protocole `0x06` indique TCP, tandis que le protocole `0x11` indique UDP.
- Hop Limit : un champ de 8 bits qui indique le nombre maximum de sauts (hops) qu'un paquet peut effectuer avant d'être rejeté. Chaque fois qu'un paquet traverse un routeur, la valeur du Hop Limit est décrémentée de 1. Si le Hop Limit atteint 0, le paquet est rejeté et un message ICMPv6 « Time Exceeded » est envoyé à l'expéditeur. Ce mécanisme est équivalent au champ TTL dans un paquet IPv4 et permet d'éviter que les paquets ne circulent indéfiniment sur le réseau en cas de boucle de routage.
- Adresse source : un champ de 128 bits qui contient l'adresse IPv6 source du périphérique qui envoie le paquet.
- Adresse de destination : un champ de 128 bits qui contient l'adresse IPv6 de destination du périphérique auquel le paquet est destiné.

La structure complète d'un paquet IPv6 est illustrée dans la Fig. 45. Contrairement à IPv4, l'en-tête d'un paquet IPv6 ne contient pas de champ pour la fragmentation. La fragmentation est gérée par la couche transport (L4) ou par des en-têtes d'extension spécifiques si nécessaire. Cela simplifie le traitement des paquets et améliore les performances du protocole. De plus, l'en-tête d'un paquet IPv6 est de taille fixe (40 octets), ce qui facilite le traitement des paquets par les périphériques réseau. Aucun checksum n'est utilisé dans l'en-tête d'un paquet IPv6, car la vérification de l'intégrité des données est déléguée aux protocoles de la couche transport (L4).

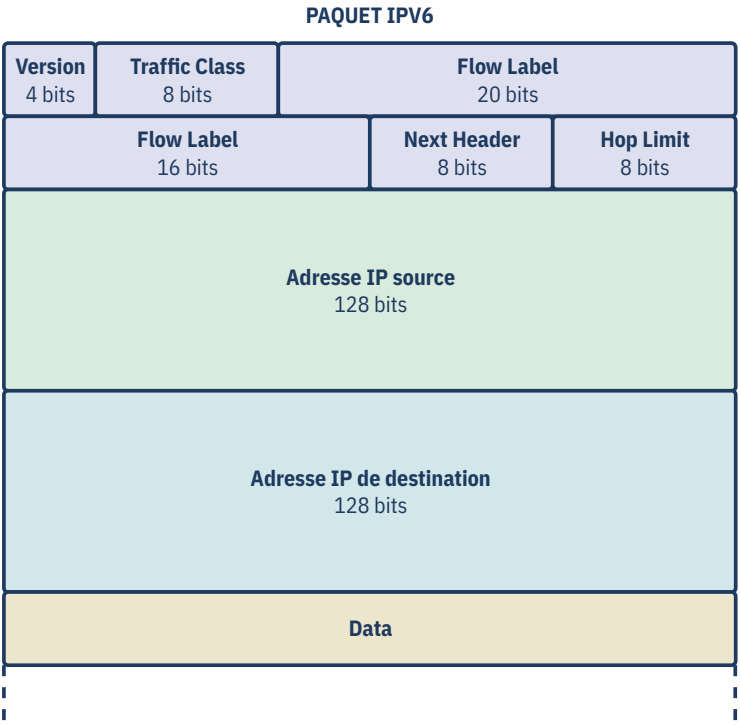


Fig. 45. – Structure d'un paquet IPv6.

ANECDOTE · Pourquoi pas IPv5 ?

Pourquoi IPv6 et pas IPv5 ? Le numéro de version 5 était déjà pris : il avait été attribué dans les années 1980 à ST (Internet Stream Protocol), un protocole expérimental de streaming en temps réel qui n'a finalement jamais été déployé à grande échelle. Lorsque le successeur d'IPv4 a été standardisé, le premier numéro disponible était donc le 6.

Le Tableau 16 résume les correspondances entre les champs de l'en-tête IPv4 et ceux de l'en-tête IPv6.

Champ IPv4	Champ IPv6	Remarque
Version	Version	Valeur `4` en IPv4, `6` en IPv6
IHL	*Supprimé*	L'en-tête IPv6 est de taille fixe (40 octets), sa longueur n'a plus besoin d'être indiquée
DSCP + ECN	Traffic Class	Qualité de service et signalisation de congestion
N'existe pas	Flow Label	Nouveau champ pour identifier les flux de paquets
Total Length	Payload Length	En IPv4, l'en-tête est compris dans la longueur. En IPv6, seule la charge utile est comptée
IPID, Flags, Fragment Offset	*Supprimés*	La fragmentation est gérée par la couche transport
TTL	Hop Limit	Même mécanisme, décrémenté à chaque routeur traversé
Protocole	Next Header	Mêmes valeurs (`0x06` pour TCP, `0x11` pour UDP)
Checksum	*Supprimé*	L'intégrité est vérifiée par la couche liaison (FCS) et la couche transport

Champ IPv4	Champ IPv6	Remarque
Adresses source et destination (32 bits)	Adresses source et destination (128 bits)	Même rôle, taille quadruplée (mais nombres d'adresses disponibles $\times 2^{96}$)

CHAPITRE 8

ARP et ICMP

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- expliquer le rôle du protocole ARP dans la résolution d'adresses IP en adresses MAC
- décrire le fonctionnement du protocole ARP et son processus de résolution d'adresses
- identifier les différents types de messages ARP (ARP Request, ARP Reply)
- décrire le rôle du protocole ICMP dans la communication réseau et le diagnostic des problèmes de connectivité
- nommer les remplaçants d'ARP et ICMP en IPv6 (NDP et ICMPv6)

Ce chapitre aborde deux protocoles essentiels de la couche réseau : le protocole ARP (Address Resolution Protocol) et le protocole ICMP (Internet Control Message Protocol).

8.1 Address Resolution Protocol (ARP)

Le protocole ARP permet de répondre à la question « à quelle adresse MAC correspond cette adresse IP ? ». Lorsqu'on souhaite envoyer un paquet à une adresse IP, il est nécessaire de connaître l'adresse MAC correspondante pour pouvoir l'envoyer sur le réseau local. ARP est utilisé pour résoudre cette correspondance entre adresses IP et adresses MAC.

Il s'agit en réalité d'un protocole à mi-chemin entre la couche réseau et la couche liaison de données. C'est un bon rappel que le modèle OSI est une abstraction, et que dans la pratique, les protocoles ne respectent pas toujours strictement les frontières entre les couches. Néanmoins, pour simplifier la compréhension, on peut considérer ARP comme un protocole de la couche réseau qui interagit avec la couche liaison de données pour résoudre les adresses.

Les périphériques qui utilisent ARP maintiennent une table ARP, qui contient les correspondances entre les adresses IP et les adresses MAC des périphériques du réseau local. Lorsqu'un périphérique souhaite envoyer un paquet à une adresse IP, il vérifie d'abord sa table ARP pour voir s'il connaît déjà l'adresse MAC correspondante. Si ce n'est pas le cas, il envoie une requête ARP (ARP Request) sur le réseau local pour demander l'adresse MAC associée à l'adresse IP cible. Le périphérique possédant cette adresse IP répond avec une réponse ARP (ARP Reply) contenant son adresse MAC.

> **REMARQUE**
> Pourquoi avoir une table ? Cela permet d'avoir un **cache**, c'est-à-dire un accès rapide en mémoire à l'information plutôt que de devoir interroger le réseau à chaque fois. Les entrées de la table ARP ont une durée de vie limitée, après quoi elles sont supprimées pour éviter d'avoir des informations obsolètes.
>
> Le principe de cache en informatique est important et omniprésent, que ce soit au niveau hardware ou logiciel.

Les étapes du processus de résolution d'adresses avec ARP sont les suivantes (illustré par la Fig. 46) :

1. L'émetteur souhaite envoyer un paquet à une adresse IP spécifique.
2. Il vérifie sa table ARP pour voir s'il connaît déjà l'adresse MAC correspondante.
3. Si l'adresse MAC n'est pas trouvée, l'émetteur envoie une requête ARP (ARP Request) sur le réseau local, demandant « Qui possède cette adresse IP ? ». Cette requête est envoyée en broadcast, avec l'adresse MAC de destination `ff:ff:ff:ff:ff:ff`.
4. Le périphérique possédant l'adresse IP cible répond avec une réponse ARP (ARP Reply) contenant son adresse MAC, en unicast vers l'émetteur.
5. L'émetteur reçoit la réponse ARP et met à jour sa table ARP avec la correspondance entre l'adresse IP et l'adresse MAC.
6. L'émetteur peut maintenant envoyer le paquet à l'adresse MAC correspondante.

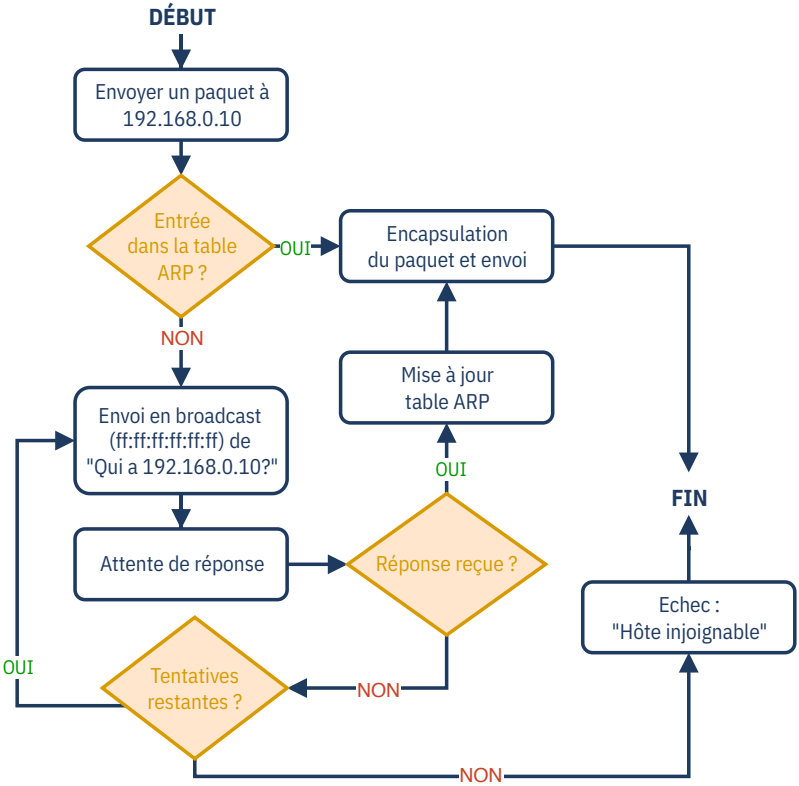


Fig. 46. – Illustration du fonctionnement du protocole ARP du point de vue d'un périphérique qui souhaite envoyer un paquet à une certaine adresse IP. Si personne ne répond après *n* tentatives, le périphérique émetteur abandonne la requête et considère que l'adresse IP est injoignable.

Ce processus fonctionne pour les périphériques sur le même réseau local (même domaine de diffusion), puisque la condition pour que ARP fonctionne est que les périphériques puissent recevoir la trame avec `ff:ff:ff:ff:ff:ff` comme adresse MAC de destination.

Un paquet ARP est encapsulé dans une trame Ethernet, son EtherType est `0x0806`. Il contient les champs suivants (tel qu'illustré par la Fig. 47) :

- Hardware type : indique le type de matériel utilisé (`1` la plupart du temps, pour Ethernet).

- Protocol type : indique le type de protocole utilisé (typiquement `0x0800` pour IPv4).
- Hardware length : la longueur de l'adresse MAC (6 octets pour Ethernet).
- Protocol length : la longueur de l'adresse IP (4 octets pour IPv4).
- Operation (type d'opération) : indique si le paquet est une requête ARP (ARP Request, `0x01`) ou une réponse ARP (ARP Reply, `0x02`).
- Adresse MAC source : l'adresse MAC du périphérique émetteur.
- Adresse IP source : l'adresse IP du périphérique émetteur.
- Adresse MAC de destination : l'adresse MAC du périphérique cible. Mis à `00:00:00:00:00:00` pour une requête ARP, ou l'adresse MAC du périphérique cible pour une réponse ARP.
- Adresse IP de destination : l'adresse IP du périphérique cible. Contient l'adresse IP que l'émetteur souhaite résoudre en adresse MAC si c'est une requête ARP, ou l'adresse IP du demandeur initial si c'est une réponse ARP.

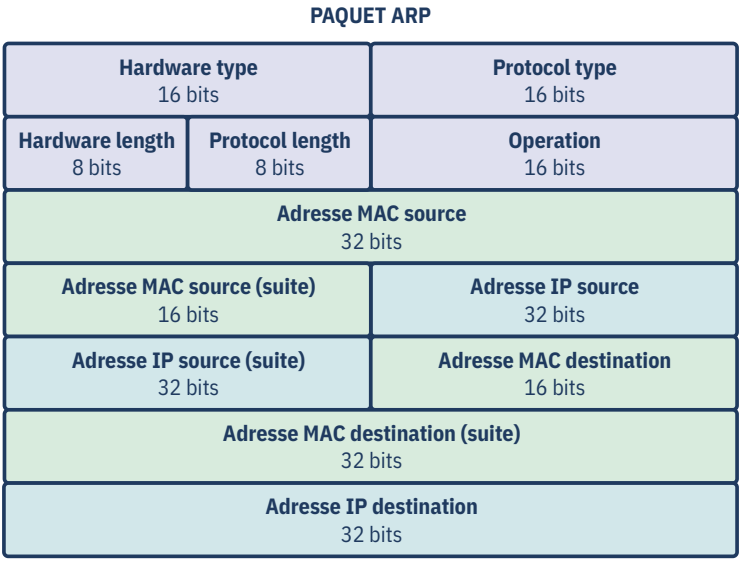


Fig. 47. – Structure d'un paquet ARP : le paquet contient les adresses IP et MAC de l'émetteur et du destinataire, ainsi que le type d'opération (ARP Request ou ARP Reply).

REMARQUE
En théorie, ARP peut fonctionner avec d'autres protocoles que IPv4, mais dans la pratique, il est principalement utilisé avec IPv4. Pour cette raison le schéma dans la Fig. 47 montre la structure d'un paquet ARP pour IPv4.

Si le périphérique cible est sur un autre réseau, la requête ARP ne sera pas reçue et le processus échouera. Comment faire dans un tel cas ? Typiquement, si je souhaite contacter un serveur web sur Internet, il m'est impossible de savoir quelle est son adresse MAC. C'est là qu'intervient le rôle du routeur : il me permet d'envoyer mes paquets à un périphérique sur un autre réseau, en utilisant l'adresse MAC du routeur comme destination. Le routeur se charge ensuite de faire suivre le paquet vers sa destination finale. Pour atteindre un hôte distant, la configuration minimale qu'une machine doit avoir est une adresse IP, un masque de sous-réseau et l'adresse IP du routeur par défaut (« default gateway »).

Pourquoi le masque de sous-réseau est obligatoire pour atteindre un hôte distant ? Le masque de sous-réseau est obligatoire pour déterminer si l'adresse IP de destination se trouve sur le même réseau local ou sur un réseau distant. Si l'adresse IP de destination est sur un réseau distant, le périphérique sait qu'il doit envoyer le paquet au routeur par défaut pour qu'il soit acheminé vers sa destination finale. L'adresse de destination reste l'adresse IP finale du périphérique distant, mais le paquet est encapsulé dans une trame Ethernet avec l'adresse MAC du routeur comme destination. Le routeur se charge ensuite de décapsuler le paquet et de le transmettre vers sa destination finale en utilisant les informations de routage appropriées (voir le chapitre *Routage*).

Si au contraire l'adresse IP de destination est sur le même réseau local, le périphérique peut utiliser ARP pour résoudre l'adresse MAC correspondante et envoyer le paquet directement au périphérique cible.

EXERCICE 8.1 · CONSULTER LA TABLE ARP SUR VOTRE ORDINATEUR

Sur votre ordinateur, consultez la table ARP.

MacOS: `arp -a`

Linux: `ip neigh`

Windows: `arp -a`

- Que contient la table ARP ?
- Quelle adresse IP correspond à l'adresse MAC `ff:ff:ff:ff:ff:ff` ? Pourquoi ?

8.2 Internet Control Message Protocol (ICMP)

ICMP (Internet Control Message Protocol) est un protocole de la couche réseau utilisé pour envoyer des messages de contrôle et d'erreur entre les périphériques réseau. Il est principalement utilisé pour diagnostiquer les problèmes de connectivité et pour informer les périphériques des erreurs rencontrées lors de la transmission des paquets. L'utilisation la plus connue d'ICMP est la commande `ping`, qui envoie des requêtes ICMP « Echo Request » à une adresse IP cible et attend des réponses ICMP « Echo Reply » pour mesurer la latence et vérifier la connectivité.

Ci-dessous une liste non exhaustive des types de messages ICMP les plus courants :

- Type 8 : Requête d'écho (Echo Request), utilisé par la commande `ping` pour tester la connectivité avec un périphérique cible.
- Type 0 : Réponse d'écho (Echo Reply), utilisé pour répondre aux requêtes d'écho ICMP (Type 8) et confirmer que le périphérique cible est joignable.
- Type 3 : Destinataire inaccessible (Destination Unreachable), indique que le périphérique de destination est injoignable pour diverses raisons (réseau, hôte, protocole, port, TTL à zéro, etc.).
- Type 11 : Temps dépassé (Time Exceeded), indique que le paquet a été abandonné parce que son temps de vie (TTL) a expiré avant d'atteindre sa destination.

Un paquet ICMP est encapsulé dans un paquet IP, ce qui signifie qu'il est transporté par le protocole IP. C'est un cas particulier de protocoles de même couche qui sont encapsulés l'un dans l'autre. La Fig. 48 illustre la structure d'un paquet ICMP, qui contient les champs suivants :

- Type : indique le type de message ICMP (par exemple, 8 pour Echo Request, 0 pour Echo Reply).
- Code : fournit des informations supplémentaires sur le type de message ICMP.
- Checksum : utilisé pour vérifier l'intégrité du paquet ICMP.

- Rest of Header : contient des informations spécifiques au type de message ICMP.
- Data : contient les données du message ICMP. La taille est variable.

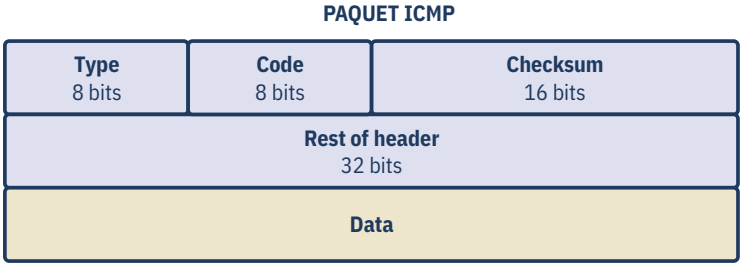


Fig. 48. – Structure d'un paquet ICMP : le paquet contient le type, le code, le checksum, le reste de l'en-tête et les données du message ICMP.

ANECDOTE · Ping of death

Jusqu'en 1997, un simple ping pouvait suffire à faire planter la plupart des systèmes de l'époque. Le « Ping of Death » était une attaque par déni de service (DoS) qui exploitait une faille dans le réassemblage des paquets IP. Un paquet IP ne peut pas dépasser 65 535 octets, mais il peut être découpé en fragments pour être transmis. L'astuce consistait à envoyer des fragments qui, une fois réassemblés, dépassaient cette limite : de nombreux systèmes ne vérifiaient pas ce cas et réservaient une mémoire tampon trop petite, provoquant un dépassement de mémoire tampon et le plantage ou le redémarrage de la machine. Cette vulnérabilité a depuis été corrigée, mais elle reste célèbre pour son extrême simplicité : une seule commande `ping` avec une taille de paquet trop grande suffisait à faire tomber une machine distante.

Plus d'informations sur cette attaque : https://en.wikipedia.org/wiki/Ping_of_death.

EXERCICE 8.2 · TESTER LA CONNECTIVITÉ AVEC ICMP ET LA COMMANDE PING

Envoyer 4 paquets Echo Request à l'adresse IP de www.heia-fr.ch avec la commande `ping` et répondre aux questions suivantes :

A. Quelle est l'adresse IP de www.heia-fr.ch ?

B. Combien de paquets ont été envoyés et combien ont été reçus ?

C. Quelle est la latence moyenne (en ms) pour atteindre www.heia-fr.ch ?

Sur Linux et MacOS, utilisez la commande `ping -c 4 www.heia-fr.ch`. Sur Windows, utilisez la commande `ping -n 4 www.heia-fr.ch`.

8.3 ARP et ICMP avec IPv6

En IPv6, ICMP est remplacé par ICMPv6 et ARP n'est plus utilisé. À la place, le protocole NDP (Neighbor Discovery Protocol) est utilisé pour la découverte des voisins et la résolution d'adresses. NDP utilise ICMPv6 pour envoyer des messages de découverte et de résolution d'adresses entre les périphériques IPv6 sur un réseau local.

CHAPITRE 9

Routage

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir le routage et son rôle dans la communication entre périphériques sur un réseau
- expliquer le rôle des routeurs dans la communication entre périphériques sur différents réseaux
- décrire les différents types de routage (statique, dynamique) et leurs caractéristiques
- expliquer le rôle des protocoles de routage (RIP et OSPF) et leur fonctionnement
- lire, comprendre et remplir les tables de routage de plusieurs réseaux

Le routage est un aspect important de la couche réseau (L3) qui permet de déterminer le chemin que les paquets de données doivent emprunter pour atteindre leur destination. Il est essentiel pour la communication entre périphériques situés sur différents réseaux. Il est géré par les routeurs.

9.1 Définition du routage

Le routage consiste à déterminer le meilleur chemin pour faire transiter un paquet d'un réseau A vers un réseau B. Il est géré par les routeurs, qui utilisent des tables de routage pour décider du chemin à emprunter. Le routage peut être statique ou dynamique, et il existe différents protocoles de routage pour faciliter la communication entre les routeurs.

9.1.1 Principes de base

Le routage se fait sur la couche réseau (L3) du modèle OSI, qui est responsable de l'adressage logique et de la transmission des paquets entre les périphériques. Les routeurs utilisent les adresses IP pour déterminer la destination des paquets et les acheminer vers le réseau approprié.

L'un des principes de base du routage est qu'un routeur n'a pas besoin de connaître l'ensemble du réseau pour acheminer un paquet : il lui suffit de consulter sa table de routage pour savoir vers quel voisin transmettre le paquet. La décision est prise localement, saut par saut, par chaque routeur le long du chemin. Ce principe d'acheminement reste vrai quel que soit le protocole utilisé pour construire la table de routage, y compris pour les protocoles à état de lien vus plus loin, qui construisent pourtant une carte complète du réseau **en amont**, lors du calcul de la table.

Dans la définition ci-dessus, le routage est présenté comme un processus pour déterminer le « meilleur chemin ». Que veut dire meilleur chemin ? La réponse est : ça dépend.

Il existe différents types de métriques pour évaluer la qualité d'un chemin, comme le nombre de sauts (hops), la bande passante, la latence, la fiabilité, etc. Le choix du meilleur chemin dépend du protocole de routage utilisé et des critères définis par l'administrateur réseau.

Il existe deux types de routage : le routage statique et le routage dynamique. Le routage statique consiste à configurer manuellement les routes sur chaque routeur, tandis que le routage dynamique utilise des protocoles de routage pour échanger des informations sur les routes disponibles entre les routeurs. Parmi les protocoles de routage dynamique, on distingue les protocoles à vecteur de distance (comme RIP) et les protocoles à état de lien (comme OSPF). Les protocoles à vecteur de distance utilisent des informations sur la distance (nombre de sauts) pour déterminer le meilleur chemin, tandis que les protocoles à état de lien utilisent des informations sur l'état des liens pour déterminer le meilleur chemin. La connaissance du réseau diffère également entre les deux familles : avec un protocole à vecteur de distance, chaque routeur ne connaît que son voisinage, alors qu'avec un protocole à état de lien, chaque routeur construit une carte complète de la topologie du réseau et l'utilise pour calculer lui-même le meilleur chemin vers chaque destination (typiquement avec l'algorithme de Dijkstra). Cette carte sert uniquement à **construire** la table de routage : une fois celle-ci calculée, le routeur achemine chaque paquet exactement comme décrit plus haut, par un simple lookup local dans sa table, sans reconsulter la carte à chaque paquet. La Fig. 49 illustre les différentes familles de routage.

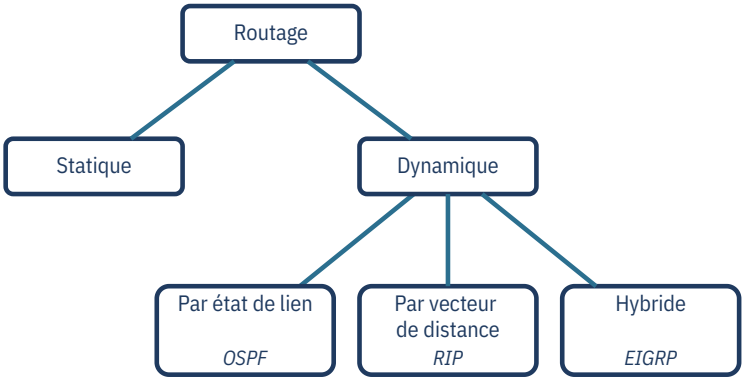


Fig. 49. – Illustration des différentes familles de routage : le routage statique, le routage dynamique à vecteur de distance et le routage dynamique à état de lien. OSPF, RIP et EIGRP sont des exemples de protocoles de routage dynamique. La liste n'est pas exhaustive, il existe d'autres protocoles de routage dynamique.

Dans le cadre du cours, le routage statique et les protocoles RIP et OSPF sont abordés.

9.1.2 Control plane et data plane

Pour bien comprendre la nuance entre « un routeur ne connaît pas tout le réseau » et « OSPF construit une carte complète du réseau », il est utile de distinguer deux plans de fonctionnement d'un routeur :

- Le **control plane** (plan de contrôle) est le processus par lequel un routeur détermine le contenu de sa table de routage. C'est ici que les protocoles de routage entrent en jeu : le routage statique (configuration manuelle), le vecteur de distance (RIP, échange de distances avec les voisins) et l'état de lien (OSPF, diffusion de l'état des liens à tout le réseau puis calcul du plus court chemin) sont trois façons différentes d'alimenter le control plane. C'est uniquement à ce niveau qu'OSPF construit et exploite une carte complète de la topologie.
- Le **data plane** (plan de données) est le processus par lequel le routeur utilise sa table de routage, déjà calculée, pour acheminer chaque paquet reçu.

Cette distinction explique l'apparente contradiction entre les deux principes énoncés plus haut : un routeur OSPF **possède** une carte complète du réseau

(control plane), mais il ne s'en sert pas pour acheminer un paquet individuel (data plane), cette décision reste, dans tous les cas, locale et saut par saut.

9.1.3 Table de routage

La table de routage est une structure de données utilisée par les routeurs pour stocker les informations sur les routes disponibles vers différents réseaux. Elle contient des informations sur les réseaux connus, les interfaces de sortie et les métriques associées à chaque route. La table de routage est utilisée par le routeur pour déterminer le chemin à emprunter pour acheminer un paquet vers sa destination. Elle est construite manuellement (routage statique) ou automatiquement (routage dynamique) en fonction des protocoles de routage utilisés.

Les colonnes typiques d'une table de routage incluent l'adresse réseau, le masque de sous-réseau, l'interface de sortie, la passerelle (gateway) et la métrique. On y trouve également souvent une colonne de protocole de routage, qui indique le protocole utilisé pour apprendre la route. La Tableau 17 illustre un exemple typique de table de routage.

Dest.	Masque	Passerelle	Inter.	Métrique	Protocole
192.168.1.0	/24	—	eth0	0	Connecté
10.0.0.0	/30	—	eth1	0	Connecté
192.168.2.0	/24	10.0.0.2	eth1	1	RIP
172.16.0.0	/16	10.0.0.2	eth1	20	OSPF
0.0.0.0	/0	10.0.0.2	eth1	—	Statique

Tableau 17. – Exemple typique de table de routage.

Dans toute table de routage, il existe généralement une route par défaut (default route) qui est utilisée lorsque le routeur ne trouve pas de correspondance pour l'adresse de destination d'un paquet dans sa table de routage. La route par défaut est souvent représentée par l'adresse réseau `0.0.0.0` avec le masque `0.0.0.0`. Si un routeur n'a pas de route par défaut configurée et qu'il ne trouve pas de correspondance pour l'adresse de destination d'un paquet, il rejette le paquet et envoie potentiellement un paquet ICMP « Destination Unreachable » à l'expéditeur pour l'informer que la destination est inaccessible.

Lorsqu'un routeur consulte sa table, plusieurs routes peuvent correspondre à la même adresse de destination — la route par défaut, par exemple, correspond à toutes les destinations. Dans ce cas, le routeur choisit la route **la plus spécifique**, c'est-à-dire celle dont le masque est le plus long (*longest prefix match*). Dans la table de la Tableau 17, un paquet destiné à `192.168.1.42` correspond à la fois à la route `192.168.1.0/24` et à la route par défaut `0.0.0.0/0` : le routeur choisit la première, dont le masque est plus long. C'est ce mécanisme qui fait de la route par défaut un dernier recours : toute autre route correspondante est nécessairement plus spécifique.

9.2 Processus d'envoi de paquets

Afin d'approfondir la compréhension du routage, il est important de comprendre le processus d'envoi de paquets entre périphériques sur un réseau. Pour cette section, le réseau présenté dans la Fig. 50 est utilisé comme exemple.

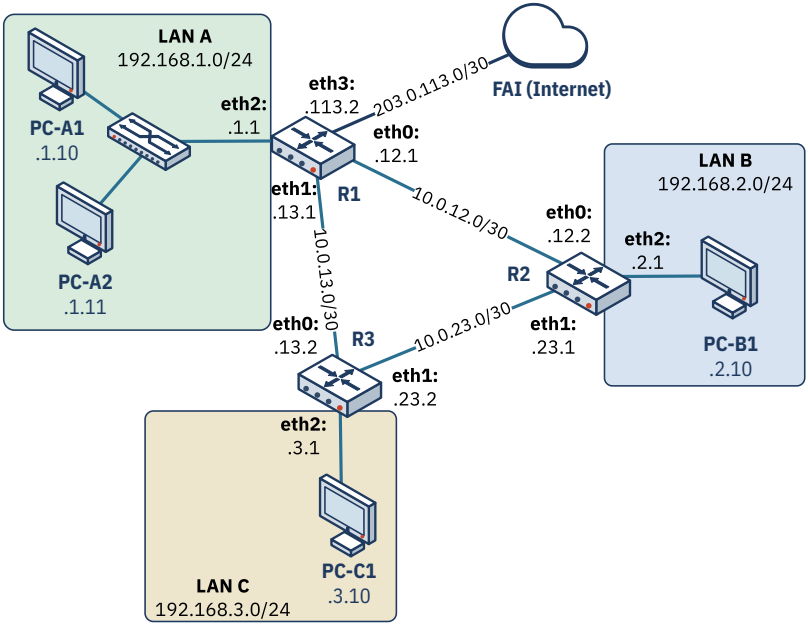


Fig. 50. – Exemple de réseau utilisé pour illustrer le processus d'envoi de paquets entre périphériques sur un réseau. Le réseau est composé de trois réseaux LAN (A, B et C) interconnectés par trois routeurs (R1, R2 et R3). Chaque périphérique est identifié par son adresse IP. Les différentes interfaces des routeurs (eth0, eth1 et eth2) sont également identifiées par leurs adresses IP. R1 a accès à internet par le FAI (Fournisseur d'Accès Internet) à travers l'interface eth3.

EXERCICE 9.1 · LECTURE DU SCHÉMA RÉSEAU

En se basant sur le schéma réseau illustré dans la Fig. 50, répondre aux questions suivantes:

A. Quels sont les sous-réseaux directement connectés au routeur `R2` ?

B. Pourquoi le masque `/30` est-il utilisé entre les routeurs ?

C. Quelle est l'adresse IP de l'interface `eth1` du routeur `R3` ?

D. Par déduction, quelle est l'adresse IP du routeur côté FAI ?

Aux côtés du schéma, la Tableau 18 illustre la table de routage du routeur R1, qui est utilisée pour déterminer le chemin à emprunter pour acheminer les paquets vers les différents réseaux. Les tables de routage des routeurs R2 et R3 sont illustrées dans les Tableau 19 et Tableau 20 respectivement.

Destination	Gateway	Interface	Protocole	Métrique
192.168.1.0/24	—	eth2	Connecté	0
10.0.12.0/30	—	eth0	Connecté	0
10.0.13.0/30	—	eth1	Connecté	0
203.0.113.0/30	—	eth3	Connecté	0
192.168.2.0/24	10.0.12.2 (R2)	eth0	RIP	1
192.168.3.0/24	10.0.13.2 (R3)	eth1	RIP	1
10.0.23.0/30	10.0.12.2 (R2)	eth0	RIP	1
0.0.0.0/0	203.0.113.1 (FAI)	eth3	Statique	—

Tableau 18. – Table de routage du routeur R1.

Destination	Gateway	Interface	Protocole	Métrique
192.168.2.0/24	—	eth2	Connecté	0
10.0.12.0/30	—	eth0	Connecté	0
10.0.23.0/30	—	eth1	Connecté	0
192.168.1.0/24	10.0.12.1 (R1)	eth0	RIP	1
192.168.3.0/24	10.0.23.2 (R3)	eth1	RIP	1
10.0.13.0/30	10.0.12.1 (R1)	eth0	RIP	1

Destination	Gateway	Interface	Protocole	Métrique
0.0.0.0/0	10.0.12.1 (R1)	eth0	Statique	—

Tableau 19. – Table de routage du routeur R2.

Destination	Gateway	Interface	Protocole	Métrique
192.168.3.0/24	—	eth2	Connecté	0
10.0.13.0/30	—	eth0	Connecté	0
10.0.23.0/30	—	eth1	Connecté	0
192.168.1.0/24	10.0.13.1 (R1)	eth0	RIP	1
192.168.2.0/24	10.0.23.1 (R2)	eth1	RIP	1
10.0.12.0/30	10.0.13.1 (R1)	eth0	RIP	1
0.0.0.0/0	10.0.23.1 (R2)	eth1	Statique	—

Tableau 20. – Table de routage du routeur R3.

9.2.1 Sur un réseau local

En prenant le réseau présenté ci-dessus comme base, que se passe-t-il lorsque le PC `PC-A1` veut envoyer un paquet à `PC-A2` ? Le paquet est destiné à un périphérique sur le même réseau local (LAN A), donc il n'y a pas besoin de routage. Le PC `PC-A1` le sait grâce à son masque de sous-réseau : il peut voir que l'adresse IP de destination est sur le même réseau que lui. Il peut donc envoyer le paquet directement à `PC-A2` en utilisant son adresse MAC, sans passer par un routeur. S'il n'a pas l'adresse MAC de `PC-A2`, il peut utiliser le protocole ARP pour la découvrir.

Ce qui est important à retenir c'est que le paquet reste tel quel tout au long de son chemin, il n'est pas modifié par les périphériques intermédiaires. Le paquet est encapsulé dans une trame Ethernet pour être transmis sur le réseau local, **le header de la trame reste inchangé** tout au long de son chemin.

En résumant étape par étape le processus d'envoi de paquets du PC `PC-A1` vers le PC `PC-A2`, cela se déroule comme suit :

- A1 détermine que l'adresse IP de destination est sur le même réseau local grâce à son masque de sous-réseau.
- A1 vérifie dans sa table ARP s'il connaît l'adresse MAC de A2. Si ce n'est pas le cas, il envoie une requête ARP pour découvrir l'adresse MAC de A2.
- A2 répond à la requête ARP en fournissant son adresse MAC à A1.
- A1 encapsule le paquet dans une trame Ethernet II en utilisant l'adresse MAC de A2 comme adresse de destination et son adresse MAC comme adresse source. L'adresse de destination IP du paquet est l'adresse IP de A2, et l'adresse source IP est l'adresse IP de A1.
- La trame est transmise sur le réseau local, est aiguillée par le switch (grâce à sa table de commutation) et arrive finalement à A2.
- A2 reçoit la trame, extrait le paquet et le traite. Le paquet est intact, avec les mêmes adresses IP source et destination qu'à l'origine.

9.2.2 Sur un réseau distant

Si à présent `PC-A1` veut envoyer un paquet à `PC-B1`, le processus est différent car les deux périphériques sont sur des réseaux différents. Le paquet doit passer par les routeurs pour atteindre sa destination. Le long du chemin, **la trame Ethernet est modifiée à chaque saut** (passage par un routeur), mais le paquet reste intact, avec les mêmes adresses IP source et destination qu'à l'origine.[7]

Le processus d'envoi de paquets du PC `PC-A1` vers le PC `PC-B1` se déroule comme suit :

- A1 détermine que l'adresse IP de destination est sur un réseau différent grâce à son masque de sous-réseau.
- A1 vérifie dans sa table ARP s'il connaît l'adresse MAC de la passerelle par défaut (R1). Si ce n'est pas le cas, il envoie une requête ARP pour découvrir l'adresse MAC de R1.
- R1 répond à la requête ARP en fournissant son adresse MAC à A1.
- A1 encapsule le paquet dans une trame Ethernet II en utilisant l'adresse MAC de R1 comme adresse de destination et son adresse MAC comme adresse source. L'adresse de destination IP du paquet est l'adresse IP de B1, et l'adresse source IP est l'adresse IP de A1.

[7] En toute rigueur, chaque routeur modifie légèrement l'en-tête IP : il décrémente de 1 le champ TTL (*Time To Live*) et recalcule le checksum de l'en-tête. Les adresses IP source et destination, elles, ne changent pas. Le TTL évite qu'un paquet circule indéfiniment en cas de boucle de routage : arrivé à 0, le paquet est détruit.

- La trame est transmise sur le réseau local, est aiguillée par le switch (grâce à sa table de commutation) et arrive finalement à R1.
- R1 reçoit la trame, extrait le paquet et consulte sa table de routage pour déterminer le meilleur chemin vers le réseau de destination (LAN B). Il trouve que le meilleur chemin est via l'interface eth0, qui est connectée à R2.
- R1 encapsule le paquet dans une nouvelle trame Ethernet II en utilisant l'adresse MAC de R2 comme adresse de destination et son adresse MAC comme adresse source. L'adresse de destination IP du paquet reste l'adresse IP de B1, et l'adresse source IP reste l'adresse IP de A1.
- La trame est transmise sur le réseau entre R1 et R2, et arrive finalement à R2.
- R2 reçoit la trame, extrait le paquet et consulte sa table de routage pour déterminer le meilleur chemin vers le réseau de destination (LAN B). Il trouve que le réseau de destination est directement connecté à son interface eth2.
- R2 encapsule le paquet dans une nouvelle trame Ethernet II en utilisant l'adresse MAC de B1 comme adresse de destination et son adresse MAC comme adresse source. L'adresse de destination IP du paquet reste l'adresse IP de B1, et l'adresse source IP reste l'adresse IP de A1.
- La trame est transmise sur le réseau local du LAN B et arrive finalement à B1.
- B1 reçoit la trame, extrait le paquet et le traite. Le paquet est intact, avec les mêmes adresses IP source et destination qu'à l'origine. En revanche, le header de la trame Ethernet II a été modifié plusieurs fois par les routeurs intermédiaires.

9.2.3 Vers l'extérieur

Si un PC dans l'un des LAN veut envoyer un paquet à un hôte distant, comme `1.2.3.4`, le processus est similaire à celui décrit précédemment (Chapitre 9.2.2). La différence principale est que les routeurs R1, R2 et R3 ne connaissent pas le chemin exact vers l'hôte distant. Ils utilisent donc la route par défaut pour acheminer le paquet vers le FAI (Fournisseur d'Accès Internet), qui se charge ensuite de le transmettre vers sa destination finale.

Comme exemple, si `PC-C1` veut envoyer un paquet à `1.2.3.4`, le processus se déroule comme suit :

- `PC-C1` encapsule le paquet dans une trame Ethernet II en utilisant l'adresse MAC de R3 comme adresse de destination et son adresse

MAC comme adresse source. L'adresse de destination IP du paquet est `1.2.3.4`, et l'adresse source IP est l'adresse IP de `PC-C1`.

- La trame est transmise sur le réseau local.
- R3 reçoit la trame, extrait le paquet et consulte sa table de routage. Comme il ne connaît pas le chemin exact vers `1.2.3.4`, il utilise la route par défaut pour acheminer le paquet vers R2.
- R2 a le même comportement : il utilise la route par défaut pour acheminer le paquet vers R1.
- À partir de là, le routeur du FAI prend le relais et achemine le paquet vers sa destination finale, en utilisant les informations de routage disponibles sur Internet.

EXERCICE 9.2 · CHEMINEMENT D'UN PAQUET

Le PC `PC-A1` (192.168.1.10) envoie un paquet à destination du PC `PC-C1` (192.168.3.10). En se basant sur le schéma réseau illustré dans la Fig. 50 et sur les tables de routage des routeurs, remplir le tableau ci-dessous décrivant le contenu de la trame Ethernet II sur chaque segment du chemin. Les adresses MAC sont notées `MAC(X)`, par exemple `MAC(R1-eth1)` pour l'adresse MAC de l'interface `eth1` du routeur `R1`.

Segment	MAC source	MAC destination	IP source	IP destination
1. LAN A (`PC-A1` → `R1`)				
2. Liaison `R1` → `R3`				
3. LAN C (`R3` → `PC-C1`)				

EXERCICE 9.3 · REMPLIR LES TABLES DE ROUTAGE

La Fig. 51 présente un réseau composé de trois LAN (X, Y et Z) interconnectés en chaîne par trois routeurs : R1 <—> R2 <—> R3. Le routeur R3 est connecté au FAI. Les hypothèses suivantes s'appliquent :

- Aucun protocole de routage dynamique n'est actif : le routage est entièrement manuel, toutes les routes sont configurées par l'administrateur.
- Chaque routeur doit pouvoir atteindre tous les réseaux internes du schéma (les trois LAN et les deux liaisons `/30`).
- Chaque routeur dispose d'une route par défaut en direction du FAI.

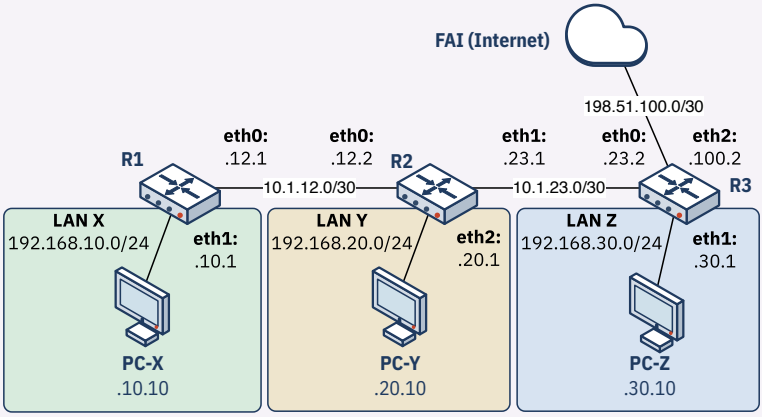


Fig. 51. – Réseau utilisé pour l'exercice de remplissage des tables de routage.

Sur une fiche à part, écrire la table de routage complète de chaque routeur, en reprenant les colonnes des tables de routage du chapitre : Destination, Gateway, Interface, et Protocole. La colonne Métrique est optionnelle, mais peut être remplie si vous le souhaitez (par convention elle vaut généralement 0 pour les routes directement connectées et `-` pour les routes statiques). Les adresses IP des interfaces des routeurs sont indiquées sur le schéma.

Indice : le nombre de lignes dans les tables de routage est respectivement de 6, 6 et 7 pour les routeurs R1, R2 et R3.

CHAPITRE 10

TCP et UDP

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir les protocoles TCP et UDP, ainsi que leur rôle dans le modèle en couches
- expliquer les différences entre les protocoles TCP et UDP, leurs cas d'utilisation et leurs avantages et inconvénients respectifs
- expliquer la notion de « well-known ports » et leur rôle dans l'identification des services réseau
- décrire le concept d'architecture client-serveur
- décrire le fonctionnement global du protocole TCP, y compris l'établissement de la connexion (three-way handshake) et la terminaison de la connexion (four-way handshake)

Ce chapitre aborde les protocoles TCP et UDP, et plus largement la couche transport des modèles OSI et TCP/IP.

10.1 Couche transport

Les trois premières couches du modèle OSI permettent d'envoyer des paquets d'un réseau à un autre, d'un hôte à un autre. A présent, il est nécessaire de distribuer les données à la bonne application sur l'hôte de destination. C'est le rôle de la couche transport, qui est la quatrième couche du modèle OSI et la troisième couche du modèle TCP/IP.

La couche transport fournit un « tunnel » de communication entre deux applications. Pour cela, elle s'appuie sur deux protocoles principaux : TCP (Transmission Control Protocol) et UDP (User Datagram Protocol). Ces protocoles permettent de gérer la communication entre les applications, avec un focus sur la fiabilité, le contrôle de flux et la gestion des erreurs pour TCP, et sur la rapidité et la simplicité pour UDP.

10.2 Ports

Une application sait qu'un paquet lui est destiné grâce à l'utilisation de ports. Un port est un identifiant numérique associé à une application ou un service spécifique sur un hôte. Cette méthode permet de multiplexer les communications sur un même hôte, en distinguant les différents flux de données selon le port utilisé. On peut voir les ports comme des boites aux lettres, où chaque application a sa propre boîte aux lettres (port) pour recevoir les messages (paquets).

Un port est représenté par un nombre entier sur 16 bits, ce qui permet d'avoir 65536 ports possibles (de 0 à 65535). Le couple adresse IP + port permet d'identifier de manière unique une application sur un hôte (généralement noté IP:port). Par exemple, `127.0.0.1:80` fait référence à une application écoutant sur le port 80 de l'adresse IP locale.

> **REMARQUE**
> Grâce au système de port, on peut avoir plusieurs applications de même types qui s'exécutent sur une même machine, mais pas sur le même port.

Les 1024 premiers ports (de 0 à 1023) sont appelés « well-known ports » et sont réservés aux services et applications les plus courants. Par exemple, le port 80 est utilisé pour le protocole HTTP, le port 443 pour HTTPS, le port 22 pour SSH, etc. Ci-dessous, une liste non exhaustive des well-known ports et de leurs services associés :

- Port 20 : FTP (File Transfer Protocol): transfert de fichiers
- Port 21 : FTP (File Transfer Protocol): contrôle de la connexion
- Port 22 : SSH (Secure Shell): accès sécurisé à distance
- Port 23 : Telnet: accès à distance non sécurisé
- Port 80 : HTTP (Hypertext Transfer Protocol): navigation web
- Port 443 : HTTPS (Hypertext Transfer Protocol Secure): navigation web sécurisée

Les ports de 1024 (2^{10}) à 49151 ($2^{15} + 2^{14} - 1$) sont appelés « registered ports » et peuvent être utilisés par des applications spécifiques, ainsi qu'être enregistrés auprès de l'IANA (Internet Assigned Numbers Authority). Il est possible d'en consulter la liste complète sur le site officiel de l'IANA[8].

[8] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

l'IANA réserve souvent les protocoles TCP et UDP pour un même port par précaution, mais dans la pratique, c'est souvent un seul protocole qui est réellement utilisé.

Les ports de 49152 ($2^{15} + 2^{14}$) à 65535 ($2^{16} - 1$) sont appelés « dynamic » ou « private ports » et sont généralement utilisés pour des communications temporaires ou pour des applications personnalisées.

Dans les faits, rien n'empêche d'héberger un serveur SSH sur le port 5235 par exemple. Cependant, il est recommandé de respecter les conventions de ports pour faciliter la configuration et l'interopérabilité des applications. En hébergeant un service sur un port non standard, on peut aussi rendre l'accès légèrement plus difficile pour un attaquant, mais cela ne constitue pas une véritable mesure de sécurité.

> **ATTENTION**
> Un port est lié à son protocole de transport. Le port 53 TCP n'est pas le même que le port 53 UDP. Il est donc important de préciser le protocole de transport utilisé pour chaque port. C'est possible par exemple d'avoir une application X sur le port 1000 TCP et une application Y sur le port 1000 UDP, même si elles n'ont rien à voir l'une avec l'autre (dans les faits, on évite ce genre de pratique pour éviter les confusions).

EXERCICE 10.1

À l'aide du registre des ports de l'IANA, indiquez pour chaque service le(s) port(s) et le protocole de transport. Attention : l'IANA réserve souvent TCP **et** UDP par précaution, alors qu'un seul est utilisé en pratique. Indiquez le transport **réellement** employé, et signalez les cas où le registre diffère de l'usage.

- IMAP, POP3, SMTP, DNS, NTP, MQTT

10.3 Architecture client-serveur

La plupart des services d'internet suivent le modèle d'architecture client-serveur. Dans ce modèle, un serveur est une application qui écoute les demandes des clients sur un port spécifique et répond à ces demandes. Un client est une application qui initie la communication avec le serveur en envoyant des requêtes. Un exemple typique que tout le monde utilise au quotidien est la navigation web : le navigateur web (client) envoie des

requêtes HTTP ou HTTPS au serveur web, qui répond avec les pages demandées.

Dans ce modèle, c'est le client qui initie la communication. Le système d'exploitation du client lui attribue généralement un **port éphémère** (un numéro choisi automatiquement dans une plage haute, typiquement entre 49152 et 65535). Il est possible que le client s'auto-attribue un port spécifique dans certains cas, mais c'est plus rare. Le serveur, quant à lui, écoute sur un port bien connu (well-known port) pour recevoir les requêtes des clients. La Fig. 52 illustre ce modèle.

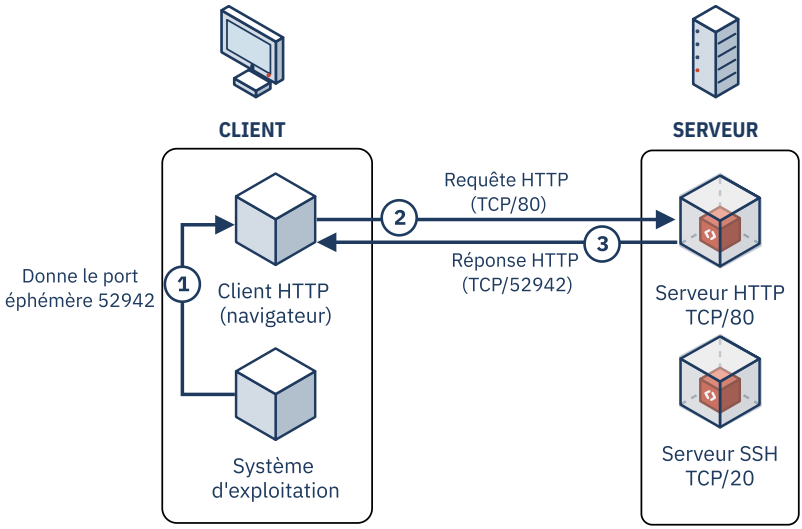


Fig. 52. – Architecture client-serveur. (1) le client reçoit un port éphémère de son système d'exploitation. (2) le client envoie une requête au serveur sur le port bien connu du service. (3) le serveur répond au client sur le port éphémère du client. Dans cet exemple, le serveur possède aussi une application qui écoute sur le port 22 (SSH), mais le client n'y accède pas, car il atteint le serveur sur le port 80 (HTTP).

Naturellement, l'architecture client-serveur n'est pas la seule possible. Il existe d'autres modèles, comme le modèle pair-à-pair (peer-to-peer), où chaque noeud peut agir à la fois comme client et serveur. Cependant, le modèle client-serveur reste le plus répandu pour les services internet.

10.4 UDP

UDP (User Datagram Protocol) est un protocole de communication de la couche transport qui permet l'envoi de messages appelés **datagrammes** entre applications. UDP est orienté vitesse et simplicité, mais ne garantit pas la fiabilité de la transmission. C'est le modèle du « fire-and-forget » : les datagrammes sont envoyés sans établir de connexion préalable, et il n'y a pas de mécanisme de retransmission en cas de perte de paquets. Cela le rend idéal pour des applications où la rapidité est cruciale et où la perte de quelques paquets n'est pas critique, comme le streaming vidéo ou audio.

UDP supporte les modes de transmission unicast, multicast et broadcast, ce qui le rend flexible pour différents types de communication. Il est dit « connectionless » car il n'établit pas de connexion entre l'émetteur et le récepteur avant d'envoyer les données. Le récepteur ne remet pas les datagrammes dans l'ordre, et ne signale pas les pertes de paquets.

L'en-tête d'un datagramme UDP est simple et efficace, avec seulement quatre champs principaux :

- le port source (16 bits) : identifie le port de l'application émettrice
- le port de destination (16 bits) : identifie le port de l'application destinataire
- la longueur (16 bits) : indique la taille totale du datagramme UDP, en incluant l'en-tête et les données
- le checksum (16 bits) : utilisé pour vérifier l'intégrité des données du datagramme.

La Fig. 53 illustre la structure d'un datagramme UDP.

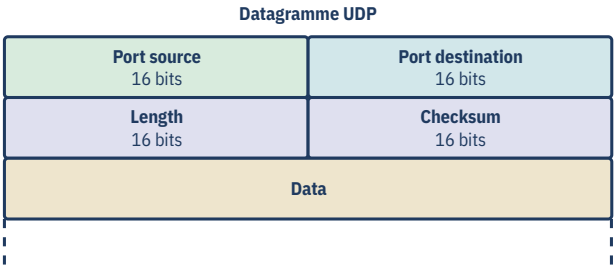


Fig. 53. – Format d'un datagramme UDP. Le champ « Length » indique la taille totale du datagramme, y compris l'en-tête. Le champ « Checksum » est utilisé pour vérifier l'intégrité des données. Les ports source et destination sont utilisés pour identifier les applications émettrices et destinataires.

10.5 TCP

TCP (Transmission Control Protocol) est un protocole de communication de la couche transport qui fournit une transmission fiable des données entre applications. Contrairement à UDP, TCP établit une connexion avant de transmettre les données et assure que les paquets sont reçus dans le bon ordre et sans perte (« connection oriented »). Il utilise des mécanismes de contrôle de flux, de retransmission et d'accusé de réception pour garantir la fiabilité. C'est le protocole idéal pour des applications où l'intégrité des données est cruciale, comme le transfert de fichiers, les emails ou la navigation web. Si les phrases d'un email sont mélangées ou perdues c'est problématique, alors que si quelques images d'une vidéo sont perdues, ce n'est pas grave.

TCP transmet des **segments** de données. Son mode de transmission est l'unicast, car par nature il nécessite de maintenir une connexion entre deux hôtes. Il est possible d'utiliser TCP pour des communications multicast ou broadcast, mais cela nécessite des mécanismes supplémentaires et n'est pas la norme.

TCP permet de gérer les éléments suivants :

- **Transfert des données ordonné** : les segments sont numérotés et réassemblés dans le bon ordre à la réception.
- **Retransmission des segments perdus** : si un segment n'est pas accusé de réception, il est retransmis.
- **Contrôle de flux** : TCP ajuste la vitesse d'envoi des segments en fonction de la capacité du récepteur à traiter les données, évitant ainsi la surcharge.
- **Contrôle de congestion** : TCP ajuste la vitesse d'envoi des segments en fonction de l'état du réseau, réduisant ainsi le risque de congestion.
- **Détection des erreurs** : chaque segment contient un checksum pour vérifier l'intégrité des données. Si une erreur est détectée, le segment est retransmis.

EXERCICE 10.2 · CONTRÔLE DE FLUX OU CONTRÔLE DE CONGESTION ?

Pour chacune des affirmations suivantes, indiquez si elle décrit le **contrôle de flux** ou le **contrôle de congestion**.

A. Le mécanisme s'appuie sur le champ « Window » de l'en-tête TCP, renseigné par le récepteur.

B. Le mécanisme réagit à la perte de segments ou à l'augmentation des délais de transmission, symptômes d'un encombrement du réseau.

C. Le mécanisme protège le récepteur d'une surcharge, par exemple si l'application réceptrice ne consomme pas les données assez vite.

D. Le mécanisme protège les équipements intermédiaires du réseau (routeurs, liens) d'une surcharge, indépendamment de la capacité du récepteur.

E. Un récepteur dont le buffer de réception est plein annoncera une fenêtre réduite, voire nulle, pour ralentir l'émetteur.

Le format de l'en-tête d'un segment TCP est plus complexe que celui d'UDP, avec plusieurs champs supplémentaires pour gérer la fiabilité et le contrôle de flux. Les champs sont les suivants :

- le port source (16 bits) : identifie le port de l'application émettrice
- le port de destination (16 bits) : identifie le port de l'application destinataire
- le numéro de séquence (32 bits) : utilisé pour ordonner les segments et détecter les pertes de données.
- le numéro d'acquittement (32 bits) : indique le prochain numéro de séquence attendu par l'émetteur, permettant ainsi de confirmer la réception des segments précédents.
- la longueur de l'en-tête (4 bits) : indique la taille de l'en-tête TCP, en nombre de mots de 32 bits. Cela permet de déterminer où commencent les données dans le segment.
- les flags (8 bits) : différents indicateurs de contrôle, tels que SYN (synchronisation), ACK (accusé de réception), FIN (fin de transmission), RST (réinitialisation de la connexion), etc. Ces flags sont utilisés pour gérer l'établissement, la maintenance et la terminaison des connexions TCP.
- la fenêtre « Window » (16 bits) : utilisée pour le contrôle de flux, elle indique la quantité de données que l'émetteur est prêt à recevoir. Cela permet d'éviter la surcharge du récepteur.
- le checksum (16 bits) : utilisé pour vérifier l'intégrité des données du segment TCP.
- l'option « Urgent Pointer » (16 bits) : utilisé pour indiquer que certaines données dans le segment sont urgentes et doivent être traitées en priorité.

- éventuellement des options supplémentaires (variable) : TCP permet d'ajouter des options pour des fonctionnalités avancées. Ces options ne sont pas obligatoire. Si elles sont présentes, elles sont placées après le champ « Urgent Pointer » et avant les données. La longueur de l'en-tête est ajustée en conséquence pour inclure ces options.

La Fig. 54 illustre la structure de l'en-tête d'un segment TCP.

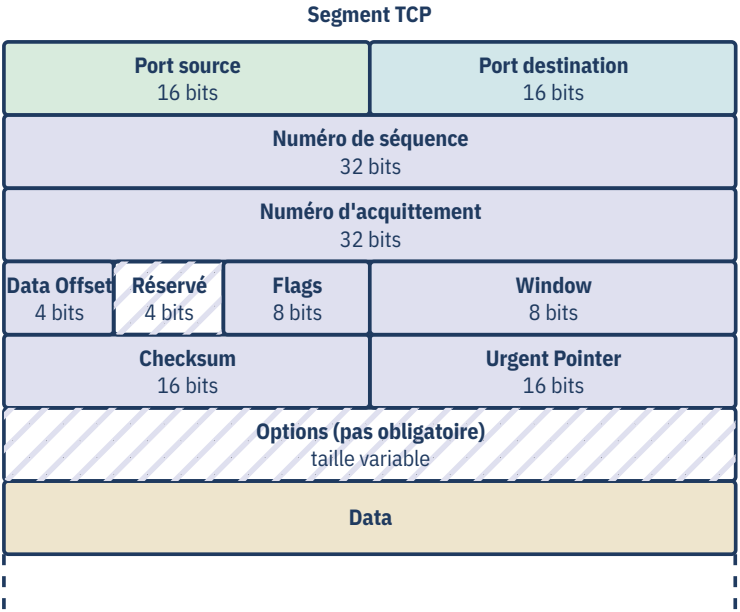


Fig. 54. – En-tête d'un segment TCP.

10.5.1 Établissement de la connexion TCP (three-way handshake)

L'établissement d'une connexion TCP se fait en trois étapes. Ce processus est appelé le « three-way handshake » et permet de synchroniser les numéros de séquence entre le client et le serveur, ainsi que d'établir les paramètres de la connexion. Les flags SYN et ACK sont utilisés pour gérer l'établissement de la connexion. Le flag SYN est utilisé pour synchroniser les numéros de séquence entre le client et le serveur, tandis que le flag ACK est

utilisé pour confirmer la réception des segments précédents. Tel qu'illustré dans la Fig. 55, le processus se déroule comme suit :

1. Le client envoie un segment avec le flag SYN activé pour initier la connexion.
2. Le serveur répond avec un segment ayant les flags SYN et ACK activés pour confirmer la réception du SYN et indiquer qu'il est prêt à établir la connexion.
3. Le client envoie un segment avec le flag ACK activé pour confirmer la réception du SYN-ACK, et la connexion est alors établie.

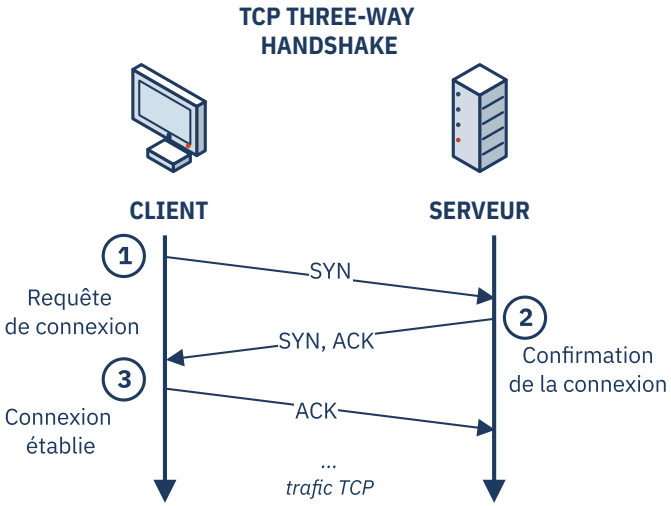


Fig. 55. – Établissement de la connexion TCP (three-way handshake). Le client envoie un segment SYN pour initier la connexion (1). Le serveur répond avec un segment SYN-ACK pour confirmer la réception du SYN et indiquer qu'il est prêt à établir la connexion (2). Enfin, le client envoie un segment ACK pour confirmer la réception du SYN-ACK, et la connexion est établie (3).

A partir de là, la connexion est établie et permet aux deux parties d'envoyer des données de manière fiable. On peut voir la connexion comme deux «tunnels» de communication, un dans chaque sens, chacun avec son propre numéro de séquence et ses mécanismes de contrôle de flux et de

retransmission. Les deux tunnels peuvent communiquer simultanément, permettant une transmission en full-duplex.

10.5.2 Terminaison de la connexion TCP (four-way handshake)

Comme pour l'établissement de la connexion, un processus précis est nécessaire pour terminer une connexion TCP. Ce processus est appelé le « four-way handshake » et permet de fermer la connexion de manière ordonnée, en s'assurant que toutes les données ont été transmises et reçues avant de libérer les ressources associées à la connexion.

Les flags FIN et ACK sont utilisés pour gérer la terminaison de la connexion. Le flag FIN est utilisé pour indiquer que l'émetteur a terminé d'envoyer des données et souhaite fermer la connexion, tandis que le flag ACK est utilisé pour confirmer la réception du segment FIN.

Le four-way handshake se déroule comme illustré dans la Fig. 56 :

1. Le client envoie un segment avec le flag FIN activé pour indiquer qu'il a terminé d'envoyer des données et souhaite fermer la connexion.
2. Le serveur répond avec un segment ayant le flag ACK activé pour confirmer la réception du FIN.
3. Le serveur envoie ensuite un segment avec le flag FIN activé pour indiquer qu'il a également terminé d'envoyer des données et souhaite fermer la connexion.
4. Enfin, le client répond avec un segment ayant le flag ACK activé pour confirmer la réception du FIN du serveur, et la connexion est alors terminée.

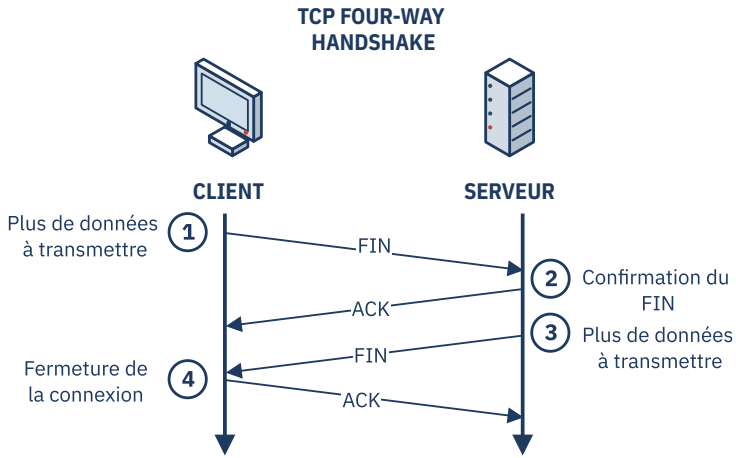


Fig. 56. – Terminaison de la connexion TCP (four-way handshake). Le client envoie un segment FIN pour indiquer qu'il a terminé d'envoyer des données et souhaite fermer la connexion (1). Le serveur répond avec un segment ACK pour confirmer la réception du FIN (2). Le serveur envoie ensuite un segment FIN pour indiquer qu'il a également terminé d'envoyer des données et souhaite fermer la connexion (3). Enfin, le client répond avec un segment ACK pour confirmer la réception du FIN du serveur, et la connexion est terminée (4).

ANECDOTE · Différences entre TCP et UDP avec des chats

Pour les curieux·ses, une vidéo expliquant les différences entre TCP et UDP… avec des chats ![9]

Scanner le QR code ci-dessus pour accéder à la vidéo.

[9] https://youtube.com/shorts/Jsl4Ks4_GQI

EXERCICE 10.3

Pour les cas d'utilisation suivants, indiquez si TCP ou UDP est le protocole de transport le plus approprié, et justifiez votre choix.

A. Streaming vidéo en direct (ex. Twitch)

B. Transfert d'une facture d'un client à un serveur

C. Synchronisation d'une base de données entre deux serveurs

D. Synchronisation de l'emplacement de personnages dans un jeu vidéo multijoueur en ligne

E. Envoi d'un message instantané entre deux utilisateurs sur une application de messagerie

CHAPITRE 11

Linux

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- utiliser les fonctionnalités de base de Linux, y compris la navigation dans le système de fichiers, la gestion des fichiers et des répertoires, et l'utilisation des commandes de base
- comprendre les concepts de base de la gestion des utilisateurs et des permissions dans Linux
- expliquer l'arborescence des répertoires et la structure du système de fichiers Linux

Ce chapitre aborde le système d'exploitation Linux, son organisation, ses commandes de base et la gestion des utilisateurs et des permissions. Le fonctionnement de Linux n'est pas vu en profondeur, car cela dépasse le cadre du cours. L'objectif est de fournir une base solide pour l'utilisation de Linux dans un contexte d'apprentissage de la téléinformatique.

11.1 Introduction à Linux

Linux est un système d'exploitation libre et open source basé sur le noyau Linux. Il a été créé en 1991 par Linus Torvalds et est devenu incontournable sur les serveurs, dans le cloud et sur les appareils embarqués (dont les smartphones, via Android). Sa part sur les ordinateurs de bureau reste en revanche marginale : environ 3 à 4% du marché mondial en 2026, loin derrière Windows et macOS [12]. Lorsqu'il combine le noyau Linux avec les outils et bibliothèques du projet GNU, on parle de système « GNU/Linux ».

La mascotte de Linux est un manchot nommé Tux (Fig. 57). Tux est voulu attachant et sympathique [13], ce qui donne un contraste avec l'image sérieuse et technique que l'on peut avoir d'un système d'exploitation.

Aujourd'hui, Linus Torvalds continue de superviser le développement du noyau Linux, qui est maintenu par une communauté mondiale de dévelop-

peurs. En janvier 2026, le projet a d'ailleurs formalisé pour la première fois un plan de succession, définissant comment la direction du noyau serait assurée si Torvalds venait à se retirer [14].

Fig. 57. – Tux, la mascotte de Linux

Linux s'utilise essentiellement via une interface en ligne de commande (CLI), mais il existe également des environnements de bureau graphiques (GUI) tels que GNOME, KDE et XFCE.

11.1.1 Historique de Linux

Avant Linux, le système d'exploitation Unix a été développé dans les années 1970 et a influencé de nombreux systèmes d'exploitation modernes. En 1983, Richard Stallman a lancé le projet GNU pour créer un système d'exploitation libre et open source compatible avec Unix. En 1991, Linus Torvalds a annoncé le développement du noyau Linux, qui a été publié sous la licence publique générale GNU (GPL) en 1992. La première version stable du noyau Linux, la version 1.0, a été publiée en 1994. L'intention de Linux était de créer un système « Unix-like » libre et open source, combinant le noyau Linux avec les outils et bibliothèques du projet GNU.

Aujourd'hui, on peut aussi mentionner Android, qui est un exemple de système d'exploitation basé sur le noyau Linux (sans GNU), utilisé sur les smartphones et les tablettes. MacOS, le système d'exploitation d'Apple, est basé sur le noyau XNU, qui est dérivé de BSD Unix (entre autres). Pour cette raison, certains concepts et commandes de Linux sont également applicables à MacOS. Windows est quant à lui basé sur le noyau NT, mais il existe des outils comme le Windows Subsystem for Linux (WSL) qui permettent d'exécuter des applications Linux sur Windows.

1969

Unix

Ken Thompson et Dennis Ritchie développent le système d'exploitation Unix à Bell Labs, qui servira de base à de nombreux systèmes d'exploitation modernes.

1973 — **Réécriture en C**
Unix est réécrit en langage C, le rendant portable d'une machine à l'autre — une innovation majeure qui explique sa diffusion rapide dans les universités.

1977 — **BSD**
L'université de Berkeley publie sa propre variante d'Unix, la Berkeley Software Distribution (BSD), qui donnera plus tard naissance à FreeBSD et servira de base à macOS.

1983 — **Projet GNU**
Richard Stallman lance le projet GNU (« GNU's Not Unix ») pour créer un système d'exploitation libre et compatible Unix. Il en produira tous les outils (GCC, Bash, coreutils…), mais pas le noyau.

1991 — **Noyau Linux**
Linus Torvalds, étudiant finlandais, annonce et publie son noyau de type Unix. Combiné aux outils GNU existants, il forme un système d'exploitation libre et complet.

1992 — **Passage sous licence GPL**
Linux est relicencié sous la licence publique générale GNU (GPL v2), scellant son alliance avec l'écosystème GNU et permettant sa diffusion libre.

1994 — **Linux 1.0**
Publication de la première version stable du noyau Linux (1.0), marquant sa maturité pour un usage sérieux.

2008 — **Android**
Google publie Android, bâti sur le noyau Linux mais avec son propre espace utilisateur (« Bionic ») : du « Linux sans GNU ».

La Fig. 58 illustre l'évolution des systèmes d'exploitation depuis Unix jusqu'à Linux, MacOS et Android, en mettant en évidence les relations entre eux. Le schéma est naturellement simplifié et ne reflète pas toutes les subti-

lités de l'histoire des systèmes d'exploitation, mais il permet de comprendre les grandes lignes de l'évolution et des influences entre ces systèmes.

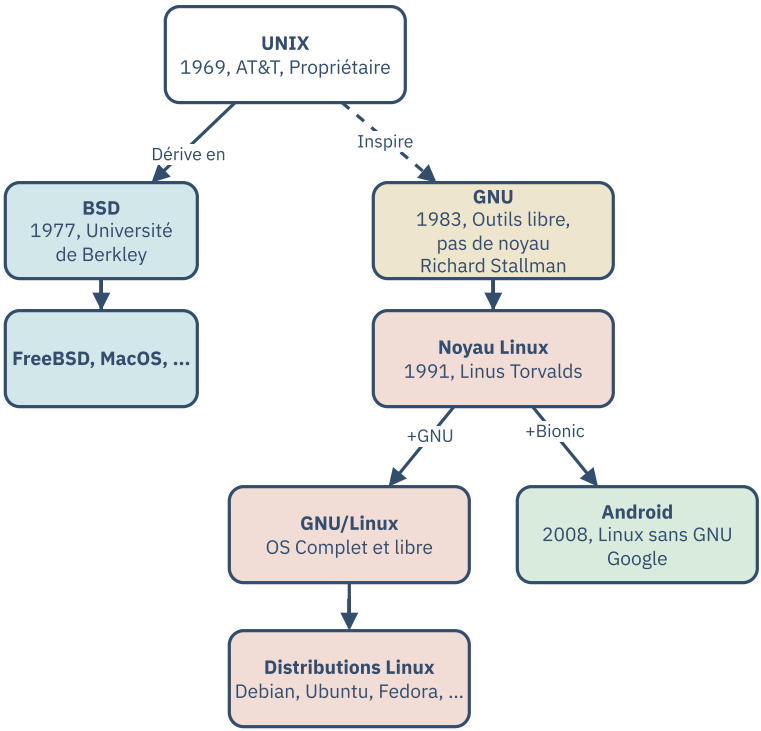


Fig. 58. – Évolution des systèmes d'exploitation depuis Unix jusqu'à Linux, MacOS et Android. Les flèches indiquent les influences et les relations entre les différents systèmes.

ANECDOTE

Richard Stallman et Linus Torvalds, bien que tous les deux soient des figures emblématiques du logiciel libre, ont un désaccord philosophique sur la manière dont le logiciel libre devrait être développé et distribué. Stallman est le fondateur du projet GNU et de la Free Software Foundation (FSF), et il est un ardent défenseur de la **liberté logicielle**. Pour

lui, le logiciel libre doit être un droit fondamental, au nom de l'éthique et de la liberté des utilisateurs.

Linus Torvalds, quant à lui, est le créateur du noyau Linux et a une approche plus pragmatique du développement logiciel. Il se concentre sur l'efficacité et la qualité du code. Lui défend l'idée du logiciel **open source**, car cela permet une collaboration plus large et une amélioration continue du logiciel, mais il n'insiste pas autant sur les aspects éthiques et philosophiques que Stallman [15], [16].

11.1.2 Distribution Linux

Linux s'organise en différentes distributions, qui sont des ensembles de logiciels regroupés autour du noyau Linux. Chaque distribution peut avoir ses propres objectifs, sa propre philosophie et ses propres outils de gestion des paquets (un paquet dans ce contexte est un ensemble de fichiers permettant d'installer un logiciel). Parmi les distributions les plus populaires, on peut citer Ubuntu, Fedora, Debian ou Arch Linux. Chaque distribution peut être adaptée à différents usages, allant des serveurs aux ordinateurs de bureau en passant par les systèmes embarqués.

REMARQUE

Pourquoi ne pas se contenter du noyau Linux et des outils GNU pour créer un système d'exploitation complet ? La réponse est que le noyau Linux et les outils GNU ne suffisent pas à eux seuls pour créer un système d'exploitation utilisable. Il faut également des bibliothèques, des applications, des gestionnaires de paquets, des environnements de bureau, etc. Les distributions Linux regroupent tous ces éléments et les configurent pour offrir une expérience utilisateur cohérente et fonctionnelle, clé en main.

Le Tableau 21 présente un aperçu de quelques distributions Linux populaires, leur philosophie, leurs usages typiques et leurs gestionnaires de paquets. Le tableau n'est pas exhaustif.

Distribution	Philosophie	Usage typique	Gestionnaire de paquets
Ubuntu	Facilité d'utilisation, communauté active	Ordinateurs de bureau, serveurs, cloud	APT (Advanced Package Tool)
Debian	Stabilité, logiciel libre	Serveurs, systèmes embarqués	APT (Advanced Package Tool)
Fedora	Innovation, logiciels récents	Ordinateurs de bureau, serveurs	DNF (Dandified Yum)
Arch Linux	Simplicité, contrôle total	Ordinateurs de bureau avancés, utilisateurs expérimentés	pacman

Tableau 21. – Aperçu de quelques distributions Linux populaires, leur philosophie, leurs usages typiques et leurs gestionnaires de paquets.

ATTENTION
Tout au long de ce livre, les commandes présentées et les informations données sont basées sur Ubuntu, qui est une distribution Linux populaire et largement utilisée. Cependant, la plupart des concepts et commandes sont applicables à d'autres distributions Linux, bien que certaines différences puissent exister en fonction de la distribution spécifique.

11.1.3 Avantages et inconvénients de Linux

Linux propose de nombreux avantages. En voici une liste non exhaustive :

- **Gratuit et open source** : Linux est librement disponible et peut être modifié et redistribué par quiconque, ce qui favorise l'innovation et la collaboration.
- **Sécurité** : Linux est réputé pour sa sécurité, grâce à sa conception multi-utilisateur et à sa gestion stricte des permissions. Les mises à jour de

sécurité sont fréquentes et les vulnérabilités sont rapidement corrigées par la communauté.
- **Stabilité et fiabilité** : Linux est souvent utilisé sur des serveurs et des systèmes critiques en raison de sa stabilité et de sa capacité à fonctionner pendant de longues périodes sans redémarrage.
- **Flexibilité** : Linux peut être utilisé sur une grande variété de matériels, des ordinateurs de bureau aux serveurs, en passant par les systèmes embarqués et les superordinateurs. Il peut être personnalisé pour répondre à des besoins spécifiques, grâce à sa modularité et à la disponibilité de nombreux logiciels libres.
- **Performances et légèreté** : Linux peut être optimisé pour fonctionner sur des systèmes avec des ressources limitées, ce qui le rend adapté aux anciens ordinateurs et aux appareils embarqués.

Il existe cependant quelques inconvénients à utiliser Linux :

- **Courbe d'apprentissage** : Pour les utilisateurs habitués à Windows ou macOS, Linux peut sembler complexe au début, en particulier pour la gestion des logiciels et des configurations système.
- **Compatibilité logicielle** : Certains logiciels propriétaires, notamment dans les domaines du graphisme, de la vidéo et des jeux, ne sont pas disponibles sur Linux.
- **Support technique** : Bien que la communauté Linux soit très active, le support officiel peut être limité pour certaines distributions, et les utilisateurs peuvent devoir se tourner vers des forums et des ressources en ligne pour résoudre leurs problèmes.

11.2 Principes de base de Linux

Dans Linux, la notion d'utilisateurs est essentielle. L'utilisateur « root » possède tous les droits et peut effectuer toutes les opérations sur le système. Les autres utilisateurs ont des droits limités, ce qui contribue à la sécurité du système. Chaque fichier et répertoire a un propriétaire et un groupe, et les permissions déterminent qui peut lire, écrire ou exécuter ces fichiers.

Un utilisateur non-root peut utiliser la commande `sudo` pour exécuter des commandes avec les privilèges de l'utilisateur root, à condition d'être autorisé à le faire. Cela permet de limiter l'utilisation des privilèges élevés et de réduire les risques de sécurité.

Il existe des interfaces graphiques pour avoir un environnement similaire à Windows ou macOS, mais la ligne de commande reste un outil puissant et incontournable pour l'administration et la configuration du système. Un

utilisateur Linux doit être capable d'utiliser les commandes de base pour naviguer dans le système de fichiers, gérer les fichiers et les répertoires, et effectuer des tâches d'administration (en tout cas pour les utilisateurs avancés et les administrateurs système).

Dans Linux, **tout est considéré comme un fichier**, y compris les périphériques et les informations système sur les processus par exemple. Cela permet d'unifier la gestion des ressources et de simplifier l'interaction avec le système. Les fichiers sont organisés dans une structure hiérarchique, avec un répertoire racine (`/`) à partir duquel tous les autres répertoires et fichiers sont accessibles.

11.3 Arborescence des répertoires

Le répertoire racine contient plusieurs sous-répertoires importants, qui sont listés dans le Tableau 22 (liste non-exhaustive). Chaque répertoire a un rôle spécifique et contient des fichiers et des sous-répertoires liés à ce rôle.

Répertoire	Description
/bin	Contient les programmes exécutables essentiels pour tous les utilisateurs (bin pour « binary »).
/boot	Contient les fichiers nécessaires au démarrage du système.
/dev	Contient les fichiers de périphériques représentant les périphériques matériels du système (dev pour « devices »).
/etc	Contient les fichiers de configuration du système et des applications (etc pour « et cetera », plus tard un « rétro-acronyme » désignera etc comme « Editing Text Config »).
/home	Contient les répertoires personnels des utilisateurs.
/lib	Contient les bibliothèques partagées essentielles pour le fonctionnement des programmes dans /bin et /sbin (lib pour « library »).
/media	Point de montage pour les supports amovibles comme les clés USB. Monter signifie « rendre accessible ».

Répertoire	Description
/mnt	Point de montage temporaire pour monter des systèmes de fichiers manuellement. Un système de fichiers est un ensemble de fichiers et de répertoires organisés sur un support de stockage.
/proc	Système de fichiers virtuel qui fournit des informations sur les processus en cours d'exécution et l'état du noyau.
/root	Répertoire personnel de l'utilisateur root (administrateur).
/sbin	Contient les programmes exécutables essentiels pour l'administration du système, généralement réservés à l'utilisateur root.
/tmp	Contient les fichiers temporaires créés par les applications et le système. Le contenu de ce répertoire peut être supprimé à tout moment, généralement au redémarrage du système.
/usr	Répertoire d'installation des logiciels (usr pour « user », et plus tard un « rétro-acronyme » désignera usr comme « Unix System Resources »).

Tableau 22. – Arborescence des répertoires Linux et description de leur rôle. La liste se concentre sur les répertoires les plus courants et essentiels pour la navigation et la gestion du système. Elle n'est pas exhaustive.

11.4 Commandes de base

Comme mentionné précédemment, Linux est principalement utilisé via une interface en ligne de commande (CLI). Les commandes de base permettent de naviguer dans le système de fichiers, de gérer les fichiers et les répertoires, et d'effectuer des tâches d'administration. Cette section présente quelques-unes des commandes les plus courantes et utiles pour les utilisateurs débutants et avancés.

> **ASTUCE**
> Pour connaître l'utilité et l'utilisation d'une commande à tout moment, il existe la commande `man` (pour « manual »). Par exemple, `man ls` affiche

> le manuel de la commande `ls`. Vous pouvez naviguer dans le manuel avec les flèches du clavier et quitter avec la touche `q`.
>
> La plupart des commandes Linux ont également une option `--help` qui affiche un résumé de leur utilisation. Par exemple, `ls --help` affiche les options disponibles pour la commande `ls`.
>
> Des « cheatsheets » (fiches de référence) sont également disponibles en ligne pour les commandes Linux les plus courantes. Elles sont accessibles facilement avec `curl cheat.sh/commande` (remplacez `commande` par la commande souhaitée). Par exemple, `curl cheat.sh/ls` affiche une fiche de référence pour la commande `ls`. Cette dernière méthode nécessite néanmoins une connexion Internet et la présence de l'outil `curl` sur le système.

11.4.1 Connexion et gestion des utilisateurs

Pour accéder à la console d'un système Linux, il y a deux moyens principaux : soit en se connectant directement sur la machine (physiquement ou via un terminal série), soit en utilisant un protocole de connexion à distance comme SSH (Secure Shell). SSH permet de se connecter à un système Linux depuis un autre ordinateur, en utilisant une connexion sécurisée. Dans tous les cas, il est nécessaire de disposer d'un compte utilisateur sur le système pour pouvoir se connecter. On précise le nom d'utilisateur, et le système demande ensuite le mot de passe associé à ce compte.

Une fois dans la ligne de commande, on voit le nom de l'utilisateur courant, le nom de la machine et le répertoire courant dans l'invite de commande. Par exemple, une invite de commande peut ressembler à ceci :

```
user@machine:/some/directory$
```

Par convention, le symbole `$` indique que l'utilisateur courant n'est pas root (administrateur), tandis que le symbole `#` indique que l'utilisateur courant est root. Cela permet de savoir rapidement si l'on dispose des privilèges d'administration ou non.

L'invite de commande Linux est gérée par le « shell ». Le shell est un programme qui interprète les commandes que l'utilisateur tape et les exécute. Il existe plusieurs shells disponibles sur Linux, mais le plus courant est Bash (Bourne Again SHell). D'autres shells populaires incluent Zsh, Fish

et Ksh. Pour savoir quel shell est utilisé, on peut exécuter la commande `echo $SHELL`. Par exemple :

```
$ echo $SHELL
```

```
/bin/bash
```

Ici, c'est bien le shell Bash qui est utilisé.

Une fois connecté sur un système, on peut regarder les informations sur l'utilisateur courant avec la commande `whoami`. Par exemple :

```
$ whoami
```

```
darthvader
```

Ici, l'utilisateur courant est « darthvader ». On peut également utiliser la commande `id` pour obtenir des informations plus détaillées sur l'utilisateur courant.

```
$ id
```

```
uid=1001(darthvader) gid=1001(darthvader)
groups=1001(darthvader),100(users)
```

La sortie donne plusieurs informations :

- `uid` : l'identifiant unique de l'utilisateur (User ID)
- `gid` : l'identifiant unique du groupe principal de l'utilisateur (Group ID)
- `groups` : la liste des groupes auxquels l'utilisateur appartient

Chaque utilisateur Linux fait partie d'un ou plusieurs groupes, qui permettent de gérer les permissions d'accès aux fichiers et aux ressources du système. Le groupe principal de l'utilisateur est généralement créé automatiquement lors de la création du compte utilisateur (homonyme du nom d'utilisateur). Les autres groupes peuvent être ajoutés pour accorder des permissions supplémentaires à l'utilisateur.

Il est possible de changer le mot de passe d'un utilisateur avec la commande `passwd`. Par exemple, pour changer le mot de passe de l'utilisateur courant, on peut exécuter :

```
$ passwd
```

Si on a les privilèges d'administration, on peut également changer le mot de passe d'un autre utilisateur en précisant son nom d'utilisateur :

```
$ sudo passwd nom_utilisateur
```

Pour se connecter à un utilisateur différent, on peut utiliser la commande `su` (pour « substitute user »). Par exemple, pour se connecter en tant qu'utilisateur « luke », on peut exécuter :

```
$ su luke
```

Pour ajouter ou supprimer un utilisateur, on peut utiliser les commandes `adduser` ou `useradd` pour ajouter un utilisateur, et `deluser` ou `userdel` pour supprimer un utilisateur. Par exemple, pour ajouter un utilisateur nommé « leia », on peut exécuter :

```
$ sudo adduser leia
```

Pour supprimer cet utilisateur, on peut exécuter :

```
$ sudo deluser leia
```

EXERCICE 11.1 · DIFFÉRENCES ENTRE ADDUSER/DELUSER ET USERADD/USERDEL

Pourquoi existe-il deux commandes pour ajouter et supprimer des utilisateurs (`adduser` / `deluser` et `useradd` / `userdel`) ? Quelles sont les différences entre ces deux paires de commandes ?

Trouvez l'information sur Internet ou avec la commande `man` pour comprendre les différences entre ces commandes.

11.4.2 Navigation dans le système de fichiers

Dans l'invite de commande, le répertoire courant est indiqué après le nom de l'utilisateur et de la machine. Certains systèmes n'affichent que le nom du dernier répertoire du chemin courant, mais on peut toujours obtenir le chemin complet avec la commande `pwd` (pour « print working directory »). Par exemple :

```
$ pwd
```

```
/home/darthvader
```

Les chemins sur Linux sont une succession de répertoires séparés par des barres obliques (`/`). Le répertoire racine est représenté par une seule barre oblique (`/`), et tous les autres répertoires sont situés sous ce répertoire racine. Par exemple, le chemin `/home/darthvader/Documents` indique que le répertoire `Documents` se trouve dans le répertoire `darthvader`, qui lui-même se trouve dans le répertoire `home`, qui est directement sous le répertoire racine.

Le répertoire `home` de l'utilisateur courant a pour alias `~` (tilde). Par exemple, le chemin `~/Documents` est équivalent à `/home/darthvader/Documents` si l'utilisateur courant est « darthvader ».

Pour changer de répertoire, on utilise la commande `cd` (pour « change directory »). Par exemple, pour aller dans le répertoire `Documents`, on peut exécuter :

```
$ cd Documents
```

Le chemin donné en argument à la commande (un argument est une information que l'on fournit à une commande pour qu'elle sache quoi faire) peut être relatif ou absolu. Un chemin relatif est relatif au répertoire courant, tandis qu'un chemin absolu commence toujours par le répertoire racine (`/`). Par exemple, si l'on est dans le répertoire `/home/darthvader`, le chemin relatif `Documents` fait référence à `/home/darthvader/Documents`, tandis que le chemin absolu `/home/darthvader/Documents` fait référence au même répertoire, peu importe le répertoire courant.

Pour lister les fichiers et répertoires dans le répertoire courant, on utilise la commande `ls` (pour « list »). Par exemple :

```
$ ls
Documents  Downloads  Pictures  Videos
```

Pour obtenir plus d'informations sur les fichiers et répertoires, on peut utiliser l'option `-l` (pour « long format ») avec la commande `ls` :

```
$ ls -l
```

```
total 16
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Documents
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Downloads
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Pictures
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Videos
```

Il est possible de lister les fichiers d'un répertoire spécifique en fournissant le chemin du répertoire en argument à la commande `ls`. De la même manière que pour `cd`, le chemin peut être relatif ou absolu. Par exemple, pour lister les fichiers du répertoire `/home/darthvader/Documents`, on peut exécuter :

```
$ ls /home/darthvader/Documents
```

11.4.3 Gestion des fichiers et des répertoires

Il existe deux types principaux de fichiers dans un système informatique : les fichiers textes et les fichiers binaires. Les fichiers textes contiennent des données lisibles par l'homme, tandis que les fichiers binaires contiennent des données qui ne sont pas directement lisibles par l'homme (par exemple, des images, des vidéos ou des programmes compilés). Sur Linux, on peut connaître le type d'un fichier avec la commande `file`. Par exemple, pour connaître le type du fichier `example.txt`, on peut exécuter :

```
$ file example.txt
```

```
example.txt: ASCII text
```

Dans cet exemple, on voit que le fichier `example.txt` est un fichier texte ASCII.

Pour lire le contenu de fichiers texte, on peut utiliser des commandes comme `cat`, `less`, ou `more`. Par exemple, pour afficher le contenu du fichier `example.txt`, on peut exécuter :

```
$ cat example.txt
```

```
This is an example text file.
```

Les commandes `less` et `more` permettent de lire le contenu d'un fichier page par page, ce qui est utile pour les fichiers longs.

Pour créer un nouveau fichier texte, on peut utiliser la commande `touch`. Par exemple, pour créer un fichier nommé `newfile.txt`, on peut exécuter :

```
$ touch newfile.txt
```

Une fois créé, on peut modifier le fichier avec un éditeur de texte en ligne de commande comme `nano`, `vi`, ou `vim`. Par exemple, pour éditer le fichier `newfile.txt` avec `nano`, on peut exécuter :

```
$ nano newfile.txt
```

Une interface graphique rudimentaire s'affiche alors, permettant d'écrire du texte. Pour sauvegarder et quitter `nano`, on utilise les raccourcis clavier `Ctrl + O` (pour « write Out ») puis `Ctrl + X` (pour « eXit »).

Pour supprimer un fichier, on peut utiliser la commande `rm`. Par exemple, pour supprimer le fichier `newfile.txt`, on peut exécuter :

```
$ rm newfile.txt
```

Au niveau des répertoires, on peut créer un nouveau répertoire avec la commande `mkdir`. Par exemple, pour créer un répertoire nommé `newdir`, on peut exécuter :

```
$ mkdir newdir
```

Pour supprimer un répertoire vide, on peut utiliser la commande `rmdir`. Par exemple, pour supprimer le répertoire `newdir`, on peut exécuter :

```
$ rmdir newdir
```

Si le répertoire contient des fichiers ou d'autres répertoires, on peut utiliser la commande `rm -r` pour supprimer le répertoire et tout son contenu. Par exemple, pour supprimer le répertoire `newdir` et tout ce qu'il contient, on peut exécuter :

```
$ rm -r newdir
```

ANECDOTE · Tout supprimer ?!

La commande `sudo rm -rf /` est célèbre pour être extrêmement dangereuse. Elle supprime de manière récursive (`-r`) et forcée (`-f`) tous les fichiers et répertoires à partir du répertoire racine (`/`), ce qui entraîne la destruction complète du système d'exploitation et de toutes les données. Cette commande est souvent utilisée comme une blague ou un avertissement pour souligner l'importance de faire attention avec les commandes en ligne de commande, surtout lorsqu'on utilise `sudo` , qui donne des privilèges d'administrateur.

Dans les faits, la plupart des distributions Linux modernes ont mis en place des protections pour empêcher l'exécution de cette commande par inadvertance. Par exemple sur Ubuntu :

```
darthvader@workstation:~$ sudo rm -rf /
```

Le système nous prévient que c'est dangereux et refuse d'exécuter la commande, affichant le message suivant :

```
rm: it is dangerous to operate recursively on '/'
rm: use --no-preserve-root to override this failsafe
```

Bien sûr, il est possible de forcer l'exécution de la commande avec l'option `--no-preserve-root` , mais à partir de là, vous l'aurez vraiment cherché ! :-)

11.4.4 Commandes réseau

Dans le cadre du cours, il est important de connaître quelques commandes réseau de base pour vérifier la connectivité et diagnostiquer les problèmes réseau. Voici quelques-unes des commandes les plus courantes.

Pour vérifier la connectivité réseau, on peut utiliser la commande `ping`. Par exemple, pour tester la connectivité avec le site web « example.com », on peut exécuter :

```
$ ping example.com
```

EXERCICE 11.2 · OPTIONS DE LA COMMANDE PING

Regarder sur la page `man` à quoi servent les options suivantes :

- `-c`
- `-i`
- `-4`
- `-6`

Pour afficher les informations de configuration réseau de l'ordinateur, on peut utiliser la commande `ip addr`. Par exemple :

```
$ ip addr
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state
UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
pfifo_fast state UP group default qlen 1000
    link/ether bc:24:11:7c:ab:77 brd ff:ff:ff:ff:ff:ff
    altname enp0s18
    altname enxbc24117cab77
    inet 160.98.22.140/24 brd 160.98.22.255 scope global ens18
       valid_lft forever preferred_lft forever
    inet6 fe80::be24:11ff:fe7c:ab77/64 scope link proto
kernel_ll
       valid_lft forever preferred_lft forever
```

EXERCICE 11.3 · ANALYSE DE LA SORTIE DE LA COMMANDE IP ADDR

Dans la sortie de la commande `ip addr` ci-dessus, quelles sont les adresses IPv4 et IPv6 de l'interface réseau `ens18` ? Quelle est l'adresse de diffusion (broadcast) pour cette interface ?

Que représente l'interface `lo` ? Son adresse IPv4 est-elle routable sur Internet ?

Un autre outil utile pour diagnostiquer les problèmes réseau est la commande `traceroute`, qui permet de suivre le chemin emprunté par les paquets pour atteindre une destination. Par exemple, pour tracer le chemin vers « example.com », on peut exécuter :

```
$ traceroute example.com
```

```
traceroute to example.com (104.20.23.154), 30 hops max, 60
byte packets
 1  vl-1510.ngf-vip.isc.heia-fr.ch (160.98.22.1)  2.836 ms
1.951 ms  2.709 ms
 2  160.98.6.105 (160.98.6.105)  2.917 ms  3.482 ms  2.805 ms
 3  te-0-3-0-99.rfron01.hefr.ch (160.98.7.221)  2.962 ms
2.115 ms  2.866 ms
 4  swifr1-10ge-0-0-0-22.switch.ch (195.176.1.20)  2.733 ms
2.885 ms  2.634 ms
 5  swils1-100ge-0-0-0-6.switch.ch (130.59.37.146)  4.441 ms
3.322 ms  4.342 ms
 6  swice1-100ge-0-0-0-1.switch.ch (130.59.37.152)  6.724 ms
5.354 ms  4.922 ms
 7  cixp-cloudflare.cern.ch (192.65.185.228)  3.332 ms  3.226
ms  4.435 ms
 8  * * 104.20.23.154 (104.20.23.154)  3.882 ms
```

EXERCICE 11.4 · ANALYSE DE LA SORTIE DE LA COMMANDE TRACEROUTE

Dans la sortie de la commande `traceroute` ci-dessus, combien de « sauts » (hops) sont nécessaires pour atteindre le site « example.com » ? Quels sont les adresses IP des routeurs intermédiaires rencontrés sur le chemin ? Que signifie l'astérisque (`*`) dans la dernière ligne de la sortie ?

Pour afficher les connexions réseau actives et les ports ouverts sur le système, on peut utiliser la commande `netstat`. Par exemple :

```
$ netstat -tuln
```

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.53:53           0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.54:53           0.0.0.0:*
LISTEN
tcp6       0      0 :::22                   :::*
LISTEN
udp        0      0 127.0.0.54:53
0.0.0.0:*
udp        0      0 127.0.0.53:53
0.0.0.0:*
udp        0      0 127.0.0.1:323
0.0.0.0:*
udp6       0      0 ::1:323                 :::*
```

Dans cet exemple, la commande `netstat -tuln` affiche les connexions réseau actives et les ports ouverts sur le système. L'option `-t` affiche les connexions TCP, l'option `-u` affiche les connexions UDP, l'option `-l` affiche uniquement les ports en écoute (listening), et l'option `-n` affiche les adresses et ports sous forme numérique. On a par exemple un serveur SSH écoutant sur le port 22 (TCP) et un serveur DNS écoutant sur le port 53 (UDP).

EXERCICE 11.5 · ANALYSE DE LA SORTIE DE LA COMMANDE NETSTAT

Dans la sortie de la commande `netstat` ci-dessus, pourquoi les ports TCP sont notés avec le statut « LISTEN » tandis que les ports UDP n'ont pas ce statut ? Pourquoi les services écoutent tous sur `0.0.0.0:*` ou `:::*` ? Que signifie cette notation ?

Sur quelle adresse IPv4 et quel port devrait écouter un serveur web HTTP standard pour accepter les connexions entrantes depuis l'hôte lui-même uniquement ?

Dans les commandes plus classiques, la commande `ip neigh` permet de visualiser la table ARP (Address Resolution Protocol) du système, qui associe les adresses IP aux adresses MAC des périphériques sur le réseau local. Par exemple :

```
$ ip neigh
```

```
160.98.22.2 dev ens18 lladdr 00:50:56:92:ca:59 STALE
160.98.22.149 dev ens18 lladdr bc:24:11:18:48:12 STALE
160.98.22.1 dev ens18 lladdr 00:50:56:92:ca:59 DELAY
160.98.22.141 dev ens18 lladdr bc:24:11:36:4d:f2 STALE
```

11.4.5 Installation de logiciels (Ubuntu/Debian)

Parfois, il est nécessaire d'installer des logiciels supplémentaires sur un système Linux. Sur les distributions basées sur Debian (comme Ubuntu), on utilise le gestionnaire de paquets APT (Advanced Package Tool) pour installer, mettre à jour et supprimer des logiciels.

Avant toute opération d'installation, il est recommandé de mettre à jour la liste des paquets disponibles avec la commande `sudo apt update`. Cela permet de s'assurer que l'on dispose des informations les plus récentes sur les logiciels disponibles dans les dépôts.

```
$ sudo apt update
```

Pour installer un logiciel, on utilise la commande `sudo apt install nom_du_paquet`. Par exemple, pour installer `traceroute`, on peut exécuter :

```
$ sudo apt install traceroute
```

Pour lister les logiciels disponibles dans les dépôts, on peut utiliser la commande `apt search nom_du_paquet`. Par exemple, pour rechercher des paquets liés à `python`, on peut exécuter :

```
$ apt search python
```

A noter que la commande `apt search` ne nécessite pas de privilèges d'administration, contrairement à `apt update` et `apt install`, qui nécessitent l'utilisation de `sudo`.

Enfin, pour désinstaller un logiciel, on utilise la commande `sudo apt remove nom_du_paquet`. Par exemple, pour désinstaller `traceroute`, on peut exécuter :

```
$ sudo apt remove traceroute
```

ANECDOTE · Paquets sympas à essayer

Si vous avez un petit peu de temps et que vous voulez vous entraîner à utiliser la ligne de commande, vous pouvez installer quelques paquets amusants (mais certainement pas utiles !) :

- `nyancat` : un programme qui affiche une animation du célèbre chat Nyan Cat dans le terminal.
- `sl` : un petit programme qui affiche une animation d'un train à vapeur lorsqu'on tape `sl` au lieu de `ls`. C'est une blague pour les utilisateurs qui font souvent des fautes de frappe.
- `cowsay` : un programme qui affiche un message dans une bulle de dialogue avec une vache ASCII. Par exemple, `cowsay "Hello, world!"` affichera une vache disant « Hello, world! ».
- `fortune` : un programme qui affiche une citation ou un proverbe aléatoire. Par exemple, `fortune` affichera une citation aléatoire à chaque exécution.

Ci-dessous un exemple de sortie de `cowsay` :

```
 ______________
< Vive Linux ! >
 --------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

11.4.6 Autres commandes utiles

Cette section présente quelques commandes supplémentaires qui peuvent être utiles pour les utilisateurs Linux, en vrac, sans ordre particulier. Il est possible que certaines commandes ne soient pas installées par défaut sur toutes les distributions, mais elles peuvent généralement être installées via le gestionnaire de paquets.

- `htop` : permet d'afficher l'utilisation des ressources système (CPU, mémoire, processus) de manière interactive et en temps réel. C'est une version améliorée de la commande `top`.
- `tree` : permet d'afficher l'arborescence des répertoires et fichiers sous forme d'un arbre. Par exemple, `tree /home/darthvader` affichera l'arborescence du répertoire personnel de l'utilisateur « darthvader ».
- `du` : permet d'afficher l'utilisation de l'espace disque par les fichiers et répertoires. Par exemple, `du -h /home/darthvader` affichera l'utilisation de l'espace disque du répertoire personnel de l'utilisateur « darthvader » en format lisible pour un humain (avec des unités comme K, M, G).
- `help` : affiche la liste des commandes intégrées de base au shell courant, ainsi que des informations sur leur utilisation. Par exemple, `help cd` affichera des informations sur la commande `cd`.

11.5 Gestion des droits sous Linux

La gestion des droits sous Linux est un aspect fondamental de la sécurité et de l'administration du système. Pour rappel, tout est considéré comme un fichier sous Linux, donc savoir gérer les droits d'accès aux fichiers est la brique fondamentale pour sécuriser un système Linux.

Chaque fichier et répertoire possède un propriétaire, un groupe et des permissions qui déterminent qui peut lire, écrire ou exécuter le fichier.

Lorsqu'on liste les fichiers d'un répertoire avec la commande `ls -l`, on obtient une sortie qui inclut les informations sur les permissions, le propriétaire et le groupe de chaque fichier. Par exemple :

```
ls -l
```

```
total 24
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Documents
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Downloads
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Pictures
drwxrwxr-x 2 darthvader darthvader 4096 Jul 15 13:02 Videos
-rw-rw-r-- 1 darthvader darthvader   16 Jul 15 13:09
example.txt
-rwxrwxr-x 1 darthvader darthvader    6 Jul 15 13:44 script.sh
```

Une ligne est construite de la manière suivante :

- Le premier bloc de caractères indique le type de fichier et les permissions.

  - Le premier caractère indique le type de fichier : `-` pour un fichier ordinaire, `d` pour un répertoire, `l` pour un lien symbolique, etc.
  - Les neuf caractères suivants sont regroupés en trois blocs de trois caractères, représentant les permissions pour le propriétaire, le groupe et les autres utilisateurs, respectivement.
- Le deuxième bloc indique le nombre de liens physiques vers le fichier.
- Le troisième bloc indique le propriétaire du fichier.
- Le quatrième bloc indique le groupe du fichier.
- Le cinquième bloc indique la taille du fichier en octets.
- Ensuite il y a la date, l'heure et le nom du fichier.

Reprenons le bloc de permissions pour le fichier `example.txt` :

```
-rw-rw-r-- 1 darthvader darthvader   16 Jul 15 13:09
example.txt
```

Que signifient ces permissions ? Les permissions sont divisées en trois groupes de trois caractères :

- Le premier groupe (`rw-`) correspond aux permissions du propriétaire (ici `darthvader`). - Le second groupe (`rw-`) correspond aux permissions du groupe (ici `darthvader`).
- Le troisième groupe (`r--`) correspond aux permissions des autres utilisateurs (qui ne sont ni le propriétaire ni dans le groupe).

Chaque caractère représente une permission spécifique :

- `r` : permission de lecture (read)
- `w` : permission d'écriture (write)
- `x` : permission d'exécution (execute)
- `-` : absence de permission

La permission de lecture est toujours la première lettre du bloc, la permission d'écriture est la deuxième lettre, et la permission d'exécution est la troisième lettre. Ainsi, pour le fichier `example.txt`, le propriétaire et le groupe ont les permissions de lecture et d'écriture, tandis que les autres utilisateurs n'ont que la permission de lecture.

11.5.1 Modification des permissions

Pour modifier les permissions d'un fichier ou d'un répertoire, on utilise la commande `chmod` (pour « change mode »). Il existe deux façons de spécifier les permissions : en utilisant la notation symbolique ou la notation octale.

La notation symbolique utilise des lettres pour représenter les utilisateurs et les permissions. Par exemple, pour ajouter la permission d'exécution pour le propriétaire du fichier `example.txt`, on peut exécuter :

```
$ chmod u+x example.txt
```

Ici, `u` représente le propriétaire (user), `+` signifie ajouter une permission, et `x` représente la permission d'exécution. Après cette commande, les permissions du fichier `example.txt` seront :

```
-rwxrw-r-- 1 darthvader darthvader   16 Jul 15 13:09
example.txt
```

Les opérations possibles avec la notation symbolique sont :
- `+` : ajouter une permission
- `-` : retirer une permission
- `=` : définir les permissions exactement comme spécifié (remplace les permissions existantes)

Comme préfixe à l'opération, on peut utiliser :
- `u` : pour le propriétaire (user)
- `g` : pour le groupe (group)
- `o` : pour les autres utilisateurs (others)
- `a` ou rien du tout : pour tous les utilisateurs (user, group et others)

Par exemple, pour retirer la permission d'écriture pour le groupe, on peut exécuter :

```
$ chmod g-w example.txt
```

Pour modifier les droits avec la notation octale (la plus courante), on utilise un nombre à trois chiffres, où chaque chiffre représente les permissions pour le propriétaire, le groupe et les autres utilisateurs, respectivement. Chaque chiffre est la somme des valeurs des permissions :
- Lecture (r) = 4
- Écriture (w) = 2
- Exécution (x) = 1

La Fig. 59 montre les différents blocs de permissions et leurs valeurs correspondantes.

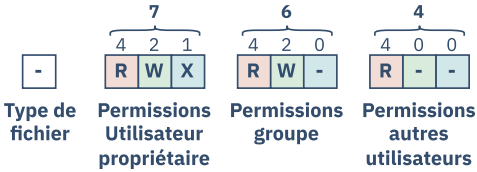


Fig. 59. – Représentation des blocs de permissions et de leurs valeurs correspondantes pour la notation octale. Si une permission est accordée, la valeur correspondante est ajoutée pour former le chiffre final. Par exemple, pour accorder les permissions de lecture et d'écriture (4 + 2) au propriétaire, la valeur finale sera 6.

Ainsi, pour définir les permissions du fichier `example.txt` à `rw-r-----`, on peut exécuter :

```
$ chmod 640 example.txt
```

Dans cet exemple, le chiffre `6` (4 + 2) correspond aux permissions du propriétaire (lecture et écriture), le chiffre `4` correspond aux permissions du groupe (lecture uniquement), et le chiffre `0` correspond aux permissions des autres utilisateurs (aucune permission).

APPROFONDISSEMENT · Protection des fichiers dans un répertoire partagé avec le sticky bit

Comment faire lorsqu'un répertoire est partagé en écriture pour protéger les fichiers qu'il contient ? On peut utiliser le « sticky bit » pour empêcher les utilisateurs de supprimer ou de renommer les fichiers d'autres utilisateurs dans un répertoire partagé. Pour activer le sticky bit sur un répertoire, on peut exécuter :

```
$ chmod +t nom_du_repertoire
```

Le sticky bit est représenté par un `t` à la fin des permissions du répertoire. Par exemple, si un répertoire a les permissions `drwxrwxrwt`, cela signifie que tous les utilisateurs peuvent écrire dans le répertoire, mais seuls les propriétaires des fichiers peuvent les supprimer ou les renommer. On peut aussi l'ajouter sous forme octale avec la valeur

`1` au début du chiffre à trois chiffres. Par exemple, pour définir les permissions d'un répertoire à `rwxrwxrwt`, on peut exécuter :

```
$ chmod 1777 nom_du_repertoire
```

11.5.2 Modification du propriétaire et du groupe

Pour modifier le propriétaire et le groupe d'un fichier ou d'un répertoire, on utilise la commande `chown` (pour « change owner »). Par exemple, pour changer le propriétaire du fichier `example.txt` à l'utilisateur `luke`, on peut exécuter :

```
$ sudo chown luke example.txt
```

Pour changer le groupe du fichier `example.txt` à `jedi`, on peut exécuter :

```
$ sudo chown :jedi example.txt
```

Pour changer à la fois le propriétaire et le groupe, on peut exécuter :

```
$ sudo chown luke:jedi example.txt
```

CHAPITRE 12

DHCP et DNS

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir le protocole DHCP et son rôle dans l'attribution dynamique d'adresses IP aux périphériques sur un réseau
- expliquer le fonctionnement du protocole DHCP, les messages échangés entre le client et le serveur DHCP, et le processus d'attribution d'une adresse IP
- décrire la notion de bail DHCP et son importance dans la gestion des adresses IP
- définir le protocole DNS et son rôle dans la résolution de noms de domaine en adresses IP
- expliquer le déroulement d'une requête DNS, y compris les étapes de résolution récursive et itérative
- décrire les notions de FQDN, autorité de zone, serveur principal et serveur secondaire
- expliquer les différents types d'enregistrements DNS (A, AAAA, CNAME, MX, NS, TXT) et leur rôle dans la résolution de noms de domaine

Ce chapitre aborde les protocoles DHCP et DNS qui sont essentiels pour la gestion des adresses IP et la résolution de noms de domaine sur un réseau. Le protocole DHCP permet d'attribuer automatiquement des adresses IP aux périphériques, tandis que le protocole DNS traduit les noms de domaine en adresses IP, facilitant ainsi la navigation sur Internet et l'accès aux ressources réseau.

12.1 DHCP

DHCP, pour « Dynamic Host Configuration Protocol », est un protocole réseau qui permet d'attribuer automatiquement des adresses IP et d'autres

paramètres de configuration réseau aux périphériques sur un réseau. Cela simplifie la gestion des adresses IP et évite les conflits d'adresses.

Prenons un cas d'école (littéralement !) : lorsque vous vous êtes connecté au Wi-Fi de l'école la première fois, vous n'avez rien dû configurer manuellement, n'est-ce pas ? C'est grâce au protocole DHCP que votre périphérique a reçu automatiquement une adresse IP et les informations nécessaires pour communiquer sur le réseau, et que l'utilisation d'Internet a été « plug-and-play ».

DHCP est un protocole dit « applicatif » c'est à dire un protocole de L7 (couche application) du modèle OSI. Il fonctionne sur le protocole UDP (couche transport) et utilise les ports 67 et 68 pour la communication entre le client et le serveur DHCP.

Il est capable de fournir aux périphériques une configuration réseau essentielle comprenant :
- L'adresse IP dans le réseau local
- Le masque de sous-réseau
- La passerelle par défaut (gateway)
- Les serveurs DNS (Domain Name System) pour la résolution de noms de domaine

12.1.1 Fonctionnement

DHCP fonctionne selon le modèle client-serveur. Lorsqu'un périphérique (client DHCP) se connecte à un réseau, il envoie en **broadcast** un message `DHCP DISCOVER` pour rechercher un serveur DHCP disponible. Le serveur DHCP répond avec un message `DHCP OFFER` contenant une adresse IP disponible et d'autres paramètres de configuration. Le client envoie ensuite un message `DHCP REQUEST` pour accepter l'offre, et le serveur confirme avec un message `DHCP ACK`, finalisant ainsi le processus d'attribution d'adresse IP. Ce processus est connu sous le nom de « DORA » (Discover, Offer, Request, Acknowledge) et est représenté dans la Fig. 60. Le serveur DHCP est configuré avec une plage d'adresses IP disponibles (pool) et peut attribuer des adresses IP dynamiques aux clients pour une durée limitée, appelée « bail » (lease). Le client doit renouveler son bail avant son expiration pour conserver la même adresse IP.

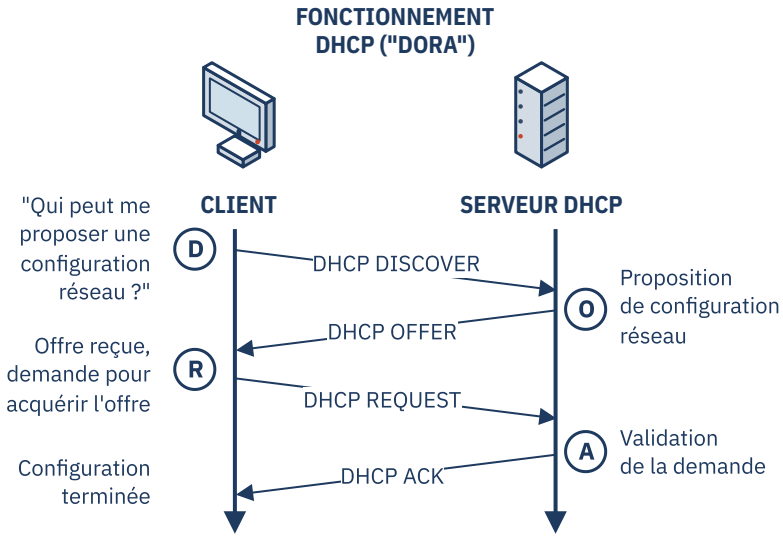


Fig. 60. – Processus DORA (Discover, Offer, Request, Acknowledge) pour l'attribution d'une configuration IP via DHCP.

Pour l'envoi du `DHCP DISCOVER`, le client utilise les headers suivants, couche par couche :

- L2: Mac source du client, `ff:ff:ff:ff:ff:ff` comme Mac destination (broadcast)
- L3: IP source `0.0.0.0` (il n'a pas encore d'adresse IP), IP destination `255.255.255.255` (adresse de « limited broadcast »).
- L4: UDP source port 68, UDP destination port 67 (serveur DHCP)

L'adresse IP de broadcast limité n'est jamais routée, elle reste toujours sur le lien local (L2). En prenant pour exemple le schéma réseau présenté dans la Fig. 61, on peut se demander lequel des deux serveurs DHCP répondra à une requête `DHCP DISCOVER` envoyée par le périphérique client.

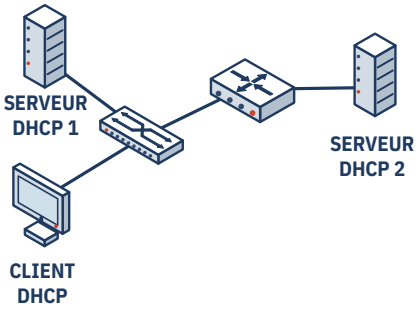


Fig. 61. – Schéma réseau pour l'exercice sur le DHCP. Un serveur DHCP est connecté au même switch que le périphérique client, tandis qu'un autre serveur DHCP est connecté de l'autre côté du routeur. Le client envoie un message DHCP DISCOVER pour obtenir une adresse IP.

Le serveur DHCP qui répondra à la requête du client est celui qui est connecté au même switch que le périphérique client. En effet, le message `DHCP DISCOVER` est envoyé en broadcast limité (IP destination `255.255.255.255`), qui ne traverse pas le routeur. Il faut que le serveur DHCP soit dans le **même domaine de diffusion** que le client pour pouvoir recevoir et répondre à la requête. Pour rappel, le routeur coupe les domaines de diffusion, donc le serveur DHCP de l'autre côté du routeur ne recevra pas le message `DHCP DISCOVER` et ne pourra pas répondre.

Il existe cependant le mécanisme de **DHCP relay** qui permet à un routeur de relayer les messages DHCP entre différents domaines de diffusion. En quelques mots, le routeur agit comme un intermédiaire entre le client et le serveur DHCP, en recevant les messages `DHCP DISCOVER` du client et en les retransmettant au serveur DHCP, puis en renvoyant les réponses du serveur DHCP au client. Ce mécanisme est utile dans les réseaux où le serveur DHCP n'est pas directement accessible par tous les clients, mais il nécessite une configuration spécifique sur le routeur pour fonctionner correctement.

12.1.2 Bail DHCP

Lorsqu'un serveur DHCP délivre une adresse IP à un périphérique, il ne s'agit pas d'une attribution permanente. Le serveur DHCP attribue un **bail** (lease) pour une durée déterminée, après laquelle l'adresse IP peut être réattribuée à un autre périphérique si le bail n'est pas renouvelé. Le client DHCP doit donc renouveler son bail avant son expiration pour conserver

la même adresse IP. Si le bail expire et que le client n'a pas renouvelé, il devra demander une nouvelle adresse IP au serveur DHCP. La durée du bail est communiquée dans le message `DHCP OFFER` et peut varier en fonction de la configuration du serveur DHCP. En général, les baux sont configurés pour durer plusieurs heures ou jours, mais ils peuvent être ajustés selon les besoins du réseau.

Pour renouveler un bail, le client envoie un message `DHCP REQUEST` au serveur DHCP avant l'expiration du bail. Le serveur répond avec un message `DHCP ACK` pour confirmer le renouvellement du bail et prolonger la durée de l'attribution de l'adresse IP.

Sans ce mécanisme de bail, une pénurie d'adresse IP pourrait vite survenir dans un réseau avec de nombreux périphériques qui se connectent et se déconnectent fréquemment. Le bail DHCP permet donc une gestion efficace des adresses IP disponibles, en s'assurant que les adresses inutilisées sont libérées et réattribuées à d'autres périphériques.

12.1.3 Types de messages DHCP

Les différents types de messages DHCP sont définis dans les RFC 2132, 3203, 4388, 6926 et 7724. Toutes ces RFCs représentent les évolutions du protocole DHCP depuis sa première version. Ci-dessous les types de messages DHCP les plus courants (ceux définis dans la RFC 2132) :

- `DHCP DISCOVER` (0x01) : message envoyé par le client pour découvrir les serveurs DHCP disponibles sur le réseau.
- `DHCP OFFER` (0x02) : message envoyé par le serveur DHCP en réponse au message `DHCP DISCOVER`, contenant une configuration IP proposée pour le client.
- `DHCP REQUEST` (0x03) : message envoyé par le client pour accepter l'offre du serveur DHCP et demander la configuration IP proposée. Egalement utilisé pour renouveler un bail existant.
- `DHCP DECLINE` (0x04) : message envoyé par le client pour refuser l'offre du serveur DHCP.
- `DHCP ACK` (0x05) : message envoyé par le serveur DHCP pour confirmer l'attribution de la configuration IP au client.
- `DHCP NAK` (0x06) : message envoyé par le serveur DHCP pour refuser la demande du client.
- `DHCP RELEASE` (0x07) : message envoyé par le client pour libérer l'adresse IP attribuée par le serveur DHCP.

- `DHCP INFORM` (0x08) : message envoyé par le client pour demander des informations supplémentaires au serveur DHCP, sans demander d'adresse IP.

EXERCICE 12.1

Selon-vous, qu'est-ce qui peut pousser un client à refuser une offre (`DHCP DECLINE`) ? Et qu'est-ce qui peut pousser un serveur à refuser une demande (`DHCP NAK`) ?

12.1.4 Que se passe-t-il si aucun serveur DHCP ne répond ?

Lorsqu'aucun serveur DHCP ne répond à la requête `DHCP DISCOVER` d'un client, le périphérique ne peut pas obtenir automatiquement une adresse IP et d'autres paramètres de configuration réseau. Dans ce cas, le périphérique peut adopter l'une des stratégies suivantes :

1. **Adresse IP statique** : L'utilisateur peut configurer manuellement une adresse IP statique sur le périphérique, en s'assurant que l'adresse choisie est unique et ne provoque pas de conflit avec d'autres périphériques sur le réseau. Cette méthode nécessite une connaissance préalable de la configuration réseau et peut être moins pratique pour les utilisateurs non techniques.
2. **Adresse IP APIPA (Automatic Private IP Addressing)** : Certains systèmes d'exploitation, comme Windows, utilisent le mécanisme APIPA pour attribuer automatiquement une adresse IP dans la plage réservée `169.254.0.1` à `169.254.255.254` lorsque aucun serveur DHCP n'est disponible. Cette méthode permet au périphérique de communiquer avec d'autres périphériques sur le même réseau local, mais ne fournit pas de connectivité Internet.

12.1.5 Que se passe-t-il si plusieurs serveurs DHCP répondent ?

Lorsque plusieurs serveurs DHCP répondent à la requête `DHCP DISCOVER` d'un client, le périphérique peut recevoir plusieurs offres (`DHCP OFFER`) de différents serveurs. Dans ce cas, le client doit choisir l'une des offres pour continuer le processus d'attribution d'adresse IP. Le choix peut être basé sur différents critères, tels que la priorité des serveurs DHCP, la proximité du serveur, ou d'autres facteurs définis par le client.

En général, le client sélectionne la première offre reçue et envoie un message `DHCP REQUEST` pour accepter cette offre. Les autres serveurs DHCP qui

ont envoyé des offres non acceptées peuvent alors libérer les adresses IP proposées et les rendre disponibles pour d'autres clients sur le réseau.

> **REMARQUE**
> En principe, on ne déploie pas plusieurs serveurs DHCP avec des plages d'adresses qui se chevauchent sur le même réseau local, car cela peut entraîner des conflits d'adresses IP : deux serveurs qui ne partagent pas leur base de baux risqueraient d'attribuer la même adresse à des clients différents. En revanche, disposer de plusieurs serveurs DHCP pour assurer la redondance est une pratique courante et recommandée, à condition de coordonner leur configuration, soit en découpant le pool en plages disjointes (split-scope), soit en activant un protocole de partage d'état (failover).

12.1.6 Avantages et inconvénients

DHCP présente plusieurs avantages et inconvénients par rapport à l'attribution d'adresses IP statiques. Voici un résumé des principaux points :

- **Comportement « plug-and-play »** : Les périphériques peuvent se connecter au réseau et obtenir automatiquement une configuration IP sans intervention manuelle.
- **Gestion centralisée** : Les administrateurs réseau peuvent gérer les adresses IP et les paramètres de configuration à partir d'un serveur DHCP unique, simplifiant ainsi la maintenance et la mise à jour des configurations.

En revanche, il existe également des inconvénients à l'utilisation de DHCP :

- **Moins de contrôle sur les adresses IP attribuées** : Les administrateurs réseau ont moins de contrôle sur les adresses IP attribuées aux périphériques, ce qui peut poser des problèmes pour certaines applications ou services nécessitant des adresses IP fixes.
- **Dépendance au serveur DHCP** : Si le serveur DHCP tombe en panne ou devient indisponible, les périphériques ne pourront pas obtenir de nouvelles adresses IP ou renouveler leurs baux, ce qui peut entraîner des problèmes de connectivité sur le réseau.
- **Sécurité** : Le DHCP offre des vecteurs d'attaque potentiels, tels que les attaques DHCP spoofing (un faux serveur DHCP qui attribue des adresses IP malveillantes) ou les attaques par DHCP starvation (un attaquant qui épuise le pool d'adresses IP disponibles en envoyant de nombreuses requêtes DHCP).

Dans les faits, les serveurs DHCP sont principalement utilisés dans les réseaux destinés aux utilisateurs finaux, tels que les réseaux domestiques, les réseaux d'entreprise et les réseaux publics. Dans ces environnements, la facilité d'utilisation et la gestion centralisée offertes par DHCP l'emportent généralement sur les inconvénients potentiels. Cependant, dans les réseaux où un contrôle strict des adresses IP est nécessaire, comme les serveurs ou les périphériques réseau critiques, l'attribution d'adresses IP statiques est souvent privilégiée pour garantir la stabilité et la sécurité du réseau.

12.2 DNS

DNS (pour « Domain Name System ») est un protocole réseau qui traduit les noms de domaine en adresses IP et vice versa. Il permet aux utilisateurs d'accéder aux ressources réseau en utilisant des noms de domaine conviviaux, plutôt que de mémoriser des adresses IP numériques. Le DNS est essentiel pour le fonctionnement d'Internet, car il facilite la navigation sur le Web et l'accès aux services en ligne.

Lorsque vous ouvrez un navigateur Web et que vous tapez une URL, comme `www.heia-fr.ch`, le DNS est responsable de la résolution de ce nom de domaine en une adresse IP, permettant à votre navigateur de se connecter au serveur Web correspondant et d'afficher le site.

Sans DNS, nous devrions nous souvenir de l'adresse IP de chaque site Web que nous voulons visiter, ce qui serait peu pratique et difficile à gérer. Le DNS agit comme un annuaire téléphonique pour Internet, en associant des noms de domaine lisibles par l'homme à des adresses IP numériques utilisées par les ordinateurs pour communiquer entre eux.

ANECDOTE

Avant le DNS, créé en 1983, toutes les correspondances entre noms de domaine et adresses IP étaient stockées dans un fichier texte appelé `hosts.txt`, maintenu par le NIC (Network Information Center) du Stanford Research Institute. Ce fichier était distribué aux utilisateurs d'Internet, mais il devenait rapidement obsolète et difficile à gérer à mesure que le nombre de sites Web et d'adresses IP augmentait. Le DNS a été développé pour résoudre ce problème en fournissant un système hiérarchique et distribué pour la résolution des noms de do-

maine. Ils ont consultables dans un repository Github : https://github.com/ttkzw/hosts.txt/tree/master.

12.2.1 Hiérarchie et FQDN

Comme introduit précédemment, le DNS est un service de correspondance entre une adresse IP (L3) et un nom de domaine (L7). Un nom de domaine est une chaîne de caractères hiérarchique, composée de plusieurs parties séparées par des points. Chaque partie du nom de domaine représente un niveau dans la hiérarchie DNS, allant du domaine racine (le point final) aux sous-domaines et aux noms d'hôtes spécifiques. Un nom de domaine complet est appelé FQDN (Fully Qualified Domain Name) et inclut tous les niveaux de la hiérarchie, du domaine racine jusqu'au nom d'hôte. Par exemple, le FQDN `www.example.com` se compose des éléments suivants :

- `www` : le nom d'hôte (host) ou sous-domaine
- `example` : le domaine de second niveau
- `com` : le domaine de premier niveau
- `.` : le domaine racine (root domain)

Tout FQDN est un nom de domaine, mais tous les noms de domaine ne sont pas des FQDN. Un nom de domaine peut être partiel, ne comprenant que certains niveaux de la hiérarchie, tandis qu'un FQDN inclut tous les niveaux jusqu'à la racine (il désigne le chemin complet vers un hôte spécifique).

> **REMARQUE**
> Vous vous dites peut-être « attends, mais il n'y a pas de point à la fin du FQDN ! Lorsque je vais sur `www.google.ch` , je ne tape pas `www.google.ch.` avec un point final. »
>
> La raison est très simple : **le point final, représentant la racine, est implicite**. Les serveurs DNS savent que la racine est toujours là, donc il n'est pas nécessaire de l'inclure dans les requêtes DNS. Pour l'expérience, il est possible d'aller sur `www.google.ch.` avec le point final, et vous verrez que cela fonctionne également.

La hiérarchie est organisée en 4 niveaux principaux : le domaine racine, les domaines de premier niveau (TLD), les domaines de second niveau et les sous-domaines. Le domaine racine est représenté par un point final (.) et est géré par l'ICANN (Internet Corporation for Assigned Names and Numbers). Les TLD sont les extensions de domaine, telles que `.com` , `.org` , `.net` , ou

les TLD géographiques comme `.ch` pour la Suisse. Les domaines de second niveau sont généralement choisis par les organisations ou les individus qui enregistrent un nom de domaine, tandis que les sous-domaines peuvent être créés pour organiser davantage les ressources au sein d'un domaine. La figure Fig. 62 illustre la hiérarchie du DNS et les relations entre les différents niveaux de noms de domaine.

Les TLD ne peuvent pas être créés librement par les utilisateurs : leur nombre est limité et leur création passe par l'ICANN. Ce n'est toutefois pas l'ICANN qui gère directement les TLD ; elle délègue cette responsabilité à des organisations appelées **registries** (registres). Un registre est responsable de la banque de données et des enregistrements DNS pour un TLD spécifique, et se coordonne avec les bureaux d'enregistrement (**registrars**) qui vendent les noms de domaine aux utilisateurs finaux. Par exemple, le TLD `.ch` est exploité par Switch en tant que registre, sur mandat de l'Office fédéral de la communication (OFCOM), l'autorité responsable des domaines suisses. Certains TLD imposent des conditions d'attribution : le `.swiss`, dont l'OFCOM est lui-même le registre, est réservé aux résidents et organisations suisses. Des entreprises comme Infomaniak font partie des registrars qui vendent des noms de domaine sous le TLD `.ch` aux utilisateurs finaux.

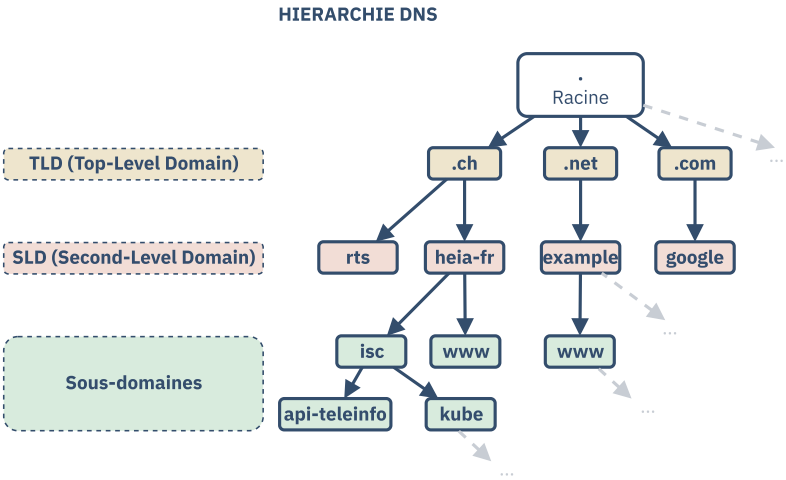


Fig. 62. – Hiérarchie du DNS, illustrant les relations entre le domaine racine, les TLD, les domaines de second niveau et les sous-domaines.

On parle de zone DNS pour désigner un ensemble de noms de domaine gérés par un ensemble de serveurs DNS autoritaires. Une zone peut contenir plusieurs enregistrements DNS, tels que des enregistrements A, AAAA, CNAME, MX, NS ou TXT, qui définissent les correspondances entre les noms de domaine et les adresses IP ou d'autres informations pertinentes. Les serveurs DNS autoritaires sont responsables de la gestion et de la mise à jour des enregistrements dans leur zone respective.

12.2.2 Résolution récursive et itérative

La hiérarchie des noms de domaine se retrouve également dans la manière dont les serveurs DNS sont organisés. Les serveurs DNS sont répartis en plusieurs niveaux, avec des serveurs racine au sommet de la hiérarchie, suivis de serveurs TLD, de serveurs de second niveau et de serveurs autoritaires pour les sous-domaines. Cette organisation permet une résolution efficace des noms de domaine, en permettant aux requêtes DNS de descendre la hiérarchie, de la racine jusqu'au serveur autoritaire pour le domaine spécifique, afin d'obtenir la réponse finale. Un serveur DNS est appelé serveur de nom (ou « nameserver »). Il est dit authoritaire pour une zone lorsqu'il détient les enregistrements DNS pour cette zone et peut fournir des réponses définitives aux requêtes concernant les noms de domaine dans cette zone.

Par exemple, lorsqu'un utilisateur tente d'accéder à `www.example.com`, la requête DNS est d'abord envoyée à un serveur racine, qui redirige la requête vers un serveur TLD pour le domaine `.com`. Le serveur TLD renvoie ensuite la requête à un serveur autoritaire pour le domaine `example.com`, qui fournit finalement l'adresse IP associée au nom d'hôte `www`. Pour résoudre un nom de domaine, il existe deux méthodes principales : la résolution récursive et la résolution itérative. La résolution récursive est généralement utilisée par les clients DNS, tandis que la résolution itérative est souvent utilisée par les serveurs DNS pour interroger d'autres serveurs DNS.

Une résolution récursive implique que le serveur DNS prend en charge l'ensemble du processus de résolution pour le client. La requête est déléguée au serveur DNS, qui continue à interroger d'autres serveurs DNS jusqu'à obtenir la réponse finale. La Fig. 63 illustre le processus de résolution récursive. Le serveur qui effectue la descente récursive est appelé (recursive) resolver. Le client est appelé **stub resolver** car il ne fait que transmettre la requête au resolver et attendre la réponse finale.

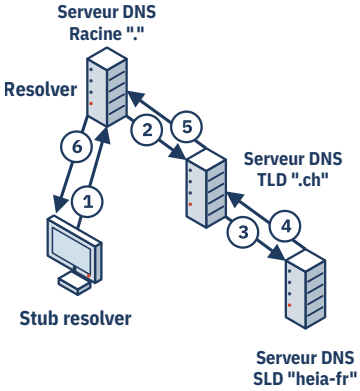


Fig. 63. – Processus de résolution récursive d'un nom de domaine par un serveur DNS.

Une résolution itérative, en revanche, implique que le serveur DNS réponde au client avec des informations partielles, en indiquant les serveurs DNS suivants à interroger pour poursuivre la résolution. Le client doit ensuite interroger ces serveurs DNS supplémentaires pour obtenir la réponse finale. La Fig. 64 illustre le processus de résolution itérative.

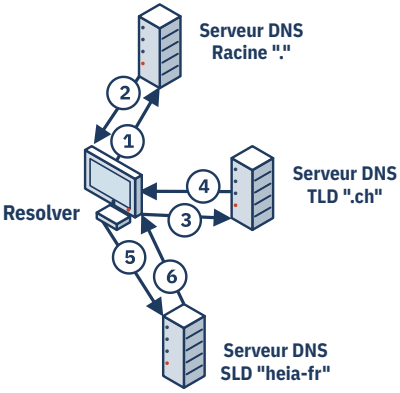


Fig. 64. – Processus de résolution itérative d'un nom de domaine par un serveur DNS.

Le type de résolution utilisé est déterminé par la configuration du serveur DNS et les préférences du client. En pratique, un mélange des deux méthodes est souvent utilisé, avec des serveurs DNS effectuant une résolution récursive pour les clients et fournissant des réponses itératives aux autres serveurs DNS.

12.2.3 Types d'enregistrements DNS

Il existe plusieurs types d'enregistrements DNS, chacun ayant un rôle spécifique dans la résolution des noms de domaine. Les enregistrements les plus courants sont :

- `A` (Address) : associe un nom de domaine à une adresse IPv4.
- `AAAA` (IPv6 Address) : associe un nom de domaine à une adresse IPv6.
- `CNAME` (Canonical Name) : crée un alias pour un autre nom de domaine, permettant de rediriger les requêtes vers le nom canonique.
- `MX` (Mail Exchange) : spécifie les serveurs de messagerie responsables de la réception des e-mails pour un domaine.
- `NS` (Name Server) : indique les serveurs DNS autoritaires pour un domaine, permettant de déléguer la gestion des enregistrements DNS à d'autres serveurs. Un serveur de TLD par exemple sera rempli d'enregistrements NS pointant vers les serveurs autoritaires des domaines de second niveau.
- `TXT` (Text) : permet de stocker des informations textuelles associées à un nom de domaine, souvent utilisées pour la vérification de domaine et la configuration de services.
- `SOA` (Start of Authority) : contient l'adresse du serveur DNS principal pour la zone, ainsi que des informations sur la zone, telles que le numéro de série et les paramètres de mise à jour.
- `PTR` (Pointer) : utilisé pour la résolution inverse, associant une adresse IP à un nom de domaine.

12.2.4 Types de serveurs DNS

Il existe deux types de serveurs DNS : les serveurs principaux et les serveurs secondaires. Les serveurs principaux (ou « primary ») sont responsables de la gestion et de la mise à jour des enregistrements DNS pour une zone spécifique. Ils détiennent la copie principale de la base de données DNS pour cette zone et sont autorisés à apporter des modifications aux enregistrements. Les serveurs secondaires (ou « secondary ») sont des copies de sauvegarde des serveurs principaux et sont utilisés pour assurer la redondance et la disponibilité des informations DNS. Ils reçoivent les mises à jour des enregistrements DNS à partir des serveurs principaux et peuvent

répondre aux requêtes DNS aux côté du serveur principal. Cela a pour avantage d'alléger la charge sur le serveur principal et d'améliorer la résilience du système DNS en cas de panne ou de surcharge du serveur principal.

Le processus de synchronisation entre les serveurs principaux et secondaires est appelé «zone transfer» (transfert de zone). Il permet aux serveurs secondaires de récupérer les enregistrements DNS mis à jour à partir des serveurs principaux, garantissant ainsi que toutes les copies des enregistrements DNS sont cohérentes et à jour.

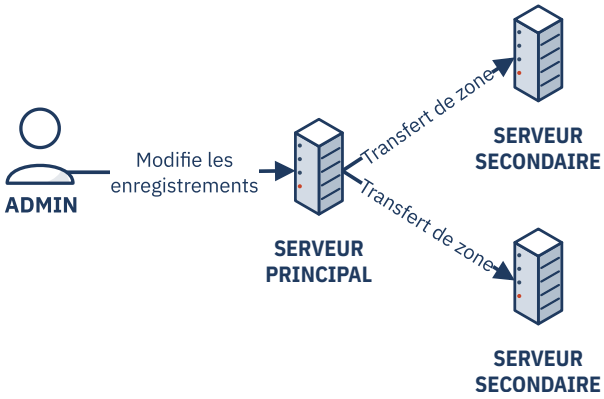


Fig. 65. – Illustration des serveurs DNS principaux et secondaires, montrant la relation entre les serveurs et le processus de transfert de zone.

Qu'un serveur soit principal ou secondaire, il peut être autoritaire pour une zone s'il détient les enregistrements DNS pour cette zone et peut fournir des réponses définitives aux requêtes concernant les noms de domaine dans cette zone.

12.2.5 Serveurs racine

Les serveurs racine sont les serveurs DNS situés au sommet de la hiérarchie du DNS. Ils sont responsables de la gestion des informations sur les domaines de premier niveau (TLD) et servent de point de départ pour la résolution des noms de domaine. Il existe 13 ensembles de serveurs racine, identifiés par les lettres A à M, et chacun est géré par une organisation différente. Ces serveurs sont répartis dans le monde entier et utilisent des techniques de distribution et de redondance pour assurer leur disponibilité et leur fiabilité. Les enregistrements des serveurs racine ont typiquement un TTL élevé pour éviter toute surcharge inutile. En effet, les serveurs racine

sont très stables et ne changent pas souvent, donc il n'est pas nécessaire de les interroger fréquemment.

Le site web https://root-servers.org/ permet de visualiser la localisation des serveurs racine.

12.2.6 Exemple de résolution DNS

Sur Linux / MacOS, avec la commande `dig +trace`, il est possible de suivre le cheminement d'une requête DNS depuis le serveur racine jusqu'au serveur autoritaire pour un nom de domaine spécifique. Cette commande permet d'effectuer une résolution « à froid », donc en ne consultant pas le cache du serveur DNS local. Ci-dessous le résultat (partiel et simplifié) d'un `dig +trace` sur `www.swisscom.ch` (l'option `+additional` permet d'afficher les adresses IP des serveurs DNS autoritaires dans la réponse) :

```
dig +trace +additional www.swisscom.ch
.                       507752  IN      NS      e.root-
servers.net.
.                       507752  IN      NS      l.root-
servers.net.
.                       507752  IN      NS      c.root-
servers.net.
.                       507752  IN      NS      f.root-
servers.net.
.                       507752  IN      NS      b.root-
servers.net.
.                       507752  IN      NS      a.root-
servers.net.
.                       507752  IN      NS      g.root-
servers.net.
.                       507752  IN      NS      d.root-
servers.net.
.                       507752  IN      NS      k.root-
servers.net.
.                       507752  IN      NS      h.root-
servers.net.
.                       507752  IN      NS      j.root-
servers.net.
.                       507752  IN      NS      i.root-
servers.net.
.                       507752  IN      NS      m.root-
servers.net.
a.root-servers.net.     262183  IN      A       198.41.0.4
b.root-servers.net.     600011  IN      A       170.247.170.2
c.root-servers.net.     273038  IN      A       192.33.4.12
```

```
d.root-servers.net.     583003  IN      A       199.7.91.13
e.root-servers.net.     231481  IN      A       192.203.230.10
f.root-servers.net.     139     IN      A       192.5.5.241
g.root-servers.net.     590706  IN      A       192.112.36.4
h.root-servers.net.     142     IN      A       198.97.190.53
i.root-servers.net.     96804   IN      A       192.36.148.17
j.root-servers.net.     516382  IN      A       192.58.128.30
k.root-servers.net.     106504  IN      A       193.0.14.129
l.root-servers.net.     187081  IN      A       199.7.83.42
m.root-servers.net.     249699  IN      A       202.12.27.33
;; Received 1109 bytes from 160.98.2.110#53(160.98.2.110) in 4
ms

ch.                     172800  IN      NS      a.nic.ch.
ch.                     172800  IN      NS      b.nic.ch.
ch.                     172800  IN      NS      d.nic.ch.
ch.                     172800  IN      NS      e.nic.ch.
ch.                     172800  IN      NS      f.nic.ch.
a.nic.ch.               172800  IN      A       130.59.31.41
b.nic.ch.               172800  IN      A       130.59.31.43
d.nic.ch.               172800  IN      A       194.0.25.39
e.nic.ch.               172800  IN      A       194.0.17.1
f.nic.ch.               172800  IN      A       194.146.106.10
;; Received 683 bytes from 192.203.230.10#53(e.root-
servers.net) in 8 ms

swisscom.ch.            3600    IN      NS
dns1.swisscom.com.
swisscom.ch.            3600    IN      NS
dns2.swisscom.com.
swisscom.ch.            3600    IN      NS
dns3.swisscom.com.
;; Received 252 bytes from 194.0.25.39#53(d.nic.ch) in 12 ms

www.swisscom.ch.        3600    IN      CNAME   www-swisscom-
ch.hdb-cs04.ellb.ch.
;; Received 88 bytes from 138.190.34.196#53(dns1.swisscom.com)
in 10 ms
```

On voit les commentaires `;; Received ...` qui indiquent le serveur DNS qui a répondu à la requête précédente, ce qui nous permet de suivre le cheminement de la requête à travers les différents serveurs DNS.

1. Le serveur racine `e.root-servers.net` a répondu avec les serveurs DNS autoritaires pour le TLD `.ch`. Le serveur `d.nic.ch` en est un exemple.

2. Le serveur `d.nic.ch` a répondu avec les serveurs DNS autoritaires pour le domaine `swisscom.ch`. Le serveur `dns1.swisscom.com` en est un exemple.
3. Le serveur `dns1.swisscom.com` a répondu avec l'enregistrement CNAME pour le nom d'hôte `www.swisscom.ch`, qui pointe vers `www-swisscom-ch.hdb-cs04.ellb.ch`.

Ensuite on peut regarder l'IP de l'enregistrement CNAME `www-swisscom-ch.hdb-cs04.ellb.ch` avec un simple `dig www-swisscom-ch.hdb-cs04.ellb.ch` (ou `dig www.swisscom.ch` directement, le CNAME sera résolu automatiquement) :

```
dig www-swisscom-ch.hdb-cs04.ellb.ch

;; ANSWER SECTION:
www-swisscom-ch.hdb-cs04.ellb.ch. 2 IN  A
195.186.208.154
```

Il est également possible d'obtenir le ou les serveurs DNS autoritaires pour un nom de domaine spécifique en utilisant la commande `dig NS <nom_de_domaine>`. Par exemple, pour obtenir les serveurs DNS autoritaires pour le domaine `swisscom.ch`, vous pouvez exécuter la commande suivante :

```
dig NS swisscom.ch

;; ANSWER SECTION:
swisscom.ch.            2188    IN      NS
dns2.swisscom.com.
swisscom.ch.            2188    IN      NS
dns1.swisscom.com.
swisscom.ch.            2188    IN      NS
dns3.swisscom.com.

;; ADDITIONAL SECTION:
dns1.swisscom.com.      929     IN      A       138.190.34.196
dns2.swisscom.com.      2954    IN      A       138.190.34.204
dns3.swisscom.com.      1483    IN      A       193.222.76.54
```

EXERCICE 12.2

La commande `dig +trace` effectue quel type de résolution DNS (récursive ou itérative) ? Pourquoi ?

12.2.7 Systèmes de caches

Comme pour les baux DHCP, les enregistrements DNS on un TTL associé définissant leur temps de validité. Ce temps de vie est exprimé en secondes et indique combien de temps un enregistrement DNS peut être mis en cache par les serveurs DNS intermédiaires et les clients avant d'être considéré comme obsolète. Un TTL plus court permet de mettre à jour plus rapidement les informations DNS, mais peut entraîner une charge plus élevée sur les serveurs DNS, tandis qu'un TTL plus long réduit la charge mais peut retarder la propagation des modifications.

Il existe aussi un temps de cache négatif, qui est utilisé pour les réponses DNS indiquant qu'un nom de domaine n'existe pas. Le cache négatif permet aux serveurs DNS de stocker temporairement les informations sur les noms de domaine inexistants, évitant ainsi de répéter inutilement les requêtes pour ces noms de domaine et réduisant la charge sur les serveurs DNS autoritaires.

12.2.8 Fonctionnement typique

Dans une configuration classique, un serveur DNS local s'occupe de faire les résolution itératives pour les clients du réseau local. Il est configuré avec l'adresse IP d'un ou plusieurs serveurs DNS racine, et il interroge ces serveurs pour obtenir les informations nécessaires à la résolution des noms de domaine demandés par les clients. Il contient un cache pour stocker les réponses aux requêtes DNS précédentes, ce qui permet de réduire le temps de réponse pour les requêtes ultérieures et de diminuer la charge sur les serveurs DNS externes.

Lorsqu'un client dans le réseau local a besoin d'une adresse IP à partir d'un FQDN, il commence par vérifier son fichier `hosts` local (fichier qui défini des enregistrements manuels. `/etc/hosts` sur Linux et MacOS, `C:\Windows\System32\drivers\etc\hosts` sur Windows). Si le nom de domaine n'est pas trouvé dans ce fichier, il regarde dans son cache DNS local pour voir si une résolution précédente a été effectuée récemment. Si le nom de domaine n'est pas trouvé dans le cache, le client demande une résolution recursive à son serveur DNS local. Le serveur DNS local, s'il ne connaît pas la réponse, effectue une résolution itérative en interrogeant les serveurs DNS racine, puis les serveurs TLD, et enfin le serveur autoritaire pour le domaine demandé. Une fois que le serveur DNS local obtient la réponse finale, il la renvoie au client et la stocke dans son cache pour les futures requêtes.

La figure Fig. 66 représente l'architecture typique d'une résolution DNS dans un réseau local, avec un serveur DNS local qui interagit avec les

serveurs DNS racine et autoritaires pour résoudre les noms de domaine demandés par les clients.

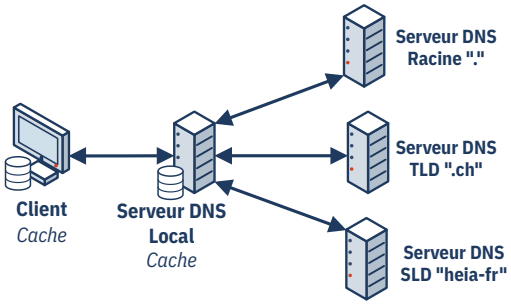


Fig. 66. – Architecture typique d'une résolution DNS dans un réseau local, avec un serveur DNS local qui interagit avec les serveurs DNS racine et autoritaires.

EXERCICE 12.3 · EXERCICE : FICHIER HOSTS ET LOCALHOST

Consulter le fichier `hosts` de votre système d'exploitation et expliquer l'enregistrement lié à `localhost`. Quel type d'enregistrement est-ce et à quoi sert-il ? Essayer de faire un ping sur `localhost` et expliquer le résultat.

Sur MacOS et Linux, on peut le consulter avec la commande `cat /etc/hosts`. Sur Windows, on peut le consulter avec un éditeur de texte en ouvrant le fichier `C:\Windows\System32\drivers\etc\hosts`.

CHAPITRE 13

HTTP

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- décrire l'utilité du protocole HTTP dans la communication entre clients et serveurs web
- lister les principales méthodes HTTP (GET, POST, PUT, DELETE, etc.) et expliquer leur rôle dans les requêtes et réponses
- expliquer une mesure WireShark d'un échange HTTP
- décrire les différences entre HTTP/1.1 et HTTP/2, ainsi que les avantages de HTTP/2 par rapport à HTTP/1.1

Ce chapitre aborde le protocole HTTP (Hypertext Transfer Protocol), qui est le protocole de communication utilisé pour transférer des données sur le World Wide Web.

13.1 World Wide Web et HTTP

Lorsqu'on parle d'Internet dans la vie quotidienne, on fait souvent référence au World Wide Web (WWW) que l'on abrège souvent en « web ». Le web est un système « hypertexte » public fonctionnant sur Internet qui permet de consulter des pages web et d'autres ressources en ligne, au moyen d'un navigateur (browser). La Fig. 67 montre la différence entre Internet et le World Wide Web. Le WWW est un service qui fonctionne sur Internet, alors qu'Internet est le réseau mondial qui permet la communication entre ordinateurs et autres dispositifs connectés.

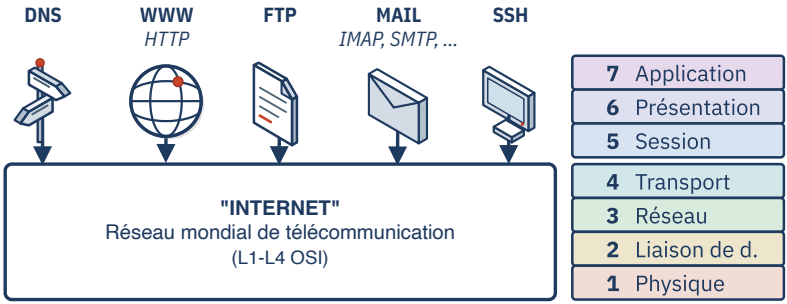


Fig. 67. – Différence entre Internet et le World Wide Web (WWW). Le WWW est un service qui fonctionne sur Internet, au même titre que d'autres services comme les mails (SMTP, IMAP, etc.), le transfert de fichiers (FTP), le DNS ou encore le protocole SSH (Shell sécurisé à distance).

APPROFONDISSEMENT

HyperText signifie « au-delà du texte ». Hyper est un préfixe grec qui signifie « au-delà » ou « supérieur à ». L'idée derrière l'HyperText est de permettre d'obtenir des informations supplémentaires en cliquant sur des liens (hyperliens) qui mènent à d'autres documents ou ressources. Le terme « HyperText » a été inventé par Ted Nelson dans les années 1960, et il a été popularisé par le World Wide Web, qui utilise le langage HTML (HyperText Markup Language) pour créer des pages web contenant du texte, des images, des vidéos et d'autres éléments multimédias interconnectés par des hyperliens.

Le WWW repose sur le protocole HTTP pour la communication entre les clients (navigateurs) et les serveurs web. HTTP est un protocole de communication basé sur le modèle client-serveur, où le client envoie des requêtes au serveur, et le serveur répond avec les ressources demandées.

13.2 URI et URL

Une URI (Uniform Resource Identifier) est une chaîne de caractères qui identifie de manière unique une ressource. Cette ressource peut être située sur Internet ou ailleurs.

Une URL (Uniform Resource Locator) est un type spécifique d'URI qui fournit des informations sur la localisation de la ressource et le protocole à utiliser pour y accéder. Une URL se compose de plusieurs parties :

- La méthode d'accès (ou schéma), qui indique le protocole à utiliser pour accéder à la ressource (par exemple, `http`, `https`, `ftp`, etc.).
- Le nom de domaine ou l'adresse IP du serveur hébergeant la ressource. Pour une IPv6, il est nécessaire de mettre l'adresse entre crochets, par exemple `[2001:db8::1]`.
- Éventuellement, le port utilisé pour la communication (si il n'est pas précisé, le port well-known du protocole est utilisé, par exemple 80 pour HTTP et 443 pour HTTPS).
- Le chemin d'accès à la ressource sur le serveur. N'est pas obligatoire, l'absence de chemin signifie que l'on demande la ressource par défaut du serveur.

Les espaces et autres caractères spéciaux comme le `#` ou le `?` ne sont pas permis tel quel dans une URL. Ils doivent être encodés en utilisant le format d'encodage URL (par exemple, un espace devient `%20`). Lorsqu'on trouve un `%` dans une URL, il est suivi de deux chiffres hexadécimaux représentant le code ASCII du caractère encodé.

Un segment du chemin d'accès peut être vide (tel que défini dans le RFC 3986). Néanmoins, c'est un cas de figure très rare et plutôt anecdotique. Il est plus courant de voir des segments de chemin d'accès non vides, qui représentent des répertoires ou des fichiers sur le serveur.

EXERCICE 13.1 · VALIDITÉ D'URL

Pour chaque URL ci-dessous, indiquer si elle est valide ou non. Si elle n'est pas valide, justifier pourquoi.

Indice : tous les schémas de protocole utilisés ci-dessous existent et sont valides. Le problème, s'il y en a un, se situe donc ailleurs que dans le nom du protocole.

A. `https://www.heia-fr.ch/home.html`

B. `ftp://www.heia-fr.ch/accueil`

C. `http://www.heia-fr.ch:8080/%travaux`

D. `http://www.heia-fr.ch/travaux pratiques/`

E. `http://127.0.0.1:8080/`

F. `smb://server/share`

G. `ws://localhost:8080/level1/level2/level3`

H. `http://www.heia-fr.ch:80/////////////`

13.3 Introduction à HTTP

HTTP est un protocole applicatif qui fonctionne sur le modèle client-serveur. Ses well-known ports sont le port 80 pour HTTP et le port 443 pour HTTPS (HTTP sécurisé). Du côté client, le port n'est pas standardisé et est généralement choisi de manière aléatoire par le système d'exploitation.

Plusieurs versions de HTTP ont été développées au fil du temps, chacune apportant des améliorations et des fonctionnalités supplémentaires. Les principales versions sont :

- HTTP/1.1 : longtemps la version dominante, toujours très répandue mais désormais dépassée en usage par HTTP/2. Publiée en 1999 (RFC 2616), sa spécification à jour est le RFC 9112 (2022), qui a remplacé le RFC 2616.
- HTTP/2 : introduite en 2015 (RFC 7540), elle améliore les performances (multiplexage des requêtes, compression des en-têtes). C'est aujourd'hui la version la plus utilisée. Spécification à jour : RFC 9113 (2022), qui a remplacé le RFC 7540. Supportée par tous les navigateurs modernes.
- HTTP/3 : la version la plus récente (2022, RFC 9114). Elle tourne sur QUIC (au-dessus d'UDP) plutôt que sur TCP, ce qui réduit la latence.

Depuis 2022, la sémantique commune à toutes les versions (méthodes, codes de statut, en-têtes) est définie à part dans le RFC 9110. Les RFC 9112, 9113 et 9114 ne décrivent que le format d'échange propre à HTTP/1.1, /2 et /3.

La version HTTP utilisée est négociée entre le client et le serveur lors de l'établissement de la connexion.

13.4 Requêtes et réponses HTTP

HTTP est un protocole qui fonctionne avec du texte brut dans les requêtes et les réponses. Les requêtes HTTP sont envoyées par le client au serveur, et les réponses HTTP sont renvoyées par le serveur au client. Les requêtes HTTP sont articulées autour du concept de méthode. Une méthode HTTP indique l'action que le client souhaite effectuer sur la ressource identifiée par l'URL. Les méthodes les plus courantes sont `GET`, `POST`, `PUT`, `DELETE`,

`HEAD`, `OPTIONS` et `PATCH`. Chaque méthode a un rôle spécifique dans la manipulation des ressources sur le serveur.

Une requête HTTP se compose de plusieurs parties :

- Une méthode HTTP (GET, POST, etc.) qui indique l'action que le client souhaite effectuer sur la ressource identifiée par l'URL.
- Une URL qui identifie la ressource demandée.
- Des en-têtes HTTP qui fournissent des informations supplémentaires sur la requête.
- Un corps de requête (optionnel) qui contient les données à envoyer au serveur, principalement utilisé avec des méthodes comme `POST`.

Une réponse HTTP quant à elle se compose de :

- Un code de statut HTTP qui indique le résultat de la requête (par exemple, 200 pour succès, 404 pour ressource non trouvée, etc.).
- Des en-têtes HTTP qui fournissent des informations supplémentaires sur la réponse.
- Un corps de réponse (optionnel) qui contient les données renvoyées par le serveur, comme le contenu d'une page web.

Le format textuel des requêtes et réponses HTTP doit être suivi scrupuleusement pour que le client et le serveur puissent se comprendre. L'entête et le corps de la requête ou de la réponse sont séparés par une ligne vide, et chaque en-tête est écrit sur une ligne distincte. Ci-dessous, un exemple de requête HTTP GET et de réponse HTTP correspondante :

```
GET /index.html HTTP/1.1
Host: www.example.com
Accept: text/html
```

```
HTTP/1.1 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html>
<head>
    <title>Example</title>
</head>
<body>
    <h1>Hello, world!</h1>
</body>
</html>
```

Dans l'exemple de réponse, on trouve du HTML (HyperText Markup Language) dans le corps de la réponse, qui est le langage de balisage utilisé pour créer des pages web. Le serveur renvoie le code HTML au client, qui l'interprète et l'affiche dans le navigateur.

> **ATTENTION**
> Il ne faut pas confondre l'HTML et HTTP, ce sont deux concepts différents et complémentaires : HTTP est le protocole de communication, tandis que l'HTML est le format de contenu utilisé pour structurer et présenter les informations sur le web.

13.4.1 Méthodes HTTP

Comme mentionné précédemment, les différentes méthodes HTTP permettent de spécifier l'action que le client souhaite effectuer sur la ressource identifiée par l'URL. Voici quelques-unes des méthodes les plus courantes et leur rôle :

- `GET` : demander une ressource. Avec cette méthode, tous les paramètres de la requête sont inclus dans l'URL après le caractère `?`. Elle est typiquement utilisée pour obtenir une page web.
- `POST` : envoyer des données au serveur pour créer une ressource. Les données sont généralement incluses dans le corps de la requête (donc non visibles dans l'URL). Elle est souvent utilisée pour soumettre des formulaires.
- `HEAD` : similaire à `GET`, mais ne récupère que les en-têtes de la réponse, sans le corps. Utile pour vérifier l'existence d'une ressource ou obtenir des informations sur celle-ci sans télécharger son contenu.
- `PUT` : envoyer des données au serveur pour créer ou remplacer une ressource à l'URL spécifiée. Très similaire à `POST`, mais avec l'intention de remplacer la ressource existante.
- `DELETE` : demander la suppression d'une ressource à l'URL spécifiée. Elle est utilisée pour supprimer une ressource existante sur le serveur.

13.4.2 Statuts HTTP

Les status HTTP sont les codes renvoyés par le serveur pour indiquer le résultat de la requête. Ils sont regroupés en cinq classes, identifiées par le premier chiffre du code :

- 1xx : Informations (ex. 100 Continue)
- 2xx : Succès (ex. 200 OK, 201 Created)
- 3xx : Redirection (ex. 301 Moved Permanently, 302 Found)
- 4xx : Erreur côté client (ex. 400 Bad Request, 404 Not Found)
- 5xx : Erreur côté serveur (ex. 500 Internal Server Error, 503 Service Unavailable)

EXERCICE 13.2

Ouvrir un navigateur web, et atteindre l'URL `https://www.heia-fr.ch`. Ouvrir les outils de développement du navigateur (souvent accessibles via F12 ou Ctrl+Shift+I), et aller dans l'onglet « Réseau » (Network). Recharger la page et observer les requêtes HTTP effectuées par le navigateur.

Quelle est la première requête HTTP effectuée par le navigateur ? Quelle méthode HTTP est utilisée ? Quel est le code de statut de la réponse du serveur pour cette requête ? Quelle version de HTTP est utilisée pour cette requête et réponse ?

13.5 HTTP/1.1 vs HTTP/2

HTTP/2 est une version améliorée du protocole HTTP, qui apporte plusieurs avantages par rapport à HTTP/1.1. Voici quelques différences clés entre les deux versions :

- Multiplexage : HTTP/2 permet d'envoyer plusieurs requêtes et réponses simultanément sur une seule connexion TCP, ce qui réduit la latence et améliore les performances. HTTP/1.1, en revanche, ne peut traiter qu'une seule requête à la fois par connexion.
- Format binaire : HTTP/2 utilise un format binaire pour les messages, ce qui permet une meilleure compression et une transmission plus efficace des données. HTTP/1.1 utilise un format textuel, ce qui peut entraîner une surcharge de données et une latence accrue.
- Server Push : HTTP/2 permet au serveur d'envoyer des ressources supplémentaires au client avant même que celui-ci ne les demande. Néanmoins,

cette fonctionnalité a été désactivée ou supprimée dans la plupart des navigateurs modernes entre 2022 et 2024, car elle a été jugée comme étant peu efficace.

- Et encore d'autres améliorations, comme la compression des en-têtes ou la priorisation des flux.

Un site web, https://moebuta.org/posts/http3-demo/, propose de visualiser la différence de vitesse entre HTTP/1.1, HTTP/2 et même HTTP/3.

EXERCICE 13.3

Lorsqu'on clique une première fois sur les boutons de test des version de HTTP sur le site https://moebuta.org/posts/http3-demo/, l'opération prend un certain temps. Lorsqu'on clique une deuxième fois sur les mêmes boutons, l'opération est beaucoup plus rapide. Pourquoi ?

CHAPITRE 14

Cryptographie, Telnet et SSH

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- Expliquer les concepts de base de la cryptographie moderne, y compris les notions de chiffrement, de déchiffrement, de clés symétriques et asymétriques, ainsi que les fonctions de hachage et la signature numérique
- décrire l'utilité des protocoles Telnet et SSH dans la communication à distance avec des périphériques réseau
- expliquer le fonctionnement des protocoles Telnet et SSH

Ce chapitre aborde tout d'abord les bases de la cryptographie moderne, puis les protocoles Telnet et SSH, qui permettent d'établir une connexion à distance avec un hôte. Le protocole Telnet est obsolète et non sécurisé, tandis que le protocole SSH est sécurisé et largement utilisé pour l'administration à distance des systèmes et des périphériques réseau.

14.1 Cryptographie

La cryptographie est le domaine de la science qui étudie les techniques de chiffrement et de déchiffrement des informations pour garantir la confidentialité, l'intégrité et l'authenticité des données échangées.

DÉFINITION 14.1 · CRYPTOGRAPHIE

Ensemble des techniques de chiffrement qui assurent l'inviolabilité de textes et, en informatique, de données.

- Larousse.fr

En téléinformatique, la cryptographie joue un rôle crucial dans la sécurisation des communications et des données. Les protocoles ont en principe une version sécurisée qui utilise la cryptographie pour protéger les infor-

mations échangées entre les parties. Par exemple, HTTPS est la version sécurisée de HTTP alors que FTPS est la version sécurisée de FTP.

> **ATTENTION**
> La cryptographie est un domaine complexe et sensible. Il est vivement déconseillé de programmer des algorithmes de chiffrement soi-même.

14.1.1 Critères de sécurité

Les critères de la sécurité représentent trois aspects essentiels de la sécurité des données : la confidentialité, l'intégrité et l'authenticité. Ce sont les principes fondamentaux qu'on souhaite respecter en fonction des besoins de sécurité d'une application ou d'un système.

- **La confidentialité** garantit que les informations échangées ne peuvent être lues que par les parties autorisées.
- **L'intégrité** garantit que les données échangées sont complètes et n'ont pas été modifiées de manière non autorisée.
- **L'authenticité** garantit que les parties impliquées dans la communication sont bien celles qu'elles prétendent être et que les messages proviennent de sources légitimes.

14.1.2 Terminologies

Les terminologies utilisées en cryptographie sont nombreuses et peuvent prêter à confusion. Voici quelques définitions pour clarifier certains termes clés :

- **Cryptographie** : science et art de chiffrer et déchiffrer des informations pour assurer leur sécurité.
- **Cryptanalyse** : étude et pratique de la détection des faiblesses dans les systèmes de chiffrement, afin de les casser ou de les contourner.
- **Clé** : élément secret (ou non) permettant de chiffrer et déchiffrer des données. Il existe différents types de clés, notamment les clés symétriques et asymétriques.
- **Chiffrement** : processus de transformation des données en un format illisible pour les parties non autorisées, à l'aide d'un algorithme et d'une clé de chiffrement.
- **Déchiffrement** : processus inverse du chiffrement, qui permet de récupérer les données originales à partir des données chiffrées, en utilisant l'algorithme et la clé appropriée.
- **Décryptage** : processus de récupération des données originales à partir des données chiffrées, mais **sans** posséder la clé de chiffrement.

- **Données en clair** : données lisibles et compréhensibles par les parties autorisées, avant le chiffrement.
- **Données chiffrées** : données transformées en un format illisible pour les parties non autorisées, après le chiffrement.

> **ATTENTION**
> Le chiffrement et le codage sont des notions complètement différentes. Le codage est un processus de transformation des données pour les rendre compatibles avec un format ou un protocole spécifique, tout en préservant leur lisibilité et leur signification. Le chiffrement, en revanche, est un processus de transformation des données pour les rendre illisibles et inaccessibles aux parties non autorisées, dans le but de garantir la confidentialité et la sécurité des informations échangées.
>
> Parmis les différents encodages, on peut citer l'encodage Base64, qui est utilisé pour représenter des données binaires sous forme de texte ASCII, ou encore l'encodage URL, qui permet de représenter des caractères spéciaux dans les URL.

14.1.3 Algorithmes de chiffrement

Un algorithme de chiffrement est une méthode pour transformer des données en clair en données chiffrées, de manière à ce qu'elles ne puissent être lues que par les parties autorisées possédant la clé appropriée.

De nombreux algorithmes de chiffrement existent et ont défilé au cours de l'histoire. On retrouve l'origine des premiers algorithmes de chiffrement dans l'Antiquité, avec des méthodes comme le chiffre de César. De nos jours, les algorithmes de chiffrement modernes sont basés sur des principes mathématiques complexes et sont conçus pour résister aux attaques cryptographiques. On peut citer des algorithmes de chiffrement symétriques comme AES (Advanced Encryption Standard) et des algorithmes de chiffrement asymétriques comme RSA (Rivest-Shamir-Adleman).

14.1.4 Fonction de hachage

Une fonction de hachage est un algorithme qui prend en entrée des données de taille variable et produit une valeur de taille fixe, appelée « empreinte » ou « hash ». Les fonctions de hachage sont utilisées pour vérifier l'intégrité des données, car même un petit changement dans les données d'origine entraîne un changement significatif dans la valeur de hachage.

Le hachage **n'est pas** du chiffrement, car il est conçu pour être irréversible. Contrairement au chiffrement, il n'existe pas de clé permettant de retrouver les données originales à partir de la valeur de hachage.

Les fonctions de hachage plus anciennes, comme MD5 et SHA-1, sont désormais considérées comme vulnérables aux attaques et ne sont plus recommandées pour les applications de sécurité. De nos jours, des fonctions de hachage plus sécurisées comme SHA-256 et SHA-3 sont utilisées pour garantir l'intégrité des données.

14.1.5 Clés symétriques et asymétriques

Comme mentionné précédemment, il existe deux types principaux de clés utilisées dans les algorithmes de chiffrement : les clés symétriques et les clés asymétriques.

Les clés symétriques sont utilisées pour le chiffrement et le déchiffrement des données. La même clé est utilisée pour les deux opérations, ce qui signifie que les parties autorisées doivent partager la clé secrète avant de pouvoir communiquer en toute sécurité. Les algorithmes de chiffrement symétriques sont généralement plus rapides que les algorithmes asymétriques, mais ils présentent des défis en matière de distribution et de gestion des clés. La Fig. 68 illustre l'utilisation d'une clé symétrique pour chiffrer et déchiffrer un message entre Alice et Bob.

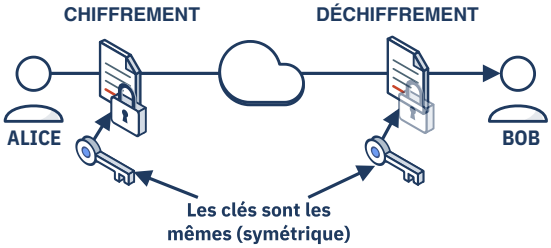


Fig. 68. – Chiffrement et déchiffrement avec une clé symétrique. Alice chiffre le message avec la clé avant de l'envoyer à Bob. Bob utilise la même clé pour déchiffrer le message et le lire.

Les clés asymétriques, en revanche, utilisent une paire de clés : une clé privée, à ne jamais divulguer, et une clé publique, qui peut être partagée avec n'importe qui. Chaque clé à un rôle, et une clé ne peut pas être retrouvée à partir de l'autre (en tout cas pas facilement !). Par exemple, si Alice veut envoyer un message chiffré à Bob, elle utilise la clé publique de Bob pour chiffrer le message. Bob peut ensuite utiliser sa clé privée pour déchiffrer le message. La Fig. 69 illustre l'utilisation d'une paire de clés asymétriques pour chiffrer et déchiffrer un message entre Alice et Bob.

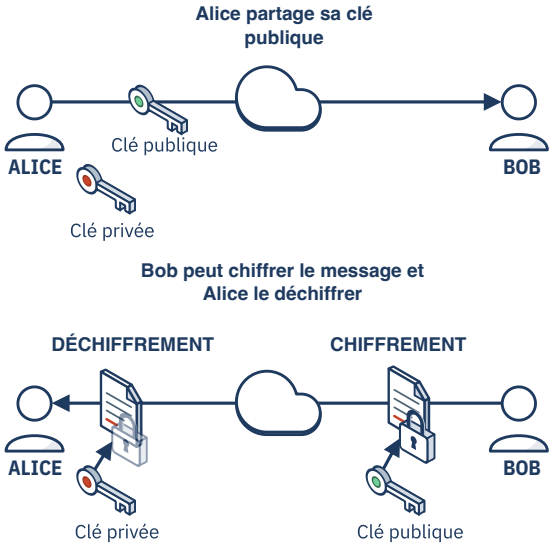


Fig. 69. – Chiffrement et déchiffrement avec une paire de clés asymétriques. Tout d'abord, Alice envoie sa clé publique à Bob. Ensuite, Bob chiffre le message avec la clé publique d'Alice avant de l'envoyer. Alice utilise sa clé privée pour déchiffrer le message et le lire.

RSA est « réversible » dans le sens où il est possible de chiffrer avec la clé publique et déchiffrer avec la clé privée, mais aussi de chiffrer avec la clé privée et déchiffrer avec la clé publique. C'est une exception parmi les algorithmes asymétriques, qui sont généralement conçus pour être utilisés dans un seul sens (chiffrement avec la clé publique et déchiffrement avec la clé privée).

EXERCICE 14.1

Pour les deux cas d'utilisation suivants de RSA, expliquer quelles critères de sécurités sont garantis :

- Alice chiffre le message avec la clé publique de Bob et lui envoie le message. Bob déchiffre le message avec sa clé privée.
- Bob chiffre le message avec sa clé privée et l'envoie à Alice. Alice déchiffre le message grâce à la clé publique de Bob.

14.1.6 Signature numérique avec RSA

La signature numérique est un procédé permettant de garantir l'authenticité et l'intégrité d'une donnée. Elle repose sur l'utilisation d'une fonction de hachage et d'une clé privée pour générer une signature unique pour la donnée à signer. Le procédé, tel qu'illustré dans la Fig. 70, est le suivant :

1. Alice souhaite envoyer un message à Bob et garantir que le message est authentique et n'a pas été modifié en transit. Pour ce faire, elle commence par passer son message dans une fonction de hachage, qui génère une empreinte unique du message.
2. Alice signe l'empreinte du message avec sa clé privée, créant ainsi une signature numérique.
3. Alice envoie ensuite le message original et la signature numérique à Bob.
4. Bob reçoit le message et la signature numérique. Pour vérifier l'authenticité et l'intégrité du message, il passe d'abord le message original dans la même fonction de hachage qu'Alice, générant ainsi une nouvelle empreinte du message.
5. Bob utilise ensuite la clé publique d'Alice pour vérifier la signature numérique.
6. Bob compare l'empreinte qu'il a générée avec l'empreinte signée par Alice. Si les deux empreintes correspondent, cela signifie que le message est authentique et intègre.

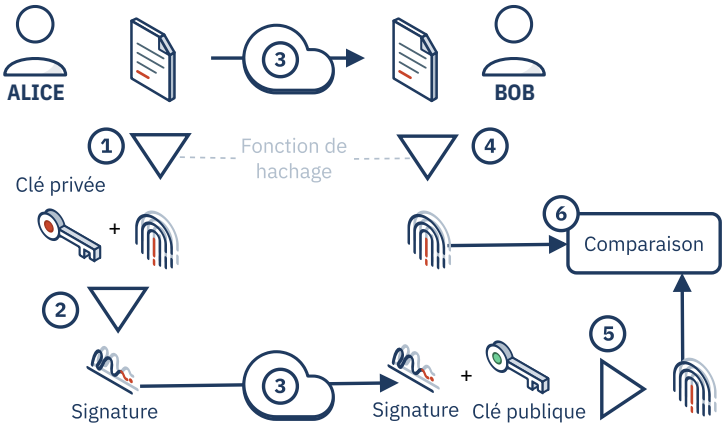


Fig. 70. – Déroulement de la signature numérique, entre Alice et Bob. Les numéros indiquent l'ordre des étapes et correspondent aux explications ci-dessus.

> **ATTENTION**
> Une confusion fréquente est de croire que la signature numérique chiffre la donnée. En réalité, elle ne fait que signer un hachage de la donnée, et non la donnée elle-même.

La signature numérique a également l'avantage de fournir une preuve de « non-répudiation », c'est-à-dire que le signataire ne peut pas nier avoir signé la donnée, car seule sa clé privée aurait pu générer cette signature.

14.2 Telnet

Telnet est un protocole applicatif (L7) qui utilise le well-known port 23 pour établir une connexion à distance avec un hôte. Il est basé sur TCP et fonctionne selon l'architecture client-serveur.

L'objectif de Telnet est d'émuler un terminal à distance (terminal virtuel) sur l'hôte distant, permettant à l'utilisateur d'exécuter des commandes comme s'il était physiquement présent devant le système.

Telnet n'est pas sécurisé, car il transmet les données en clair, y compris les identifiants et mots de passe. Cela le rend vulnérable à différents types d'attaques. Il est déconseillé d'utiliser Telnet aujourd'hui, mais il reste un

outil intéressant pour l'apprentissage et la compréhension des concepts de communication à distance.

Telnet peut être utilisé pour manuellement tester des services réseau qui utilisent des protocoles textuels comme HTTP ou SMTP. Par exemple, vous pouvez utiliser Telnet pour envoyer une requête HTTP à un serveur web et observer la réponse. Il est également possible d'envoyer un email manuellement via un client Telnet en se connectant à un serveur SMTP et en suivant le protocole de communication.

ANECDOTE · Voir Star Wars en ligne de commande

Il est possible de regarder Star Wars en ligne de commande, grâce à Telnet. Pour ce faire, ouvrez un terminal et tapez la commande suivante :

```
telnet telehack.com
```

Une fois connecté, tapez la commande `starwars` pour lancer le film. Vous pouvez également utiliser la commande `help` pour obtenir une liste des commandes disponibles sur le serveur Telnet. Vous verrez alors un film ASCII de Star Wars Episode IV : A New Hope. Que la force soit avec vous !

14.3 SSH

SSH (Secure Shell) est un protocole applicatif (L7) qui utilise le well-known port 22 pour établir une connexion sécurisée à distance avec un hôte. Il est basé sur TCP et fonctionne selon l'architecture client-serveur.

C'est un protocole sécurisé équivalent à Telnet dans ses fonctionnalités, mais pas dans son fonctionnement. SSH **chiffre** les données échangées entre le client et le serveur, garantissant la confidentialité et l'intégrité des informations transmises. SSH supporte différentes formes de **chiffrement** et d'**authentification**.

Il s'agit d'un protocole très utilisé pour l'administration à distance des systèmes et des périphériques réseau, ainsi que pour le transfert de fichiers via des protocoles comme SFTP (SSH File Transfer Protocol) et SCP (Secure Copy Protocol).

SSH permet d'utiliser deux méthodes principales d'authentification pour établir une connexion sécurisée : l'authentification par mot de passe et l'authentification par paire de clés asymétriques.

L'authentification par mot de passe est basique dans son fonctionnement : l'utilisateur fournit un nom d'utilisateur et un mot de passe pour se connecter à l'hôte distant. Cette méthode est simple à mettre en place, mais elle est moins sécurisée que l'authentification par paire de clés asymétriques.

L'authentification par paire de clés asymétriques repose sur un couple de clés : une clé publique et une clé privée. L'utilisateur génère une paire de clés sur son système local, puis copie la clé publique sur l'hôte distant. Lorsqu'il tente de se connecter, le serveur utilise la clé publique pour vérifier l'identité de l'utilisateur en lui demandant de prouver qu'il possède la clé privée correspondante. Cette méthode est plus sécurisée que l'authentification par mot de passe.

EXERCICE 14.2

Avec les connaissances acquises dans ce chapitre, expliquer comment SSH peut concrètement vérifier que l'utilisateur possède la clé privée correspondante à la clé publique stockée sur le serveur, sans que l'utilisateur ait besoin de transmettre sa clé privée au serveur.

Lorsqu'un client se connecte à un serveur SSH, le serveur envoie sa clé publique au client. Le client utilise cette clé publique pour chiffrer les données qu'il envoie au serveur, garantissant que seul le serveur peut déchiffrer ces données avec sa clé privée. De plus, cette clé est stockée dans le fichier `known_hosts` du client.

EXERCICE 14.3

Un utilisateur se connecte pour la première fois à un serveur SSH et accepte l'empreinte de la clé publique du serveur sans la vérifier. Un mois plus tard, en se reconnectant, il voit apparaître le message d'avertissement suivant : «WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!»

Que signifie ce message ? Pourquoi SSH stocke-t-il la clé publique du serveur après la première connexion ? Citez une raison légitime qui pourrait expliquer ce changement, et une raison qui devrait inquiéter

l'utilisateur. Quel critère de sécurité (confidentialité, intégrité, authenticité) ce mécanisme de vérification permet-il de garantir, et pourquoi son absence rendrait-elle risquée l'authentification par clé publique déjà vue dans ce chapitre ?

CHAPITRE 15

Mail et FTP

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir les différents protocoles de messagerie électronique (SMTP, POP3, IMAP) et leur rôle dans l'envoi et la réception d'e-mails
- expliquer les commandes SMTP et POP3
- expliquer le processus d'acheminement d'un e-mail, de l'expéditeur au destinataire, en passant par les serveurs de messagerie
- décrire l'utilité du protocole FTP
- expliquer le fonctionnement global du protocole FTP, y compris les modes actif et passif, ainsi que les commandes FTP courantes

Ce chapitre aborde deux services essentiels dans le domaine de la téléinformatique : la messagerie électronique et le transfert de fichiers.

15.1 Mail

Le mail ou l'email (messagerie électronique) est un service de communication qui permet l'envoi et la réception de messages via Internet. Il repose sur trois protocoles principaux :

- SMTP (Simple Mail Transfer Protocol) : utilisé pour l'envoi d'e-mails depuis le client vers le serveur de messagerie et entre serveurs de messagerie.
- IMAP (Internet Message Access Protocol) : utilisé pour la gestion des e-mails sur le serveur, permettant aux utilisateurs de synchroniser leurs messages sur plusieurs appareils.
- POP3 (Post Office Protocol version 3) : utilisé pour la récupération des e-mails depuis un client à partir d'un serveur de messagerie. Plus simple que IMAP, il télécharge les messages sur l'appareil de l'utilisateur et les supprime généralement du serveur.

La Fig. 71 illustre le processus global d'envoi et de réception d'un e-mail, mettant en évidence les interactions entre le client de messagerie, le

serveur SMTP et le serveur de messagerie du destinataire. Les numéros sur la figure correspondent aux étapes ci-après :

1. L'email est envoyé par le client de messagerie de l'expéditeur au serveur SMTP de l'expéditeur.
2. L'email est ensuite transféré du serveur SMTP de l'expéditeur au serveur SMTP du destinataire.
3. Le client de messagerie du destinataire récupère l'email depuis le serveur de messagerie du destinataire en utilisant IMAP ou POP3.

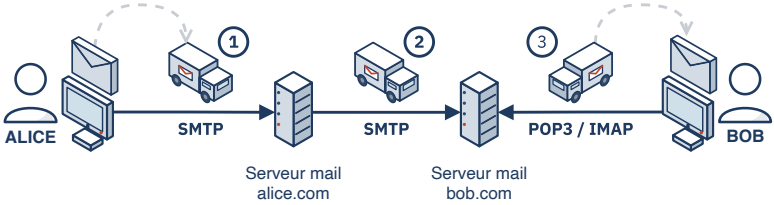


Fig. 71. – Schéma illustrant le processus d'envoi et de réception d'un e-mail, mettant en évidence les interactions entre le client de messagerie, le serveur SMTP et le serveur de messagerie du destinataire. Les numéros indiquent l'ordre des étapes dans le processus d'envoi et de réception d'un e-mail, et correspondent aux explications détaillées dans le texte.

15.1.1 Structure d'un e-mail

Un email est composé de deux parties principales : l'en-tête et le corps du message. L'en-tête contient des informations telles que l'expéditeur, le destinataire, la date et l'objet du message. Le corps du message contient le contenu réel de l'email, qui peut inclure du texte, des images et des pièces jointes.

La RFC 5322 définit le format d'un email tel que stocké sur un serveur de messagerie et lu par un client de messagerie. Un message est composé d'un en-tête et d'un corps, séparés par une ligne vide. Ci-dessous une liste non-exhaustive des champs qui composent l'en-tête d'un e-mail, avec une brève description de chacun :

- `From` : l'adresse e-mail de l'expéditeur. Obligatoire.
- `Date` : la date et l'heure d'envoi du message. Obligatoire.
- `Sender` : l'adresse e-mail de l'expéditeur réel si différente de `From`. Facultatif.
- `Reply-To` : l'adresse e-mail à laquelle les réponses doivent être envoyées. Facultatif.
- `To` : l'adresse e-mail du destinataire principal. Facultatif.

- `Cc` : les adresses e-mail des destinataires en copie carbone. Facultatif.
- `Bcc` : les adresses e-mail des destinataires en copie carbone invisible. Ce champ est retiré avant la transmission du message. Facultatif.
- `Message-ID` : un identifiant unique pour le message. Facultatif.
- `Subject` : l'objet du message. Facultatif.
- `MIME-Version` : la version MIME utilisée pour le message. Facultatif.

Voici un exemple d'email simple avec un en-tête et un corps de message, au format texte brut :

```
Date: Mon, 20 Jul 2026 17:28:22 +0200
From: Alice <alice@labo.local>
To: bob <bob@labo.local>
Subject: Salut bob
Reply-To: alice@labo.local
Message-ID: <43de404a2601db42c5cd3080302922ba@labo.local>
MIME-Version: 1.0
Content-Type: text/plain; charset=US-ASCII;
 format=flowed
Content-Transfer-Encoding: 7bit

Salut Bob,
J'espere que tu vas bien. Je voulais te rappeler notre reunion
de demain a 10h00. N'oublie pas d'apporter les documents
necessaires.
A demain,
Alice
```

APPROFONDISSEMENT

MIME (Multipurpose Internet Mail Extensions) est une extension du format de message qui contourne une limite historique de SMTP : le protocole ne transporte que du texte US-ASCII sur 7 bits. Sans MIME, il est impossible d'envoyer un simple accent ou un fichier binaire. MIME définit des en-têtes supplémentaires décrivant le type de contenu et son encodage, ainsi qu'un mécanisme de découpage du corps en plusieurs parties, chacune avec son propre type et son propre encodage. Le contenu non-ASCII est réencodé en ASCII (base64 ou quoted-printable) pour la traversée du réseau, puis reconstitué par le client destinataire. Les types de médias définis par MIME (`text/html` ,

`image/png`, `application/pdf`, ...) ont depuis été réutilisés par d'autres protocoles notamment par HTTP.

15.2 SMTP

Le protocole SMTP est utilisé pour l'envoi d'e-mails. Il fonctionne sur le port 25 en TCP et utilise un modèle client-serveur. De manière similaire à HTTP, SMTP est un protocole textuel basé sur des commandes et des réponses. Le client SMTP envoie des commandes au serveur SMTP, qui répond avec des codes de statut pour indiquer le succès ou l'échec des opérations.

SMTP ajoute une « enveloppe » autour du message, qui contient les adresses de l'expéditeur et du destinataire. Cette enveloppe est utilisée par les serveurs de messagerie pour acheminer le message vers le bon destinataire, indépendamment des adresses spécifiées dans l'en-tête du message.

L'enveloppe SMTP contient deux champs :

- `MAIL FROM` : l'adresse e-mail de l'expéditeur.
- `RCPT TO` : l'adresse e-mail du destinataire.

Ce sont eux qui font foi pour l'acheminement du message, et non les champs `From` et `To` du contenu de l'e-mail. Cela signifie que l'adresse de l'expéditeur et du destinataire dans l'en-tête du message peut être différente de celle utilisée dans l'enveloppe SMTP.

> **REMARQUE**
> Cette différence potentielle entre l'enveloppe et l'en-tête peut être la source d'attaques de type « spoofing » ou « usurpation d'identité », où un expéditeur malveillant peut falsifier l'adresse de l'expéditeur dans l'en-tête pour tromper le destinataire. C'est pourquoi il est important de vérifier les en-têtes des e-mails et d'utiliser des mécanismes d'authentification pour réduire les risques de spoofing.

15.2.1 Commandes

Les commandes principales de SMTP incluent :

- `HELO` ou `EHLO` : pour s'identifier auprès du serveur SMTP.
- `MAIL FROM` : pour spécifier l'adresse e-mail de l'expéditeur (enveloppe SMTP).
- `RCPT TO` : pour spécifier l'adresse e-mail du destinataire (enveloppe SMTP).

- `DATA` : pour indiquer que le corps du message va suivre. Se termine par une ligne contenant uniquement un point (`.`), puis un retour à la ligne.
- `QUIT` : pour terminer la session SMTP.

Aucune authentification n'est demandée tout au long de la session SMTP, ce qui permet à n'importe qui d'envoyer des e-mails en se faisant passer pour quelqu'un d'autre. C'est pourquoi il existe des extensions de sécurité telles que STARTTLS ou SMTPS pour chiffrer les communications et ajouter une couche d'authentification (voir Chapitre 15.2.4).

15.2.2 Codes de réponse

Les codes de réponse SMTP sont organisés en classes, à la manière de HTTP, avec des codes à trois chiffres. Par exemple, un code 250 indique que la commande a été acceptée avec succès. Les catégories de codes sont les suivantes :

- 1xx : réservée (non utilisée).
- 2xx : la requête a été traitée avec succès.
- 3xx : le serveur demande des informations supplémentaires.
- 4xx : une erreur temporaire s'est produite, le client peut réessayer.
- 5xx : une erreur rend la requête impossible à traiter.

15.2.3 Exemple d'échange

Ci-dessous un exemple d'échange SMTP entre un client et un serveur :

```
S: 220 smtp.server.com Simple Mail Transfer Service Ready
C: HELO client.example.com
S: 250 Hello client.example.com
C: MAIL FROM:<mail@example.com>
S: 250 OK
C: RCPT TO:<john.doe@mail.com>
S: 250 OK
C: DATA
S: 354 Send message content; end with <CRLF>.<CRLF>
C: <Contenu de l'email, incluant le corps, l'objet, les pièces
jointes, etc.>
C: .
S: 250 OK: queued as 12345
C: QUIT
S: 221 Bye
```

15.2.4 Sécurité avec SMTP

Aujourd'hui, SMTP est utilisé avec des extensions de sécurité telles que STARTTLS ou SMTPS pour chiffrer les communications entre le client et le serveur, assurant ainsi la confidentialité et l'intégrité des e-mails échangés.

Avec un SMTP non sécurisé, n'importe qui peut intercepter les e-mails en clair. De plus, l'authentification forte n'est pas requise, ce qui permet d'envoyer des e-mails en se faisant passer pour quelqu'un d'autre (spoofing). Les ports utilisés avec les versions sécurisées de SMTP sont généralement le port 465 pour SMTPS et le port 587 pour STARTTLS.

SMTPS et STARTTLS ajoutent le chiffrement des communications à SMTP. Ils sont généralement combinés avec d'autres mécanismes de sécurité complémentaires, mais indépendants du chiffrement lui-même :

- SMTP AUTH, pour exiger une authentification avant l'envoi d'e-mails.
- SPF, DKIM et DMARC, qui reposent sur le DNS et la signature des messages pour vérifier la légitimité de l'expéditeur, indépendamment de la connexion utilisée.

> **REMARQUE**
> Dans le cadre du cours, nous ne rentrons pas dans les détails des versions sécurisées de SMTP. Néanmoins il est important de savoir que SMTP seul n'est pas sécurisé.

15.3 Routage des emails

Pour comprendre comment un email est transmis d'un expéditeur à un destinataire, il est essentiel de comprendre la structure d'une adresse e-mail. Une adresse e-mail est composée de deux parties principales : le nom d'utilisateur et le domaine, séparés par le symbole `@`. Par exemple, dans l'adresse `contact@alice.com`, `contact` est le nom d'utilisateur et `alice.com` est le domaine. Une adresse email ne peut pas contenir d'espaces, de caractères spéciaux (à l'exception de certains caractères autorisés comme le point, le tiret et l'underscore) ou de caractères accentués. La casse[10] des lettres n'a pas d'importance dans le domaine, et en pratique dans le nom d'utilisateur non plus (formellement, ce n'est pas garanti pour le nom d'utilisateur, mais en pratique, la plupart des serveurs de messagerie ne font pas de distinction entre majuscules et minuscules dans le nom d'utilisateur).

[10] La casse désigne la distinction entre les lettres majuscules et minuscules.

Dans le cadre d'un envoi d'email, on distingue trois parties principales :

- Le MUA (Mail User Agent) : le client de messagerie utilisé par l'expéditeur pour composer et envoyer l'email.
- Le MTA (Mail Transfer Agent) : le serveur SMTP de l'expéditeur qui reçoit l'email du MUA et le transmet au serveur SMTP du destinataire.
- Le MDA (Mail Delivery Agent) : le serveur de messagerie du destinataire qui reçoit l'email du MTA et le stocke dans la boîte de réception du destinataire.

Un serveur de messagerie, ou serveur mail, peut être un MTA, un MDA ou les deux.

L'analogie avec le courrier postal est la suivante : le MUA est l'expéditeur qui écrit la lettre, le MTA est le service postal qui transporte la lettre, et le MDA est la boîte aux lettres du destinataire où la lettre est déposée.

Le MTA de l'expéditeur doit déterminer l'adresse IP du MTA du destinataire. Pour ce faire, il utilise le DNS pour rechercher les enregistrements MX (Mail Exchange) associés au domaine du destinataire. La Fig. 72 illustre le processus de résolution DNS pour trouver le serveur SMTP du destinataire :

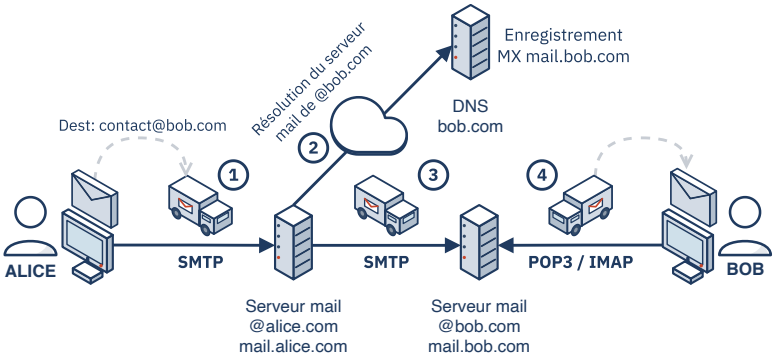


Fig. 72. – Résolution DNS pour trouver le serveur SMTP du destinataire. Le serveur SMTP de l'expéditeur interroge le DNS pour obtenir les enregistrements MX du domaine du destinataire, puis sélectionne le serveur SMTP approprié pour transmettre l'e-mail. Les étapes sont numérotées pour correspondre aux explications détaillées dans le texte.

1. Le MUA de l'expéditeur envoie son email à son MTA.
2. Le MTA de l'expéditeur interroge le DNS pour obtenir le(s) enregistrement(s) MX du domaine du destinataire.

3. Le MTA de l'expéditeur transmet l'email au MTA du destinataire en utilisant l'adresse IP obtenue à partir de l'enregistrement MX.
4. Le MUA du destinataire récupère l'email depuis le MDA en utilisant IMAP ou POP3.

EXERCICE 15.1 · IDENTIFICATION DES RÔLES

A partir de la Fig. 72, identifier le rôle des acteurs suivants dans le processus d'envoi et de réception d'un e-mail de Alice à Bob :

1. Le client de messagerie qu'utilise Alice pour envoyer son email
2. Le serveur mail @bob.com
3. Le serveur mail @alice.com
4. Le client de messagerie qu'utilise Bob pour récupérer son email

15.4 POP3

Le protocole POP3 est utilisé pour récupérer les e-mails depuis un serveur de messagerie vers un client de messagerie. Il fonctionne sur le port 110 en TCP et utilise un modèle client-serveur. Le fonctionnement de POP3 est simple : le client se connecte au serveur, télécharge les e-mails et les supprime du serveur (dans la majorité des cas d'utilisation). Cependant, certains clients peuvent être configurés pour laisser une copie des e-mails sur le serveur.

POP3 peut également être utilisé avec SSL/TLS pour sécuriser les communications, généralement sur le port 995 (POP3S).

Les commandes principales de POP3 incluent :

- `USER` : pour spécifier le nom d'utilisateur du compte de messagerie.
- `PASS` : pour spécifier le mot de passe du compte de messagerie (envoyé en clair dans le cas de POP3 !).
- `LIST` : pour lister les e-mails disponibles sur le serveur, avec leur taille en octets.
- `DELE <numéro>` : pour supprimer un e-mail spécifique du serveur.
- `RETR <numéro>` : pour récupérer un e-mail spécifique depuis le serveur.
- `STAT` : pour obtenir le nombre total d'e-mails et la taille totale des e-mails sur le serveur.
- `TOP <numéro> <nombre de lignes>` : pour récupérer uniquement les entêtes et un nombre spécifié de lignes du corps d'un e-mail.

POP3 est un protocole simple et efficace pour la récupération des e-mails, mais il présente certaines limitations, notamment le fait qu'il ne permet pas de gérer les dossiers ou de synchroniser les e-mails entre plusieurs appareils. C'est pourquoi IMAP est souvent préféré pour une utilisation plus avancée de la messagerie électronique.

Ci-dessous un exemple d'échange POP3 entre un client et un serveur :

```
S: +OK POP3 GreenMail Server v2.1.3 ready
C: USER bob
S: +OK
C: PASS bob123
S: +OK
C: LIST
S: +OK
   1 160
   2 160
   ...
   22 167
C: RETR 22
S: +OK
   X-Antivirus: avg (VPS 26072012)
   X-Antivirus-Status: Clean
   Return-Path: <alice@labo.local>
   Received: from 10.0.0.10 (HELO poste-alice); Mon, 20 Jul
2026 17:28:22 +0200
   Subject: Salut bob
   From: Alice <alice@labo.local>
   Reply-To: <alice@labo.local>
   To: bob@labo.local

   Salut Bob,
   J'espère que tu vas bien. Je voulais te rappeler notre
réunion de demain à 10h00. N'oublie pas d'apporter les
documents nécessaires.
   À demain,
   Alice
```

EXERCICE 15.2 · ANALYSE D'UN ÉCHANGE POP3

A partir de l'exemple d'échange POP3 ci-dessus, identifier les éléments suivants :

- L'adresse e-mail de l'expéditeur

- L'adresse e-mail du destinataire
- La taille en octets du message récupéré
- Le mot de passe de l'utilisateur utilisé pour se connecter au serveur POP3
- Le `HELO` utilisé par le client pour s'identifier auprès du serveur

EXERCICE 15.3 · RÉCUPÉRATION ET SUPPRESSION DES E-MAILS AVEC POP3

Quelle est la suite de commandes POP3 classique pour récupérer les e-mails d'un utilisateur et les supprimer du serveur après récupération ?

15.5 IMAP

IMAP (Internet Message Access Protocol) est un protocole de messagerie électronique qui permet aux utilisateurs de gérer leurs e-mails directement sur le serveur, plutôt que de les télécharger sur leur appareil. Contrairement à POP3, IMAP permet la synchronisation des e-mails entre plusieurs appareils et offre des fonctionnalités avancées telles que la gestion des dossiers et la recherche d'e-mails. C'est en général le protocole préféré pour une utilisation plus avancée de la messagerie électronique.

IMAP suit le modèle client-serveur et fonctionne sur le port 143 en TCP. Il utilise également SSL/TLS pour sécuriser les communications, généralement sur le port 993 (IMAPS).

Les commandes du protocole IMAP sont plus complexes que celles de POP3, car elles permettent une gestion plus fine des e-mails et des dossiers. Ci-dessous une liste non-exhaustive des commandes IMAP courantes (elles ne sont pas sensibles à la casse) :

- `LOGIN <nom_utilisateur> <mot_de_passe>` : pour s'authentifier auprès du serveur IMAP.
- `SELECT <dossier>` : pour sélectionner un dossier spécifique (par exemple, « INBOX ») et accéder à ses e-mails.
- `FETCH <numéro> <attributs>` : pour récupérer des informations sur un e-mail spécifique, comme son en-tête ou son corps.
- `STORE <numéro> <attributs>` : pour modifier les attributs d'un e-mail, comme le marquer comme lu ou non lu.

- `SEARCH <critères>` : pour rechercher des e-mails correspondant à des critères spécifiques.
- `LOGOUT` : pour terminer la session IMAP.

Voici un exemple d'échange IMAP entre un client et un serveur :

```
S: * OK IMAP4rev1 Server GreenMail v2.1.3 ready
C: a01 login bob bob123
S: a01 OK LOGIN completed.
C: a02 select inbox
S: * FLAGS (\Answered \Deleted \Draft \Flagged \Seen)
   * 24 EXISTS
   * 0 RECENT
   * OK [UIDVALIDITY 1784557357]
   * OK [UIDNEXT 25]
   * OK [UNSEEN 23] Message 23 is the first unseen
   * OK [PERMANENTFLAGS (\Answered \Deleted \Draft \Flagged
\Seen \*)]
   a02 OK [READ-WRITE] SELECT completed.
C: a03 fetch 24 envelope
S: * 24 FETCH (ENVELOPE ("Tue, 21 Jul 2026 14:52:04 +0000
(GMT)" "Salut bob" (("Alice" NIL "alice" "labo.local"))
(("Alice" NIL "alice" "labo.local")) ((NIL NIL "alice"
"labo.local")) ((NIL NIL "bob" "labo.local")) NIL NIL NIL
NIL))
  a03 OK FETCH completed.
C: a04 fetch 24 full
S: * 24 FETCH (FLAGS (\Seen) INTERNALDATE "21-Jul-2026
14:52:04 +0000" RFC822.SIZE 447 ENVELOPE ("Tue, 21 Jul 2026
14:52:04 +0000 (GMT)" "Salut bob" (("Alice" NIL "alice"
"labo.local")) (("Alice" NIL "alice" "labo.local")) ((NIL NIL
"alice" "labo.local")) ((NIL NIL "bob" "labo.local")) NIL NIL
NIL NIL) BODY ("text" "plain" NIL NIL NIL "7BIT" 167 4))
   a04 OK FETCH completed.
C: a05 fetch 24 body[]
S: * 24 FETCH (BODY[] {447}
   X-Antivirus: avg (VPS 26072012)
   X-Antivirus-Status: Clean
   Return-Path: <alice@labo.local>
   Received: from 160.98.22.148 (HELO poste-direction); Tue
Jul 21 14:52:04 GMT 2026
   Subject: Salut bob
   From: Alice <alice@labo.local>
   Reply-To: <alice@labo.local>
   To: bob@labo.local
```

```
   Salut Bob,
   J'espère que tu vas bien. Je voulais te rappeler notre
réunion de demain à 10h00. N'oublie pas d'apporter les
documents nécessaires.
   À demain,
   Alice)
   a05 OK FETCH completed.
C: a06 search FROM "alice@labo.local"
S: * SEARCH 4 5 19 20 22 23 24
   a06 OK SEARCH completed.
C: a07 logout
S: * BYE IMAP4rev1 Server logging out
   a07 OK LOGOUT completed.
```

EXERCICE 15.4 · ANALYSE D'UN ÉCHANGE IMAP

A partir de l'échange IMAP ci-dessus, identifier les éléments suivants :

- La commande pour visualiser l'enveloppe d'un e-mail spécifique
- La commande pour récupérer le corps complet d'un e-mail spécifique
- La liste des emails qu'a envoyé Alice à Bob

15.6 FTP

FTP (File Transfer Protocol) est un protocole de communication utilisé pour transférer des fichiers entre un client et un serveur sur un réseau TCP/IP. Il a deux modes de fonctionnements et utilise les ports 20 (mode actif uniquement) et 21 en TCP. Le port 21 est utilisé pour établir la connexion de contrôle, tandis que le port 20 est utilisé pour le transfert de données lorsqu'il fonctionne en mode actif. FTP est un protocole textuel basé sur des commandes et des réponses, similaire à SMTP et HTTP.

Il existe une version sécurisée de FTP appelée FTPS (FTP Secure), qui utilise SSL/TLS pour chiffrer les communications entre le client et le serveur. FTPS peut fonctionner sur les ports 989 et 990 (FTPS implicite) ou 20 et 21 (FTPS explicite).

ATTENTION

SFTP (SSH File Transfer Protocol) est un protocole différent de FTP et FTPS. Il fonctionne sur le port 22 en TCP et utilise le protocole SSH pour sécuriser les communications. SFTP est souvent préféré à FTPS pour

sa simplicité : un seul port est utilisé pour le contrôle et le transfert de données.

Parmi les clients FTP populaires, on peut citer FileZilla, WinSCP ou encore Cyberduck. Ces clients permettent de se connecter à un serveur FTP, de naviguer dans les répertoires, de télécharger et d'uploader des fichiers. FTP a pour fonctionnalité de base le transfert de fichiers, mais il peut également être utilisé pour créer, supprimer et renommer des fichiers et des répertoires sur le serveur distant.

15.6.1 Architecture et fonctionnement

FTP fonctionne avec un modèle client-serveur et utilise deux canaux de communication distincts : le canal de contrôle et le canal de données. Le canal de contrôle est utilisé pour envoyer des commandes et recevoir des réponses entre le client et le serveur, tandis que le canal de données est utilisé pour transférer les fichiers. Chaque canal utilise un port TCP différent (côté serveur) : le port 21 pour le canal de contrôle et le port 20 pour le canal de données (en mode actif). Chaque transfert de fichier nécessite l'établissement d'une nouvelle connexion TCP pour le canal de données (séquentiellement). La règle est la suivante : à un instant donné, maximum une connexion TCP de transfert de données par session de contrôle FTP (si un hôte souhaite transférer plusieurs fichiers en parallèle, mais il faut alors ouvrir plusieurs sessions FTP).

La Fig. 73 illustre l'architecture de FTP, montrant les deux canaux de communication et les ports utilisés pour le contrôle et le transfert de données.

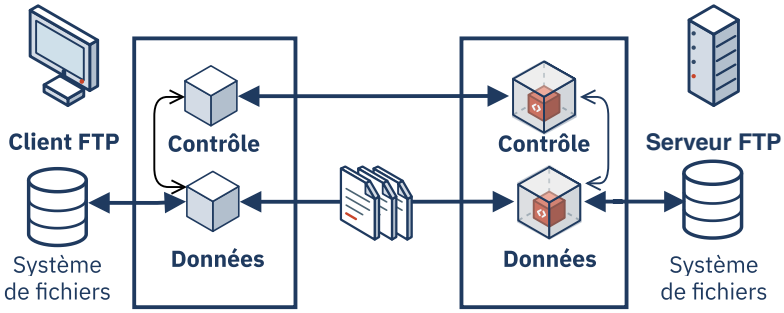


Fig. 73. – Architecture de FTP, montrant les deux canaux de communication (contrôle et données).

FTP a deux modes de fonctionnement pour le transfert de données : le mode actif et le mode passif. Le choix du mode dépend de la configuration du client et est choisi en fonction de la configuration du réseau et des pare-feu.

15.6.2 Mode actif

Dans le mode actif, c'est le serveur qui initie la connexion TCP pour le transfert d'un fichier. Tout d'abord, le client demande le mode actif au serveur en lui fournissant un port local « Y » sur lequel faire le three-way handshake. Ensuite, le serveur s'occupe d'ouvrir la connexion TCP de transfert de données, avec le port 20 côté serveur et le port « Y » donné par le client côté client. La Fig. 74 illustre ce processus.

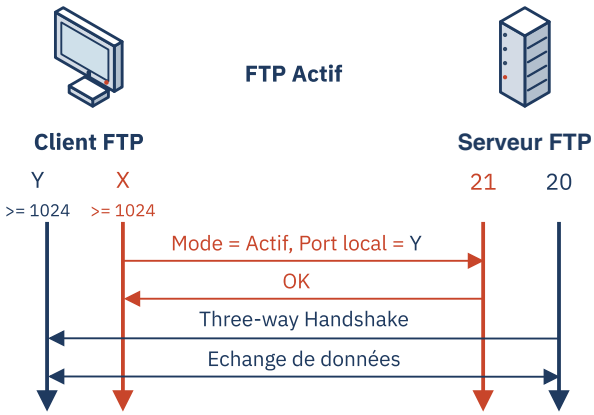


Fig. 74. – Illustration du mode actif de FTP. Le client utilise les ports X et Y >= 1024 et le serveur les ports 20 et 21.

Le défaut de ce mode est le risque qu'un pare-feu côté client empêche l'ouverture d'une connexion vers le client de la part du serveur.

15.6.3 Mode passif

Dans le mode passif, c'est le client qui initie une connexion TCP pour chaque transfert de fichier. Pour cela, le port 20 côté serveur ne suffit plus. Pour pouvoir gérer les N sessions FTP il faut pouvoir allouer plusieurs ports. Le processus est le suivant : le client demande le mode passif au serveur, qui lui répond avec un port >= 1024 sur lequel ouvrir la connexion de transfert de données. Le client initie la connexion, donc il y a moins de risque de pare-feu bloquant si la configuration a bien été faite côté serveur. La Fig. 75 illustre le fonctionnement du mode passif.

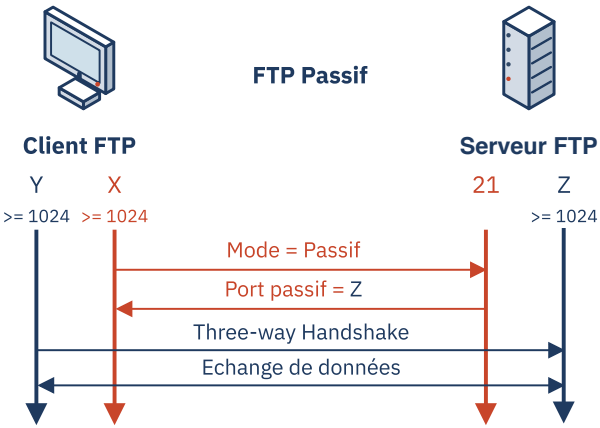


Fig. 75. – Illustration du mode passif de FTP. Le client utilise les ports X et Y >= 1024. Côté serveur le port 21 est utilisé pour le contrôle et le port Z >= 1024 pour le transfert de données.

15.6.4 Pourquoi pas le port 20 en mode passif ?

Une question qui revient souvent est : pourquoi le port 20 n'est pas utilisé pour le transfert de données en mode passif ? Après tout, le port 20 est connu et le client pourrait simplement se connecter à ce port pour le transfert de données. La raison tient dans la gestion des sessions FTP : il faut pouvoir différencier quelle connexion TCP de transfert de données correspond à quelle session FTP.

Lorsque le port 20 est utilisé comme port source (mode actif), cela ne pose pas de problème, car le client sait à quelle session FTP correspond le couple IP destination + port destination (port communiqué par le client au préalable). Le Fig. 76 illustre à quoi ressemble plusieurs connexions FTP actives simultanées sur un serveur FTP, avec le port 20 comme port source pour toutes les sessions. Le port TCP de destination côté client est différent pour chaque session FTP et connu.

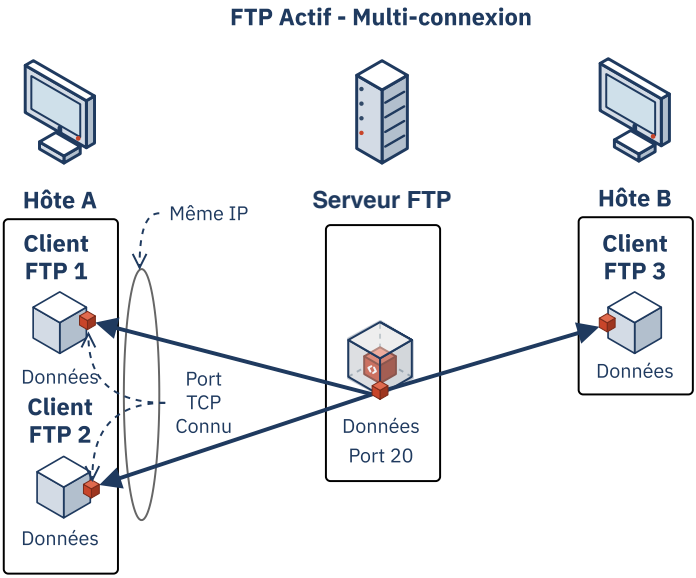


Fig. 76. – Illustration de plusieurs connexions FTP actives simultanées sur un serveur FTP, avec le port 20 comme port source pour toutes les sessions.

En mode passif, les rôles s'inversent : c'est désormais le serveur qui accepte la connexion de données, et c'est donc à lui de déterminer à quelle session FTP elle appartient. Or, s'il écoutait sur le port 20 pour toutes les sessions, il n'aurait aucun moyen fiable d'y parvenir. Le port source du client est attribué aléatoirement par son système d'exploitation, donc imprévisible pour le serveur. Quant à l'adresse IP source, elle ne permet pas davantage de trancher, puisque plusieurs sessions peuvent provenir d'une même adresse : plusieurs clients sur un même hôte, ou tout un réseau derrière un routeur NAT (voir le chapitre *NAT*). Le serveur doit donc avoir un port distinct par connexion de données. La Fig. 77 illustre à quoi ressemblent plusieurs connexions FTP passives simultanées sur un serveur FTP.

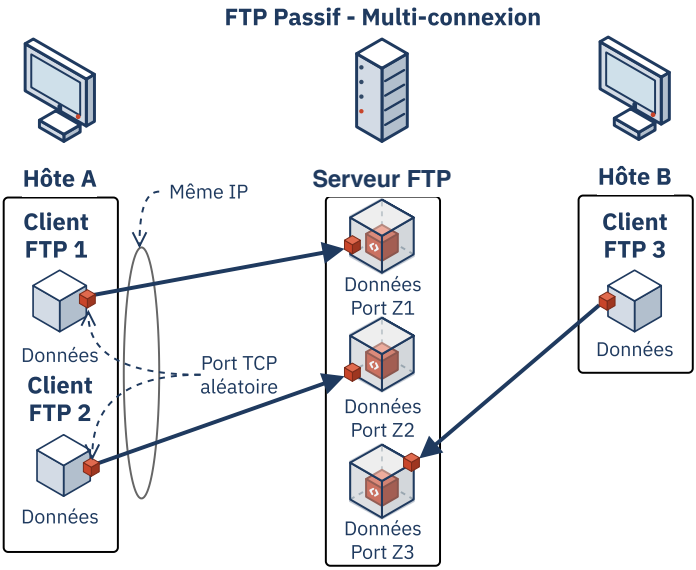


Fig. 77. – Illustration de plusieurs connexions FTP passives simultanées sur un serveur FTP, avec des ports de transfert de données différents pour chaque session.

EXERCICE 15.5 · IDENTIFICATION DU MODE FTP À PARTIR D'UNE CAPTURE DE TRAFIC

Un administrateur réseau capture le trafic entre un client FTP (192.168.1.50) et un serveur FTP (203.0.113.10) lors de deux transferts de fichiers distincts.

Transfert 1 : après l'établissement de la connexion de contrôle sur le port 21, une nouvelle connexion TCP est établie depuis 203.0.113.10:20 vers 192.168.1.50:52001.

Transfert 2 : après l'établissement de la connexion de contrôle sur le port 21, une nouvelle connexion TCP est établie depuis 192.168.1.50:52011 vers 203.0.113.10:50123.

1. Quel mode FTP (actif ou passif) est utilisé lors du transfert 1 ? Justifier en identifiant qui initie la connexion de données.
2. Quel mode FTP est utilisé lors du transfert 2 ? Justifier.

3. Lequel des deux transferts a le plus de chances d'échouer si le client est derrière un pare-feu qui bloque toute connexion entrante non sollicitée ? Pourquoi ?

15.6.5 Commandes et statuts

Comme SMTP, POP3, IMAP ou encore HTTP, FTP est un protocole textuel basé sur des commandes et des réponses. Le client FTP envoie des commandes au serveur FTP, qui répond avec des codes de statut pour indiquer le succès ou l'échec des opérations.

Les status sont organisés en classes, à la manière de HTTP, avec des codes à trois chiffres. Ci-dessous une liste non-exhaustive des codes de statut FTP les plus courants :

- 1xx : réponse positive provisoire, indiquant que l'action est en cours et que le client doit attendre une réponse finale.
- 2xx : réponse positive finale, indiquant que l'action a été complétée avec succès.
- 3xx : réponse positive intermédiaire, indiquant que le serveur attend une action supplémentaire de la part du client pour compléter l'action.
- 4xx : réponse négative provisoire, indiquant qu'une erreur temporaire s'est produite et que le client peut réessayer plus tard.
- 5xx : réponse négative finale, indiquant qu'une erreur s'est produite et que l'action n'a pas pu être complétée.
- 6xx : réponse « protégée », cas spécial défini par le RFC 2228. La réponse doit être décodée puis se trouve ensuite dans les catégories 1xx à 5xx.

Les commandes FTP sont séparées en deux catégories : les commandes de contrôle et les commandes de transfert de données. Les commandes de contrôle et de données sont toutes envoyées sur le canal de contrôle, mais le résultat des commandes de transfert de données est envoyé sur le canal de données. Les commandes de contrôle incluent :

- `USER <nom>` : pour spécifier le nom d'utilisateur pour se connecter au serveur FTP.
- `PASS <mot_de_passe>` : pour s'authentifier auprès du serveur FTP.
- `QUIT` : pour terminer la session FTP.
- `PORT <adresse_IP>,<port>` : demande le port actif pour le transfert de données, avec l'adresse IP et le port du client.

- `PASV` : demande le mode passif pour le transfert de données, le serveur répond avec un port >= 1024 sur lequel le client doit se connecter pour le transfert de données.
- `TYPE A` : spécifie le mode ASCII pour le transfert de fichiers texte.
- `TYPE I` : spécifie le mode binaire pour le transfert de fichiers binaires.
- `CWD`, `PWD`, `MKD`, `RMD` : pour changer de répertoire, afficher le répertoire courant, créer un répertoire et supprimer un répertoire, respectivement.

Les commandes de transfert de données incluent :

- `RETR <nom_fichier>` : pour récupérer un fichier depuis le serveur FTP (transfert serveur vers client).
- `STOR <nom_fichier>` : pour envoyer un fichier vers le serveur FTP (transfert client vers serveur).
- `LIST` : pour lister les fichiers et répertoires dans le répertoire courant du serveur FTP.

15.6.6 Exemple d'échange FTP

Voici un exemple d'échange FTP entre un client et un serveur, illustrant le processus de connexion, d'authentification et de transfert de fichiers :

Session de contrôle (port 21) :

```
220 Serveur FTP de demonstration
USER alice
331 Please specify the password.
PASS alice
230 Login successful.
PWD
257 "/" is the current directory
PASV
227 Entering Passive Mode (172,28,0,10,195,80).
LIST
150 Here comes the directory listing.
226 Directory send OK.
PASV
227 Entering Passive Mode (172,28,0,10,195,86).
RETR README.txt
150 Opening BINARY mode data connection for README.txt (43
bytes).
226 Transfer complete.
```

Session de données 1 :

```
-rw-r--r--    1 1000     1000          43 Jul 22 14:47
README.txt
drwxr-xr-x    2 1000     1000        4096 Jul 22 14:47
archives
-rw-r--r--    1 1000     1000      248310 Jul 22 14:47
rapport.bin
```

Session de données 2 :

```
Ceci est le contenu du fichier README.txt.
```

EXERCICE 15.6 · ANALYSE D'UN ÉCHANGE FTP

A partir de l'exemple d'échange FTP ci-dessus, identifier les éléments suivants :

- Quel mode FTP est utilisé ?
- Le port utilisé pour la première session de données
- Le port utilisé pour la deuxième session de données
- La taille en octets du fichier README.txt transféré
- Quel est le format utilisé pour lister les fichiers et répertoires dans le répertoire courant du serveur FTP (commande LIST)

CHAPITRE 16

NAT

OBJECTIFS DU CHAPITRE

À l'issue de ce chapitre, vous saurez :

- définir le concept de NAT (Network Address Translation) et son rôle dans la traduction des adresses IP entre un réseau privé et un réseau public
- expliquer les différents types de NAT (NAT statique, NAT dynamique, PAT) et leurs caractéristiques
- expliquer les avantages et les inconvénients de l'utilisation du NAT dans les réseaux informatiques

Ce chapitre introduit le concept de NAT (Network Address Translation), qui est une technique utilisée pour traduire les adresses IP entre un réseau privé et un réseau public. Le NAT est couramment utilisé dans les réseaux domestiques et d'entreprise pour permettre à plusieurs appareils de partager une seule adresse IP publique.

16.1 Introduction

Une adresse IP privée ne peut pas être routée sur Internet. Pour qu'un hôte dans un réseau privé puisse communiquer avec des hôtes sur Internet, il doit utiliser une adresse IP publique. Or, les adresses IP publiques sont limitées et coûteuses, il n'est pas possible dans la majorité des cas d'attribuer une adresse IP publique à chaque hôte d'un réseau privé. Le NAT (Network Address Translation) permet de résoudre ce problème en traduisant les adresses IP privées en adresses IP publiques et vice versa.

Il existe plusieurs types de NAT, chacun ayant ses propres caractéristiques et utilisations. Les principaux types de NAT sont le NAT statique, le NAT dynamique et le PAT (Port Address Translation).

Le NAT utilise une table de traduction pour mapper les adresses IP privées aux adresses IP publiques. Cette table peut être statique ou dynamique, selon le type de NAT utilisé.

16.2 NAT statique

Le NAT statique possède une table de traduction fixe, où chaque adresse IP privée est associée à une adresse IP publique spécifique. Cela signifie que chaque hôte dans le réseau privé a une correspondance directe avec une adresse IP publique. Le NAT statique est souvent utilisé pour les serveurs qui doivent être accessibles depuis Internet, car il permet de garantir que l'adresse IP publique reste constante. C'est un type de NAT simple à configurer mais coûteux en termes d'adresses IP publiques, car chaque hôte nécessite une adresse IP publique dédiée. La Fig. 78 illustre le fonctionnement du NAT statique.

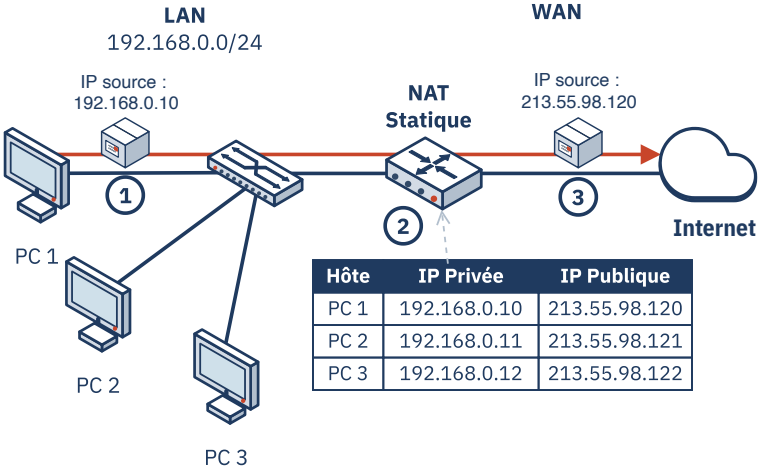


Fig. 78. – Illustration du fonctionnement du NAT statique. Dans cet exemple, le PC 1 envoie des données vers internet (1), l'adresse IP privée du PC 1 est définie comme source dans le paquet. Le routeur NAT traduit l'adresse IP privée en adresse IP publique (2) et envoie le paquet vers internet. L'adresse IP source est changée pour l'adresse IP publique assignée au PC 1 (3). Le fonctionnement est le même dans l'autre sens.

16.3 NAT dynamique

Le NAT dynamique utilise une table de traduction qui est remplie dynamiquement à mesure que les hôtes du réseau privé établissent des connexions vers Internet. Contrairement au NAT statique, le NAT dynamique n'associe pas une adresse IP privée à une adresse IP publique spécifique. Au lieu de cela, il attribue une adresse IP publique disponible à un hôte privé lorsqu'il initie une connexion vers Internet. L'IP publique est choisie dans un pool d'adresses IP publiques disponibles, et est prêtée à l'hôte privé un temps limité. La Fig. 79 illustre le fonctionnement du NAT dynamique.

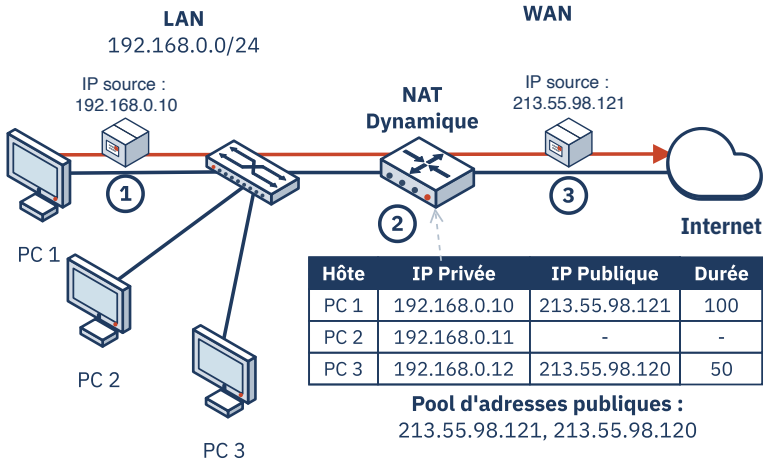
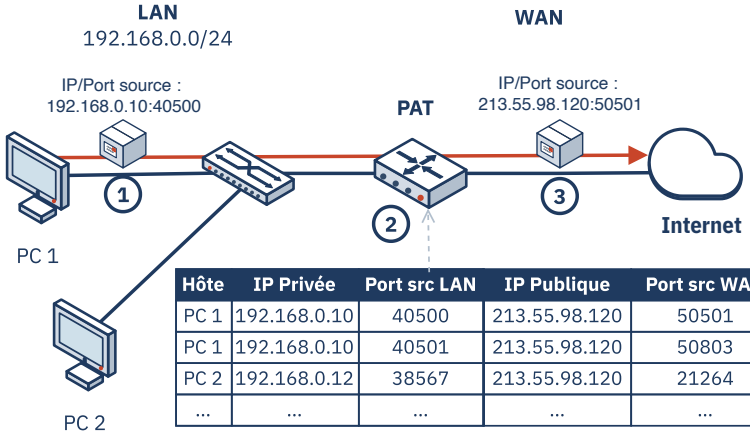


Fig. 79. – Illustration du fonctionnement du NAT dynamique. Dans cet exemple, le PC 1 envoie des données vers internet (1), l'adresse IP privée du PC 1 est définie comme source dans le paquet. Le routeur NAT attribue une adresse IP publique disponible au PC 1 (2) et envoie le paquet vers internet. L'adresse IP source est changée pour l'adresse IP publique assignée au PC 1 (3). Chaque allocation d'adresse IP publique a une durée d'inactivité limitée, exprimée par exemple en secondes.

Le NAT dynamique est plus efficace en termes d'utilisation des adresses IP publiques, mais il peut poser des problèmes pour les applications qui nécessitent une adresse IP publique constante et peut limiter le nombre d'hôtes qui ont accès à Internet en simultané si le pool d'adresses IP publiques est épuisé.

16.4 PAT

Le PAT (Port Address Translation), également connu sous le nom de NAT surchargé, est une forme de NAT dynamique qui permet à plusieurs hôtes d'un réseau privé de partager une seule adresse IP publique en utilisant des numéros de port différents pour chaque connexion. C'est le type de NAT le plus couramment utilisé dans les réseaux domestiques et d'entreprise. Le principe est le suivant : non seulement l'adresse IP privée est traduite en adresse IP publique, mais le numéro de port source de la connexion est également modifié pour permettre à plusieurs connexions de partager la même adresse IP publique. La Fig. 80 illustre le fonctionnement du PAT.



Hôte	IP Privée	Port src LAN	IP Publique	Port src WAN
PC 1	192.168.0.10	40500	213.55.98.120	50501
PC 1	192.168.0.10	40501	213.55.98.120	50803
PC 2	192.168.0.12	38567	213.55.98.120	21264
...	...	...	...	...

Fig. 80. – Illustration du fonctionnement du PAT. Dans cet exemple, le PC 1 envoie des données vers internet (1), l'adresse IP privée du PC 1 est définie comme source dans le paquet et le port source est défini par le système d'exploitation. Le routeur PAT traduit l'adresse IP privée par l'adresse IP publique (2) et modifie le port source pour un port disponible sur l'adresse IP publique. Le paquet est ensuite envoyé vers internet (3).

Le PAT est efficace en termes d'utilisation des adresses IP publiques et permet à un grand nombre d'hôtes de se connecter à Internet simultanément. Ses limitations principales sont :

- Le nombre de connexion simultanées est limité par le nombre de ports disponibles (64512 ports TCP/UDP, car les ports utilisés sont après les well-known ports, soit 1024 à 65535). Si le nombre de connexions simul-

tanées dépasse ce nombre, certaines connexions échoueront, rendant la connexion instable.
- Certaines applications nécessitant des connexions entrantes (comme le FTP en mode actif) peuvent rencontrer des problèmes avec le PAT, car elles nécessitent que le serveur puisse initier une connexion vers l'hôte privé, ce qui n'est pas possible si l'adresse IP publique est partagée entre plusieurs hôtes.

De plus, le PAT a pour effet de bord de renforcer la sécurité du réseau privé, car les hôtes privés ne sont pas directement accessibles depuis Internet. Pour héberger un service accessible depuis Internet, il est nécessaire de configurer une redirection de port sur le routeur NAT pour permettre aux connexions entrantes d'atteindre l'hôte privé approprié. Une redirection de port est une configuration qui permet de rediriger les connexions entrantes sur un port spécifique de l'adresse IP publique vers un port spécifique d'une adresse IP privée dans le réseau local.

16.5 Résumé

Le Tableau 23 résume les différents types de NAT, leurs caractéristiques, avantages et inconvénients.

Critère	NAT statique	NAT dynamique	PAT
Correspondance IP	1:1, chaque hôte privé a une IP publique dédiée	1:1, mais l'IP publique est attribuée dynamiquement	Plusieurs hôtes privés partagent une seule IP publique
Connexions entrantes possibles ?	Oui, chaque hôte a une IP publique dédiée	Oui, mais l'IP publique peut changer donc difficile	Non, sauf si une redirection de port est configurée
Economie d'adresses IP publiques	Non, chaque hôte a une IP publique dédiée	Oui, en fonction du pool d'IP publiques	Oui, économie maximale

Critère	NAT statique	NAT dynamique	PAT
Complexité de configuration	Simple, mais nécessite une IP publique par hôte	Plus complexe, nécessite un pool d'IP publiques	Complexe, nécessite la gestion des ports et des redirections
Avantages	Connexion stable, chaque hôte a une IP publique dédiée	Moins coûteux en IP publique que le NAT statique	Économie maximale d'IP publiques, permet à de nombreux hôtes de se connecter simultanément
Inconvénients	Coût élevé en IP publiques, pas d'économie	Peut poser des problèmes pour les applications nécessitant une IP publique constante	Limité par le nombre de ports disponibles, certaines applications peuvent rencontrer des problèmes

Aujourd'hui, la majorité des réseaux utilisent le PAT pour permettre à de nombreux hôtes privés de se connecter à Internet en utilisant une seule adresse IP publique. Cette technologie permet aux FAI (Fournisseurs d'Accès à Internet) de gérer efficacement les adresses IP publiques limitées et de fournir un accès Internet à un grand nombre d'utilisateurs.

EXERCICE 16.1 · NAT ET IPV6

Qu'en est-il d'IPv6 ? Le NAT est-il nécessaire avec IPv6 ?

Expliquez pourquoi ou pourquoi pas.

EXERCICE 16.2 · FTP ET PAT

En sachant que la majorité des réseaux utilisent le PAT, quel mode FTP (actif ou passif) est le plus adapté pour la majorité des utilisations ? Expliquez pourquoi.

ANNEXE

Corrigés des exercices

Solutions de tous les exercices, classées par chapitre.

Chapitre 1 · Introduction à la téléinformatique

Exercice 1.1 · Chronologie de la téléinformatique

1. Télégraphe électrique (1837)
2. ARPANET (1969)
3. Adoption de TCP/IP (1983)
4. Le Web (1991)
5. iPhone (2007)

Exercice 1.2 · Nommer des périphériques

- Terminaux:
  - Ordinateur
  - Smartphone
  - Console de jeu
  - Smartwatch
  - Tablette
  - Caméra IP
  - Imprimante réseau
  - Capteur IoT
- Intermédiaires:
  - Routeur
  - Switch
  - Pare-feu (Firewall)
  - Point d'accès Wi-Fi
- Et bien d'autres !

Exercice 1.3 · Identifier les éléments d'un réseau

- Périphériques terminaux : la tablette d'Alice, le serveur de streaming
- Périphériques intermédiaires : le routeur de la maison (et tous les équipements de l'opérateur et d'Internet entre les deux)

- Médiums : les ondes radio (Wi-Fi) entre la tablette et le routeur, la fibre optique vers Internet
- Message : le flux vidéo

Exercice 1.4 · Fiabilité, débit et latence selon l'application

1. Pour un email, la **fiabilité** est primordiale : il est essentiel que le message arrive intact et complet.
2. Pour le streaming vidéo, le **débit** est crucial : une bande passante suffisante est nécessaire pour assurer une lecture fluide sans interruptions. La latence est également importante, mais moins critique que le débit pour ce type d'application, sauf si l'on parle de streaming en direct, où la latence peut affecter l'expérience utilisateur.
3. Pour les jeux en ligne, la **latence** est l'aspect le plus important : une faible latence (communément appelée « ping » dans le jargon des joueurs) est essentielle pour une expérience de jeu réactive et compétitive. Un débit suffisant est également nécessaire, mais la latence est souvent le facteur déterminant pour la jouabilité.

Chapitre 2 · Signaux, topologies et classifications des réseaux

Exercice 2.1 · Identifier les éléments d'une chaîne de transmission

- Émetteur : Alice
- Canal : Le facteur et le transport de la lettre
- Récepteur : Bob
- Sources potentielles de bruit : Conditions météorologiques défavorables (pluie, vent), mauvaise écriture d'Alice, etc.

Exercice 2.2 · Analogique ou numérique ?

- A. Analogique, la voix humaine est un signal continu qui peut prendre une infinité de valeurs.
- B. Numérique, un fichier MP3 est un signal discret qui ne peut prendre qu'un nombre limité de valeurs. Dès le moment où il tient sur un ordinateur, il est numérique.
- C. Analogique, un disque vinyle est un support analogique qui reproduit un signal continu.
- D. Numérique, un SMS est un signal discret qui ne peut prendre qu'un nombre limité de valeurs.

- E. Analogique, la température affichée par un thermomètre à mercure est un signal continu qui peut prendre une infinité de valeurs.

Exercice 2.3 · Identifier le mode de transmission

- A. Unicast
- B. Multicast (seuls les abonnés qui regardent le flux reçoivent le signal)
- C. Broadcast
- D. Unicast

Exercice 2.4 · Nombre de liens d'un maillage complet

A. Chaque périphérique est relié aux 5 autres :

$$\frac{6 \times 5}{2} = 15 \text{ liens}$$

(on divise par 2 car chaque lien relie deux périphériques).

B.

$$\frac{n(n-1)}{2}$$

C. Le coût : le nombre de liens croît quadratiquement avec le nombre de périphériques, ce qui rend le maillage complet très coûteux à grande échelle. Par exemple, pour 100 périphériques, il faudrait déjà $\frac{100 \times 99}{2} = 4950$ liens.

Exercice 2.5 · Identifier la classe de réseau

A. LAN (Local Area Network) - Réseau domestique

B. MAN (Metropolitan Area Network) - Réseau d'entreprise dans une ville

C. PAN (Personal Area Network) - Connexion Bluetooth entre la manette et la console

D. GAN (Global Area Network) - Les satellites Galileo permettent la communication à l'échelle mondiale

Chapitre 3 · Binaire, hexadécimal et opérations logiques

Exercice 3.1 · Conversions

A. Comme 1 « B » = 8 « b » :

$$10 \text{ MB} = 10 \times 8 \text{ Mb} = 80 \text{ Mb}$$

B. Comme 1000 « kB » = 1 « MB » et 1000 « MB » = 1 « GB » :

$$1000\ \text{kB} = 1\ \text{MB} = \frac{1}{1000}\ \text{GB} = 0.001\ \text{GB}$$

C. Comme 1 « o » = 8 « b » :

$$250\ \text{o} = 250 \times 8\ \text{b} = 2000\ \text{b}$$

D. Comme 1 « Gb » = 1/8 « GB » :

$$1\ \text{Gb} = \frac{1}{8}\ \text{GB} = 0.125\ \text{GB}$$

E. Ici on passe d'une unité en base 10 (ko) à une unité en base 2 (KiB). Comme 1 « ko » = 1000 « o » et 1 « KiB » = 1024 « o » :

$$500\ \text{ko} = 500 \times 1000\ \text{o} = 500000\ \text{o}$$

$$500000\ \text{o} = \frac{500000}{1024}\ \text{KiB} \approx 488.28\ \text{KiB}$$

Exercice 3.2 · Calcul du temps de téléchargement: modem 56k

Tout d'abord, convertissons 5 Mo en bits. Comme 1 Mo = 8 Mb (mégaoctets en mégabits) :

$$5\ \text{Mo} = 5 \times 8\ \text{Mb} = 40\ \text{Mb}$$

Ensuite, convertissons les mégabits en kilobits :

$$40\ \text{Mb} = 40 \times 1000\ \text{kb} = 40000\ \text{kb}$$

Maintenant, avec un débit de 56 kbps, le temps nécessaire pour télécharger le fichier est :

$$t = \frac{40000\ \text{kb}}{56\ \text{kb/s}} \approx 714.29\ \text{s}$$

Convertissons les secondes en minutes :

$$t \approx \frac{714.29}{60} \approx 11.9\ \text{minutes}$$

Donc, il faudrait environ 12 minutes pour télécharger un PDF de 5 Mo à un débit de 56 kbps ! Rien que ça.

Exercice 3.3 · Calcul du temps de téléchargement: fibre optique

Tout d'abord, convertissons 5 Mo en bits. Comme 1 Mo = 8 Mb (mégaoctets en mégabits) :

$$5\ \text{Mo} = 5 \times 8\ \text{Mb} = 40\ \text{Mb}$$

Ensuite, convertissons les mégabits en gigabits :

$$40 \text{ Mb} = \frac{40}{1000} \text{ Gb} = 0.04 \text{ Gb}$$

Maintenant, avec un débit de 1 Gbps, le temps nécessaire pour télécharger le fichier est :

$$t = \frac{0.04 \text{ Gb}}{1 \text{ Gb/s}} = 0.04 \text{ s}$$

Donc, il faudrait environ 0.04 secondes pour télécharger un PDF de 5 Mo à un débit de 1 Gbps ! C'est légèrement plus rapide que le modem 56k, n'est-ce pas ?

Exercice 3.4 · Conversion de la base 2 à la base 10

A. $1011_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 0 + 2 + 1 = 11_{10}$

B. $1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = 13_{10}$

C. $10010_2 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 16 + 0 + 0 + 2 + 0 = 18_{10}$

D. $111111_2 = 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 32 + 16 + 8 + 4 + 2 + 1 = 63_{10}$

Exercice 3.5 · Valeurs maximales sur n bits

A. Sur n bits, la plus grande valeur représentable est $2^n - 1$: soit $2^8 - 1 = 255$ sur 8 bits et $2^{16} - 1 = 65535$ sur 16 bits.

B. Avec 9 bits, on peut représenter au maximum $2^9 - 1 = 511 < 1000$. Avec 10 bits, on monte à $2^{10} - 1 = 1023 \geq 1000$. Il faut donc 10 bits.

Exercice 3.6 · Conversion de la base 16 aux bases 10 et 2

A. $1\text{A}_{16} = 1 \times 16^1 + A \times 16^0 = 1 \times 16 + 10 \times 1 = 16 + 10 = 26_{10}$ En binaire : $1\text{A}_{16} = 00011010_2$

B. $\text{FF}_{16} = F \times 16^1 + F \times 16^0 = 15 \times 16 + 15 \times 1 = 240 + 15 = 255_{10}$ En binaire : $\text{FF}_{16} = 11111111_2$

C. $2\text{B}_{16} = 2 \times 16^1 + B \times 16^0 = 2 \times 16 + 11 \times 1 = 32 + 11 = 43_{10}$ En binaire : $2\text{B}_{16} = 00101011_2$

D. $\text{C3}_{16} = C \times 16^1 + 3 \times 16^0 = 12 \times 16 + 3 \times 1 = 192 + 3 = 195_{10}$ En binaire : $\text{C3}_{16} = 11000011_2$

Exercice 3.7 · Conversion de la base 10 aux bases 2 et 16

A. $100 = 64 + 32 + 4$, donc $100_{10} = 1100100_2$

B. $173 = 128 + 32 + 8 + 4 + 1$, donc $173_{10} = 10101101_2$

C. $255 = 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1$, donc $255_{10} = 11111111_2$ (la valeur maximale sur 8 bits !)

D. $300 = 1 \times 256 + 2 \times 16 + 12$, donc $300_{10} = 12\mathrm{C}_{16}$

E. $4096 = 16^3$, donc $4096_{10} = 1000_{16}$

Exercice 3.8 · Opérateurs logiques fondamentaux

A. $0 \text{ AND } 1 = 0$

B. $1 \text{ XOR } 0 = 1$

C. D'abord, calculons les deux parties de l'expression :

- $0 \text{ AND } 1 = 0$
- $1 \text{ XOR } 0 = 1$

Ensuite, nous combinons les résultats avec l'opérateur OR :

- $0 \text{ OR } 1 = 1$

Le résultat final est donc 1.

D. D'abord, calculons la partie entre parenthèses :

- $0 \text{ OR } 1 = 1$

Ensuite, nous combinons le résultat avec l'opérateur AND :

- $1 \text{ AND } 1 = 1$

Le résultat final est donc 1.

E. D'abord, calculons les deux parties de l'expression :

- $1 \text{ XOR } 1 = 0$
- $0 \text{ AND } 1 = 0$

Ensuite, nous combinons les résultats avec l'opérateur OR :

- $0 \text{ OR } 0 = 0$

Le résultat final est donc 0.

Exercice 3.9 · Opérateurs logiques sur des mots binaires

A.

```
  1 0 1 0
AND 1 1 0 0
  ---------
  1 0 0 0
```

B.

$$\begin{array}{rcccc} & 1 & 0 & 1 & 0 \\ \textbf{XOR} & 0 & 1 & 1 & 0 \\ \hline & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{array}$$

C. D'abord, calculons la partie entre parenthèses :

- $A \text{ OR } B = 1110$

$$\begin{array}{rcccc} & 1 & 0 & 1 & 0 \\ \textbf{OR} & 1 & 1 & 0 & 0 \\ \hline & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} \end{array}$$

Ensuite, nous combinons le résultat avec l'opérateur AND :

- $1110 \text{ AND } C = 0110$

$$\begin{array}{rcccc} & 1 & 1 & 1 & 0 \\ \textbf{AND} & 0 & 1 & 1 & 0 \\ \hline & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \end{array}$$

Le résultat final est donc 0110_2.

D. D'abord, calculons les deux parties de l'expression :

- $A \text{ XOR } B = 0110$

$$\begin{array}{rcccc} & 1 & 0 & 1 & 0 \\ \textbf{XOR} & 1 & 1 & 0 & 0 \\ \hline & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \end{array}$$

- $B \text{ AND } C = 0100$

$$\begin{array}{rcccc} & 1 & 1 & 0 & 0 \\ \textbf{AND} & 0 & 1 & 1 & 0 \\ \hline & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{array}$$

Ensuite, nous combinons les résultats avec l'opérateur OR :

- $0110 \text{ OR } 0100 = 0110$

$$\begin{array}{rcccc} & 0 & 1 & 1 & 0 \\ \textbf{OR} & 0 & 1 & 0 & 0 \\ \hline & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \end{array}$$

Le résultat final est donc 0110_2.

E. D'abord, calculons les deux parties de l'expression :

- $A \text{ AND } C = 0010$

```
  1 0 1 0
AND 0 1 1 0
  ---------
  0 0 1 0
```

- $B \text{ OR } C = 1110$

```
  1 1 0 0
OR 0 1 1 0
  ---------
  1 1 1 0
```

Ensuite, nous combinons les résultats avec l'opérateur XOR :

- $0010 \text{ XOR } 1110 = 1100$

```
  0 0 1 0
XOR 1 1 1 0
  ---------
  1 1 0 0
```

Le résultat final est donc 1100_2.

Exercice 3.10 · Décodage d'un message binaire en ASCII

B (0x42), r (0x72), a (0x61), v (0x76), o (0x6F), ! (0x21)

Bravo!

Exercice 3.11 · Encodage d'un message en ASCII

- O (79, 0x4F) : 01001111
- k (107, 0x6B) : 01101011
- ! (33, 0x21) : 00100001

Soit : 01001111 01101011 00100001

Chapitre 4 · Protocoles et modèles en couches

Exercice 4.1 · Associer les éléments aux couches OSI

- A. Couche 7 (application)
- B. Couche 2 (liaison de données)
- C. Couche 4 (transport)
- D. Couche 1 (physique)
- E. Couche 3 (réseau)

- F. Couche 6 (présentation)
- G. Couche 4 (transport)

Exercice 4.2 · Ordre d'encapsulation

1. Message (couches 7 à 5)
2. Segment (couche 4)
3. Paquet (couche 3)
4. Trame (couche 2)
5. Bits (couche 1)

À la réception, l'ordre est inversé (décapsulation).

Chapitre 5 · Matériel

Exercice 5.1 · Choisir le bon équipement

- A. Switch
- B. Routeur
- C. Pare-feu
- D. Hub (solution historique ; aujourd'hui, un switch d'entrée de gamme fait mieux pour un prix comparable)

Exercice 5.2 · Équipements et couches OSI

- Hub : couche 1
- Switch : couches 1 et 2
- Routeur : couches 1 à 3
- Pare-feu simple : couches 1 à 4 (les pare-feu applicatifs montent jusqu'à la couche 7)

Exercice 5.3 · Choisir le bon médium

A. Fibre optique : distance trop grande pour le cuivre (100 m au maximum), débit élevé, insensible aux interférences électromagnétiques.

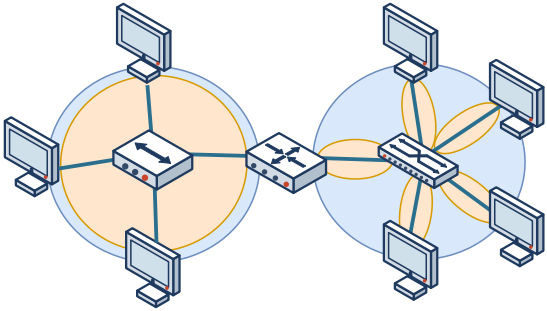
B. Paires torsadées : fiable, économique, débit largement suffisant.

C. LoRa : très faible débit, mais longue portée et faible consommation, idéal pour ce cas d'usage IoT.

D. 4G/5G : mobilité et couverture étendue.

Chapitre 6 · Accès au médium

Exercice 6.1 · Compter les domaines de collision et de diffusion



- Domaines de collision : 6. Chaque port actif du switch forme un domaine de collision : 4 pour les PC et 1 pour le lien switch-routeur, soit 5. Le hub ne sépare pas les domaines de collision : le hub, ses 3 PC et l'interface du routeur forment un seul domaine, soit 1 de plus.
- Domaines de diffusion : 2. Le routeur sépare les domaines de diffusion : un par interface.

Exercice 6.2 · Vrai ou faux : hub, switch et routeur

- A. Faux : un switch sépare les domaines de collision (un par port), mais tous ses ports appartiennent au même domaine de diffusion.
- B. Faux : un hub ne sépare ni les domaines de collision, ni les domaines de diffusion ; il répète le signal sur tous ses ports.
- C. Vrai : les routeurs ne transmettent pas les trames de diffusion d'une interface à l'autre.
- D. Vrai : le hub répète tout signal reçu sur tous ses autres ports, les collisions y sont donc possibles entre tous les périphériques.

Exercice 6.3 · Duplex et méthodes d'accès

Sur un lien Ethernet commuté moderne, chaque périphérique est relié à son propre port de switch par un lien point-à-point full-duplex : chaque sens de transmission dispose de son propre canal, sur lequel il n'y a qu'un seul émetteur possible. Le médium n'est donc plus partagé et aucune collision n'est possible : CSMA/CD devient inutile. En Wi-Fi au contraire, tous les périphériques émettent sur le même canal radio, qui est half-duplex et

partagé : il y a plusieurs émetteurs potentiels sur le même médium, une méthode d'accès reste donc indispensable.

Exercice 6.4 · Dérouler CSMA/CD pas à pas

1. B et C écoutent le médium (carrier sense) : il est occupé par A, elles attendent donc sans émettre.
2. Lorsque A termine sa transmission, le médium devient libre. B et C le détectent et commencent toutes deux à émettre en même temps.
3. Une collision se produit. B et C la détectent (collision detection) et interrompent immédiatement leur transmission.
4. B et C attendent chacune un temps aléatoire différent, déterminé par l'algorithme de Binary Exponential Backoff.
5. La station dont le temps d'attente est le plus court (par exemple B) réessaie la première : le médium est libre, sa trame est transmise avec succès.
6. Lorsque C réessaie à son tour, elle écoute le médium : si B émet encore, elle attend la fin de la transmission, puis émet sa trame.

Chapitre 7 · Ethernet et IP

Exercice 7.1

L'adresse MAC `78:ac:44:64:7e:4c` appartient à un périphérique fabriqué par Dell Inc. (OUI: `78:AC:44`).

Exercice 7.2 · Adresse MAC, IP ou invalide ?

A. Adresse MAC (format hexadécimal, 6 octets)

B. Adresse IP (format décimale pointée, 4 octets)

C. Adresse MAC (format hexadécimal, 6 octets)

D. Adresse IP invalide (les octets doivent être compris entre 0 et 255, puisque 8 bits sont utilisés pour chaque octet)

E. Adresse MAC invalide (les caractères doivent être compris entre 0-9 et A-F, puisqu'il s'agit d'un format hexadécimal)

Exercice 7.3 · Taille du masque de sous-réseau

A. /23

B. /32

C. Invalide

D. /0

E. Invalide (251 en base 2 donne `11111011`, ce qui n'est pas une suite continue de 1 suivie de 0)

Exercice 7.4 · Nombre d'hôtes disponibles

A. $2^{32-24} - 2 = 2^8 - 2$ = 254 hôtes

B. $2^{32-26} - 2 = 2^6 - 2$ = 62 hôtes

C. 255.255.240.0 correspond à un masque /20, donc $2^{32-20} - 2 = 2^{12} - 2$ = 4094 hôtes

D. $2^{32-30} - 2 = 2^2 - 2$ = 2 hôtes

E. 255.255.0.0 correspond à un masque /16, donc $2^{32-16} - 2 = 2^{16} - 2$ = 65 534 hôtes

Exercice 7.5 · Calcul de l'adresse de sous-réseau

A. Adresse de sous-réseau : `192.168.0.128`

B. Adresse de sous-réseau : `10.0.64.0`

C. Adresse de sous-réseau : `172.16.4.0`

D. Adresse de sous-réseau : `160.98.31.0`

Exercice 7.6 · Calcul de l'adresse de diffusion

A. Adresse de diffusion : `192.168.0.255`

B. Adresse de diffusion : `10.0.65.255`

C. Adresse de diffusion : `172.16.7.255`

D. Adresse de diffusion : `160.98.31.255`

Exercice 7.7

La réponse dépend de l'ordinateur et du réseau sur lequel il est connecté. Exemple avec un ordinateur Windows:

```
C:\> ipconfig
```

```
Windows IP Configuration

Ethernet adapter Ethernet0:

 Connection-specific DNS Suffix  . :
 IPv4 Address. . . . . . . . . . . : 192.168.34.89
 Subnet Mask . . . . . . . . . . . : 255.255.254.0
 Default Gateway . . . . . . . . . : 192.168.34.1
```

Dans cet exemple, l'adresse IP de l'ordinateur est `192.168.34.89/23`, qui appartient au sous-réseau privé `192.168.34.0/23`.

Exercice 7.8 · Plage privée ou publique ?

A. Privée. Broadcast : `192.168.10.255`, $2^8 - 2 = 254$ hôtes.

B. Privée. Broadcast : `10.130.127.255`, $2^{14} - 2 = 16\,382$ hôtes.

C. Publique. Broadcast : `160.98.21.255`, $2^9 - 2 = 510$ hôtes.

D. Privée. Broadcast : `172.23.255.255`, $2^{18} - 2 = 262\,142$ hôtes.

E. Publique. Broadcast : `172.32.19.255`, $2^{10} - 2 = 1022$ hôtes.

Exercice 7.9 · Raccourcissement des adresses IPv6

A. `2001:db8::1`

B. `fe80::202:b3ff:fe1e:8329`

C. `2001:db8::`

D. `2001:db8::1:0:0:1`

Chapitre 8 · ARP et ICMP

Exercice 8.1 · Consulter la table ARP sur votre ordinateur

- La table ARP contient les correspondances entre les adresses IP et les adresses MAC des périphériques du réseau local.
- L'adresse IP correspondant à l'adresse MAC `ff:ff:ff:ff:ff:ff` est l'adresse IP de diffusion (broadcast) du réseau local (`192.168.34.255`). C'est une entrée spéciale car elle est directement calculée à partir de l'adresse IP et du masque de sous-réseau.

Exemple sur un ordinateur Windows:

```
C:\> arp -a

Interface: 192.168.34.89 --- 0xe
  Internet Address      Physical Address      Type
  192.168.34.1          c0-f8-7f-ec-4d-49     dynamic
  192.168.34.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
```

Exercice 8.2 · Tester la connectivité avec ICMP et la commande ping

Exemple sur un ordinateur Linux / MacOS :

```
ping -c 4 www.heia-fr.ch
PING www.heia-fr.ch (160.98.8.45): 56 data bytes
64 bytes from 160.98.8.45: icmp_seq=0 ttl=252 time=1.187 ms
64 bytes from 160.98.8.45: icmp_seq=1 ttl=252 time=1.056 ms
64 bytes from 160.98.8.45: icmp_seq=2 ttl=252 time=0.869 ms
64 bytes from 160.98.8.45: icmp_seq=3 ttl=252 time=1.050 ms

--- www.heia-fr.ch ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.869/1.040/1.187/0.113 ms
```

A. L'adresse IP de www.heia-fr.ch est 160.98.8.45

B. 4 paquets ont été envoyés et 4 ont été reçus, donc aucun paquet n'a été perdu.

C. La latence moyenne pour atteindre www.heia-fr.ch est de 1.040 ms. (notée `avg` dans les statistiques de ping)

Chapitre 9 · Routage

Exercice 9.1 · Lecture du schéma réseau

A. Les sous-réseaux directement connectés au routeur `R2` sont :

- `192.168.2.0/24` (LAN B)
- `10.0.12.0/30` (liaison entre R1 et R2)
- `10.0.23.0/30` (liaison entre R2 et R3)

B. Le masque `/30` est utilisé entre les routeurs pour créer des sous-réseaux avec uniquement deux adresses IP utilisables (une pour chaque routeur). Cela permet d'économiser des adresses IP et de limiter le nombre d'hôtes sur ces liaisons point-à-point.

C. L'adresse IP de l'interface `eth1` du routeur `R3` est `10.0.23.2`. Sur le schéma, l'adresse est abrégée en `.23.2` pour simplifier la lecture (la première partie de l'adresse est implicite, étant donné le contexte du réseau).

D. `203.0.113.1` car `R1` possède le `.2` et le CIDR est `203.0.113.0/30` (donc 2 adresses utilisables : `.1` et `.2`).

Exercice 9.2 · Cheminement d'un paquet

Segment	MAC source	MAC destination	IP source	IP destination
1. LAN A (`PC-A1` → `R1`)	`MAC(PC-A1)`	`MAC(R1-eth2)`	192.168.1.10	192.168.3.10
2. Liaison `R1` → `R3`	`MAC(R1-eth1)`	`MAC(R3-eth0)`	192.168.1.10	192.168.3.10
3. LAN C (`R3` → `PC-C1`)	`MAC(R3-eth2)`	`MAC(PC-C1)`	192.168.1.10	192.168.3.10

Points clés :

- `PC-A1` détermine grâce à son masque que la destination est sur un autre réseau : il adresse donc la trame à sa passerelle par défaut, `R1` (interface `eth2`, 192.168.1.1), dont il obtient l'adresse MAC par ARP.
- `R1` consulte sa table de routage (Tableau 18) : la route vers `192.168.3.0/24` a pour passerelle `10.0.13.2` (R3) via l'interface `eth1`. Il ré-encapsule donc le paquet dans une nouvelle trame à destination de `MAC(R3-eth0)`.
- `R3` consulte sa table de routage : `192.168.3.0/24` est directement connecté à son interface `eth2`. Il remet donc la trame directement à `PC-C1` (adresse MAC obtenue par ARP si nécessaire).
- Les adresses IP source et destination **ne changent jamais** pendant tout le trajet ; seules les adresses MAC (source **et** destination) sont réécrites à chaque saut.

Exercice 9.3 · Remplir les tables de routage

Table de routage de R1 :

Destination	Gateway	Interface	Protocole	Métrique
192.168.10.0/24	—	eth1	Connecté	0
10.1.12.0/30	—	eth0	Connecté	0

Destination	Gateway	Interface	Protocole	Mé-triqu
192.168.20.0/24	10.1.12.2 (R2)	eth0	Statique	—
192.168.30.0/24	10.1.12.2 (R2)	eth0	Statique	—
10.1.23.0/30	10.1.12.2 (R2)	eth0	Statique	—
0.0.0.0/0	10.1.12.2 (R2)	eth0	Statique	—

Table de routage de R2 :

Destination	Gateway	Interface	Protocole	Mé-triqu
192.168.20.0/24	—	eth2	Connecté	0
10.1.12.0/30	—	eth0	Connecté	0
10.1.23.0/30	—	eth1	Connecté	0
192.168.10.0/24	10.1.12.1 (R1)	eth0	Statique	—
192.168.30.0/24	10.1.23.2 (R3)	eth1	Statique	—
0.0.0.0/0	10.1.23.2 (R3)	eth1	Statique	—

Table de routage de R3 :

Destination	Gateway	Interface	Proto-cole	Mé-triqu
192.168.30.0/24	—	eth1	Connecté	0
10.1.23.0/30	—	eth0	Connecté	0
198.51.100.0/30	—	eth2	Connecté	0

Destination	Gateway	Interface	Proto-cole	Mé-triqu
192.168.20.0/24	10.1.23.1 (R2)	eth0	Statique	—
192.168.10.0/24	10.1.23.1 (R2)	eth0	Statique	—
10.1.12.0/30	10.1.23.1 (R2)	eth0	Statique	—
0.0.0.0/0	198.51.100.1 (FAI)	eth2	Statique	—

Points clés :

- La passerelle d'une route est toujours une adresse **directement joignable** par le routeur, c'est-à-dire le prochain saut sur un réseau connecté. Vu de R1, la route vers le LAN Z pointe donc vers `10.1.12.2` (R2), et non vers `10.1.23.2` (R3) : R1 n'a aucun moyen d'atteindre directement cette adresse.
- En routage manuel, l'administrateur doit penser à **chaque destination sur chaque routeur**. Oublier une route casse la connectivité — parfois dans un seul sens : si R3 n'a pas de route vers le LAN X, les paquets de `PC-X1` arrivent à `PC-Z1`, mais les réponses ne reviennent jamais.
- Les routes par défaut se suivent en chaîne vers le FAI : R1 pointe vers R2, R2 pointe vers R3, et R3 pointe vers le routeur du FAI (`198.51.100.1`, seule autre adresse utilisable du `/30`).
- La liaison `198.51.100.0/30` n'apparaît que dans la table de R3 (réseau directement connecté), et aucune route explicite vers ce réseau n'est configurée sur R1 et R2. Pourquoi ? D'abord, ce réseau ne contient aucun hôte final mais uniquement les deux interfaces du lien R3-FAI : on ne communique pas **vers** ce réseau, mais **à travers** lui. Ensuite, une telle route serait redondante : si un paquet lui est tout de même destiné (par exemple un `ping` vers l'interface du FAI), la route par défaut de R1 et R2 l'achemine déjà dans la bonne direction, une route explicite emprunterait exactement le même chemin.

Chapitre 10 · TCP et UDP

Exercice 10.1

- **IMAP** : 143 TCP (UDP réservé par l'IANA, mais non utilisé en pratique)
- **POP3** : 110 TCP (UDP réservé par l'IANA, mais non utilisé en pratique)
- **SMTP** : 25 TCP (UDP réservé par l'IANA, mais non utilisé en pratique)
- **DNS** : 53 TCP et UDP
- **NTP** : 123 UDP (peut utiliser TCP si explicitement configuré)
- **MQTT** : 1883 TCP (UDP réservé par l'IANA, mais non utilisé en pratique)

Remarques : pour la plupart des services, il existe une version sécurisée utilisant TLS, qui utilise généralement un port différent (par exemple, IMAPS sur le port 993 TCP, POP3S sur le port 995 TCP, SMTPS sur le port 465 TCP).

Exercice 10.2 · Contrôle de flux ou contrôle de congestion ?

- A. Contrôle de flux
- B. Contrôle de congestion
- C. Contrôle de flux
- D. Contrôle de congestion
- E. Contrôle de flux

Remarque : le contrôle de flux est une négociation entre l'émetteur et **le récepteur** (le protéger), tandis que le contrôle de congestion est une réaction à l'état du **réseau** entre les deux hôtes (protéger les équipements intermédiaires).

Exercice 10.3

A. UDP : le streaming vidéo en direct nécessite une faible latence et peut tolérer la perte de quelques paquets, ce qui rend UDP plus approprié.

B. TCP : le transfert d'une facture nécessite une transmission fiable et ordonnée des données, ce qui est assuré par TCP.

C. TCP : la synchronisation d'une base de données entre deux serveurs nécessite une transmission fiable et ordonnée des données, ce qui est assuré par TCP.

D. UDP : la synchronisation de l'emplacement de personnages dans un jeu vidéo multijoueur en ligne nécessite une faible latence et peut tolérer la perte de quelques paquets, ce qui rend UDP plus approprié.

E. TCP : l'envoi d'un message instantané entre deux utilisateurs sur une application de messagerie nécessite une transmission fiable et ordonnée des données, ce qui est assuré par TCP.

Chapitre 11 · Linux

Exercice 11.1 · Différences entre adduser/deluser et useradd/userdel

Les versions `adduser` et `deluser` sont propres à certaines distributions Linux (comme Debian et Ubuntu) et sont des scripts Perl qui fournissent une interface plus conviviale pour ajouter et supprimer des utilisateurs. De l'autre côté, `useradd` et `userdel` sont des commandes plus basiques et standardisées, disponibles sur la plupart des distributions Linux. Elles sont plus difficiles à utiliser et nécessitent de spécifier plus d'options pour configurer correctement un utilisateur.

Exercice 11.2 · Options de la commande ping

- `-c` : permet de spécifier le nombre de paquets à envoyer.
- `-i` : permet de spécifier l'intervalle de temps entre l'envoi de chaque paquet (en secondes).
- `-4` : force l'utilisation de l'IPv4.
- `-6` : force l'utilisation de l'IPv6.

Exercice 11.3 · Analyse de la sortie de la commande ip addr

- L'adresse IPv4 de l'interface `ens18` est `160.98.22.140/24` et l'adresse de diffusion est `160.98.22.255`.
- L'adresse IPv6 de l'interface `ens18` est `fe80::be24:11ff:fe7c:ab77/64`.
- L'interface `lo` est l'interface de loopback (boucle locale). Son adresse IPv4 `127.0.0.1` n'est pas routable sur Internet. Il s'agit d'une interface virtuelle utilisée pour la communication interne de l'ordinateur avec lui-même.

Exercice 11.4 · Analyse de la sortie de la commande traceroute

- Il y a 8 sauts nécessaires pour atteindre le site « example.com ». Les adresses IP des routeurs intermédiaires sont :
  1. 160.98.22.1
  2. 160.98.6.105
  3. 160.98.7.221
  4. 195.176.1.20
  5. 130.59.37.146
  6. 130.59.37.152

7. 192.65.185.228
8. 104.20.23.154

- L'astérisque (`*`) dans la dernière ligne indique que le routeur n'a pas répondu à la requête ICMP dans le délai imparti. Cela peut être dû à un pare-feu, à une configuration de sécurité ou à une surcharge du routeur.

Exercice 11.5 · Analyse de la sortie de la commande netstat

- Les ports TCP sont notés avec le statut « LISTEN » car le protocole TCP est orienté connexion, ce qui signifie qu'un serveur doit écouter les demandes de connexion entrantes. Les ports UDP, en revanche, sont sans connexion et n'ont pas besoin d'écouter activement, donc ils n'ont pas de statut « LISTEN ».
- Les services écoutent sur `0.0.0.0:*` ou `:::*` pour indiquer qu'ils acceptent les connexions entrantes sur toutes les interfaces réseau disponibles. `0.0.0.0` représente toutes les adresses IPv4 locales, tandis que `:::` représente toutes les adresses IPv6 locales.
- Un serveur web HTTP standard devrait écouter sur l'adresse IP `127.0.0.1` et le port `80` pour accepter les connexions entrantes depuis l'hôte lui-même uniquement.

Chapitre 12 · DHCP et DNS

Exercice 12.1

Un client peut refuser une offre (`DHCP DECLINE`) si l'adresse IP proposée est déjà utilisée par un autre périphérique sur le réseau, ce qui pourrait entraîner un conflit d'adresses IP. Le client peut détecter ce conflit en regardant dans sa table ARP ou en envoyant un message ARP pour vérifier si l'adresse IP est déjà attribuée à un autre périphérique. Si le client détecte que l'adresse IP est déjà utilisée, il envoie un message `DHCP DECLINE` au serveur pour refuser l'offre et demander une nouvelle configuration IP.

Un serveur peut refuser une demande (`DHCP NAK`) si le client tente de renouveler un bail avec une adresse IP qui n'est plus valide ou qui a été attribuée à un autre périphérique. Cela peut se produire si le serveur a réattribué l'adresse IP à un autre client après l'expiration du bail initial, ou si le client tente d'utiliser une adresse IP en dehors de la plage d'adresses configurée sur le serveur DHCP. Dans ce cas, le serveur envoie un message `DHCP NAK` pour informer le client que la demande est refusée et qu'il doit demander une nouvelle configuration IP.

Exercice 12.2

La commande `dig +trace` effectue une résolution itérative. Cela est dû au fait que la commande suit le cheminement de la requête DNS à travers les différents serveurs DNS, en commençant par les serveurs racine et en descendant dans la hiérarchie jusqu'à atteindre le serveur autoritaire pour le nom de domaine demandé. À chaque étape, `dig` reçoit des informations partielles sur les serveurs suivants à interroger, ce qui est caractéristique d'une résolution itérative.

Exercice 12.3 · Exercice : Fichier hosts et localhost

L'enregistrement lié à `localhost` dans le fichier `hosts` est généralement de type A (Address) et associe le nom d'hôte `localhost` à l'adresse IP `127.0.0.1` (l'adresse de loop-back). Cet enregistrement permet de donner le nom «localhost» pour désigner l'hôte local. En faisant un ping sur `localhost`, le système envoie des paquets ICMP à l'adresse IP `127.0.0.1`, donc sur l'interface virtuelle de loop-back de l'ordinateur lui-même.

Voici le contenu typique du fichier `hosts` sur un système Linux, contenant un enregistrement A (IPv4) et un enregistrement AAAA (IPv6) pour `localhost` :

```
127.0.0.1   localhost
::1         localhost ip6-localhost ip6-loopback
```

Chapitre 13 · HTTP

Exercice 13.1 · Validité d'URL

A. Valide

B. Valide

C. Non valide : le chemin contient un caractère `%` suivi d'un caractère non hexadécimal (`t`). Le caractère `%` doit être suivi de deux chiffres hexadécimaux pour représenter un caractère encodé.

D. Non valide : le chemin contient une espace non encodée. L'espace n'est pas un caractère autorisé dans une URL ; il doit être encodé en `%20` (soit `travaux%20pratiques`).

E. Valide : adresse IPv4 littérale avec port explicite.

F. Valide

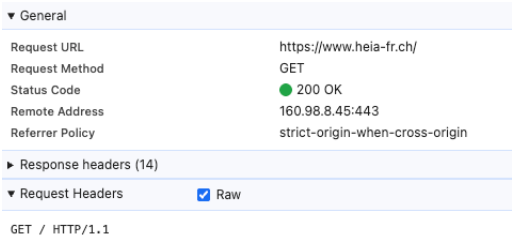
G. Valide : un chemin à plusieurs niveaux est tout à fait autorisé

H. Valide (cas spécial) : les segments de chemin d'accès peuvent être vides, ce qui est le cas ici. Le chemin contient plusieurs segments vides consécutifs, mais cela reste valide selon le RFC 3986. Il est d'ailleurs possible de tester cette URL dans un navigateur web, qui l'interprétera correctement.

Exercice 13.2

La première requête HTTP effectuée par le navigateur est celle qui est demandée à la base, soit l'obtention de la page d'accueil du site `https://www.heia-fr.ch`. La méthode HTTP utilisée est `GET`, car le navigateur demande la ressource (la page web) au serveur. Le code de statut de la réponse du serveur pour cette requête est `200 OK`. La version HTTP utilisée peut dépendre du navigateur et du serveur. Pour la trouver il faut afficher la requête en « raw » dans les outils de développement du navigateur.

Ci-dessous un exemple de ce que l'on peut observer dans les outils de développement du navigateur :



Exercice 13.3

La première fois que l'on clique sur les boutons de test des versions de HTTP, le navigateur doit établir une connexion avec le serveur et télécharger les ressources nécessaires pour effectuer le test. Lors de la deuxième fois, le navigateur peut utiliser le cache pour récupérer les ressources déjà téléchargées, ce qui réduit considérablement le temps nécessaire pour effectuer l'opération.

Chapitre 14 · Cryptographie, Telnet et SSH

Exercice 14.1

- Dans le premier cas, la confidentialité est garantie, car seul Bob peut déchiffrer le message avec sa clé privée. L'intégrité et l'authenticité ne sont pas garanties, car un attaquant pourrait envoyer le message à la place d'Alice et se faire passer pour elle.

- Dans le second cas, l'authenticité est garantie, car seul Bob peut chiffrer le message avec sa clé privée. L'intégrité est également garantie, car si le message est modifié en transit, Alice ne pourra pas le déchiffrer correctement avec la clé publique de Bob. La confidentialité n'est pas garantie, car n'importe qui ayant accès à la clé publique de Bob peut déchiffrer le message.

Exercice 14.2

SSH utilise un mécanisme de challenge-réponse basé sur la cryptographie asymétrique. Lorsqu'un utilisateur tente de se connecter à un serveur SSH avec une clé publique, le serveur génère un challenge (un message aléatoire) et l'envoie au client. Le client utilise sa clé privée pour signer ce challenge, créant ainsi une signature numérique. Le client renvoie ensuite la signature au serveur. Le serveur utilise la clé publique de l'utilisateur pour vérifier la signature. Si la signature est valide, cela prouve que le client possède la clé privée correspondante à la clé publique stockée sur le serveur, sans que la clé privée elle-même soit transmise.

Exercice 14.3

Ce message signifie que la clé publique du serveur a changé depuis la dernière connexion. SSH stocke la clé publique du serveur après la première connexion pour vérifier l'identité du serveur lors des connexions futures, afin de prévenir les attaques de type « man-in-the-middle ». C'est à dire une attaque où un attaquant intercepte la communication entre le client et le serveur, se faisant passer pour le serveur légitime.

Une raison légitime pour laquelle la clé publique du serveur pourrait changer est une mise à jour ou un remplacement du serveur, entraînant la génération d'une nouvelle paire de clés. Une raison inquiétante pourrait être qu'un attaquant a compromis le serveur et a remplacé sa clé publique par la sienne, ce qui pourrait permettre à l'attaquant d'intercepter les communications.

Chapitre 15 · Mail et FTP

Exercice 15.1 · Identification des rôles

1. Le client de messagerie qu'utilise Alice pour envoyer son email : MUA (Mail User Agent)
2. Le serveur mail @bob.com : MTA (Mail Transfer Agent) & MDA (Mail Delivery Agent)

3. Le serveur mail @alice.com : MTA (Mail Transfer Agent) (& MDA, mais pas utilisé dans cet exemple)
4. Le client de messagerie qu'utilise Bob pour récupérer son email : MUA (Mail User Agent)

Exercice 15.2 · Analyse d'un échange POP3

- L'adresse e-mail de l'expéditeur : `alice@labo.local`
- L'adresse e-mail du destinataire : `bob@labo.local`
- La taille en octets du message récupéré : 167
- Le mot de passe de l'utilisateur utilisé pour se connecter au serveur POP3 : `bob123`
- Le `HELO` utilisé par le client pour s'identifier auprès du serveur : `poste-alice`

Exercice 15.3 · Récupération et suppression des e-mails avec POP3

1. `USER <nom_utilisateur>` : pour spécifier le nom d'utilisateur
2. `PASS <mot_de_passe>` : pour spécifier le mot de passe
3. `LIST` : pour lister les e-mails disponibles sur le serveur
4. Pour chaque e-mail à récupérer :
   - `RETR <numéro>` : pour récupérer l'e-mail spécifique
   - `DELE <numéro>` : pour supprimer l'e-mail du serveur après récupération
5. `QUIT` : pour terminer la session POP3

Exercice 15.4 · Analyse d'un échange IMAP

- La commande pour visualiser l'enveloppe d'un e-mail spécifique : `FETCH <numéro> ENVELOPE`
- La commande pour récupérer le corps complet d'un e-mail spécifique : `FETCH <numéro> BODY[]`
- La liste des emails qu'a envoyé Alice à Bob : les numéros d'e-mails retournés par la commande `SEARCH FROM "alice@labo.local"`, qui sont : 4, 5, 19, 20, 22, 23, 24

Exercice 15.5 · Identification du mode FTP à partir d'une capture de trafic

1. Transfert 1 : mode **actif**. La connexion de données est initiée par le serveur (depuis son port 20) vers le client, ce qui est la caractéristique du mode actif.
2. Transfert 2 : mode **passif**. La connexion de données est initiée par le client vers le serveur, ce qui est la caractéristique du mode passif.
3. Le transfert 1 (mode actif), car c'est le serveur qui doit ouvrir une connexion entrante vers le client ; un pare-feu côté client bloquant les connexions non sollicitées empêchera cette connexion de s'établir. Le

transfert 2 (mode passif) n'est pas affecté puisque c'est le client qui initie la connexion.

Exercice 15.6 · Analyse d'un échange FTP

- Le mode FTP utilisé : le mode passif
- Le port utilisé pour la première session de données : 50000 (il est donné en deux parties dans la réponse du serveur à la commande PASV. Le port est donné en deux octets séparés en base 10, ici 195 et 80. La manière la plus simple de calculer le port est : $195 \times 256 + 80 = 50000$). On peut également transformer les deux octets en hexadécimal, les concaténer et convertir en décimal, ce qui donne le même résultat (195 = C3 et 80 = 50, donc C350 en hexadécimal = 50000 en décimal).
- Le port utilisé pour la deuxième session de données : 50006, même procédé que pour la première session ($195 \times 256 + 86 = 50006$).
- La taille en octets du fichier README.txt transféré : 43
- Il s'agit du format de liste de fichiers standard Unix. Ce format n'est pas standardisé, mais est généralement utilisé par les serveurs FTP. D'autres formats existent, comme le format Windows.

Chapitre 16 · NAT

Exercice 16.1 · NAT et IPv6

Avec IPv6, le NAT n'est généralement pas nécessaire. IPv6 offre un espace d'adressage beaucoup plus vaste que IPv4, ce qui permet à chaque appareil d'avoir une adresse IP unique et routable sur Internet. Ainsi, les problèmes de pénurie d'adresses IP qui ont conduit à l'utilisation du NAT avec IPv4 sont largement résolus avec IPv6.

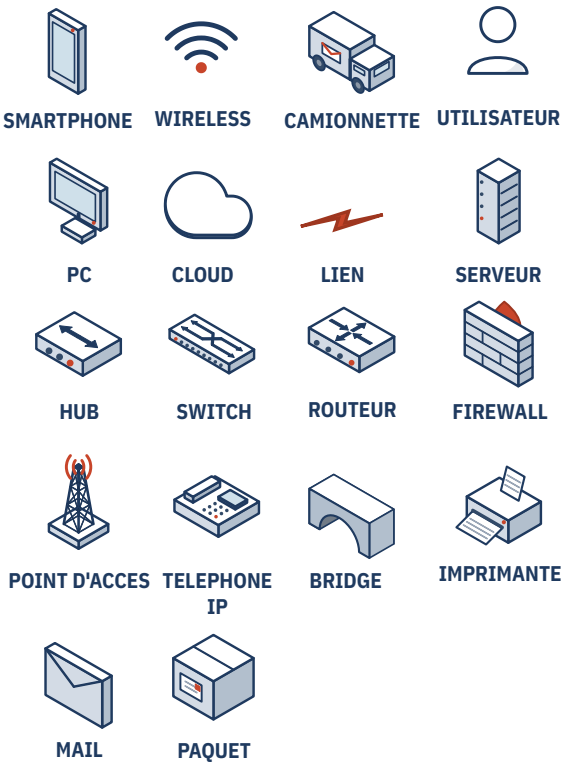
Il existe cependant des technologies de traduction d'adresses pour IPv6, comme le NAT64, qui permettent la communication entre les réseaux IPv6 et IPv4.

Exercice 16.2 · FTP et PAT

Le mode FTP passif est généralement le plus adapté pour la majorité des utilisations dans les réseaux utilisant le PAT. En mode passif, le client FTP initie toutes les connexions, y compris la connexion de données, ce qui permet de contourner les problèmes liés aux connexions entrantes que peuvent survenir avec le PAT.

ANNEXE

Symboles réseau



ANNEXE

Glossaire

Termes et acronymes employés dans l'ouvrage, classés par ordre alphabétique.

A

ARP Address Resolution Protocol : protocole utilisé pour résoudre les adresses IP en adresses MAC sur un réseau local.

ASCII American Standard Code for Information Interchange : encodage de caractères sur 7 bits, permettant 128 caractères.

B

Bash Un shell populaire pour les systèmes Linux, qui offre des fonctionnalités avancées pour l'interprétation des commandes et la programmation de scripts.

Binaire Système de numération en base 2, utilisant les symboles 0 et 1.

Bit Binary digit : unité fondamentale de l'information, valant 0 ou 1.

Bitwise Se dit d'une opération logique appliquée bit par bit sur des mots binaires.

Bridge Pont : commutateur à deux ports reliant deux segments de réseau.

Broadcast Mode de transmission d'un émetteur vers tous les hôtes du domaine de diffusion.

Bruit Perturbation qui altère le signal pendant sa transmission et peut provoquer des erreurs de communication.

C

Canal Support par lequel le signal transite entre l'émetteur et le récepteur.

CRC Cyclic Redundancy Check : un algorithme utilisé pour détecter les erreurs de transmission dans une trame Ethernet.

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance : protocole d'accès au média utilisé dans les réseaux sans fil pour gérer l'accès au support physique partagé et éviter les collisions de données.

CSMA/CD Carrier Sense Multiple Access with Collision Detection : protocole d'accès au média utilisé dans les réseaux Ethernet pour gérer l'accès au support physique partagé et détecter les collisions de données.

Câble coaxial Câble à conducteur central en cuivre entouré d'une tresse métallique, utilisé historiquement (10BASE2, 10BASE5, télévision par câble).

D

Débit Quantité d'information transmise par unité de temps, exprimée en bits par seconde.

Décapsulation Retrait, par chaque couche, de l'en-tête correspondant lors de la remontée du modèle en couches.

E

EMI ElectroMagnetic Interference : interférences électromagnétiques, auxquelles le cuivre est sensible, contrairement à la fibre optique.

En-tête (header) Informations de contrôle ajoutées par une couche devant les données ; la couche liaison de données ajoute aussi une queue (trailer).

Encapsulation Ajout, par chaque couche, d'un en-tête aux données reçues de la couche supérieure lors de la descente du modèle en couches.

EtherType Un champ de 2 octets dans l'en-tête d'une trame Ethernet de type II qui indique le protocole de la couche supérieure (L3, réseau) utilisé pour traiter les données contenues dans la trame.

F

FCS Frame Check Sequence : un champ de 4 octets dans une trame Ethernet utilisé pour détecter les erreurs de transmission.

Fibre monomode Fibre optique à un seul mode de propagation de la lumière, utilisée pour les longues distances.

Fibre multimode Fibre optique à plusieurs modes de propagation, utilisée pour les courtes distances (réseaux locaux, centres de données).

Fibre optique Médium transmettant les données par la lumière, offrant de très hauts débits et une très faible atténuation.

G

GAN Global Area Network : réseau à l'échelle mondiale, p. ex. réseaux satellitaires.

GNU Un projet de logiciels libres qui fournit de nombreux outils et utilitaires pour les systèmes d'exploitation de type Unix, souvent utilisés avec le noyau Linux.

H

Hexadécimal Système de numération en base 16, utilisant les symboles 0 à 9 et A à F.

Hub Concentrateur : équipement de couche 1 qui retransmet chaque signal reçu à tous les autres ports.

Hôte (host) Noeud d'un réseau, qu'il soit terminal (ordinateur, smartphone) ou intermédiaire (routeur, switch).

I

ICMP Internet Control Message Protocol : protocole de la couche réseau utilisé pour envoyer des messages de contrôle et de diagnostic, comme les messages d'erreur et les requêtes de ping.

ICMPv6 Internet Control Message Protocol version 6 : version d'ICMP utilisée dans les réseaux IPv6 pour envoyer des messages de contrôle et de diagnostic.

IEEE Institute of Electrical and Electronics Engineers : organisme de normalisation, notamment des réseaux locaux (famille 802.x).

IETF Internet Engineering Task Force : organisme qui normalise les protocoles d'Internet à travers les RFC.

IoT Internet of Things : Internet des objets, désigne l'ensemble des objets connectés à Internet et capables de communiquer entre eux ou avec des serveurs.

IP Internet Protocol : protocole de la couche réseau responsable de l'adressage logique et du routage des paquets de données sur Internet.

IPv4 Internet Protocol version 4 : version du protocole IP utilisant des adresses sur 32 bits, permettant d'adresser environ 4,3 milliards d'hôtes.

IPv6 Internet Protocol version 6 : version du protocole IP utilisant des adresses sur 128 bits, permettant d'adresser un nombre quasi infini d'hôtes et de résoudre le problème de pénurie d'adresses IPv4.

ISO International Organization for Standardization : organisme international de normalisation, à l'origine du modèle OSI.

L

LAN Local Area Network : réseau local à l'échelle d'un bâtiment ou d'un site.

Latence Temps nécessaire à un signal pour parcourir le canal de l'émetteur au récepteur, généralement exprimé en millisecondes.

Linux Un système d'exploitation de type Unix, open-source et basé sur le noyau Linux. Il est utilisé sur une variété de dispositifs, allant des serveurs aux ordinateurs personnels et aux appareils embarqués.

M

MAC Media Access Control : sous-couche de la couche liaison de données responsable de la gestion des adresses physiques et de l'accès au support physique partagé.

MAN Metropolitan Area Network : réseau métropolitain à l'échelle d'une ville ou d'un campus.

MDI Medium Dependent Interface : câblage d'interface Ethernet des équipements terminaux et des routeurs.

MDIX MDI crossover : câblage d'interface inversé, utilisé par les switches et les hubs ; l'auto-MDI/MDIX rend le choix du câble droit ou croisé transparent.

MTU Maximum Transmission Unit : la taille maximale d'un paquet qui peut être transmis sur un réseau. Pour Ethernet, la MTU est généralement de 1500 octets, ce qui correspond à la taille maximale d'un payload de trame Ethernet II.

Multicast Mode de transmission d'un émetteur vers un groupe de récepteurs.

Médium Support de transmission (câble, fibre optique, ondes radio) permettant la circulation des données entre les périphériques.

N

NDP Neighbor Discovery Protocol : protocole utilisé dans les réseaux IPv6 pour la découverte des voisins et la résolution d'adresses, remplaçant ARP.

Nibble Groupe de 4 bits, correspondant à un chiffre hexadécimal.

O

Octet Groupe de 8 bits, unité usuelle de quantification des données (byte en anglais).

OSI Open Systems Interconnection : modèle de référence en sept couches pour la communication entre systèmes informatiques.

OSPF Open Shortest Path First : protocole de routage de la couche réseau utilisé pour échanger des informations de routage entre les périphériques d'un réseau, basé sur l'algorithme du plus court chemin.

OUI Organizationally Unique Identifier : les trois premiers octets d'une adresse MAC, identifiant le fabricant.

P

PAN Personal Area Network : réseau personnel de très courte portée (quelques mètres), p. ex. Bluetooth.

Pare-feu Firewall : équipement de sécurité filtrant le trafic réseau selon des règles prédéfinies (adresses IP, ports, voire contenu applicatif).

Passerelle (gateway) Équipement d'interconnexion entre réseaux ; désigne couramment le routeur de sortie d'un réseau local.

PDU Protocol Data Unit : unité de données utilisée par une couche du modèle OSI pour communiquer avec les autres couches.

Port Identifiant de la couche transport permettant de distinguer les applications s'exécutant sur un même hôte.

Protocole Ensemble de règles et de conventions régissant la communication entre les périphériques d'un réseau.

R

RFC Request for Comments : document publié par l'IETF définissant les protocoles et normes d'Internet.

RIP Routing Information Protocol : protocole de routage de la couche réseau utilisé pour échanger des informations de routage entre les périphériques d'un réseau.

RJ45 Connecteur standard des câbles à paires torsadées (nom technique : 8P8C).

Routeur Équipement de couches 1 à 3 qui achemine les paquets entre réseaux à l'aide d'une table de routage.

Réseau informatique Mise en relation d'au moins deux systèmes informatiques au moyen d'un médium de communication.

S

Shell Un programme qui interprète les commandes de l'utilisateur et les exécute. Il fournit une interface en ligne de commande pour interagir avec le système d'exploitation.

Signal Grandeur physique porteuse d'information, transmise d'un émetteur à un récepteur à travers un canal.

Signal analogique Signal continu pouvant prendre une infinité de valeurs.

Signal numérique Signal discret ne pouvant prendre qu'un nombre fini de valeurs.

SPOF Single Point of Failure : élément dont la panne entraîne l'arrêt de l'ensemble du système.

STP Shielded Twisted Pair : câble à paires torsadées blindées, plus résistant aux interférences.

Switch Commutateur : équipement de couches 1 et 2 qui commute les trames vers le port du destinataire grâce à sa table d'adresses MAC.

T

Table de routage Table utilisée par un routeur pour déterminer le chemin des paquets vers leur réseau de destination.

Table de vérité Tableau énumérant les sorties d'un opérateur logique pour toutes les combinaisons d'entrées possibles.

TCP Transmission Control Protocol : protocole de la couche transport assurant une communication fiable, orientée connexion, avec contrôle de flux et retransmission des segments perdus.

TCP/IP Transmission Control Protocol/Internet Protocol : ensemble de protocoles utilisés pour la communication sur Internet.

Topologie Forme d'organisation des liens d'un réseau : bus, anneau, étoile, maillée ou arbre.

Téléinformatique Association des techniques des télécommunications et de l'informatique pour réaliser l'échange de données à distance.

U

UDP User Datagram Protocol : protocole de la couche transport assurant une communication non fiable, sans connexion, avec un faible overhead et une faible latence.

Unicast Mode de transmission d'un émetteur vers un récepteur unique (point à point).

UTF-8 Encodage de caractères à longueur variable, compatible ASCII, couvrant l'ensemble des langues.

UTP Unshielded Twisted Pair : câble à paires torsadées non blindées.

W

WAN Wide Area Network : réseau étendu couvrant de grandes zones géographiques et interconnectant plusieurs LAN.

WLAN Wireless Local Area Network : réseau local utilisant des technologies sans fil.

ANNEXE

Bibliographie

[1] International Telecommunication Union, « Measuring Digital Development: Facts and Figures 2025 », technical report, 2025. Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://www.itu.int/itu-d/reports/statistics/facts-figures-2025/

[2] C. Külling, G. Waller, L. Suter, I. Willemse, J. Bernath, et D. Süss, « JAMES — Jeunes, activités, médias : enquête Suisse 2024 », technical report, 2024. Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://www.zhaw.ch/storage/psychologie/upload/forschung/medienpsychologie/james/2018/JAMES_2024_FR.pdf

[3] UNESCO, « Global Education Monitoring Report 2023. Technology in education: A tool on whose terms? », technical report, 2023. Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://www.unesco.org/gem-report/en/technology

[4] G. Flodgren, A. Rachas, A. J. Farmer, M. Inzitari, et S. Shepperd, « Interactive telemedicine: effects on professional practice and health care outcomes », *Cochrane Database of Systematic Reviews*, nº 9, p. CD002098, 2015, doi: 10.1002/14651858.CD002098.pub2.

[5] Office fédéral de la statistique, « Enquête sur l'utilisation d'internet 2025 », technical report, 2025. Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://www.bfs.admin.ch/bfs/fr/home/statistiques/culture-medias-societe-information-sport/societe-information/indicateurs-generaux/menages-population/utilisation-internet.html

[6] International Energy Agency, « Energy and AI », technical report, 2025. Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://www.iea.org/reports/energy-and-ai

[7] C. Freitag, M. Berners-Lee, K. Widdicks, B. Knowles, G. S. Blair, et A. Friday, « The real climate and transformative impact of ICT: A

critique of estimates, trends, and regulations », *Patterns*, vol. 2, nº 9, p. 100340, 2021, doi: 10.1016/j.patter.2021.100340.

[8] Cambridge Centre for Alternative Finance, « Cambridge Bitcoin Electricity Consumption Index (CBECI) ». Consulté le: 10 juillet 2026. [En ligne]. Disponible sur: https://ccaf.io/cbnsi/cbeci

[9] IoT Analytics, « State of IoT 2025 ». Consulté le: 13 juillet 2026. [En ligne]. Disponible sur: https://iot-analytics.com/number-connected-iot-devices/

[10] Google, « IPv6 adoption statistics ». Consulté le: 13 juillet 2026. [En ligne]. Disponible sur: https://www.google.com/intl/en/ipv6/statistics.html

[11] APNIC Labs, « IPv6 Deployment Statistics ». Consulté le: 13 juillet 2026. [En ligne]. Disponible sur: https://stats.labs.apnic.net/ipv6

[12] StatCounter, « Desktop Operating System Market Share Worldwide ». Consulté le: 15 juillet 2026. [En ligne]. Disponible sur: https://gs.statcounter.com/os-market-share/desktop/worldwide/

[13] Wikipedia, « Tux (mascot) ». Consulté le: 15 juillet 2026. [En ligne]. Disponible sur: https://en.wikipedia.org/wiki/Tux_(mascot)

[14] J. Fay, « Linux kernel gets continuity plan for post-Linus era ». Consulté le: 15 juillet 2026. [En ligne]. Disponible sur: https://www.theregister.com/2026/01/27/linux_continuity_plan/

[15] Leonardo, « Linus Torvalds and Richard Stallman ». Consulté le: 15 juillet 2026. [En ligne]. Disponible sur: https://www.linux.com/training-tutorials/linus-torvalds-and-richard-stallman/

[16] Wikipedia, « GNU/Linux naming controversy ». Consulté le: 15 juillet 2026. [En ligne]. Disponible sur: https://en.wikipedia.org/wiki/GNU/Linux_naming_controversy

ANNEXE

Index

P

R

S

T

U

W