

Promethee Spathis

14 Followers

About

Follow

Sign in

Get started



Comprendre Internet et son fonctionnement



Promethee Spathis Aug 29 · 99 min read



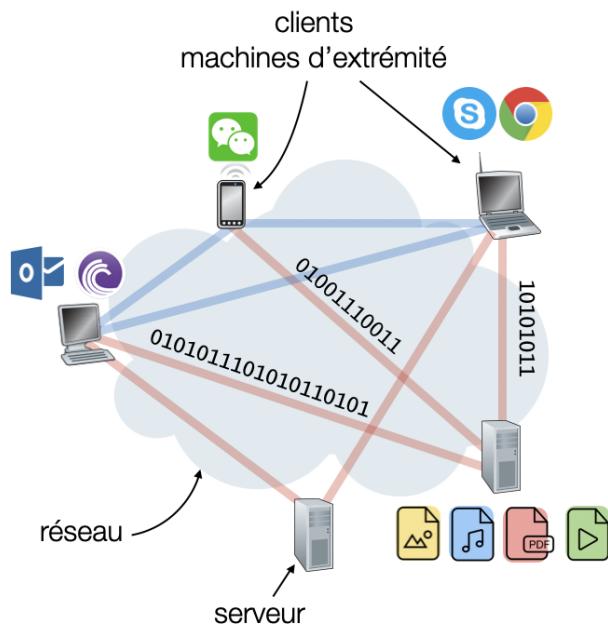
Les pionniers de l'Internet : de gauche à droite, Jon Postel, Steve Crocker et Vint Cerf.

Dans cet article, j'introduis les notions et concepts sous-jacents à Internet et aux réseaux de données en général. Il s'agit d'un article incrémental que je mets à jour régulièrement. Cet article concentre en grande partie les notes de cours de l'unité d'enseignement LU3IN033 que **Promethee Spathis** enseigne à Sorbonne Université et que vous pouvez [visionner sur YouTube](#).

La table des matières et la liste des mots-clés sont disponibles [ici](#).

Les données, d'où viennent-elles et pourquoi les transporter ?

Les **données** font référence aux bits que génèrent les **applications** exécutées à l'initiative des **utilisateurs** sur leur ordinateur. Des exemples d'applications sont les navigateurs Web, les clients mail ou un client BitTorrent. Toutes ces applications sont des programmes qui communiquent avec un programme homologue, exécuté sur une machine distante.



Les **machines hôtes** situées à la périphérie du réseau hébergent les **applications**. Dans le cadre des applications client serveur, on distingue les **serveurs** qui proposent des contenus et des services que les **clients** accèdent en soumettant des requêtes.

Modèle client-serveur

Originellement, ces deux programmes étaient conçus de manière **asymétrique** : le programme distant diffère de celui

exécuté sur l'ordinateur des utilisateurs. En tant qu'utilisateurs, nous nous connectons à un réseau de données en vue de recevoir une page web, des images, un email ou une vidéo. Ces objets sont hébergés sur des machines appelées **serveurs**. Nous sommes les **clients** de ces machines qui, du fait de répondre à nos requêtes, sont appelées serveurs.

Les serveurs offrent des capacités de stockage qui leur permettent d'héberger des contenus accessibles à tous. Les serveurs disposent également de capacités de traitement qui leur permettent de traiter les requêtes de leurs clients.

Modèle pair-à-pair

Récemment, on a vu l'avènement d'un nouveau type d'applications résultant de l'exécution de programmes identiques. On parle d'applications **pair-à-pair** où une même machine peut agir aussi bien en tant que client que serveur.

Interopérabilité

Applications client-serveur ou pair-à-pair sont mises à disposition au téléchargement par leurs développeurs. Les utilisateurs sont libres d'installer ces applications sur leur ordinateur. Il n'y a pas, à proprement parler, de réel contrôle concernant les applications qu'un utilisateur peut installer sur son ordinateur.

Qu'ils s'agissent des constructeurs de matériel informatique ou des concepteurs de systèmes d'exploitation qui équipent nos

ordinateurs à leur achat, tous œuvrent dans le but de concevoir des équipements ouverts et interopérables qui rendent leurs produits attractifs du fait de leur versatilité. C'est ce qui a fait le succès de l'Internet.

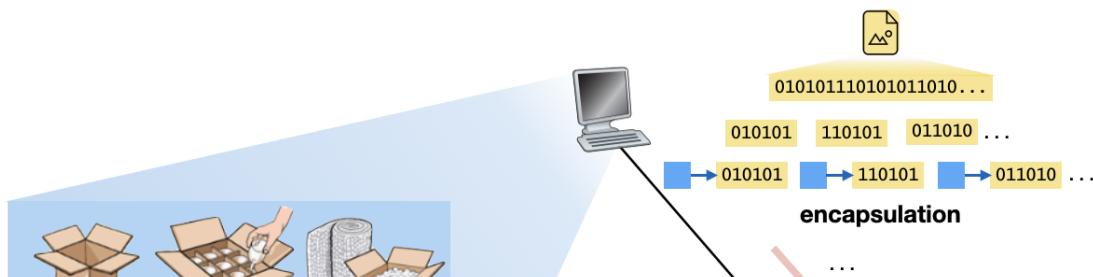
L'utilisation de certains logiciels peut être restreinte. Un fournisseur d'accès Internet (FAI) peut éventuellement être soumis aux lois nationales contre le piratage informatique ou la violation des droits d'auteur par exemple. Pour autant, les FAI ne sont pas toujours en mesure de filtrer avec succès les trafics dérogeant à ces lois.

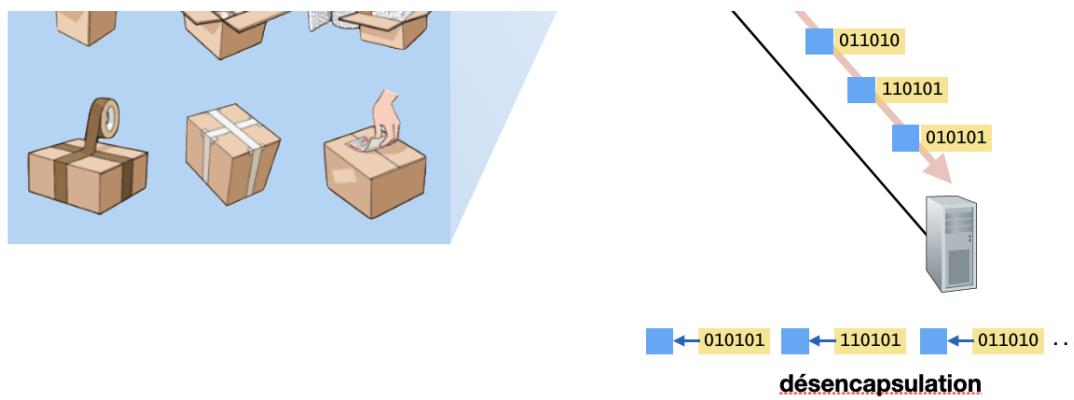
Comment les données sont-elles ‘emballées’ en vue de leur transport ?

A la manière des produits ou denrées transportés par un transporteur tel que Colissimo ou UPS, les données ont besoin d'être emballées.

Fragmentation

Dans un réseau de données, les données sont **fragmentées** : une longue série de bits de données appartenant à un même fichier est répartie dans plusieurs colis appelés **messages**.





Les bits de données appartenant à un même fichier sont **fragmentés** et répartis dans plusieurs messages. Un **entête** est ajouté à chaque fragment. Les entêtes contiennent des bits de contrôle nécessaires au transport des données.

Encapsulation et désencapsulation

L'équivalent des cartons d'emballage sont des bits supplémentaires ajoutés généralement en tête des bits de données appelés de ce fait, **entête**. A la manière des dimensions ou du poids d'un colis, la longueur des messages exprimée en nombre de bits est soumise à des restrictions qui imposent une longueur minimale et maximale. L'ajout d'un entête est appelé **encapsulation** et le retrait de l'entête **désencapsulation**.

Entête et charge utile

Dans un réseau de données, un colis est donc un message constitué d'un entête et des bits de données. Les bits de données forment le corps du message que l'on appelle aussi **charge utile** du message. Les messages ayant une longueur maximale, plusieurs messages sont nécessaires pour transmettre un même objet telle qu'une image.

Dans l'entête d'un message, on retrouve des bits appelés

informations de contrôle telles que l'adresse du destinataire et celle de l'expéditeur. A la manière du papier-bulle utilisé pour éviter la casse des produits fragiles, l'entête peut également contenir des bits appelés **somme de contrôle** permettant de détecter les bits reçus en erreur. Une fois détectées, on peut réparer les erreurs en retransmettant les messages erronés.

Les bits correspondant aux adresses ou la somme de contrôle sont des exemples de champs. Un entête est une succession de **champs** dont la longueur est variable ou fixe. La **structure d'un entête** aussi appelé **format d'un entête** fait référence aux champs qui composent cet entête. Pour chaque champ est précisé le nom, la longueur en bits, et sa signification selon la valeur qu'il contient. La valeur d'un champ peut être fournie sous la forme d'un code binaire (la valeur 0x06 fait référence au protocole TCP) ou du code ASCII de la valeur en question (la valeur 0x474554 indique la méthode GET).

Transporter les données avec des garanties

L'entête d'un message étant ajouté dans un but précis, plusieurs entêtes sont nécessaires afin d'assurer les nombreuses propriétés attendues concernant le transport des données.

De telles propriétés comprennent la fiabilité (réception de l'intégralité des données, exemptes d'erreurs), l'efficacité (la

quantité des données émises maximise l'utilisation de la capacité du réseau) ou la résistance au facteur d'échelle.

Fiabilité

Des messages pouvant être perdus, reçus en erreur, en désordre ou dupliqués, des bits de contrôle sont inclus dans leur entête. Ces bits font référence à un numéro appelé **numéro de séquence** qui permet de détecter les messages manquants, de réordonner les messages ou de supprimer les doublons. On trouve également une **somme de contrôle** dont le rôle est de vérifier si le message contient des bits en erreur. Comme l'indique son nom, son calcul résulte de la somme des bits du message avant son envoi. A sa réception, le destinataire refait le même calcul et vérifie si le résultat coïncide avec la somme calculée par la source.

Efficacité

L'efficacité dans un réseau de données est similaire à celle qu'un transporteur met en œuvre pour s'assurer que ses fourgons sont tous utilisés au maximum de leur capacité de transport. L'efficacité peut également faire référence à la proportion du message occupée par son entête comparée à sa charge utile. Si envoyer un colis est facturé sur la base de son poids brut, l'expéditeur cherchera à minimiser le poids du carton d'emballage. Il en va du même des autoroutes dont le nombre de voies est déterminé de façon à éviter les bouchons mais également leur sous-utilisation. Le coût de réalisation et d'exploitation des autoroutes doit être en adéquation avec leur

utilisation, leur financement dépendant des recettes aux péages. On parlera alors d'utilisation efficace des autoroutes. De la même manière, les coûts liés au déploiement et à la maintenance d'un réseau de données ainsi que sa capacité doivent être en adéquation avec le volume des données transportées, le nombre d'utilisateurs et le prix d'un abonnement Internet.

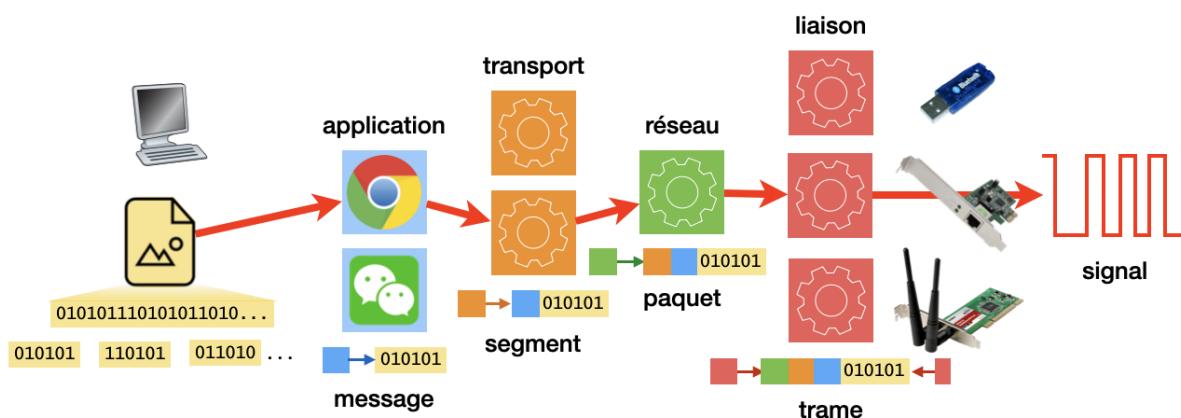
Résistance au facteur d'échelle

Une autre propriété est la résistance au facteur d'échelle (ou scalabilité). Il s'agit d'un principe de conception qui caractérise un système dont les performances ne sont pas, ou sont peu sensibles à une grandeur dénombrable croissante, caractérisant ce système. Cette grandeur peut faire référence au nombre d'utilisateurs ou de nœuds dans un réseau. Une autre grandeur peut aussi faire référence au nombre de messages en transit dans le réseau ou de requêtes que reçoit un serveur. Un transporteur de colis doit s'assurer du bon déroulement de ses opérations, y compris à l'occasion des fêtes de fin d'années.

Dans un réseau de données, pour assurer les nombreuses propriétés attendues concernant la livraison des données, un seul entête ne suffit pas.

Architecture en couches

Un message comprend donc une série d'entêtes ajoutée aux bits de données. Un entête est ajouté dans un but spécifique qui détermine le type d'informations contenues dans cet entête. L'ajout d'un entête se fait en sollicitant un programme qui prend en paramètre les bits de données.



Préalablement à leur envoi, les données transitent le long d'une chaîne de programmes qui chacun ajoute un entête. Le dernier programme à ajouter un **entête** ajoute également un **enqueue**.

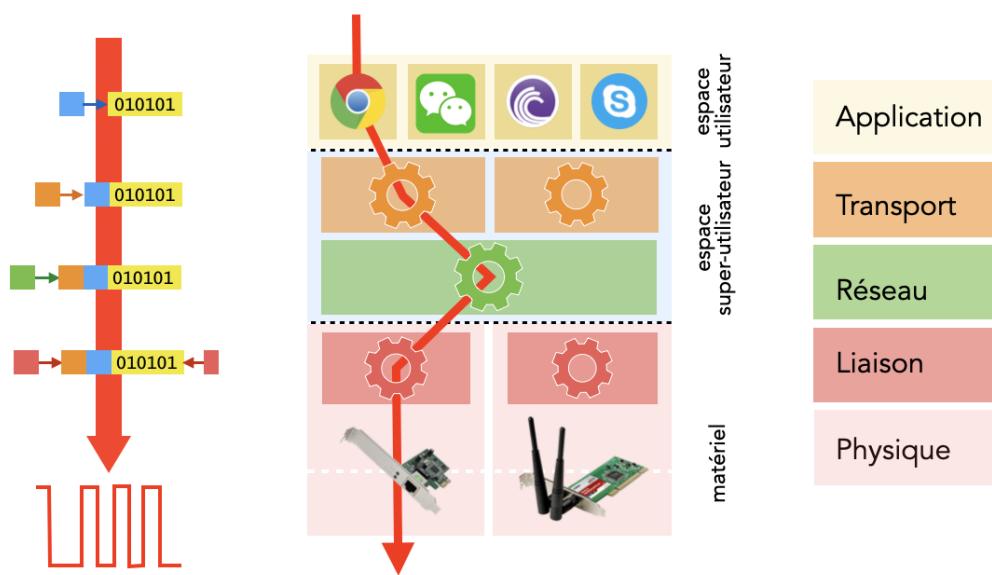
Les bits de données passent par une série de programmes qui se relaient pour ajouter un entête supplémentaire avant que le message final soit prêt à être transmis sur le support de transmission. Le dernier programme à ajouter un entête, ajoute également des bits en fin du message appelés de ce fait **enqueue** du message.

A la réception du message final, les entêtes sont progressivement retirées en traversant une séquence de programmes similaire à celle traversée avant l'émission de ce message mais en sens inverse. Le dernier entête à avoir été ajouté est le plus externe (situé à gauche) et donc le plus

accessible. Chaque programme accède uniquement à l'entête ajouté côté émetteur par son programme homologue.

Couches et protocoles

Dans Internet, les programmes qui contribuent à la préparation des données en vue de leur transmission sont au nombre de 5. En première position, on retrouve les programmes applicatifs qui génèrent les données côté émetteur et les consomment côté récepteur. En dernière position, on retrouve la carte réseau dont les composants matériels transforment le message sous la forme d'un signal émis sur le support de transmission.



Les programmes qui contribuent à la préparation des données en vue de leur transmission sont au nombre de 5. **Par souci de représentation**, ces programmes sont **empilés** et on parle, de ce fait, de **couches** en lieu en place de ces programmes.

Empilement. Par souci de représentation, ces programmes sont empilés les uns sur les autres et la traversée des données se fait de haut en bas côté émetteur et de bas en haut côté récepteur. On parle alors de **couches** en lieu et place des

programmes, une même couche pouvant être implémentée de plusieurs manières différentes selon les propriétés attendues de cette couche.

Protocole. Une couche collabore avec sa couche homologue distante selon des règles bien précises concernant la structure de l'entête qu'elle ajoute et retire aux messages. Ces règles définissent également les actions à exécuter en vue de l'émission ou suite à la réception d'un message. De ces actions peuvent résulter l'installation, la mise à jour ou la suppression d'un **état**. Un état fait référence à une valeur stockée en mémoire qui caractérise l'état de l'échange entre deux machines. L'ensemble de ces règles forme un **protocole**. Pour communiquer entre elles, deux machines exécutent simultanément plusieurs protocoles, un par couche.

Standardisation. Certains protocoles ont été standardisés rendant ainsi leurs spécifications accessibles à tous. C'est le cas par exemple de HTTP, le protocole qui régit les communications entre navigateurs et serveurs Web. Les spécifications de HTTP étant publiques, plusieurs navigateurs sont proposés par différentes sociétés telles que Microsoft ou Google. Du fait d'avoir été développés conformément au standard, navigateurs et serveurs savent communiquer entre eux, le logiciel côté serveur étant également conforme au standard.

Entête de protocole. À la manière des étiquettes apposées sur

un colis, un protocole ajoute un entête à l'émission d'un message. La position de cet entête est déterminée par la couche à laquelle le protocole appartient. L'entête est ajouté en vue d'être lu par sa couche homologue une fois le message reçu. Ce sont les informations de contrôle contenues dans les champs de cet entête qui permettent de déterminer les actions à entreprendre à la réception du message. Des exemples d'actions inclus l'installation ou la mise à jour d'un **état**, la génération d'une réponse sous la forme d'un message retourné à la source ou la suppression du message si reçu en erreur par exemple. De l'exécution de ces actions résultera le service attendu de cette couche.

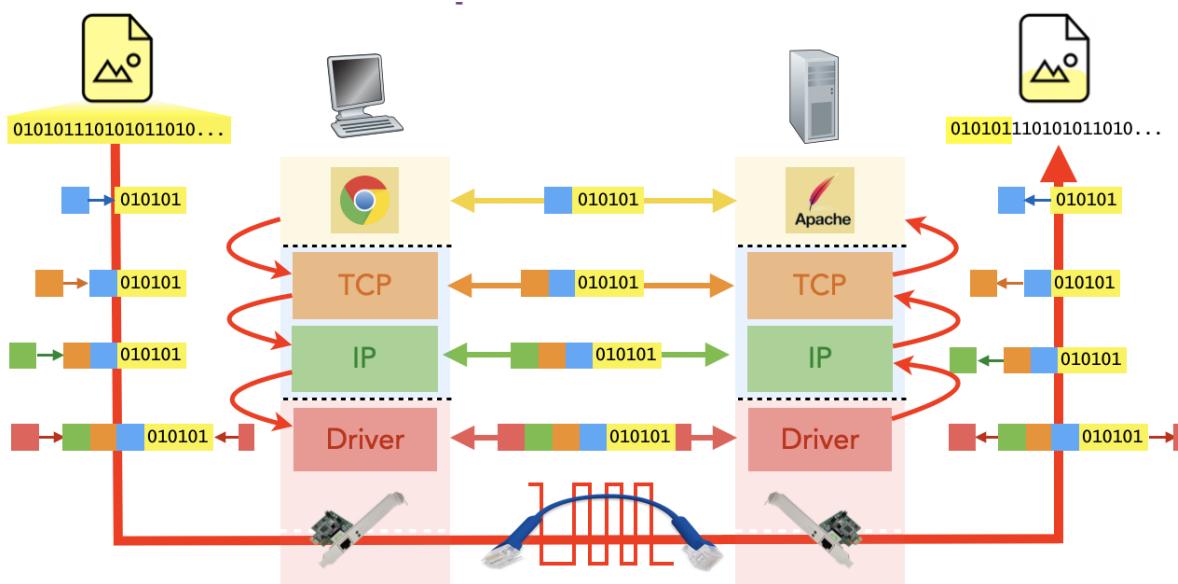
Services de communication. Une même couche pouvant offrir différents types de service, certaines couches implement plusieurs protocoles. C'est le cas de la couche application qui implémente autant de protocoles que d'applications installées sur nos ordinateurs. Un service de communication résulte de l'exécution du protocole spécifique à la couche qui offre ce service.

L'ensemble des couches et des protocoles qu'elles implémentent constitue ce qui est communément appelée une **architecture de réseau**. Du fait de l'empilement des couches qui implémentent les protocoles, on parle aussi d'**architecture en couches**.

Internet et modèle OSI

Les composants logiciels et matériels qui permettent aux utilisateurs d'envoyer ou de recevoir des données via les applications qu'ils installent sur leur machine sont donc structurés en couches, au nombre de 5 dans Internet.

Les fonctions prises en charge par chacune de ses couches sont habituellement décrites en faisant référence au **modèle OSI** (Open Systems Interconnection) normalisé dans les années 1970 par l'ISO (International Organization for Standardization). Le modèle OSI comporte 7 couches identifiées par un numéro et un nom. La numérotation commence à 1 pour la couche la plus basse appelée couche Physique.



Source et destination implémentent une pile protocolaire composée de protocoles qui chacun, ajoute côté émetteur un entête aux données provenant de la couche immédiatement supérieure. Côté récepteur, les couches retirent l'entête aux messages passés par la couche immédiatement inférieure.

Sur les 7 couches du modèle OSI, seules 5 ont été implémentées dans Internet, les fonctions des couches numérotées 5 et 6, appelées Session et Présentation respectivement, étant prises en charge pour certaines par la couche 7. L'ensemble de ces couches et les protocoles qu'elles implémentent forment une **architecture de réseau**. On parle aussi de **pile protocolaire** pour faire référence à l'ensemble des protocoles réunis au sein d'une même architecture de réseau.

Couche Application (7)

L'architecture de Internet comprend au niveau le plus élevé la couche 7 appelée couche Application. Les programmes qui implémentent cette couche sont par exemple, les navigateurs Web côté client et les serveurs Web côté serveur. Leur but est d'exécuter le protocole HTTP (Hypertext transfer protocol) qui permet le téléchargement des pages Web.

Les deux couches suivantes nommées couche Transport (couche 4) et couche Réseau (couche 3) sont fournies avec les systèmes d'exploitation installés sur les ordinateurs personnels, les smartphones ou autres objets communicants.

Couche Transport (4)

La couche 4 appelée couche Transport offre deux services résultant de deux protocoles : TCP (Transmission control protocol) et UDP (User datagram protocol).

TCP segmente les données générées par l'application émettrice

et ajoute à chaque segment un entête. Le message résultant de cette encapsulation est appelé **segment**. Dans le cas de UDP, les données générées par l'application ne sont pas segmentées. UDP se contente d'ajouter son entête aux données applicatives. Le message résultant de l'encapsulation des données applicatives est appelé **datagramme**. Si la taille du datagramme le requiert, c'est la couche Réseau (3) sous-jacente qui se chargera de sa fragmentation.

C'est la couche Transport qui fournit aux développeurs l'interface de programmation, appelée **socket**, qui permet de concevoir des applications réseau.

Couche Réseau (3)

La couche 3 appelée couche Réseau implémente le protocole IP (Internet Protocol) qui ajoute à son tour un entête et le message qui en résulte est appelé **paquet**. Un paquet est donc un segment TCP ou un fragment de datagramme UDP auquel a été ajouté un entête de couche 3. Cet entête contient notamment les adresses source et destination du paquet. La couche 3 a pour rôle l'acheminement des paquets vers leur destination.

Couche Liaison de données (2)

La couche 2 appelée couche Liaison de données est à la fois logicielle et matérielle. La partie logicielle correspond aux pilotes qui permettent d'interagir avec la partie matérielle qui elle, correspond aux cartes réseau telles que Ethernet ou Wifi. La couche Liaison de données a la particularité d'ajouter un

entête mais également un enqueue à la suite des bits de données du paquet pour former une **trame**.

Couche Physique (1)

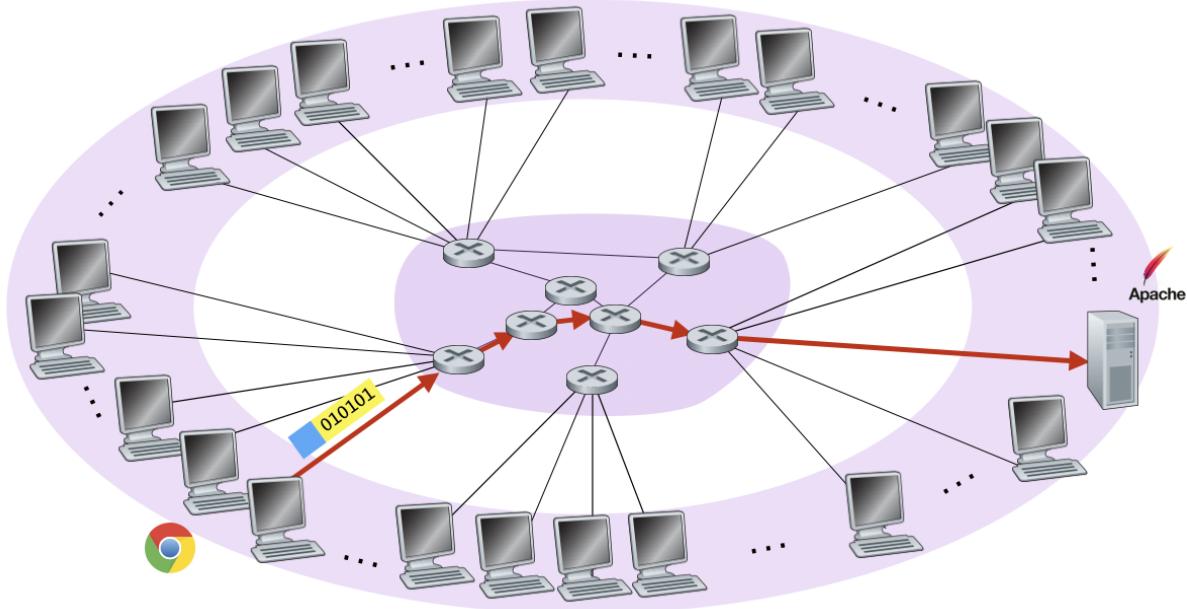
La partie matérielle comprend également la couche 1 appelée couche Physique. Elle est responsable de la conversion des bits sous la forme d'un signal adapté aux supports de transmission. Dans l'architecture Internet, les couches 1 et 2 forment un tout dénommé **interface réseau** que l'on désigne usuellement par l'identifiant *eth0* dans le cas des cartes Ethernet ou Wifi.

Les couches 1 et 2 dépendent des caractéristiques physiques du support de communication et du nombre de machines connectées à ce support.

Cœur vs. périphérie du réseau

Dans Internet, la pile TCP/IP fait référence aux deux protocoles implémentés respectivement par la couche Transport (4) et Réseau (3). Pour comprendre le rôle des couches 4 et 3 dans Internet, il est nécessaire de présenter les deux types de machines que l'on trouve dans les réseaux de données. Ces deux types de machines diffèrent selon :

1. leur position dans le réseau et
2. les fonctions qui leur incombent.



Les **machines hôtes** sont situées à la périphérie du réseau. Le cœur du réseau est formé d'une interconnexion partiellement maillée d'équipements appelés **routeurs** en charge d'acheminer les données entre machines hôtes source et destination.

Machines hôtes vs. routeurs

Machines hôtes. Le premier type de machines sont les ordinateurs personnels, tablettes, smartphones et autres dispositifs que nous utilisons pour accéder à Internet. Ces machines sont les seules où peuvent être installées et exécutées les applications dans un réseau. Du fait d'héberger les applications, ces machines sont appelées **machines hôtes**. Elles forment la **périphérie du réseau** où les données sont générées et consommées. Les machines hôtes sont donc situées aux extrémités des communications.

Routeurs. Le second type de machines sont les noeuds internes du réseau appelés **routeurs**. Ces derniers sont reliés entre eux par des liens selon une configuration formant un maillage partiel. Les routeurs et les liens les reliant forment ce qu'on

appelle le **cœur du réseau** en opposition à la périphérie du réseau où se retrouvent les machines hôtes. Cette interconnexion de routeurs permet de minimiser le nombre de liens nécessaires pour connecter les machines hôtes entre elles.

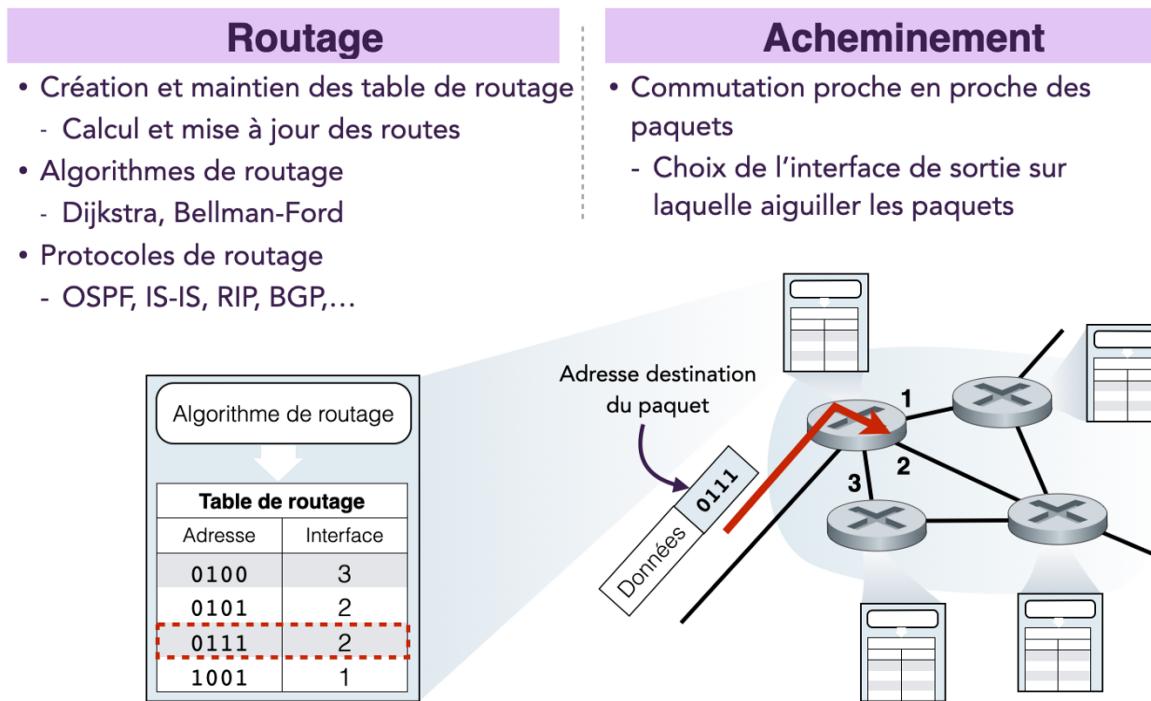
Parmi l'ensemble des routeurs, on distingue ceux situés en bordure du cœur auxquels se connectent les machines hôtes pour accéder au réseau. Ces routeurs sont de ce fait dénommés **routeurs d'accès**. Les machines hôtes accèdent à Internet en contactant un premier routeur qui agit comme la passerelle par défaut appelée en anglais **default gateway**. Les machines hôtes qui utilisent les services d'une même passerelle sont connectées à un même réseau physique appelé **réseau local**.

Le rôle des routeurs découle de la topologie du réseau de cœur. Le maillage partiel entraîne l'existence de plusieurs chemins alternatifs entre paires de machines hôtes. Il est donc nécessaire pour chaque destination de sélectionner un chemin unique, le long duquel seront acheminés invariablement les messages tant que ce chemin reste disponible. Nous verrons que cette constance dans le choix du chemin emprunté par l'ensemble des messages pour une même destination est primordial pour la réparation efficace de leur perte.

Routage vs. acheminement

Le rôle d'un routeur est donc double. Le premier est l'acheminement des données, le second est le routage (ou sélection de chemins). Un routeur achemine les messages en les

transmettant sur le lien de sortie qui les rapprochera de leur destination finale. Ce lien de sortie permet de joindre un nœud voisin appelé **saut suivant**. Le choix du saut suivant résulte d'un algorithme de routage dont l'objectif est de choisir un chemin pour chaque destination du réseau.



Routage vs. acheminement.

Algorithme de routage. Pour ce faire, certains algorithmes de routage calculent le chemin le plus court, la distance des chemins pouvant faire référence au nombre de routeurs qu'ils les constituent par exemple. C'est le cas de l'algorithme de Bellman-Ford. Si les liens sont configurés avec un coût, le chemin choisi est alors celui dont la somme des coûts des liens qui le constituent, est la plus petite. C'est le cas de l'algorithme de Dijkstra. D'autres algorithmes considèrent les accords commerciaux entre FAI (Fournisseurs d'accès Internet) qui

collaborent pour acheminer les données sur des distances couvrant des zones géographiques étendues telles que des pays ou des continents.

Protocole de routage. Pour s'exécuter, ces algorithmes ont besoin de connaître la topologie partielle ou complète du réseau. Les routeurs la découvrent en exécutant un protocole de routage implémenté par la couche Réseau (3) des routeurs. Un protocole de routage permet aux routeurs de collaborer en échangeant des messages contenant des informations sur la topologie du réseau et selon les règles définies par le protocole de routage. Des exemples de protocole de routage sont RIP (Routing information protocol), OSPF (Open shortest path first) ou BGP (border gateway protocol).

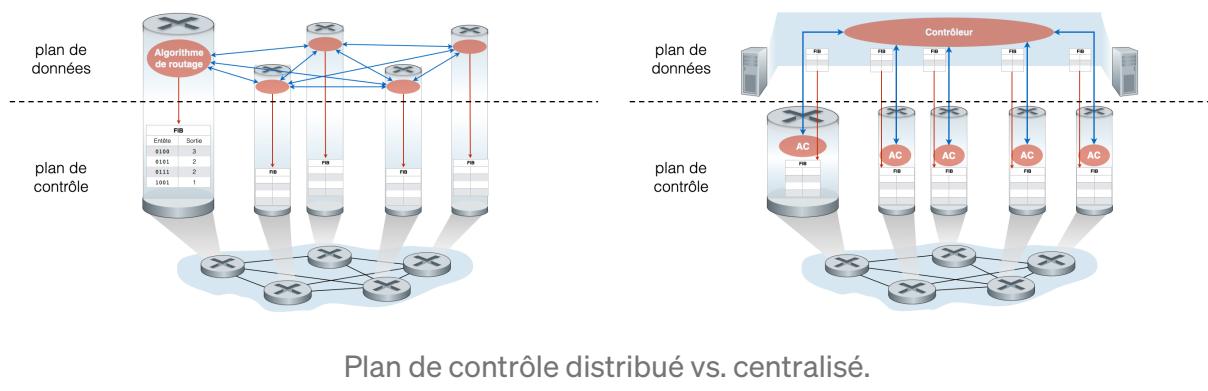
Adressage. Pour calculer les chemins, les routeurs doivent connaître la position des nœuds au sein de la topologie du réseau. Pour ce faire, une valeur unique appelée **adresse** est attribuée à chacun des nœuds. Dans le cas de Internet, cette adresse est une valeur binaire longue de 32 bits appelée **adresse IP**. Les adresses IP sont présentées au format décimal pointé : les quatre octets qui les forment sont convertis en décimal et les valeurs décimales résultant de cette conversion séparées par un point.

A la manière des numéros de téléphone, les adresses IP sont attribuées hiérarchiquement par les fournisseurs Internet. Tout comme les indicatifs téléphoniques qui identifient une zone

géographique, les premiers bits d'une adresse identifient le réseau auquel est connecté la machine à qui appartient l'adresse. Ces bits appelés **préfixe réseau** ont une longueur variable déterminée par le nombre de bits positionnés à 1 dans une adresse particulière appelée **masque de réseau**, défini pour tout réseau.

Nous verrons par la suite que les machines sont identifiées par une seconde adresse appelée **adresse MAC**, gérée par la couche Liaison de données.

Plan de données vs plan de contrôle. L'acheminement des données et la sélection des chemins qu'empruntent les données sont des fonctions réalisées par la couche Réseau (3). Les fonctions nécessaires à l'acheminement des données font partie du plan de données et celles nécessaires à la sélection des chemins font partie du plan de contrôle des routeurs.



Plan de gestion. En plus des plans de données et de contrôle, on trouve le **plan de gestion** qui réunit les fonctions

nécessaires à la configuration et la supervision des plan de données et de contrôle des routeurs. C'est du plan de gestion que résulte par exemple la configuration du coût des liens nécessaire au protocole OSPF. Si les plans de données et de contrôle sont automatisés, le plan de gestion lui nécessite l'intervention manuelle d'un administrateur.

L'ensemble des autres fonctions nécessaires à la livraison des données sont prises en charge par les machines hôtes. Ces fonctions sont réalisées par la couche Transport, numérotée 4. Elles répondent aux besoins des applications qui s'exécutent au niveau 7. Ces fonctions incluent la fiabilisation des messages ou l'utilisation efficace et équitable de la bande passante du réseau.

Fiabilisation. Au cours de leur acheminement, des messages peuvent être perdus ou reçus en erreur. Dans l'architecture Internet, la correction des pertes de messages et des messages en erreur est à la charge de la couche Transport (4). Ces pertes pouvant résulter de l'absence de ressources nécessaires pour traiter les messages si en surnombre, il est nécessaire d'ajuster le débit d'émission des sources de façon à éviter les pertes qui pourraient s'ensuivre. Cet ajustement se fait équitablement parmi l'ensemble des sources dont les messages empruntent le même chemin. La bande passante utilisée par chacune de ces sources est répartie équitablement.

La couche Réseau (3) se retrouve sur l'ensemble des machines

de l'Internet y compris les machines hôtes, ces dernières étant responsables de définir les adresses IP source et destination des paquets. La couche Transport (4), quant à elle, est présente uniquement sur les machines hôtes. Ce choix résulte d'un principe de conception appelé le **principe du bout en bout**.

Principe de bout en bout

Le principe de bout en bout exerce une force centrifuge sur les fonctions nécessaires à la livraison des données. Lorsque le choix est donné de concevoir une fonction en faisant intervenir, soit toutes les machines du réseau y compris les routeurs, ou uniquement les machines hôtes de la périphérie, il est recommandé de privilégier en priorité la seconde solution.

La logique du principe de bout en bout est d'éviter l'exécution répétée d'une même fonction sur l'ensemble des routeurs situés le long du chemin que traverse un message. Dans le cas de l'Internet, la longueur d'un chemin alterne en moyenne entre 15 et 20 sauts. En cantonnant l'installation d'une fonction uniquement aux extrémités, cette même fonction est exécutée deux fois. C'est le cas des fonctions nécessaires à la réparation d'un message en erreur ou perdu.

Fiabilisation de bout en bout

Dans Internet, la détection des messages en erreur résulte de

l'utilisation d'une somme de contrôle qui permet au récepteur de détecter les messages contenant des bits de données en erreur. Ces erreurs peuvent intervenir plus en amont le long du chemin mais ce n'est qu'une fois le message acheminé jusqu'à sa destination finale qu'il sera rejeté par le récepteur qui prétendra ainsi ne pas l'avoir reçu.

L'absence d'accusé de réception provoquera l'expiration d'un temporisateur armé par la source à l'émission du message et la retransmission du message. La valeur du temporisateur de retransmission est calculée de façon à refléter la durée d'un aller-retour entre source et récepteur du message. Pour garantir la validité de cette valeur, il est nécessaire de préserver la longueur du chemin entre source et récepteur. De ce fait, l'acheminement des paquets dans Internet se fait le long d'un chemin unique qui ne changera qu'en cas de défaillance.

En limitant l'exécution de ces fonctions aux seules extrémités, on retarde la réparation d'un message en erreur. Ce retard est cependant acceptable en comparaison au délai et à la charge qu'aurait induit l'exécution d'une fonction similaire mais implémentée sur l'ensemble des routeurs du réseau.

Serveurs proxy

Il en va de même pour la plupart des fonctions nécessaires à la livraison des données. Cependant, des exceptions existent, notamment sous la forme des serveurs dits mandataires, appelés **proxy servers** en anglais. Un proxy est un équipement

installé le long des chemins entre clients et serveurs. Leur rôle est d'intercepter les requêtes des clients et de se substituer, lorsque possible, aux serveurs. C'est le cas des caches Web qui stockent temporairement les pages Web les plus populaires afin d'accélérer leur livraison, les caches Web étant situés au plus près des clients comparés aux serveurs auxquels ils se substituent. Les caches Web assurent des fonctions applicatives censées être cantonnées à la périphérie du réseau. C'est dans ce sens que les proxys font exception au principe de bout en bout.

Mode datagramme vs mode circuit virtuel

Il existe d'autres architectures de réseau qui, contrairement à Internet, ont fait le choix d'impliquer les nœuds internes de leur réseau dans la fiabilisation des messages échangés. C'est le cas de Transpac par exemple dont les nœuds internes appelés **commutateurs** réservent préalablement les ressources nécessaires à l'acheminement des messages en vue d'éviter leurs pertes, faute de ressources. Ces ressources sont réservées au niveau des commutateurs situés le long du chemin qu'emprunteront par la suite les messages et ce, invariablement. L'ensemble des ressources réservées forme une **connexion** et le chemin, un **circuit virtuel**. On parle de **service en mode circuit virtuel**. Une rupture de lien ou la perte d'un des commutateurs situés le long d'un circuit virtuel entraîne l'interruption de la communication. Pour la rétablir, une nouvelle connexion est nécessaire.

Mode circuit virtuel

Mode datagramme



Mode circuit virtuel vs. datagramme.

Les mécanismes de réservation de ressources complexifient le fonctionnement des commutateurs et inexorablement, leur coût. Ce choix de conception n'a pas été suivi dans Internet. Le chemin n'étant pas fixé au préalable, on parle de service en **mode datagramme**. Un datagramme suit un chemin sans la garantie que les ressources nécessaires à son acheminement soient présentes. L'absence de ressources suffisantes fait référence à la **congestion** qui provoque la destruction des datagrammes en excès. Dans Internet, les datagrammes font référence aux paquets IP qu'on appelle aussi de ce fait, datagramme IP.

Couche Transport (4) : Protocoles UDP & TCP

La couche Transport (4) offre deux services résultant chacun de l'exécution des protocoles UDP ou TCP.

- UDP offre un service de livraison sans garantie (best effort) en mode datagramme.

- TCP offre un service de restitution fiable et efficace en mode flux d'octets.

Modes datagramme et flux d'octets font référence aux type d'interactions entre protocoles de couche Application et protocoles de couche Transport.

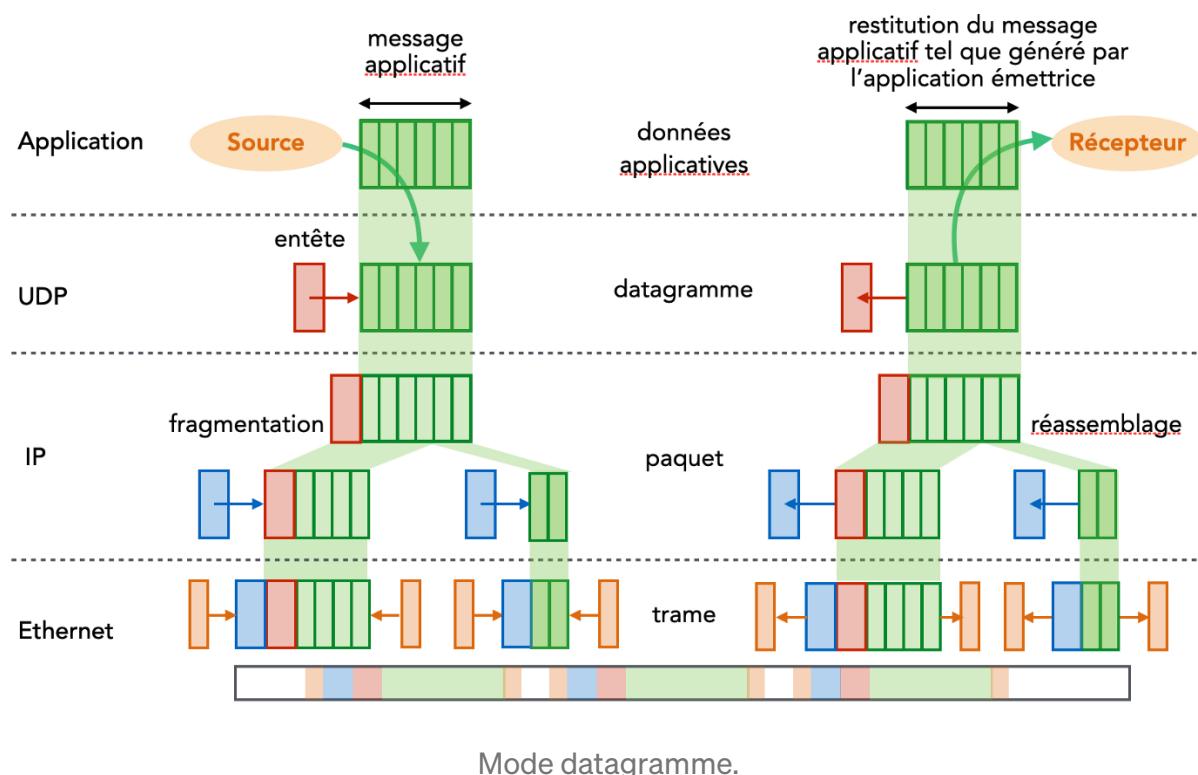
Interactions entre couches Application et Transport

Les applications sollicitent les services de la couche Transport (4) en écrivant des octets de données dans un tampon d'émission côté émetteur. Côté récepteur, les octets de données sont mis en attente dans un tampon de réception, le temps que l'application vienne les lire.

- **UDP.** Coté émetteur, UDP prélève les octets de données résultant d'une écriture, leur ajoute un entête long de 8 octets pour former un message appelé **datagramme**. UDP passe le datagramme aussitôt que formé à la couche IP. Coté récepteur, UDP désencapsule les octets de données et les place dans un tampon de réception dans l'attente que l'application vienne les lire.
- **TCP.** Coté émetteur, TCP prélève une quantité fixe d'octets de données et ajoute un entête long d'au moins 20 octets pour former un message appelé **segment**. La taille de la charge utile des segments TCP est appelée la **MSS (Maximum segment size)**. Les segments sont mis en attente avant d'être transmis, le temps que TCP juge opportun pour assurer un service efficace.

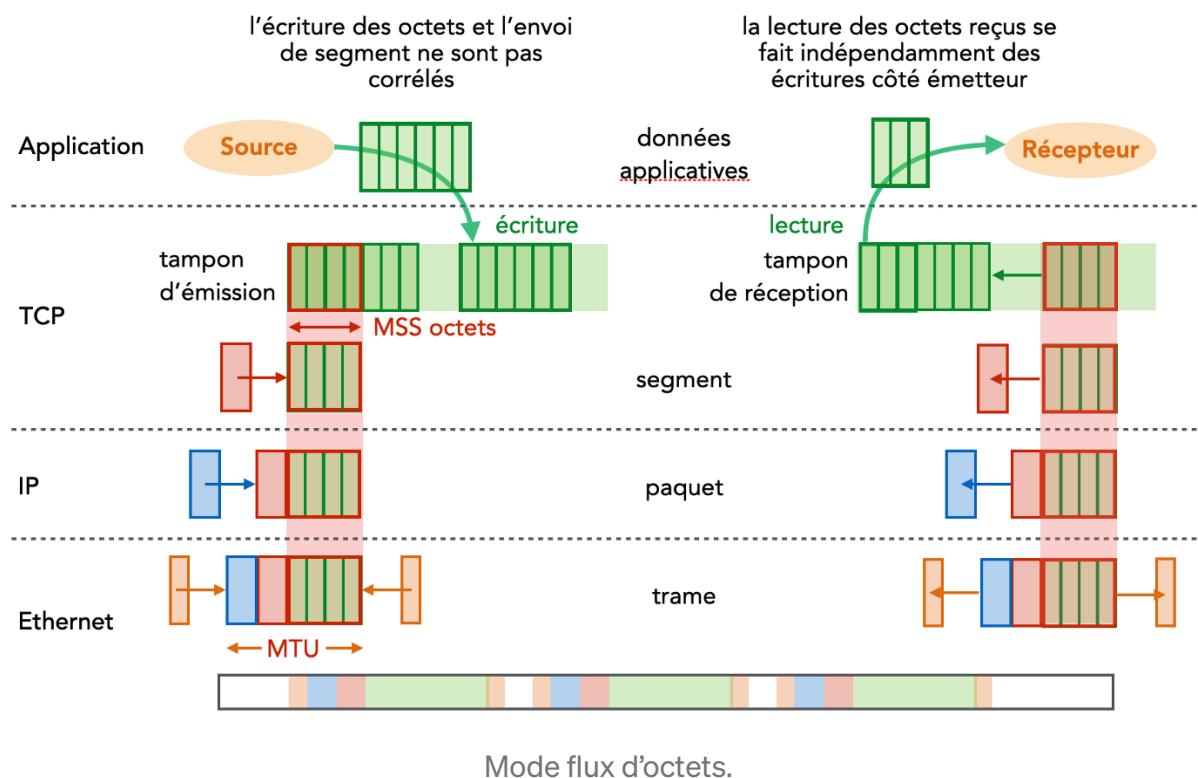
Mode datagramme vs. mode flux d'octets

Mode datagramme. La taille de la charge utile des datagrammes UDP dépend de la quantité d'octets que l'application écrit d'un coup, dans les tampons d'émission. Si la longueur d'un datagramme UDP excède la taille de la MTU locale, c'est la couche IP qui se charge de fragmenter le datagramme UDP. L'entête UDP du datagramme se retrouve alors dans les données du premier fragment IP. Le module IP du récepteur reconstitue le datagramme dans son état d'origine (aux erreurs éventuellement près) en réassemblant les fragments IP.



Mode flux d'octets. Le module TCP est configuré avec un paramètre appelé **MSS (Maximum segment size)**. La MSS indique la taille de la charge utile des segments TCP. La valeur

de la MSS est déterminée pour que les trames qui transporteront les segments TCP soient pleines. Dans le cas de Ethernet, la MSS est donc égale à 1460 octets, la MTU de Ethernet étant 1500 octets auxquels on soustrait la taille des entêtes IP et TCP longs de 20 octets chacun si sans option. Envoyer des trames pleines est une des raisons qui permet à TCP de rendre un service efficace. En l'absence d'octets en nombre suffisant dans ses tampons d'émission, TCP attend la durée d'un temporisateur à l'expiration duquel un segment sera formé, quelle que soit la taille de sa charge utile. Ce temporisateur évite à TCP d'attendre indéfiniment si l'application n'a plus d'octets à envoyer.



Mode non connecté (UDP) vs. mode connecté (TCP)

Une **connexion** fait référence aux états qu'un protocole installe

pour rendre les services résultant de son exécution. Les états font référence à des variables dont la valeur caractérise l'état de l'échange entre entités impliquées dans l'exécution du protocole. C'est en exploitant ces états que des protocoles comme TCP rendent un service avec des garanties, notamment en termes de fiabilité.

TCP. Dans le cas de TCP, ces états sont appelés **Transmission control block (TCB)**. Les TCB font référence aux tampons d'émission et de réception, aux numéros de séquence concernant par exemple le dernier octet reçu, la taille courante des fenêtres de contrôles de flux et de congestion, la valeurs des temporiseurs, etc.

Les entités TCP installent et initialisent leur TCB lors d'une phase d'établissement qui fait intervenir des segments spécifiques appelés SYN. La valeur des états évoluent tout au long de l'échange. Les TCB sont supprimés lors d'une phase de libération faisant intervenir des segments spécifiques appelés FIN.

Les TCB associés à une connexion TCP sont identifiés par un quadruplet comprenant les numéros de port source et destination et les adresses IP source et destination qui identifient respectivement les processus applicatifs client et serveur et les machines hôtes qui les hébergent.

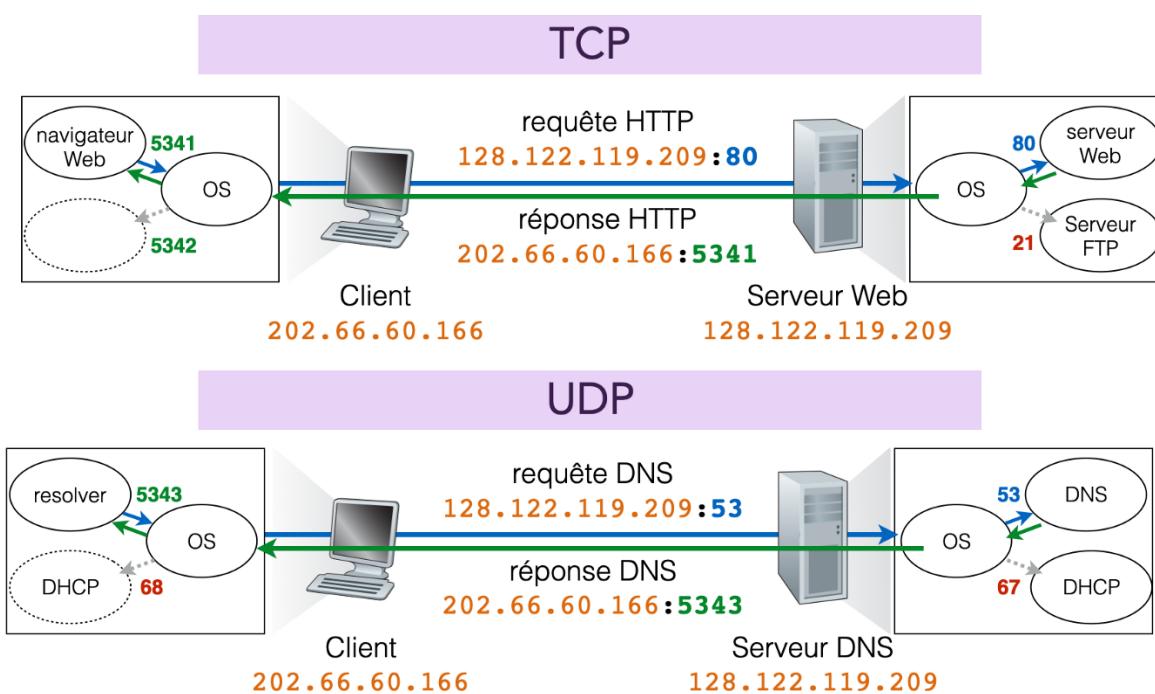
UDP. UDP est un protocole dont les fonctions ne nécessitent pas

d'état. UDP envoie ses datagrammes sans attendre la compléion d'une phase d'établissement de connexion.

Services offerts par la couche Transport

Services communs à UDP et TCP. UDP et TCP offrent les deux services de base suivants :

- **Multiplexage/démultiplexage** : un identifiant appelé **numéro de port** identifie les processus applicatifs qui produisent et consomment les octets de données que la couche transport prend en charge. Grâce au numéros de port, une machine hôte peut envoyer et recevoir des octets de données destinés à des processus qu'elles exécutent parallèlement.



Utilisation des numéros de port dans le multiplexage/démultiplexage UDP et TCP.

- **Détection d'erreur** : les entêtes UDP et TCP comprennent un champ **Somme de contrôle (checksum)** qui permet de vérifier l'intégrité des octets de l'entête et ceux de la charge utile. Nous verrons que la somme de contrôle permet également de vérifier des informations propres au protocole IP.

Services spécifiques à TCP. TCP offre les services spécifiques suivants :

- Détection des pertes d'octets.
- Correction des octets détectés en erreur ou perdus.
- Remise en séquence des octets reçus.
- Suppression des octets dupliqués.
- Contrôle de flux.
- Contrôle de congestion.

Numéros de port

Un numéro de port est une valeur binaire longue de 16 bits qui identifie un processus applicatif. La portée des numéros de port est locale : un numéro de port est garantie unique pour les seuls processus exécutés sur une même machine hôte. Un même numéro de port peut donc identifier deux processus si hébergés par deux machines distinctes. Pour identifier un processus de manière unique, l'adresse IP de la machine hôte qui héberge le processus doit être associé au numéro de port de ce processus.

Numéros de port côté client. Les numéros de ports choisis côté client sont supérieurs à 1024 et sont généralement tirés aléatoirement. Le choix de leur valeur est généralement laissé au système d'exploitation. Les seules exceptions concernent les services système tel que DHCP. Les clients utilisent le port 68 pour identifier leur processus DHCP.

NAT exploite le fait que la valeur des numéros de port côté client n'ait pas de signification particulière. NAT modifie leur valeur pour identifier de manière unique les machines hôtes configurées avec une adresse IP privée.

Numéros de port réservés (well-known). Les valeurs inférieures à 1023 sont réservées aux applications populaires de l'Internet tels que le Web et le mail. Elles identifient les processus côté serveur qui résultent de l'exécution de ces applications. Des exemples de numéros de port bien connus sont :

- 20/21 pour FTP
- 22 pour SSH
- 23 pour Telnet
- 25 pour SMTP, 143 pour IMAP, 110 pour POP3
- 80 pour le Web

Comme pour les clients, certains numéros de port côté serveur font référence à des services systèmes tels que :

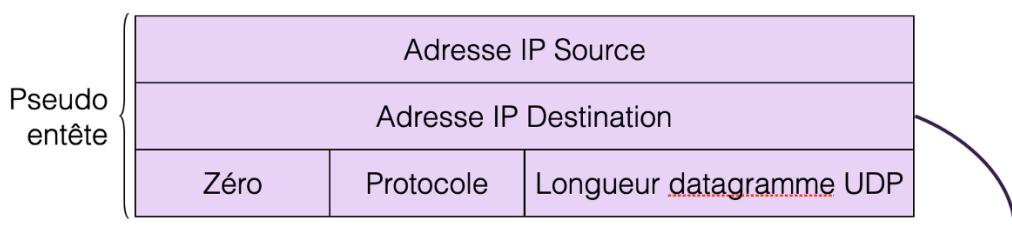
- 53 pour DNS
- 67 pour DHCP
- 123 pour NTP

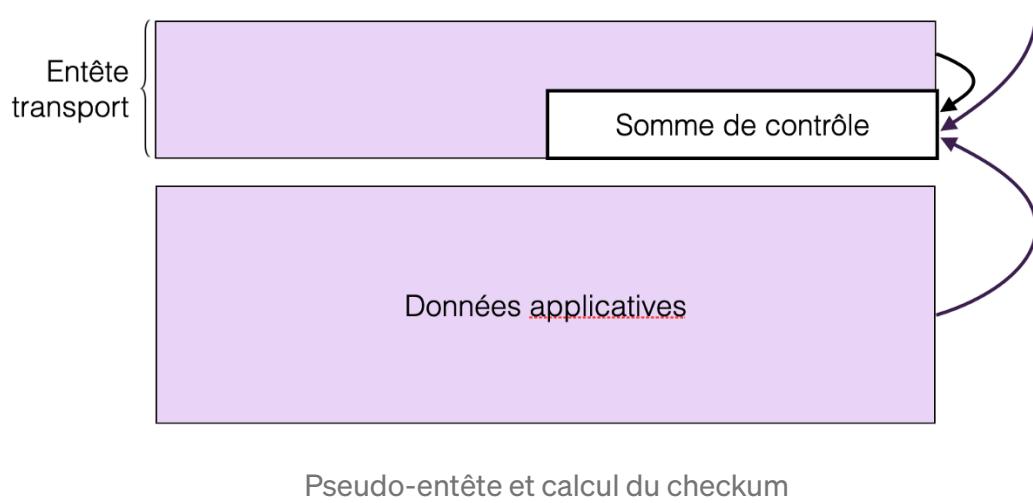
Ces numéros de port ont été fixés arbitrairement côté serveur afin d'affranchir les clients d'un mécanisme de résolution spécifique. Les pare-feux exploitent la valeur des numéros de port réservés pour bloquer ou autoriser les trafics échangés dans le cadre d'applications spécifiques.

Somme de contrôle

TCP et UDP détectent les erreurs grâce à une somme de contrôle située dans l'entête des segments TCP et des datagrammes UDP. La somme de contrôle permet de vérifier :

- l'intégrité de l'entête TCP et UDP,
- l'intégrité de la charge utile des segments TCP et des datagrammes UDP,
- l'intégrité de certains champs d'entête du paquet IP encapsulant le segment TCP ou le datagramme UDP. Ces champs IP font référence à un entête factice appelé **pseudo-entête**.





Le pseudo-entête contient :

- les adresses IP source et destination,
- la valeur du champ Protocole (qui vaut 0x11 dans le cas de UDP et 0x06 dans le cas de TCP), et
- la longueur totale du datagramme UDP ou du segment TCP.

La somme de contrôle est calculée par la source avant l'envoi du segment TCP (datagramme UDP). Pour ce faire, TCP (UDP) construit et ajoute le pseudo-entête en tête du segment TCP (datagramme UDP). Le calcul du checksum consiste à sommer les bits du segment (datagramme) muni du pseudo-entête pris 16 à 16. Les bits de retenue sont injectée dans la somme. La valeur finale utilisée dans le champ Checksum est le complément à 1 du résultat de cette somme. Une fois la somme de contrôle calculée, TCP (UDP) retire le pseudo-entête et envoie le segment (datagramme).

10.0.0.1

10.0.0.2	
00 00	12
5341	80
12	54964
00 01	00 02

←———— 16 bits —————→

10.0	→	00001010 00000000
0.1	→	00000000 00000001
10.0	→	00001010 00000000
0.2	→	00000000 00000010
0000	→	00000000 00000000
12	→	00000000 00001100
5341	→	00010100 11011101
80	→	00000000 01010000
12	→	00000000 00001100
0001	→	00000000 00000001
0002	→	00000000 00000010
Sum	→	00101001 01001011
Checksum	→	11010110 10110100

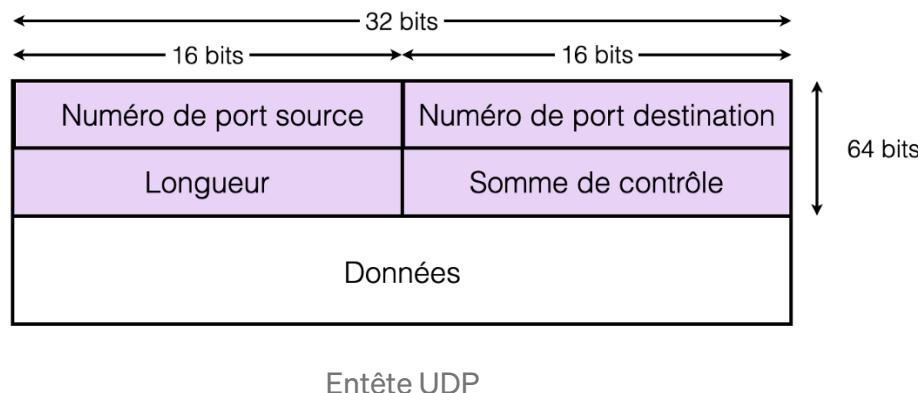
Calcul du checksum UDP.

Pour vérifier l'intégrité d'un segment (d'un datagramme), le récepteur calcule cette somme sur le segment (datagramme) après construction et ajout du pseudo-entête. Du fait d'inclure la valeur du checksum telle que reçue dans le segment (datagramme), les 16 bits résultant de cette somme sont tous à 1 pour un segment (datagramme) sans erreur. La présence d'un ou plusieurs bits à 0 indique que le segment (datagramme) est en erreur.

User datagram protocol UDP

Une application sollicite les services de UDP en écrivant les octets de données dans un tampon d'émission. UDP prélève les

octets de données résultant d'une écriture et y ajoute une entête longue de 8 octets.



Entête UDP

L'entête UDP est longue de 8 octets et comprend quatre (4) champs long de 2 octets (16 bits) chacun :

- Numéro de port source (16) identifie le processus à l'origine des octets de données contenues dans le datagramme UDP.
- Numéro de port destination (16) identifie le processus à qui sont destinés ces octets de données.
- Longueur (16) contient la longueur totale du datagramme en octets.
- Somme de contrôle (16) contient la valeur qui permet de vérifier l'intégrité du datagramme UDP ainsi que celle du pseudo-entête qui contient la valeur de certains champs IP.

Services rendus par UDP

UDP rend un service en mode non connecté. UDP évite ainsi la

complexité découlant des états auxquels fait référence une connexion. De ce fait, UDP rend un service sans garantie, similaire à IP.

- Conséquence 1 : Les datagrammes UDP sont envoyés sans attendre l'établissement d'une connexion. UDP évite également la complexité liée à la gestion des états qui auraient été associés à une éventuelle connexion.
- Conséquence 2 : UDP est non fiable. Les datagrammes UDP perdus ne sont jamais retransmis. De ce fait, l'envoi des datagramme UDP n'est pas retardé par un temporisateur dont il faudrait attendre l'expiration. Les datagrammes reçus sont ceux qui ont été envoyés une première fois. Aucun d'eux n'a fait l'object d'une retransmission provoquée par l'expiration du temporisateur. Tel aurait le cas si UDP réparait la perte des datagrammes.
- Conséquence 3 : UDP est agnostique à la congestion dans le réseau. UDP consomme toute la bande passante du réseau sans se soucier des autres sources qui pour certaines, diminuent leur débit lorsqu'elles enregistrent des pertes. C'est le cas des sources TCP.
- Conséquence 4: Les récepteurs UDP ne contrôlent pas le débit d'émission des sources UDP. Les datagrammes peuvent donc engorger les récepteurs et être perdus si leurs tampons de réception viennent à manquer.

UDP ajoute au service offert par IP, les deux services suivants :

- Le multiplexage et démultiplexage des octets de données en se basant sur les numéros de port source et destination,
- La détection des datagrammes en erreur grâce à la somme de contrôle.

UDP hérite donc des manquements de la couche IP : un datagramme peut être perdu, dupliqué, reçu en erreur et/ou dans le désordre tout comme peuvent l'être les paquets IP.

Applications populaires fonctionnant au-dessus d'UDP

Applications à contraintes temporelles. Du fait de ne pas être fiable, UDP est adapté aux applications à contraintes temporelles fortes telles que les applications de streaming multimédia. En effet, la réparation des données perdues nécessite des retransmissions qui retardent inexorablement la réception des données. De plus, les utilisateurs d'applications multimédia ne sont pas ou sont peu sensibles aux pertes. C'est le cas par exemple, des applications de streaming multimédia tel que IPTV, les applications téléphoniques, de téléconférences, ou de jeux en ligne.

Services systèmes incompatibles avec TCP. UDP est également utilisé par certains services exécutés au-dessus de la couche Transport faute de pouvoir utiliser TCP.

Il s'agit par exemple des services rendus par des serveurs en nombre réduit comparé à celui des clients que ces serveurs desservent. Des clients en trop grand nombre empêchent

l'utilisation de TCP en raison des mémoires que requiert les états installés par TCP, les services de ce dernier étant offerts en mode connecté. C'est le cas par exemple de DNS. Dans le cas de DNS, les échanges se limitent à deux messages : une requête pour laquelle une réponse unique est attendue. De ce fait, le service est rendu en un aller-retour. Utiliser une connexion aurait retardé cet aller-retour en introduisant préalablement, un aller-retour supplémentaire.

Un autre exemple de services incompatibles avec les services de TCP sont ceux offerts aux clients qui n'ont pas d'adresse IP. TCP installe aux deux extrémités d'une connexion, des états identifiés par un quadruplet comprenant les numéros de port et les adresses IP des deux extrémités de la connexion. Si une de ces extrémités n'a pas d'adresse IP, cela compromet l'utilisation de TCP. C'est le cas des clients DHCP configurés avec l'adresse IP non-spécifiée 0.0.0.0.

Si l'utilisation de UDP est adaptée aux applications interactives, il est à noter que les fonctions des box Internet ne sont pas compatibles avec UDP. En effet, NAT ne sait pas laisser passer les datagrammes UDP. De plus, les pares-feux bloquent par défaut le trafic UDP.

Transmission control protocol (TCP)

Les propriétés de TCP sont les suivantes :

- TCP offre un service orienté flux d'octets. L'application émettrice écrit ses octets dans un tampon en émission. TCP y prélève une quantité d'octets appelée **MSS (Maximum segment size)**, indépendante des blocs écrits par l'application. La MSS dépend de la MTU locale
- TCP est orienté connexion : TCP ouvre (ferme) une connexion qui entraîne l'installation (la libération) des états nécessaires pour rendre un service garantie. Ces états font référence à des variables contenues dans des blocs mémoires appelés **TCB (Transmission control block)**. L'installation (la libération) de ces états est provoquée par la réception de segments particuliers appelés **SYN (FIN)**.
- TCP offre un service de **livraison d'octets fiable et en séquence** : TCP utilise une **somme de contrôle** pour détecter les octets en erreur. TCP **numérote en séquence** les octets de données pour détecter leur perte et les réordonner. TCP utilise des segments particuliers appelés **ACK** (accusés de réception) et des **temporiseurs et retransmissions** pour réparer les octets perdus et corriger ceux reçus en erreur. TCP évite les pertes provoquées par l'engorgement des récepteurs en contrôlant le débit des sources : c'est le rôle du **contrôle de flux**. TCP évite les pertes qui provoque la congestion dans le réseau en adaptant le débit d'émission des sources à la charge du réseau : c'est le rôle du **contrôle de congestion**.

Fiabilité

TCP rend un service fiable en mettant en œuvre les mécanismes suivants :

- **Etablissement/libération de connexion.** TCP repose sur la présence d'états installés au niveau du client et du serveur. Ces états font référence à des variables maintenues dans des emplacements mémoires appelés TCB (Transmission control block). Client et serveur identifient ces états par un quadruplet contenant leurs numéros de port et adresses IP respectifs.
- **Détection des erreurs.** L'entête TCP contient un champ somme de contrôle portant sur le segment entier (entête et données) et sur l'entête IP partiel (pseudo-entête)
- **Détection des pertes et remise en ordre des octets.** TCP numérote les octets en séquence et utilise des tampons de réception où les octets reçus hors séquence sont mis le temps que le trou de séquence soit comblé.
- **Correction des pertes et des erreurs.** Un émetteur TCP enclenche un temporisateur de retransmission pour chaque segment de données envoyé. Sur réception d'un segment de données, un récepteur TCP retourne un segment de type particulier appelé accusé de réception cumulatif (ACK).

Efficacité

L'efficacité de TCP résulte des actions suivantes :

- **Construction des segments TCP.** La MSS est un paramètre

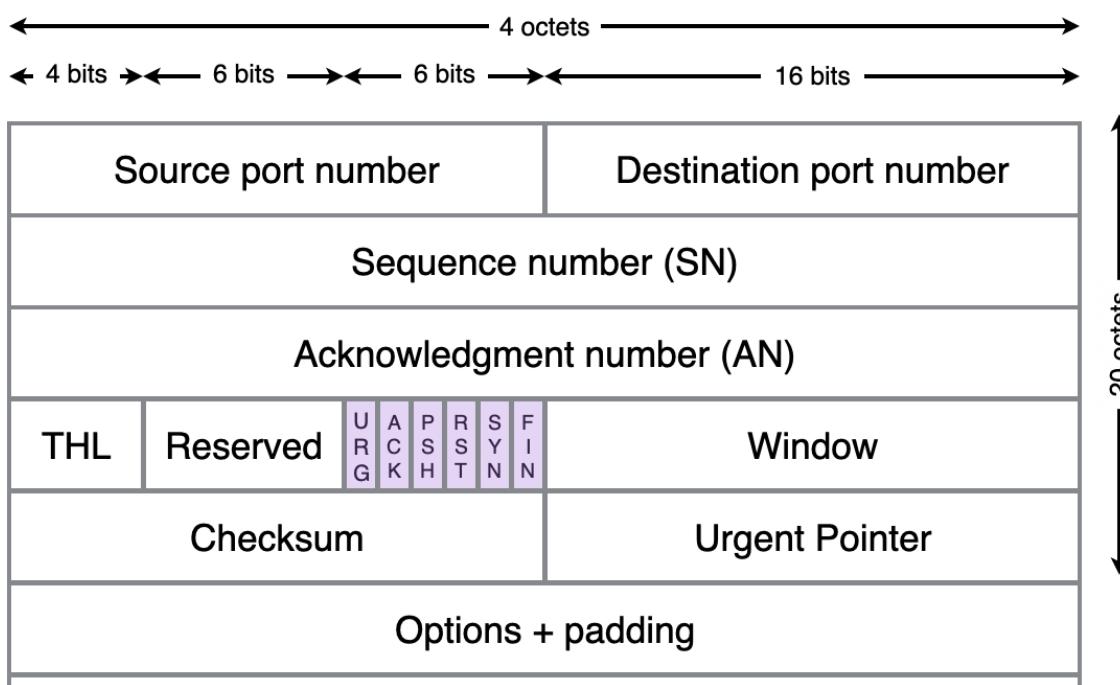
TCP qui indique la charge utile des segments. La MSS est déterminée en fonction de la MTU locale. La MTU locale représente la taille du champ Données des trames qui encapsuleront les segments après ajout de l'entête IP. C'est grâce à la MSS que TCP assure l'envoi de trames pleines tout en évitant la fragmentation des paquets IP contenant des segments TCP.

- **Évitements des pertes.** TCP évite les pertes en limitant la quantité d'octets de données qu'un émetteur TCP peut envoyer en un coup. Ces octets sont ceux qui appartiennent à deux fenêtres : la fenêtre de **contrôle de flux** dont la taille correspond aux tampons de réception libres où le récepteur placera les octets de données en attente, le temps que l'application réceptrice décide de les lire. La seconde fenêtre est la **fenêtre de congestion** dont la taille est déterminée en fonction de la bande passante disponible le long du chemin emprunté par les segments. C'est l'intersection des octets couverts par ces deux fenêtres qu'un émetteur TCP peut envoyer sans attendre la réception d'un segment ACK.
- **Partage équitable de la bande passante.** Le débit d'émission des émetteurs TCP découlent de la taille des fenêtres de flux et de congestion. Dans le cas de la fenêtre de congestion, son calcul assure une répartition équitable de la bande passante entre émetteurs TCP dont les segments empruntent le même chemin.

Entête TCP

Longueur d'entête. L'entête TCP sans option a une longueur minimale de 20 octets. A la suite de ces 20 octets, des options TCP peuvent occuper une longueur maximale de 40 octets. Un entête avec options doit respecter l'alignement de sa longueur sur des mots de 32 bits. En d'autres termes, la longueur de l'entête TCP exprimée en octets est un multiple de 4. Si la longueur des options n'est pas suffisante pour garantir l'alignement de l'entête, l'option End of option list (EOL) permet d'ajouter autant d'octets de bourrage (0x00) que nécessaire.

Drapeaux et types de segment. TCP définit plusieurs types de segment différenciés par la valeur combinée des six (6) drapeaux lus de gauche à droite : FIN, SYN, RST, PSH, ACK et URG. L'utilisation des drapeaux comme differentiateurs de segment impose l'utilisation d'un format d'entête unique pour tous les segments TCP.

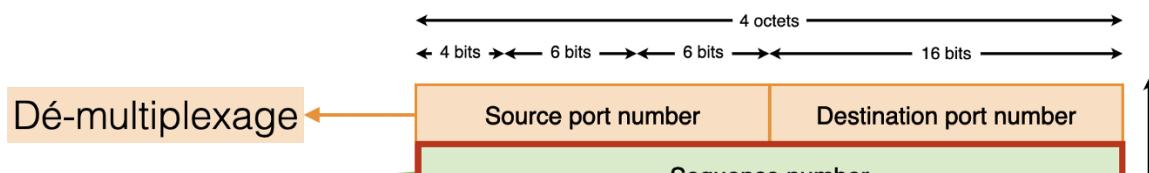


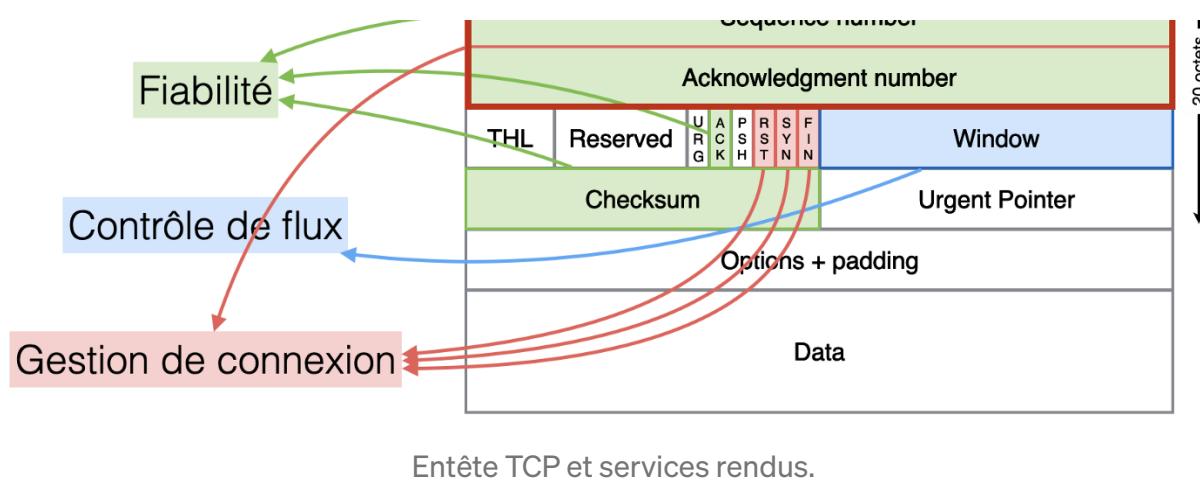
Data

Entête TCP et drapeaux TCP.

- **Les drapeaux FIN, SYN et RST** sont positionnés pour les segments chargés de gérer la connexion TCP et les états qui l'accompagnent. Une connexion TCP est établie à l'initiative du client qui envoie un segment où est positionné le drapeau SYN. Sur réception de ce premier SYN, le serveur répond à son tour avec un segment où les drapeaux SYN et ACK sont positionnés.
- **Le drapeau ACK** indique qu'il s'agit d'un segment d'accusé de réception. Ces segments sont utilisés pour fiabiliser l'envoi des segments TCP, hormis les accusés de réception seuls (segment vide où le drapeau ACK est le seul positionné).
- **Les drapeaux PSH et URG** régissent les interactions entre applications et TCP en déterminant à quel instant et quels octets lire ou écrire dans les tampons d'émission et de réception de TCP.

Aucun drapeau n'a été défini pour identifier les segments de données. Il s'agit du type de segment TCP par défaut.





Entête TCP et services rendus.

L'entête TCP est composé de champs qui interviennent dans la fourniture des services attendus de TCP :

- **Multiplexage/démultiplexage** : les champs numéros de port source et destination autorisent l'émission et la réception simultanées d'octets émanant de processus applicatifs concurrents, exécutées sur une même machine hôte.
- **Contrôle des pertes** : les numéros de séquence et d'acquittement permettent la détection et la réparation des segments perdus.
- **Contrôle des erreurs** : le champ Checkum permet la détection des segments en erreur.
- **Remise en séquence** : les numéros de séquence permettent de réordonner les segments reçus hors séquence dans leur ordre d'émission.
- **Contrôle de flux** : le champ Window permet à un récepteur TCP d'indiquer à la source, la quantité d'octets que le

récepteur peut recevoir. Cette quantité correspond aux tampons de réception libres où les octets de données seront mis en attente le temps que les applications décident de les lire.

- **Gestion de connexion** : les champs numéro de séquence et numéro d'acquittement combinés aux drapeaux SYN, FIN et RST permettent de gérer les connexions TCP.

Champs numéros de port source (16) et destination (16).

Le numéro de port source dans le cas d'un client contient une valeur tirée aléatoirement dont le choix est généralement laissé au système d'exploitation. Le numéro de port destination indique dans le cas d'un serveur la valeur bien connue associée à l'application qu'il exécute.

Champ numéro de séquence (32). Indique dans le cas des segments de données, le numéro de séquence du premier octet véhiculé par ce segment. Dans le cas d'un segment SYN, la valeur du champ numéro de séquence est appelé Initial sequence number ou ISN. Le numéro de séquence du premier octet de données envoyé dans les deux sens d'une connexion TCP est tiré aléatoirement. Ce sont les segments SYN qui annoncent au préalable la valeur du numéro de séquence qui, une fois la connexion établie, sera incrémenté et utilisé pour numérotter les octets de données.

Champ accusé de réception (32). Annonce le numéro de séquence du prochain octet de données attendu. Cette valeur

indique que les octets antérieurs à ce numéro ont été reçus correctement et en séquence. Ce numéro est valide à condition que le drapeau ACK soit positionné.

Champ Transport header length (THL) (5). Comme dans le cas de l'IHL de IP, ce champ indique la longueur de l'entête, exprimée en mots de 32 bits. C'est la valeur du THL qui indique si l'entête TCP contient des options et en quelle quantité. La valeur 0x5 (i.e., 20 octets) est la longueur minimale d'un entête TCP. Il s'agit d'un entête TCP sans option. 0xF indique la longueur maximale d'un entête TCP qui dans ce cas, contient 40 octets d'options.

Drapeaux (6). La valeur de ces bits indique quel est le type du segment TCP :

- SYN intervient dans la phase d'établissement d'une connexion ;
- RST permet de réinitialiser les paramètres de la connexion ;
- FIN intervient dans la phase de fermeture de la connexion ;
- ACK intervient dans la fiabilisation des segments échangés ;
- PSH force la lecture et l'envoi des octets de données côté émetteur, leur lecture côté récepteur ;
- URG permet l'envoi de données urgentes.

Champ Window (16). Ce champ est utilisé par un récepteur

pour indiquer la quantité des tampons de réception disponibles où les octets reçus seront mis en attente, le temps que les applications décident de les lire.

Champ Checksum (16). Ce champ contient la somme de contrôle portant sur le pseudo-entête et le segment TCP entier (entête et données). Il permet de détecter la présence de bits en erreur. Si tel est le cas, le segment est rejeté en silence.

Champ Urgent pointer (16). Le champ Pointeur urgent permet aux applications d'indiquer à TCP la présence dans les tampons d'émission, d'octets à envoyer en priorité. De ce fait, ces octets sont appelés urgents. Une fois reçus, les octets urgents sont lus en priorité sans attendre que les octets déjà reçus mais non urgents soient lus.

Champ options TCP. Une valeur de THL strictement supérieure à 5 indique la présence d'options TCP situées à la suite des 20 octets de l'entête fixe.

Ouverture et fermeture de connexion TCP

Les services de TCP sont sollicités par une application cliente qui tente d'obtenir les services offerts par un serveur. Pour rendre ces services, TCP repose sur des états que maintiennent dans leurs mémoires, client et serveur. Ces états font référence à des variables dont la valeur caractérise :

- les tampons d'émission et de réception,

- des numéros de séquence significatifs,
- les fenêtres de contrôles de flux et de congestion,
- des temporiseurs.

Ces états sont maintenus dans des emplacements mémoires appelés TCB (Transmission control block). Client et serveur identifient leur TCB respectif par un quadruplet contenant leur numéro de port et adresse IP respectifs. Une connexion TCP fait référence aux TCB identifiés par un même quadruplet.

Les TCB sont créés et initialisés lors d'une phase d'ouverture de connexion. Une fois leurs données échangées, client et serveur libèrent leurs mémoires en supprimant les états qu'elles contiennent lors d'une phase appelée fermeture de connexion TCP.

Ouverture de connexion. L'ouverture d'une connexion TCP résulte de l'envoi des trois (3) segments suivants :

- Client → Serveur : segment SYN
- Serveur → Client : segment SYN-ACK
- Client → Serveur : segment ACK

On parle de poignée de main à trois voies (3-way handshake ou 3WHS).

Une connexion TCP est créée à l'initiative d'un client qui

sollicite les services d'un serveur. Pour ce faire, le client envoie un **segment SYN** dont le numéro de port source est tiré aléatoirement et le numéro de port destination est connu par avance.

- La valeur du champ numéro de séquence du segment SYN appelée ISN (Initial sequence number) est tirée aléatoirement. Cette valeur une fois incrémentée, est celle qui numérotera le premier octet de données envoyé par le client.
- Le champ numéro d'accusé contient 32 bits tous à 0.

Sur réception de ce segment SYN, le serveur répond avec un **segment SYN-ACK**.

- La valeur du champ numéro de séquence contient lui aussi un ISN tiré aléatoirement dont la valeur une fois incrémentée, numérotera le premier octet de données envoyé par le serveur.
- Le champ numéro d'accusé contient la valeur de l'ISN du client incrémenté de 1.

Le client considère la connexion TCP établie une fois le segment SYN-ACK du serveur reçu. Le client envoie un **segment ACK** pour accuser la réception du segment SYN-ACK.

- Le champ numéro de séquence contient la valeur de l'ISN

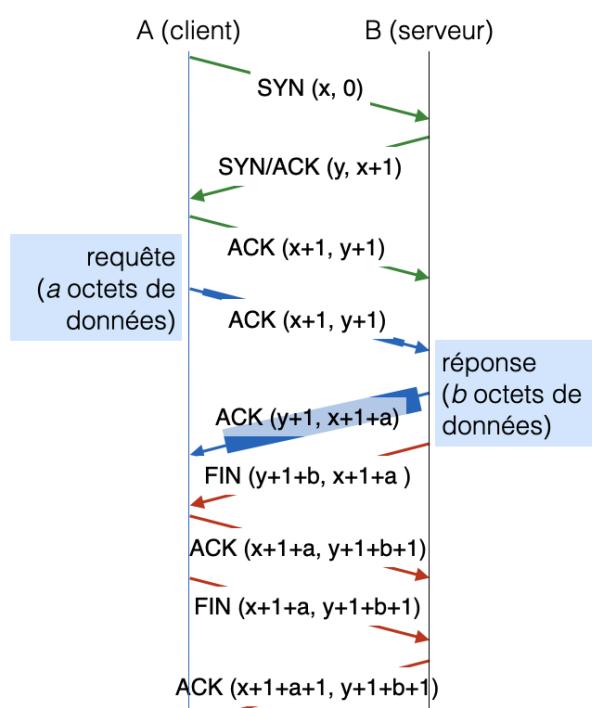
du client incrémentée de 1.

- Le champ numéro d'accusé contient la valeur de l'ISN du serveur incrémenté de 1.

Ce segment ACK peut contenir des octets de données. On parle alors de données piggybackées. Si le segment ACK est vide, la valeur de l'ISN du client incrémentée de 1 sera réutilisée dans le premier segment de données envoyé par le client.

Ce n'est qu'une fois le segment ACK du client reçu que le serveur considère la connexion TCP établie et peut dès lors envoyer des données.

Les segments SYN du client et du serveur contiennent des options TCP dont la valeur permet d'initialiser les états qu'ils installent.





Ouverture et fermeture de connexion TCP.

Fermeture de connexion. La fermeture d'une connexion TCP résulte de l'envoi de quatre (4) segments. On parle de poignée de mains à quatre (4) voies (4-way handshake ou 4WHS).

La fermeture est initiée par une des deux extrémités qui notifie ainsi qu'elle n'a plus de données à envoyer. On parle de **fermeture passive**. C'est généralement le cas du client qui une fois sa dernière requête envoyée, attend que le serveur lui envoie les données demandées. Le client envoie pour ce faire un **segment FIN**. L'envoi de ce premier FIN ne provoque pas la suppression des états côté client, le client pouvant encore recevoir ou retransmettre des données.

Le serveur envoie un **segment ACK** qui accuse la réception du segment FIN envoyé par le client et continue à envoyer des données. Ce n'est qu'une fois toutes ses données envoyées que le serveur enverra un **segment FIN** à son tour et attendra l'accusé de réception du client pour supprimer les états de la connexion. On parle de **fermeture active**.

Le client envoie un **segment ACK** qui accuse la réception du segment FIN envoyé par le serveur. Sur envoi du segment ACK, le client arme un temporisateur appelé **2MSL** (2 Maximum segment life). Le temporisateur 2MSL permet de prendre en

compte la perte éventuelle du segment FIN envoyé par le serveur ou du segment ACK envoyé par le client pour accuser le segment FIN du serveur. Sur expiration du temporisateur 2MSL, le client supprime tous les états de la connexion. La connexion TCP est dès lors complètement fermée et ses identifiants peuvent être réutilisés pour une nouvelle connexion.

Numéro de séquence

Un segment TCP contient un champ Numéro de séquence dont la valeur dépend du type de segment :

- **Segment SYN** : il s'agit de l'ISN qui est tiré aléatoirement. L'ISN incrémenté de 1 numérotera les octets de données. Le numéro d'acquittement du segment ACK qui accuse un segment SYN contient la valeur de l'ISN incrémenté de 1.
- **Segment FIN** : il s'agit du numéro de séquence courant qui permet d'identifier le segment ACK qui accusera sa réception. Le numéro d'acquittement du segment ACK contiendra la valeur du numéro de séquence du segment FIN incrémenté de 1.
- **Segment de données** : il s'agit du numéro de séquence du premier octet véhiculé dans la charge utile du segment de données.
- **Segment ACK** : la valeur du numéro de séquence d'un segment ACK n'est pas pertinente. Le numéro de séquence d'un segment ACK est donc laissé à la valeur courante du

numéro de séquence sans la consommer ; cette valeur sera réutilisée pour numérotter l'octet de données véhiculé dans le segment de données suivant.

Numéro d'acquittement

Les segments ACK accusent la réception de tous les segments hormis les segments vides où le drapeau ACK est le seul positionné. Pour identifier quel segment il acquitte, un segment ACK contient un champ Numéro d'acquittement :

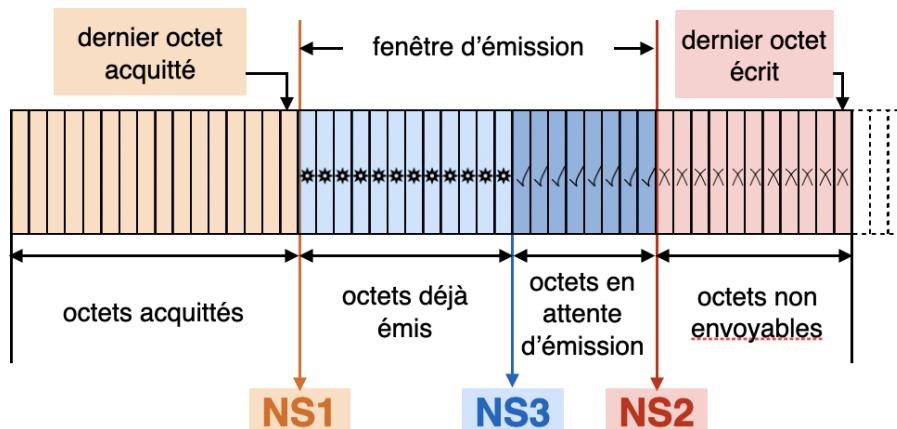
- **Segments SYN et FIN** : un segment SYN ou FIN dont le numéro de séquence vaut x est acquitté par un segment ACK dont le numéro d'acquittement vaut $x + 1$.
- **Segments de données (avec drapeaux URG et/ou PSH)**. Le segment ACK qui accuse un segment de données reçus sans erreur et en séquence, a un numéro d'acquittement égal au numéro de séquence du dernier octet que le segment de données véhicule incrémenté de 1. Un segment de données numéroté 10 et dont la charge utile est longue de 1460 octets est donc accusé par un segment ACK dont le numéro d'accusé vaut 1470.
- **Segments ACK**. Les segments ACK ne sont pas acquittés.

Contrôle de flux et fenêtre

TCP évite les pertes en limitant la quantité d'octets de données qu'un émetteur TCP peut envoyer en un coup. Ces octets sont ceux qui appartiennent à deux fenêtres :

- la fenêtre de **contrôle de flux** dont la taille correspond aux tampons de réception libres où le récepteur placera les octets de données en attente, le temps que l'application réceptrice décide de les lire.
- la fenêtre de **contrôle de congestion** dont la taille est déterminée en fonction de la bande passante disponible le long du chemin emprunté par les segments de données.

Une fenêtre fait donc référence à une liste d'octets numérotés en séquence. Ce sont les octets couverts par l'intersection de ces deux fenêtres qu'un émetteur TCP peut envoyer d'un coup.



Fenêtre de contrôle de flux.

Chaque fenêtre est caractérisée par la valeur du numéro de séquence de trois (3) octets particuliers :

1. NS1 : Le numéro de séquence de l'octet situé le plus à gauche. Il s'agit du premier octet de données non encore

acquitté.

2. NS2 : Le numéro de séquence de l'octet situé le plus à droite. Il s'agit du dernier octet de données que l'émetteur TCP peut envoyer sans attendre la réception d'un segment ACK.
3. NS3 : Le numéro de séquence du premier octet contenu dans la fenêtre et non encore envoyé. Par définition : $NS1 \leq NS3 \leq NS2$.

La valeur de ces numéros de séquence évoluent au cours du temps. De ce fait, on parle de **fenêtre glissante** ou coulissante.

- NS1 croît sur réception d'un segment ACK accusant la réception des octets contenus en début de fenêtre. Tous les octets situés à gauche de la fenêtre sont ceux qui ont déjà été acquittés.
- Dans le cas de la fenêtre de contrôle de flux, NS2 est mis à jour sur réception d'un segment TCP. NS2 indique l'octet dont le numéro de séquence est obtenu en sommant NS3 et la valeur du champ Window contenu dans le segment reçu. Dans le cas de la fenêtre de contrôle de congestion, NS2 est déterminée par deux algorithmes appelés **démarrage lent** (slow start) et **évitement de congestion** (congestion avoidance).
- NS3 croît au fur et à mesure de l'envoi des segments de données. A chaque envoi, la valeur du NS3 croît du nombre d'octets contenus dans le segment de données envoyé.

Drapeaux PSH et URG

Le drapeau PSH (Push) impacte les fonctions de l'émetteur et du récepteur :

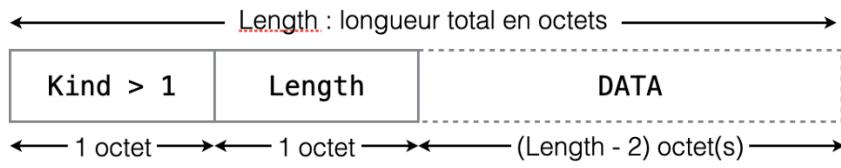
- Côté émetteur, l'application invite TCP à construire un segment sans attendre d'octets supplémentaires. L'application s'assure ainsi de l'envoi des octets, y compris si en nombre insuffisant pour former un segment plein (charge utile égale MSS octets) et ce, sans attendre l'expiration du temporisateur que TCP arme dans l'attente que MSS octets soient inscrits dans les tampons d'émission.
- Côté récepteur, l'application est notifiée de la présence d'octets dans les tampons de réception qu'elle est invitée à venir lire dès que peut se faire.

Le drapeau URG (Urgent) est positionné si le segment contient des octets de données que l'application a indiqué comme urgents. La quantité d'octets urgents est précisée dans champ Urgent pointer (16).

- Côté émetteur, les octets urgents sont envoyés sans attendre leur tour. Ces octets sont envoyés en priorité, y compris lorsque les tampons d'émission contiennent des octets non encore envoyés.
- Côté récepteur, l'application réceptrice lit les octets urgents en priorité sans attendre que les octets déjà reçus soient préalablement lus.

Options TCP

Un entête TCP peut contenir une ou plusieurs options. Tel est le cas des segments dont la valeur du champ Header length est strictement supérieur à 5. Tout comme la partie fixe de l'entête TCP longue de 20 octets, la longueur des options TCP doivent être alignée sur des mots de 32 bits. Pour ce faire, une option TCP particulière longue de 8 bit appelé **End of option list (EOL)** permet d'ajouter autant d'octets de bourrage (0x00) que nécessaire pour garantir l'alignement de l'entête TCP complète. Une seconde option TCP longue de 8 bits est l'option **No operation (NOP)** qu'on intercale entre deux options consécutives. L'option NOP (0x01) permet de forcer le début d'une option ou de sa valeur en début ou fin de ligne.



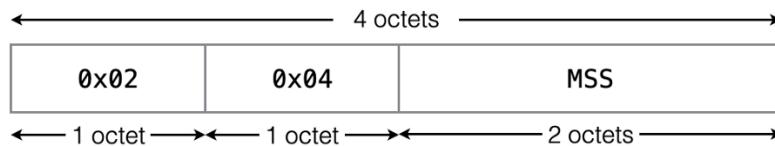
Format des options TCP de type > 1.

Une option TCP est caractérisée par son type aussi appelé *kind*. Le type d'une option est codé sur 8 bits. Les options TCP dont le type est strictement supérieur à 1 suivent un format Type-Length-Value (TLV) :

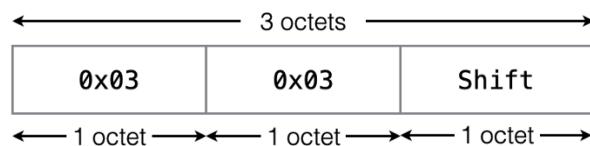
- **Type (8)** indique le type de l'option,
- **Length (8)** indique la longueur totale de l'option,
- **Value (>16)** indique la valeur de l'option.

Les options TCP les plus courantes sont les suivantes :

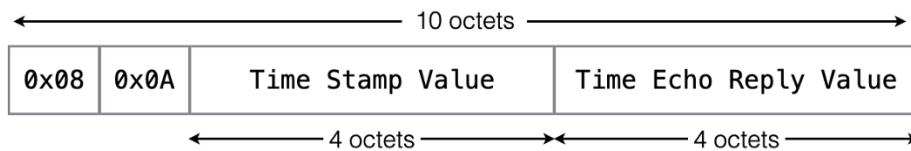
Maximum Segment Size (MSS)



Windows Scale (WSopt)



Time Stamp (TS)



Options TCP courantes.

Maximum segment size (MSS). Le type de cette option vaut 0x02. Elle est longue de 4 octets et sa valeur contient la longueur de la MSS. En annonçant cette valeur, un récepteur TCP indique la taille des segments qu'il souhaite recevoir. Cette valeur est déterminée en fonction des capacités de réception du récepteur. Un récepteur avec des tampons de réception en nombre limité privilégie des segments de petite taille.

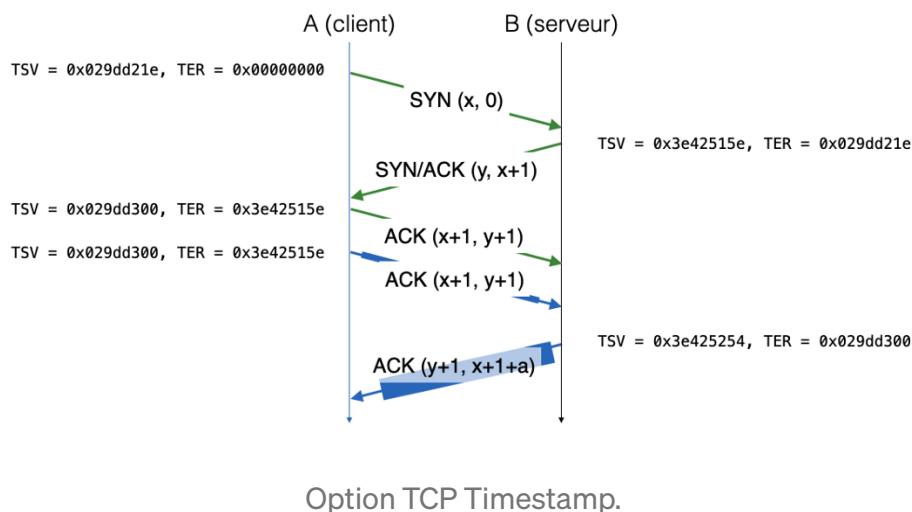
Window Scale (WSopt). Le type de cette option vaut 0x03. Elle est longue de 3 octets. Sa valeur contient le décalage à gauche à appliquer au champ Window. La valeur 1 indique un décalage de 1 bit, ce qui revient à doubler la valeur du champ

Window. L'option WSopt permet de spécifier des tailles de fenêtre supérieures à 2^{16} .

Timestamp (TS). Le type de cette option vaut 0x08. Elle est longue de 10 octets. Sa valeur est composée de deux champs :

- Le champ Time Stamp value (TSval) contient la valeur de l'horloge de l'émetteur à l'émission du segment.
- Le champ Time Echo Reply value (TSecr) contient une copie de la valeur du TSval du dernier segment reçu.

L'option TS permet de 1) calculer la valeur du temporisateur de retransmission et de 2) distinguer les octets non encore acquittés qui partagent le même numéro de séquence.



La figure suivante montre un exemple de segment SYN TCP. Ce segment contient 6 options parmi lesquels on trouve trois (3) occurrences de l'option NOP.

Port source : 0xFFAD, Port destination : 0X1389
 Numéro de séquence : 0x5C3E066A
 Numéro d'acquittement : 0x00000000
 THL : 0xA (40 octets), SYN : 1, Fenêtre : 0x4000
 Somme de contrôle : 0x9C2E, Pointeur urgent : 0x0000
 Option 1 : Type : 0x02 (MSS), MSS : 0x05A0 (1440)
 Option 2 : 0x01 (NOP),
 Option 3 : Type : 0x03 (WScale), Décalage : 0x00
 Option 4 : Type : 0x01 (NOP)
 Option 5 : Type : 0x01 (NOP)
 Option 6 : Type : 0x08 (Time Stamp), TSV : 0x9E7BC4,
 TERV : 0x000000

0xFFAD		0x1389	
0x5C3E066A			
0x00000000			
0xA002		0x4000	
0x9C2E		0x0000	
0x02	0x04	0x05A0	
0x01	0x03	0x03	0x00
0x01	0x01	0x08	0x0A
0x009E		0x7BC4	
0x0000		0x0000	

Exemple de segment TCP SYN véhiculant 6 options TCP.

Génération des segments ACK

Les fonctions de détection et de réparation des pertes est la charge des émetteurs TCP. Les récepteurs TCP se contentent d'accuser le dernier octet de données reçu sans erreur et en séquence en annonçant le numéro de séquence du prochain octet de données attendu. On parle de ‘dumb receiver’.

On distingue trois types d'ACK :

- **Les ACK retardés** : sur réception d'un segment de données, un récepteur attend pour une durée maximale de 500 ms, la réception d'un second segment. Sur réception de ce second segment, le récepteur envoie un **ACK cumulatif**. Si les 500 ms s'écoulent sans recevoir de second segment de données, le récepteur envoie un **ACK retardé**.
- **Les ACK dupliqués** : La réception d'un segment de données hors séquence (segment de données dont le numéro de séquence est supérieur au numéro de séquence attendu) entraîne un trou de séquence. Sur détection d'un trou de

séquence, le récepteur envoie immédiatement un segment ACK réclamant à nouveau l'octet qui n'a pas encore été reçu. On parle de **segment ACK dupliqué**. Il s'agit du numéro de séquence de l'octet le plus à gauche dans le trou de séquence.

- **Les ACK immédiats et partiels** : Sur réception d'un segment de données qui comblent le début d'un trou de séquence partiellement, le récepteur envoie immédiatement un segment ACK qui accuse la réception des seuls octets contenus dans ce segment de données. Si le segment de données comblent complètement le trou de séquence, le récepteur envoie immédiatement un segment ACK qui accuse les octets qui ont comblé le trou de séquence ainsi que tous ceux reçus hors séquence à la suite du trou de séquence.
- **Les ACKS sélectifs** : En présence d'un trou de séquence, les **ACK sélectifs** (SACK) permettent d'accuser les octets de données reçus hors séquence. Sur réception d'un segment de données qui introduit un trou de séquence, le récepteur envoie un SACK où il précise le premier et le dernier octet des données véhiculées par ce segment. L'émetteur découvre ainsi quels segments retransmettre.

Calcul du temporisateur de retransmission

Les émetteurs TCP arment un temporisateur sur envoi d'un segment de données. L'expiration de ce temporisateur avant réception du segment ACK qui acquitte la réception de ce

segment indique à l'émetteur la perte du segment de données ou du segment ACK. Il provoque donc la réception du segment de données.

La valeur du temporisateur de retransmission dépend du délai d'un aller-retour appelé **Round trip time ou RTT** entre l'émetteur et le récepteur.

- Un émetteur mesure RTT, la durée qui s'écoule entre l'envoi d'un segment de données non perdu et la réception du segment ACK correspondant.
- L'émetteur calcule SRTT, la moyenne pondérée des valeurs mesurées de RTT : $SRTT_i = a * SRTT_{i+1} + (1-a) * RTT$, où a un facteur de lissage (smoothing factor).

RTO, la valeur du temporisateur de retransmission est pris à deux fois la valeur du SRTT : $RTO = 2 * SRTT$. La valeur initiale du RTO utilisée pour un segment SYN est de 1 seconde.

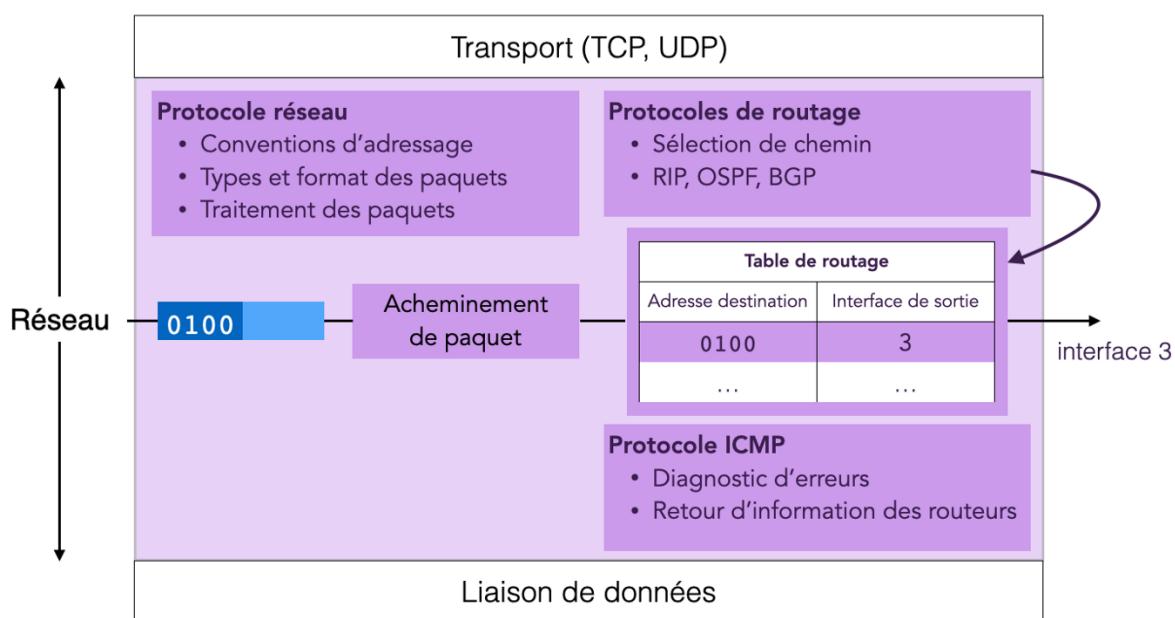
Couche Réseau (3) : Protocoles IP & ICMP

La couche réseau (3) dans Internet assure les fonctions d'acheminement (forwarding) des paquets IP. Ces fonctions consistent à envoyer un paquet IP au nœud suivant situé sur le chemin menant à la destination finale du paquet.

L'acheminement des paquets IP nécessite :

- l'identification des nœuds du réseau au travers d'une adresse IP
- la présence d'une table appelé table d'acheminement (forwarding table).

La construction de cette table peut être statique ou dynamique. Dans le cas des tables d'acheminement dynamiques, leur construction résulte d'un protocole de routage que la couche réseau exécute. En plus du protocole IP et du protocole de routage, la couche réseau dans Internet implémente le protocole ICMP qui permet aux machines de rendre compte des erreurs qui surviennent lors de l'acheminement ou la livraison des données.



La couche Réseau dans Internet.

Protocole IP

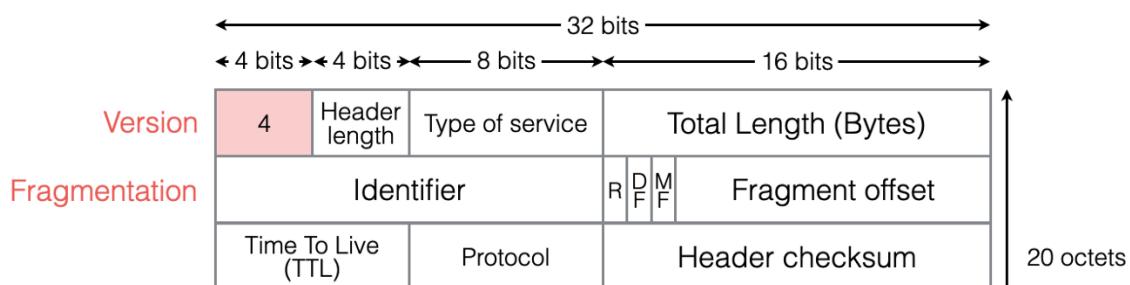
Le protocole IP définit :

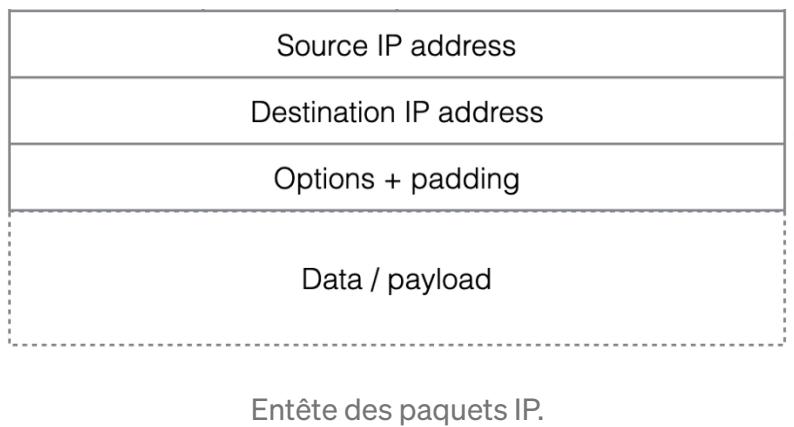
1. le format de l'entête des paquets IP,
2. le format des adresses IP,
3. les actions à entreprendre lors de l'émission et la réception d'un paquet IP.

Entête des paquets IP

L'entête des paquets IP a une longueur minimale de 20 octets. C'est le cas des paquets dont l'entête ne contient pas d'options. La longueur maximale de l'entête IP est 60 octets. Les 40 octets supplémentaires correspondent aux options IP que peut comprendre l'entête IP.

Une autre contrainte portant sur la longueur des entêtes IP concerne son alignement sur des mots longs de 32 bits (4 octets). En d'autres termes, la longueur des entêtes IP exprimée en octets doit être une valeur multiple de 4. Si la longueur des options IP ne suffit pas à garantir l'alignement de l'entête sur des mots de 4 octets, IP utilise une option IP spécifique appelée EOL (End of options list) pour ajouter autant d'octets de bourrage (0x00) que nécessaire.





Entête des paquets IP.

Les champs de l'entête IPv4 sont les suivants :

Version (4) : indique la version du protocole IP. La valeur 0x4 indique qu'il s'agit d'un paquet IPv4.

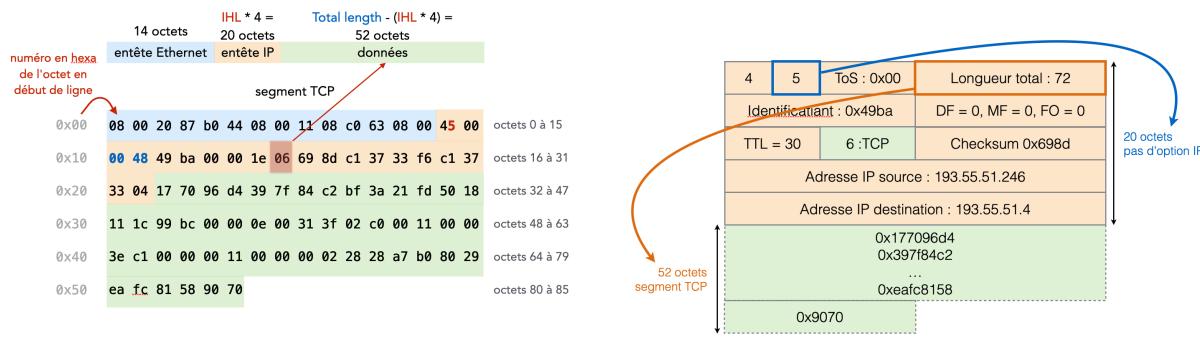
Internet header length (8) : indique la longueur de l'entête du paquet exprimée en mots de 32 bits. La valeur 0x5 (20 octets) indique que l'entête est sans option. La valeur 0xF (60 octets) indique que l'entête comprend 40 octets d'options.

Type of service (8) : les machines n'inspectent jamais la valeur du champ TOS. En effet, l'utilisation du champ TOS a été abandonnée. De ce fait, le champ TOS contient toujours la valeur 0x00.

Grâce au champ TOS, les routeurs sont censés rendre un service différencié, spécifique à la valeur indiquée dans le champ TOS. Le champ TOS est composé du sous-champ Precedence (3) suivi des 4 drapeaux D, T, R et C et d'un dernier bit toujours à 0. Le sous-champ Precedence contient une valeur qui indique l'importance d'un paquet. En cas de congestion, un routeur supprime en

priorité les paquets dont la précédence est la plus petite. De nos jours, les routeurs adoptent une approche FIFO en supprimant les derniers paquets reçus. Les drapeaux D (delay), T (throughput), R (reliability), et C (cost) indiquent le chemin le long duquel acheminer un paquet IP. Ces drapeaux supposent que les routeurs calculent pour une destination donnée, plusieurs chemins optimisant chacun, un des critères correspondant aux drapeaux D, T, R ou C. De nos jours, les routeurs calculent un seul chemin par destination.

Total length (16) : indique la longueur totale du paquet. Si 65536 est la valeur maximale que peut contenir ce champ, la longueur des paquets est limitée par un paramètre appelé la **MTU (Maximum transmission unit)**. Il s'agit de la longueur maximale du champ Données des trames qui encapsuleront les paquets IP. La MTU dans le cas de Ethernet vaut 1500 octets.



Header length et Total length.

Identifier (16), R (1), DF (1), MF (1) et Fragment Offset (13) : Ces champs sont utilisés lorsqu'un paquet IP dont la taille excède la valeur de la MTU locale, doit être fragmenté. La

fragmentation fera l'objet d'un paragraphe spécifique situé plus loin dans ce chapitre.

Time to live (8) : limite la durée de vie des paquets IP dans le réseau. Le champ TTL permet de supprimer les paquets emprisonnés dans une boucle d'acheminement. La valeur du champ TTL indique la longueur maximale du chemin qu'un paquet IP peut emprunter avant d'atteindre sa destination finale. La longueur de ce chemin représente le nombre de routeurs qui le constitue.

Les routeurs décrémentent de 1 le TTL des paquets qu'ils voient passer et suppriment ceux dont le TTL atteint la valeur 0. La suppression des paquets IP dont le TTL a atteint la valeur 0 provoque l'envoi d'un message ICMP Time exceeded.

Le TTL permet également de supprimer les paquets IP issus de la fragmentation d'un paquet IP pour lequel un récepteur n'a pas reçu au moins un fragment. A la façon des routeurs, le récepteur décrémente régulièrement la valeur du TTL des fragments déjà reçus et supprime ceux dont le TTL atteint la valeur 0. Une fois tous les fragments supprimés localement, le récepteur retourne un message ICMP Time exceeded.

La valeur initiale du TTL varie en fonction des systèmes d'exploitation : MacOS utilise la valeur 64, MS Windows la valeur 128, et Unix/Linux la valeur 64.

Protocol (8) : indique le type de l'entête situé à la suite de l'entête IP. La valeur 0x06 indique qu'il s'agit de l'entête TCP, 0x11 l'entête UDP, 0x01 l'entête ICMP.

Header checksum (16) : comprend la somme de contrôle (checksum) portant sur l'entête IP. La somme de contrôle permet la détection des bits en erreur dans l'entête IP. La somme de contrôle est calculée par la source avant l'envoi du paquet IP.

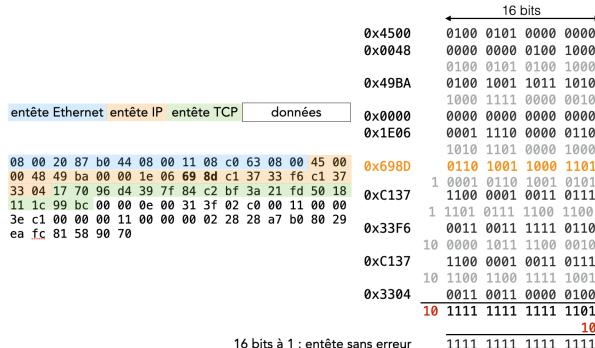
Le calcul du checksum consiste à sommer les bits de l'entête pris 16 à 16. Les bits de retenue sont injectés dans la somme. La valeur finale utilisée dans le champ Header checksum est le complément à 1 du résultat de cette somme.

Pour vérifier l'intégrité de l'entête des paquets qu'il reçoit, le récepteur calcule cette somme sur l'entête en incluant la valeur du checksum telle que reçue dans l'entête des paquets. Si les 16 bits du résultat sont tous à 1, le récepteur en déduit que l'entête est sans erreur, sinon le récepteur rejette le paquet silencieusement.

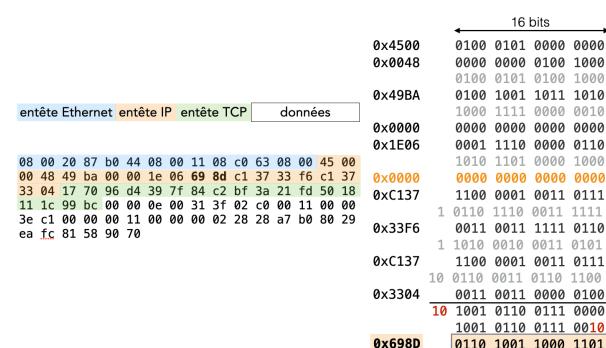
De nos jours, les routeurs ne vérifient pas la somme de contrôle des paquets qu'ils acheminent. Du fait de modifier la valeur du TTL, les routeurs se contentent de reporter cette modification en mettant à jour la somme de contrôle des paquets. Du fait que les routeurs n'inspectent pas le champ TTL, celui-ci a disparu de l'entête des paquet IPv6. La vérification de l'intégrité de

l'entête d'un paquet IPv6 est à la charge du protocole encapsulé par le paquet IPv6.

Vérification du checksum



Calcul du checksum



Calcul et vérification du checksum.

Source IP address (32) : cette adresse identifie la source à l'origine du paquet IP. La valeur de ce champ est modifiée par NAT si la source est connectée à un réseau privé.

Destination IP address (32) : cette adresse identifie la destination du paquet IP. La valeur de ce champ est modifiée par NAT si la destination est connectée à un réseau privé.

Fragmentation IP

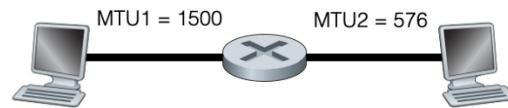
Le module IP est configuré avec un paramètre appelé la MTU (Maximum transfer unit) qui dépend du protocole de couche Liaison de données. La valeur de la MTU indique la longueur maximale du champ Données des trames qui encapsuleront les paquets IP. Si la longueur d'un paquet IP excède la valeur de la MTU, IP doit fragmenter le paquet IP sans quoi ce paquet sera supprimé. Une fois fragmenté, le réassemblage du paquet IP

aura lieu au niveau du récepteur final.

- **Le bit DF** (Don't fragment) indique si la source du paquet a autorisé sa fragmentation. Tel est le cas si le bit DF n'est pas positionné (DF = 0). IP doit alors générer des fragments dont la longueur hors entête doit être un multiple de 8. IP supprime les paquets avec le bit DF positionné à 1 et dont la taille excède la MTU locale. La source à l'origine d'un paquet trop gros est informée de sa suppression par l'envoi d'un message ICMP Destination unreachable.
- Les paquets IP issus de la fragmentation d'un paquet IP sont tous identifiés par la valeur du **champ Identifier** que tous héritent du paquet IP original. C'est la valeur du champ Identifier qui indique au récepteur les paquets issus de la fragmentation d'une même paquet IP.
- **Le bit MF** (More fragment) est positionné à 0 s'il s'agit d'un paquet non fragmenté ou du dernier fragment. Le bit MF des fragments précédents est positionné à 1.
- **Le champ Fragment Offset** permet d'ordonner les fragments dont le bit MF est positionné à 1. La valeur du Fragment Offset indique la position exprimée en mots de 8 octets du premier bit de données du fragment par rapport aux données du paquet IP initial avant fragmentation. Le Fragment Offset du premier fragment vaut 0, celui du dernier fragment correspond à la longueur totale des autres fragments divisée par 8. Si la valeur du Fragment Offset est divisée par 8, c'est pour prendre en compte la longueur de

ce champ. Le champ Fragment Offset a une longueur de 13 bits. Pour être interprétée efficacement, sa valeur est mise sur 16 bits en décalant de 3 bits sa position à gauche (ajout de 3 bits à 0 à droite), ce qui revient à multiplier cette valeur par 2^3 soit 8.

- **Flags** : 3 bits (Réservé : 0, DF, MF)
 - DF : Don't Fragment (les paquets trop grands sont rejetés)
 - MF : More Fragment (positionné si dernier fragment)
- **Fragment Offset** :
 - taille en octets hors entête des fragments précédant le fragment courant divisée par 8
- **Exemple :**
 - Données encapsulées : 1300 octets
 - Entêtes des fragments sur le réseau 2 :
 - $576 - 20 = 556$, valeur multiple de 8 la plus proche : $552 = 69 * 8$
 - F1 : offset 0 MF = 1 (taille des données : 552 octets)
 - F2 : offset 69 = $552/8$ MF = 1 (taille des données : 552 octets)
 - F3 : offset $69*2$ MF = 0 (taille des données : 196 octets)



Exemple de fragmentation IP.

Options IP

Une entête peut contenir une ou plusieurs options. Tel est le cas des paquets dont la valeur du champ Header length est strictement supérieure à 5. Tout comme la partie fixe de l'entête IP longue de 20 octets, la longueur totale des options IP doivent être alignée sur des mots de 32 bits. Pour ce faire, une option IP particulière appelé **End of option list (EOL)** permet d'ajouter autant d'octets de bourrage (0x00) que nécessaire pour garantir l'alignement de l'entête IP complète.

Une option IP est caractérisée par son type aussi appelé valeur.

Le type d'une option est codé sur 8 bits.

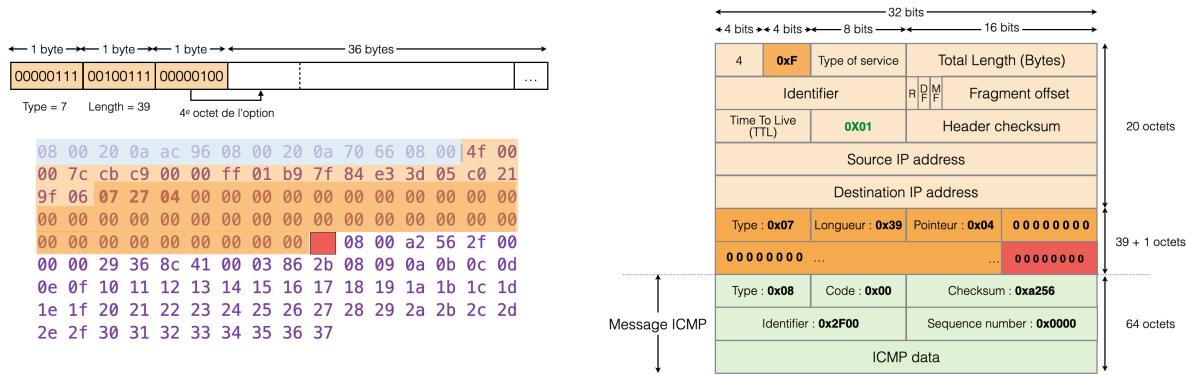
- Le premier bit si positionné à 1 indique que l'option doit être copiée dans l'ensemble des fragments en cas de fragmentation du paquet original.
- La valeur des deux bits suivants indique la classe de l'option. La valeur 0 indique qu'il s'agit d'une option de contrôle et la valeur 2, qu'il s'agit d'une option utilisée à des fins de débogage ou de mesures.
- Les cinq (5) derniers bits indiquent le numéro de l'option.

Le type de l'option Record Route est 0x07 (0b00000111). Cette valeur indique que l'option Route Record est copié uniquement dans le premier fragment en cas de fragmentation du paquet. La classe indique qu'il s'agit d'une option de contrôle. Le numéro de l'option vaut 7 et est identique à son type.

Le format d'une option IP définit trois champs, le premier champ étant toujours présent, les deux derniers étant optionnels :

- le champ Type (8) appelé aussi valeur,
- le champ Longueur (8) qui porte sur l'ensemble de l'option,
- le champ Données qui dépend du type de l'option. La longueur du champ Données est égale à la valeur du champ Longueur moins 2.

L'exemple suivant montre la trace d'un paquet IP dont l'entête contient deux options : l'option Record Route suivi de l'option EOL. Ce paquet encapsule un message ICMP Echo Request.



Option Record Route.

Protocole ICMP

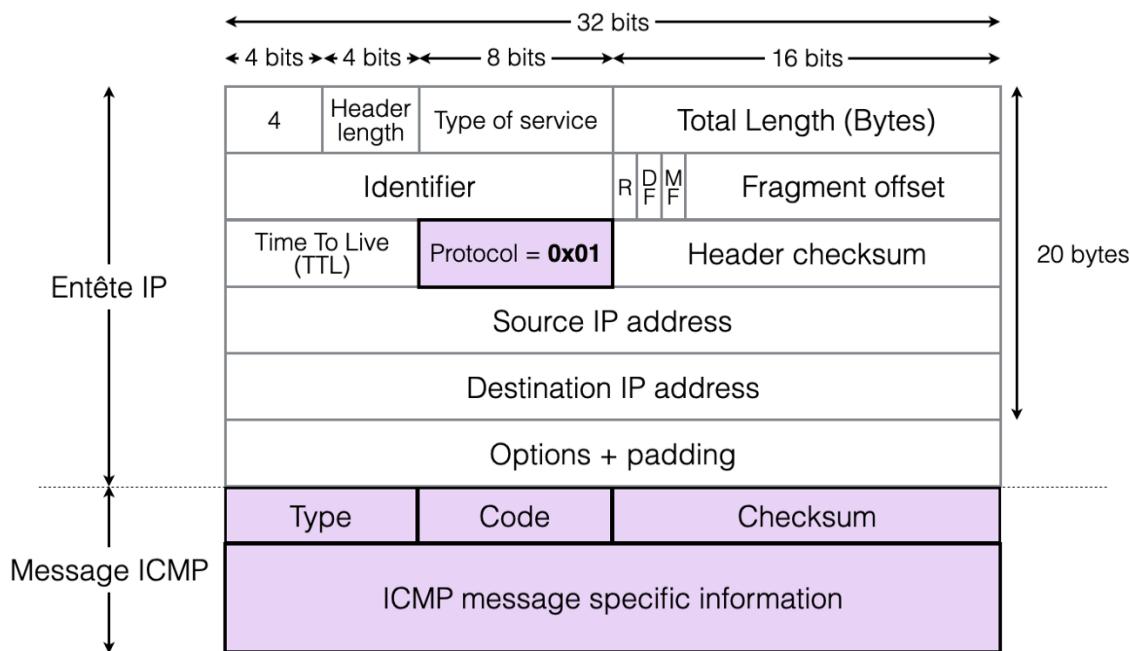
Le protocole ICMP (Internet control message protocol) permet aux machines IP :

- de rendre compte des erreurs qui surviennent lors de l'acheminement ou la livraison des paquets IP,
- de tester la connectivité de machines distantes.

Pour ce faire, ICMP introduit plusieurs types de messages, certains étant des messages d'erreur, d'autres étant utilisés pour les tests de connectivité.

ICMP est implémenté au même niveau que les protocoles de la couche transport (4) tels que TCP et UDP. Les messages ICMP

sont encapsulés dans des paquets IP dont le champ Protocol a la valeur 0x01.



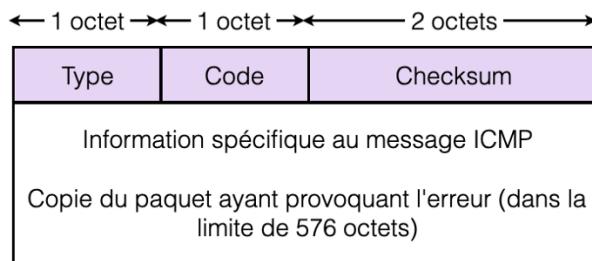
Encapsulation IP d'un message ICMP.

Entête ICMP

L'entête des messages ICMP contient :

- **Le champ Type (8)** qui indique la nature de l'erreur.
- **Le champ Code (8)** qui indique la raison qui a provoqué cette erreur.
- **Le champ Checksum (16)** porte sur l'ensemble du message ICMP et permet de détecter les bits en erreur.

Les quatre (4) octets suivants comprennent des champs qui dépendent du type de message.



- Type: nature du message ICMP

- messages d'erreur
- messages d'information

- Code: cause de l'erreur

- Checksum:

- Vérification de l'intégrité :
 - du message ICMP
 - du pseudo-entête IP (similaire à TCP et UDP)

Entête des messages ICMP.

Charge utile ICMP

La charge utile des messages ICMP d'erreur contient une copie partielle du paquet IP à l'origine de l'erreur ayant provoqué l'envoi du message ICMP. Cette copie contient l'entête IP complète du paquet en erreur et au minimum les huit (8) premiers octets des données de ce paquet. La copie du paquet en erreur ne peut excéder 576 octets. Cette copie est nécessaire, IP étant sans état. Une fois les paquets IP envoyés, leur source ne garde aucune trace concernant l'envoi de ces paquets. Pour rappeler à la source qu'elle est à l'origine du paquet ayant provoqué l'erreur pour laquelle elle reçoit un message d'erreur, ICMP copie un extrait du paquet en erreur.

Time exceeded (Type 0x0B)

Un message ICMP Time exceeded avec le code 0 est renvoyé lorsqu'un paquet IP dont le TTL a été atteint la valeur 0 est

supprimé. Un messages ICMP Time exceeded avec le code 1 est retourné lorsque le réassemblage d'un paquet IP échoue faute d'avoir reçu tous les fragments qui le compose.

Les messages ICMP Time exceeded avec le code 0 sont utilisés par la commande traceroute. La commande traceroute permet de découvrir l'adresse IP des routeurs situés le long du chemin permettant de joindre une destination dont l'adresse ou le nom est passé en argument de la commande traceroute. Pour ce faire, la machine source envoie une série de messages ICMP Echo Request (Type 0x07) encapsulés dans des paquets IP dont la valeur du TTL est incrémentée progressivement. Le premier Echo Request est encapsulé dans un paquet avec un TTL de 1. Le premier routeur rencontré supprime ce paquet et retourne à la source un message ICMP Time exceeded. Le message ICMP Time exceeded permet à la source de découvrir l'adresse IP du premier routeur. La valeur du TTL des paquets IP est incrémentée jusqu'à ce qu'un des paquets atteigne enfin la destination finale. Cette dernière renvoie le message ICMP Echo reply que la source attendait.

Destination unreachable (Type 0x03)

Un message ICMP Destination unreachable est retournée lorsqu'un paquet n'a pas pu atteindre sa destination finale. Plusieurs raisons peuvent être à l'origine d'un ICMP Destination unreachable.

Le code 0x04 indique qu'un paquet IP avec le bit DF positionné

à 1 a été supprimé aux abords d'un réseau dont la MTU était plus petite que la taille du paquet. C'est pour cette raison que l'entête des messages ICMP Destination unreachable contiennent le champ Next-Hop MTU. La valeur du champ Next-Hop MTU est exploitée par path MTU discovery, le mécanisme qui permet de découvrir la path MTU. La path MTU est la MTU la plus restrictive d'un des liens situé le long du chemin qui permet de joindre une destination en particulier.

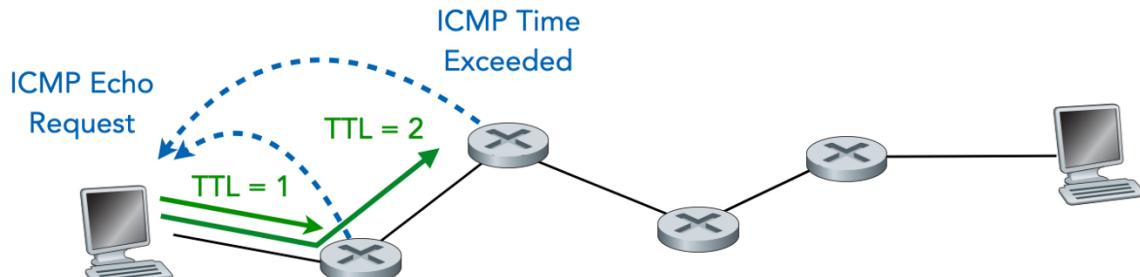
Le code 0x00 (Network Unreachable) est utilisé par un routeur pour indiquer qu'il n'a pas pu acheminer un paquet faute d'avoir pu trouver une route pour la destination du paquet dans sa table d'acheminement. Le code 0x01 (Host Unreachable) est utilisé par un routeur connecté au même réseau que la destination d'un paquet qu'il a dû supprimer faute d'avoir pu déterminer l'adresse MAC.

Commandes ping et traceroute

Hormis les messages d'erreur, ICMP introduit deux types de message pour tester la connectivité d'une machine distante. Il s'agit des messages Echo request (Type 0x07) et Echo reply (Type 0x01).

Les commandes ping et traceroute exploitent ces deux messages pour vérifier la réceptivité d'une machine et dans le cas contraire, de déterminer si un des routeurs situés le long du chemin qui permet de joindre cette machine est fautif. La commande ping fournit des statistiques concernant les pertes

enregistrées et le délai aller-retour du chemin entre la source à l'origine de la commande ping et la destination dont l'adresse IP ou le nom a été passé en argument de la commande ping.

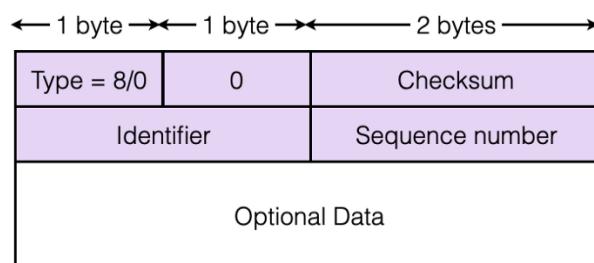


Commande Traceroute.

La commande ping accepte des options qui permettent de définir la taille des messages ICMP Echo request envoyés (option -s), de limiter le nombre des Echo request envoyés (option -c) ou de définir la valeur du TTL des paquets IP encapsulant les messages Echo request (option -m).

Echo request (Type 0x07) et Echo reply (Type 0x01)

L'en-tête des messages ICMP Echo request et reply contient deux champs supplémentaires : le champ Identifier (16) et Sequence number (16).



Message ICMP Echo request et Echo reply.

La valeur du **champ Identifier** identifie la machine vers laquelle les Echo request ont été envoyés, une machine pouvant lancer plusieurs ping simultanés à destination de plusieurs machines différentes.

La valeur du champ Identifier est utilisée par NAT pour identifier les messages ICMP émis par une machine située sur un réseau privé. Cette valeur permet à NAT de faire correspondre les messages ICMP reçus en réponse.

La valeur du **Sequence number** permet de mettre en relation un message ICMP Echo reply avec le message ICMP Echo request qui a provoqué son envoi, dans le cas où la commande ping est configurée pour envoyer plusieurs Echo reply. Les informations contenues dans la charge utile des messages Echo Request dépendent de l'implémentation de la commande ping. Ces informations permettent de générer les statistiques attendues concernant le taux d'erreur et le délai aller-retour du chemin emprunté par les messages ICMP Echo Reply et Request.

Adresses IP

Une adresse IP est un localisateur (locator) qui indique la position d'une machine dans Internet. Les adresses IP sont également utilisées par TCP comme des identifiants (identifiers) qui combinés aux numéros de ports, permettent d'identifier une connexion TCP.

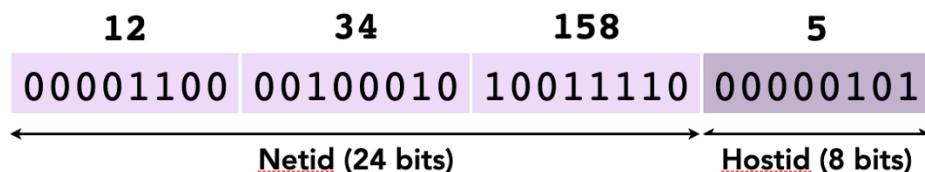
Notation décimale pointée

Dans le cas des adresses IPv4, il s'agit d'une valeur binaire longue de 32 bits présentée au **format décimal pointé** : chacun des quatre (4) octets qui la forment sont convertis en décimal et chaque valeur décimale résultant de cette conversion séparée par un point.

- Notation binaire : 0000 0110 0010 0010 1001 1110 0000
0101
- Notation décimale pointée : 12.34.158.5

Identifiants réseau et hôte

A la manière des numéros de téléphone, les adresses IP ont une structure hiérarchique. Les premiers bits d'une adresse IP appelés **identifiant réseau (netid)** ou **préfixe réseau** identifie le réseau auquel est connecté la machine à qui appartient cette adresse. Les bits restant forment l'**identifiant hôte (hostid)**. Sur un même réseau, toutes les machines ont un hostid unique.



Structure hiérarchique des adresses IP.

On distingue deux valeurs de hostid particulières :

- la valeur de hostid tout à 0 : les adresses IP dont les bits de l'identifiant hôte sont tous positionnés à 0 correspondent aux **adresses réseaux**.
- la valeur de hostid tout à 1 : les adresses IP dont l'identifiant hôte est positionné tout à 1 correspondent aux adresses de **diffusion locale**.

Ces deux valeurs ne peuvent pas être utilisées pour identifier une machine.

Attribution des adresses IP

Les adresses IP sont attribués hiérarchiquement par les fournisseurs d'accès Internet. On commence par numérotier les réseaux en leur attribuant un préfixe unique. Une fois le préfixe attribué à un réseau, on numérote les machines connectées à ce réseau en leur attribuant un identifiant hôte unique. C'est le rôle du DHCP.

La structure hiérarchique des adresses IP présente les trois (3) avantages suivants :

- **Restreindre la taille** des tables d'acheminement : une entrée suffit pour savoir comment joindre les machines dont les adresses IP partagent le même préfixe et ce, quelque soit leur nombre.
- **Limiter les retombées** des changements de topologie internes : l'ajout ou le retrait d'une machine est masqué par

l'adresse réseau contenu dans la colonne Destination des tables d'acheminement.

- **Masquer la topologie** interne des réseaux : les machines appartenant à un même réseau sont masquées par l'adresse réseau contenu dans la colonne Destination des tables d'acheminement.

Masque de réseau

Pour connaître la longueur des préfixes, les adresses IP sont fournies avec une adresse particulière appelée **masque** où sont positionnés à 1 les bits correspondant au préfixe d'une adresse IP. Le masque 255.255.248.0 indique un préfixe long de 21 bits. Associé à l'adresse 12.34.158.5, le masque indique que l'adresse réseau est égale à 12.32.152.0. L'adresse de diffusion locale sur ce réseau est égale à 12.32.159.255.

	12	34	158	5
Adresse	00001100	00100010	10011	110 00000101
	255	255	248	0
Masque	11111111	11111111	11111	000 00000000
	12	34	152	0
Adresse réseau	00001100	00100010	10011	000 00000000

notation habituelle : **12.34.152.0/21**

Masque et adresse réseau.

Adressage avec classes (classful)

Historiquement, le préfixe des adresses IP était déterminé par

la valeur de leurs premiers bits :

- Les adresses IP dont le premier bit est à 0 ont un préfixe long de 8 bits. Ils appartiennent à la classe A. Un réseau de classe A peut connecter jusqu'à $2^{24} - 2$ machines soit un total de 16.777.214 machines.
- Les adresses IP dont les deux premiers bit sont 0b10 est long de 16 bits. Ils appartiennent à la classe B. Un réseau de classe B peut connecter jusqu'à $2^{16} - 2$ machines soit un total de 65 534 machines
- Les adresses IP dont le trois premiers bits sont 0b110 ont un préfixe long de 24 bits. Ils appartiennent à la classe C. Un réseau de classe C peut connecter au maximum 254 machines.

Les adresses IP étaient donc réparties dans trois (3) classes d'adresse. L'attribution des adresses se faisait en numérotant les réseaux avec une adresse appartenant à une des classes A, B ou C. Une telle attribution est notoirement inefficace.

Un réseau avec 300 machines nécessite une adresse de classe B, les adresses de classe C ne permettant d'accueillir au maximum que 254 ($2^8 - 2$) machines. En attribuant une adresse réseau de classe B, seules 300 des adresses IP contenues dans cette adresse réseau sont utilisées. Les $2^{16} - 2 - 300$ adresses restantes sont gaspillées ce qui donne un taux d'utilisation de 0,46%.

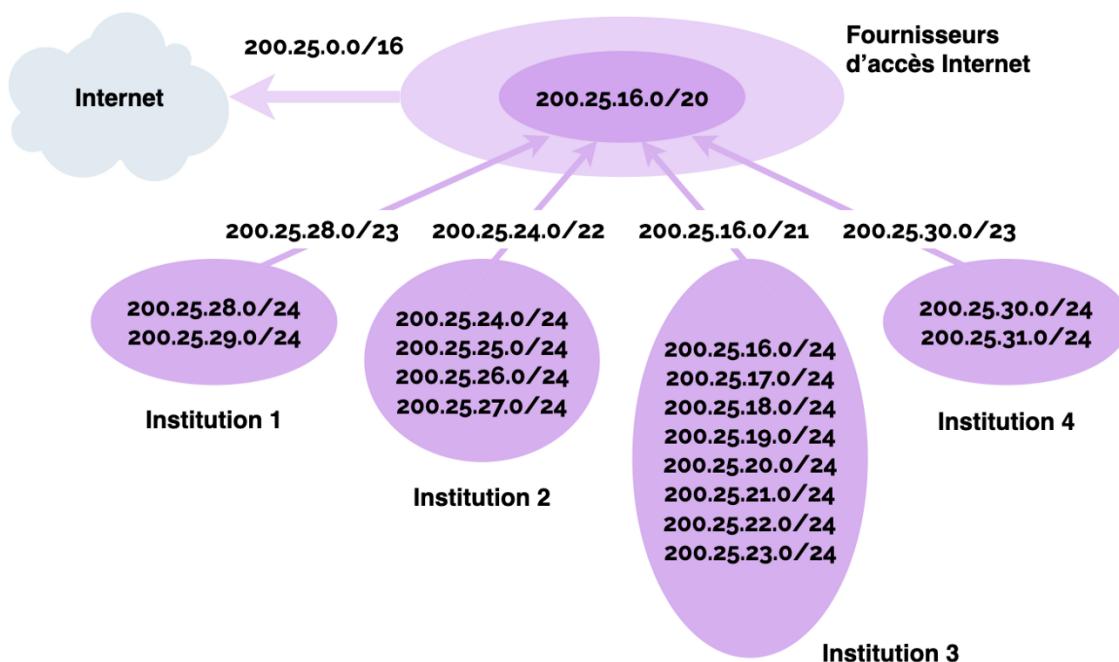
Avec classes				
Adresse	12	34	158	5
Classe A	00001100	00100010	10011110	00000101
netid: 8 bits hostid : 24 bits				
Adresse réseau	12	0	0	0
	00001100	00000000	00000000	00000000
Sans classe				
Adresse	12	34	158	5
Masque	00001100	00100010	10011	110 00000101
	255	255	248	0
Masque	11111111	11111111	11111	000 00000000
netid (préfixe) : 21 bits hostid : 11 bits				
Adresse réseau	12	34	152	0
	00001100	00100010	10011	000 00000000

Adressage classless vs. classful.

Adressage sans classe (classless)

En raison de leur inefficacité, la notion de classe a été remplacée par le **CIDR (Classless inter-domain routing)**. C'est le CIDR qui a normalisé l'utilisation des masques. Les masques permettent de créer des préfixes dont la longueur est ajustée en fonction des réseaux selon le nombre des machines qu'ils connectent.

Selon le CIDR, un réseau hébergeant 2^{n-2} machines se voit attribuer un préfixe long de 2^{32-n} bits. Si un réseau qui héberge 2046 machines au maximum se voit attribuer un préfixe long de 21 bits. Le masque de ce réseau est donc 255.255.248.0.

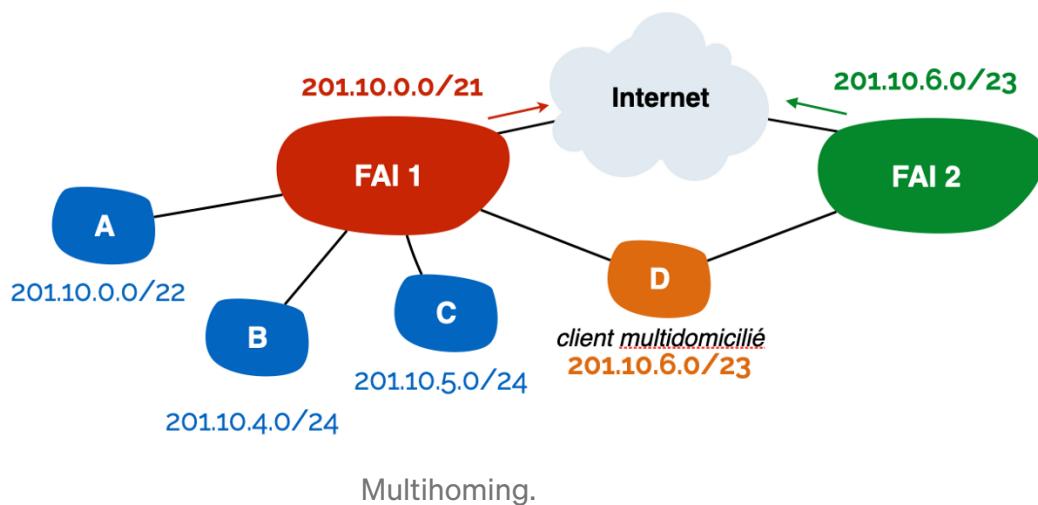


Allocation des adresses IP et CIDR.

En attribuant des préfixes dont la longueur est en adéquation avec la taille des réseaux, le CIDR assure une meilleure utilisation des adresses IP. Le CIDR permet également de réduire le nombre d'entrées dans les tables d'acheminement.

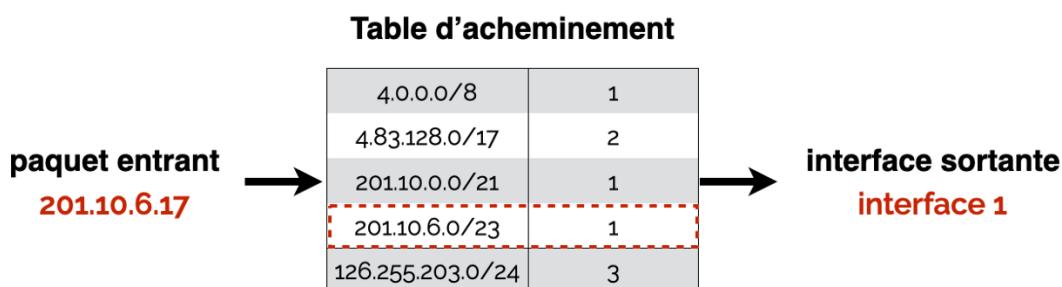
Longest prefix match (LPM)

Un des effets de bord du CIDR découle de la présence dans les tables d'acheminement, de plusieurs chemins possibles pour une destination donnée. Une table d'acheminement peut contenir plusieurs entrées qui toutes, permettent de joindre une même adresse IP destination. Ces entrées contiennent des préfixes de longueur différente imbriqués les uns dans les autres. Le préfixe 201.10.6.0/23 est contenu dans le préfixe 201.10.6.0/21. Ces deux préfixes sont cependant traités comme des destinations distinctes, leur longueur étant différente.



Multihoming. La présence de ces préfixes imbriqués est due notamment au multihoming (multidomiciliation). Le multihoming est une pratique qui consiste pour un réseau à utiliser les services de deux fournisseurs d'accès Internet pour accéder à Internet.

Supposons qu'un FAI possède le bloc d'adresses 201.10.0.0/21 qu'il annonce au reste d'Internet pour le compte de ses clients. Le FAI attribue à un de ses clients l'adresse 201.10.6.0/23. Ce même client peut contracter les services d'un second FAI à qui il demande d'annoncer le préfixe 201.10.6.0/23. Les routeurs se retrouvent alors avec deux chemins distincts qui permettent tous les deux de joindre le réseau du client.



Longest prefix match.

Le problème du multihoming réside dans le fait que les deux FAI annoncent au reste de l'Internet deux préfixes différents mais imbriqués. Ces annonces provoquent la création de deux entrées distinctes dans la table d'acheminement des routeurs. Les routeurs se retrouvent devant un dilemme, à savoir quelle entrée choisir pour acheminer les paquets destinés à cette organisation. Rappelons qu'Internet a été conçu de façon à acheminer les paquets sur un chemin unique dont la stabilité est cruciale notamment pour garantir l'efficacité des services offerts par TCP.

Pour régler le dilemme découlant de la présence de chemins multiples, le choix a été fait de sélectionner systématiquement le chemin avec le préfixe le plus long. On parle de **Longest prefix match (LPM)**. Pour ce faire, les routeurs cherchent l'entrée dont l'adresse destination est bien celle avec le préfixe le plus long. Le LPM ralentit fortement l'acheminement des paquets IP, le temps nécessaire à la recherche du préfixe le plus long partagé étant proportionnel à la taille des tables d'acheminement.

Pour éviter aux routeurs d'avoir à consulter l'ensemble des entrées de leur table, des algorithmes de recherche empruntés au domaine de la recherche d'information (Information retrieval) ont été mis en place. Ces algorithmes organisent les tables selon des structures arborescentes appelées Tries.

En parallèle, a émergé **MPLS (Multi-path label switching)** qui est une technique d'acheminement apparue à la même période que le CIDR. Cette technique consiste à bypasser la couche Réseau. MPLS est implanté dans une couche 2bis intercalée entre la couche Liaison de données (2) et la couche Réseau (3). MPLS ajoute un entête situé entre les entêtes Ethernet et IP, contenant une valeur appelée **label**. La portée d'un label est locale à chacun des liens qui forment un chemin complet. Un chemin est donc identifié par une succession de labels. Sur chaque lien, la valeur locale du label identifie l'interface de sortie sur laquelle acheminer les paquets dont l'adresse destination appartient à un ensemble de préfixes appelé **FEC (Forwarding equivalence class)**.

Subnetting vs. Supernetting

Pour améliorer l'efficacité d'utilisation des adresses de classe B déjà attribuées, on utilise la technique du **subnetting**. Cette technique consiste à étendre la longueur du masque primaire des adresses de classe B. Pour ce faire, on découpe une adresse de classe B en plusieurs adresses de sous-réseaux en introduisant à la suite du netid, un identifiant appelé **subnetid**. Les bits du subnetid sont empruntés au hostid. La longueur du subnetid dépend du nombre réseaux physiques situés au sein du site à qui appartient l'adresse de classe B.

Subnetting

- concerne principalement les adresses de classe B
- configuration d'un masque de longueur supérieur à 16

Supernetting

- concerne principalement les adresses de classe C
- configuration d'un masque de longueur supérieur à 24

supérieur à 16				supérieur à 24			
<ul style="list-style-type: none"> découpage d'une classe B en plusieurs adresses de sous-réseaux 				<ul style="list-style-type: none"> agrégation de plusieurs classes C en une adresse de réseau 			
132.227.0.0/16 → 132.227.(0-224).0/19				192.168.(0-7).0/24 → 192.168.0.0/21			
132	227	0000	0000	192	168	0000	0000
132	227	0010	0000	192	168	0000	0001
		0100	0000			0000	0010
		0110	0000			0000	0011
...		1000	0000	...		0000	0100
		1010	0000			0000	0101
		1100	0000			0000	0110
132	227	1110	0000	192	168	0000	0111
							0

Subnetting vs Supernetting.

Un site utilise l'adresse 132.227.0.0 qui est une adresse de classe B. Son masque primaire est long de 16 bits. Supposons que ce site est composé au maximum de 8 sous-réseaux. Il est donc nécessaire d'emprunter au hostid 3 bits utilisés pour le subnetid. Les adresses des sous-réseaux sont les suivantes :

- 132.227.0.0/19
- 132.227.32.0/19
- 132.227.64.0/19
- ...
- 132.227.224.0/19

/19 fait référence à la longueur du préfixe des sous-réseaux. C'est une notation alternative au masque 255.255.224.0.

De l'extérieur, le site est vu au travers de l'adresse 132.227.0.0 dont le préfixe est long de 16 bits. A l'intérieur du site, chaque sous-réseau est configuré avec une adresse réseau dont le

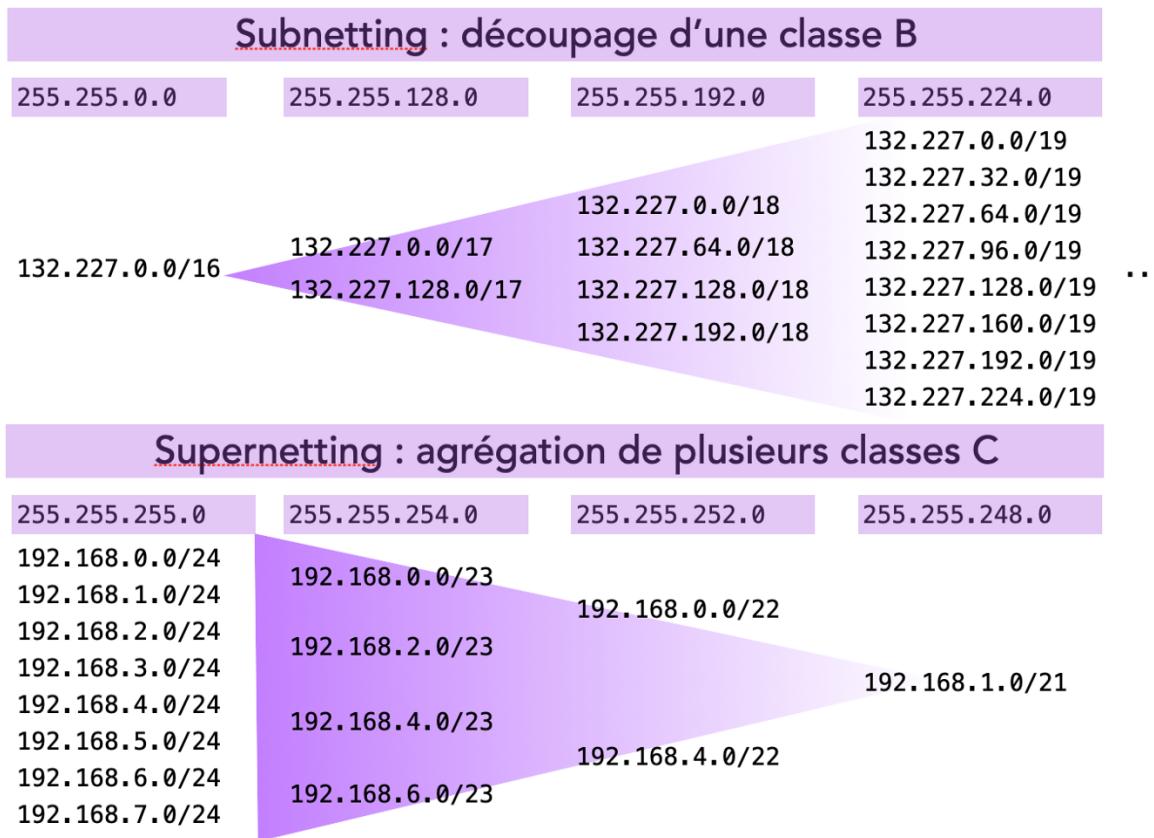
préfixe est long de 19 bits.

Le **supernetting** est une technique qui concerne principalement les adresses de classe C. L'objectif est de réduire la longueur du masque primaire des adresses de classe C. Pour ce faire, on agrège plusieurs adresses de classe C contigües. Agréger deux adresses de classe C contigües revient à créer un adresse réseau dont le préfixe est long de 23 bits, agréger quatre adresses de classe C revient à créer une adresse réseau dont le préfixe est long de 22 bits et ainsi de suite.

Considérons à nouveau le site précédent composé de 8 réseaux physiques. Si ces réseaux hébergent au maximum 254 machines, on peut leur attribuer les adresses de classe C suivantes :

- 192.168.0.0 (0000 0000)
- 192.168.1.0 (0000 0001)
- 192.168.2.0 (0000 0010)
- 192.169.3.0 (0000 0011)
- 192.169.4.0 (0000 0100)
- 192.169.5.0 (0000 0101)
- 192.169.6.0 (0000 0110)
- 192.169.7.0 (0000 0111)

Du fait d'être contigües, ces adresses de classe C couvrent l'ensemble des adresses IP contenues dans la plage d'adresses définie par le préfixe 192.168.0.0/21.



Subnetting vs. Supernetting.

Interactions entre couches Réseau (3) et Liaison de Données (2)

La fonction principale de la couche Réseau (3) est d'acheminer les paquets en sélectionnant le saut suivant vers lequel diriger les paquets. Le saut suivant est un nœud voisin connecté au même réseau physique. La sélection de saut suivant est réalisée

manuellement ou par un protocole de routage en fonction de la destination finale.

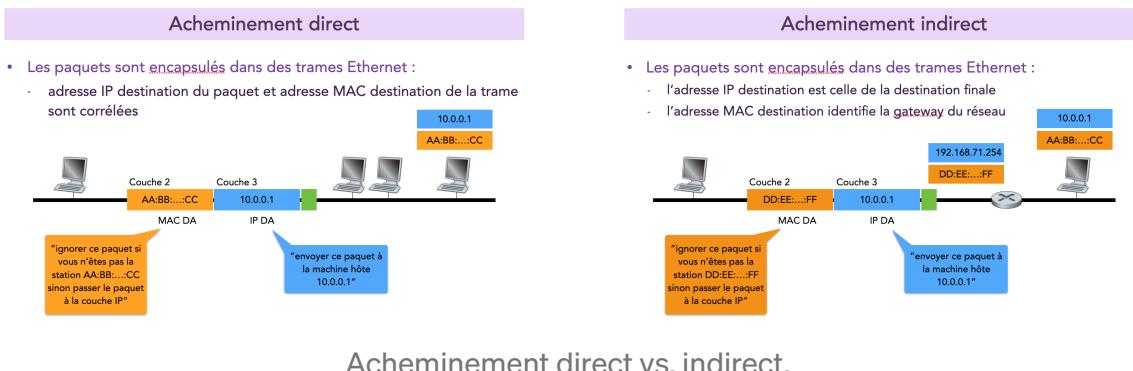
Afin d'envoyer le paquet vers ce voisin, machines hôtes et routeurs utilisent les fonctions de la couche Liaison de données (2) dont le rôle est d'encapsuler un paquet dans une trame qui résulte de l'ajout d'un entête et d'un enqueue.

Acheminement direct/indirect

La trame ainsi formée est envoyée vers une machine connectée au même réseau physique. Cette machine peut être la destination finale du paquet. On parle d'acheminement direct. Si le nœud suivant est un routeur, on parle alors d'acheminement indirect. Le routeur va à son tour, une fois le saut suivant déterminé par sa couche Réseau (3), faire appel à sa couche Liaison de données (2) pour envoyer le paquet encapsulé dans une nouvelle trame vers la machine voisine suivante située sur le chemin déterminé par le protocole de routage pour joindre la destination finale du paquet. C'est ainsi que de proche en proche (hop-by-hop), le paquet atteindra sa destination finale.

Pour identifier leur destination – et leur source – paquets et trames contiennent deux adresses. Dans le cas des paquets, il s'agit d'une adresse IP dont la portée est globale. L'adresse IP source est connue grâce au service offert par DHCP. L'adresse IP destination est connue grâce au service de résolution offert par DNS (Domain name system) qui permet de déterminer

l'adresse IP d'un serveur, à condition d'en connaître le nom.



Acheminement direct vs. indirect.

Adresses MAC

Dans le cas des trames Ethernet, il s'agit des adresses MAC dont la portée est locale. L'adresse MAC d'une machine n'est connue que des machines voisines connectées au même réseau physique. Le protocole ARP permet de déterminer l'adresse MAC d'une machine voisine à condition de connaître au préalable l'adresse IP de cette machine. Contrairement aux adresses IP, les adresses MAC sont **plates**. Elles sont longues de 48 bits et présentées selon un format hexadécimal : les octets sont convertis en hexadécimal et les deux symboles résultant de cette conversion sont séparés par un double point (":") ou un tiret (-). Les adresses MAC sont vissées en dur sur les cartes réseau par leur constructeur avant leur commercialisation. Leur attribution ne peut donc dépendre ni de la topologie du réseau, ni de la position des machines.

La destination d'un paquet correspond à l'adresse IP du récepteur final tandis que celle d'une trame est l'adresse MAC

du saut suivant du paquet encapsulé si en transit. L'adresse MAC destination d'une trame et l'adresse IP destination du paquet qu'elle encapsule identifient la même machine si le paquet a atteint le réseau physique du récepteur final. Les machines source et destination d'un trame appartiennent à des réseaux physiques appelés réseaux locaux.

Hormis l'encapsulation des paquets, la couche Liaison de données (2) prend en charge les méthodes d'accès nécessaires notamment aux réseaux locaux dits à diffusion naturelle. CSMA/CD est la méthode d'accès qui régit les accès aux réseaux locaux Ethernet.

Trame Ethernet

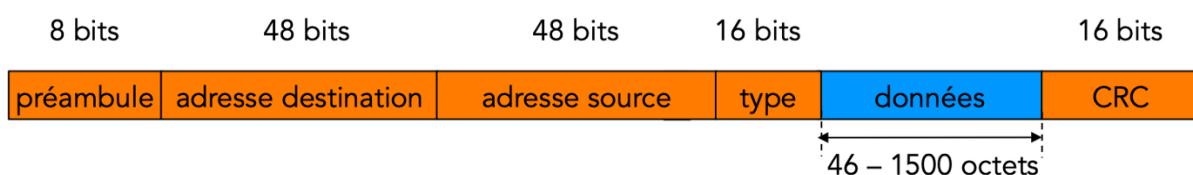
Le format des trames Ethernet définit la structure de leur entête et la présence d'un enqueue. Entête et enqueue encadrent le champ Données dont la taille est limitée par une valeur min de 46 octets et une valeur max de 1500 octets.

L'entête des trames Ethernet comprend 4 champs :

- **Préambule (7)** et **SFD (1)** : la valeur du préambule vaut 0xAAAAAAAAAAAAAA et celle du SFD (Start frame delimiter) 0xAB. Ces deux champs ne font pas à proprement parler de l'entête Ethernet. Ils permettent aux récepteurs de se caler sur le début de la trame. Du fait de contenir une alternance de bits 0 et de bits 1, le préambule permet également de régler l'horloge d'un récepteur à la cadence à

laquelle les bits de la trame ont été transmis par sa source.

- **Adresse MAC destination (6)** : il s'agit de l'adresse d'une machine voisine située sur le même réseau physique que la source. Ce qui explique la position de l'adresse MAC destination, c'est le fait que Ethernet a été conçu pour les réseaux à diffusion naturelle : une trame Ethernet est reçue de toutes les machines connectées au même réseau physique que la source. A la réception d'une trame, il est donc nécessaire de vérifier si une trame nous est destinée avant de se lancer dans l'analyse de la trame.
- **Adresse MAC source (6)** : il s'agit de l'adresse MAC de la machine à l'origine de la trame.
- **Type (2)** : comme l'indique son nom, le champ Type indique le type des données encapsulées dans une trame. La valeur 0x0800 indique qu'il s'agit d'un paquet IPv4, la valeur 0x0806 indique qu'il s'agit d'un message ARP.



Entête des trames Ethernet.

L'enqueue des trames Ethernet contient le **champ FCS** (Frame check sequence) qui permet la détection et le rejet des trames en erreur. La valeur du FCS est calculée par un mécanisme de

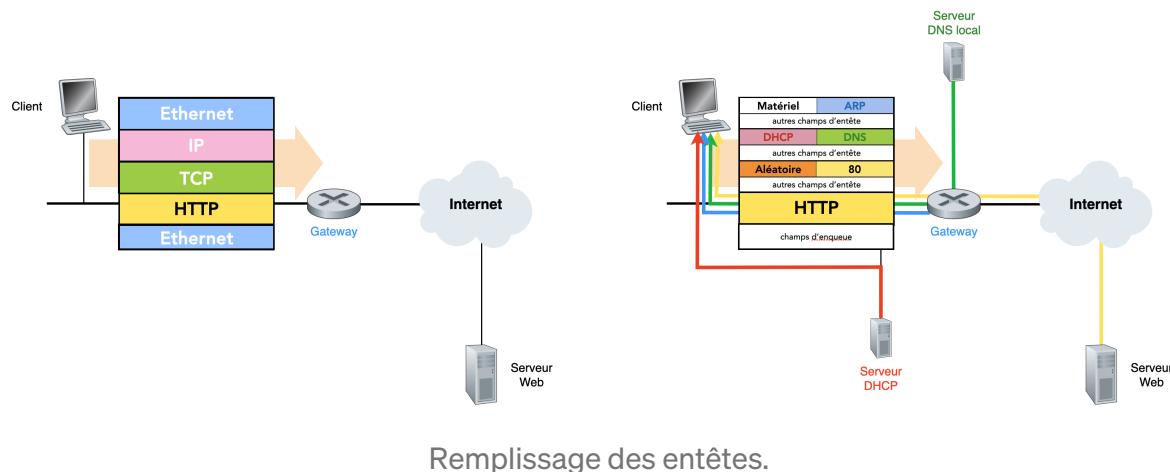
division polynomiale qui utilise un polynôme générateur de degré 32. Une fois rejetée, la réparation des données contenue dans une trame est laissée à la charge des couches supérieures grâce à des protocoles tel que TCP.

Bourrage. Si un paquet IP a une taille inférieure à 46 octets, Ethernet utilise des bits de bourrage pour remplir le champ Données de la trame qui encapsule ce paquet. A la réception des trames contenant des bits de bourrage, c'est la couche IP qui aura la charge de retirer ces bits de bourrage, l'entête Ethernet n'ayant pas de champ Longueur. L'absence de champ Longueur dans l'entête Ethernet autorise l'encapsulation des seuls protocoles qui incluent un champ Longueur dans l'entête de leurs messages.

Configuration des machines hôtes

Les machines hôtes accèdent à Internet en étant connectées à un réseau local. Un réseau local connecte les machines appartenant à une même organisation telle qu'une PME ou une université. De ce fait, les réseaux locaux ont une portée géographique limitée et peuvent connecter quelques centaines de machines. Les réseaux locaux offrent des services liés notamment à la présence d'une passerelle (gateway) et de serveurs DHCP et DNS. Ces services sont nécessaires pour permettre aux machines hôtes connectées au réseau local d'accéder à Internet.

Pour utiliser ces services, une machine hôte doit connaître certaines valeurs clefs qui lui permettront d'envoyer des paquets IP encapsulés dans des trames Ethernet. Ces valeurs sont cruciales dans le sens où elles permettront aux machines de compléter les champs des entête de couche 2, 3 et 4, les paquets IP pouvant contenir des segments TCP ou des datagrammes UDP.



Remplissage des entêtes.

Remplissage des entêtes

La liste des champs d'entête comprend notamment :

- l'adresse MAC source de la machine hôte (vissée en dur sur la carte réseau)
- l'adresse MAC destination (destination finale ou nœud suivant)
- l'adresse IP source de la machine hôte
- l'adresse IP destination (déterminée par DNS)
- le numéro de port source (aléatoire ou prédéterminé pour les services système)
- le numéro de port destination (prédéterminé en fonction de l'application)

L'adresse MAC source est celle de l'interface réseau de la machine hôte source. Elle est vissée en dur par son

constructeur avant d'être commercialisé. La machine hôte source connaît donc son adresse MAC.

Concernant les numéros de port, le numéro de port côté client est tiré aléatoirement ou prédéterminé s'il s'agit d'un service système. Côté serveur, le numéro de port est déterminé connaissant l'application utilisée par la machine hôte : si c'est un navigateur qui est à l'origine du paquet alors le port destination aura la valeur 80.

Les valeurs manquantes sont donc l'adresse IP source et les adresses MAC et IP destination. Pour déterminer la valeur de ces adresses, une machine hôte doit connaître la valeur des paramètres réseau suivants :

- Serveurs DNS locaux
- Masque du réseau local
- Adresse IP de la gateway du réseau local

Paramètres réseau

Serveur DNS local. Pour déterminer l'adresse IP destination d'un paquet IP, une machine hôte sollicite les services rendus par DNS. Pour cela, la machine hôte doit connaître une des adresses IP du serveur DNS local. Le serveur DNS local est un serveur proxy qui joue le rôle de point d'entrée au service offert par les serveurs DNS racine, TLD et d'autorité.

Masque du réseau local. Pour déterminer l'adresse MAC destination de la trame qui encapsule le paquet IP, la machine hôte doit déterminer si la destination finale du paquet IP est une machine voisine ou distante. Pour ce faire, la machine hôte utilise le masque de son réseau local. Le masque permet de comparer l'adresse du réseau local au préfixe réseau supposé de la destination. La destination finale du paquet IP est une machine voisine à condition que ces deux valeurs coïncident entre elles.

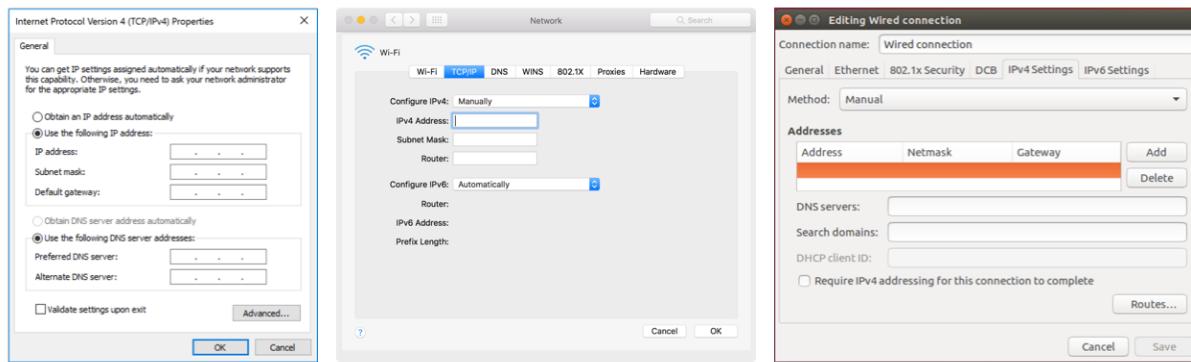
Si la destination finale est une machine voisine, la machine hôte consulte sa table ARP. Si l'adresse MAC recherchée n'est pas trouvée, elle envoie une requête ARP. Si la destination finale est distante, l'adresse MAC destination est celle du nœud suivant, i.e., la gateway du réseau local.

Adresse IP de la gateway. Si la destination du paquet IP que souhaite envoyer la machine hôte est distante, l'adresse MAC destination de la trame encapsulant ce paquet IP est l'adresse MAC de la gateway du réseau local. Pour déterminer l'adresse MAC de la gateway, la source doit connaître l'adresse IP de sa gateway locale. Elle pourra alors utiliser le protocole ARP.

Configuration statique vs. dynamique

Configuration statique. Qu'il s'agisse de l'adresse IP de la machine source, le masque de son réseau local, les adresses IP du serveur DNS local et l'adresse IP de sa gateway, ces valeurs peuvent être configurées manuellement par l'administrateur du

réseau local ou l'utilisateur de la machine hôte. La configuration manuelle des paramètres réseau présente l'inconvénient d'être statique : en cas de changement, les paramètres doivent être modifiés manuellement.



Configuration statique.

Configuration dynamique. Les machines hôtes étant de nos jours pour la plupart mobiles, elles se connectent à Internet en changeant de réseau local. A chaque changement de réseau, il est nécessaire de changer les valeurs des paramètres qui leur permettent d'accéder à Internet. Pour faciliter et accélérer ces changements de paramètres, la configuration automatique et à la demande est plus adaptée. Le protocole DHCP a été conçu dans le but d'autoriser la configuration dynamique des machines hôtes souhaitant accéder à Internet et ce, quelque soit le réseau local auquel elles se connectent.

Dynamic Host Configuration Protocol DHCP

L'objectif du protocole DHCP est de communiquer les paramètres nécessaires aux machines hôtes d'un réseau local pour accéder à Internet. DHCP repose sur un serveur répliqué,

installé sur le réseau local.

Paramètres réseau et durée de bail. Un serveur DHCP est chargé de communiquer les paramètres nécessaires aux machines hôtes connectées au réseau local pour accéder à Internet. Ces paramètres sont valables pour une durée préétablie par le serveur appelée durée de bail (**lease time**). Ces paramètres sont les suivants :

- l'adresse IP que la machine hôte pourra utiliser pendant la durée du bail tant qu'elle reste connectée sur le même réseau local que le serveur DHCP,
- le masque du réseau local,
- l'adresse IP de la gateway du réseau local,
- les adresses IP des serveurs DNS locaux (primaire et secondaires),
- la durée du bail.

A l'issue de la durée du bail, une machine hôte peut contacter à nouveau le serveur DHCP pour rafraîchir et ainsi vérifier la validité des valeurs précédemment communiquées. S'il n'est pas contacté avant la fin de la durée du bail, le serveur DHCP récupère autoritairement l'adresse IP précédemment attribuée. Du fait d'être implicite, la libération des adresses IP précédemment attribuées est robuste aux défaillances des clients.

Messages DHCP. Le protocole DHCP définit quatre (4) types de messages :

- DHCP Discovery (client → serveur)
- DHCP Offer (serveur → client)
- DHCP Request (client → serveur)
- DHCP ACK (serveur → client)

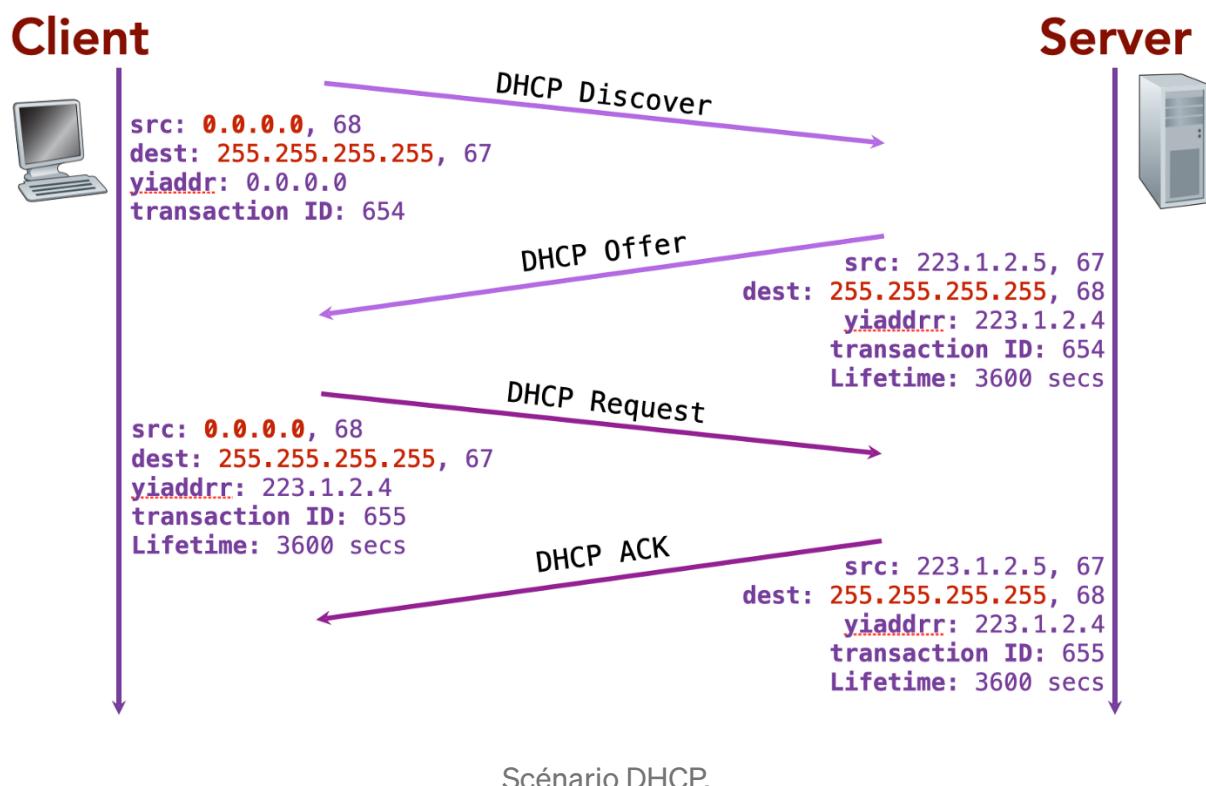
La machine hôte ne connaissant ni son adresse IP ni celle des serveurs DHCP locaux, les messages DHCP utilisent l'adresse IP non spécifiée et sont diffusés sur le réseau local :

- l'adresse IP source des messages envoyées par la source (Discovery et Request) est l'adresse IP non-spécifiée (tout-à-0) : 0.0.0.0 ;
- l'adresse IP destination des 4 messages est l'adresse IP de diffusion (tout-à-1) : 255.255.255.255 ;
- l'adresse MAC destination des 4 messages est l'adresse MAC de diffusion (tout-à-1) : FF:FF:FF:FF:FF:FF .

Un client qui tente de renouveler la durée du bail de son adresse IP peut contacter le serveur DHCP en utilisant l'adresse IP déjà connue de ce serveur.

Un autre conséquence découlant du fait que la machine hôte ne connaît pas son adresse IP, est l'utilisation de UDP. Au niveau

transport, Internet propose deux protocoles : TCP et UDP. TCP est un protocole à états appelés TCB (Transmission control block) dont l'initialisation requiert l'établissement d'une connexion. Pour identifier ces états, les machines utilisent des identifiants uniques comprenant notamment les adresses IP source et destination de la connexion TCP. Dans le cas de DHCP, les clients n'ont pas encore d'adresse IP. Il leur est donc impossible d'utiliser TCP. Le seul choix restant est UDP. Le numéro de port du client DHCP est 68 et côté serveur 67.



Opérations. Un client commence par envoyer un message DHCP Discovery qui contient l'adresse MAC du client et se met en attente des DHCP Offer que les serveurs DHCP primaire et secondaires enverront en réponse.

Les DHCP Offer contiennent une adresse IP libre, la valeur des paramètres réseau du réseau local (masque du réseau local, serveurs DNS locaux, adresse IP de la gateway locale) et une durée de bail. Les DHCP Offer contiennent également l'adresse MAC du client. Le client peut ainsi mettre en correspondance les DHCP Offer et son DHCP Discovery.

Le client choisit alors une des offres et diffuse un DHCP Request contenant l'adresse IP contenue dans le DHCP Offer sélectionné ainsi que l'adresse IP du serveur DHCP à l'origine de cette offre. Du fait d'être diffusé, l'ensemble des serveurs DHCP qui avaient fait une offre reçoit le DHCP Request. Les serveurs DHCP dont l'offre n'a pas été retenue rendent l'adresse IP qu'ils avaient offerte à nouveau disponible.

Le serveur DHCP dont l'offre a été retenue, envoie un DHCP ACK contenant à nouveau l'adresse IP, les paramètres réseau et la durée du bail déjà communiqués dans le DHCP Offer. A la réception du DHCP ACK, la phase de découverte des paramètres réseau est terminée. Le client peut dès lors configurer son module IP avec les valeurs communiquées par le serveur DHCP dans son DHCP ACK.

Choix de l'adresse IP offerte. Le client peut s'assurer que l'adresse IP offerte dans les messages DHCP Offer et ACK est disponible en envoyant une requête ARP. Si cette requête ARP reste sans réponse, le client aura confirmation que l'adresse IP offerte est bien disponible. Le choix de l'adresse IP offerte par

un serveur DHCP peut être aléatoire ou fixé en fonction de l'adresse MAC du client. Pour ce faire, l'administrateur réseau doit configurer au préalable les serveurs DHCP du réseau local avec l'adresse IP spécifique associée à un client étant donné son adresse MAC.

Renouvellement de la durée du bail. Un client peut envoyer un DHCP Renew pour étendre la durée du bail et continuer à utiliser les paramètres réseau précédemment négociés. Le client peut envoyer un premier DHCP Renew une fois la moitié de la durée du bail écoulée. Les DHCP Renew sont envoyés au serveur DHCP à l'origine de l'offre initiale qui répondra alors avec un DHCP Request. En l'absence de réponse, le client peut retransmettre le DHCP Renew avant d'abandonner après plusieurs tentatives en échec.

Protocole ARP

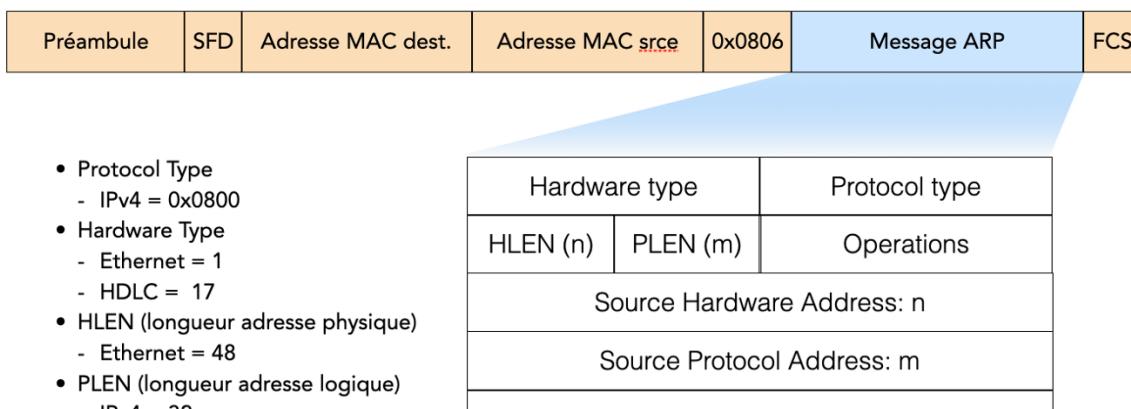
Les machines hôtes envoient leurs paquets IP encapsulés dans des trames Ethernet. L'en-tête d'une trame Ethernet contient l'adresse MAC source et destination de la trame. L'adresse MAC source est celle de la machine hôte à l'origine du paquet IP encapsulé dans la trame. L'adresse MAC destination est celle du nœud voisin qui peut être la destination finale du paquet IP ou celle de la gateway du réseau local. Dans les deux cas, la machine hôte doit déterminer quelle est l'adresse MAC du nœud voisin. Pour ce faire, les machines hôtes utilisent le protocole ARP.

Le protocole ARP permet de découvrir l'adresse MAC d'une machine voisine, i.e., située sur le réseau local, à condition de connaître son adresse IP.

Messages ARP. Le protocole ARP utilise deux types messages : les requêtes et les réponses. Qu'il s'agisse d'une requête ou d'une réponse, le format des messages est identique. La valeur du champ appelé Operation, long de 2 octets permet de déterminer le type du message ARP : la valeur 0x0001 indique qu'il s'agit d'une requête, la valeur 0x0002 indique qu'il s'agit d'une réponse.

Encapsulation Ethernet. Les messages ARP sont encapsulés dans des trames Ethernet. 0x0806 est la valeur du champ Type des trames Ethernet qui encapsulent un message ARP.

Opérations. Tout comme DHCP, le protocole ARP repose sur la diffusion de requêtes reçus de toutes les machines connectées au réseau local. L'adresse MAC destination des trames Ethernet qui encapsulent une requête ARP est l'adresse Ethernet de diffusion (tout-à-1) FF:FF:FF:FF:FF:FF.



- II v4 - 32
- Operations
 - Requête ARP = 1
 - Réponse ARP = 2

Destination Hardware Address: n
Destination Protocol Address: m

Entête des messages ARP.

Les messages ARP contiennent les quatre 4 adresses suivantes :

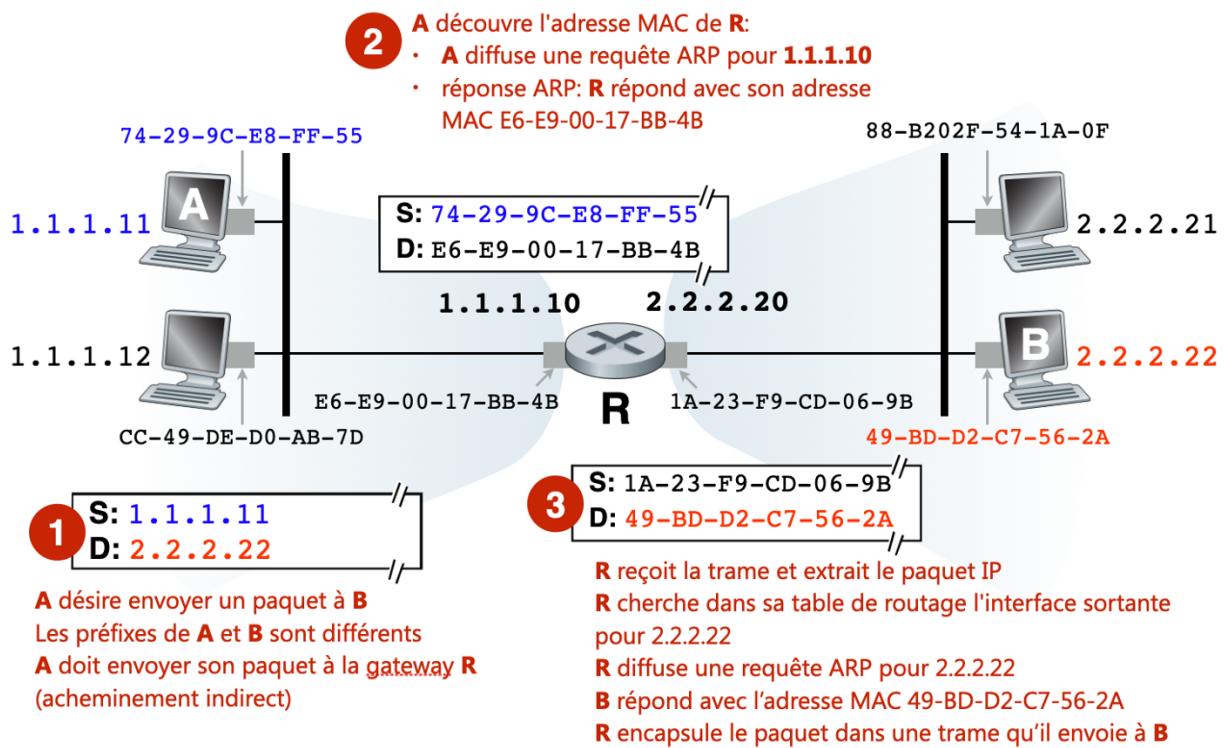
- l'adresse MAC de la source
- l'adresse IP de la source
- l'adresse MAC de la cible
- l'adresse IP de la cible

La cible dans le cas des requêtes ARP fait référence à la machine dont on cherche à déterminer l'adresse MAC. Dans le cas des réponses ARP, la cible fait référence à la machine qui cherche à déterminer l'adresse MAC d'une machine voisine.

La valeur de l'adresse MAC de la cible contenue dans une requête est l'adresse MAC non-spécifiée (toute-à-0) : 00:00:00:00:00:00.

Les requêtes contiennent les adresses IP et MAC de la machine source. Ces informations permettent à la cible d'une requête de répondre en point-à-point (unicast). En l'absence de réponse, une requête échoue et la machine hôte à l'origine de la demande de résolution correspondante abandonne sa demande. Une nouvelle demande pourra être émise à

l'initiative d'un des protocoles de couches supérieures tel que TCP.



Opérations du protocole ARP.

Table ARP. Une fois l'adresse IP découverte, la machine à l'origine de la requête ARP crée une entrée temporaire dans sa table ARP où sont maintenues temporairement les correspondances entre adresse MAC et adresse IP récemment découvertes. Les tables ARP évitent ainsi la diffusion systématique de requêtes ARP.

Les entrées sont maintenues temporairement pour :

- 1) maintenir la taille des tables ARP petite en gardant uniquement les entrées concernant les adresses MAC

récemment découvertes ;

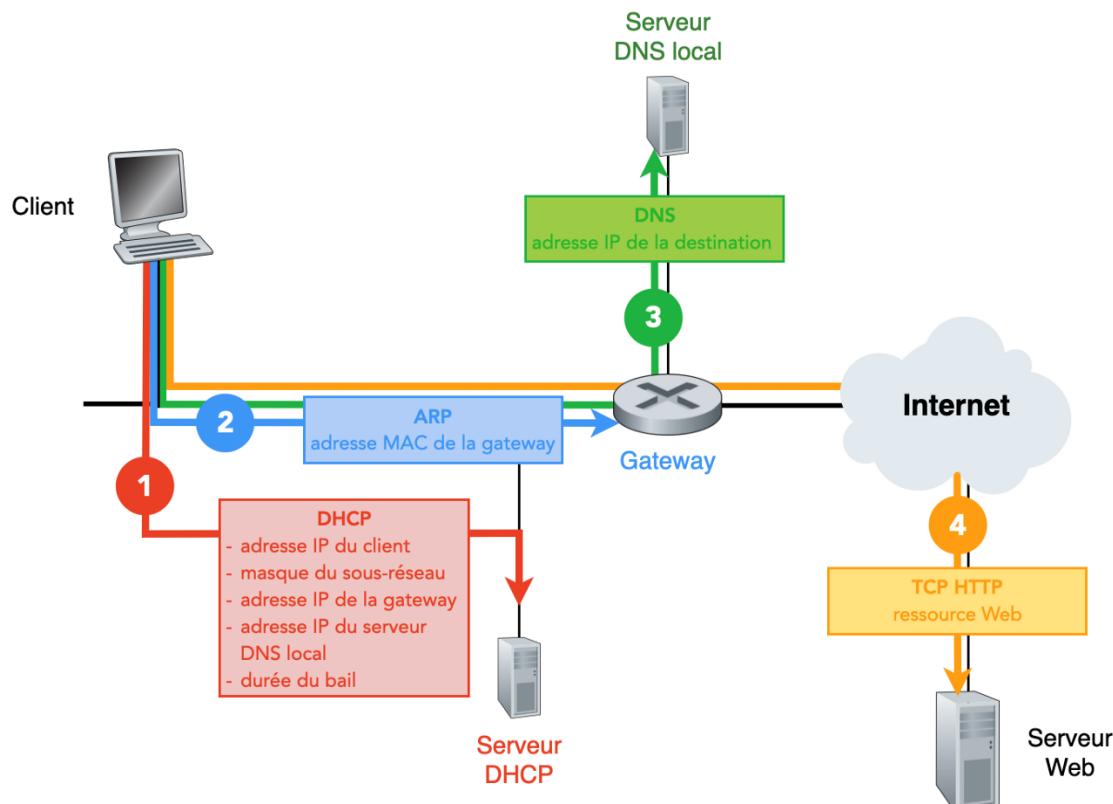
2) mettre à jour les correspondances en forçant l'envoi d'une nouvelle requête.

L'administrateur d'une machine peut également créer manuellement des entrées statiques qui n'expirent jamais. Ces entrées concernent généralement des machines fréquemment contactées. C'est le cas de la gateway local, par exemple.

Gratuitous ARP. ARP peut être détourné de son utilisation initiale. ARP peut être utilisé par une machine pour annoncer un changement concernant son adresse IP ou son adresse MAC. Pour ce faire, la machine envoie une requête prétendant résoudre sa propre adresse IP : les adresses MAC et IP source sont celles de la machine source à l'origine de l'annonce, les adresses MAC et IP de la cible sont respectivement l'adresse MAC tout-à-0 et l'adresse IP de la machine source. Un autre moyen consiste à diffuser une réponse non sollicitée où la machine à l'origine de l'annonce est à la fois la source et la cible : les adresses IP et MAC précisées dans la réponse diffusée sont celles de la machine à l'origine de cette réponse. Du fait d'être non sollicités, ces messages sont appelés messages Gratuitous ARP (GARP).

L'usage des messages Gratuitous ARP (GARP) est dangereux puisqu'ils peuvent être utilisés pour empoisonner les tables ARP des machines d'un réseau local. Grâce aux messages GARP, une

machine mal intentionnée peut aisément impersonnifier une autre machine sur le réseau et intercepter les messages qui lui sont destinés à l'insu de tous. Pour ce faire, il lui suffit d'envoyer des message GARP dans lesquels elle indique l'adresse IP de la machine victime comme étant la sienne, adresse IP qu'elle associe à son adresse MAC. Ces messages GARP créeront des entrées erronées dans les tables ARP de toutes les machines du réseau local.



Services d'un réseau local : DHCP, DNS et ARP.

En résumé, le remplissage des entêtes des paquets IP envoyés par une machine hôte est rendu possible grâce aux protocoles DHCP, DNS et ARP. Ces protocoles reposent sur la présence d'un serveur DHCP, d'un serveur DNS local et d'une gateway

sur le réseau local dont il s'agit de découvrir les identifiants tels que l'adresse IP et/ou l'adresse MAC. Pour ce faire, DHCP et ARP exploitent l'adresse MAC et/ou IP de diffusion (FF:FF:FF:FF:FF/255.255.255.255) pour joindre sans connaître préalablement les adresses MAC et IP du serveur DHCP et l'adresse MAC de la gateway.

Middlebox et NAT

Un des problèmes que connaît Internet découle de l'épuisement des adresses IPv4. Sur les 2^{32} adresses IPv4, il n'en reste que très peu encore disponibles. Il est donc nécessaire de trouver un moyen de ménager l'utilisation des adresses IPv4 dans l'attente qu'IPv6 soit largement accepté.

Une des solutions très largement utilisées est un mécanisme de translation d'adresses appelé **Network address translation ou NAT**.

Ce mécanisme consiste :

1. à attribuer des adresses IP privées aux machines d'un réseau local appelé de ce fait réseau privé,
2. à réécrire les adresses IP et les numéros de port contenus dans les entêtes IP et TCP des paquets IP sortants et entrants du réseau privé,

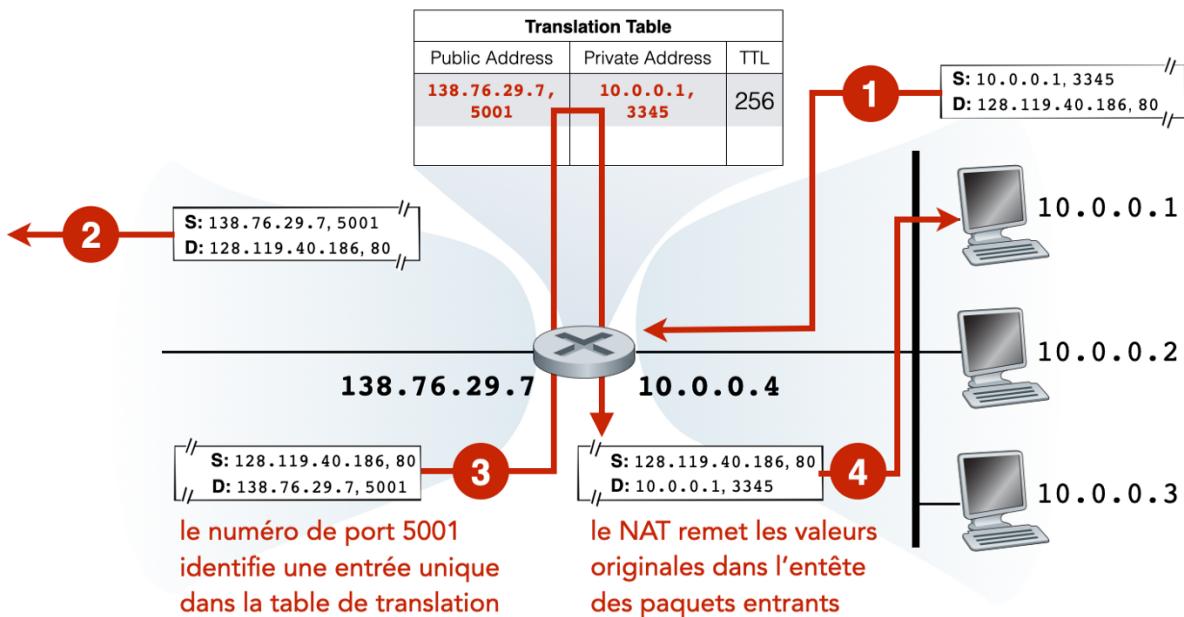
3. à maintenir les états qui permettent de remettre les adresses et les numéros de port contenus dans les entêtes IP et TCP des paquets entrants à leur valeur initiale.

Ce mécanisme est mise en œuvre par les organisations ne disposant pas d'adresses IP publiques en nombre suffisant pour numérotter l'ensemble des machines de leur réseau local.

Réseau privé et adresses IP privées. Pour contourner le manque en adresses IP, les machines d'un réseau local sont numérotées avec des adresses IP factices appelées **adresses IP privées**. Un tel réseau est de ce fait appelé **réseau privé**. Les adresses IP sont privées dans le sens où elles peuvent être utilisées uniquement pour les communications entre les machines connectées au réseau privé. De l'extérieur, toutes les machines du réseau privé sont masquées par une adresse IP publique unique. L'adresse IP publique est utilisée par la gateway du réseau privé qui est de ce fait, la seule machine joignable du réseau privé. C'est la gateway du réseau privé qui assume la translation des adresses IP et des numéros de ports contenus dans les entêtes des paquets IP entrants et sortants. Il en va de même des réseaux domestiques où la box Internet assume les fonctions de NAT.

Réécriture des entêtes. L'adresse IP source d'un paquet IP sortant est l'adresse IP privée de la machine à l'origine de ce paquet. NAT réécrit l'adresse IP source avec l'adresse IP publique du réseau privé. Plusieurs machines pouvant à un

instant donné émettre des paquets sortants simultanément, NAT réécrit également le numéro de port source. NAT exploite le fait que côté client, la valeur des numéros de port source est aléatoire pour leur donner une signification spécifique. NAT utilise une valeur unique qui permet d'identifier la machine à l'origine de chacun des paquets sortants. Du fait de changer la valeur des entêtes IP et TCP, NAT recalcule la somme de contrôle de l'entête IP et celle de l'entête TCP.



NAT et translation d'adresses.

Table de translation. NAT maintient une table de translation où est inscrit la correspondance entre adresse IP privée source et adresse IP publique d'une part et les valeurs avant et après réécriture du numéro de port source des paquets sortants.

C'est la valeur du numéro de port destination des paquets entrants qui identifie l'entrée contenant l'adresse IP privée et le

numéro de port de la machine à qui est destiné le paquet.

La durée de vie des entrées dans la table de translation est limitée par un TTL (Time to live) qui permet 1) de limiter la taille de cette table et 2) d'éviter les intrusions qui pourraient exploiter la présence de ces entrées en envoyant des paquets destinés aux numéros de ports ouverts par ces entrées.

Controverses. L'utilisation de NAT fait polémique.

- NAT va à l'encontre au principe de bout en bout : NAT modifie l'entête des paquets IP en réécrivant la valeur des adresses IP et le contenu des paquets IP en réécrivant les numéros de port TCP.
- NAT modifie la sémantique des numéros de port en leur faisant endosser le rôle d'identifier des machines hôtes en plus d'identifier les processus applicatifs.
- NAT installe des états dans Internet alors que le protocole IP est censé fonctionner sans état. Si la durée de vie de ces états est en théorie limitée par un TTL, il est compliqué de trouver une valeur réaliste concernant la durée du TTL.
- NAT bloque les paquets entrants pour lesquels un paquet sortant n'a pas préalablement installé une entrée dans la table de translation du NAT. Installer un serveur sur le réseau privé est possible à condition d'installer des états en dur dans la table de translation. Cependant, cette pratique ne permet pas d'avoir, sur un même réseau privé, plusieurs

serveurs tournant sur le même numéro de port.

Domain Name System (DNS)

DNS fait référence à plusieurs notions interdépendantes :

- Le format des noms utilisés pour identifier les machines dans Internet.
- La méthode utilisée pour attribuer un nom à une machine.
- La base de données répartie distribuée contenant les correspondances entre noms et adresses IP.
- L'ensemble des serveurs chargés de gérer une partie des correspondances entre noms et adresses IP.
- Le protocole de résolution des noms de machine en adresse IP.

Base de données et enregistrements DNS

DNS fait référence à une base de données dont les enregistrements (appelés Resource Records) sont composés des champs suivants :

| {*NAME*, *TYPE*, *CLASS*, *TTL*, *RDATA_LENGTH*, *RDATA*}

où :

- *NAME* correspond à un nom de domaine,
- *TYPE* (16) correspond au type de l'enregistrement DNS (i.e., A, AAAA, CNAME, MX, ...),
- *CLASS* (16) contient la valeur IN (0x0001) pour Internet,
- *TTL* (32) est la durée en secondes durant laquelle l'enregistrement restera valide,
- *RDATA_LENGTH* (16) contient la longueur du champ RDATA,
- *RDATA* contient la valeur correspondant au nom de domaine de cet enregistrement.

Les champs *NAME* et *RDATA_LENGTH* et *RDATA* dépendent de la valeur du champ *Type* de l'enregistrement DNS. Les types d'enregistrement les plus fréquents sont :

- Type A : les enregistrements de type A contiennent l'adresse IPv4 correspondant à un nom de domaine.
- Type AAAA : les enregistrements de type AAAA contiennent l'adresse IPv6 correspondant à un nom de domaine.
- Type CNAME : les enregistrements de type CNAME contiennent le nom canonique correspondant à un nom de domaine faisant office d'alias.
- Type NS : les enregistrements de type NS contiennent le nom du serveur d'autorité qui fait autorité sur le nom de domaine donné dans le champ *NAME*.

- Type MX : les enregistrements de type MX contiennent le nom du serveur de mail associé à un nom de domaine.

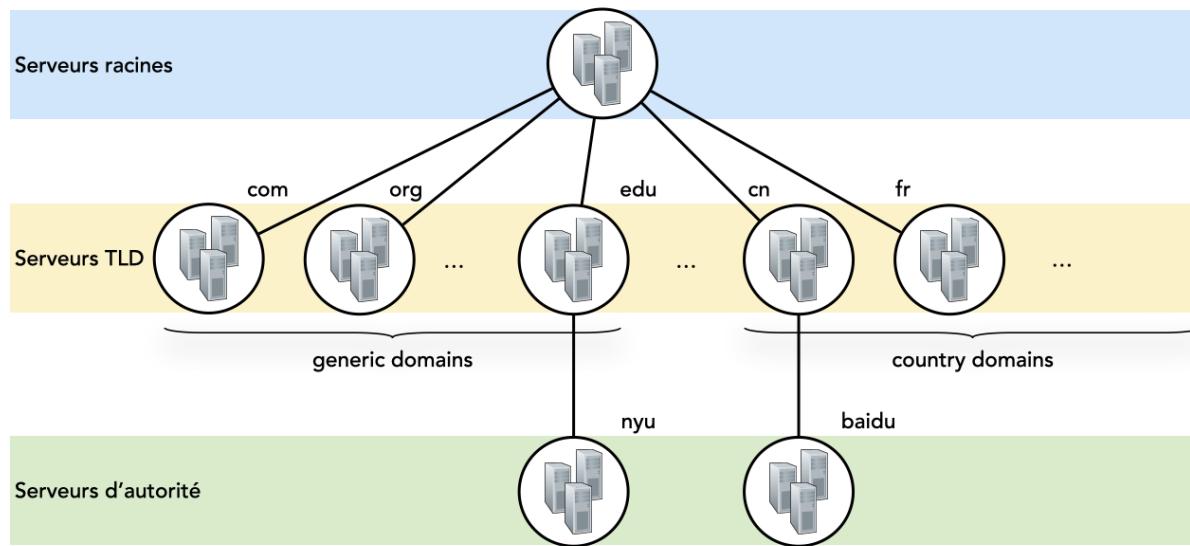
Serveurs DNS

Serveur d'autorité. Les enregistrements DNS sont contenus dans une base de données répartie. Cette base de données est divisée en plusieurs parties chacune maintenue par un serveur DNS appelé **serveur d'autorité**. Chaque partie correspond à un domaine particulier sur lequel un serveur a autorité : c'est au niveau de ce serveur que les enregistrements pour ce domaine sont créés et mis à jour.

Zone DNS. Un domaine contient donc les noms appartenant à une même organisation et pour lesquels elle fournit les enregistrements qui permettent leur résolution en adresses IPv4 et IPv6. Si un serveur d'autorité est chargé de maintenir les enregistrements de plusieurs domaines, l'ensemble de ces domaines forme une **zone DNS**. Le serveur d'autorité d'une zone DNS contient les enregistrements DNS de première main pour les domaines de cette zone.

Serveurs DNS racine et TLD. Pour déterminer quel serveur d'autorité est responsable d'un domaine en particulier, DNS repose sur deux autres types de serveurs DNS : les **serveurs racines** et les **serveurs TLD** (Top Level Domain ou domaine de premier niveau). Serveurs racines, TLD et d'autorités sont connectés entre eux selon une structure arborescence hiérarchique qui permet, par interrogation successive, de

déterminer le serveur d'autorité : les serveurs TLD ont pour serveur parent un serveur racine et les serveurs d'autorité ont pour parent un serveur TLD.

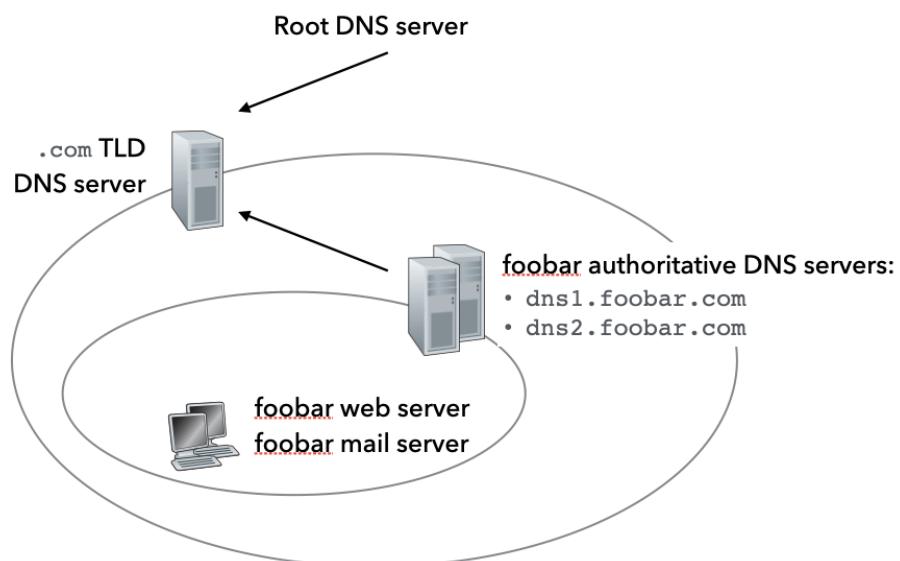


Hiérarchie arborescente des serveurs DNS.

Le serveur TLD parent d'un domaine est identifié par le label le plus à droite situé dans un nom de domaine. On distingue les labels TLD de pays (fr, cn, ...) et les labels TLD génériques (edu, org, ...). Les labels situés à gauche à la suite du label TLD identifie le domaine et les sous-domaines en charge de la ressource identifiée par ce nom de domaine.

Création d'un domaine

Pour créer un domaine, une institution doit s'assurer de l'unicité du nom choisi pour identifier son domaine. Pour ce faire, l'institution choisit le domaine de premier niveau TLD pertinent du point de vue de son activité à qui elle soumet le nom qu'elle a choisi, foobar.com par exemple.



Création du domaine foobar.com.

Une fois l'unicité du nom établi, le TLD installe dans son serveur un enregistrement DNS de type NS pour chacun des serveurs d'autorité en charge du domaine créé par l'institution. Le nom contenu dans un enregistrement DNS est celui du domaine et la valeur contient le nom du serveur d'autorité.

{foobar.com, dns1.foobar.com, NS}

{foobar.com, dns2.foobar.com, NS}

En plus des enregistrements de types NS, l'institution fournit les enregistrements de type A où le nom est celui des serveurs d'autorité et la valeur l'adresse IPv4 de ses serveurs d'autorité primaire et secondaires.

{dns1.foobar.com, 212.212.212.1, A}

{dns2.foobar.com, 212.212.212.2, A}

Les serveurs d'autorité de foobar.com contiennent les enregistrements de type A pour son serveur web et de type MX pour son serveur de mail.

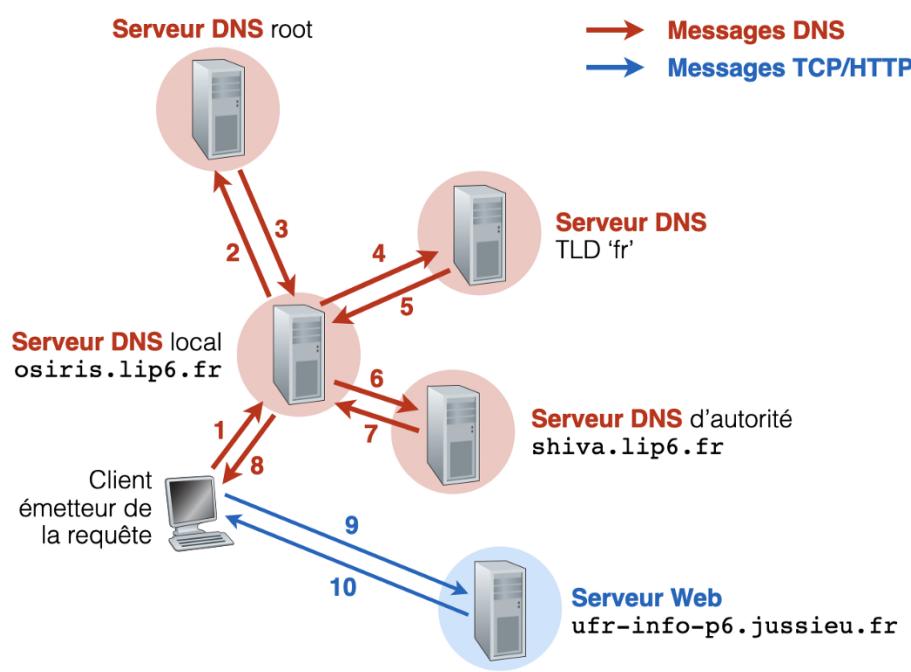
Résolution des requêtes DNS

Un client souhaitant obtenir un enregistrement DNS doit connaître un des serveurs racines DNS qu'il commence par interroger en soumettant le nom de domaine à résoudre. Le serveur DNS racine inspecte le dernier label contenu dans le nom de domaine et détermine ainsi le serveur TLD à interroger. Il s'agit du serveur TLD parent du domaine en charge du nom à résoudre. Le serveur TLD inspecte les labels situés immédiatement à gauche du TLD et identifie ainsi les serveurs d'autorité en charge du nom de domaine à résoudre. On distingue deux modes de résolution : le mode itératif, le mode récursif. Un troisième mode est le mode hybride.

- **Dans le mode itératif**, le client à l'origine de la requête s'adresse à un premier serveur DNS qui retourne l'enregistrement demandée ou l'adresse d'un autre serveur DNS que le client aura la charge d'interroger et ainsi de suite, jusqu'à ce que la réponse demandée lui parvienne.
- **Dans le mode récursif**, le client interroge le premier serveur DNS qui connaît la réponse ou qui s'adresse au serveur DNS suivant sans repasser par le client. Si le second serveur DNS a la réponse, il répondra au premier serveur

qui la renverra au client. Sinon le second serveur s'adressera à un troisième serveur qui procédera comme dans le cas du premier serveur et ainsi de suite.

- **Dans le mode hybride**, la requête est traitée en mode récursif entre le client et le premier serveur DNS et en mode itératif entre le premier serveur DNS et les serveurs DNS suivants : le premier DNS renvoie la réponse finale au client et se charge entre temps d'interroger à tour de rôle les serveurs DNS successifs, au fur et à mesure qu'il en découvre l'adresse.



Résolution des requêtes DNS en mode hybride.

Caching et Proxy DNS

En théorie, il est nécessaire d'interroger successivement au minimum trois serveurs avant d'obtenir l'enregistrement DNS demandé : le serveur racine, le serveur TLD et le serveur

d'autorité. Ce traitement des requête DNS 1) rallonge le temps nécessaire à la résolution des requêtes DNS et 2) augmente la charge des serveurs racines et dans une moindre mesure les serveurs TLD.

Pour réduire la charge sur le serveur DNS racine, une première solution consiste à le répliquer. C'est la raison pour laquelle il existe à ce jour 13 serveurs racines. Une seconde solution consiste à introduire un serveur proxy (mandataire) entre clients et serveurs DNS racines. C'est le rôle des serveurs DNS locaux que mettent à disposition les FAI pour leurs clients. Les clients n'ont plus à connaître les adresses IP des serveurs DNS racines. Ils doivent connaître l'adresse IP du serveur DNS local que le serveur DHCP leur communique au titre des paramètres nécessaires pour se connecter à Internet. C'est le serveur DNS local qui est configuré avec l'adresse IP des 13 serveurs DNS racines. Les serveurs DNS locaux facilitent l'installation des mises à jour. En cas de changement dans la liste des serveurs DNS racine, les administrateurs se contentent d'intervenir au niveau du serveur DNS local.

Une requête DNS est adressée en premier lieu au serveur DNS local qui l'adresse si nécessaire à un des 13 serveurs DNS racines. En l'absence de réponse, le serveur DNS local soumet à nouveau la requête au serveur DNS racine suivant et ainsi de suite. La requête est abonnée à la 16e tentatives.

En plus de prendre en charge les requêtes DNS, les serveurs

DNS locaux disposent de mémoires caches où sont maintenus temporairement les enregistrements DNS obtenus récemment. La durée de mise en mémoire cache d'un enregistrement DNS est déterminée par la valeur de son champ TTL. Les caches DNS sont présents sur les clients et tous les serveurs DNS. Ils permettent ainsi de réduire la durée nécessaire à la résolution d'une requête DNS et la charge des serveurs situés en aval dans la hiérarchie arborescente DNS.

Encodage des noms de domaine

Un nom de domaine peut être fourni sous la forme d'une séquence de labels de données. Un **label de données** commence avec un octet qui indique la longueur du label. S'il s'agit du dernier label contenu dans le nom, il est suivi du label de fin 0x00.

- Exemple: la séquence de labels de données correspondant au nom www.upmc.fr en hexadécimal est le suivant :

0x03777770475706D6302467200

1. Label de données 1 : 0x0377777. Le premier octet indique une longueur de 3 octets (0x03). Les trois octets suivants 0x777777 correspondent aux codes ASCII des lettres ‘www’.
2. Label de données 2 : 0x0475706D63. Le premier octet indique une longueur de 4 octets (0x04). Les quatre octets suivants 0x75706D63 correspondent aux codes ASCII des lettres ‘upmc’.

3. Label de données 3 : 0x026772. Le premier octet indique une longueur de 2 octets. Les deux octets suivants 0x6772 correspondent aux codes ASCII des lettres ‘fr’.
4. Label de fin : 0x00.

Nous verrons plus loin dans ce chapitre qu’un nom de domaine peut être fourni sous une forme compressée.

Messages DNS

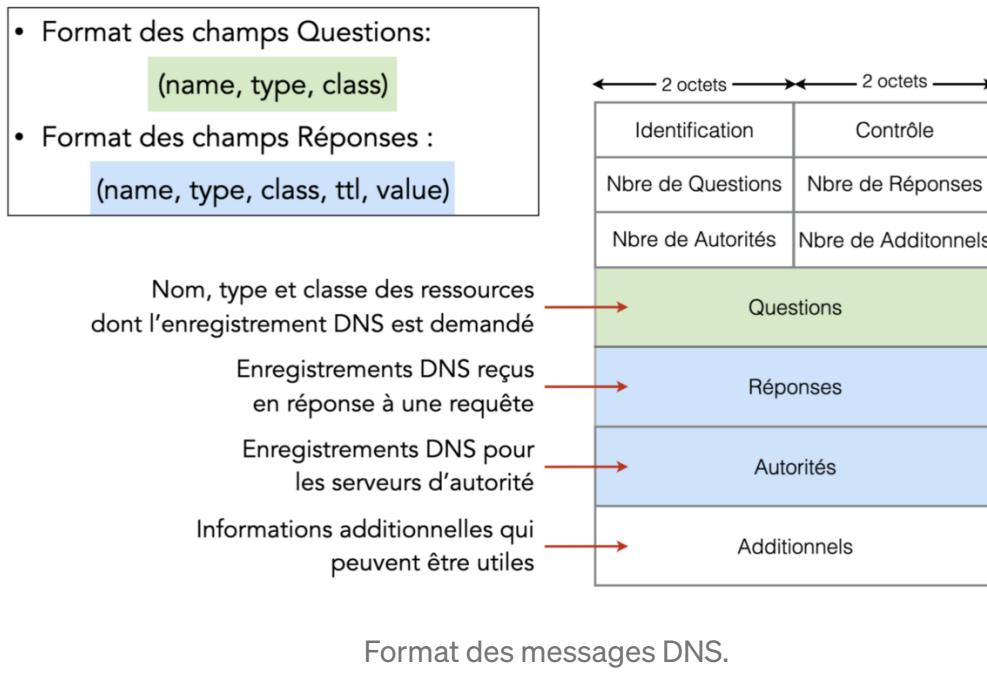
Un client DNS souhaitant résoudre un nom de domaine envoie une requête DNS à un serveur DNS. En retour il reçoit une réponse contenant le ou les enregistrements correspondant au nom de domaine. Qu’il s’agisse des requêtes ou des réponses, tous les messages DNS partagent le même format.

Les messages DNS sont encapsulés dans un datagramme DNS où le numéro de port identifiant le client a une valeur aléatoire et le numéro de port côté serveur a la valeur 53 (0x35).

Un message DNS contient un entête suivi des quatre sections suivantes, certaines sections pouvant être vides :

- Question
- Réponse
- Autorité
- Additionnel

Une section non vide contient un ou plusieurs enregistrements DNS.



L'entête DNS comprend les six (6) champs longs de 16 bits suivants :

- Identification
- Contrôle
- Nombre de questions
- Nombre de réponses
- Nombre d'autorités
- Nombre d'additionnel

Champ Identification (16) permet à une source à l'origine

d'une requête de mettre en correspondance la réponse DNS à la requête DNS.

Champ Contrôle (16) contient les sous-champs suivants :

- Bit QR (1) : si à 0 indique qu'il s'agit d'une requête, d'une réponse sinon.
- Bits OpCode (4) : indique le type de requête. La valeur 0b0000 indique une requête standard.
- Bit AA (1) : si à 1 indique que la réponse provient d'un serveur d'autorité, d'un cache sinon.
- Bit TC (1) : si à 1 indique que le message DNS a été tronqué car trop long.
- Bit RD (1) : si à 1 indique que le client souhaite que sa requête soit traitée de manière récursive, itérative sinon.
- Bit RA (1) : si à 1 indique que le serveur accepte les requêtes récursives, itératives sinon.
- Bit Zero (1) : bit réservé.
- Bit AD (1) : bit utilisé pour DNSSEC.
- Bit CF (1) : bit utilisé pour DNSSEC.
- Bits Rcode (4) : contient un code d'erreur (0b0000 NOERROR).

Nombre de ... (16) : les quatre derniers champs indiquent le nombre d'enregistrements contenu dans chacune des quatre

sections situées à la suite de l'entête.

Les enregistrements DNS compris dans la section Question sont incomplets. Ces enregistrements contiennent les champs suivants :

| {*QNAME*, *QTYPE*, *QCLASS*}

où :

- *QNAME* est le nom de domaine pour lequel la requête est envoyée. Le nom de domaine est précisé sous la forme d'une séquence de label de données.
- *QTYPE* (16) est le type de l'enregistrement DNS demandé (i.e., A, AAAA, CNAME, MX, ...).
- *QCLASS* (16) contient la valeur IN (0x0001) pour Internet.

Les enregistrements listés dans dans les sections Réponse, Autorité et Additionnel contiennent des enregistrements DNS complets tel qu'enregistrer dans la bases de données DNS.

Compression des messages DNS

Un message DNS contient quatre sections pouvant chacune contenir plusieurs enregistrements comprenant chacun un ou plusieurs noms de domaine. Un même message DNS peut donc contenir plusieurs noms de domaine. Un même nom ou la même portion d'un nom peuvent être dupliquées dans

plusieurs des enregistrements DNS contenus dans le message DNS. Ces duplications augmentent la taille des messages et donc la bande passante nécessaire pour les véhiculer.

Une technique appelée compression des messages DNS a été mise en place afin de réduire la taille des messages DNS. Elle consiste à supprimer les informations dupliquées. Cette technique agit sur les noms contenus dans les enregistrements DNS compris dans les quatre sections des messages DNS.

Conformément à cette technique, un nom de domaine peut être fourni sous la forme d'une séquence de labels de données, d'un label de compression ou d'un mix des deux.

- Un label de données comprend un premier octet qui indique la longueur en octets du label. Si le dernier label de données d'un nom de domaine est suivi du label de fin 0x00.
- Un label de compression fait référence à un label de données ou une séquence de labels de données situés en amont dans le message DNS. Au lieu de recopier le ou les labels de données, le label de compression indique où se trouve la première instance du label de données à réutiliser. Tous les labels situés à la suite sont recopiés jusqu'au label de fin qui termine tous les noms DNS. Un label est long de deux octets. La valeur des deux premiers bits sont positionnés à 1 (0b11). Les 14 bits suivants indique la position du label de données qui a été supprimé pour éviter sa duplication. La position correspond au nombre d'octets

qui sépare le premier label de données à recopier du début du message DNS.

0xC010 est un label de compression qui indique que le premier label de données à recopier est situé 16 (0x0010) octets après le début du message DNS. Tous les labels situés à la suite de ce premier label de données sont recopiés jusqu'au label de fin 0x00.

Réseaux Locaux (LAN)

Un **réseau local (Local area network)** connecte les machines appartenant à une même organisation telle qu'une PME ou une université. De ce fait, les réseaux locaux ont une portée géographique limitée et peuvent connecter plusieurs centaines de machines. Un réseau local offre aux machines hôtes qu'il connecte les services résultant de la présence d'une passerelle par défaut (gateway), de serveurs DCHP et de serveurs DNS locaux. Un réseau local est composé d'un réseau physique unique ou de plusieurs réseaux physiques interconnectés par des équipements spécifiques.

Contrairement aux routeurs, les équipements chargés d'interconnecter les réseaux physiques au sein d'un même réseau local n'implémentent pas la couche Réseau (3). **Les commutateurs** (switches) et **les ponts** (bridges) sont des équipements qui implémentent les couches Physique (1) et Liaison de données (2). **Les répéteurs et les concentrateurs** (hubs) sont des équipements qui implémentent uniquement la

couche Physique (1).

Quelque soit les équipements d'interconnexion du réseau local, leur présence est transparente : pour communiquer entre elles, les machines hôtes du réseau local n'ont pas à connaître leur existence. Le seul équipement que les machines hôtes d'un réseau local doivent connaître est la gateway du réseau local qui permet l'acheminement des paquets IP dont la destination est distante.

Pour comprendre les différences entre commutateurs et ponts d'un côté, et répéteurs et concentrateurs de l'autre, nous devons introduire les autres fonctions assurées par la couche **Liaison de données (2)** telles que le contrôle d'accès au support

à diffusion naturelle aux réseaux locaux dits à diffusion naturelle.

Réseaux locaux à diffusion naturelle

Les fonctions de la couche Liaison (2) dépendent des propriétés du réseau local notamment concernant sa topologie. Les réseaux physiques les plus répandus sont les **réseaux dits à diffusion naturelle**. L'envoi d'un message en un seul exemplaire suffit pour qu'il soit reçu par l'ensemble des autres machines connectées au même réseau physique.

A la façon de l'air qui transporte notre voix, s'adresser à une seule personne ou à une large assemblée revient au même : les tympans des auditeurs sont en contact direct avec l'air qui véhicule la voix de l'orateur. Du fait que l'air soit un canal de

communication partagé, un message oral unique est reçu de quiconque situé à une portée suffisamment proche.

Confidentialité. Une première conséquence résultant du partage du canal de communication est l'**absence de confidentialité** : une machine située sur le même réseau local est à même de recevoir les messages, y compris ceux dont elle n'est pas le destinataire.

Collisions. Une seconde conséquence découle des prises de paroles simultanées : du fait que l'air soit partagé par l'ensemble de l'assemblée, un seul participant doit prendre la parole sous peine de cacophonie. Sur un canal de communication, les informations sont transmises sous la forme d'un signal électromagnétique caractérisé notamment par sa fréquence. La superposition de deux transmissions simultanées émises sur une même plage de fréquence entraîne une **collision** qui rendent les deux signaux inintelligibles.

Pour éviter les collisions et donc le gaspillage de la bande passante consommée par les transmissions simultanées, il est nécessaire qu'une seule machine transmette à un instant donné. Déterminer si une machine peut transmettre à un instant donné et ce, en évitant une éventuelle collision, c'est le rôle des méthodes de contrôle d'accès implémentées par la couche Liaison de données (2).

Méthodes de contrôle d'accès

Dans le cas d'un auditoire composé de plusieurs personnes, l'orateur doit rester le seul à s'exprimer sous peine de ne pas être compris. Sur un réseau local à diffusion naturelle, il en va de même : si deux stations transmettent simultanément, une **collision** s'ensuivra et les signaux résultant des deux transmissions deviennent inintelligibles.

Afin d'éviter les collisions, la couche Liaison de données (2) implémente une **méthode de contrôle au medium de communication** (Medium access control ou MAC). Il s'agit d'un algorithme distribué qui détermine la station autorisé à transmettre à un instant donné. Les algorithmes les plus répandus sont ceux implantés par Ethernet et Wifi.

Ethernet (CSMA/CD). Ethernet fait référence au CSMA/CD (Carrier sense multiple access with collision detection) qui est un algorithme distribué dont le but est de réduire les risques de collision sur les réseaux filaires à diffusion naturelle. Une collision résulte des transmissions simultanées sur un tel réseau. Le CSMA/CD permet de détecter les collisions et de limiter le risque qu'une collision se produise à nouveau lors des tentatives de transmission suivantes.

Le principe du CSMA/CD repose sur la capacité des machines de recevoir en même temps qu'elles transmettent. Elles utilisent pour ce faire un équipement matériel appelé **transceiver (trans-mitter/re-ceiver)**. Cet équipement leur permet de détecter les collisions : si le signal qu'une machine transmet ne

correspond pas au signal qu'elle reçoit au même instant, la machine en déduit qu'il y a collision.

En cas de collision, les machines impliquées arrêtent leur transmission respective et transmettent une **séquence de brouillage (jam sequence)**. Le but de la séquence de brouillage est d'informer toutes les autres machines du réseau local de la collision, y compris celles qui n'étaient pas impliquées dans une des transmissions à l'origine de la collision. Après transmission de la séquence de brouillage, les machines dont la transmission a échoué déroulent un algorithme de retrait aléatoire exponentiel décrit ci-après.

En plus de détecter les collisions, le CSMA/CD tente de limiter les risques de collision en mettant en œuvre :

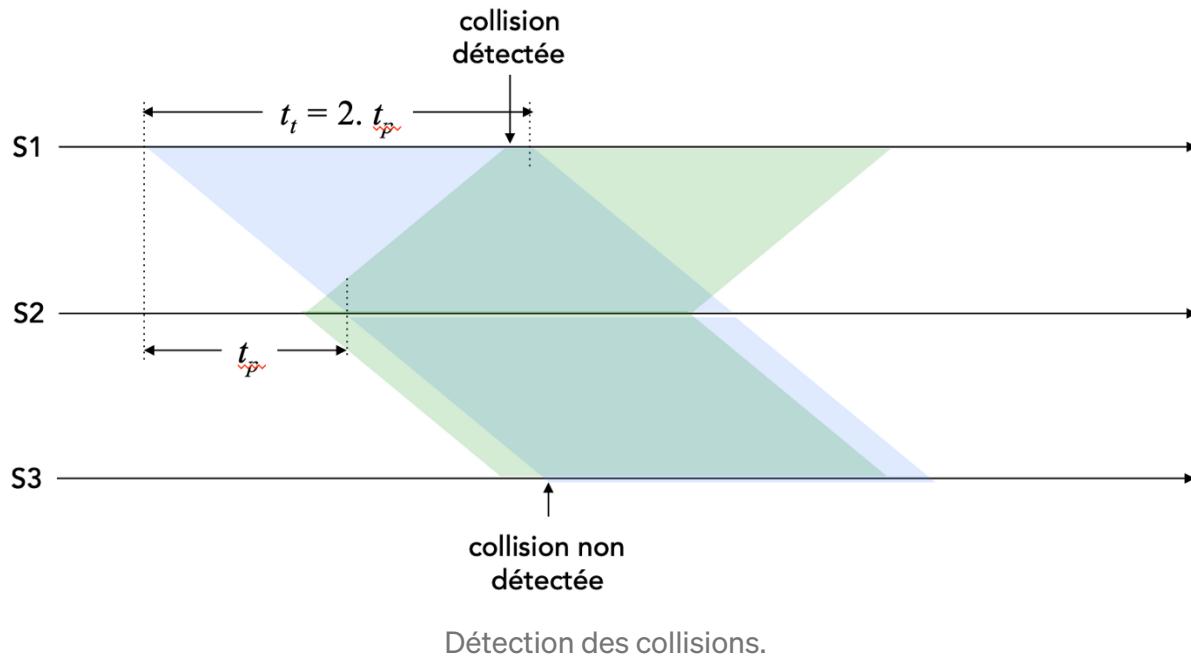
1. **L'écoute préalable du support** : avant de transmettre, une machine écoute le support pour vérifier qu'il est libre. Si tel est le cas, la machine transmet sa trame. Si le support est occupé, la machine continue son écoute dans l'attente que celui-ci se libère. La machine commence à transmettre sa trame immédiatement à la libération du support. La machine continue l'écoute du support, cette fois-ci dans le but de détecter une collision éventuelle. Une fois sa transmission terminée en succès, la machine doit observer une période de silence appelée **interframe gap** avant de transmettre à nouveau. Ce silence rend le CSMA/CD équitable puisqu'il permet aux autres machines en attente

de transmission de détecter le support libre et donc d'avoir une chance de transmettre.

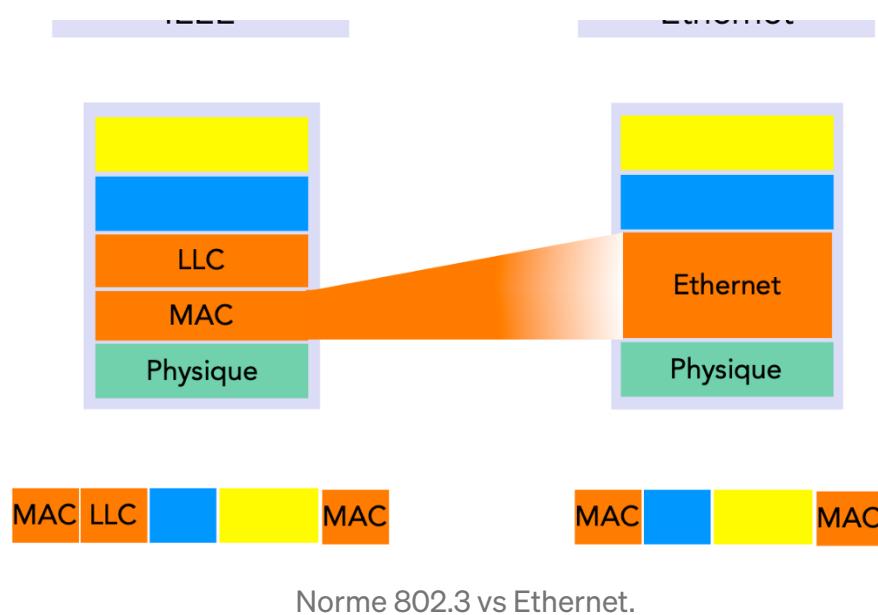
2. Le retrait aléatoire exponentiel : en cas de collision, les machines impliquées annulent leur transmission et déroule un algorithme de retrait aléatoire exponentiel. Elles tirent aléatoirement la valeur d'un temporisateur à l'expiration duquel elles tenteront de transmettre à nouveau leur trame (après s'être assurées que le support est libre). Le fait de tirer aléatoirement la valeur du temporisateur permet de diminuer le risque de collision sans pourtant éliminer ce risque. Des collisions successives peuvent advenir dans un réseau local où un nombre élevé de machines tentent de transmettre simultanément. Pour *calmer les ardeurs* des machines, la solution consiste à échelonner les tentatives de transmission sur une période de temps plus longue. Pour ce faire, la plage des valeurs que peut prendre le temporisateur est multipliée par deux en cas de collisions successives. C'est dans ce sens que l'algorithme de retrait est exponentiel en plus d'être aléatoire. A la 16ème tentative, une machine renonce en abandonnant la transmission de sa trame.

La détection des trames en collision impose une contrainte sur la longueur des trames. Pour être en mesure de détecter que sa transmission est en échec en raison d'une collision, une machine doit transmettre assez longtemps pour être en mesure de recevoir le signal résultant de la collision. Cette durée correspond au délai aller-retour maximal sur le réseau local. La durée de transmission d'une trame doit donc être égale à deux

fois le temps de propagation maximal sur le réseau local. Cette contrainte se traduit pour Ethernet par des trames longues au minimum de 64 bits.



Le CSMA/CD a été standardisé par l'IEEE dans la norme numéroté 802.3. La valeur 802 fait référence au groupe de travail, créé en février 1980, responsable au sein de l'IEEE des normes concernant les réseaux locaux (LAN). Ethernet était la version du CSMA/CD déjà commercialisée avant la parution de la norme 802.3. De ce fait, certains des changements introduits par cette norme n'ont pas été pris en compte par les constructeurs. Ces changements concernent entre autre le format des trames et la présence de la sous-couche LLC (Logical link control) située entre la sous-couche MAC et la couche Réseau (3).



Interconnexion intra-LAN

Un réseau local offre aux machines d'une organisation des services rendus possibles par la présence d'une passerelle par défaut (gateway) et de serveurs tels que DHCP et DNS. Le protocole DHCP tout comme ARP dépendent de la possibilité de diffuser (broadcast) des trames à toutes les machines du réseau local.

Les réseaux physiques à diffusion naturelle sont de ce fait adaptés aux services dont dépendent les machines hôtes pour accéder à Internet. Cependant, un réseau physique est limité par sa portée et le nombre de machines qu'il peut connecter. Pour étendre sa portée et sa capacité d'accueil, un réseau local est généralement composé de plusieurs réseaux physiques qu'il s'agit de connecter entre eux.

Contrainte de conception 1. Les équipements utilisés pour interconnecter les réseaux physiques au sein d'un réseau local

doivent être **tolérants à l'égard des trames broadcastées** (adresse MAC destination FF:FF:FF:FF:FF:FF). Un équipement qui interconnecte deux réseaux physiques dans un réseau local doit laisser passer ces trames. Ce n'est pas le cas des routeurs qui bloquent les paquets broadcastés (adresse IP destination 255.255.255.255). De plus, des boucles pouvant apparaître dans la topologie décrivant l'interconnexion des réseaux physiques, il est nécessaire d'empêcher les trafics de tourner à l'infini, les trames Ethernet étant dépourvues de champ TTL.

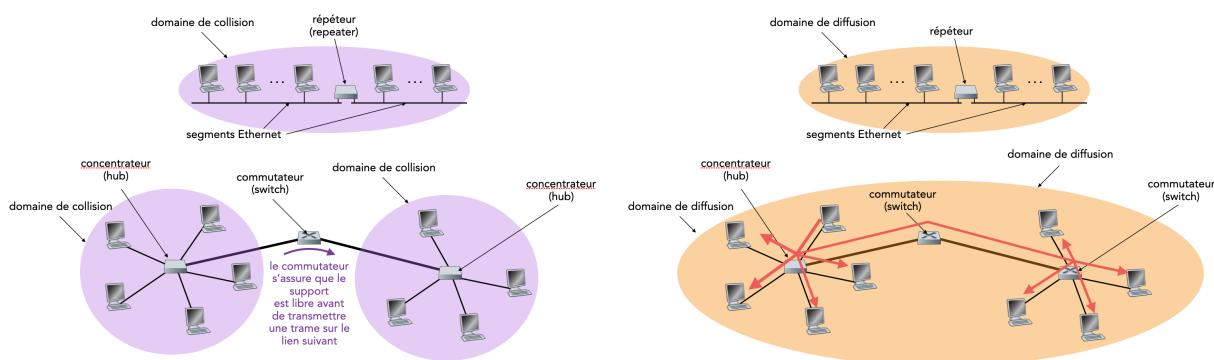
Augmenter le nombre de machines dans un réseau local augmente inexorablement la charge du réseau et diminue les opportunités de transmettre. Une transmission initiée sur un des réseaux physiques retardent inutilement les transmissions sur les autres réseaux physiques, en particulier lorsque la destination ne s'y trouve pas.

Contrainte de conception 2. Les équipements d'interconnexion internes aux réseaux locaux doivent assurer la compartimentation des trafics. Compartimenter le trafic consiste à bloquer les trames en entrée des ports qui ne mènent pas à la destination des trames. En compartimentant les trafics, plusieurs transmissions peuvent advenir en parallèle ce qui garantit une meilleure utilisation de la bande passante du réseau local.

Contrainte de conception 3. Il est également nécessaire d'éviter les collisions résultant des transmissions simultanées

sur des réseaux physiques distincts. Pour ce faire, l'équipement d'interconnexion doit participer au CSMA/CD en même temps que les machines situées sur les réseaux physiques qu'il connecte.

Commutateurs et ponts. Pour répondre à ces contraintes, les commutateurs ont été introduits. Un commutateur est un équipement d'interconnexion doté de plusieurs interfaces réseaux appelés **ports**. Un port peut connecter une machine, un commutateur ou un bus. Dans ce dernier cas, on parle de **pont**.



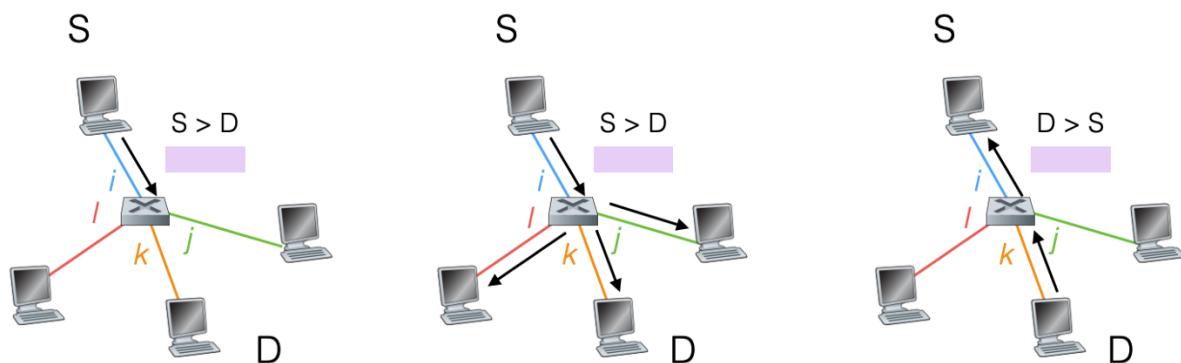
Domaines de collision vs domaine de diffusion.

Les fonctions assurées par un commutateur sont les suivantes :

- Le CSMA/CD (cf. contrainte de conception 3)
- L'auto-apprentissage (cf. contrainte de conception 2)
- Le Spanning tree protocol (STP) (cf. contrainte de conception 1)

CSMA/CD. Pour transmettre une trame, un commutateur déroule l'algorithme du CSMA/CD avec les machines connectées sur chacun de ses ports de sortie. Il évite ainsi de provoquer des collisions sur des réseaux physiques autres que celui où les trames ont été initialement transmises.

Auto-apprentissage. Il s'agit pour un commutateur de déterminer pour chacun de ses ports, la liste des machines qu'il permet de joindre. Pour ce faire, un commutateur est doté d'un table appelé **CAM table** qu'il peuple en inspectant l'adresse source des trames. La table CAM comporte des entrées composé d'une adresse MAC et du port qui permet de joindre la machine à qui appartient cette adresse. Les entrées ont une durée de vie limitée par la valeur d'un TTL.



S	i

S	i
D	k

Auto-apprentissage.

Lorsqu'il reçoit une trame à destination d'un adresse D pour

laquelle il n'a pas d'entrée dans sa table CAM, le commutateur crée une entrée pour l'adresse source S de la trame qu'il associe au port d'entrée de la trame. Une fois cette entrée créée, le commutateur envoie la trame sur l'ensemble de ses interfaces hormis celle sur laquelle la trame est arrivée. La trame est ainsi assurée d'atteindre D qui finira par retourner une trame à la machine S . La trame reçue en retour entraînera la création d'une entrée supplémentaire concernant D . L'apprentissage découlant des trames que les commutateurs inondent. Le rôle d'un commutateur n'est donc évidemment pas de bloquer l'inondation des trames.

L'algorithme d'apprentissage que déroule un commutateur sur réception d'une trame provenant de S sur le port d'entrée i et à destination de D est le suivant :

- Chercher l'adresse de D dans la table CAM.
- Si l'entrée est trouvée et indique un port identique à i , le port d'entrée de la trame, le commutateur supprime la trame.
- Si l'entrée trouvée indique un port différent de i , le commutateur envoie la trame sur ce port.
- Si aucune entrée n'est trouvée, le commutateur 1) crée une entrée pour S qu'il associe à i et 2) inonde la trame sur tous ses ports à l'exception de i .

La durée de vie des entrées de la table CAM est limitée par un

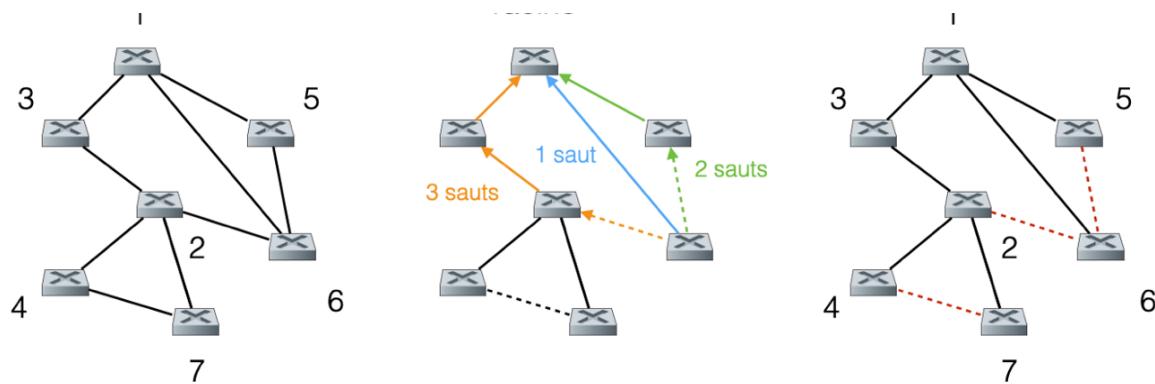
TTL à l'expiration duquel une entrée est supprimée. Le TTL permet :

1. de maintenir la taille des tables CAM réduite en supprimant les entrées les moins récemment utilisées,
2. de forcer la mise à jour des entrées notamment pour prendre en compte les machines qui auraient changé d'adresse ou de point d'attachement dans le réseau.

Spanning tree protocol. Un réseau local peut contenir plusieurs commutateurs. La topologie qui décrit l'interconnexion des commutateurs peut contenir des boucles. Ces boucles peuvent résulter d'une erreur de câblage ou être volontairement créées pour introduire plusieurs chemins alternatifs entre paires de commutateurs. L'existence de plusieurs chemins alternatifs augmente la résistance du réseau local au rupture de liens et aux défaillances de commutateurs.

Les trames Ethernet étant dépourvues de champ TTL, il est nécessaire de les empêcher de tourner à l'infini. Pour ce faire, les commutateurs exécute le protocole STP.

Le STP consiste à créer un arbre couvrant qui contient les seuls liens le long desquels les trames seront acheminées. Le STP empêche ainsi la circulation des trames sur les liens à l'origine des boucles.



Arbre couvrant STP.

Les ingrédients du STP sont les suivants :

- **Élection de la racine de l'arbre STP :** un arbre est caractérisé par sa racine. Dans le cas du STP, la racine est le commutateur dont l'identifiant est le plus petit dans le réseau local. L'identifiant par défaut est l'adresse MAC la plus petite du commutateur. La valeur des identifiants par défaut peut être altérée par l'utilisation d'une valeur ajoutée en tête des adresses MAC appelée **priorité**.
- **Identification du port racine :** les commutateurs identifient le port qui leur permet de joindre le commutateur racine précédemment élue le long du chemin le plus court. Ce port est appelé **port racine**. Une fois le port racine déterminé, les commutateurs bloquent les autres ports en installant un état dont la durée de vie est limitée par la valeur d'un TTL. Un port précédemment bloqué peut ainsi être débloqué si la racine de l'arbre change.

Pour élire la racine de l'arbre STP et identifier le port racine, le STP utilise des messages que les commutateurs envoient

périodiquement à leurs voisins directs. Ces messages contiennent trois champs (X, d, Y) :

- Y est l'identifiant du commutateur à l'origine du message,
- X est l'identifiant du commutateur racine du point de vue de Y et
- d est la distance du chemin le plus court qui sépare Y de X.

La distance représente la nombre de commutateurs que comprend ce chemin, les liens du réseau local étant configurés par défaut avec un coût égal à 1. Les liens peuvent être configurés avec des coûts différents pour modifier la configuration de l'arbre STP.

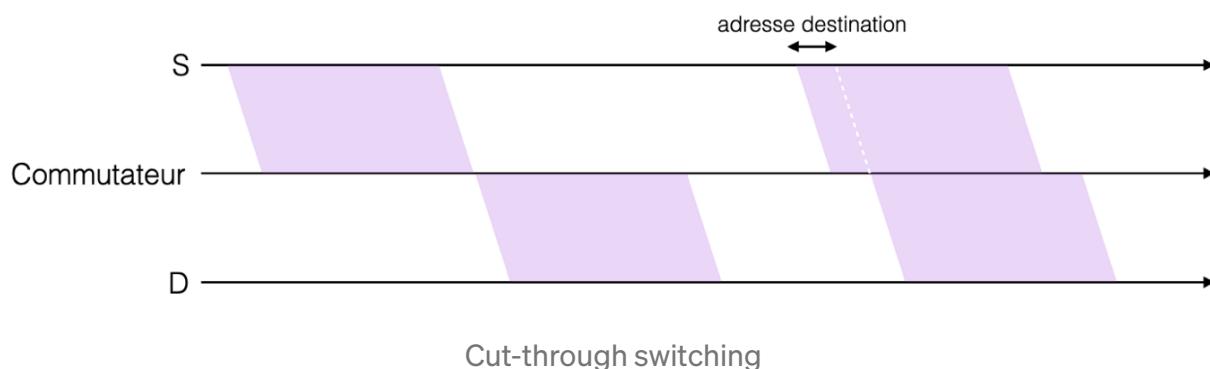
L'algorithme du STP fonctionne de la façon suivante :

- Initialement, tous les commutateurs s'autoproclament racine de l'arbre : le commutateur X envoie le message (X, 0, X) à ses voisins directs et reçoit à son tour les messages envoyés par ses voisins directs.
- Si X reçoit de son voisin Y le message (Z, d, Y) où Z, l'identifiant de la racine annoncée par Y, est inférieur à celui de X, X remplace alors l'identifiant du commutateur qu'il pense être la racine par Z et met à jour la distance du chemin le plus court qui le sépare de la racine avec la valeur $d+1$. X marque le port qui lui permet de joindre Y comme étant le port racine et bloque tous ses autres ports.

- X envoie à présent le message $(Z, d+1, X)$ périodiquement à ses voisins directs et jusqu'à ce qu'il reçoive un message annonçant une distance plus courte pour Z ou un message annonçant une racine dont l'identifiant est inférieure à Z. X mettra alors à jour son port racine et les champs concernant l'identifiant de la racine et la distance qui le sépare de cette racine.

Les changements de topologie du réseau local sont pris en compte, les messages STP étant envoyés périodiquement et la durée de vie des états installés concernant les ports bloqués et le port racine étant limitée par un TTL.

Cut-through switching. Le fonctionnement par défaut d'un commutateur requiert qu'une trame soit reçue dans sa totalité avant que le commutateur commence à la transmettre sur un ou plusieurs de ses ports de sortie. Ce mécanisme retarde l'acheminement des trames et augmente le prix des commutateurs, ces derniers devant disposer de capacités de stockage conséquentes.



Le **cut-through switching** propose une alternative qui consiste à transmettre les trames aussi tôt que possible. Pour rappel, le premier champ d'entête contient l'adresse destination de la trame. Un commutateur peut consulter sa table CAM aussitôt ce champ reçu. Une fois le port de sortie déterminée, le commutateur commence alors à transmettre la trame sur ce port avant que la trame n'ait été reçue dans sa totalité. C'est ainsi que le cut-through switching accélèrent l'acheminement des trames. L'inconvénient du cut-through switching découle du fait que le champ FCS nécessaire pour détecter les trames en erreur arrive en dernier dans la trame. Les commutateurs ne sont plus en mesure de vérifier l'intégrité des trames et de rejeter celles en erreur. Le cut-through switching présente donc l'inconvénient de propager les erreurs.

A suivre ...

[Internet](#) [Education](#) [Lecture Notes](#) [IP](#) [Tcp](#)

[About](#) [Write](#) [Help](#) [Legal](#)