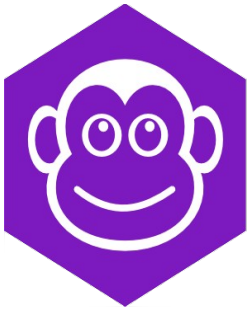
Séquence 2: ”Le protocole IPv6”

Bruno Stévant, Jacques Landru, Jean-Pierre Rioual, Véronique Vèque, Pascal Anelli



MOOC

Objectif IPv6 !

vers l'internet nouvelle génération

Document Compagnon

Séquence 2

Le protocole IPv6

Licence Creative Nommons CC BY-SA 4.0 International

Attribution - Partage dans les Mêmes Conditions 4.0 International (CC BY-SA 4.0)

Avertissement Ce résumé n'indique que certaines des dispositions clé de la licence. Ce n'est pas une licence, il n'a pas de valeur juridique. Vous devez lire attentivement tous les termes et conditions de la licence avant d'utiliser le matériel licencié.

Creative Commons n'est pas un cabinet d'avocat et n'est pas un service de conseil juridique. Distribuer, afficher et faire un lien vers le résumé ou la licence ne constitue pas une relation client-avocat ou tout autre type de relation entre vous et Creative Commons.

Clause C'est un résumé (et non pas un substitut) de la licence.

http://creativecommons.org/licenses/by-sa/4.0/legalcode

Vous êtes autorisé à :

- **Partager** — copier, distribuer et communiquer le matériel par tous moyens et sous tous formats
- **Adapter** — remixer, transformer et créer à partir du matériel
- pour toute utilisation, y compris commerciale.

L'Offrant ne peut retirer les autorisations concédées par la licence tant que vous appliquez les termes de cette licence.

Selon les conditions suivantes :

Attribution — You must give **appropriate credit**, provide a link to the license, and **indicate if changes were made**. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

Partage dans les Mêmes Conditions — Dans le cas où vous effectuez un remix, que vous transformez, ou créez à partir du matériel composant l'Oeuvre originale, vous devez diffuser l'Oeuvre modifiée dans les même conditions, c'est à dire avec **la même licence** avec laquelle l'Oeuvre originale a été diffusée.

No additional restrictions — Vous n'êtes pas autorisé à appliquer des conditions légales ou des **mesures techniques** qui restreindraient légalement autrui à utiliser l'Oeuvre dans les conditions décrites par la licence.

Notes: Vous n'êtes pas dans l'obligation de respecter la licence pour les éléments ou matériel appartenant au domaine public ou dans le cas où l'utilisation que vous souhaitez faire est couverte par une **exception.**

Aucune garantie n'est donnée. Il se peut que la licence ne vous donne pas toutes les permissions nécessaires pour votre utilisation. Par exemple, certains droits comme l**es droits moraux, le droit des données personnelles et le droit à l'image** sont susceptibles de limiter votre utilisation.

Les informations détaillées sont disponibles aux URL suivantes :

- http://creativecommons.org/licenses/by-sa/4.0/deed.fr
- http://fr.wikipedia.org/wiki/Creative_Commons

Les auteurs

Bruno Stévant

Bruno STEVANT est enseignant chercheur à l'IMT Atlantique. Il intervient dans l'enseignement et sur les projets de recherche autour d'IPv6 depuis plus de 10 ans. Il est secrétaire et responsable des activités de formation de l'association G6, association pour la promotion et le déploiement d'IPv6 en France.

Jacques Landru

Enseignant chercheur au CERI - Systèmes Numériques à l'IMT Nord Europe, Jacques est responsable de l'UV de spécialisation ARES (Architecture des RESeaux) à la fois dans le mode traditionnel présentiel que dans sa déclinaison à distance dans le cadre de la filière apprentissage.

Jean-Pierre Rioual

Ingénieur Conseil Réseaux – EURÊKOM. Fort de 30 années d'expérience dans le domaine des réseaux, il intervient auprès des entreprises pour des missions d'expertise sur leurs réseaux de transmission de données (intégration, mesures, optimisation, administration), conçoit et anime des actions de formation "réseaux".

Véronique Vèque

Véronique Vèque est Professeur des Universités à l'Université Paris-Saclay. Elle enseigne les réseaux depuis plus de 20 ans en Master Réseaux et Télécoms. Elle poursuit ses recherches au sein du L2S (Laboratoire des Signaux et Systèmes) où elle est responsable de l'équipe Réseaux, optimisation et codage. Elle est directrice-adjointe de l'école doctorale STIC de l'Université Paris-Saclay.

Pascal Anelli

Pascal ANELLI est enseignant-chercheur à l'Université de la Réunion. Il enseigne les réseaux depuis plus 20 ans. Il est membre du G6 depuis sa création. A ce titre, il est un des contributeurs du livre IPv6. En 1996, il a participé au développement d'une version de la pile IPv6 pour Linux.

Remerciements à :

- Vincent Lerouvillois, pour son travail de relecture attentive ;
- Joël GROUFFAUD (IUT de la Réunion) ;
- Pierre Ugo TOURNOUX (Université de la Réunion) ;
- Bruno Di Gennaro (Association G6) ;
- Bruno Joachim (Association G6) pour sa contribution à l'activité « Contrôler la configuration réseau par DHCPv6 » ;
- Richard Lorion (Université de la Réunion) pour sa contribution à l'activité « Etablir la connectivité IPv6 tunnels pour IPv6 ».

----- oOo -----

Tables des activités

Activité 20 : L'architecture des protocoles de l'Internet

L'Internet est une interconnexion de réseaux physiques. Pour réaliser cette interconnexion, une couche est superposée à chaque noeud. Cette couche dite de réseau met en oeuvre le protocole IP afin de rendre le service de connectivité qui consiste à transférer des paquets d'une source à la destination. Un protocole se définit comme les règles et le format des échanges entre au moins 2 entités communicantes en vue de réaliser une action ou de rendre un service. Nous avons vu dans la séquence précédente la notion d'adressage qui est indispensable pour identifier et localiser les noeuds du réseau. Nous allons dans cette séquence apprendre comment les communications de données sont mises en oeuvre dans l'Internet.

Le protocole de réseau IP

Le protocole IP (*Internet Protocol*) a pour fonction d'organiser le transfert de données d'un point d'extrémité à un autre d'un réseau. Les points d'extrémités sont les équipements terminaux tels que les stations des clients et les serveurs. Ils génèrent et reçoivent les paquets IP. Ils sont les sources et les destinations du trafic de données.

Tout nœud connecté peut communiquer avec un autre nœud de l'Internet en utilisant le protocole IP sans qu'il ait besoin de changer le format de l'unité de transfert, à savoir le paquet IP. IP est en quelque sorte le langage commun de tous les nœuds de l'Internet. Les points importants du fonctionnement d'IP sont les suivants :

- *indépendance vis à vis des données : aucun nœud intermédiaire ne traite de l'information contenue dans le paquet ; seuls l'émetteur et le destinataire de l'information sont concernés ;*
- *transfert de bout en en bout : un paquet est transmis à un noeud qui le transmet à son tour au noeud suivant et ainsi de suite, jusqu'à l'hôte destination ; ainsi le protocole IP effectue un relayage de paquets par sauts successifs ;*
- *acheminement en mode message (datagramme) : cela signifie que chaque paquet dispose des éléments nécessaires et suffisants pour son acheminement à travers le réseau. Chaque paquet est traité indépendamment des autres paquets. Il comporte l'adresse IP source et l'adresse IP destination.*
- *Service en mode non connecté : Le transfert de données est fait sans échange préalable. A la différence du téléphone comme nous l'utilisons, IP n'a pas besoin d'établir une connexion avec le noeud de destination."*

La notion de paquet est essentielle dans le fonctionnement de l'Internet. Nous allons par la suite définir la notion de paquet.

Notion de paquet

Les données échangées dans l'internet sont découpées en blocs de taille limitée appelés *paquets*. Ce bloc de données est une séquence structurée d'octets qui est transférée sans modification de son contenu d'une source à la destination finale selon le principe du bout-en-bout.

Ainsi lorsqu'un fichier doit être transféré, celui-ci va être découpé en paquets et à destination, les paquets seront ré-assemblés pour reconstituer le fichier. La raison d'un transfert en mode paquet trouve sa motivation dans le partage efficace des ressources du réseau. Dans les systèmes de communications, la ressource principale est la capacité d'écoulement des liens qui est appelée abusivement la bande passante. La bande passante normalement s'exprime en Hertz. Par abus de langage, en numérique elle s'exprime par un débit et l'unité est le bit/s.

Quand il existe des communications entre différentes paires de noeuds (sources et destinations) reliés entre eux par une suite de liens. La problématique porte sur le partage des ressources entre ces noeuds qui communiquent en même temps. Au niveau le plus fin, la question revient à définir comment partager un lien entre des communications simultanées. Il s'agit de la fonction de multiplexage. La figure 1 illustre la notion de multiplexage. Sur cette figure, 3 communications sont multiplexées sur un même lien.

En numérique le partage s'effectue en fonction du temps. La bande passante est découpée dans le temps. A un instant donné, un utilisateur utilise toute la ressource disponible mais pour une période de temps limitée. L'unité de partage élémentaire est donc l'intervalle de temps. Sachant que le support a un débit exprimé bit/s, l'intervalle de temps est une unité de temps, le produit de l'intervalle de temps et du débit donne l'équivalent en terme de quantité de données. D'un point de vue pratique l'intervalle de temps est occupé par la transmission d'un bloc d'octets. Le temps de transmission d'un bloc d'octets est équivalent à l'intervalle de temps.

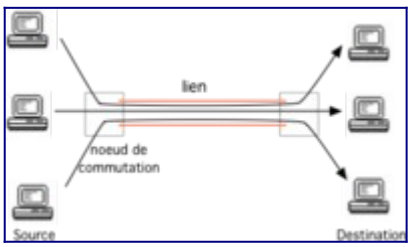


Figure 1: multiplexage de communications.

Si le principe du découpage de la ressource est posé, il reste la politique d'attribution des intervalles de temps à définir. Le transfert de données en informatique présente un caractère très sporadique. Des périodes d'activités sont suivies de périodes de silence. En pendant les périodes d'activités, il est souhaitable d'obtenir un débit important afin de limiter le temps de transfert du fichier. Dans ces conditions, l'attribution dynamique des intervalles de temps aux communications qui sont en activités représente une solution efficace. Les intervalles de temps

qui pourraient être attribuées au communications inactives sont récupérés par celles qui sont actives. En fait une communication en période de silence n'utilise aucune ressource. Comme nous l'avons préalablement indiqué, l'intervalle de temps se présente sous la forme d'un bloc d'octets de taille limitée. Ce bloc est constitué par les données produites par la source. Ce bloc de données est identifié comme appartenant à une communication à l'aide d'une en-tête. La combinaison de l'en-tête et du bloc de données forment le paquet. Le paquet est donc l'unité de partage des ressources dans un réseau. La figure 2 montre une représentation du partage du lien de la figure 1. La transmission des paquets des 3 communications simultanées s'entrelacent au niveau du lien. En l'absence de toute activité, le support reste libre de toute utilisation. Ce mécanisme de multiplexage des communications par entrelacement des paquets des différentes communications simultanées est un moyen simple et efficace de partage dynamique des ressources du lien. Ainsi, lorsqu'une communication est silencieuse ou inactive, l'ensemble des ressources du lien restent partagées par les autres communications du multiplex.

Figure 2: Représentation du partage d'un support.

La figure 3.a illustre la transmission de blocs de données dont la taille ne serait pas limitée. On observe que lorsque un gros bloc est en cours de transmission, il monopolise le support. En coupant ce bloc en bloc de taille plus limitée, la figure 3.b montre un entrelacement des blocs de données. Le partage de la ressource est dans cette situation est bien meilleur.

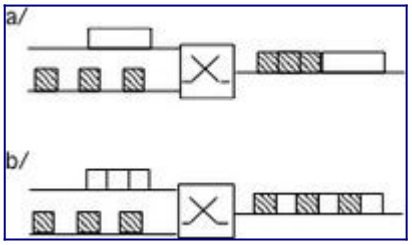


Figure 3: Représentation du partage d'un support.

Le paquet c'est aussi plus qu'une unité de partage des ressources dans un réseau. C'est aussi l'unité de transfert. Un noeud de commutation reçoit des paquets. Il doit les aiguiller vers un lien en sortie pour atteindre sa destination. La commutation dans un noeud fonctionne sur le principe dit du "store and forward". Le paquet doit être reçu dans son intégralité pour être commuté. C'est à dire transmis sur le lien suivant. Un réseau à commutation de paquets comme l'Internet signifie que l'unité élémentaire d'acheminement est le paquet.

Le paquet a une taille limitée comme vue précédemment. Il a aussi une taille variable toujours par soucis d'efficacité. Le transfert de données en informatique est fondamentalement asymétrique. Dans un sens, circulent les données dans le sens inverse circulent les acquittements pour signaler la bonne ou mauvaise réception des données. Dans cet échange, il y a une grosse quantité de données dans un sens et une quantité plus faible dans l'autre sens avec les acquittements. La taille variable des paquets permet de prendre en compte cette asymétrie dans les quantités échangés.

Acheminement de paquet

Les "matriochkas" des architectures en couches

Les modèles protocolaires d'architectures réseaux en couches, qu'ils soient ISO ou IETF, fonctionnent selon le principe d'encapsulation que l'on peut illustrer avec les célèbres poupées gigognes dites "poupées russes". Dans ce modèle, les unités de données protocolaires (PDU : Protocol Data Unit) échangées par le protocole d'une couche N sont déléguées au service de transfert de données de la couche sous-jacente. En émission, cela consiste pour l'entité de la couche N-1 à encapsuler la PDU-N, confiée pour un transfert, dans la charge utile (champ données) de sa propre unité de données. Ainsi sur un réseau, dont le niveau liaison de données (niveau 2) fonctionne en Ethernet, le datagrammes IP (PDU de niveau 3) est transféré dans le champ données de la trame Ethernet (charge utile de la PDU de niveau 2). En réception c'est le principe inverse (décapsulation) qui s'applique; après vérification protocolaire de la PDU-N-1, l'entité de niveau N-1 remet à l'entité supérieure de niveau N le contenu de sa charge utile à savoir la PDU-N. Ainsi l'entité Ethernet du récepteur, après contrôle d'intégrité de la trame reçue, remet le contenu du champ données à l'entité supérieure (la couche IP). Ce mécanisme d'émission par encapsulation / réception par décapsulation s'applique à toutes les couches du modèle. A l'instar des poupées russes, les PDU des différents niveaux "s'emboitent" en commençant par les couches hautes. Au niveau le plus bas de la transmission, la couche physique, l'unité de données de la couche liaison contient toutes les unités de données imbriquées. *Par exemple, pour les applications web modernes basées sur les API de type REST l'enchaînement des imbrications peut être le suivant : les données applicatives (REST) sont encapsulées dans l'unité de données du protocole applicatif web (HTTPS), qui est encapsulée dans le message de transport sécurisé (TLS sur TCP) encapsulé dans le datagramme IPv6 lui même véhiculé dans une trame Ethernet*.

Figure 5: Encapsulation des protocoles : les PDU gigognes.

Nous savons depuis la séquence de bienvenue que le protocole IP est dans une couche logicielle superposée au dessus des différents systèmes de transmission. Cette couche est présente dans tous les nœuds de l'Internet. *La conséquence de la superposition, et le propre d'une architecture en couches, consiste à ce que chaque unité de données d'une couche lorsqu'elle est passée à la couche inférieure est encapsulée dans l'unité de données de cette couche.* La figure 4 illustre le transfert d'un fichier d'un client vers le serveur. Ce fichier est par exemple la requête effectuée par l'application du client. Le fichier (l'unité de données applicative) est découpée en paquets IP pour les raisons que nous avons exposé précédemment. Un paquet IP doit atteindre le serveur qui est la destination finale du transfert de données. Sur le client ce paquet IP, pour être transféré, est passé à l'interface réseau 1 (un système de transmission comme Ethernet par exemple). Celle-ci va procéder à l'encapsulation

du paquet IP dans l'unité de données propre au protocole du lien réseau 1 (à savoir une trame Ethernet dans le cas de notre exemple). Le paquet est ainsi transmis du client au routeur qui en recevant la trame, sur son interface d'entrée, décapsule le paquet IP. Il route le paquet en sélectionnant l'interface de sortie vers la destination et lui délègue ce paquet. L'interface de sortie procède alors à l'encapsulation du paquet IP dans la trame propre au protocole de liaison du réseau 2 (par exemple le protocole Point to Point Protocol) pour le transmettre au serveur. L'interface de celui-ci décapsule le paquet de la trame (trame PPP dans notre exemple) qu'elle a reçue. La couche IP du serveur va réassembler toutes les données des paquets IP qu'il reçoit pour reconstituer le fichier original. Une fois ce fichier complet, l'application traite le fichier selon le contexte applicatif. Le paquet IP est transféré de la source à la destination par une succession de sauts (ou hop) d'un noeud à un autre. Pour ce transfert, le paquet subit une série d'encapsulations et de décapsulations.

Chaque nœud utilise l'adresse de destination contenu dans l'en-tête pour prendre une décision de routage à savoir a qui doit être transmis le paquet IP pour le faire converger vers sa destination. Lorsque le système de transmission utilise un support multipoint, il faut avoir recours à une adresse physique, la fameuse adresse MAC qui est mise dans les cartes réseaux. L'adresse MAC du prochain nœud doit être mise dans la trame. Le résultat du routage du paquet IP retourne l'adresse IP du prochain nœud, le destinataire de la trame qui va encapsuler le paquet IP. Pour que la trame soit effectivement remise à ce nœud, il faut que l'adresse de destination de la trame soit justement l'adresse MAC de ce nœud. Cette adresse MAC est obtenu par résolution de l'adresse IP retournée par le routage. La trame avec la bonne adresse MAC de destination sera alors reçu par le next HOP qui pourra alors décapsuler le paquet et procéder au routage pour trouver le nœud suivant qui doit recevoir le paquet. Et ainsi de saut en saut le paquet va arriver à sa destination finale en empruntant des liens de différentes natures.

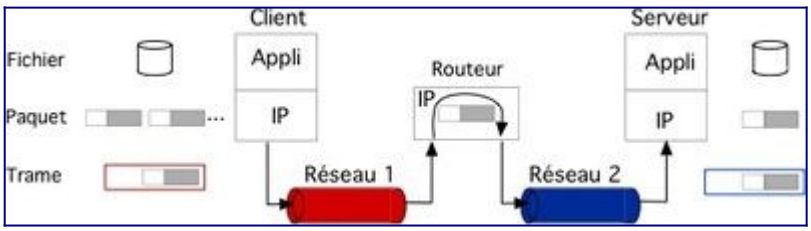


Figure 4: Transfert d'un paquet IP.

Les mécanismes d'encapsulation

La notion d'encapsulation de protocoles repose sur le principe de l'empilement des couches représentatives des traitements nécessaires à effectuer dans les différents composants d'un réseau. Ces traitements affecteront toutes les couches dans les nœuds d'extrémité, et certaines seulement pour les nœuds intermédiaires.

L'organisation internationale de normalisation ISO a défini le modèle OSI (*Open System Interconnection*) par une décomposition de l'architecture du réseau en 7 couches représentées du niveau Physique jusqu'au niveau Application comme représentée par la figure 1. Le modèle TCP/IP (ou DOD *Department of Defense*) a eu une approche plus pragmatique en

décomposant l'architecture de réseau en 4 couches. Les couches session, présentation et application sont agrégées en une seule couche applicative propre à chaque protocole.

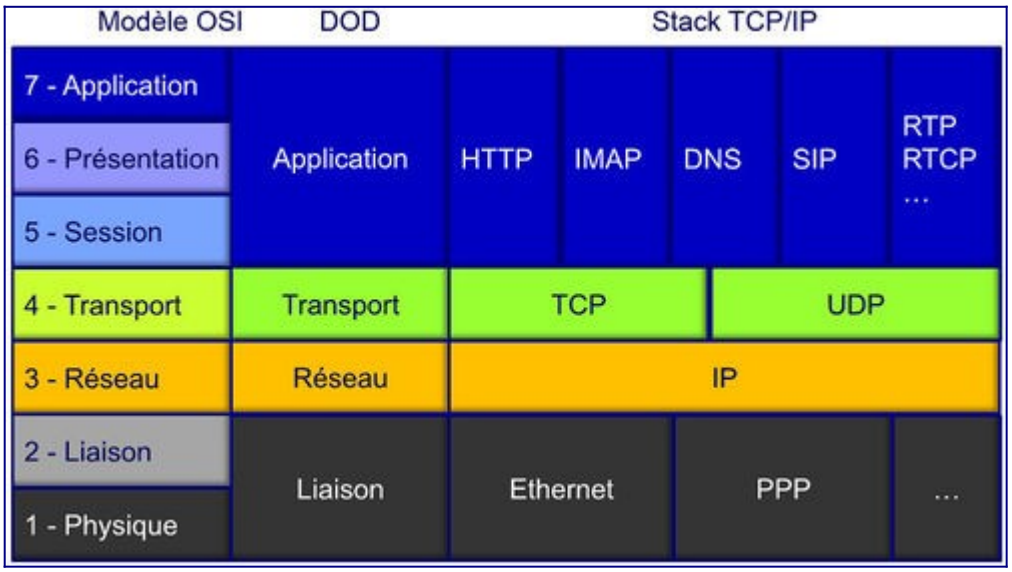


Figure 1 : Comparaison modèle OSI - modèle TCP/IP.

Pour simplifier l'organisation, pour un nœud d'extrémité du réseau, nous pouvons considérer que la carte réseau réalise les fonctions de niveau Physique et Liaison, que le traitement des couches Réseau et Transport est réalisé par les couches intermédiaires installées dans le système d'exploitation, et que le reste du système, avec les programmes applicatifs, gère les couches Session, Présentation et Application.

Cette activité vise à présenter l'intégration d'IPv6 dans la pile protocolaire. Aussi, nous aborderons l'encapsulation sous l'angle de deux points importants et impactés par le remplacement de IPv4 par IPv6 au sein de la couche de réseau à savoir: la longueur maximale des unités de transfert (*Maximum Transmission Unit* (MTU)) et la détection des erreurs d'intégrité (ou erreurs binaires). La question de la détection d'erreur binaire se pose dans la mesure où le calcul de redondance a disparu de l'en-tête IPv6.

Traitement des couches basses

La méthode de transfert d'un datagramme IPv6 entre deux nœuds directement reliés entre eux par un lien physique est le même que pour IPv4. Le datagramme est tout d'abord transmis vers l'interface d'émission qui l'encapsule dans une trame. La trame est une PDU (*Protocol Data Unit*) de niveau 2 dans le modèle de référence OSI. Cette trame est transmise sur le lien avec l'adresse physique du nœud de destination. Cette adresse sur un lien sera appelée adresse MAC dans la suite. Le nœud de destination reçoit la trame sur son interface, désencapsule le datagramme et le traite.

Les différences avec IPv4 sont les suivantes :

- sur le support Ethernet, le RFC 2464 précise que le code protocole encapsulé de la trame est différent (champ `EtherType` d'une trame Ethernet). Par exemple, pour les réseaux à diffusion, le code est `0x86DD` alors que, pour IPv4, le code est `0x0800`. À

l'origine, il était prévu de garder le même code et d'assurer l'aiguillage entre IPv4 et IPv6 en utilisant le champ `Version` du paquet. Mais certains nœuds ne vérifient pas la valeur de ce champ et auraient eu un comportement incontrôlable en essayant de traiter un paquet IPv6 comme un paquet IPv4 ;

- la résolution de l'adresse MAC de destination à partir de l'adresse IP de destination du paquet change. Par exemple, sur un réseau à diffusion, cette résolution est faite en IPv4 par le protocole ARP alors qu'en IPv6 on utilise le protocole de découverte de voisins comme nous le verrons dans la séquence 3 ;
- la taille minimale d'une trame est passée à 1 280 octets ; ceci peut forcer certains protocoles à utiliser plusieurs trames par datagramme IPv6 ;
- enfin, certains protocoles ont des parties propres à IPv4. Ces parties doivent être modifiés. C'est le cas des protocoles de contrôle et de compression de PPP.

Couche physique

Commençons par la couche *physique*, qui est à la base de l'édifice de ce modèle. Les spécifications de cette couche dépendent du support lui-même. Nous devons gérer la transmission des informations binaires issues du codage des trames et des paquets sur un support cuivre, optique ou sans fil ; d'où la nécessité d'adaptation aux caractéristiques des composants (câbles, connecteurs ou antennes) et d'une méthode appropriée de codage des données (représentation physique des données binaires).

La représentation binaire utilisée dépend du support. Sur du cuivre, on utilise des variations d'impulsions électriques ; sur une fibre optique, ce sont des variations lumineuses sur une ou plusieurs longueurs d'ondes ; en transmission sans fil, ce sont généralement des signaux radio, laser ou infrarouge. La couche *physique* coordonne le débit et la synchronisation de l'émetteur et du récepteur réseau, tout en tentant de garantir la transparence et l'intégrité d'un flux d'information binaire, sans notion d'interprétation du contenu.

Hélas, cette couche est fréquemment soumise à différentes perturbations issues de l'environnement extérieur au canal de transmission : rayonnements électromagnétiques, micro-coupures ou altérations des signaux par différents facteurs. Les coupleurs intégrés dans les cartes réseau réalisent les fonctions nécessaires et utiles au niveau *physique*, et on dispose d'un indicateur de qualité de la transmission avec le calcul du CRC (*Cyclic Redondancy Check*).

Couche liaison

Le rôle de la couche *liaison* est, entre autres, de contrôler l'accès à la couche physique, en réalisant le multiplexage temporel sur le support de transmission, et de transformer la couche *physique* en une liaison à priori exempte d'erreurs de transmission pour la couche *réseau*. La couche *liaison* doit être capable d'écarter le trafic nécessaire à la synchronisation sur le lien physique, et de reconnaître les débuts et fins de trames. Cette couche écarte les trames en cas de réception erronée, comme en cas de non respect du format, ou bien en cas de problème sur la ligne de transmission. La vérification du champ CRC aide à faire ce tri. Cette couche intègre également une fonction de contrôle de flux pour éviter l'engorgement d'un récepteur incapable de suivre un rythme imposé.

L'unité de données de protocole de la couche *liaison de données* est la trame (*Link Protocol Data Unit* (LPDU)), qui est composée de plusieurs champs permettant d'identifier l'origine des échanges, le rôle de la trame, le contenu de l'enveloppe et, en fin de trame, le champ CRC, le tout étant encadré par une séquence particulière de codage de début et de fin de trame.

Si nous prenons l'exemple de la trame Ethernet originale (cf. Figure 2), un délai inter-trame minimum de 96 intervalles de temps est spécifié comme silence sur un support cuivre alors que, sur un support optique, tout silence est comblé par la transmission d'un ou plusieurs symboles particuliers « idle » ; une parfaite synchronisation est alors maintenue entre les extrémités du lien optique.

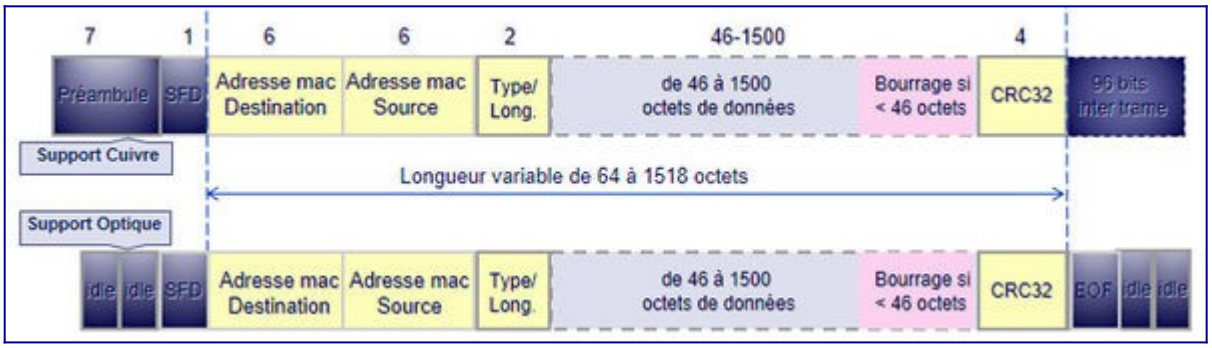


Figure 2 : Format de la trame Ethernet.

Une fois que l'arrivée d'une trame Ethernet est détectée par le coupleur, les premiers champs immédiatement accessibles correspondent aux adresses MAC `Destination` et `Source` puis, soit au champ `Longueur` dans le cas d'une encapsulation au standard 802.3, ou bien au champ `EtherType` dans le cas d'une encapsulation avec le standard Ethernet original. Ensuite, l'enveloppe de la trame transporte les données, qui correspondent aux paquets IPv6 dès lors que le champ `EtherType` vaut `0x86dd`. Vient ensuite le champ CRC codé sur 32 bits. Dans le cas d'une encapsulation au standard 802.1Q, d'autres champs permettent la reconnaissance du numéro de VLAN (*Virtual Local Access Network*) et du niveau de priorité défini dans le standard 802.1p.

Un des éléments particulièrement importants est la capacité de transport de la trame. Dans l'exemple ci-dessus, nous voyons que la trame Ethernet traditionnelle dispose d'une enveloppe qui autorise le transport de 1500 octets maximum : MTU = 1500.

D'autres formats de trames permettent des échanges plus ou moins importants. Citons quelques MTU :

- PPPoE = 1492
- PPPoA = 1468
- MPLS = 1500 à 65535
- 802.15.4 (LowPAN) = 81
- Ethernet Jumboframe = 9000

Rappelons que la spécification du protocole IPv6 impose une taille minimale du paquet de 1280 octets. Pour les couches liaison imposant des tailles inférieures, il est donc obligatoire de mettre en place une *couche d'adaptation* comme 6LowPAN (RFC 4944) pour les réseaux 802.15.4.

Cette couche située entre la couche *liaison* et la couche *réseau* IPv6 prend en charge le découpage des paquets IPv6 en fragments pouvant être transportés dans les trames et leur ré-assemblage au niveau du premier routeur de sortie.

Couches intermédiaires

Couche réseau

Étant donné que la taille minimum de l'en-tête IPv6 est de 40 octets, le MTU résiduel d'une trame Ethernet classique est de 1500 - 40 = 1460 octets ; sachant que ces 1460 octets de données seront probablement encore amputées d'en-têtes de niveau transport, par exemple 20 octets minimum pour TCP et 8 octets pour UDP.

Parmi les différences existant entre les datagrammes IPv4 et IPv6, il y a la disparition de la somme de contrôle d'erreur (*checksum*) dans les en-têtes IPv6. Cette somme de contrôle était utilisée pour vérifier l'absence d'erreur binaire de l'en-tête du paquet traité. Une erreur binaire est le changement de valeur d'un bit effectué lors de la transmission. En IPv4, il est nécessaire de la vérifier et de l'ajuster lors de chaque retransmission par un routeur, ce qui entraîne une augmentation du temps de traitement du paquet. Cette somme ne vérifie que l'en-tête IPv4, pas le reste du paquet. Aujourd'hui, les supports physiques sont de meilleure qualité et savent détecter les erreurs (par exemple, Ethernet a toujours calculé sa propre somme de contrôle ; PPP, qui a presque partout remplacé SLIP, possède un CRC). L'intérêt de la somme de contrôle au niveau réseau a diminué et ce champ a été supprimé de l'en-tête IPv6.

Le checksum sur l'en-tête IPv6 n'existant plus, il faut néanmoins se prémunir des erreurs de transmission. En particulier, une erreur sur l'adresse de destination va faire router un paquet dans une mauvaise direction. Le destinataire doit donc vérifier que les informations d'en-tête IP sont correctes avant d'accepter ces paquets. Dans les mises en œuvre des piles de protocoles Internet, les entités de niveau transport remplissent certains champs du niveau réseau. Il a donc été décidé que tous les protocoles au-dessus d'IPv6 devaient utiliser une somme de contrôle intégrant à la fois les données et les informations de l'en-tête IPv6. La notion de *pseudo-en-tête* dérive de cette conception. Pour un protocole comme TCP, qui possède une somme de contrôle, cela signifie qu'il faut modifier le calcul de cette somme. Pour un protocole comme UDP, qui possède une somme de contrôle facultative, cela signifie qu'il faut modifier le calcul de cette somme et le rendre obligatoire.

IPv6 a unifié la méthode de calcul des différentes sommes de contrôle. Le RFC 8200 définit, dans sa section 8.1, un *pseudo-en-tête* (cf. Figure 3), résultat de la concaténation d'une partie de l'en-tête IPv6 et du PDU du protocole concerné. L'algorithme de calcul du checksum est celui utilisé en IPv4. Il est très simple à mettre en œuvre et ne demande pas d'opérations complexes. Il s'agit de faire la somme en complément à 1 des mots de 16 bits du *pseudo-en-tête*, de l'en-tête du protocole de transport, et des données, puis de prendre le complément à 1 du résultat.

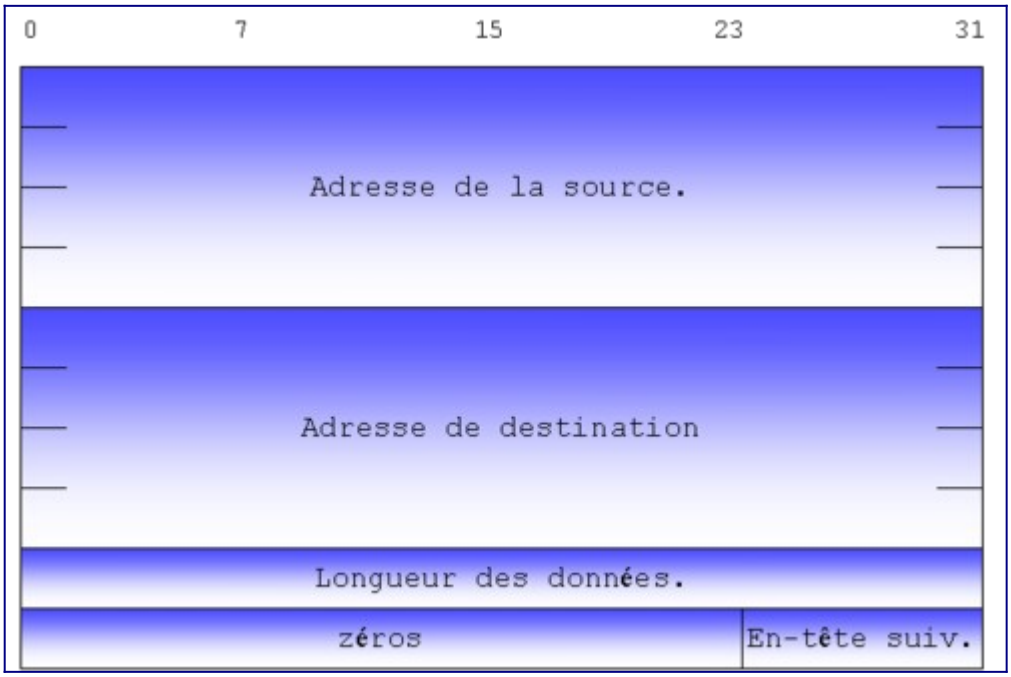


Figure 3 : Champs du `pseudo-en-tête`.

Il faut noter que les informations contenues dans le *pseudo-en-tête* ne seront pas émises telles quelles sur le réseau. Le champ `en-tête suivant` du *pseudo-en-tête* ne reflète pas celui qui sera émis dans les paquets puisque les extensions ne sont pas prises en compte dans le calcul du checksum. Ainsi, si l'extension de routage est mise en œuvre, l'adresse de la destination est celle du dernier nœud. De même, le champ `longueur` est codé sur 32 bits pour contenir la valeur de l'option *jumbogramme* si celle-ci est présente.

Couches Transport

UDP et TCP

Les modifications apportées aux protocoles de niveau transport sont minimes. L'un des pré-requis à la mise en œuvre d'IPv6 était de laisser en l'état aussi bien TCP (*Transmission Control Protocol*) qu'UDP (*User Datagram Protocol*). Ces protocoles de transport sont utilisés par la très grande majorité des applications réseau et l'absence de modification facilitera grandement le passage de IPv4 à IPv6.

La principale modification de ces protocoles concerne le calcul de la somme de contrôle (*checksum*) pour vérifier l'intégrité des données. Comme la couche *réseau* ne possède pas de champ de contrôle, c'est à la couche de transport que revient cette tâche. Le calcul de la somme de contrôle au niveau du transport a donc été adapté pour inclure des données de l'en-tête. De plus, la présence d'une somme de contrôle devient obligatoire pour tout protocole de niveau supérieur transporté au-dessus d'IPv6.

Nota : *les RFC 6935 et RFC 6936 ont introduit une dispense à cette obligation, dans les cas des techniques de tunnel au dessus d'UDP. La nouvelle philosophie est résumée ainsi par S. Bortzmeyer : "par défaut, la somme de contrôle doit être calculée et mise dans le paquet. Mais on peut s'en dispenser dans le cas de tunnels (et uniquement celui-ci), sur le paquet extérieur.*

Le protocole à l'intérieur du paquet (qui n'est pas forcément de l'UDP et même pas forcément de l'IPv6) doit rester protégé par sa propre somme de contrôle. Cela ne doit s'appliquer que pour une liste explicite de ports (ceux utilisés par le tunnel) et pas pour tout le trafic UDP du nœud. Et le tout doit se faire en lisant le RFC 6936 qui précise les conditions d'application de cette nouvelle règle."

Un autre changement au niveau des protocoles de niveau 4 concerne la prise en compte de l'option *jumbogramme* de l'extension *proche-en-proche*. Le RFC 2675 définit le comportement d'UDP et de TCP quand les jumbogrammes sont utilisés. En effet, les en-têtes de ces messages contiennent eux aussi un champ `longueur` codé sur 16 bits et, par conséquent, insuffisant pour coder la longueur du jumbogramme :

- pour le protocole UDP, si la longueur des données excède 65 535 octets, le champ `longueur` est mis à 0. Le récepteur détermine la longueur des données par la connaissance de la taille dans l'option *jumbogramme* ;
- le protocole TCP pose plus de problèmes. En effet, bien que les messages TCP ne contiennent pas de champ `longueur`, plusieurs compteurs sont codés sur 16 bits ;
- le champ `longueur` de la fenêtre de réception ne pose pas de problème depuis que le RFC 1323 a défini l'option *TCP window scale* qui donne le facteur multiplicatif qui doit être appliqué à ce champ ;
- à l'ouverture de connexion, la taille maximale des segments (MSS) est négociée. Le RFC 2675 précise que, si cette taille doit être supérieure à 65 535, la valeur 65 535 est envoyée et le récepteur prend en compte la longueur déterminée par l'algorithme de découverte du MTU.

Pour l'envoi de données urgentes avec TCP, on utilise un bit spécifique de l'en-tête TCP (bit URG) ainsi que le champ `pointeur urgent`. Ce dernier sert à référencer la fin des données à traiter de manière particulière. Trois cas peuvent se présenter :

- le premier, qui est identique à IPv4, est celui où le pointeur indique une position de moins de 65 535 ;
- le second se produit lorsque le déplacement est supérieur à 65 535 et supérieur ou égal à la taille des données TCP envoyées. Cette fois-ci, on place la valeur 65 535 dans le champ `pointeur urgent` et on continue le traitement normal des paquets TCP ;
- le dernier cas intervient quand le pointeur indique un déplacement de plus de 65 535 qui est inférieur à la taille des données TCP. Un premier paquet est alors envoyé, dans lequel on met la valeur 65 535 dans le champ `pointeur urgent`. L'important est de choisir une taille de paquet de manière à ce que le déplacement dans le second paquet, pour indiquer la fin des données urgentes, soit inférieur à 65 535.

Il existe d'autres propositions pour faire évoluer TCP. Il faut remarquer que le travail n'est pas de même ampleur que pour IP. En effet, TCP est un protocole de bout en bout. La transition vers une nouvelle génération du protocole peut se faire par négociation entre les deux extrémités. Pour IP, tous les routeurs intermédiaires doivent prendre en compte les modifications.

UDP-lite

UDP-lite permet de remonter aux couches supérieures des données erronées pendant leur transport. Si, dans un environnement informatique, une erreur peut avoir des conséquences relativement graves quant à l'intégrité des données, il est normal de rejeter ces paquets. Dans le domaine du multimédia, cette exigence peut être relâchée. En effet, la plupart des décodeurs de flux audio ou vidéo sont capables de supporter un certain nombre d'erreurs binaires dans un flux de données. Pour améliorer la qualité perçue par l'utilisateur, il est donc préférable d'accepter des paquets erronés plutôt que de rejeter un bloc complet d'informations qui se traduirait par une coupure perceptible du flux.

En IPv4, l'utilisation du *checksum UDP* étant optionnelle (la valeur 0 indique que le checksum n'est pas calculé), UDP peut être utilisé pour transporter des flux multimédias. Avec IPv6, l'utilisation du checksum a été rendue obligatoire puisque le niveau 3 n'en possède pas. Pour éviter qu'un paquet comportant des erreurs ne puisse pas être remonté aux couches supérieures, le protocole UDP-lite a été défini par le RFC 3828. Les modifications sont minimes par rapport à UDP. Le format de la trame reste le même ; seule la sémantique du champ `longueur` est changée. Avec UDP, ce champ est inutile puisqu'il est facilement déduit du champ `longueur` de l'en-tête IP. UDP-lite le transforme en champ `couverture du checksum`. Si la longueur est 0, UDP-lite considère que le checksum couvre tout le paquet. La valeur 8 indique que seul l'en-tête UDP est protégé par le checksum (ainsi qu'une partie de l'en-tête IP grâce au pseudo-header). Les valeurs comprises entre 1 et 7 sont interdites car le checksum UDP-lite doit toujours couvrir l'en-tête. Une valeur supérieure à 8 indique qu'une partie des données sont protégées. Si la couverture est égale à la longueur du message, on se retrouve dans un cas compatible avec UDP.

SCTP

Le protocole SCTP (*Stream Control Transmission Protocol*) RFC 4960 est fortement lié au protocole IPv6. SCTP est un protocole de niveau 4 initialement conçu pour transporter des informations de signalisation[1]. La fiabilité est donc un prérequis important et la gestion de la multi-domiciliation est prise en compte. L'idée est de permettre aux deux nœuds d'extrémité d'échanger, à l'initialisation de la connexion (appelée, dans le standard, *association*), l'ensemble de leurs adresses IPv4 et IPv6. Chaque nœud choisit une adresse privilégiée pour émettre les données vers l'autre extrémité et surveille périodiquement l'accessibilité des autres adresses. Si le nœud n'est plus accessible par l'adresse principale, une adresse secondaire est choisie.

SCTP permet une transition douce d'IPv4 vers IPv6 puisque l'application n'a plus à se préoccuper de la gestion des adresses. Si les deux entités possèdent une adresse IPv6, celle-ci sera privilégiée. De plus, SCTP peut servir de brique de base à la gestion de la multi-domiciliation IPv6. En effet, avec TCP, une connexion est identifiée par ses adresses. Si une adresse n'est plus accessible, le fait d'en changer peut conduire à la coupure de la connexion. Il faut avoir recours à des subterfuges, comme la mobilité IP, pour maintenir la connexion. SCTP brise ce lien entre la localisation du nœud et l'identification des associations.

Conclusion

Cette activité a fait un rappel des différentes couches de la pile réseau d'un système. La couche réseau, ou niveau 3, est le langage commun de tous les nœuds connectés à l'Internet. L'introduction d'IPv6 a donc un impact sur les différentes couches, notamment la couche sous-jacente, couche liaison, et les couches supérieures, notamment la couche transport.

Même si ces couches sont censées être indépendantes dans le modèle théorique OSI, force est de remarquer que dans la pratique, elles sont interdépendantes. Pour preuve le champ de contrôle d'erreur n'a pas été retenu au niveau de la couche IP car il y a déjà un contrôle fait au niveau *liaison*. Et un autre contrôle d'erreur a été placé dans les couches supérieures afin de vérifier l'intégrité des données transportées. Cette vérification se fait uniquement par le destinataire. Le contrôle d'erreur est un exemple qui illustre l'interdépendance des couches.

Nous venons d'apprendre comment les communications de données sont mises en oeuvre dans l'Internet.

Introduction de la séquence 2

Dans la première séquence, les différents types d'adresses IPv6 ont été présentés. Cette deuxième séquence va détailler les mécanismes protocolaires. Le fil rouge est la performance du traitement des datagrammes dans tous les équipements et en particulier les équipements intermédiaires tels que les routeurs ou les pare-feux.

Dans cette deuxième séquence du MOOC Objectif IPv6, vous aborderez les différents aspects de ce protocole au travers de cinq activités thématiques :

- A21 : tout d'abord, vous allez découvrir le format et les fonctions de l'en-tête des paquets IPv6 ;
- A22 : puis, vous aborderez les principes de l'acheminement et du routage ;
- A23 : ensuite, les points essentiels de contrôle et diagnostic de l'acheminement par ICMPv6 ;
- A24 : et enfin, vous décomposerez les extensions de l'en-tête IP par le biais de l'exemple de gestion de taille des paquets ;
- Activité pratique : en complément vous pourrez observer l'acheminement de paquets IPv6. Au sein de la même machine virtuelle utilisée lors de la première activité pratique, vous pourrez expérimenter la communication IPv6 sans déstabiliser la configuration de votre ordinateur.

Références bibliographiques

1. ↥ Fu, S. et Atiquzzaman, M. (2004). IEEE Communications Magazine, Vol. 42, No. 4, April. SCTP: State of the Art in Research, Products, and Technical Challenges.

Pour aller plus loin

RFC et leur analyse par S. Bortzmeyer :

- RFC 1323 : TCP Extensions for High Performance Analyse
- RFC 2464 : Transmission of IPv6 Packets over Ethernet Networks
- RFC 2675 : IPv6 Jumbograms
- RFC 3828 : The Lightweight User Datagram Protocol (UDP-Lite) Analyse
- RFC 4944 : Transmission of IPv6 Packets over IEEE 802.15.4 Networks
- RFC 4960 Stream Control Transmission Protocol Analyse
- RFC 5692 : Transmission of IP over Ethernet over IEEE 802.16 networks Analyse
- RFC 6691 : TCP Options and MSS Analyse
- RFC 6935: IPv6 and UDP Checksums for Tunneled Packets Analyse
- RFC 6936: Applicability Statement for the Use of IPv6 UDP Datagrams with Zero ChecksumsAnalyse
- RFC 6951 : UDP Encapsulation of SCTP Packets for End-Host to End-Host Communication Analyse
- RFC 8200 : Internet Protocol, Version 6 (IPv6) Specification Analyse

Activité 21: L’en-tête IPv6

Introduction

Comme rappelé en introduction de la séquence, la fonction principale du protocole IP est d'acheminer un paquet d'un point à un autre de l'Internet. Un paquet se définit comme une unité de transfert dans un réseau. Il est composé d'un bloc de données de taille variable mais limitée et identifié par un en-tête. Le mode d'acheminement par datagramme, utilisé dans l'Internet, impose que chaque paquet soit traité indépendamment, sans avoir besoin d'informations issues d'autres paquets déjà transmis. L'en-tête IP doit donc comporter toutes les informations pour réaliser cet acheminement. C'est la raison pour laquelle chaque paquet doit contenir l'adresse globale du nœud source ainsi que celle du nœud de destination du paquet. Pour marquer cette spécificité, ces paquets auto-descriptifs sont appelés des datagrammes.

L'adresse du destinataire est fondamentale pour le routage du paquet effectué par les nœuds intermédiaires. C'est en effet à partir de cette adresse que le nœud décide vers quelle interface le paquet doit sortir. La lecture de l'adresse du destinataire dans l'en-tête IP est donc une étape cruciale pour la performance globale de l'acheminement du paquet. Afin d'accélérer cette étape, deux principes ont été appliqués dans la spécification de l'en-tête IP :

- une adresse IP est de taille fixe, permettant ainsi d'être récupérée dans l'en-tête directement en lisant un nombre de bits prédéterminé, sans avoir à lire ni interpréter au préalable un champ de longueur d'adresse ;
- l'adresse IP destination est à un emplacement fixe dans l'en-tête, emplacement aligné en mémoire, facilitant ainsi son extraction de l'en-tête par un simple décalage en mémoire, qui est une opération simple au niveau matériel.

Format de l'en-tête du datagramme IPv6

Le format d'en-tête du datagramme IPv6 est spécifié par le RFC 8200 (page 5). Cet en-tête, avec les champs le composant, est représenté par la figure 1. L'en-tête IPv6 est de taille fixe et se compose de 5 mots de 64 bits (contre 5 mots de 32 bits pour IPv4). La taille de l'en-tête IPv6 est donc de 40 octets.

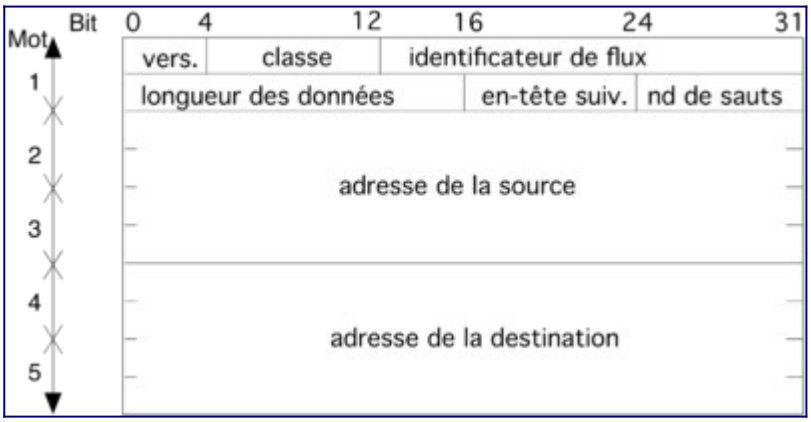


Figure 1 : Format de l'en-tête d'un paquet IPv6.

Signification des champs de l'en-tête

Version

Le champ `Version` est au même emplacement et de même longueur, quelle que soit la version du protocole IP (IPv4 ou IPv6). Cette similarité permet de vérifier que le paquet reçu est au format attendu. Dans le cas d'IPv6, sa valeur est de 6 ; elle est de 4 pour IPv4. Le numéro de version 5 avait déjà été attribué au protocole Stream qui finalement n'a pas eu le succès attendu [RFC 1190, RFC 1819].

Classe de trafic (*Traffic Class*)

Dans la version du document de spécification du protocole IPv6 [RFC 8200], le champ `Classe de trafic`, sur 8 bits reprend la spécification pour le codage de la différenciation de services RFC 2474.

Le champ `Classe de trafic` est défini de façon similaire au champ `Differentiated Services` (DS, ou *DiffServ*) de l'en-tête IPv4, qui a lui-même pris la place de l'octet `Type of Service` de la spécification initiale d'IPv4. Ce champ est découpé en deux parties (cf. figure 2). Le sous-champ DSCP (*Differentiated Services Code Point*) contient les valeurs identifiant les différents traitements demandés aux routeurs. La valeur par défaut 000000 correspond au service classique dit *au mieux* (ou *best effort*). Les deux derniers bits du champ, notés ECN (*Explicit congestion Notification*), servent aux routeurs à rapporter un risque de congestion [RFC 8311].

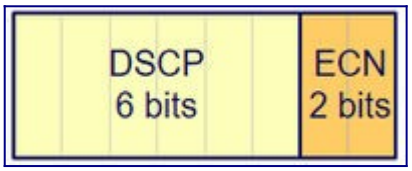


Figure 2 : Format de l'octet `classe de trafic`.

Pour le lecteur qui veut en savoir plus, l'annexe 1 décrit l'utilisation du champ classe de trafic dans l'objectif de fournir des services de transfert à qualité différente.

Identificateur de flux (*Flow Label*)

Ce champ, contient un numéro unique, choisi par la source, pour identifier le flux de données d'une application comme par exemple l'ensemble des paquets d'une application de voix sur IP. L'unicité de l'identificateur de flux se comprend dans le contexte de l'adresse source. L'utilisation de ce champ est spécifiée par le RFC 6437.

Ce champ vise à faciliter la classification des flux par les routeurs afin de différencier le traitement des paquets et donc notamment la mise en œuvre des fonctions de qualité de service. Les datagrammes d'un même flux ayant un même identifiant, le routeur peut alors les reconnaitre et leur appliquer un traitement particulier comme le choix d'une route ou une priorité à l'accès à la transmission.

Habituellement, les routeurs se basent sur les valeurs de cinq champs pour identifier le même flux de paquets : adresses de la source et de la destination, numéros de port de la source et de la destination, et protocole. Cette identification implique l'analyse d'en-têtes de niveau 3 pour les adresses et de niveau 4 pour les autres informations.

Avec IPv6, l'identification du flux n'implique plus que le niveau 3. Ainsi, l'identification est faite indépendamment du protocole de niveau 4. Le champ `Identificateur de flux` peut être rempli avec une valeur aléatoire mais qui doit être unique. La source gardera cette valeur pour tous les paquets qu'elle émettra pour cette application et cette destination. Le traitement est simplifié puisque le routeur n'a plus à consulter cinq champs pour déterminer l'appartenance d'un paquet à un flux, mais un seul. De plus, la confidentialité peut être préservée, les informations concernant les numéros de port peuvent être masquées aux routeurs intermédiaires.

Passage à l'échelle

Le passage à l'échelle désigne la capacité d'un produit ou d'un mécanisme à s'adapter à un changement d'ordre de grandeur de la demande (montée en charge), en particulier sa capacité à maintenir ses fonctionnalités et ses performances en cas de forte demande[1]. Il faut donc comprendre qu'une difficulté à passer à l'échelle signifie que la propriété d'extensibilité n'est pas acquise.

À la date de rédaction de ce document, l'utilisation de l'identificateur de flux reste encore floue. Les micro-flux, c'est-à-dire les flux applicatifs, ne sont pas analysés dans le cœur du réseau pour des raisons de passage à l'échelle. De plus, l'idée d'utiliser une étiquette pour acheminer des paquets est déjà utilisée par la technologie MPLS[2] de l'Internet, ce qui retire tout sens à utiliser ce champ du paquet IPv6 pour cette fonction. Pour l'instant, ce champ peut être vu comme réservé. Son utilisation devra être détaillée dans le futur.

Longueur de la charge du paquet (*Payload Length*)

Ce champ indique la quantité d'octets qui suit l'en-tête du datagramme (en-tête exclu donc), c'est-à-dire la charge du paquet, autrement dit les données. Les éventuelles extensions à l'en-tête IPv6 sont incluses dans ces données. Ce champ, sur 16 bits, peut donc indiquer la taille des données utiles, allant jusqu'à 64 kibioctets (64 * 1024 octets). Pour des paquets dont la taille serait supérieure, ce champ vaut 0 et l'émetteur ajoute une extension d'en-tête de "proche en proche" avec l'option *jumbogramme* définie par le RFC 2675. Avec l'option jumbogramme, la quantité de données transportée est codée sur 32 bits. La taille maximale des données utiles du paquet monte alors jusqu'à 4 gibioctets (4* 2^30 octets). Nous reviendrons sur le jumbogramme dans l'activité "Taille des paquets IPv6".

En-tête suivant (*Next Header*)

Ce champ identifie le prochain en-tête de protocole se trouvant à la suite de l'en-tête IPv6. Il peut s'agir d'un protocole de niveau supérieur (ICMP, UDP, TCP...) ou de la désignation d'une extension (cf. tableau *Valeurs du champ en-tête suivant pour IPv6*).

Valeurs du champ en-tête suivant pour IPv6

valeur	Hexa	Protocole ou *Extension*
0	0x00	*Proche-en-proche*
4	0x04	IPv4
6	0x06	TCP
17	0x11	UDP
41	0x29	IPv6
43	0x2b	*Routage*
44	0x2c	*Fragmentation*
50	0x32	*Confidentialité*
51	0x33	*Authentification*
58	0x3a	ICMPv6
59	0x3b	Fin des en-têtes
60	0x3c	*Destination*
132	0x84	SCTP
135	0x87	*Mobilité*
136	0x88	UDP-lite
140	0x8c	*Shim6*

Nombre maximal de sauts (*Hop Limit*)

Ce champ représente le nombre maximal de routeurs que le paquet peut traverser. Il est initialisé par le nœud source du paquet et, ensuite, décrémenté à chaque routeur traversé. Si la valeur, après décrémentation, atteint 0, le datagramme est supprimé et un message d'erreur ICMPv6 est émis pour la source de ce paquet. Ce champ vise à limiter le nombre de relayages (ou traversées de routeurs) que peut subir un paquet suite à des problèmes dans le réseau comme des boucles de routage. Ainsi, un paquet ne peut pas circuler indéfiniment dans l'Internet. Ce champ est à la base de l'outil `traceroute` pour identifier la suite des routeurs traversés jusqu'à une destination.

La valeur initiale de ce champ, à l'émission du paquet, devrait être donnée dans un document annexe de l'IANA (http://www.iana.org/) ; ce qui permettrait de la modifier en fonction de l'évolution de la topologie du réseau. La valeur n'est pas encore officiellement attribuée mais certaines implantations prennent actuellement la valeur conseillée pour IPv4 : 64.

La valeur par défaut peut être dynamiquement attribuée aux hôtes émetteurs par les annonces des routeurs en configuration automatique. Une modification de ce paramètre sera donc relativement simple quand la limite actuelle sera atteinte. On peut noter une limitation puisque

ce champ, codé sur 8 bits, n'autorise la traversée que de 255 routeurs. En réalité, dans l'Internet actuel, le nombre maximal de routeurs traversés est d'une quarantaine, ce qui laisse une bonne marge pour l'évolution du réseau.

Adresses source et destination

Ces adresses sont renseignées par le nœud source du paquet pour désigner l'émetteur et le destinataire du paquet. Pour renseigner l'adresse source, l'émetteur choisit de préférence une adresse IPv6 parmi celles configurées sur l'interface utilisée pour transmettre le paquet sur le réseau. Cette adresse est choisie pour avoir une portée compatible avec l'adresse de destination et, ainsi, permettre au destinataire de répondre. Par exemple, il serait problématique d'essayer de joindre un nœud de l'Internet en donnant comme adresse source une adresse *lien-local*. Le choix de l'adresse source est spécifié dans le RFC 6724.

L'adresse destination, elle aussi, peut être choisie dans la liste des adresses valables pour le destinataire ; liste pouvant provenir de la résolution d'un nom en adresses par le DNS. L'adresse choisie doit être compatible avec la portée des adresses disponibles au niveau de l'émetteur. Par exemple, si l'émetteur ne possède que des adresses de type ULA, et que le destinataire est connu avec des adresses globales et ULA, ces dernières adresses seront à préférer. Le RFC 6724 précise l'algorithme du choix de l'adresse destination.

Extensions à l'en-tête IPv6

L'ajout de nouvelles fonctionnalités est problématique quand l'en-tête est de taille fixe comme c'est le cas d'IPv6. La solution proposée consiste à ajouter des en-têtes spécifiques pour chaque fonctionnalité. Les extensions sont ajoutées par le nœud source du paquet pour demander un traitement spécifique à réaliser, soit par les routeurs intermédiaires, soit par le destinataire du paquet. Par exemple, si un paquet doit être fragmenté, une extension de fragmentation sera ajoutée par le nœud source afin que le destinataire puisse rassembler les morceaux correctement.

Il existe plusieurs types d'extensions selon le traitement demandé. Elles se placent après l'en-tête IPv6 et avant la charge utile du paquet (voir la figure 3). La présence d'une extension est signalée par le champ `En-tête suivant` (*Next Header*) de l'en-tête IPv6 qui possède alors la valeur correspondant à cette extension. Ainsi, elle est traitée par les routeurs intermédiaires comme un protocole de niveau supérieur à IP. L'utilisation des différents types d'extensions d'en-tête IPv6 sera abordé dans une activité ultérieure de ce cours.

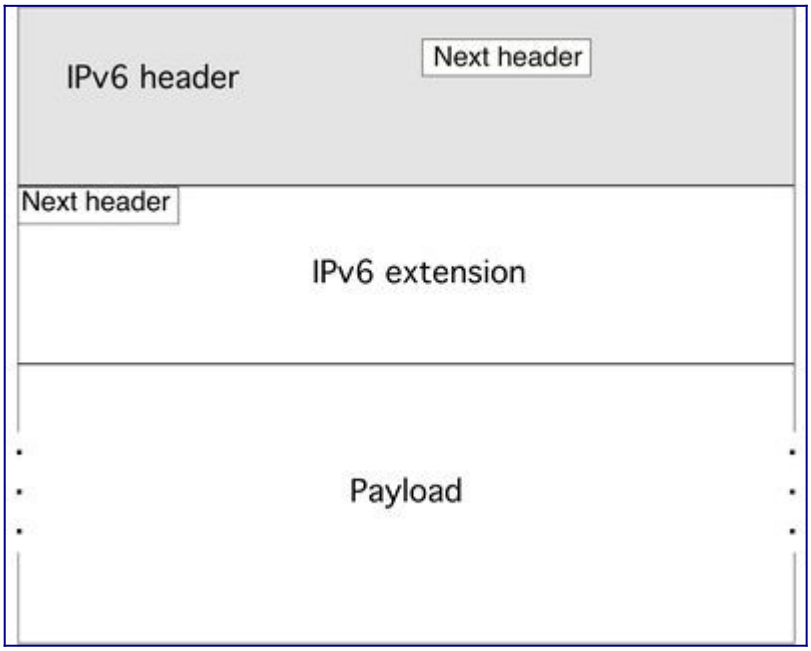


Figure 3: Format d'un paquet IPv6.

Evolution de l'en-tête depuis IPv4

Hormis la modification de la taille des adresses, ce qui conduit à une taille d'en-tête de 40 octets (le double d'un en-tête IPv4 sans options), le protocole IP a subi un toilettage[3], reprenant l'expérience acquise au fil d'une trentaine d'années avec IPv4, défini en 1981 (!) par le RFC 791. Le format des en-têtes IPv6 est ainsi simplifié et permet aux routeurs de meilleures performances dans leurs traitements. L'idée est de retirer du cœur de réseau les traitements compliqués. Les routeurs ne font que retransmettre les paquets vers la destination, les autres traitements sont faits par l'émetteur et le destinataire du paquet.

L'en-tête ne contient plus de contrôle d'erreur (*checksum*), qui devait être ajusté, pour IPv4, par chaque routeur en raison, entre autres, de la décrémentation du champ `durée de vie`. Par contre, pour éviter qu'un paquet, dont le contenu est erroné -- en particulier sur l'adresse de destination --, ne se glisse dans une autre communication, tous les protocoles de niveau supérieur doivent mettre en œuvre un mécanisme de contrôle d'erreur de bout en bout, incluant un pseudo-en-tête qui prend en compte les adresses source et destination. Le contrôle d'erreur d'UDP, facultatif pour IPv4, devient ainsi obligatoire. Pour ICMPv6, le contrôle d'erreur intègre le pseudo-en-tête alors que, pour ICMPv4, il ne portait que sur le message ICMP.

La fonction de fragmentation a aussi été retirée des routeurs. Les champs de l'en-tête IPv4 qui s'y reportent (`Identification, Drapeau, Place du fragment`) ont été supprimés. Normalement, les algorithmes de découverte du PMTU (*Path MTU*) évitent d'avoir recours à la fragmentation. Si celle-ci s'avère nécessaire, une extension est prévue et le découpage en fragments est réalisé uniquement au niveau de l'émetteur.

Une autre évolution majeure depuis l'en-tête IPv4 est la spécification des extensions d'en-tête pour remplacer les options. En effet, dans le cas d'IPv4, les options sont incluses dans l'en-tête. Celui-ci est donc de taille variable (taille indiquée dans le champ `Internet Header Length`),

ce qui peut compliquer le traitement dans les routeurs intermédiaires. Les extensions à l'en-tête IPv6 simplifient la mise en œuvre de ces fonctionnalités et permettent de garder la taille de l'en-tête IPv6 fixe à 40 octets.

Conclusion

Cette activité a présenté l'en-tête IPv6, définissant une nouvelle version du protocole IP. Cette version reprend les caractéristiques communes du protocole IPv4 : des adresses de taille fixe, un en-tête définissant une position fixe pour l'adresse de destination, position de plus alignée en mémoire. L'objectif est d'avoir un traitement simple et donc rapide de l'en-tête dans les routeurs. IPv6 garde donc ces principes et va même plus loin que son prédécesseur IPv4 en reconsidérant des mécanismes jugés coûteux, comme la fragmentation, le *checksum* ou les options. Certains de ces mécanismes ont été éliminés dans la nouvelle version du protocole, d'autres remplacés par des mécanismes plus performants. Enfin, nous pouvons signaler ce formulaire qui présente l'essentiel du paquet IPv6 en une seule page[4]. Ceci peut vous être utile pour la suite.

Références bibliographiques

1. ↑ Wikipedia.Définition de la scalabilité
2. ↑ MultiProtocol Label Switching https://fr.wikipedia.org/wiki/Multiprotocol_Label_Switching
3. ↑ Lee, D.C.; Lough, D.L.; Midkiff, S.F.et al. (1998). IEEE Network, Vol. 12, No. 1, January. The next generation of the Internet: aspects of the Internet protocol version 6.
4. ↑ Stretch, J. (2009) packetlife.net. Aide-mémoire tout en une page. IPv6

Pour aller plus loin

RFC et leur analyse par S. Bortzmeyer :

- RFC 791 Internet protocol (IPv4)
- RFC 1190 Experimental Internet Stream Protocol, Version 2 (ST-II)
- RFC 1819 Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+
- RFC 2205 Resource ReSerVation Protocol (RSVP)
- RFC 2474 Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers Analyse
- RFC 2675 IPv6 Jumbograms
- RFC 6040 Tunnelling of Explicit Congestion Notification
- RFC 6437 IPv6 Flow Label Specification
- RFC 6724 Default Address Selection for Internet Protocol version 6 (IPv6) Analyse
- RFC 7098: Using the IPv6 Flow Label for Load Balancing in Server Farms Analyse
- RFC 8200 Internet Protocol, Version 6 (IPv6) Specification Analyse
- RFC 9098 Operational Implications of IPv6 Packets with Extension Headers Analyse

Annexe 1: la gestion de la qualité de service

L'Internet différencié permet aux fournisseurs d'accès de gérer différemment les congestions qui surviennent dans le réseau. Sans différenciation, les paquets ont la même probabilité de rejet. Avec la différenciation, plusieurs classes de trafic sont définies. Les paquets appartenant aux classes les plus élevées ont une probabilité de rejet plus faible. Bien entendu, pour que l'introduction de telles classes de trafic soit efficace, il faut introduire une gestion des ressources différente pour chacune des classes, et des mécanismes de contrôle pour vérifier que les flux des utilisateurs n'utilisent pas que les classes les plus élevées ou qu'ils ne dépassent pas leur contrat. Par exemple, un client peut établir un contrat de niveau de services appelé SLA (*Service Level Agreement*) avec son fournisseur d'accès.

L'intérêt principal de la différenciation de services est qu'elle ne casse pas le modèle initial de l'Internet (version 4 ou version 6). Les flux sont toujours traités en *Best Effort* même si certains sont plus *Best* que d'autres. Il n'y a aucune garantie qu'un trafic d'une classe haute arrive à destination, mais la probabilité est plus importante. L'autre intérêt des classes de trafic vient de la possibilité d'agrégation des flux. La classe d'appartenance est indiquée dans l'en-tête du paquet. Les applications peuvent marquer les paquets en fonction de paramètres locaux (flux multimédia, flux interactif, trafic priorisé...). Le fournisseur d'accès qui récupère le trafic n'a plus à se préoccuper des applicatifs. Il vérifie que le trafic d'une classe ne dépasse pas le contrat préalablement établi.

Dans le cœur du réseau, les routeurs prennent en compte les différentes classes. Le fournisseur d'accès devra également passer des accords avec les autres opérateurs pour pouvoir faire transiter les flux avec un traitement approprié. Cet aspect de dimensionnement de réseau et de négociation d'accords d'échange est au coeur du métier d'opérateur.

Pour signifier l'appartenance d'un paquet à une certaine classe de trafic, une valeur est renseignée au niveau de l'en-tête IP dans l'octet `Traffic Class` afin qu'elle puisse être analysée par tous les routeurs mettant en œuvre le traitement différencié. Le format de ce champ est rappelé en figure 2.

Le tableau 1 présente les différentes valeurs définies pour le champ DSCP (*Differential Service Code Point*). Les valeurs sont présentées en format binaire avec les 6 bits les plus significatifs de l'octet `Traffic Class`, puis leur conversion en décimal, leur nommage, la probabilité d'écartement, et l'équivalence avec les anciennes valeurs du champ TOS de l'IPv4 :

DSCP Value	Decimal Value	Meaning	Drop Probability	Equivalent Precedence Value
101 110	46	EF Expedited Forwarding	N/A	101 – Critical
101 100	44	Voice Admit	N/A	101 – Critical (RFC 5865)
000 000	00	Best Effort / Default	N/A	000 – Routine
001 010	10	AF11 Assured Forwarding	Low	001 – Priority
001 100	12	AF12	Medium	001 – Priority
001 110	14	AF13	High	001 – Priority
010 010	18	AF21	Low	010 – Immediate
010 100	20	AF22	Medium	010 – Immediate
010 110	22	AF23	High	010 – Immediate
011 010	26	AF31	Low	011 – Flash
011 100	28	AF32	Medium	011 – Flash
011 110	30	AF33	High	011 – Flash
100 010	34	AF41	Low	100 – Flash Override
100 100	36	AF42	Medium	100 – Flash Override
100 110	38	AF43	High	100 – Flash Override
001 000	08	CS1 Class Selector		1 – Priority
010 000	16	CS2		2 – Immediate
011 000	24	CS3		3 – Flash
100 000	32	CS4		4 – Flash Override
101 000	40	CS5		5 – Critic /ECP
110 000	48	CS6		6 – Internetwork Control
111 000	56	CS7		7 – Network Control

Tableau 1 : Format du champ DSCP.

Pour l'instant, deux types de comportement sont standardisés :

- Assured Forwarding [RFC 2597] : ce comportement définit quatre classes de trafic et trois priorités, suivant que l'utilisateur respecte son contrat, le dépasse légèrement, ou est largement en dehors. Les classes sont donc choisies par l'utilisateur et restent les mêmes tout au long du trajet dans le réseau. La priorité, par contre, peut être modifiée dans le réseau par les opérateurs en fonction du respect ou non des contrats. Par exemple, pour la classe AF n°2, on dispose des 3 priorités suivantes : AF21, AF22 et AF23. Plus le chiffre est élevé, plus la priorité est faible. C'est-à-dire qu'en cas de saturation de cette classe de trafic, les paquets AF23 seront écartés avant AF22, puis AF21.
- Expedited Forwarding [RFC 2598] : ce comportement est comparable à un circuit à débit constant réservé dans le réseau. Le trafic est mis en forme à l'entrée du réseau, en retardant l'émission des paquets qui sont hors contrat. En plus de ces comportements, l'octet DS a gardé, pour des raisons de compatibilité avec les équipements existants, les valeurs du bit ToS qui étaient le plus fréquemment utilisées. La valeur est 0xB8 (1011 1000 en binaire, et en tenant compte uniquement des 6 bits de poids forts : 46 en décimal).
- Voice Admit : cette autre valeur a été par la suite proposée dans le RFC 5865 pour affiner le traitement de flux *temps réel* de différentes natures : voix, vidéo, signalisation *temps*

réel… (La valeur est 0xB0 soit 1011 0100 en binaire, et en tenant compte uniquement des 6 bits de poids forts : 44 en décimal).

- Network Control : autre particularité : la valeur 0xE0 (1110 0000 en binaire, et en tenant compte uniquement des 6 bits de poids forts : 56 en décimal) correspond à CS7, la classe de contrôle du réseau (*Network Control*). Elle est utilisée dans des mises en œuvre d'IPv6 pour l'émission de certains paquets ICMPv6. Cette valeur est dépréciée. Il est conseillé d'utiliser la valeur CS6 comme spécifié dans le RFC 4594.

Références bibliographiques

- RFC 2597 Assured Forwarding PHB Group
- RFC 2598 An Expedited Forwarding PHB
- RFC 4594 Configuration Guidelines for DiffServ Service Classes
- RFC 5865 A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic

Annexe 2: Le mécanisme d'extension de l'en-tête IPv6

Les extensions de l'en-tête IPv6 visent à ajouter des fonctionnalités supplémentaires à l'acheminement d'un paquet dans l'Internet. Ces extensions vont impliquer le traitement de ces fonctionnalités au niveau réseau, soit par le destinataire du paquet IPv6, soit par les routeurs intermédiaires en charge de l'acheminement du paquet IPv6.

De nombreuses extensions ont été définies, afin d'assurer des fonctions comme le routage par la source, la gestion de la fragmentation, la confidentialité des communications (mécanisme *ipsec*), etc.

Le mécanisme d'extension de l'en-tête IPv6 est assez souple pour pouvoir inclure d'autres fonctionnalités futures. Dans cette activité, nous allons nous intéresser au principe du traitement des extensions au moyen d'exemples simples et démonstratifs de ce mécanisme.

Cette annexe décrit les différents mécanismes qui enrichissent les fonctions disponibles au niveau de la couche réseau : routage, fragmentation, sécurité, etc. Ces mécanismes tirent parti de la possibilité d'ajouter des champs supplémentaires à l'en-tête IPv6 grâce aux extensions d'en-tête. Ces extensions sont ajoutées par les extrémités de la communication et sont transparentes pour les routeurs (exception faite des extensions de type *Hop-by-Hop*). De plus, le mécanisme de chainage par le champ *Next Header* permet d'ajouter des extensions de manière souple.

L'usage des extensions est encore assez limité sur l'Internet. Certaines fonctionnalités ont été dépréciées (comme l'extension de routage RH0) et d'autres peinent à se développer (comme la mobilité IPv6). La présence potentielle de ces extensions doit être cependant prise en compte dans le traitement des paquets, notamment le filtrage sur les valeurs du champ *Next Header* de l'en-tête IPv6.

Principe des extensions IPv6

Les extensions peuvent être vues comme un protocole 3.5, entre les couches 3 (réseau) et 4

(transport) du modèle OSI. En effet, à part l'extension de proche-en-proche, qui est traitée par tous les routeurs traversés, les autres extensions ne sont traitées que par le destinataire du paquet (i.e. celui spécifié dans le champ adresse de destination du paquet IPv6). Une extension a une longueur multiple de 8 octets (64 bits). Elle commence par un champ `Next header` qui définit, sur un octet, le type d'extension ou de protocole qui suit. Pour les extensions de longueur variable, l'octet suivant contient la longueur de l'extension en mots de 8 octets, le premier mot n'étant pas compté. Par exemple, une extension de 16 octets aura une longueur de 1.

Si, d'un point de vue théorique, les extensions sont supérieures aux options d'IPv4, dans la réalité très peu sont utilisées à grande échelle et restent du domaine de la recherche.

Nota : *dans la pratique, l'extension la plus couramment rencontrée est probablement l'option de fragmentation à la source. En effet, certains protocoles applicatifs sur UDP, tel que NFS, supposant que la fragmentation existe au niveau réseau, produisent des messages de taille quelconque sans se soucier du MTU. Comme il n'est pas envisageable de modifier ces applications largement déployées, la couche réseau IPv6 doit être capable de gérer la fragmentation. IPv6 impose simplement que cette dernière se fasse à la source.*

Une présentation illustrée des extensions IPv6 peut être consultée sur le site de Cisco[1].

Le champ *Next Header*

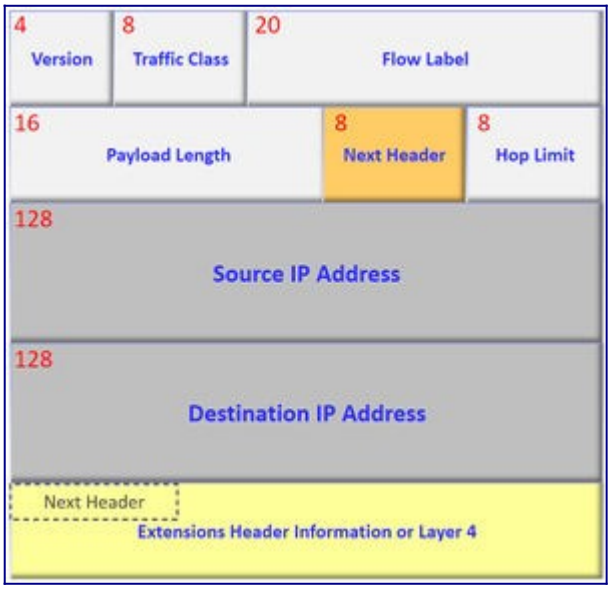


Figure 1 : Localisation du champ `Next Header` dans l'en-tête IPv6.

Le champ `Next Header` de l'en-tête IPv6, comme illustré sur la figure 1, identifie généralement le protocole de niveau supérieur comme, par exemple, le transport TCP/UDP. Mais, dans le cas des extensions, plusieurs mécanismes particuliers sont également disponibles parmi la liste suivante :

Valeurs du champ en-tête suivant pour IPv6

valeur Hexa Protocole ou

		Extension
0	0x00	*Proche-en-proche*
4	0x04	IPv4
6	0x06	TCP
17	0x11	UDP
41	0x29	IPv6
43	0x2b	*Routage*
44	0x2c	*Fragmentation*
50	0x32	*Confidentialité*
51	0x33	*Authentification*
58	0x3a	ICMPv6
59	0x3b	Fin des en-têtes
60	0x3c	*Destination*
132	0x84	SCTP
135	0x87	*Mobilité*
136	0x88	UDP-lite
140	0x8c	*Shim6*

Intégration des extensions d'en-tête dans le paquet IPv6

Quand il y a plusieurs extensions dans un même datagramme, les extensions sont placées selon un ordre qui dépend de leur portée :

- extensions impliquant tous les routeurs intermédiaires : *Hop-by-Hop* ;
- extensions impliquant seulement certains routeurs désignés : *Routing* ;
- extension impliquant le destinataire : *Authentication*, *Encapsulating Security Payload*, Fragmentation, Destination.

La figure 2 montre la souplesse avec laquelle plusieurs extensions peuvent être chaînées. Chaque extension contient dans son en-tête un champ `en-tête suivant` et un champ `longueur`. Le premier paquet ne contient pas d'extension ; le champ `en-tête suivant` pointe sur ICMPv6. Le second paquet ne contient pas d'extension ; le champ `en-tête suivant` pointe sur TCP. Le troisième paquet contient une extension de protection qui pointe ensuite sur UDP. Dans le dernier paquet, une extension de routage, qui pointe sur une extension de fragmentation, pointe finalement sur ICMPv6.

- L'enchaînement des extensions se fait dans un ordre bien déterminé. Par exemple, une extension concernant tous les routeurs sur le chemin (*Hop-by-Hop*) devra forcément se trouver en première position. Si elle se trouvait à la suite d'une extension *Destination*, elle ne pourrait être lue, l'extension *Destination* n'étant interprétée que par le destinataire du paquet.
- Si cet enchaînement d'extension offre beaucoup plus de souplesse que les options d'IPv4, il rend difficile la lecture des numéros de port. Il faut en effet lire tout l'enchaînement d'extension pour arriver au protocole de niveau 4. Ceci a servi de justification à l'identificateur de flux qui permettait de refléter au niveau 3 un flux particulier et évitait de dérouler l'enchaînement. Bien entendu, les pare-feux devront vérifier les numéros de ports.
- Les extensions peuvent être vues comme un protocole 3.5 (entre la couche 3 et la couche 4). En effet, à part l'extension de "proche en proche", qui est traitée par tous les routeurs traversés, les autres extensions ne sont traitées que par le destinataire du paquet (*i.e.* celui spécifié dans le champ `adresse de destination` du paquet IPv6).

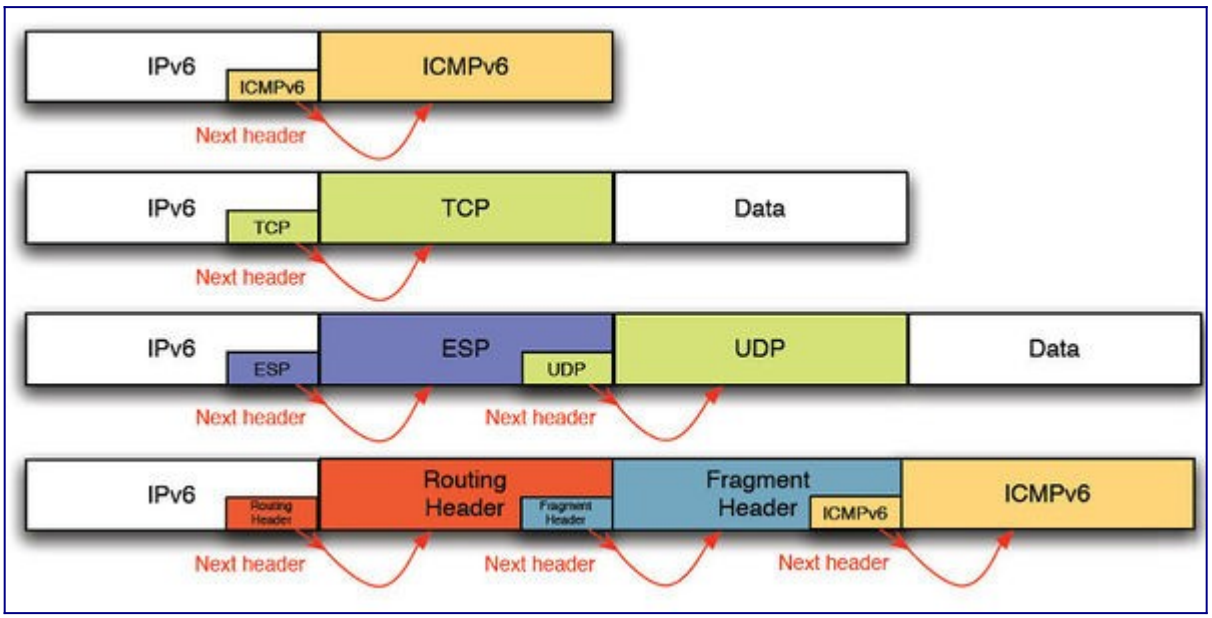


Figure 2 : Enchaînement d'extensions.

Quelques exemples

Fragmentation

Nous avons vu la fragmentation en détail lors de l'activité consacrée à la taille des paquets IPv6. Les points essentiels à rappeler sont que la fragmentation en IPv6 est une fonction d'hôtes exclusivement, et que par souci d'efficacité, l'adaptation de la taille de l'unité de données applicatives en unité de transfert est du ressort de la couche de transport. Cependant, lorsque ce n'est pas possible, en dernier ressort, la fragmentation est faite par IPv6. Les informations pour la fragmentation sont placées dans une extension d'en-tête IPv6 identifiée par la valeur 44. D'un point de vue du codage, l'en-tête et les extensions qui concernent les routeurs intermédiaires (pour l'instant "proche en proche" et "routage par la source") sont

recopiées dans chaque fragment, tandis que les autres extensions et l'en-tête de niveau 4 ne seront présents que dans le premier fragment.

La fragmentation en IPv4 n'est pas très performante. Quand un routeur ne peut pas transmettre un paquet à cause de sa trop grande taille, il doit le transmettre en fragments. Comme l'Internet n'est pas fiable, un fragment peut se perdre. Le destinataire ne peut rassembler les fragments. Cela revient à perdre tous les fragments. Il est donc plus efficace de faire les fragments avant la couche IP. Ainsi le fragment devient un paquet IP à la bonne taille. Maintenant, en cas de perte, seul le paquet perdu sera retransmis, évitant ainsi la retransmission de gros blocs de données.

Il est plus intéressant d'adapter la taille des paquets à l'émission. Ceci est fait en utilisant les techniques de découverte du MTU (voir RFC 8201 : Mécanisme de découverte du PMTU). En pratique, une taille de paquets de MTU 1280 octets est une bonne garantie d'acheminement du paquet.

Extensions d'authentification (AH) et de confidentialité (ESP)

L'extension d'authentification (AH : Authentication Header) décrite dans le RFC 4302 permet de s'assurer que l'émetteur du message est bien celui qu'il prétend être. Elle sert aussi au contrôle d'intégrité pour garantir au récepteur que personne n'a modifié le contenu d'un message lors de son transfert sur le réseau. Elle peut optionnellement être utilisée pour détecter les rejeux.

Le principe de l'authentification est relativement simple. L'émetteur calcule un authentificateur sur un datagramme et l'émet avec le datagramme sur lequel il porte. Le récepteur récupère cette valeur et vérifie qu'elle est correcte. Si un code d'authentification de message [2] est utilisé, il suffit au récepteur de calculer de son côté le code sur le même datagramme à l'aide de la clé symétrique partagée et de le comparer avec le code reçu. Si le mécanisme de signature numérique est employé, le récepteur doit alors récupérer la signature, la déchiffrer avec la clé publique de l'émetteur et comparer le condensat ainsi obtenu avec celui calculé de son côté sur le datagramme reçu. Si les deux codes, ou les deux condensats diffèrent, soit l'émetteur ne possède pas la bonne clé, soit le message a subi des modifications en chemin.

L'extension ESP *Encapsulating Security Payload* décrite dans le RFC 4303 permet de chiffrer l'ensemble des paquets ou leur partie transport et de garantir l'authentification et l'intégrité de ces paquets. Cette extension permet optionnellement de détecter les rejeux (à condition que le service d'authentification soit assuré) et garantit de façon limitée la confidentialité du flux.

Pour obtenir ces services de sécurité, il est nécessaire, avant d'émettre un paquet IP sur le réseau, de chiffrer les données à protéger, de calculer un authentificateur, et d'encapsuler ces informations dans l'en-tête de confidentialité ESP. Cela nécessite bien entendu l'existence d'une association de sécurité précisant, entre autres, le ou les algorithmes de chiffrement, la ou les clés, et un indice de paramètres de sécurité. Pour en savoir plus sur les extensions de sécurité, nous renvoyons le lecteur vers le livre[3].

Segment Routing Header (SRH)

Cette extension est une des briques qui permet la mise en œuvre de l'ingénierie de trafic et de la qualité de service dans les réseaux IPv6. Certaines applications ont besoin que le délai, le taux de perte des paquets et le débit ne dépassent pas un certain seuil. Par exemple, une session de vidéo-conférence peut nécessiter que le délai de bout en bout reste inférieur à 100 ms, que le débit disponible soit au minimum de 2 Mbit/s et que le taux de perte reste inférieur à 1 %. La solution habituellement mise en œuvre pour le respect de ces métriques de qualité de service est RSVP-TE[4]. RSVP est un protocole de réservation de ressources sur les routeurs du chemin entre une source et une destination. La composante d'ingénierie de trafic permet notamment de choisir explicitement le chemin qui sera suivi par le paquet. Lorsqu'une application a besoin de réserver des ressources, un circuit MPLS est établi et le protocole RSVP réserve les ressources sur l'ensemble des routeurs qui composent le chemin. La figure 4 résume ce comportement : le nœud A est la source du flot de vidéo-conférence destiné à B. Elle demande l'installation d'un chemin via MPLS. Le contrôleur détermine que le chemin le plus adapté est celui transitant par G,H,I,K,N source de vidéo-conférence. Elle effectue ensuite une réservation de ressource via RSVP. Chacun des routeurs G,H,I,K,N doit accepter la requête.

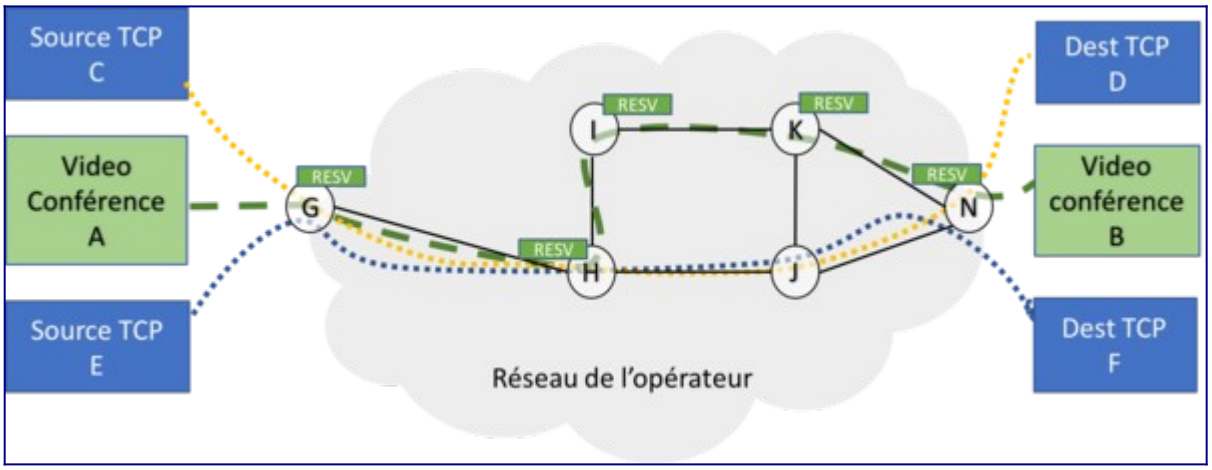


Figure 4 : Exemple d'utilisation de RSVP-TE pour assurer la qualité de service.

L'inconvénient de cette méthode est qu'avec RSVP, les réservations doivent être fréquemment renouvelées et les routeurs doivent garder des états relatifs à chacun des flots. À chaque fois qu'un routeur reçoit un paquet, il doit chercher dans la liste des flots le comportement à y associer. Cela limite le passage à l'échelle (le nombre de flots supportés). De plus, l'établissement des circuits MPLS est coûteux en temps, en états, et en communication. Pour finir, si un des nœuds ou des liens qui compose le chemin n'est plus disponible, le temps requis pour établir un nouveau chemin peut être trop important face aux besoins des flots [5].

Le concept de *segment routing* a été proposé pour palier ces désavantages. Il permet de réduire l'intelligence des nœuds du réseau. Un contrôleur choisit le chemin qui respecte les besoins de qualité de service du nœud. Le chemin à suivre ainsi que le traitement à associer

aux paquets sont ensuite explicitement listés dans leurs en-têtes. Ces traitements à appliquer au paquet sont appelées segments. Un segment peut être l'adresse d'un routeur par lequel le paquet devra transiter, un lien à emprunter ou encore un service fourni par le routeur sur lequel transite le paquet. La séquence de segments que doit suivre le paquet peut être implémentée via une liste de labels MPLS ou, dans le cas d'un réseau IPv6, via l'option *Segment Routing* de l'en-tête [6]. Contrairement à MPLS, le *segment routing* avec IPv6 ne requière pas que tous les nœuds traversés par le paquet soient compatibles avec cette option. Cela permet un déploiement progressif et la traversée de plusieurs types de réseaux. Notons qu'il n'est pas nécessaire d'inscrire l'intégralité du chemin dans la liste. Les règles de routage classique de l'IGP s'appliquent jusqu'au prochain nœud explicitement listé dans l'en-tête. Dans le cas de la figure 5, pour suivre le chemin G,H,I,K,N, il est seulement nécessaire d'ajouter I et B dans la liste. Lorsque le paquet est généré par A, l'adresse de destination est celle de I. Lorsque le paquet atteint I, l'adresse de destination est remplacée par la prochaine dans la liste ; c'est-à-dire B. Le paquet suit le plus court chemin entre I et B ; en l'occurrence I,K,N. Le surcoût induit par la taille de l'extension de l'en-tête est donc limité.

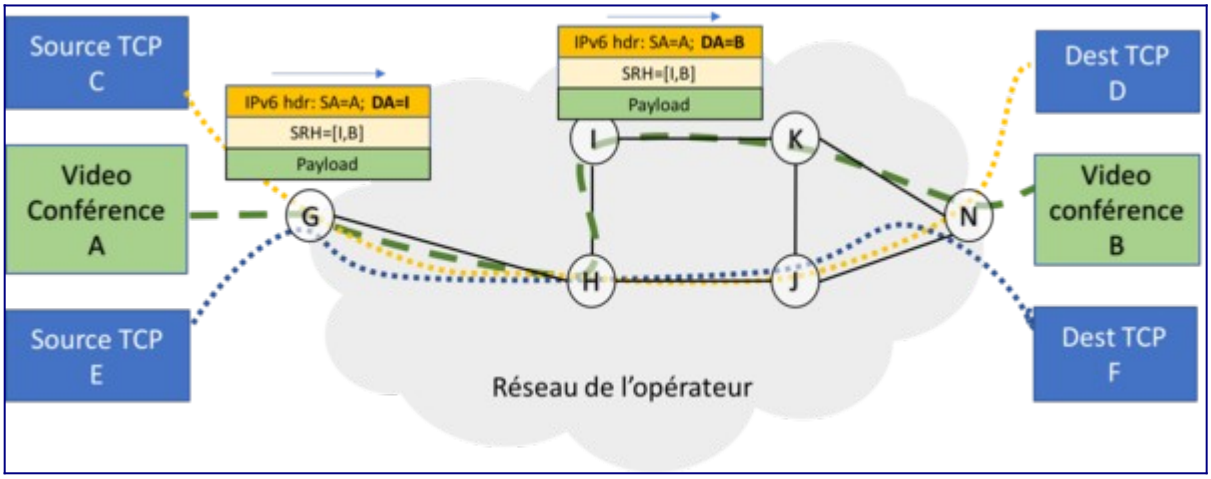


Figure 5 : Exemple d'utilisation du *Segment Routing* pour l'ingénierie de trafic et la qualité de service.

Le projet de RFC prévoit qu'un segment soit représenté sur 128 bits ; c'est-à-dire autant qu'une adresse IPv6. Le nombre de segments inclus dans l'option SRH peut être déduit du champ `Hdr Ext Len` qui indique le nombre d'octets de l'option moins un. Le prochain segment à appliquer est indiqué par le champ `Segments Left` qui indique le nombre de segments restant à appliquer au paquet. Le segment courant est inscrit dans le champ adresse de destination de l'en-tête IPv6. Lorsque le traitement associé à un segment est achevé, le prochain segment à traiter est copié dans le champ adresse de destination et le champ `Segments Left` est décrementé.

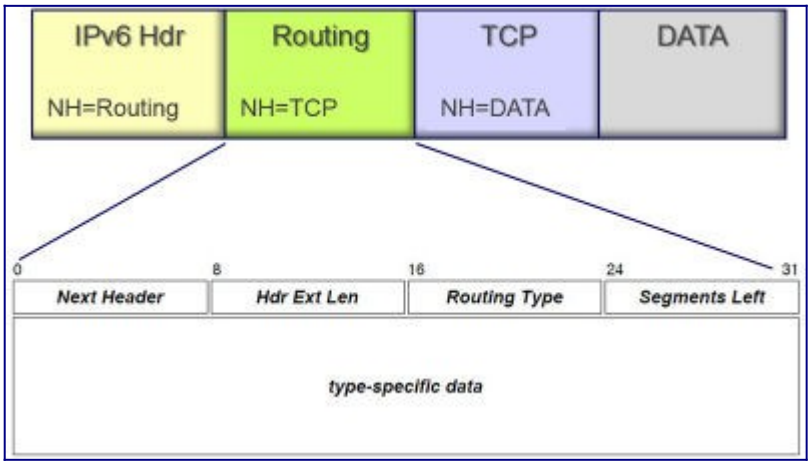


Figure 6 : Extension de routage par la source.

L'extension SRH est très proche de l'en-tête de routage défini dans le RFC 8200 de IPv6. L'en-tête de routage devait notamment permettre le routage à la source (voir la figure 6) qui consiste à lister les routeurs que devaient traverser le paquet au cours de son acheminement. Cette pratique a été dépréciée en 2007 par le RFC 5095 pour des raisons de sécurité. En effet cela permettait de générer de la congestion sur certains chemins, des attaques de déni de service, des contournements des règles de filtrages de certains pare-feux ou encore d'utiliser des machines publiques accessibles (en DMZ) pour accéder à des machines privées. [7].

Pour être acceptée, l'extension SRH doit donc garantir qu'elle ne serait pas vulnérable à ces attaques. Plusieurs cas d'utilisation sont décrits. Lorsque l'extension SRH est générée au sein du réseau de l'opérateur (i.e. routeur de bordure), les attaques listées dans le RFC 5095 sont inopérantes. Les routeurs de bordures doivent simplement filtrer les paquets provenant de l'extérieur pour éviter que des hôtes s'attribuent des privilèges ou provoquent de la congestion sur certains liens. Lorsque les SRH sont générées en dehors du réseau de l'opérateur, le projet de RFC prévoit l'utilisation d'un mécanisme d'authentification de l'en-tête défini par le RFC 2104 (option HMAC). La clé de chiffrement de l'en-tête HMAC est distribuée au sein des nœuds autorisés à générer ou manipuler des extensions d'en-têtes SRH.

Références bibliographiques

1. ↑ Cisco (2006) White paper.IPv6 Extension Headers Review and Considerations
2. ↑ Code d'authentification de message, Article Wikipedia
3. ↑ G. Cizault. livre "IPv6, Théorie et Pratique". Chapitre sur la Sécurité
4. ↑ Packet Pushers, novembre 2016.Deep dive in the RSVP-TE protocol
5. ↑ Cisco, SR Traffic Engineering
6. ↑ Previdi & al. IPv6 Segment Routing Header (SRH) (Draft)
7. ↑ P. Biondi et A. Ebalard, CanSecWest, 2017 .IPv6 Routing Header Security

Pour aller plus loin

Vous pouvez approfondir vos connaissances en consultant les documents de ce paragraphe.

RFC et leur analyse par S. Bortzmeyer :

- RFC 4302 : IP Authentication Header
- RFC 4303 : IP Encapsulating Security Payload (ESP)
- RFC 5095 : Deprecation of Type 0 Routing Headers in IPv6 Specification Analyse
- RFC 7045 : Transmission and Processing of IPv6 Extension Headers Analyse
- RFC 7872 : Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World Analyse
- RFC 8200 : Internet Protocol, Version 6 (IPv6) Specification Analyse
- RFC 8201 : Path MTU Discovery for IP version 6 Analyse

Activité 22: L'acheminement des paquets IPv6

Introduction : Qu'est ce que le routage ?

Nous avons vu les aspects statiques du protocole IPv6 dans l'activité précédente. Nous allons voir les aspects opérationnels de ce protocole. L'objectif d'IPv6 est de réaliser un réseau virtuel qui assure un service de connectivité par la remise de datagrammes. Ce réseau doit être doté de moyens d'acheminement de datagrammes jusqu'au destinataire final. Les deux fonctions essentielles de n'importe quel réseau sont l'adressage et le routage. L'adresse IP sert à l'identification des nœuds dans le réseau virtuel mais également à leur localisation. Le routage est la fonction indispensable au réseau pour acheminer un paquet vers sa destination[1]. Cette fonction fait référence au traitement des routes. Elle recouvre deux activités : l'établissement des routes (*routing*), c'est-à-dire l'identification des chemins pour atteindre les différentes destinations du réseau, et la détermination d'une route (*forwarding*) pour acheminer le datagramme. L'acheminement du datagramme consiste à trouver et suivre une route pour atteindre le destinataire. Les éléments du routage IPv6 sont l'hôte, le lien et le routeur. L'hôte est un nœud d'extrémité (source et/ou destination). Les hôtes peuvent avoir plusieurs interfaces qui ont chacune une adresse IP. L'interface réseau constitue le point d'accès au réseau physique. Les hôtes émettent et reçoivent des datagrammes mais ils n'ont pas la capacité de relayage. Les datagrammes reçus qui ne lui sont pas destinés sont purement et simplement détruits. Le Lien IPv6 est un réseau physique identifié par un préfixe réseau. Ce préfixe sert de localisateur dans l'interconnexion de réseaux (ou Internet). Enfin, le routeur relie deux liens IPv6. Il est l'élément indispensable de l'Internet. Il utilise le préfixe réseau pour le routage. Un routeur est un nœud intermédiaire connecté à deux ou plusieurs liens IPv6 simultanément, ayant la connaissance de la topologie du réseau et donc capable d'effectuer un choix de route (action de routage) pour ensuite relayer des paquets entre les interfaces (action de commutation).

Topologie

La topologie de réseau correspond à l'arrangement (physique ou logique) de ses nœuds et de ses liaisons.

Dans le modèle de l'Internet, la fonction de routage est présente dans les deux types de systèmes existants qui ont chacun leur objectif de routage. Les hôtes doivent trouver un routeur. Un hôte ignore le chemin, il envoie son trafic à un routeur local. Le routeur local signifie un routeur qui est sur le même lien que l'hôte. Cependant, un hôte doit pouvoir communiquer en l'absence de routeur, tandis que les routeurs doivent trouver un chemin. Le routeur effectue une fonction supplémentaire dans l'acheminement des paquets : le relayage, qui est l'action de commuter.

Dans un réseau en mode datagramme, le routage est une fonction commune à tous les nœuds du réseau et propre à la couche de réseau. De plus, il est utilisé par chaque paquet et il s'effectue indépendamment des systèmes de transmission sous-jacents. L'acheminement du datagramme est fait de voisin à voisin ou, encore dit, de proche en proche. Un voisin est défini comme un nœud qui partage la même connectivité physique, c'est-à-dire un nœud connecté

sur le même lien IPv6 et utilisant le même préfixe réseau IPv6. Chaque hôte (ou routeur) connait uniquement le voisin suivant où le datagramme doit être envoyé. On parle de prochain saut (*next hop*). Un datagramme est acheminé vers le destinataire final par sauts successifs de voisin à voisin. Le routage de paquets consiste à déterminer le meilleur voisin pour atteindre le destinataire. Le routage IP est un routage d'interconnexion de réseaux physiques. Le routage IP choisit donc un des réseaux physiques pour véhiculer le datagramme vers le prochain saut.

Quand un paquet IPv6 arrive à un routeur, celui-ci décide si ce paquet lui est destiné ou s'il doit le relayer. Dans ce dernier cas, le routage consiste à déterminer la route ; autrement dit vers quelle interface faire sortir le paquet afin qu'il atteigne sa destination. Cette décision s'appuie d'une part sur les informations contenues dans l'en-tête IP du paquet, principalement **l'adresse destination** ; d'autre part sur les informations obtenues du processus d'établissement des routes. Ces informations sont contenues dans la **table de routage** du nœud et constituent sa connaissance locale de la topologie du réseau. Avec ces informations, un nœud déterminera vers quelle interface faire sortir le paquet et à quel nœud le remettre. Ainsi, de proche en proche, le paquet sera acheminé depuis la source jusqu'à sa destination.

Le problème de fond du routage est : comment les routeurs acquièrent-ils l'information pour qu'ils puissent effectuer le bon choix de route ? La connaissance de la topologie du réseau peut être communiquée à chaque routeur de plusieurs façons. L'administrateur peut configurer manuellement la table de routage au niveau des différents routeurs. Mais ce mode de configuration est peu adapté lorsque le réseau évolue comme, par exemple, quand une nouvelle liaison apparait. On parle alors de **routage statique**. Une autre méthode consiste, pour chaque routeur, à propager sa connaissance locale du réseau et à intégrer les informations fournies par d'autres routeurs. Ces échanges s'effectuent grâce à des **protocoles de routage**. Avec ces échanges, une prise en compte automatique des évolutions du réseau est effectuée. On parle alors de **routage dynamique**.

Cette activité présente comment s'effectue l'acheminement des paquets et en particulier le choix de la route. Elle explique les différents types de route que l'on trouve dans une table de routage. Les protocoles de routage disponibles en IPv6 sont rappelés. Toutefois, les algorithmes de routage pour le calcul des routes sont hors du champ de ce cours.

Acheminement des paquets

Le routage d'un paquet par un routeur nécessite de prendre une décision afin de l'acheminer vers sa destination. Un paquet est à relayer lorsqu'il arrive sur un routeur et que l'adresse destination de ce paquet ne concerne aucune interface de ce routeur. Plusieurs cas sont alors possibles :

- la destination est sur un des réseaux sur lequel le routeur est directement connecté. Le paquet doit alors être remis directement à la destination ;
- la destination n'est sur aucun des réseaux directement connectés au routeur. ce dernier doit déterminer quel est le meilleur routeur voisin qui rapproche le paquet de sa destination finale. Le paquet doit alors être relayé vers cet autre routeur qui prendra en charge l'acheminement du paquet ;

- la destination est inconnue. Le routeur ne peut décider vers où le paquet doit être relayé. Le paquet doit donc être éliminé et un message d'erreur ICMP (ICMPv4 ou ICMPv6 selon la version du protocole IP utilisé) est émis vers la source du paquet pour lui indiquer le problème de routage.

La détermination du cas approprié se fait à partir des informations connues par le routeur contenues dans sa **table de routage**.

La table de routage

La table de routage d'un nœud contient la liste des réseaux accessibles depuis le nœud. La liste des réseaux se présente sous la forme d'une liste de préfixes réseau. À chacun de ces réseaux est associé le prochain saut (*Next Hop*) pour atteindre ce réseau depuis le nœud. Cette information va servir à la remise du paquet sous la forme de la transmission du paquet au prochain saut. Le prochain saut de la table de routage est un routeur qui est local au nœud. Ils partagent tous les deux le même préfixe réseau.

Parmi les réseaux connus dans la table de routage, on trouve les réseaux directement connectés au nœud ; c'est-à-dire que le nœud possède une interface connectée sur l'un de ces réseaux. Lorsque l'interface du nœud est configurée sur un réseau, elle obtient une adresse IPv6 à laquelle s'ajoute la longueur du préfixe ; c'est-à-dire le nombre de bits communs aux adresses de toutes les interfaces connectées au même réseau. À la table de routage IPv6 s'ajoute alors automatiquement le préfixe du réseau connecté, défini par les bits communs de l'adresse. Le prochain saut pour ce réseau est alors défini par l'identifiant de l'interface connectée à ce réseau. Cela signifie au nœud que les paquets destinés à ce réseau doivent être envoyés sur cette interface.

Voici un exemple de configuration d'une interface réseau et l'entrée correspondante dans la table de routage sur un système Linux. Notez bien la correspondance entre le préfixe de l'adresse de l'interface `eth0` et l'entrée correspondante dans la table de routage. La commande `ifconfig` montre l'adresse configurée à l'interface réseau `eth0` et le préfixe associé à ce réseau. La commande `netstat` affiche la table de routage. L'absence d'information dans la colonne 'Next Hop' indique que ce préfixe est associé au réseau de l'interface `eth0`. Enfin, la commande `ip` fait la même chose mais présentée sous une autre forme.

```
$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:18:73:68:21:20
          inet6 addr: 2001:db8:1:1:218:73ff:fe68:2120/64 Scope:Global
          inet6 addr: fe80::218:73ff:fe68:2120/64 Scope:Link
(...)

$ netstat -rn -A inet6
Kernel IPv6 routing table
Destination                    Next Hop                   Flag Met Ref Use
If
2001:db8:1:1::/64              ::                          UAe  256 0345733
eth0
(...)

$ ip -6 route
2001:db8:1:1::/64 dev eth0   proto kernel   metric 256   expires 2592155sec
```

```
mtu 1500 advmss 1440 hoplimit 0
(...)
```

La table de routage peut aussi comporter des préfixes de réseaux auxquels le nœud n'est pas directement connecté. Ces préfixes peuvent être ajoutés par l'administrateur réseau ou appris automatiquement à l'aide de protocoles de routage. Ces préfixes peuvent être spécifiques à un réseau local (généralement de longueur 64 bits) mais peuvent être plus larges pour désigner un ensemble de réseaux. Plus la longueur du préfixe est faible et plus l'ensemble des réseaux considéré est large. Le prochain saut est alors configuré avec l'adresse d'un routeur qui va prendre en charge la suite de l'acheminement du paquet.

L'exemple suivant montre une table de routage d'un routeur VyOS comportant un préfixe plus large que celui connecté sur son interface. Notez que l'adresse du prochain saut est une adresse **lien-local**, ce qui signifie que le nœud vers lequel transmettre le paquet est sur le réseau connecté à l'interface `eth0`.

```
vyos(config)# do show ipv6 route
C>* 2001:db8:1:1::/64 is directly connected, eth0
S>* 2001:db8:1::/48 [110/1] via fe80::290:bff:fe1e:c4fe, eth0, 1d09h16m
```

Un dernier type d'entrée de la table de routage permet à un nœud de relayer les paquets pour tous les réseaux qu'il ne connait pas, évitant ainsi de les éliminer parce qu'il n'a pas une connaissance suffisante du réseau. Cette entrée s'appelle la **route par défaut**. Le préfixe utilisé pour désigner ainsi tous les réseaux ne doit comporter aucun bit spécifié. En IPv6, ce préfixe se note `::/0` ; la longueur du préfixe à 0 signifiant qu'aucun bit n'est spécifié comme commun. La route par défaut possède comme prochain saut l'adresse du routeur qui prendra en charge le routage des paquets vers les réseaux non connus localement. Ce routeur est communément appelé **routeur par défaut**, ou **passerelle par défaut**. Dans un réseau local domestique par exemple, le routeur par défaut des hôtes, comme un ordinateur portable, est généralement le boitier de l'opérateur, car c'est lui qui sait comment joindre les différents réseaux de l'Internet.

L'exemple suivant montre l'entrée correspondant à la route par défaut d'un nœud sous Windows 7 avec l'outil en ligne de commande `netsh`.

```
netsh> interface ipv6
netsh interface ipv6> show routes
Recherche du statut actif...

Type      Mét  Préfixe                  Idx  Nom passerelle/interface
--------  ---  -----------------------  ---  ------------------------
Auto        8  2001:db8:1:1::/64          4  Connexion au réseau local 4
Auto      256  ::/0                        4  fe80::290:bff:fe1e:c4fe
```

Remise directe et remise indirecte

L'acheminement du paquet IPv6 est constitué par deux modes de remise. La **remise directe**, quand le nœud en charge du paquet et la destination sont tous les deux reliées directement au même réseau physique. La communication se fait sans routeur. Dans le cas présenté par la figure 1, les deux hôtes A et B peuvent communiquer directement car ils sont connectés sur le

même réseau local Ethernet à l'aide d'un lien mis en œuvre sous la forme d'un commutateur qui relaie les trames de manière transparente. Le préfixe IPv6 `2001:db8:0001::/64` est commun à chaque nœud. Ainsi, les échanges sont possibles directement, sans l'intervention d'un routeur. Dans la table de routage, le Next Hop est vide comme le montre la figure 1 (l'adresse IPv6 notée '::' est l'adresse nulle).

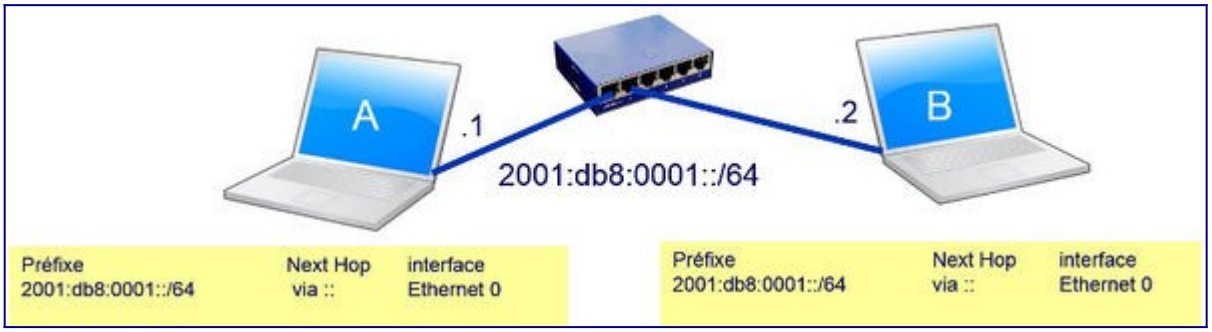


Figure 1 : Route directe.

Quand la destination n'est pas directement reliée au même réseau physique que le nœud en charge du paquet, autrement dit lorsqu'il n'y a pas de lien direct entre deux nœuds, le paquet transite par au moins un routeur. Le paquet est remis à un routeur et on parle de **remise indirecte**. Le ou les routeurs traversés s'occuperont du transfert du paquet jusqu'au nœud de destination. Le champ Hop Limit dans l'en-tête du paquet IPv6 est décrémenté d'une unité à chaque relayage effectué par un routeur.

Dans l'exemple présenté par la figure 2, le nœud A peut atteindre les deux hôtes B et C. Par contre, B et C ne peuvent pas directement communiquer car ils sont connectés sur des réseaux avec des préfixes IPv6 différents. La table de routage de l'hôte B doit être complétée avec une entrée avec le préfixe distant `2001:db8:0002::/64` en précisant l'adresse de A, `2001:db8:0001::1/64`, comme nœud intermédiaire. Les paquets émis depuis B vers C seront dès lors relayés par le routeur A. L'adresse du routeur A à retenir dans la table de routage de B est l'adresse partageant le même préfixe que l'adresse de B. De manière symétrique, il convient d'avoir la même configuration de la table de routage de C à savoir une entrée avec comme destination le préfixe réseau de B `2001:db8:0001::/64` en précisant l'adresse de A, `2001:db8:0002::1/64`, comme prochain saut. Sans quoi, l'hôte C ne sera pas en mesure de communiquer avec B.

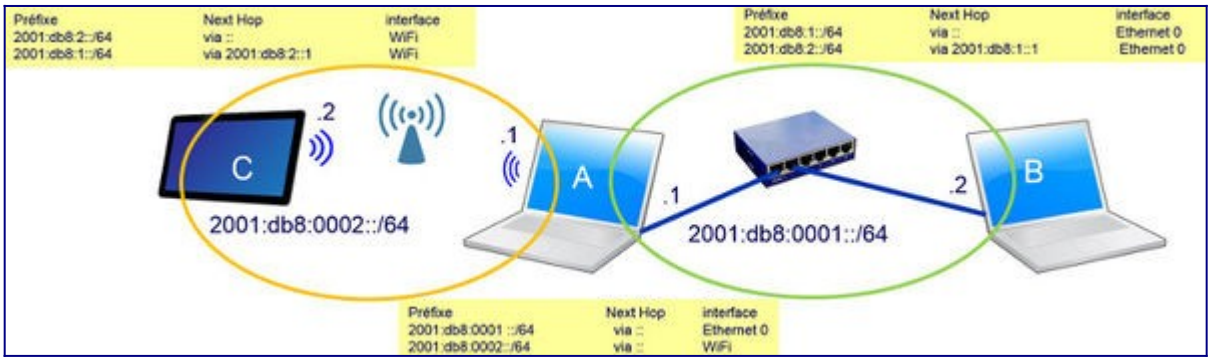


Figure 2 : Route indirecte.

Une route directe entre deux hôtes revient à effectuer une remise directe. Une route indirecte entre deux hôtes est constituée par une ou plusieurs remises indirectes et se termine par une remise directe. Le dernier routeur de la route fait une remise directe à la destination finale. À noter aussi qu'une fois que le nœud du prochain saut est connu, que ce soit en remise directe ou en remise indirecte, il faut transmettre physiquement le paquet à ce nœud. La *découverte des voisins* opérée par ICMPv6 va alors servir à déterminer l'adresse physique associée à l'adresse IPv6 du nœud voisin. Cette opération est essentielle pour effectuer ensuite la transmission du paquet. Cette fonction de résolution d'adresse IPv6 sera développée dans la prochaine séquence.

Route par défaut

Lorsqu'un réseau est raccordé à Internet, il n'est pas nécessaire de détailler toutes les destinations de l'Internet dans la table de routage. Cependant, il faut pouvoir acheminer des paquets vers des réseaux qui ne sont pas connus des nœuds. La table de routage doit contenir dans ce cas une entrée pour indiquer vers quel routeur un nœud doit émettre les paquets.

Dans le cas simple, un routeur par défaut est spécifié et tous les paquets qui visent des destinations inconnues lui seront remis. En quelque sorte, on fait confiance aux capacités et à la connectivité de ce routeur. Une route par défaut est présente dans la table de routage du routeur par défaut. Cette route par défaut joue le rôle du panneau *toutes directions* dans le réseau routier. L'exemple de la figure 3 montre la configuration avec une route par défaut sur le routeur IPv6 :

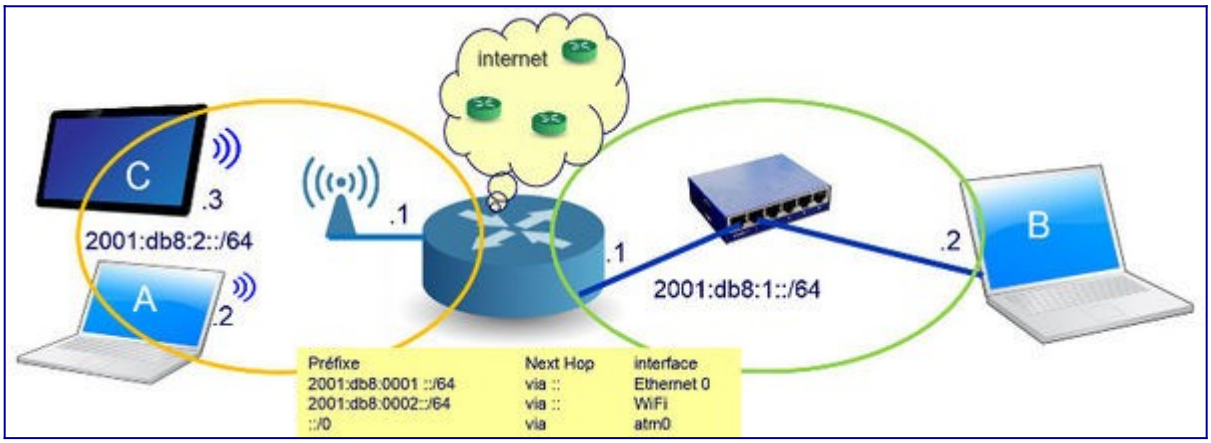


Figure 3 : Routeur par défaut.

Les hôtes n'ont pas à avoir la connaissance du routage. Aussi, pour ces nœuds, ils confient tous les paquets pour des remises indirectes au routeur local représenté dans la figure 4 par le routeur connecté à un fournisseur d'accès à Internet. Ceci se déclare dans la table de routage des hôtes par une simple route par défaut. La figure 4 montre la table de routage des hôtes avec la route par défaut. Dans notre exemple, le routeur par défaut est le routeur local de l'hôte A. Il n'y a pas de routeur intermédiaire entre l'hôte A et le routeur par défaut.

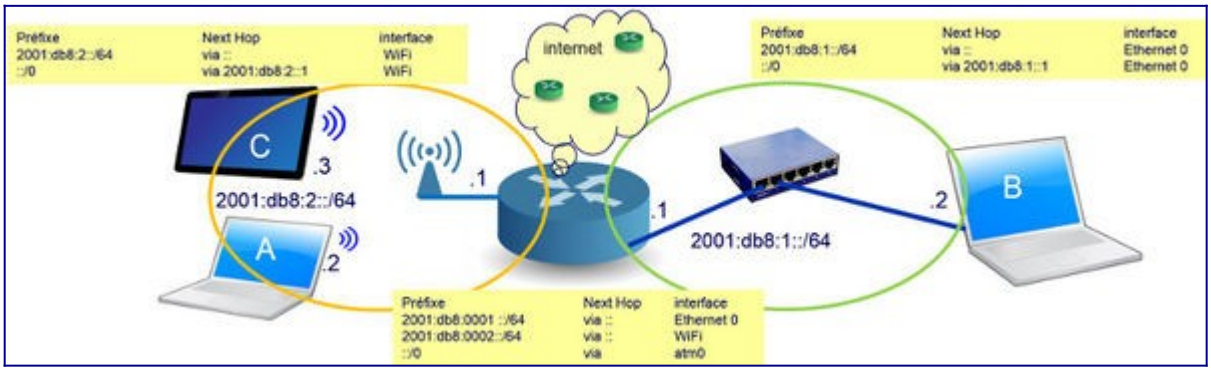


Figure 4 : Route par défaut dans les hôtes.

Choix de la route

La fonction de routage d'un nœud distingue une remise directe d'une remise indirecte à partir du test d'adjacence. Le test d'adjacence consiste à vérifier si le destinataire est directement accessible. Pour cela, le nœud va comparer le préfixe de la destination avec les préfixes des réseaux directement connectés. En cas de correspondance, le nœud peut réaliser une remise directe. Dans le cas contraire, c'est une remise indirecte. Il convient maintenant de choisir à quel routeur voisin il faut remettre le datagramme. L'adresse de destination du paquet est mise en correspondance avec les préfixes réseau contenus dans la table de routage. Le choix de la route se fait par la correspondance la plus longue (*best match* ou *longest match*) entre l'adresse de destination du datagramme et une destination de la table de routage. S'il n'y a aucune correspondance, il faut retenir la route par défaut.

Protocoles de routage

Comme pour IPv4, il faut faire la distinction entre deux grandes familles de protocoles de routage : les protocoles de routage interne (*Interior Gateway Protocols*, IGP) et les protocoles de routage externe (*Exterior Gateway Protocols*, EGP). La différence provient de la notion de **système autonome** (*Autonomous System*, AS), par la définition de la portée des échanges d'informations de routage des protocoles de routage. Ainsi, la propagation des préfixes réseaux internes à un AS se fait par un IGP, alors que les annonces de préfixes entre AS se fait par un EGP.

Pour connecter un site à l'Internet, il faut donc mettre œuvre des protocoles de routage interne et des protocoles de routage externe. Cette section traite des trois protocoles IGP suivants : RIPng (équivalent de RIPv2 pour IPv4), ISIS et OSPFv3 (équivalent d'OSPFv2 pour IPv4), ainsi que du protocole de routage externe BGP.

Les protocoles de routage interne visent à rendre automatique la configuration des tables de routage des routeurs à l'intérieur d'un même système autonome. Les routeurs déterminent le plus court chemin pour atteindre un réseau distant. Les protocoles de routage internes nécessitent une configuration minimale du routeur, notamment en ce qui concerne les annonces de routes initiées par ce routeur (exemple : réseaux directement accessibles par une interface du routeur, routes statiques...).

Deux types de protocoles de routage interne existent : les protocoles à vecteur de distance

(*distance vector*), et les protocoles à état de lien (*link state*). Les premiers génèrent des annonces de routeur transmises aux routeurs voisins, qui contiennent des informations de direction (les réseaux accessibles par le routeur) et de distance associées aux destinations annoncées (la métrique, qui peut être le nombre de routeurs à traverser, pour RIP, mais qui peut être un coût lié au débit, comme pour EIGRP). Pour les protocoles à état de lien, les annonces de routeur ne contiennent plus d'informations issues de la table de routage, mais des informations sur les liens auxquels sont connectés les routeurs (adresses IP, nature du lien, coût calculé souvent à partir du débit, etc.). Chaque routeur construit une base de données d'états de liens (*link state database*), qui permet de redessiner la topologie du réseau. Dans un second temps, un algorithme de recherche du plus court chemin permet à chaque routeur de construire sa table de routage, à partir de cette base de données.

RIPng ou RIP IPv6

RIPv2 (RFC 2453) est un algorithme à vecteur de distance, basé sur l'algorithme de Bellman-Ford et figure parmi les premiers algorithmes de routage interne utilisés dans l'Internet.

Les routeurs diffusent leurs tables de routage sur les liens auxquels ils sont connectés. Les autres routeurs modifient une route dans leur table si la **métrique** (le nombre de routeurs à traverser pour atteindre une destination) reçue est plus petite que celle déjà stockée dans la table. Si une annonce de route n'est pas présente dans la table, le routeur l'ajoute. Ces modifications sont à leur tour diffusées sur les autres réseaux auxquels sont connectés les routeurs. Elles se propagent donc sur l'ensemble du réseau à l'intérieur du système autonome. On montre que cet algorithme converge et, qu'en condition stable, aucune boucle n'est créée sur le réseau, c'est-à-dire qu'un paquet ne sera pas transmis indéfiniment de routeur en routeur sans jamais pouvoir atteindre sa destination.

Les tables sont émises périodiquement. Si un routeur tombe en panne, ou si le lien est coupé, les autres routeurs ne recevant plus l'information suppriment l'entrée correspondante de leur table de routage. RIPng est le premier protocole de routage dynamique proposé pour IPv6 (RFC 2080). RIPng est une simple extension à IPv6 du protocole RIPv2 d'IPv4. Il en hérite les mêmes limitations d'utilisation (maximum de 15 sauts par exemple).

ISIS

IS-IS (*Intermediate System to Intermediate System*) est un protocole de routage interne à état de lien. Il a été standardisé par l'ISO (ISO 10589). C'est un protocole qui opère au niveau de la couche de réseau et qui s'appuie sur une couche de liaison, contrairement à OSPF et RIP qui agissent au-dessus de la couche de transport. Cet élément mérite d'être signalé car cela rend ce protocole indépendant d'IP, que ce soit IPv4 ou IPv6. Ce protocole travaille sur deux niveaux de hiérarchie : les aires (niveau 1) et le *backbone* (niveau 2).

Un routeur IS-IS peut être :

- *level-1* (routage intra-aire),
- *level-2* (routage inter-aire),
- ou *level-1-2* (routage intra-aire et inter-aire).

Un routeur de niveau 1 n'a de voisins que dans son aire alors qu'un routeur de niveau 2 peut avoir des voisins dans une autre aire. Il n'y a pas d'aire de *backbone* (contrairement à OSPF). Le *backbone* est constitué de la réunion de tous les routeurs de *level-2*. Sur un réseau de type LAN, il y a élection d'un routeur désigné (DIS) qui a la charge de produire les annonces.

Afin de construire sa topologie, IS-IS utilise 3 types de messages :

- les messages HELLO permettant de construire les adjacences ;
- les messages LSP (*Link State Protocol*) permettant d'échanger les informations sur l'état des liens ;
- les messages SNP (*Sequence Number Packet*) permettant de confirmer la topologie.

Pour élaborer ces messages, IS-IS se base sur l'utilisation d'éléments d'informations indépendants appelés TLV (Type, Longueur, Valeur). Le message est ainsi constitué d'un en-tête suivi d'une liste de TLV. Chaque TLV véhicule une information propre, et est donc standardisée. L'exemple ci-dessous montre une TLV **Protocoles supportés** faisant partie d'un message HELLO, informant les voisins des protocoles supportés par l'émetteur du paquet :

- 0x81 0x02 0xcc 0x8e
    - Le premier octet donne le type de la TLV. Il s'agit ici du type 0x81, c'est-à-dire **Protocoles supportés**.
    - Le second octet donne la longueur en octets de la TLV : ici, les deux octets qui suivent.
    - Les autres octets composent la valeur de la TLV. Ici, nous avons deux octets indiquant des numéros de protocoles supportés (NLPID : *Network Layer Protocol IDentifier*): 0xCC pour IPv4 et 0x8E pour IPv6.

OSPFv3

Le troisième protocole de routage interne, basé sur l'algorithme du plus court chemin (SPF, *Shortest Path First*, ou algorithme de Dijkstra), s'appelle OSPF (*Open Shortest Path First*). Relativement plus complexe à mettre en œuvre que RIPng, il est beaucoup plus efficace dans les détections et la suppression des boucles dans les phases transitoires. Ce protocole est basé sur plusieurs sous-protocoles, dont un qui permet une inondation fiable du réseau. L'objectif d'OSPF est que chaque routeur possède une même copie de la carte de la topologie du réseau. À partir de là, chacun peut calculer simultanément le plus court chemin pour aller vers l'ensemble des destinations.

Pour réduire la durée des calculs, et surtout pour éviter un recalcul complet des routes à chaque changement de configuration, OSPF offre la possibilité de découper le réseau en aires. Une aire principale appelée *backbone* relie toutes les autres aires. Les réseaux trouvés dans une aire donnée sont envoyés aux autres aires par les routeurs qui sont en frontière d'aire.

OSPF a été adapté à IPv6 (RFC 2740) ; la version est passée de 2 à 3. La plupart des algorithmes implémentés dans OSPFv2 ont été réutilisés en OSPFv3. Bien évidemment, certains changements ont été nécessaires en vue de l'adaptation aux fonctionnalités d'IPv6.

BGP

BGP-4 est le protocole de routage externe actuellement utilisé pour le routage global de l'Internet IPv4 (le numéro de version 4, identique pour BGP et IP, est une pure coïncidence)[2]. Compte tenu de sa criticité, ce protocole est l'objet d'évolutions constantes. L'une d'entre elles est le RFC 4760 qui rend BGP-4 "multi-protocole" en introduisant la notion de famille d'adresses (ex. IPv4, IPv6, IPX...) et de sous-famille d'adresses (ex. *unicast*, *multicast*). Le RFC 2545 précise l'usage des extensions multi-protocoles pour le cas d'IPv6.

L'adaptation multi-protocole de BGP-4 est assez simple car elle ne concerne que les trois attributs dont le format dépend de l'adresse, soit :

- NLRI : *Network Layer Reachability Information* (suite de préfixes) ;
- NEXT_HOP : Adresse IP où il faut router les NLRI ;
- AGGREGATOR : Adresse IP du routeur qui a fait une agrégation de préfixes.

Pour réaliser pratiquement cette adaptation, BGP4+ introduit deux nouveaux attributs :

- MP_REACH_NLRI : *Multiprotocol Reachable NLRI*,
- MP_UNREACH_NLRI : *Multiprotocol Unreachable NLRI*,

qui indiquent que l'on annonce des informations de routage autres que les routes *unicast* IPv4. Ces attributs codent en premier le type de famille et de sous-famille d'adresses, puis les attributs dont le format est spécifique. Les autres attributs (comme le chemin d'AS *Autonomous System*) sont codés et annoncés sans changement.

Les mises en œuvre du RFC 4760 sont souvent appelées MP-BGP (pour faire référence à leur capacité de traitement des routes *multicast*) ou BGP4+ (pour faire référence à leur capacité de traitement de routes IPv6). Pour l'anecdote, le numéro de version du protocole n'a pas été modifié (en BGP-5 par exemple) car le passage de BGP-3 à BGP-4 rappelle trop de souvenirs douloureux à ceux qui l'ont mis en œuvre. Les numéros d'AS utilisés pour IPv4 servent aussi pour IPv6.

Conclusion

Cette activité vous a présenté le principe du routage IP et la table de routage. Cet élément essentiel de la couche réseau, présent dans chaque nœud de l'Internet, indique à un routeur qui a un paquet à acheminer à qui doit être remis ce paquet : à un routeur local, indiqué dans la table de routage comme prochain nœud associé à l'adresse de destination contenue dans la paquet, ou directement à la destination. La configuration correcte de la table de routage est donc importante aussi bien sur les routeurs que sur les hôtes.

Cette configuration peut se faire manuellement par l'administrateur du réseau : on parle alors de routage statique. Afin de pouvoir s'adapter à l'évolution du réseau, les tables de routage peuvent être mise à jour par des protocoles de routage permettant de propager les modifications de la topologie du réseau. Les tables de routage sont mises à jour automatiquement au fur et à mesure des changements dans le réseau. On parle alors de routage dynamique.

Références bibliographiques

1. ↕ Rubino, G. et Toutain, L. (2000). Techniques de l'ingénieur. Routage dans les réseaux Internet
2. ↕ Balakrishnan, H. et Feamster, N. (2005), Lecture notes. Interdomain Internet Routing.

Pour aller plus loin

RFC et leur analyse par S. Bortzmeyer : Vous pouvez approfondir vos connaissances sur les protocoles de routage en consultant les liens suivants :

RIPng :

- Article dans le livre "IPv6, Théorie et Pratique"
- RFC 2453 : RIP Version 2
- RFC 4822 : RIPv2 Cryptographic Authentication

ISIS :

- Article dans le livre "IPv6, Théorie et Pratique"
- ISO-IEC 8473 Information technology — Protocol for providing the connectionless-mode network service: Protocol specification

OSPF :

- Article dans le livre "IPv6, Théorie et Pratique"
- RFC 5340 : OSPF for IPv6 (Analyse par S.Bortzmeyer)
- RFC 7503 : OSPFv3 Autoconfiguration (Analyse par S.Bortzmeyer)

BGP :

- Article dans le livre "IPv6, Théorie et Pratique"
- RFC 2545 : Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing
- RFC 3849 : IPv6 Address Prefix Reserved for Documentation
- RFC 4760 : Multiprotocol Extensions for BGP-4 (Analyse par S.Bortzmeyer)
- RFC 5963: IPv6 Deployment in Internet Exchange Points (IXPs) (Analyse par S.Bortzmeyer)

Activité 23 : Le contrôle par ICMPv6 de l'acheminement des paquets IPv6

Introduction

Les réseaux physiques constituant Internet sont susceptibles d'être défaillants de manière plus ou moins fréquentes. Il peut s'agir d'une panne d'un équipement intermédiaire (routeur, commutateur) ou d'un lien (rupture d'un câble souterrain par une pelleteuse par exemple). L'Internet est normalement prévu pour exploiter des chemins alternatifs pour continuer à assurer l'acheminement des paquets, mais la détection et la gestion de ces incidents nécessitent des mécanismes de contrôle. Le réseau peut aussi détecter des problèmes d'acheminement liés à la nature de certains paquets : taille des données trop importante, en-tête comportant une adresse invalide ou ne correspondant à aucun équipement, etc.

Le protocole IPv6 prévoit donc un mécanisme permettant de signaler ces problèmes d'acheminement. Les messages correspondant à cette signalisation sont spécifiés dans le protocole ICMPv6 (RFC 4443). Il permet au réseau d'informer la source d'un paquet que celui ne peut pas être acheminé et en signifie la raison. C'est un mécanisme similaire au retour à l'expéditeur dans le réseau postal.

Format général d'un message ICMPv6

Les messages ICMPv6 sont encapsulés directement dans un paquet IPv6. Le protocole se voit attribuer le numéro 58 pour être représenté dans l'en-tête IPv6 comme prochain en-tête (champ *Next Header*). Le format général des messages ICMPv6 est donné par la figure 1. L'en-tête des messages ICMPv6 comporte 3 champs :

1. Le champ `Type` : il indique la nature du message ICMPv6 et donc, le format spécifique du message. Les messages ICMPv6 forment 2 groupes : un groupe pour les messages d'informations et un autre pour les messages d'erreurs. Les groupes sont identifiés par le bit de poids fort de ce champ. Les messages d'erreurs ont ce bit à zéro et donc, le champ `Type` prendra, pour ces messages, une valeur comprise entre 0 et 127. Les messages d'informations sont identifiés par un champ `Type` dont la valeur est comprise entre 128 et 255.
2. Le champ `Code` s'interprète dans le contexte du type de message. Il est utilisé pour ajouter une granularité plus fine au type.
3. Le champ `Checksum` sert à vérifier l'intégrité du message ICMP. Il porte aussi bien sur l'en-tête du message que sur sa charge utile.

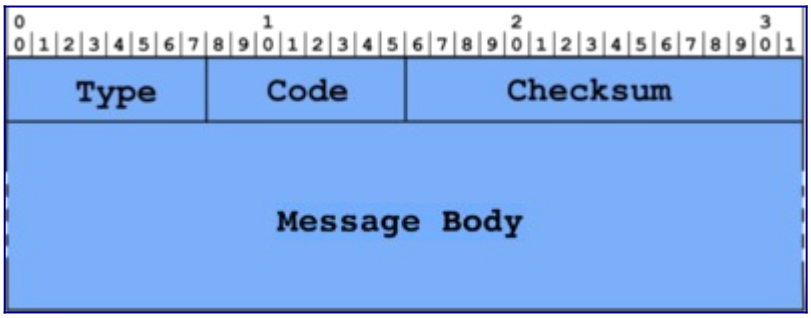


Figure 1 : Format d'un message ICMPv6.

La charge utile du message ICMPv6 (*message body*) est relative au contexte fonctionnel :

- dans le cas des messages de compte rendu d'erreurs, elle contient le paquet IPv6 ayant provoqué l'erreur. Pour éviter d'avoir à fragmenter, la longueur du message ICMPv6 est limitée à 1 280 octets. Par conséquent, le contenu du paquet IPv6 renvoyé peut être tronqué ;
- pour les messages de découverte du voisinage, la charge utile est composée de différentes options selon qu'il s'agisse d'une sollicitation de voisin ou d'une annonce de voisin ;
- les messages de test d'accessibilité embarquent des données de taille et de contenu quelconque.

Les messages sont spécifiés dans différents RFC. Cependant, la liste complète des types de messages et les différents paramètres associés sont regroupés par l'IANA (*Internet Assigned Number Authority*)[1]. Le tableau 1 présente les valeurs et les codes définis dans le RFC 4443. Ce qu'il faut noter, c'est que les valeurs de type inférieures à 128 sont pour les messages d'erreurs et qu'à partir de 128, ce sont des messages d'informations.

Type	Code	Signification
		Message de compte rendu d'erreur
1		Destination inaccessible :
	0	Aucune route vers la destination.
	1	La communication avec la destination est administrativement interdite.
	2	Hors portée de l'adresse source.
	3	L'adresse est inaccessible.
	4	Le numéro de port est inaccessible.
	5	L'adresse source est filtrée par un *firewall*.
	6	L'adresse destination est rejetée.
2		Paquet trop grand.
3		Délai expiré :

	0	Limite du nombre de sauts atteinte.
	1	Temps de réassemblage dépassé.
4		Erreur de paramètre :
	0	Champ d'en-tête erroné.
	1	Champ d'en-tête suivant non reconnu.
	2	Option non reconnue.
		Message d'information
128		Demande d'écho
129		Réponse d'écho

Tableau 1 : Messages ICMPv6 décrit dans le RFC 4443.

Test d'accessibilité d'un nœud

Les test d'accessibilité vise à vérifier qu'un nœud est joignable par l'adresse IPv6 de son interface. Autrement dit, depuis un nœud, on vérifie qu'il existe une connectivité entre deux nœuds de l'Internet. Ce test est couramment utilisé, notamment à l'aide de l'outil nommé `ping`.Le principe de fonctionnement est le même que pour IPv4. Un message de demande d'écho (*echo request*), message ICMPv6 type 128 est envoyé vers le nœud dont on veut tester la connectivité. Ce dernier répond par le message "réponse d'écho" (*echo reply*), message ICMPv6 de type 129. Le format de ces deux messages est présenté par la figure 2.

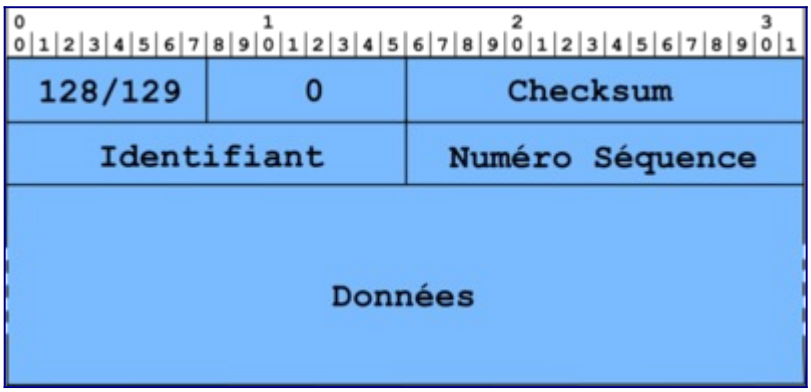


Figure 2 : Format du message d'écho.

Les réponses sont identifiées par le champ `identifiant`. Ainsi, la réponse est rapprochée de la requête. Ceci est particulièrement utile quand plusieurs commandes `ping` sont exécutées simultanément sur la machine. Le champ `numéro de séquence` complète le mécanisme de rapprochement de la réponse à la requête et va servir à mesurer la durée d'aller et retour dans le cas où les demandes sont émises en continu et que le délai de propagation est élevé. La réponse doit toujours contenir les mêmes valeurs que la requête pour ces deux champs. Le

champ `données` permet d'augmenter la taille du message (et donc la durée de transmission) pour les mesures.

Pour illustrer le test d'accessibilité, observons l'échange suivant : Le nœud nommé "Uma" teste la connectivité du nœud "Ganesha" via la commande `ping6`. La commande entrée sur Uma est la suivante :

```
uma# ping6 ganesha
trying to get source for ganesha
source should be 2001:db8:12:3:a00:20ff:fe0a:aa6d
PING ganesha (2001:db8:12:3::3): 56 data bytes
64 bytes from 2001:db8:12:3::3: icmp6_seq=0 ttl=255 time=5.121 ms
```

Les messages ICMPv6 obtenus sont les suivants.

- Le nœud "Uma", qui est l'initiateur, envoie un message *ICMPv6 Demande d'écho*. La trace 1 montre un paquet IPv6 contenant un message *ICMPv6 Demande d'écho* (en bleu dans la trace).

```
Version : 6        Classe : 0x00     Label : 000000 Longueur : 64 octets
(0x0040)
Protocole : 58 (0x3a, ICMPv6)
Nombre de sauts : 64 (0x40)
Source : 2001:db8:12:3:a00:20ff:fe0a:aa6d (Uma)
Desti. : 2001:db8:12:3::3 (Ganesha)
ICMPv6
Type : 128 (0x80, Demande d'écho)       Code : 0 Checksum : 0xcfe8
Identificateur : 0x0e02 Numéro de séquence : 0x0001
Données : b6e0 f056 ...

0000:  6000 0000 0040 3a40 2001 0db8 0012 0003
0010:  0a00 20ff fe0a aa6d 2001 0db8 0012 0003
0020:  0000 0000 0000 0003|80|00|cfe8|0e02|0001|
0030:  b6e0 f056 d947 0700 0809 0a0b 0c0d 0e0f
0040:  1011 1213 1415 1617 1819 1a1b 1c1d 1e1f
0050:  2021
```

Trace 1 : Message ICMPv6 Demande d'écho.

- Le destinataire du message "Demande d'écho", qui est Ganesha sur la figure 10, acquitte ce message en retournant un message *ICMPv6 Réponse d'écho* (voir la trace 2).

```
Version : 6        Classe : 0x00     Label : 000000 Longueur : 64 octets
(0x0040)
Protocole : 58 (0x3a, ICMPv6)
Nombre de sauts : 64 (0x40)
Source : 2001:db8:12:3::3 (Ganesha)
Desti. : 2001:db8:12:3:a00:20ff:fe0a:aa6d (Uma)
ICMPv6
Type : 129 (0x81, Réponse d'écho)       Code : 0 Checksum : 0xcee8
Identificateur : 0x0e02 Numéro de séquence : 0x0001
Données : b6e0 f056 ...

0000:  6000 0000 0040 3a40 2001 0db8 0012 0003
0010:  0000 0000 0000 0003 2001 0db8 0012 0003
0020:  0a00 20ff fe0a aa6d|81|00|cee8|0e02|0001|
0030:  b6e0 f056 d947 0700 0809 0a0b 0c0d 0e0f
```

```
0040:  1011 1213 1415 1617 1819 1a1b 1c1d 1e1f
0050:  2021
```

Trace 2 : Message ICMPv6 Réponse d'écho.

Messages ICMPv6 de rapport d'erreur

ICMPv6 permet de signaler, à l'émetteur d'un paquet, un problème dans son acheminement ou dans sa réception, par des messages de rapport d'erreur. Lorsqu'une machine émet un paquet, si une erreur est détectée par le destinataire ou par tout routeur intermédiaire le long du chemin vers le destinataire, alors l'élément qui détecte l'erreur renvoie à l'émetteur un rapport sous la forme d'un message ICMPv6. Le type du message et son code définissent précisément l'erreur détectée. L'extrait, jusqu'à concurrence de 1280 octets du paquet en défaut, est incluse pour permettre l'analyse de l'erreur. L'exemple ci-dessous illustre un message ICMP *Paquet trop grand*, généré par un routeur intermédiaire dès qu'un datagramme ne peut être retransmis en raison de la limitation de la MTU sur son interface de sortie.

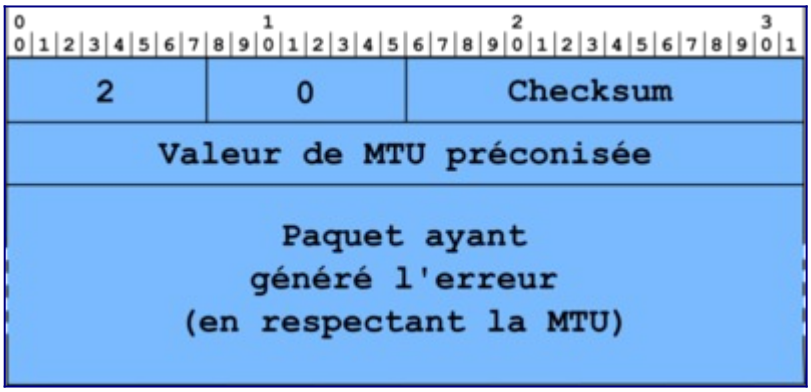


Figure 4 : Format du message ICMPv6 Paquet trop grand.

Différentes erreurs peuvent être signalées par ICMPv6. Les cas les plus courants sont :

- destination inaccessible (type 1), la raison est précisée par le champ Code ;
- paquet trop grand (type 2) ;
- délai expiré (type 3) ;
- erreur de paramètre (type 4).

Chaque cas d'erreur est défini par un message ICMP. Nous allons voir les cas d'erreurs rapportées par ICMPv6.

Destination inaccessible

Un message ICMP *Destination Unreachable* est généré dès qu'un datagramme ne peut être traité. Le format de ce message est présenté par la figure 3.

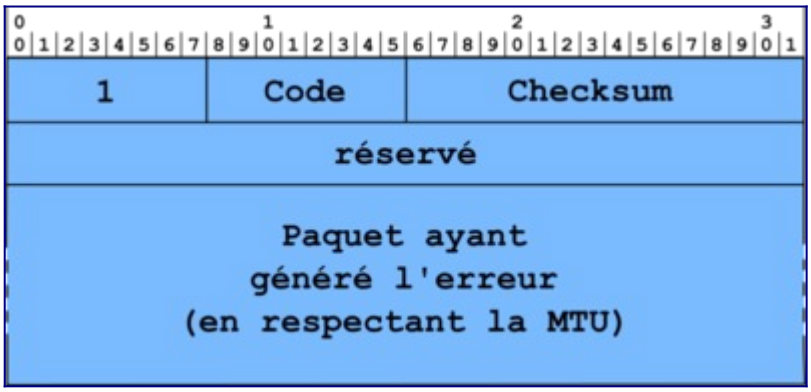


Figure 3 : Format du message de Destination inaccessible.

Ce message est émis par un routeur quand le paquet ne peut pas être transmis pour l'une des raisons suivantes :

- le routeur ne trouve pas dans ses tables la route vers la destination (`code` = 0) ;
- le franchissement d'un pare-feu (*Firewall*) est interdit ("raison administrative", `code` = 1) ;
- l'adresse destination ne peut être atteinte avec l'adresse source fournie. Par exemple, si le message est adressé à un destinataire hors du lien, l'adresse source ne doit pas être une adresse "lien-local" (`code` = 2) ;
- toute autre raison comme, par exemple, la tentative de routage d'une adresse locale au lien (`code` = 3) ;
- le destinataire peut aussi émettre un message ICMPv6 de ce type quand le port destination contenu dans le paquet n'est pas affecté à une application (`code` = 4) ;
- le paquet a été rejeté à cause de son adresse source (`code` = 5) ;
- la route vers la destination conduit à un rejet du paquet (`code` = 6).

Le champ de données contient tout ou partie du datagramme IP qui a occasionné ce message d'erreur.

Paquet trop grand

Si un routeur ne peut pas relayer le datagramme IP parce que celui-ci a une taille supérieure à la MTU (*Maximum Transmission Unit*) du lien de sortie, il émet le message d'erreur *Packet Too Big*. Les routeurs en IPv6 ne sont plus habilités à effectuer la fragmentation. Le format du message est représenté par la figure 4.

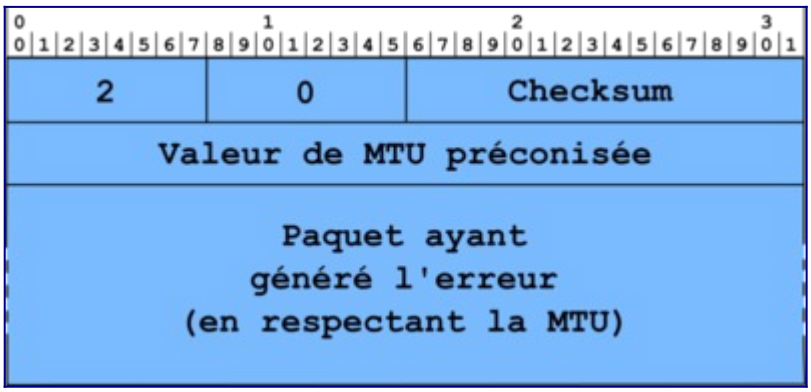


Figure 4 : Format du message ICMPv6 Paquet trop grand.

Ce message ICMPv6 est utilisé par le protocole de découverte de la MTU pour trouver la taille optimale des paquets IPv6 afin qu'ils puissent traverser les routeurs. Ce mécanisme, spécifié par le RFC 8201, est décrit dans la séquence 2. Ce message contient la taille du MTU acceptée par le routeur pour que la source puisse efficacement adapter la taille des données. Ce champ manquait cruellement dans les spécifications initiales de IPv4, ce qui compliquait la découverte de la taille maximale des paquets utilisables sur l'ensemble du chemin. Pour IPv4, le RFC 1191 proposait déjà une modification du comportement des routeurs pour y inclure cette information. À noter que le RFC 4443 indique que ce message n'est pas produit dans le cas d'une communication multicast.

Délai expiré

Quand un routeur relaie un datagramme, il décrémente la valeur du `nombre de sauts` (*Hop Limit*) de 1. Ce champ, dans l'en-tête IPv6, limite la durée de vie d'un paquet dans le réseau. La durée de vie est exprimée en nombre de routeurs traversés (ou sauts). Si un routeur reçoit un datagramme avec un `nombre de sauts` de 1, la décrémentation amène la valeur de ce champ à zéro. Le routeur supprime le datagramme et en avertit la source à l'aide du message Délai expiré (*Time Exceeded*) (voir la figure 5). Le signalement de cette erreur peut indiquer une boucle dans le réseau au niveau du routage ou que l'émetteur a positionné une valeur de `nombre de sauts` trop faible. Le dernier cas peut provenir d'un paquet émis par la commande `traceroute`. Cette commande vise à déterminer le chemin pris par les datagrammes [2].

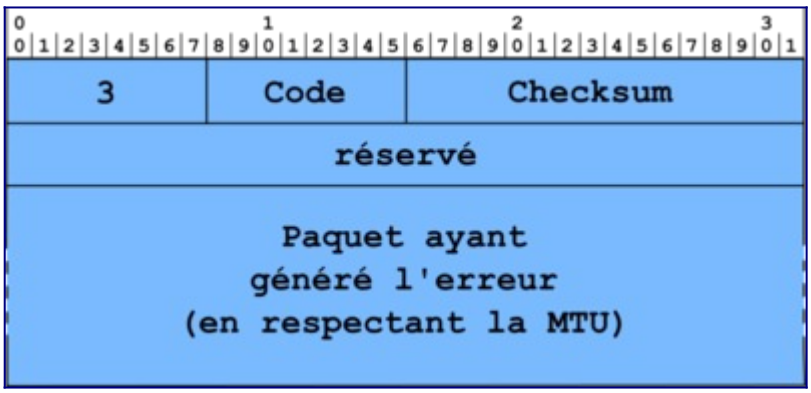


Figure 5 : Format du message de délai expiré.

Le message ICMPv6 Délai expiré indique que le paquet a été rejeté :

- soit par un routeur intermédiaire parce que le champ `nombre de sauts` a atteint 0 (`code` = 0) ;
- soit par la destination parce qu'un fragment s'est perdu et le temps alloué au réassemblage a été dépassé (`code` = 1).

Erreur de paramètre

Le message *Parameter problem* est émis quand un paquet IPv6 ne peut être traité par un nœud

du fait d'un problème dans l'en-tête du paquet ou dans ses extensions d'en-tête. Le paquet IPv6 est supprimé mais le nœud envoie à la source le message ICMPv6 dont le format est présenté par la figure 6.

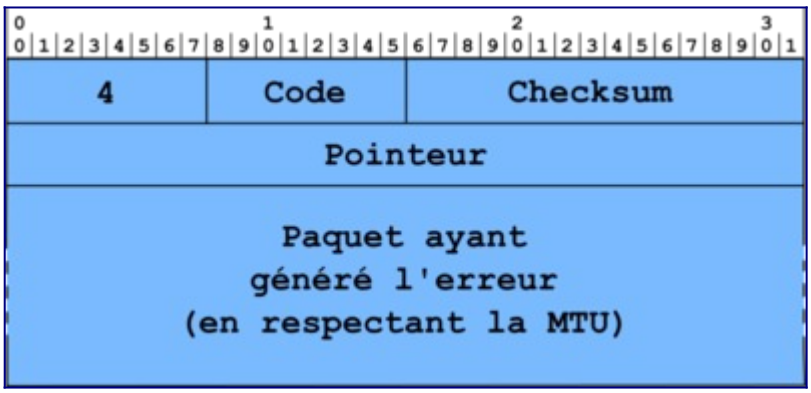


Figure 6 : Format du message Erreur de paramètre.

Le champ `type` prend la valeur 4. Le champ `code` révèle la cause de l'erreur :

- la syntaxe de l'en-tête n'est pas correcte (`code` = 0) ;
- le numéro en-tête suivant n'est pas reconnu (`code` = 1) ;
- une option de l'extension (par exemple "proche en proche" ou "destination") n'est pas reconnue et le codage des deux bits de poids fort oblige à rejeter le paquet (`code` = 2).

Le champ `pointeur` indique l'octet où l'erreur est survenue dans le paquet retourné. Le message contient tout ou partie du paquet IPv6 qui a occasionné le message lui-même.

Attention au filtrage d'ICMPv6

Contrairement à une pratique couramment répandue en IPv4, il ne faut jamais filtrer l'ensemble des messages ICMPv6 en entrée d'un réseau (en particulier le message "Paquet trop grand"). Le filtrage peut introduire des effets néfastes sur le fonctionnement du réseau[3]. Supposons que la MTU entre un client et un serveur soit limitée à 1480 octets à cause d'un tunnel IPv6 dans IPv4 et qu'un serveur filtre les messages ICMPv6 (en particulier "Paquet trop grand"). Le client et le serveur vont échanger des petits paquets pour ouvrir la connexion TCP (SYN, SYN ACK, ACK), puis le client va envoyer une requête HTTP qui est elle-même de petite taille. Le serveur va répondre en envoyant une page complète. Si le paquet a une longueur de 1500 octets, il va être détecté trop grand par le routeur à l'entrée du tunnel. Ce dernier va rejeter le paquet et envoyer au serveur un message ICMPv6 indiquant que le paquet est trop grand. Si le pare-feu du serveur le filtre, le serveur ne connaitra jamais la taille de la MTU adaptée au chemin qui mène à ce client. Il ne pourra pas adapter la taille de ses paquets. Tous les paquets de 1500 octets seront inexorablement supprimés. Le client devient alors quasiment injoignable et c'est le service de connectivité de la couche "réseau" qui est cassé. Dans cette situation, on se trouve dans une situation où certains paquets passent (ouvertures de connexion, ping, sessions SSH...) et d'autres sont bloqués. Diagnostiquer ce type d'erreur est particulièrement délicat.

Comme ICMPv6 est essentiel au fonctionnement d'IPv6, le RFC 4890 donne les bonnes

pratiques pour filtrer correctement les messages ICMPv6. L'idée est de laisser passer les messages ICMPv6 qui sont indispensables au fonctionnement du réseau et de jeter ceux qui introduisent un risque d'insécurité.

Pour aller plus loin

RFC et leur analyse par S. Bortzmeyer :

- RFC 1191 Path MTU Discovery
- RFC 2464 Transmission of IPv6 Packets over Ethernet Networks
- RFC 4443 Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification Analyse
- RFC 4620 IPv6 Node Information Queries
- RFC 4890 Recommendations for Filtering ICMPv6 Messages in Firewalls
- RFC 8201 Path MTU Discovery for IP version 6 Analyse
- RFC 8883 ICMPv6 Errors for Discarding packets Due to Processing Limits Analyse

Annexe 1 : Autres fonctions d'ICMPv6

Le contenu de cette annexe complète l'activité 23 en :

- introduisant MLD, le protocole de gestion des inscriptions aux groupes de *multicast* et les messages ICMPv6 associés ;
- introduisant des fonctions expérimentales ;

Gestion de groupes multicast sur le lien local

Pour offrir un service de distribution multicast, deux composants sont nécessaires : un protocole de gestion de groupes multicast et un protocole de routage multicast[4]. Le protocole de gestion de groupes multicast réalise la signalisation entre l'hôte et son routeur local. Le protocole de routage multicast vise à échanger les informations entre les routeurs afin qu'un arbre de distribution multicast soit construit.

En IPv6, MLD (*Multicast Listener Discovery*) sert, pour un hôte, à indiquer les groupes auxquels il souhaite souscrire. MLD est donc un protocole de gestion de groupes. Ainsi, un routeur de bordure IPv6 va pouvoir découvrir la présence de récepteurs multicast (qualifiés de *listeners*) sur ses liens directement attachés, ainsi que les adresses multicast concernées. MLD est un protocole asymétrique qui spécifie un comportement différent pour les hôtes (les *listeners*) et les routeurs. Toutefois, pour les adresses multicast sur lesquelles un routeur lui-même est récepteur, il doit exécuter les deux parties du protocole. Ceci implique notamment de répondre à ses propres messages de demande. En effet, les routeurs doivent constituer une liste des adresses multicast pour lesquelles il a un ou plusieurs récepteurs sur leur lien local. Aussi, un des récepteurs sur un lien envoie un message de rapport d'abonnement aux groupes auxquels il souhaite recevoir les messages. L'objectif est, par des communications multicast sur le lien, que le routeur local arrive à faire la liste complète des groupes multicast pour lesquels il doit

relayer le trafic localement.

MLD est une fonction d'ICMPv6 ; aussi, les messages MLD sont des messages ICMPv6. Les messages pour MLD sont envoyés avec :

- une adresse source IPv6 lien-local ;
- le champ `nombre de sauts` fixé à 1 ;
- l'option `IPv6 Router Alert` activée en ajoutant l'extension d'en-tête *Hop-by-Hop* correspondante.

Cette dernière option est nécessaire afin de contraindre les routeurs à examiner les messages MLD envoyés à des adresses multicast par lesquelles les routeurs ne sont pas intéressés. La version d'origine du protocole MLD [RFC 2710] (que nous appellerons également MLDv1) présente les mêmes fonctionnalités que le protocole IGMPv2 en IPv4. MLDv2 a été proposé par le RFC 3810 dans lequel, en plus du groupe, le récepteur peut indiquer la source. MLDV2 est une adaptation de IGMPv3 d'IPv4 à IPv6.

Format des messages pour MLD

Le format générique d'un message MLD est donné par la figure 7. Les différents types de messages ICMPv6 pour MLD sont indiqués par le tableau 2. On distingue trois types de messages pour MLD.

1. Le premier type (`type` = 130) concerne le recensement des récepteurs multicast selon plusieurs méthodes : (i) recensement général émis à l'adresse de diffusion générale sur le lien (`FF02::1`), (ii) recensement spécifique pour une adresse multicast ; l'adresse de destination est l'adresse multicast du groupe en question. Le message de requête d'abonnement (*Multicast Listener Query*) est émis par le routeur.
2. Le second type de message (`type` = 131) vise à obtenir un rapport d'abonnement multicast (*Multicast Listener Report*). Ce message est émis par le récepteur multicast. L'adresse de destination est l'adresse multicast du groupe en question. Avec MLDv2, le rapport d'abonnement à un groupe multicast a été complété par la possibilité de limiter la réception au trafic émis par certaines sources. Le trafic des sources non indiquées est alors non reçu. Cette restriction sur la source s'effectue par un message spécifique (`type` = 143).
3. Enfin, le troisième type de message (`type` = 132) va servir à un récepteur pour annoncer une résiliation d'abonnement (*Multicast Listener Done*) à un groupe. Ce message est émis à l'adresse du groupe multicast "tous les routeurs du lien local" (`FF02::2`).

Les champs des messages pour MLD ont la signification suivante :

- `Type` : prend la valeur 130, 131 ou 132 ;
- `Code` : mis à zéro par l'émetteur et ignoré par les récepteurs ;
- `Checksum` : celui du protocole ICMPv6 standard, couvrant tout le message MLD auquel s'ajoutent les champs du pseudo-en-tête IPv6 ;
- `Délai maximal de réponse` :
    - utilisé seulement dans les messages de recensement, il exprime le retard maximal

autorisé (en millisecondes) pour l'arrivée des rapports d'abonnement,
    - dans les messages de rapport ou de résiliation d'abonnement, ce champ est mis à zéro par l'émetteur et ignoré par les récepteurs ;
- `réservé` : champ non utilisé et mis à zéro par l'émetteur et ignoré par les récepteurs ;
- `Adresse multicast` :
    - pour un message de recensement général, ce champ est mis à zéro,
    - pour un message de recensement spécifique, il contient l'adresse multicast en question,
    - pour les messages de rapport et de résiliation d'abonnement, le champ contient l'adresse multicast sur laquelle l'hôte souhaite écouter ou cesser d'écouter.

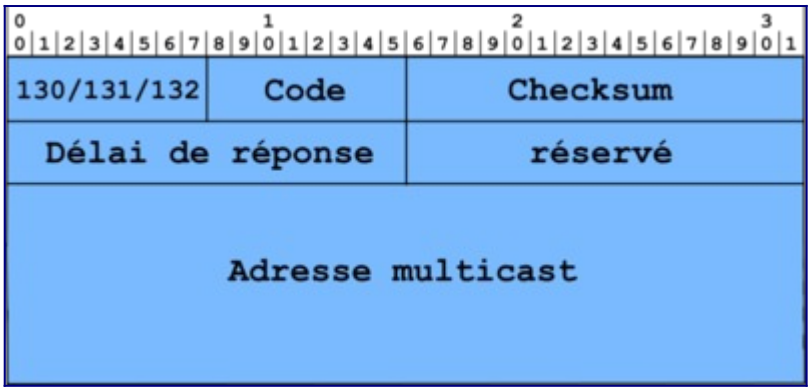


Figure 7 : Format générique d'un message ICMPv6 pour MLD.

Type	Code	Signification
Gestion des groupes multicast		
130		Requête d'abonnement
131		Rapport d'abonnement
132		Fin d'abonnement
143		Rapport d'abonnement MLDv2

Tableau 2 : Messages ICMPv6 pour MLD

Principe de MLD

Le routeur envoie régulièrement des messages de recensement général à l'adresse de multicast `FF02::1`. Cette adresse équivaut à l'adresse de diffusion sur un lien. Pour éviter que le routeur reçoive plusieurs réponses pour un même groupe, les récepteurs ne répondent pas immédiatement. Pour cela, les récepteurs arment un temporisateur pour chaque adresse multicast qui les concerne. Si le récepteur entend une réponse équivalente à la sienne, Il désarme le temporisateur. Sinon, à l'expiration du temporisateur, le récepteur envoie un rapport d'abonnement à l'adresse multicast du groupe. Avec ce système de temporisateurs, les récepteurs peuvent surveiller les rapports des autres récepteurs sur le lien et ainsi minimiser le trafic MLD.

Les changements d'état des récepteurs sont notifiés par des messages non sollicités. Un message non sollicité est un message émis à l'initiative d'un récepteur d'un groupe multicast ; contrairement au recensement, où c'est le routeur local qui prend l'initiative de l'échange. Les récepteurs peuvent envoyer des messages non sollicités pour les cas suivants :

- pour souscrire à une adresse multicast spécifique ;
- pour une résiliation rapide : le récepteur envoie un message de résiliation d'abonnement à l'adresse multicast de "tous les routeurs du lien local" (`FF02::2`). Le routeur répond avec un message de recensement spécifique à l'adresse en question. S'il n'y a plus de récepteur pour répondre à ce recensement, le routeur efface l'adresse multicast de sa table de routage.

Pour cesser d'écouter sur une adresse multicast, le récepteur peut simplement ne plus répondre aux messages de recensement du routeur. S'il est le seul récepteur de cette adresse multicast sur le lien, après un certain temps, l'état du routeur concernant cette adresse expire. Le routeur arrêtera de faire suivre les paquets multicast envoyés à l'adresse en question, s'il s'avère que le récepteur était le dernier concerné par l'adresse multicast sur le lien.

À noter qu'il est possible d'avoir plusieurs routeurs multicast sur le même lien local. Dans ce cas, un mécanisme d'élection est utilisé pour choisir le routeur recenseur. Celui-ci sera le seul responsable pour l'envoi des messages de recensement.

Fonctions autres et expérimentales

Pour être complet, nous pouvons signaler que les messages ICMPv6 servent aussi pour des fonctions expérimentales. Le tableau 4 indique les types de messages associés à ces fonctions. Nous ne détaillerons pas ici ces fonctions, limitées à des usages très spécifiques. Le lecteur curieux est invité à consulter les RFC associés.

Type	Code	Signification
Renumérotation des routeurs (expérimental, RFC 2894)		
138		Renumérotation des routeurs :
	0	Commande
	1	Résultat
	255	Remise à zéro du numéro de séquence
Recherche d'information sur un nœud (expérimental, RFC 4620)		
139		Demande d'information
140		Réponse
Mobilité (RFC 6275)		

144	Découverte d'agent mère (requête)
145	Découverte d'agent mère (réponse)
146	Sollicitation de préfixe mobile
147	Annonce de préfixe mobile
	Mobilité (expérimental, RFC 4065)
150	Protocoles de mobilité expérimentaux, tels que Seamoby

Tableau 4 : Fonctions expérimentales s'appuyant sur ICMPv6

Adresse physique de la source/cible

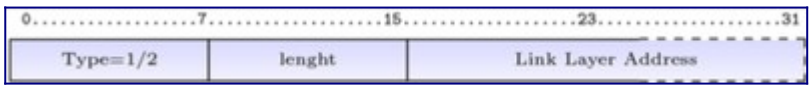


Figure : Format de l'option adresse physique source/cible.

La figure "Format de l'option adresse physique source/cible" donne le format de ces options. Le type 1 est réservé à l'adresse physique de la source et le type 2 à l'adresse de la cible.

Le champ «longueur» est la taille en mots de 64 bits de l'option. Dans le cas d'une adresse MAC, d'une longueur de 6 octets, il contient donc la valeur 1.

Le RFC 2464 définit le format pour les adresses MAC-48 utilisés dans les réseaux Ethernet et Wi-Fi. Le RFC 4944 définit le format pour les MAC-16 et MAC-64 utilisés dans les réseaux de capteurs reposant sur la norme IEEE 802.15.4.

Information sur le préfixe

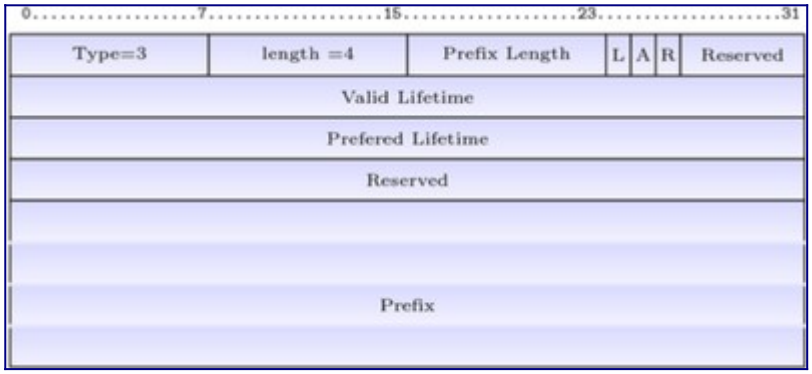


Figure : Format de l'option information sur le préfixe.

Cette option contient les informations sur le préfixe pour permettre une configuration automatique des équipements. Cette option sera présentée en détail dans l'activité d'autoconfiguration.

En-tête redirigée

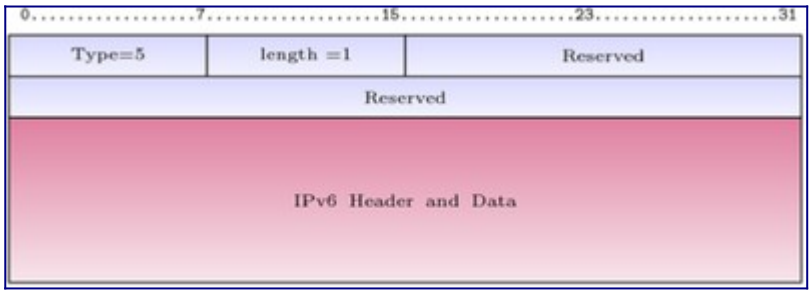


Figure : Format de l'option en-tête redirigée.

Cette option est utilisée par le message d'indication de redirection. Elle permet d'encapsuler les premiers octets du paquet IPv6 qui a provoqué l'émission de ce message comme dans le cas des messages ICMPv6 d'erreur.

Le type vaut 4 et la taille de cette option ne doit pas conduire à un paquet IPv6 dépassant 1280 octets (cf. figure Format de l'option en-tête redirigée). Par contre le paquet doit contenir le maximum d'information possible.

MTU

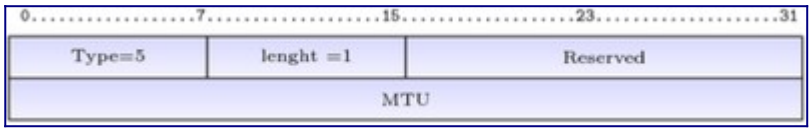


Figure : Format de l'option MTU.

Cette option permet d'informer les équipements sur la taille maximale des données pouvant être émises sur le lien. La figure "Format de l'option MTU" donne le format de cette option. Il n'est pas nécessaire de diffuser cette information si l'équipement utilise toujours la taille maximale permise. Par exemple, sur les réseaux Ethernet, les équipements utiliseront la valeur 1 500. Par contre pour les réseaux anneau à jeton ou FDDI, il est souvent nécessaire de préciser si les équipements doivent utiliser la valeur maximale permise ou une valeur inférieure pour autoriser l'utilisation de ponts.

Le champ type vaut 5 et le champ longueur 1.

Pour aller plus loin

RFC et leur analyse par S. Bortzmeyer

- RFC 2710 Multicast Listener Discovery (MLD) for IPv6
- RFC 2894 Router Renumbering for IPv6
- RFC 3122 Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification
- RFC 3971 SEcure Neighbor Discovery (SEND) Analyse
- RFC 3810 Multicast Listener Discovery Version 2 (MLDv2) for IPv6
- RFC 4065 Instructions for Seamoby and Experimental Mobility Protocol IANA Allocations
- RFC 6275 Mobility Support in IPv6

Référence bibliographique

1. ↕ IANA. *Internet Control Message Protocol* version 6 (ICMPv6) Parameters
2. ↕ Wikipédia Principe de l'utilitaire traceroute
3. ↕ Kline, E. and Townsley, M.; Google IPv6 Center. ICMPv6 is Non-Optional
4. ↕ Sébastien LOYE. (2005). Techniques de l'ingénieur. ref TE7527. Le multicast IP : principes et protocoles

Activité 24 : La taille des paquets IPv6

Introduction

La notion de datagramme, sur laquelle repose le protocole IP, implique un découpage des données provenant de la couche transport afin que ces données puissent être transportées. En effet, un paquet est une unité de transfert de taille variable et limitée. Les paquets ainsi constitués sont ensuite encapsulés dans des trames avant d'être transmis sur le support physique. Aussi, les datagrammes sont transmis dans des systèmes de transmissions mettant en œuvre des supports physiques qui peuvent être de natures différentes. Ces systèmes ont une taille de trame de valeur maximale variable. La gestion de la taille du paquet sur le chemin est donc nécessaire pour permettre le transfert des données de manière indépendante du support d'un bout à l'autre de l'Internet. Ce problème de la taille du paquet à transférer existe tout aussi bien en IPv4 qu'en IPv6. La solution pour régler le problème est différente entre IPv4 et IPv6 [1].

Dans cette activité vous étudierez les éléments qui vont définir la taille du paquet IPv6 pour un transfert de données. Vous allez commencer par définir les notions de MTU et PMTU qui posent les contraintes de la taille maximale du paquet. Vous verrez comment, en IPv6, découvrir la taille maximale du paquet pour un transfert. Le protocole IPv6 gère aussi la fragmentation. Vous aurez les explications du "quand" et du "comment" la fragmentation est mise en œuvre. Enfin, la notion de paquet IPv6 jumbogramme sera présentée.

MTU et PMTU

La couche de réseau doit placer les messages provenant de la couche de transport dans des paquets. Un message de transport se définit comme un bloc de données provenant de l'application complété par un en-tête de transport. Les paquets ainsi constitués sont ensuite encapsulés dans des trames adaptées au support physique utilisé. Cette combinaison de format de trame et de support physique constitue ce que l'on pourra qualifier de système de transmission. Chaque système de transmission définit une taille maximale du champ de données de la trame. Cette taille du champ de données de la trame limite la taille maximale du paquet et par conséquent des données que peut contenir un paquet. La taille maximale du champ de données de la trame est appelée **MTU** (*Maximum Transmission Unit*). Comme exemple, on peut citer les systèmes de transmission Ethernet avec une MTU de 1500 octets, ou Wifi avec une MTU de 2304 octets. La figure 1 illustre les encapsulations successives qui se produisent pour la transmission de données sur Ethernet. Ce qu'il faut retenir de la notion de MTU, c'est qu'elle donne la taille maximale du paquet IP pouvant être transmis par un hôte sur le lien sur lequel il est raccordé.

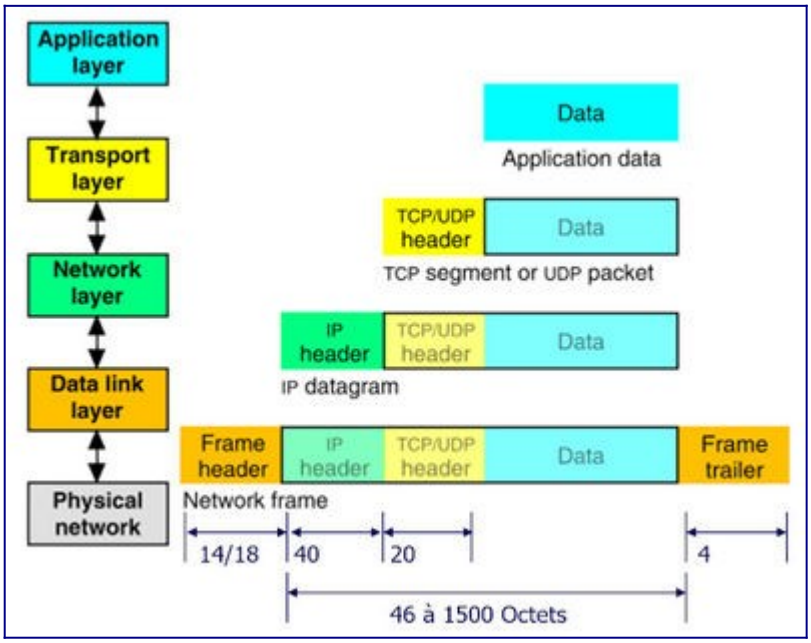


Figure 1 : Les encapsulations.

Un paquet IP peut être amené à transiter par des systèmes de transmission de natures différentes, chacun imposant une MTU différente. Pour pouvoir parcourir son chemin jusqu'à sa destination, le paquet doit donc avoir une taille inférieure ou égale à la plus petite taille autorisée par l'ensemble des liens traversés (cf. figure 2). Cette taille est appelée **PMTU** (*Path Maximum Transmission Unit*) ou taille de la MTU sur la route entre deux hôtes. Littéralement, la PMTU c'est l'unité de transmission maximum sur le chemin. Elle indique la taille maximale du paquet pour arriver à l'hôte de destination sans jamais excéder la MTU de chaque lien traversé. Ainsi, un paquet émis à la taille PMTU de sa route n'aura pas de problème d'acheminement lié à une impossibilité d'encapsuler ce paquet dans une trame quelque part sur sa route.

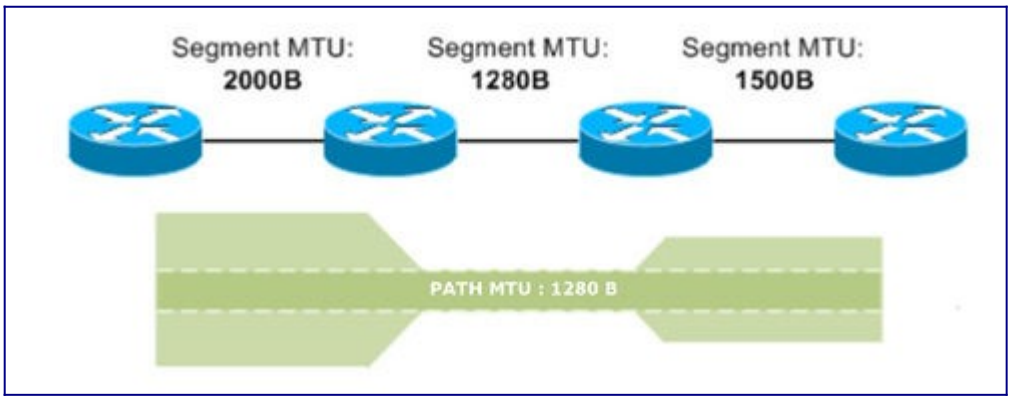


Figure 2 : Path MTU.

Pour des considérations d'efficacité, il est généralement préférable que les informations échangées entre nœuds soient contenues dans des datagrammes de taille maximale. Les paquets de petite taille ont pour effet d'augmenter le coût des en-têtes par rapport aux données transportées, et d'augmenter le nombre de paquets à traiter dans les routeurs.

Toujours dans un souci d'efficacité, IPv6 impose une valeur minimale pour la MTU de 1280 octets (et donc pour la PMTU aussi). Cette valeur a pour objectif d'éviter qu'un lien imposant une MTU plus faible n'implique la transmission de petits paquets qui serait imposée à tous les

chemins empruntant ce lien. Si un support de transmission utilise des trames de taille inférieure à 1280 octets, il est nécessaire de mettre en place une couche d'adaptation dans ce système afin que la MTU atteigne la valeur demandée. C'est le cas par exemple pour les réseaux personnels sans fil de type LR WPAN (*Low Rate Wireless Personal Area Network*) comme IEEE 802.15.4 qui véhicule des trames avec des blocs d'au maximum 81 octets. Dans ce cas, une couche d'adaptation a du être spécifiée au moyen du RFC 4944 afin de rendre cette technologie utilisable dans le contexte IPv6 (6LowPAN). L'imposition d'une taille mimimale de MTU et de sa non fragmentation par IPv6 est une nouveauté en IPv6.

Découverte de la PMTU

La valeur de la PMTU n'est pas forcément connue à l'envoi d'un premier paquet vers une destination quelconque. L'émetteur du paquet fait alors la supposition que la taille maximale vers cette destination est égale à celle du support physique sur lequel il est connecté, c'est-à-dire la MTU du réseau d'accès.

L'acheminement du paquet IPv6 s'effectue normalement jusqu'au premier routeur rencontrant une incompatibilité entre la taille du paquet à transmettre et la taille maximale autorisée sur le système de transmission à utiliser en sortie ; autrement dit lorsque le paquet à transmettre est plus grand que la taille de la MTU offerte par le système de transmission. Comme le routeur en IPv6 n'est pas habilité à faire la fragmentation, le routeur est alors dans l'incapacité de traiter correctement le paquet. Le routeur supprime le paquet et en informe l'émetteur du paquet par un message de signalisation. Ce message repose sur le protocole ICMPv6 et sera décrit dans la séquence 3. L'objectif du message est d'informer du problème d'acheminement mais aussi d'indiquer la taille maximale à retenir pour que les paquets soient acheminés correctement. Cette procédure de découverte de la PMTU est spécifié dans le RFC 8201.

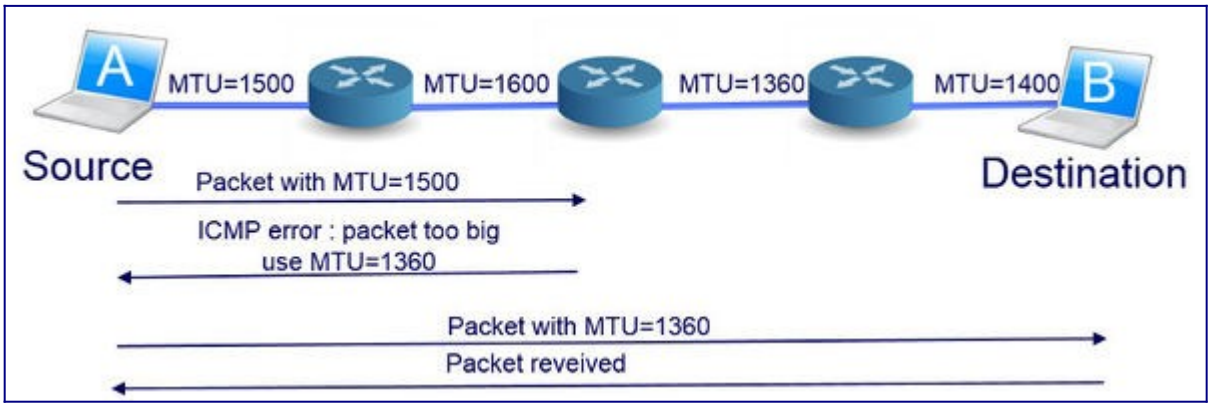


Figure 3 : Négociation pour la Path MTU Discovery.

La couche réseau de l'émetteur, à la réception de ce message, enregistre la valeur recommandée de la taille de paquet (cf. Figure 3). Cette valeur est la PMTU pour tous les prochains paquets vers cette même destination. Pour ce qui concerne les données perdues avec le paquet trop grand supprimé par le routeur, il est du ressort de la couche de transport comme TCP de gérer la fiabilité de la communication en retransmettant si nécessaire, ou de l'application de s'adapter à la perte.

Plusieurs itérations peuvent être nécessaires avant d'obtenir la PMTU, notamment si d'autres

systèmes de transmission imposent des MTU encore inférieures. L'émetteur enregistrera les valeurs recommandées successives jusqu'à arriver à la taille maximale autorisée sur le chemin. Les paquets suivants qui respecteront cette taille seront alors acheminés sans problème. En tout état de cause, le protocole IPv6 garantit que la MTU de tout lien ne peut descendre en dessous de 1 280 octets, valeur qui constitue ainsi une borne inférieure pour la PMTU.

Segmentation ou Fragmentation

Dans le cas de TCP, le terme approprié est segmentation car l'unité de protocole de TCP se nomme segment. Ainsi, la fragmentation forme des fragments et, donc, la segmentation des segments !

L'exploitation de l'information de PMTU se fait de plusieurs façons suivant la couche en charge de la fragmentation des données. Si un protocole de type TCP est utilisé, celui-ci assurera la segmentation des données de façon transparente pour les applications, en fonction des informations de PMTU que pourra lui communiquer la couche IPv6. Si un protocole de type UDP est utilisé, alors cette fragmentation devra être assurée par une couche supérieure, éventuellement par l'application elle-même. Si l'application n'est pas en mesure d'assurer cette fragmentation, alors il revient à la couche IPv6 de faire cette fragmentation. C'est ce que nous allons voir dans la section suivante.

NOTA : Attention au filtrage d'ICMPv6 ! *Contrairement à une pratique couramment répandue en IPv4,* ***il ne faut jamais filtrer l'ensemble des messages ICMPv6 en entrée d'un réseau (en particulier le message "Paquet trop grand").*** *Le filtrage peut introduire des effets néfastes sur le fonctionnement du réseau.*

Fragmentation IPv6

Il existe des cas où les données passées à la couche de réseau ne sont pas adaptées à la PMTU. C'est le cas par exemple des messages transportés sur UDP pour le système de fichiers NFS. Ces messages peuvent avoir une taille supérieure à celle autorisée sur le système de transmission. Comme on ne veut pas modifier ces applications, la couche réseau d'IPv6 doit aussi être capable de gérer la fragmentation.

La couche de réseau de l'hôte va alors adapter la taille du paquet pour son transfert. Elle va procéder à une fragmentation du paquet. Le principe de la fragmentation est de découper le paquet IPv6 en fragments pour constituer des paquets respectant la taille maximale autorisée. Ces paquets (ou fragments) sont émis et acheminés vers la destination comme n'importe quel autre paquet IP. La figure 4 illustre ce principe. Le paquet IPv6 original est découpé en fragment. Le premier fragment avec l'en-tête IPv6 est complété par l'extension de fragmentation. Les fragments suivants reprennent l'en-tête du paquet original et sont complétés avec l'extension de fragmentation. L'hôte destinataire se charge alors de rassembler les fragments et de reconstituer le paquet IP original pour le traiter comme s'il avait été transféré sans fragmentation.

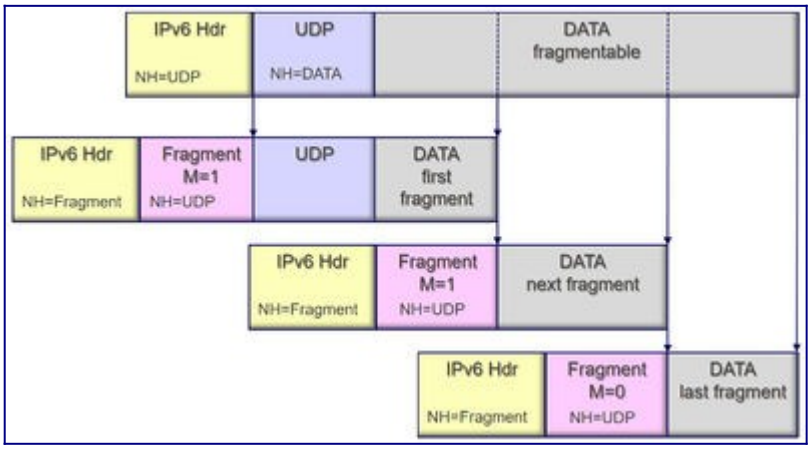


Figure 4 : Principe de la fragmentation en IPv6.

Lorsqu'il y a une fragmentation, se pose alors les problèmes de l'identification d'un fragment et de la position relative du fragment. L'identification du fragment vise à répondre à la question que peut se poser la destination : à quel paquet appartient-il ? Ce sont ce genre d'informations qui seront mises dans l'extension de fragmentation de l'en-tête IPv6. Cette extension est identifiée par le code d'en-tête 44. Le format de l'extension de fragmentation est présenté par la figure 5. Elle comporte trois champs de contrôle :

- le champ `Fragment offset` indique, lors du réassemblage, où placer le fragment dans le datagramme initial. Ce champ est sur 13 bits. L'unité de comptage est 8 octets. Aussi, la taille de tous les fragments, sauf le dernier, doit être multiple de 8 octets.
- le bit `M` (*More Fragments*) vaut 1 lorsqu'il indique qu'il y aura d'autres fragments. Seul le dernier fragment aura ce bit à zéro.
- le champ `identification` est utilisé pour identifier tous les fragments d'un même paquet initial. La valeur est différente pour chaque paquet et elle est recopiée dans chacun des fragments d'un paquet.

Cette extension est codée sur un seul mot de 64 bits et le premier mot n'est pas compté dans le calcul de longueur des extensions. Aussi, il est nécessaire d'avoir un champ longueur de l'extension.

Note : la fragmentation est permise en IPv4 au niveau des routeurs intermédiaires, leur donnant ainsi la possibilité de transmettre un paquet, même s'il est de taille supérieure à la MTU du lien suivant. Mais, dans ce cas, le mécanisme est jugé inefficace car il augmente la tâche des routeurs. En IPv6, avec la découverte de la PMTU, les routeurs intermédiaires n'ont plus besoin de fragmenter les paquets. Si la fragmentation est toujours nécessaire, elle est réservée aux hôtes.

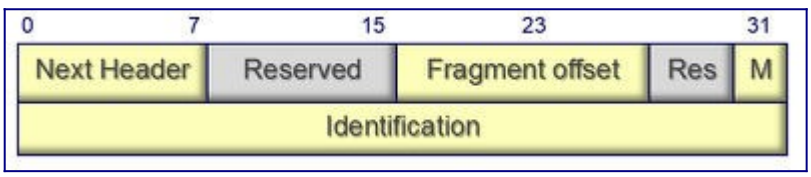


Figure 5 : Format de l'extension de fragmentation.

D'un point de vue opérationnel, la fragmentation en IPv6, et plus généralement l'ajout d'une extension d'en-tête, fragilise le transfert du paquet dans l'Internet. Le risque qu'il n'arrive pas à destination est augmenté. C'est ce que rapporte en substance le RFC 7872. Ce dernier essaie de quantifier l'ampleur exacte du problème. Trop de nœuds intermédiaires ne respectent pas le principe de "bout en bout" et les paquets *inhabituels* sont supprimés en cours de route. Devant un tel paquet, ces nœuds se permettent de supprimer le paquet. Cela peut être le résultat d'une politique délibérée (pare-feu appliquant le principe « dans le doute, on jette ») ou bien, souvent, de mises en œuvre incomplète de spécifications décrites dans les RFC.

La procédure de découverte de la PMTU et la fragmentation IPv6 ont des effets de bord indésirables. Le RFC 6946 présente les risques associés au fragment atomique. Le fragment atomique est un paquet IPv6 avec un en-tête de fragmentation sans être fragmenté du tout. Il n'a aucun intérêt pratique mais est engendré par certains systèmes lorsqu'ils reçoivent un message ICMP Packet too big indiquant une MTU inférieure à 1280 octets (la taille minimale pour une MTU IPv6). Le problème est que les fragments atomiques sont ensuite traités par bien des systèmes comme de vrais fragments, permettant ainsi certaines attaques subtiles. Le RFC 8021 estime qu'ils ne servent à rien et sont dangereux. Ce RFC demande d'abandonner la génération et le traitement des fragments atomiques. Pour cela, les hôtes doivent désormais ignorer les messages *ICMP Packet Too Big* lorsqu'ils indiquent une MTU inférieure à 1 280 octets.

Dans l'article[2], l'auteur indique que le traitement de l'en-tête de fragmentation est une cause de problèmes dans le déploiement opérationnel d'IPv6. Les paquets avec cet en-tête sont traités avec du retard dans les routeurs et peuvent subir un taux de perte plus important.

Jumbogrammes

Pour être complet sur la taille des paquets, il faut rappeler que IPv6 offre la notion de "jumbogramme" pour des paquets ayant une charge utile jusqu'à 4 Gio. Ceci est possible avec un champ `Payload Length` codé sur 32 bits comme nous l'avons déjà indiqué dans l'activité "Le format de l'en-tête IPv6". Le jumbogramme est essentiellement prévu pour le transfert à haut débit entre deux nœuds.

La mise en œuvre d'un jumbogramme demande des modifications dans la mise en œuvre de la couche de transport. Par exemple, le protocole UDP utilise un champ de longueur 16 bits pour le codage de la longueur totale du message UDP. On voit bien qu'il va falloir faire des adaptations car la longueur maximum d'un message UDP n'est pas du même ordre de grandeur que celle du jumbogrammme. Ces adaptations sont détaillées dans le RFC 2675. Si les "jumbogrammes" sont intéressants sur des liens qui disposent d'une MTU supérieure à 65583 octets (plus de 65535 octets de charge utile, plus 40 octets pour la taille fixe de l'en-tête, plus 8 octets pour l'en-tête d'extension *Hop-by-Hop*), il n'y a pas à ce jour de système de transmission utilisant des trames aussi importantes.

Il a été proposé à l'IETF de changer le statut du RFC 2675 [3] à "historique".

En effet, depuis plusieurs années, les jumbograms IPv6 n'ont pas été largement déployés, car dans les couches de transport communément utilisées, les champs longueur sont codés sur 16 bits, soit 65535 octets maximum.

Conclusion

Cette activité vous a présenté les différents éléments qui vont servir à définir la taille du paquet IPv6 à utiliser. Tout d'abord, IPv6 impose que tous les systèmes de transmission utilisés transfère un paquet jusqu'à 1280 octets sans qu'il soit besoin de le fragmenter au niveau d'IP. C'est la taille maximum d'un paquet pour être transféré sans fragmentation et sans problème lié à la taille. Au-delà de cette taille, il y a des risques de problème. Dans ce cas, la taille du paquet à transférer est conditionnée à la taille indiquée par la PMTU. Si la taille du paquet à envoyer est inférieure à cette taille, le paquet pourra être transmis vers sa destination sans problème. Si, par contre, la taille du paquet est plus grande que la valeur de PMTU, soit un ajustement de la taille du segment de données est nécessaire au niveau de la couche transport, soit la fragmentation par l'hôte est obligatoire. Dans l'article en ligne[4], l'auteur rapporte une autre idée : ne pas envoyer de paquets de plus de 1280 octets en dehors du réseau local. Ainsi, en transmettant des paquets à la taille minimale de la PMTU, il n'y a plus de besoin de fragmenter et donc d'utiliser la découverte de la PMTU. C'est une solution un peu extrême pour contourner la non mise en œuvre de la fragmentation et, lorsqu'elle est mise en œuvre, la fragmentation IPv6 peut avoir des effets néfastes[5] [6]du fait que certains nœuds jettent les fragments de paquet IPv6. Nous reviendrons sur le problème de la taille de la PMTU lors de la séquence suivante.

Pour conclure, retenez que la bonne taille du paquet IPv6 dépend du protocole de transport. Pour TCP, la valeur de 1280 octets montre de bons résultats[7].

Références bibliographiques

1. ↑ Huston, G. (2016), January. Blog of APNIC. Evaluating IPv4 and IPv6 packet fragmentation.
2. ↑ Huston, G (2021). Blog of APNIC. IPv6 fragmentation loss in 2021
3. ↑ Jones, T. (2019), May. IETF. Change Status of RFC 2675 to Historic.
4. ↑ Bortzmeyer, S. (2010) Fragmentation IPv6 : se résigner à couper à 1280 octets ?
5. ↑ Huston, G. (2018) The Internet Protocol Journal. Vol 21. N° 1. IPv6 and Packet Fragmentation
6. ↑ Huston, Geoff - Damas Joao (2021) The ISP Column. IPv6 Fragmentation Loss
7. ↑ Huston, G. (2016), May. Fragmenting IPv6.

Pour aller plus loin

Vous pouvez approfondir vos connaissances en consultant les liens suivants :

- RFC 2675 : IPv6 Jumbograms
- RFC 4821 : Packetization Layer Path MTU Discovery (Analyse par S.Bortzmeyer)
- RFC 4944 : Transmission of IPv6 Packets over IEEE 802.15.4 Networks
- RFC 6946 : Processing of IPv6 "atomic" fragments (Analyse par S.Bortzmeyer)
- RFC 7872 : Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World (Analyse par S.Bortzmeyer)
- RFC 8200 : IP version 6, Section 4.5 Fragment header (Analyse par S.Bortzmeyer)
- RFC 8021: Generation of IPv6 Atomic Fragments Considered Harmful (Analyse par S.Bortzmeyer)
- RFC 8201 : Path MTU Discovery for IP version 6 (Analyse par S.Bortzmeyer)

Conclusion

Suite à cette deuxième séquence, vous avez découvert les éléments du protocole IPv6 que sont :

- l'en-tête des paquets IPv6 ;
- les principes de l'acheminement et du routage des paquets IPv6 ;
- le contrôle par ICMPv6 de l'acheminement des paquets IPv6 ;
- Les contraintes pour le choix de la taille du paquet IPv6 ;
- l'encapsulation du paquet IPv6 mais aussi ce qui est encapsulé dans le paquet IPv6 ;
- les extensions de l'en-tête IPv6 et de leur usage ;
- Enfin vous avez pu observer le fonctionnement du protocole IPv6.

Nous rappelons qu'il peut être bien pratique d'avoir sur son poste de travail ce genre de formulaire qui présente l'essentiel du paquet IPv6 en une seule page comme [1] ou plus récent [2]. Ceci peut vous être utile pour la suite.

Dorénavent vous êtes aptes à approfondir d'autres mécanismes importants pour faciliter l'intégration du protocole dans toutes les infrastructures où IPv6 sera utile d'être déployer. C'est bien ce que vous allez découvrir dans les prochaines séquences.

Références bibliographiques

1. ↑ Stretch, J. (2009) packetlife.net. Aide-mémoire tout en une page. IPv6
2. ↑ Carrell Jeffrey L. (2020) site Teach Me IPv6. IPv6 Essentials Reference Sheet