RFC 8200 : Internet Protocol, Version 6 (IPv6) Specification

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 16 juillet 2017

Date de publication du RFC : Juillet 2017

https://www.bortzmeyer.org/8200.html

Ce RFC est la nouvelle norme du protocole IPv6. IP est le protocole de base de l'Internet, la version 6 y est minoritaire mais est bien plus répandue qu'elle ne l'était lors de la sortie de la précédente norme, qui était le RFC 2460[1], norme que notre nouveau RFC remplace (et cela fait passer IPv6 au statut de norme Internet, la précédente étant officiellement une proposition de norme <http://www.internetsociety.org/deploy360/blog/2017/07/rfc-8200-ipv6-has-been-standardized/>).

Pas de changements cruciaux, la norme est nouvelle, mais le protocole reste largement le même. Ce RFC 8200 continue à présenter IPv6 comme « "*a new version of the Internet Protocol (IP)*" ». Comme la première norme IPv6 est sortie en 1995, l'adjectif « "*new*" » n'est vraiment pas sérieux. Comme, malheureusement, la plupart des formations réseau ne parlent que d'IPv4 et traitent IPv6 de manière bâclée à la fin, le RFC présente IPv6 en parlant de ses différences par rapport à IPv4 (section 1 du RFC) :

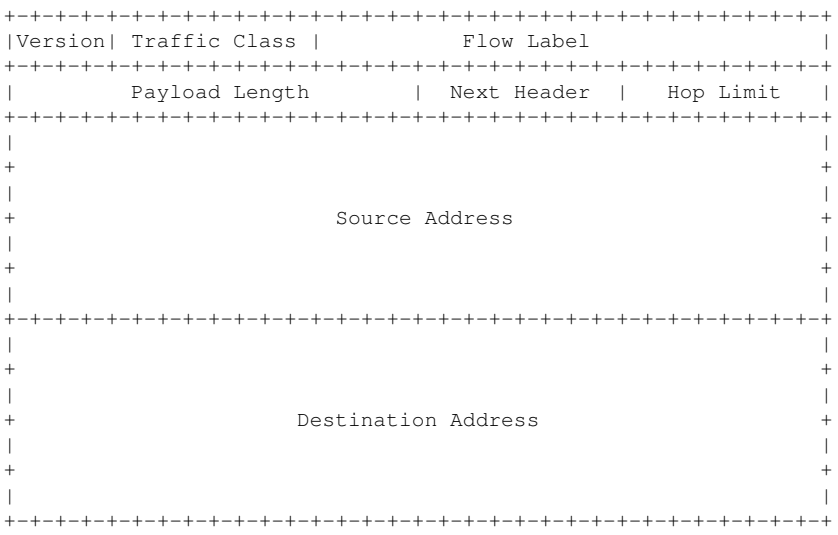
— La principale est évidemment l'espace d'adressage bien plus grand. Alors qu'IPv4, avec ses adresses sur 32 bits, ne peut même pas donner une adresse IP à chaque habitant de la planète (ce qui serait déjà insuffisant), IPv6, avec ses 128 bits d'adresse, permet de distribuer une quantité d'adresses que le cerveau humain a du mal à se représenter. Cette abondance est la principale raison pour laquelle il est crucial de migrer vers IPv6. Les autres raisons sont plus discutables.
— IPv6 a sérieusement changé le format des options. En IPv4, les options IP étaient un champ de longueur variable dans l'en-tête, pas exactement ce qui est le plus facile à analyser pour un routeur. Le RFC dit qu'IPv6 a simplifié le format mais c'est contestable : une complexité a succédé à une autre. Désormais, le premier en-tête est de taille fixe, mais il peut y avoir un nombre quelconque d'en-têtes chaînés. Le RFC utilise malheureusement des termes publicitaires assez déconnectés de la réalité, en parlant de format plus efficace et plus souple.
— IPv6 a un mécanisme standard pour étiqueter les paquets appartenant à un même flot de données. Mais le RFC oublie de dire qu'il semble inutilisé en pratique.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc2460.txt

— Et le RFC termine cette énumération par le plus gros pipeautage, prétendre qu'IPv6 aurait de meilleurs capacités d'authentification et de confidentialité. (C'est faux, elles sont les mêmes qu'en IPv4, et peu déployées, en pratique.)

Notez que ce RFC 8200 ne spécifie que le format des paquets IPv6. D'autres points très importants sont normalisés dans d'autres RFC, les adresses dans le RFC 4291, et ICMP dans le RFC 4443.

La section 3 présente le format des paquets IPv6 :



(D'ailleurs, si quelqu'un sait pourquoi l'adresse IP source est avant la destination ? Il me semblerait plus logique que ce soit l'inverse, puisque tous les routeurs sur le trajet doivent examiner l'adresse destination, alors qu'on n'a pas toujours besoin de l'adresse source.) Le numéro de version vaut évidemment 6, le « "*traffic class*" » est présenté en section 7, le « "*flow label*" » en section 6.

Le champ « "*Next header*" » remplace le « "*Protocol*" » d'IPv4. Il peut indiquer le protocole de transport utilisé (la liste figure dans un registre IANA `<https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml>` et est la même pour IPv4 et IPv6) mais aussi les en-têtes d'extension, une nouveauté d'IPv6, présentée en section 4 de notre RFC.

Le champ « "*Hop limit*" » remplace le « "*Time to Live*" » d'IPv4. En fait, les deux ont exactement la même sémantique, qui est celle d'un nombre maximal de routeurs utilisés sur le trajet (le but est d'éviter les boucles infinies dans le réseau). Autrefois, dans IPv4, il était prévu que ce champ soit réellement une durée, mais aucune mise en œuvre d'IPv4 ne l'avait jamais utilisé comme ceci. Le renommage dans IPv6 fait donc correspondre la terminologie avec une réalité ancienne (cf. aussi la section 8.2). Notez que c'est ce champ qui est utilisé par traceroute.

Voici une simple connexion HTTP en IPv6, vue avec tcpdump et tshark. Le client a demandé `/robots.txt` et a obtenu une réponse négative (404). Si vous voulez, le pcap complet est (en ligne sur `https://www.bortzmeyer.org/files/ipv6-http-connection.pcap`). Voici d'abord avec tcpdump avec ses options par défaut :

```
15:46:21.768536 IP6 2001:4b98:dc2:43:216:3eff:fea9:41a.37703 > 2605:4500:2:245b::42.80: Flags [S], seq 30530
```

On voit les deux adresses IPv6, tcpdump n'affiche rien d'autre de l'en-tête de couche 3, tout le reste est du TCP, le protocole de transport utilisé par HTTP. Avec l'option `-v` de tcpdump :

```
15:46:21.768536 IP6 (hlim 64, next-header TCP (6) payload length: 40) 2001:4b98:dc2:43:216:3eff:fea9:41a.37703 >
```

Cette fois, on voit le « "*Hop limit*" » (64), l'en-tête suivant (TCP, pas d'en-tête d'extension) et la longueur (40 octets). Pour avoir davantage, il faut passer à tshark (décodage complet en (en ligne sur `https://www.bortzmeyer.org/files/ipv6-http-connection.txt`)) :

```
Internet Protocol Version 6, Src: 2001:4b98:dc2:43:216:3eff:fea9:41a, Dst: 2605:4500:2:245b::42
    0110 .... = Version: 6
    .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00 (DSCP: CS0, ECN: Not-ECT)
        .... 0000 00.. .... .... .... .... .... = Differentiated Services Codepoint: Default (0)
        .... .... ..00 .... .... .... .... .... = Explicit Congestion Notification: Not ECN-Capable Transport (0
    .... .... .... 0000 0000 0000 0000 0000 = Flow label: 0x00000
    Payload length: 40
    Next header: TCP (6)
    Hop limit: 64
    Source: 2001:4b98:dc2:43:216:3eff:fea9:41a
    [Source SA MAC: Xensourc_a9:04:1a (00:16:3e:a9:04:1a)]
    Destination: 2605:4500:2:245b::42
    [Source GeoIP: France]
        [Source GeoIP Country: France]
    [Destination GeoIP: United States]
        [Destination GeoIP Country: United States]
```

On a cette fois tout l'en-tête IPv6 : notons le « "*Flow Label*" » (j'avais bien dit que peu de gens s'en servaient, il est nul dans ce cas).

La section 4 de notre RFC est dédiée à une nouveauté d'IPv6, les en-têtes d'extension. Au lieu d'un champ « "*Options*" » de taille variable (donc difficile à analyser) comme en IPv4, IPv6 met les options IP dans des en-têtes supplémentaires, chaînés avec l'en-tête principal. Par exemple, si un paquet UDP a un en-tête « "*Destination Options*" », le champ « "*Next header*" » de l'en-tête principal vaudra 60 (pour « "*Destination Options*" »), il sera suivi de l'en-tête d'extension « "*Destination Options*" » qui aura, lui, un « "*Next header*" » de 17 pour indiquer que ce qui suit est de l'UDP. (Je rappelle que les valeurs possibles pour « "*Next Header*" » sont dans un registre IANA <`https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml`>.) Il peut y avoir zéro, un ou davantage d'en-têtes d'extension chaînés entre eux.

Notez qu'analyser cette chaîne d'en-têtes est compliqué <`https://www.bortzmeyer.org/analyse-pcap-ipv6.html`> car les en-têtes n'ont pas tous le même format (le RFC 6564 a créé un format unique, mais cela ne concerne que les futurs en-têtes.) Il est donc exagéré de dire que la suppression du champ « "*Options*" » de taille variable a simplifié les choses.

Les différents en-têtes ne sont pas tous traités pareillement par les routeurs. Il existe notamment un en-tête, « "*Hop-by-hop Options*" » qui doit être examiné par tous les routeurs du trajet (cette obligation, jamais respectée, a de toute façon été relâchée par rapport au RFC 2460). C'est pour cela qu'il doit être placé au début des en-têtes, juste après l'en-tête principal. Les autres en-têtes d'extension doivent être ignorés par les routeurs.

Comme il est compliqué de rajouter un nouveau modèle d'en-tête (il faudrait modifier toutes les machines IPv6), une solution légère existe pour les options simples : utiliser les en-têtes d'options, « "*Hop-by-hop Options*" » et « "*Destination Options*" ». Tous les deux sont composés d'une série d'options encodées en TLV. En outre, le type de l'option indique, dans ses deux premiers bits, le traitement à appliquer au paquet si le système ne connait pas cette option. Si les deux premiers bits sont à zéro, on ignore l'option et on continue. Autrement, on jette le paquet (les trois valeurs restantes, 01, 10 et 11, indiquent si on envoie un message d'erreur ICMP et lequel). Ainsi, l'option pour la destination de numéro 0x07 (utilisée par le protocole de sécurité du RFC 5570) est facultative : elle a les deux premiers bits à zéro et sera donc ignorée silencieusement par les destinataires qui ne la connaissent pas (cf. registre IANA <https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xml#ipv6-parameters-2>.)

« "*Destination Options*" », comme son nom l'indique, n'est examinée que par la machine de destination. Si vous voulez envoyer des paquets avec cet en-tête, regardez mon article <https://www.bortzmeyer.org/destination-options-ipv6.html>.

Outre les en-têtes « "*Hop-by-hop Options*" » et « "*Destination Options*" », il existe des en-têtes :

— « "*Routing Header*" », qui permet de spécifier le routage depuis la source (un peu comme les options de « "*source routing*" » en IPv4). Il y a plusieurs types, et les valeurs possibles sont dans un registre IANA <https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xml#ipv6-parameters-3>, géré suivant les principes du RFC 5871. (Le type 0 a été retiré depuis le RFC 5095.)
— « "*Fragment Header*" », qui permet de fragmenter les paquets de taille supérieure à la MTU.
— Les autres en-têtes sont ceux utilisés par IPsec, décrits dans les RFC 4302 et RFC 4303.

La fragmentation est différente de celle d'IPv4. En IPv6, seule la machine émettrice peut fragmenter, pas les routeurs intermédiaires. Si le paquet est plus grand que la MTU, on le découpe en fragments, chaque fragment portant un « "*Fragment Header*" ». Cet en-tête porte une identification (un nombre sur 32 bits qui doit être unique parmi les paquets qui sont encore dans le réseau), un décalage ("*offset*") qui indique à combien d'octets depuis le début du paquet original se situe ce fragment (il vaut donc zéro pour le premier fragment) et un bit qui indique si ce fragment est le dernier. À la destination, le paquet est réassemblé à partir des fragments. (Il est désormais interdit que ces fragments se recouvrent, cf. RFC 5722.) Voici un exemple de fragmentation. La sonde Atlas <https://atlas.ripe.net/> n° 6271 <https://atlas.ripe.net/probes/6271/> a interrogé un serveur DNS de la racine Yeti <https://yeti-dns.org> avec le type de question ANY qui signifie « envoie-moi tout ce que tu peux / veux ». La réponse, plus grande que la MTU (plus de quatre kilo-octets !), a été fragmentée en trois paquets (le pcap complet est en (en ligne sur https://www.bortzmeyer.org/files/ipv6-dns-frag.pcap)) :

```
16:14:27.112945 IP6 2001:67c:217c:4::2.60115 > 2001:4b98:dc2:45:216:3eff:fe4b:8c5b.53: 19997+ [1au] ANY? .
16:14:27.113171 IP6 2001:4b98:dc2:45:216:3eff:fe4b:8c5b > 2001:67c:217c:4::2: frag (0|1232) 53 > 60115: 1999
16:14:27.113187 IP6 2001:4b98:dc2:45:216:3eff:fe4b:8c5b > 2001:67c:217c:4::2: frag (1232|1232)
16:14:27.113189 IP6 2001:4b98:dc2:45:216:3eff:fe4b:8c5b > 2001:67c:217c:4::2: frag (2464|637)
```

On note que tcpdump n'a interprété qu'un seul fragment comme étant du DNS, la premier, puisque c'était le seul qui portait l'en-tête UDP, avec le numéro de port 53 identifiant du DNS. Dans le résultat de tcpdump, après le mot-clé frag, on voit le décalage du fragment par rapport au début du paquet original (respectivement 0, 1232 et 2464 pour les trois fragments), et la taille du fragment (respectivement 1232, 1232 et 637 octets). Vu par tshark (l'analyse complète est en (en ligne sur https://www.bortzmeyer.org/files/ipv6-dns-frag.txt)), le premier fragment contient :

```
    Type: IPv6 (0x86dd)
Internet Protocol Version 6, Src: 2001:4b98:dc2:45:216:3eff:fe4b:8c5b, Dst: 2001:67c:217c:4::2
    Payload length: 1240
    Next header: Fragment Header for IPv6 (44)
    Fragment Header for IPv6
        Next header: UDP (17)
        Reserved octet: 0x00
        0000 0000 0000 0... = Offset: 0 (0 bytes)
        .... .... .... .00. = Reserved bits: 0
        .... .... .... ...1 = More Fragments: Yes
        Identification: 0xcbf66a8a
Data (1232 bytes)
```

On note le « "*Next Header*" » qui indique qu'un en-tête d'extension, l'en-tête « Fragmentation », suit l'en-tête principal. Le bit M (« "*More Fragments*" ») est à 1 (ce n'est que le premier fragment, d'autres suivent), le décalage (« "*offset*" ») est bien sûr de zéro. L'identificateur du paquet est de 0xcbf66a8a. Le dernier fragment, lui, contient :

```
Internet Protocol Version 6, Src: 2001:4b98:dc2:45:216:3eff:fe4b:8c5b, Dst: 2001:67c:217c:4::2
    Payload length: 645
    Next header: Fragment Header for IPv6 (44)
    Fragment Header for IPv6
        Next header: UDP (17)
        Reserved octet: 0x00
        0000 1001 1010 0... = Offset: 308 (2464 bytes)
        .... .... .... .00. = Reserved bits: 0
        .... .... .... ...0 = More Fragments: No
        Identification: 0xcbf66a8a
```

Cette fois, le « "*Next Header*" » indique que c'est de l'UDP qui suit, le décalage est de 2464, et le bit M est à zéro (plus d'autres fragments). L'identificateur est le même, c'est celui du paquet original, et c'est grâce à lui que la machine de destination saura réassembler les fragments. Notez qu'à la fin de l'analyse par tshark figure un réassemblage complet du paquet, ce qui permet une analyse DNS complète.

Et si je ne suis pas satisfait des en-têtes d'extension existants et que je veux créer le mien ? C'est en général une mauvaise idée. La plupart des cas concrets devraient être résolus avec un des en-têtes déjà normalisés. Notamment, l'en-tête « "*Destination Options*" » est là pour la majorité des situations. C'est lui qu'il faut regarder en premier (ce qu'ont fait les RFC 6744, RFC 6788 ou RFC 7837, la liste complète figurant dans un registre IANA <https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xml#ipv6-parameters-2>). Notre RFC 8200 exige donc que, si vous tenez à créer un nouvel en-tête, vous expliquiez bien pourquoi c'est indispensable, et pourquoi aucun des en-têtes existants ne convient.

Il est également déconseillé de créer de nouvelles options « "*hop-by-hop*" » (la liste actuelle est à l'IANA <https://www.iana.org/assignments/ipv6-parameters/ipv6-parameters.xml#ipv6-parameters-2>), car ces options doivent être traitées par tous les routeurs du trajet. Il y a un risque sérieux qu'ils laissent tomber les paquets ayant cet en-tête « "*Hop-by-hop Options*" », ou bien qu'ils le traitent plus lentement (logiciel du routeur et pas circuits matériels spécialisés). Là aussi, il faudra donc une justification sérieuse.

La section 5 de notre RFC se penche sur un problème délicat, la taille des paquets. IPv6 exige une MTU d'au moins 1 280 octets, mais les paquets sont souvent plus grands (par exemple 1 500 octets si on part d'un Ethernet). Beaucoup de liens ont, en pratique, une MTU de 1 500 octets mais IPv6 étant souvent porté par des tunnels (en raison de l'immobilisme de beaucoup de FAI, qui ne déploient toujours pas IPv6), un certain nombre de liens offrent moins de 1 500 octets. Normalement, l'émetteur d'un paquet IPv6 doit faire de la découverte de la MTU du chemin (RFC 8201) afin de pouvoir fragmenter, si

nécessaire. Une mise en œuvre « paresseuse » d'IPv6 pourrait se dispenser de découverte de la MTU du chemin, et se limiter à 1 280 octets par paquet.

Le problème est que la découverte de la MTU du chemin dépend du bon fonctionnement d'ICMP. L'émetteur d'un paquet doit pouvoir recevoir les paquets ICMP « "*Packet too big*" » (RFC 4443, section 3.2). Or, un certain nombre de pare-feux, stupidement configurés par des amateurs, bloquent tout ICMP « pour des raisons de sécurité » (c'est d'ailleurs une bonne question pour les entretiens d'embauche d'un administrateur de réseaux : « filtrer ICMP en entrée ou pas ? » S'il répond Oui, on est sûr qu'il est incompétent.) Ne recevant pas les « "*Packet too big*" », l'émetteur risque de croire à tort que ses paquets sont passés. En outre, si l'émetteur décide de fragmenter (en général, quand la MTU du chemin est inférieure à la sienne, avec TCP, on réduit la MSS, avec UDP, on fragmente), il faut que les fragments passent à travers le réseau. Et, là encore, un certain nombre de pare-feux bêtement configurés bloquent les fragments. Donc, en pratique, découverte du chemin + fragmentation est un processus fragile, en raison de ce véritable sabotage par des "*middleboxes*".

C'est certainement **le plus gros problème pratique** lors du déploiement d'IPv6. On peut même penser à prendre des mesures radicales et coûteuses, comme d'abaisser la MTU à 1 280 octets <https://www.bortzmeyer.org/fragmentation-ip-1280.html> pour être tranquille. Moins violent, il est fréquent de voir des MTU à 1 480 octets. Voici par exemple la configuration de mon routeur Turris Omnia <https://www.bortzmeyer.org/turris.html> pour passer par l'IPv6 de Free (un tunnel) :

```
config interface 'lan'
option ip6assign '64'
option ip6addr '2001:db8:42::1:fe/64'
option ip6prefix '2001:db8:42::/64'
option ip6gw '2001:db8:42::1'
option mtu '1480'
```

Et le « "*flow label*" », dont j'avais parlé plus haut ? Il est décrit dans la (très courte) section 6, qui renvoie surtout au RFC 6437. En pratique, ce champ semble peu utilisé comme on l'a vu dans l'exemple décodé par tshark.

Même chose pour le « "*traffic class*" », en section 7 : pour son utilisation pour la différenciation de trafic, voir les RFC 2474 et RFC 3168.

Maintenant qu'IPv6, protocole de couche 3, a été bien défini, le RFC monte vers la couche 4, en consacrant sa section 8 aux problèmes des couches supérieures. Cela concerne notamment la somme de contrôle. Si vous avez fait attention au schéma de l'en-tête IPv6 de la section 3, vous avez noté qu'il n'y avait pas de champ « "*Header Checksum*" », contrairement à ce qui existait en IPv4. En IPv6, pas de somme de contrôle en couche 3, c'était une tâche supplémentaire pour les routeurs (pour citer le RFC 791, « "*Since some header fields change (e.g., time to live), this is recomputed and verified at each point that the internet header is processed.*" »), tâche dont ils sont désormais dispensés.

Par contre, la somme de contrôle existe pour les en-têtes de couche 4 et elle devient même obligatoire pour UDP (elle était facultative en IPv4, quoique très fortement recommandée). (Voir le RFC 1071, au sujet de cette somme de contrôle.)

Un gros changement de ce RFC par rapport à son prédécesseur, le RFC 2460, concerne la sécurité. La section sur la sécurité est passée d'une annexe de deux lignes (qui se contentait de passer le bébé à

IPsec) à une analyse plus détaillée (section 10 du RFC). La question est délicate car la sécurité d'IPv6 a souvent fait l'objet de FUD, visant à justifier l'immobilisme de pas mal d'acteurs. Il est évidemment plus valorisant de dire « nous ne migrons pas vers IPv6 pour des raisons de sécurité » que de reconnaitre « nous sommes trop flemmards et paralysés par la peur du changement ». (Cf. mon exposé sur la sécurité d'IPv6 à l'ESGI <https://www.bortzmeyer.org/ipv6-securite.html>.) S'il fallait synthétiser (la deuxième partie de cette synthèse ne figure pas dans le RFC), je dirais :

— La sécurité d'IPv6 est quasiment la même qu'en IPv4 (ce qui est logique, il ne s'agit après tout que de deux versions du même protocole). Les grandes questions de sécurité d'IPv4 (usurpation d'adresse source, relative facilité à faire des attaques par déni de service, pas d'authentification ou de confidentialité par défaut) sont exactement les mêmes en IPv6. (C'est une blague courante à l'IETF de dire que si IPv4, l'un des plus grands succès de l'ingéniérie du vingtième siècle, était présenté à l'IESG aujourd'hui, il serait rejeté pour sa trop faible sécurité.)
— Par contre, en pratique, les solutions techniques d'attaque et de défense, ainsi que les compétences des attaquants et des défenseurs, sont bien plus faibles en IPv6. Pas mal de logiciels « de sécurité » ne gèrent pas IPv6, bien des logiciels de piratage ne fonctionnent qu'en IPv4, les administrateurs système sont déroutés face à une attaque IPv6, et les pirates ne pensent pas à l'utiliser. (Faites l'expérience : configurez un pot de miel SSH en IPv4 et IPv6. Vous aurez plusieurs connexions par jour en IPv4 et jamais une seule en IPv6.) L'un dans l'autre, je pense que ces deux aspects s'équilibrent.

Bon, assez de stratégie, passons maintenant aux problèmes concrets que décrit cette section 10. Elle rappelle des risques de sécurité qui sont exactement les mêmes qu'en IPv4 mais qu'il est bon de garder en tête, ce sont des problèmes fondamentaux d'IP :

— Communication en clair par défaut, donc espionnage trop facile pour les États, les entreprises privées, les pirates individuels. (Cf. Snowden.)
— Possibilité de rejouer des paquets déjà passés, ce qui permet certaines attaques. Dans certains cas, on peut modifier le paquet avant de le rejouer, et ça passera.
— Possibilité de générer des faux paquets et de les injecter dans le réseau.
— Attaque de l'homme du milieu : une entité se fait passer pour l'émetteur auprès du vrai destinataire, et pour le destinataire auprès du vrai émetteur.
— Attaque par déni de service, une des plaies les plus pénibles de l'Internet.

En cohérence avec le RFC 2460 (mais pas avec la réalité du terrain), notre RFC recommande IPsec (RFC 4301) comme solution à la plupart de ces problèmes. Hélas, depuis le temps qu'il existe, ce protocole n'a jamais connu de déploiement significatif sur l'Internet public (il est par contre utilisé dans des réseaux privés, par exemple le VPN qui vous permet de vous connecter avec votre entreprise de l'extérieur utilise probablement une variante plus ou moins standard d'IPsec). Une des raisons de ce faible déploiement est la grande complexité d'IPsec, et la complexité pire encore de ses mises en œuvre. En pratique, même si le RFC ne le reconnait que du bout des lèvres, ce sont les protocoles applicatifs comme SSH ou TLS, qui sécurisent l'Internet.

Pour les attaques par déni de service, par contre, aucune solution n'est proposée : le problème ne peut pas forcément se traiter au niveau du protocole réseau.

La différence la plus spectaculaire entre IPv4 et IPv6 est évidemment la taille des adresses. Elle rend le balayage bien plus complexe (mais pas impossible), ce qui améliore la sécurité (l'Internet IPv4 peut être exploré incroyablement vite, par exemple avec masscan <https://github.com/robertdavidgraham/masscan>, et, si on est trop flemmard pour balayer soi-même, on peut utiliser des balayages déjà faits, par exemple par Shodan ou). Le RFC 7707 fait une très bonne synthèse de l'état de l'art en matière de balayage IPv6. Par exemple, Shodan, cité plus haut, doit utiliser des techniques assez douteuses <https://isc.sans.edu/forums/diary/Targeted+IPv6+Scans+Using+poolntporg/20681/> pour récolter des adresses IPv6 à examiner.

Et qu'en est-il de la vie privée ? L'argument, largement FUDé, a été beaucoup utilisé contre IPv6. Le RFC note qu'IPv6 a entre autre pour but de rendre inutile l'usage du NAT, dont certaines personnes prétendent qu'il protège un peu les utilisateurs. L'argument est largement faux : le NAT (qui est une

réponse à la pénurie d'adresses IPv4, pas une technique de sécurité) ne protège pas contre tout "*fingerprinting*", loin de là. Et, si on veut empêcher les adresses IP des machines du réseau local d'être visibles à l'extérieur, on peut toujours faire du NAT en IPv6, si on veut, ou bien utiliser des méthodes des couches supérieures (comme un relais).

Autre question de vie privée avec IPv6, les adresses IP fondées sur l'adresse MAC. Cette ancienne technique, trop indiscrète, a été abandonnée avec les RFC 4941 et RFC 7721, le premier étant très déployé. Il est curieux de constater que cet argument soit encore utilisé, alors qu'il a perdu l'essentiel de sa (faible) pertinence.

Mais il y avait bien des problèmes de sécurité concrets avec le précédent RFC 2460, et qui sont réparés par ce nouveau RFC :

— Les « fragments atomiques » ne doivent désormais **plus** être générés (RFC 8021) et, si on en reçoit, doivent être « réassemblés » par une procédure spéciale (RFC 6946).
— Les fragments qui se recouvrent partiellement sont désormais interdits (cf. RFC 5722, le réassemblage des fragments en un paquet est déjà assez compliqué et sujet à erreur comme cela).
— Si le paquet est fragmenté, les en-têtes d'extension doivent désormais tous être dans le premier fragment (RFC 7112).
— L'en-tête de routage ("*Routing Header*") de type 0, dit « RH0 », est abandonné, il posait trop de problèmes de sécurité (cf. RFC 5095 et RFC 5871).

L'annexe B de notre RFC résume les changements depuis le RFC 2460. Pas de choses révolutionnaires, les changements les plus importantes portaient sur la sécurité, comme listé un peu plus haut (section 10 du RFC). On notera comme changements :

Section 1 Clarification que IPv6 est gros-boutien, comme IPv4 (cf. l'annexe B du RFC 791).
Section 3 Clarification de l'utilisation du « "*hop limit*" », décrémenté de 1 à chaque routeur.
Section 4 Clarification du fait qu'un équipement intermédiaire ne doit **pas** tripoter les en-têtes d'extension (à part évidemment « "*Hop-by-hop Options*" »), ni les modifier, ni en ajouter ou retirer, ni même les lire.
Section 4 Le traitement de l'en-tête « "*Hop-by-hop Options*" par les routeurs sur le trajet n'est plus obligatoire. Si un routeur est pressé, il peut s'en passer (le RFC suit ainsi la pratique).
Section 4.4 Suppression de l'en-tête de routage de type 0.
Section 4.5 Pas mal de changements sur la fragmentation (un processus toujours fragile !), notamment l'abandon des fragments atomiques.
Section 4.8 Intégration du format unique des éventuels futurs en-têtes d'extension, format initialement présenté dans le RFC 6564.
Section 6 Reconnaissance du fait que « "*Flow Label*" » n'était pas réellement bien défini, et délégation de cette définition au RFC 6437.
Section 8.1 Autorisation, dans certains cas bien délimités, d'omission de la somme de contrôle UDP (RFC 6935).
— Il y a eu également correction de diverses erreurs comme les 2541 <https://www.rfc-editor.org/errata_search.php?eid=2541> (omission sur la définition du « "*Flow Label*" ») et 4279 <https://www.rfc-editor.org/errata_search.php?eid=4279> (paquet entrant avec un « "*Hop Limit*" » déjà à zéro).

Le projet de mise à jour de la norme IPv6 avait été lancé en 2015 (voici les supports du premier exposé <https://www.ietf.org/proceedings/93/slides/slides-93-6man-3.pdf>).

D'habitude, je termine mes articles sur les RFC par des informations sur l'état de mise en œuvre du RFC. Mais, ici, il y en a tellement que je vous renvoie plutôt à cette liste <https://www.ipv6ready.org/db/index.php/public/>. Notez que des tests d'interopérabilité ont été faits sur les modifications introduites par ce nouveau RFC et que les résultats publiés n'indiquent pas de problème <https://www.ietf.org/proceedings/95/slides/slides-95-6man-2.pdf>.

Parmi les publications récentes sur le déploiement d'IPv6, signalons :

— Le rapport de l'ARCEP au gouvernement français <https://www.arcep.fr/index.php?id=8571&no_cache=0&tx_gsactualite_pi1%5buid%5d=1905&tx_gsactualite_pi1%5bannee%5d=&tx_gsactualite_pi1%5btheme%5d=&tx_gsactualite_pi1%5bmotscle%5d=&tx_gsactualite_pi1%5bbackID%5d=26&cHash=e46b8063c1ba85ae60e274e06c54f22e> sur la transition vers IPv6 et les moyens de l'accélérer.
— Le rapport 2016 de l'l[Caractère Unicode non montré [2]]Observatoire ANSSI/AFNIC de la résilience de l[Caractère Unicode non montré]Internet français <https://www.ssi.gouv.fr/agence/rayonnement-scientifique/lobservatoire-de-la-resilience-de-linternet-franc> a une section sur IPv6.
— Google publie d'intéressantes statistiques <https://www.google.com/intl/en/ipv6/statistics.html> sur l'accès en IPv6 à ses services.

2. Car trop difficile à faire afficher par LaTeX