Quelques points durs d'IPv6

Matthieu Herrb

Quelques points durs d'IPv6

Matthieu Herrb
CNRS-LAAS
matthieu.herrb@laas.fr

Résumé

Cet article aborde le déploiement d'IPv6 dans le monde réel. Il montre un certain nombre d'obstacles importants à l'adoption d'IPv6 au delà des cercles restreints des spécialistes des protocoles réseau ou de quelques grandes organisations.

Ces difficultés sont présentées en trois catégories : celles liées à la mise en place et à l'administration des réseaux, celles qui touchent aux applications du point de vue du développeur et de l'utilisateur et enfin celles qui ont un impact sur la sécurité.

Mots clefs

IPv6, applications, sécurité.

Introduction

Une nouvelle génération du protocole IP, la version 6 devrait être déployée de façon de plus en plus importante au cours des mois ou des années à venir. IPv6 est supposé apporter plusieurs améliorations par rapport à la version actuellement utilisée (IPv4) :

– un espace d'adressage beaucoup plus important (2^{128} adresses, ce qui représente plusieurs centaines de milliers d'adresses par mètre carré de la surface de la terre, contre 2^{32} pour IPv4), permettant beaucoup de souplesse dans les politiques d'adressage,
– une simplification de la structure des paquets qui devrait permettre des implémentations plus performantes et plus robustes,
– un mécanisme d'auto-configuration qui limite le besoin de configuration manuelle des systèmes,
– des mécanismes de sécurité intégrés (IPsec),
– des fonctions avancées pour la mobilité, le multicast, etc.

Certaines administrations aux États-Unis ont défini des calendriers assez serrés pour un déploiement opérationnel de leurs services réseau en IPv6 en 2008 [1].

Bien que IPv6 ne soit plus si nouveau que cela et que de nombreuses expérimentations de déploiement aient déjà été réalisées, il subsiste un certain nombre de difficultés qui pourraient bloquer ou retarder sa mise en place.

Ces difficultés sont de plusieurs natures : il y a d'abord la complexité intrinsèque plus importante du nouveau protocole. Ensuite, l'infrastructure actuelle d'IPv6 n'est pas toujours adaptée aux attentes des utilisateurs et enfin le support d'IPv6 n'est pas une préoccupation majeure des éditeurs de logiciels, ce qui limite la disponibilité et la qualité du support d'IPv6 dans les applications.

1 Administration du réseau

Pour déployer IPv6 sur un site, après la phase de planification initiale, commence la configuration et la mise en route des équipements IPv6.

1.1 Adresses difficiles à manipuler

Le premier contact avec les adresses IPv6 est toujours un peu traumatisant. Mémoriser des adresses de 128 bits exprimées sous forme de trente-deux caractères hexadécimaux et raisonner sur ces adresses n'est pas trivial.

IPv6 fournit plusieurs outils qui devraient permettre d'éviter de manipuler directement les valeurs numériques : l'auto-configuration des adresses et les mécanismes de mise à jour automatique du DNS. Néanmoins ceux-ci ne fonctionnent pas toujours de façon suffisamment satisfaisante pour pouvoir se reposer sur eux. On observe (en regardant des connexions IPv6 entrantes sur un serveur Web par exemple) que de nombreuses machines connectées au réseau internet en IPv6 ne disposent pas de serveur DNS permettant de remonter à leur nom.

Enfin certains services (configuration du résolveur dans `/etc/resolv.conf` - en attendant le déploiement de DHCPv6, dont le standard [2] n'est pas encore largement implémenté) nécessitent toujours le recours à une adresse numérique.

Par ailleurs, la multiplication des familles d'adresses (lien-locales, globales, adresses IPv4 mappées dans IPv6) accroît le nombre des adresses utilisables par une machine et rend plus complexe l'association d'une adresse avec un nom de machine. Cela a également des implications au niveau de la sécurité, comme cela sera montré plus bas.

1.2 Besoin d'adresses IP globales ?

L'une des principales motivations du passage à IPv6, la pénurie prévisible du nombre d'adresses IPv4 disponibles, semble aujourd'hui peu d'actualité en Europe et sur le continent nord-américain. Le déploiement de gadgets personnels connectés à IP de manière globale se fait bien souvent par le biais de portails propriétaires utilisant des adresses IP privées (RFC 1918), par exemple dans le cas des accès au Web pour les téléphones mobiles.

Les consultants de la société Netcraft ont recensé 72 mil-

lions de sites web actifs dans le monde début septembre 2005. Cette valeur donne une indication du nombre minimum du nombre d'adresses IP réellement nécessaires dans le modèle économique actuel de l'internet.

S'il est incontestable que certaines régions du monde ne pourront jamais obtenir suffisamment d'adresses IPv4 pour satisfaire leurs besoins de connexion, il n'est pas évident aujourd'hui que cela suffise pour pousser à un déploiement complet d'IPv6 dans nos régions [3].

Le nouveau protocole est aujourd'hui déployé essentiellement par des ingénieurs réseau et la commande `ping` est souvent le seul test de connectivité. En pratique, on observe (et le reste de cet article essaye de le montrer) que la possibilité d'atteindre par ping une adresse IPv6 n'est qu'une première étape avant de pouvoir utiliser pleinement le nouveau protocole.

1.3 Infrastructure de deuxième classe

L'infrastructure de transport au niveau national et international est une base indispensable au déploiement d'IPv6. En France, seuls deux opérateurs fournissent actuellement une connectivité IPv6 native à leurs clients : Renater et Nerim.

Dans la phase de déploiement actuelle, celle-ci souffre de deux problèmes :
- Les équipements (routeurs, pare-feux, points d'accès wifi, etc.) utilisés dans l'infrastructure ne supportent pas tous IPv6 de manière native. Cela conduit bien souvent à mettre en place des structures parallèles à l'infrastructure IPv4, parfois à base de matériel à moindre coût.
- La non-disponibilité d'équipements réseau supportant IPv6 est aujourd'hui une des causes principales de la lenteur du développement d'IPv6.

Une étude récente [4] sur les longueurs moyennes des routes dans l'internet montre que les routes IPv6 sont en moyenne plus longues que les routes IPv4.

Dans [5] les auteurs ont mis en évidence deux phénomènes touchant l'infrastructure IPv6 : environ 16% d'un échantillon de quelques milliers de sites affichant une double adresse IPv4 et IPv6 (présence d'enregistrements A et AAAA pour un même nom) ne sont pas atteignables en IPv6, alors qu'ils le sont en IPv4. La proportion opposée (atteignables en IPv6 mais pas en IPv4) est nettement plus faible. D'autre part, la mesure des temps de retour de *ping* sur ces sites montre des temps généralement supérieurs en IPv6 par rapport à IPv4. Les auteurs ont également analysé leurs données en fonction de la répartition géographique des sites : ce sont les sites de la zone ARIN (Amérique du Nord) qui souffrent le plus de ce déficit de performance de l'environnement IPv6.

L'infrastructure IPv6 offre donc souvent des performances moins bonnes que sa grande soeur IPv4, et parfois même se trouve implémentée sur des équipements ou des logiciels qui ne sont pas satisfaisants du point de vue de la conformité aux normes d'IPv6 (implémentation incomplète ou obsolète voire erronée).

1.4 Résolution des noms et des adresses

La mise en place de l'infrastructure DNS adaptée à IPv6 reproduit une partie des difficultés évoquées ci-dessus. La résolution des noms et des adresses IPv6 dans les différents systèmes se fait de façon assez hétérogène et cela (couplé à la lourdeur de manipulation des adresses) rend le DNS peu fiable.

Actuellement, on observe couramment plusieurs types de comportement anormaux relatifs à IPv6 de la part de certains serveurs de noms :
- *pas de résolution DNS inverse* : soit à cause d'une erreur de paramétrage, soit à défaut de pouvoir mettre à jour le serveur de la zone ;
- *réponse à une requête AAAA (IPv6) par un enregistrement A (IPv4)*. Ce type de comportement produit des effets variés en fonction des systèmes, mais n'est certainement pas un comportement défini dans les RFC ;
- *Trou noir 1* : le serveur ne répond pas du tout à la requête AAAA : au bout de quelques dizaines de secondes d'attente le client abandonne la requête. Normalement il essaye ensuite une requête de type A et tout se poursuit normalement en IPv4, mais avec un retard important ;
- *Trou noir 2* : au lieu de répondre qu'il n'y a pas d'adresse IPv6 correspondant au nom demandé, le serveur répond par une erreur indiquant que le domaine auquel appartient le nom n'existe pas ou est en erreur.
  Cette réponse peut être conservée dans le cache d'un serveur récursif qui retournera cette réponse pour toutes les requêtes qui lui seront adressées (y compris celles de type A). Ce comportement a pour effet de rendre tout le domaine inaccessible avec une requête DNS.

Il est possible de tenter de corriger ces problèmes en contactant les administrateurs des sites en question, mais ceux-ci n'ont pas toujours la possibilité technique ou la volonté politique de modifier la configuration de leurs serveurs.

Alors que la résolution inverse utilisant le domaine `ip6.int` est obsolète depuis plusieurs mois [6], on observe encore qu'un peu plus du quart des requêtes inverses arrivant sur les serveurs DNS de notre laboratoire utilisent `ip6.int`.

Les possibilités de mise à jour automatique offertes par les serveurs de noms ne seront probablement pas largement utilisées, à cause des problèmes de sécurité associés (le protocole de mise à jour ne demande pas d'authentification forte du client souhaitant faire un mise à jour).

1.5 Une mesure du déploiement actuel

Le tableau de la figure 1 donne la proportion de requêtes DNS de type AAAA reçues par les serveurs DNS primaire et secondaire d'un laboratoire de taille importante en une semaine, pour trois noms synthétiques correspondant à 3 services effectivement disponibles en IPv6. La dernière ligne donne la proportion globale de requêtes de type AAAA par rapport au total (A et AAAA), quel que soit le nom recherché.

FIG. 1 – *Proportion de requêtes AAAA en fonction du service*

Hôte	requêtes	requêtes AAAA (%)
`www`	24507	3.9
`ftp`	691	6.7
`ntp`	12493	18.3
tous	397538	22.7

Dans ce tableau, on peut observer qu'à peu près 22 % des machines en relation avec notre laboratoire ont une pile IPv6, et, par conséquent, émettent des requêtes de type AAAA. Mais au niveau applicatif, moins d'un cinquième de ces machines dispose d'un navigateur Web capable de faire de l'IPv6.

Il faut noter qu'une machine dotée d'une double pile IPv4/IPv6 et d'un navigateur compatible IPv6, mais ne disposant pas de la connectivité IPv6, va émettre une requête de type AAAA et se connecter finalement en IPv4.

Dans d'autres applications (FTP et NTP) le support d'IPv6 semble plus développé, mais le nombre effectif de requêtes observées est peut-être trop petit pour que le chiffre soit significatif (moins de 1 % du total des requêtes pour chacun de ces services).

2 IPv6 dans les applications

IPv6 est supporté aujourd'hui dans tous les systèmes majeurs : Windows XP, Linux (dans la plupart des distributions récentes basées sur les noyaux 2.6), Solaris (à partir de la version 8), Mac OS X et *BSD. Il est donc facile aujourd'hui de trouver un poste de travail disposant d'une pile IPv6.

Mais cela ne suffit pas pour dire qu'IPv6 est utilisable. Toutes les applications et les services doivent également être modifiés pour supporter le nouveau protocole.

L'expérience vécue de l'ajout du support d'IPv6 dans le système de multi-fenêtrage X, a montré les problèmes auxquels est confronté le développeur qui veut adapter une application réseau à IPv6. Cette expérience permet d'illustrer les points qui sont soulevés dans cette partie.

2.1 Interface socket

Le RFC 2553 [7, 8] définit les extensions de l'interface de programmation des sockets pour supporter IPv6, en étendant les interfaces existantes dans deux directions.

Il définit une nouvelle interface de programmation pour la résolution des noms en introduisant une nouvelle famille de fonctions dont la plus important est `getaddrinfo()`.

IPv6 est représenté sous la forme d'une nouvelle famille d'adresses : `AF_INET6` et d'une extension des structures utilisées pour représenter les adresses afin de pouvoir contenir les adresses de cette nouvelle famille.

Quelques modifications mineures sur les fonctions de manipulation des sockets leur permettent de prendre en compte la nouvelle famille d'adresses.

Ces nouvelles interfaces restent peu employées dans les applications et peu connues des programmeurs d'applications réseau, probablement en partie parce que trop souvent perçues comme spécifiques à IPv6. Elles sont pourtant indispensables pour rendre une application compatible IPv6.

Néanmoins la compatibilité d'une application avec IPv6 ne se limite pas à un simple remplacement de quelques fonctions par leurs pendantes multi-protocoles.

En particulier, dans le monde IPv6 une machine connectée au réseau aura presque toujours plusieurs adresses (IPv4, IPv6 globale, IPv6 lien-locale), et plusieurs représentations possibles de ces adresses. Or de nombreuses applications existantes considèrent que, à part l'adresse *localhost*, une machine n'a qu'une adresse IP.

2.2 Gestion des deux protocoles

Lorsqu'un serveur est accessible en IPv4 et en IPv6 et que le code d'une application est prêt à supporter IPv6, il reste une question difficile : comment choisir le protocole à utiliser, v4 ou v6 ?

Normalement, cette décision ne devrait pas être prise au niveau de l'application, mais devrait découler d'une politique définie globalement au niveau de la machine, voire au niveau d'un site.

Aujourd'hui, les applications compatibles IPv6 tentent d'utiliser IPv6 de façon préférentielle, sans toujours laisser à l'utilisateur le choix. Cette politique par défaut peut être nuisible à l'adoption d'IPv6 dans les cas où les premiers tests sont faits sur une infrastructure moins performante que sa contrepartie IPv4, en donnant aux utilisateurs l'impression que leurs applications fonctionnent moins bien.

2.3 Le support d'IPv6 dans X11

Le système multi-fenêtres X, développé en grande partie avant l'apparition d'IPv6, ne s'est préoccupé de son existence que récemment. Un groupe de travail mené par des ingénieurs de la société Sun a développé une spécification et une implémentation pour ajouter le support d'IPv6 à X [9], puis cette implémentation a été intégrée dans XFree86 et X.Org les deux versions de référence pour les Unix Libres.

Lors de sa conception initiale, le protocole X a été conçu de manière indépendante des protocoles réseau utilisés pour transporter ses données. Ces derniers sont isolés dans une couche particulière appelée *Xtrans* qui présente une interface indépendante du protocole de transport réellement utilisé au serveur et aux applications. Initialement, trois protocoles de transport étaient supportés : les sockets Unix locaux (et toutes les variantes, basées ou non sur les *streams*), le protocole TCP (sur IPv4) et le protocole Decnet. L'ajout

d'IPv6 ne semblait donc pas a priori trop complexe.

Le protocole X11. Pour le protocole X11 proprement dit, les modifications apportées initialement ont été relativement limitées. La seule difficulté était causée par l'utilisation du caractère « **:** » comme séparateur entre une adresse de machine et un numéro d'écran, qui rendait ambiguë la désignation d'un serveur X avec une adresse sous forme numérique. Comme dans d'autres cas similaires, l'ambiguïté est levée par l'utilisation de crochets autour de l'adresse IPv6 [**?**], par exemple : `DISPLAY=[2001:660:6602:1:2]:0.0`.

L'implémentation initiale de Sun, basée sur Solaris, se contentait de créer une nouvelle classe de transport dans *Xtrans* supportant un socket IPv6, capable d'accepter indifféremment des connexions TCP en IPv4, via des adresses mappées, ou en IPv6. Ainsi tout client ou serveur supportant IPv6 peut utiliser cette classe à la place de l'ancienne classe IPv4 : un client fera donc ses connexions par défaut en IPv6, et un serveur acceptera les deux versions du protocole.

Malheureusement cette approche s'est avérée insuffisante pour 2 raisons :
– certains systèmes (BSD) ne peuvent accepter de connexion IPv4 sur un socket appartenant à la famille `AF_INET6`. Il faut donc créer un socket par version du protocole pour les serveurs,
– le socket IPv6 unique pour le client ne fournit pas de mécanisme de repli vers une adresse IPv4 si la connexion IPv6 échoue. Or il est considéré comme normal pour une machine à double pile de basculer en IPv4 lorsque la connexion IPv6 échoue.

De plus, il s'est avéré difficile de prendre en compte cette logique de repli de façon transparente à l'intérieur de *Xtrans*. Finalement c'est donc dans la bibliothèque X11 elle-même qu'à été implémenté le mécanisme de repli. Il se retrouve donc figé et il est à craindre que toutes les implémentations du protocole ne suivent pas exactement la même logique.

Il faut d'ailleurs noter que ce problème est si complexe que dans le cas de xdm – l'outil utilisé pour la gestion de terminaux X – la couche de transport du protocole annexe XDMCP n'a pas encore été entièrement rendue capable d'utiliser IPv6. Les solutions aux deux problèmes évoqués ci-dessus introduisent des complications dans le protocole lui-même pour lesquelles aucune solution n'a été proposée à ce jour.

Enfin, dans la version actuelle, cette implémentation ne gère absolument pas les adresses lien-locales, alors que cela pourrait être intéressant, par exemple pour déployer une salle de terminaux X avec un minimum de configuration réseau.

Le protocole d'authentification. Un second problème dans le support d'IPv6 dans les applications s'est retrouvé dans X. Le mécanisme d'authentification et de contrôle d'accès utilisé transporte les adresses du client et du serveur qui tentent d'établir une connexion. Or le protocole existant (XDM-AUTHORIZATION-1) limite la taille disponible pour transporter l'adresse IP à quatre octets. Il n'était donc pas possible d'étendre simplement le protocole en ajoutant une nouvelle famille d'adresses à la liste existante. Une nouvelle spécification a donc été définie, XDM-AUTHORIZATION-2, qui accepte des adresses réseau jusqu'à seize octets. Par ailleurs elle passe d'un chiffrement DES avec clés de 56 bits à un chiffrement basé sur AES, avec des clés de 128 ou 256 bits.

Malheureusement, cette spécification n'a pas encore été implémentée par manque de disponibilité de développeurs, et cela laisse donc les utilisateurs d'IPv6 dans une situation où ils ne peuvent qu'utiliser le protocole d'authentification simpliste fondé sur les *magic-cookies*.

Leçons à tirer. Cette expérience sur un ensemble de logiciels très utilisés et largement déployés sur des systèmes différents montre bien la difficulté de la prise en compte complète d'IPv6 dans les applications. Bien que le système X ait été conçu à l'origine pour gérer plusieurs familles d'adresses, les spécificités d'IPv6 (notamment la nécessité de cohabiter avec IPv4) ont imposé de profondes modifications dans le code, voire dans les spécifications des protocoles utilisés.

Toutes les applications nécessitant une adaptation pour fonctionner avec IPv6 devront prendre en compte ces quatre aspects :
– gérer le choix de la famille d'adresse utilisée par les connexions,
– accepter des connexions IPv6,
– émettre des connexions IPv6,
– prendre en compte les adresses IPv6 (sous toutes leurs formes) pour les fonctions d'authentification ou de contrôle d'accès.

2.4 Quelques autres exemples

En observant comment est traité le support d'IPv6 dans quelques autres applications, on voit que les mêmes types de problèmes se sont posés aux développeurs, et qu'ils ont été résolus avec plus ou moins de bonheur.

OpenSSH. OpenSSH, une implémentation libre du protocole *Secure Shell*, supporte IPv6 depuis les premières versions. La qualité de ce support, assez incomplet dans les premières versions, s'est progressivement améliorée.

OpenSSH a été entièrement réécrit en utilisant exclusivement la nouvelle interface multi-protocoles. La version portable fournit une implémentation de cette interface pour les systèmes qui ne la supportent pas.

La syntaxe utilisée pour représenter les adresses IP et les numéro de port a été modifiée afin de pouvoir utiliser le caractère « / » à la place du double point pour les adresses IPv6 lorsqu'il y a risque d'ambiguïté. Cette syntaxe a été étendue dans les versions récentes pour supporter également la notation à base de crochets définie dans le RFC 3986.

Le serveur OpenSSH utilise un seul socket IPv6 sur les systèmes où celui-ci peut accepter les connexions IPv4, ou bien

deux sockets séparés pour les deux versions du protocole si le système n'accepte pas de connexion IPv4 sur un socket IPv6. Le fichier de configuration du serveur permet de forcer la liste des adresses sur lesquelles écoute le serveur.

Le client OpenSSH permet de spécifier le protocole à utiliser pour la connexion, soit par une option sur la ligne de commande, soit dans le fichier de configuration. Par défaut si le noyau supporte IPv6 et si la destination a une adresse IPv6, la connexion est essayée d'abord en IPv6, avec repli sur IPv4 en cas d'échec.

Apache. Les développeurs du serveur Web apache ont attendu une profonde réécriture du code pour la version 2 du serveur avant pour intégrer le support d'IPv6. Des rustines pour les versions 1.3 ont été diffusées par le projet Kame, mais ces modifications sont incomplètes et peuvent causer des problèmes dans certaines configurations. Elles n'ont jamais été adoptées officiellement par la fondation Apache.

La syntaxe utilisée pour la représentation des adresses sous forme numérique a dû, comme pour X et OpenSSH être modifiée afin de lever l'ambiguïté créé par l'utilisation des double-points à la fois comme séparateur à l'intérieur d'une adresse et entre l'adresse et le numéro de port.

Squid. Squid est un proxy/cache Web. Il s'agit d'une application qui aurait un intérêt particulier à supporter correctement IPv6. En agissant à la fois comme serveur et comme client du protocole HTTP, elle pourrait servir de passerelle entre les deux protocoles, permettant à des clients IPv4 seulement d'accéder à des sites IPv6 ou réciproquement.

Malheureusement, l'adaptation de squid pour IPv6 est restée coincée à un stade préliminaire, non satisfaisant pour une utilisation réelle en production. La nouvelle version 3.0 du logiciel qui est en cours de développement n'intégrera vraisemblablement pas le support d'IPv6.

2.5 Politique du choix de protocole

Le choix de la famille d'adresses (IPv4 ou IPv6) est traité de manière très hétérogène dans les applications existantes. En effet, à défaut de mécanisme générique défini dans le RFC 2553, chaque développeur d'application fait son propre choix. Voici un aperçu de quelques comportement d'applications existantes :

- telnet sous Linux Fedora Core 4 ne propose pas d'option pour choisir entre une connexion en v4 ou v6. Si une adresse v6 existe pour la destination, elle est utilisée. Il n'est pas possible de forcer une connexion en v4.
- Mozilla/Firefox ont un paramètre de configuration global (`network.dns.disablev6`) qui permet d'empêcher la recherche d'adresse IPv6 pour les noms dans les URLs, désactivant ainsi de fait l'utilisation d'IPv6.
- Internet Explorer sur Windows XP se connecte en IPv6 si le site distant dispose d'une adresse IPv6. Par contre, il ne sait pas utiliser de proxy ayant une adresse IPv6.
  Il faut remarquer que ces deux applications sont vulnérables au problème des trous noirs dans le DNS IPv6 qui a été évoqué plus haut. Il provoque des lenteurs dans le chargement de certaines pages.
- Comme cela a été vu plus haut, OpenSSH choisi IPv6 par défaut, mais propose les options **-4** et **-6** et offre une possibilité de définition par machine dans le fichier de configuration `ssh_config` (mot clé `AddressFamily`). Par contre, OpenSSH affiche un message d'erreur lorsque la connexion IPv6 échoue avant de tenter IPv4, ce qui peut induire l'utilisateur en erreur.
- Ping et traceroute : utilisent IPv4 sous Solaris par défaut ; l'option **-A inet6** permet d'utiliser icmpv6. Sous Linux et BSD : ping et traceroute sont IPv4 uniquement, les commandes ping6 et traceroute6 utilisent IPv6.

On observe bien que les possibilités dépendent de l'application. Ce qui crée de la confusion pour les utilisateurs et rend difficile le diagnostic pour les services de support technique.

3 Sécurité

IPv6 prend d'avantage la sécurité en compte dans sa conception, mais tous les problèmes de sécurité relevés dans IPv4 au cours des ans (dans les protocoles de niveau supérieur TCP ou UDP) ne sont pas résolus.

IPv6 augmente la complexité des configurations réseau, multiplie les façons de s'adresser à un hôte, et les types d'erreur qui peuvent se produire. Cette complexité nuit directement à la sécurité.

3.1 Adresses IPv4 mappées dans IPv6

Certaines fonctionnalités d'IPv6 introduisent indirectement (et involontairement) de nouvelles vulnérabilités. Les adresses IPv4 mappées dans IPv6 (de la forme ::ffff.a.b.c.d) permettant de représenter une adresse IPv4 dans IPv6 et d'établir une connexion directe d'une machine v4 sur une machine v6 sont l'une des sources de nouveaux problèmes : elles permettent à des machines v4 d'accéder à toute une nouvelle gamme d'adresses qui n'a pas toujours été prise en compte par la politique de sécurité.

On peut distinguer deux types de problèmes :

- Au niveau des protocoles de transport, il est impossible de déterminer si une connexion utilisant une adresse de ce type est une connexion IPv4 qui est traduite en IPv6 dans le noyau de la machine destination, ou si c'est une connexion IPv6 qui contient une adresse de ce type dans son en-tête. Normalement dans tous les cas, la réponse sera envoyée via IPv4 (le noyau traduisant l'adresse mappée de la destination en IPv4).
- Au niveau du contrôle d'accès, si l'on souhaite par exemple bloquer l'adresse IPv4 10.1.2.3, à partir du moment ou l'élément qui fait le filtrage dispose d'une pile IPv6, il ne faudra pas oublier de bloquer également ::ffff.10.1.2.3 si le système accepte les connexions IPv4 dans sa pile IPv6.

Il est donc indispensable que ces adresses soient reconnues

et traitées à part à tous les niveaux. Deux recommandations existent à ce sujet [10, 11] (en particulier elles recommandent de ne jamais utiliser de telles adresses dans les entêtes de paquets circulant sur le réseau) mais ne sont pas encore vraiment prises en compte.

3.2 IPsec sur IPv6 ?

Bien que IPsec fasse partie intégrante d'IPv6 (tout noeud IPv6 doit supporter IPsec), le déploiement d'IPsec sur IPv6 empile en fait deux niveaux de difficulté : celle d'IPsec et des protocoles d'échange de clés associés, additionnée à celle d'IPv6.

Paradoxalement, peu de solutions IPsec disponibles actuellement supportent IPv6. Cela ne semble donc pas une voie de sécurisation utilisée en pratique.

3.3 Bogues et failles dormantes

Enfin, tous les freins évoqués ci-dessus en créent un supplémentaire en laissant le code spécifique à IPv6 dormir sans être vraiment testé. Des bogues, avec peut-être des implications au niveau de la sécurité, ont certainement échappé aux programmeurs. Faute d'une utilisation à une échelle comparable à celle des applications IPv4, ces bogues vont persister pendant un certain temps, rendant ainsi les versions IPv6 des applications d'autant moins fiables aux yeux des intrépides testeurs.

Conclusion

L'état actuel du déploiement d'IPv6, tant en termes d'infrastructure qu'en termes de support dans les applications apparaît donc préoccupant pour l'avenir de la technologie.

Cette situation, combinée à la difficulté d'imposer une politique globale de choix du protocole de transport au niveau des machines, conduit souvent les utilisateurs potentiels d'IPv6 à y renoncer, préférant laisser à d'autres le soin d'essuyer les plâtres.

Or IPv6 ne deviendra utilisable à grande échelle et jouera son rôle dans le développement de l'internet dans les pays émergeants que si un effort important est fait par la communauté et par les fournisseurs de matériels et de logiciels pour trouver des solutions aux problèmes restants.

Sinon IPv6 risque de rester cantonné dans un rôle secondaire de protocole réservé à certaines régions du globe.

Références

[1] Omb : Agencies must use advanced Internet by 2008. Govexec daily briefings `http://www.govexec.com/dailyfed/0605/062905tdpm2.htm`, June 2005.

[2] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins et M. Carney. *RFC 3315 - Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, July 2003.

[3] J. G. Andress. IPv6 : The Next Internet Protocol. *;Login :*, 30(2) :21–28, April 2005.

[4] B. Huffacker. IPv6 BGP Geopolitical Analysis. `http://www.caida.org/analysis/geopolitical/bgp2country/ipv6.xml`, July 2005.

[5] K. Cho, M. Luckie et B. Huffacker. Identifying IPv6 Network Problems in the Dual-Stack World. Dans *SIGCOMM 2004, Network Troubleshooting Workshop*. ACM, September 2004.

[6] G. Huston. *RFC 4159 - Deprecation of "ip6.int"*, August 2005.

[7] R. Gilligan, S. Thomson, J. Bound et W. Stevens. *RFC 2553 - Basic Socket Interface Extensions for IPv6*, March 1999.

[8] J.I. Hagino. *IPv6 Network programming*. Elsevier Digital Press, 2005.

[9] Proposed enhancement to the x window system for ipv6. `http://www.x.org/IPV6_Review.html`, March 2004.

[10] C. Metz et J.I. Hagino. IPv4-Mapped Address API Considered Harmful. Internet Draft `ftp://ftp.itojun.org/pub/paper/draft-cmetz-v6ops-v4mapped-api-harmful-01.txt`, October 2003.

[11] C. Metz et J.I Hagino. IPv4-Mapped Addresses on the Wire Considered Harmful. Internet Draft `ftp://ftp.itojun.org/pub/paper/draft-itojun-v6ops-v4mapped-harmful-02.txt`, October 2003.