Étude des solutions de proxy Neighbor Discovery sécurisées et proposition basée sur la Signature Agility

Tony Cheneau, Maryline Laurent

Étude des solutions de proxy Neighbor Discovery sécurisées et proposition basée sur la Signature Agility

Tony Cheneau (tony.cheneau@it-sudparis.eu)*
Maryline Laurent (maryline.laurent@it-sudparis.eu)*

Résumé :
En 2005, le protocole de Découverte de Voisins (Neighbor Discovery, ND), sur lequel reposent les deux protocoles de mobilité Mobile IPv6 (MIPv6) et Proxy Mobile IPv6 (PMIPv6) a été étendu avec l'extension de sécurité nommée *Secure Neighbor Discovery* (SEND). Cette extension qui fait appel aux adresses générées de manière cryptographique (Cryptographically Generated Addresses, CGA) pose un problème d'incompatibilité dans un scénario spécifique à MIPv6. En effet, d'une part, CGA et SEND donnent la possibilité à un nœud de protéger son adresse, et ce en apposant sa signature électronique à chaque message ND qu'il émet sur le réseau. D'autre part, MIPv6 prévoit dans son mécanisme de *proxy Neighbor Discovery*, que l'adresse du nœud mobile est protégée par le *Home Agent* lorsque le nœud quitte le réseau.
Cet article propose de comparer les différentes solutions permettant l'utilisation conjointe du *proxy Neighbor Discovery* et de SEND. Une seconde partie du document présente une proposition reposant sur l'usage de la Signature Agility dans SEND.

Mots Clés : protection du lien, SEND, CGA, proxy Neighbor Discovery, MIP

1 Introduction

Tout comme le Proxy ARP (Address Resolution Protocol) [Plu82, CMQ87] en IPv4, le protocole de Découverte de Voisins en IPv6 (Neighbor Discovery, ND) [NNSS07] propose une fonctionnalité permettant à un nœud tiers d'envoyer des messages de signalisation de Découverte de Voisins en réponse à des messages qui ne lui sont pas directement adressés. Celle-ci est couramment nommée *proxy Neighbor Discovery* (proxy ND). Cette fonctionnalité est fort utile dans le protocole Mobile IPv6 [JPA04] car elle permet à un nœud mobile de communiquer avec ses nœuds "voisins" sur son réseau mère, même une fois qu'il a quitté ce dernier. D'autres usages, décrits dans le document RFC *Neighbor Discovery Proxies* [TTP06], proposent de partager un préfixe de sous-réseau sur plusieurs liens. Ceci dit, cette fonctionnalité est incompatible avec le protocole de Découverte de Voisins Sécurisées (Secure Neighbor Discovery, SEND) [AKZN05]. En effet, le protocole SEND repose sur des adresses générées de manière cryptographique (Cryptographically Generated Addresses, CGA) qui permettent au nœud propriétaire d'une telle adresse de

*. Institut Télécom, Télécom SudParis, CNRS Samovar UMR 5157, 9 rue Charles Fourier, 91011 Évry, France

prouver la "possession de l'adresse". La difficulté de l'utilisation simultanée de SEND et proxy ND réside dans la preuve de possession de l'adresse qui n'est pas disponible pour les proxys alors que ces derniers doivent protéger l'adresse.

Cette étude a pour but de recenser les différentes solutions connues à ce jour et de les comparer, mais elle n'établit pas de "meilleure solution".

La section 2 présente le protocole de Découverte de Voisins. Cette introduction est complétée par une présentation des différentes solutions de *proxy Neighbor Discovery* dans la section 3. La section 4 introduit la découverte de Voisins sécurisée et indique les points d'incompatibilité avec le *proxy Neighbor Discovery*. Une étude des différentes solutions permettant la mise en place d'un *proxy Neighbor Discovery* sécurisé est proposée à la section 6. Dans la section 6.5, nous proposons une solution alternative basée sur nos précédents travaux sur la Signature Agility [CBLM09]. Finalement, nous concluons à la section 7.

Glossaire

CGA Cryptographically Generated Addresses

NA Neighbor Advertisement

ND Neighbor Discovery (ou *découverte de voisins*)

NS Neighbor Solicitation

RA Router Advertisement

RS Router Solicitation

SEND Secure Neighbor Discovery

2 Protocole de découverte de voisins en IPv6

Le protocole de Découverte de Voisins (NDP) [NNSS07] couvre un ensemble de fonctionnalités qui se limite au voisinage immédiat d'un nœud, c'est-à-dire aux nœuds qui sont situés à un saut (appelés aussi *voisins*). Parmi les fonctionnalités les plus importantes, on peut citer la *résolution d'adresse* (similaire à ARP [Plu82] pour IPv4), la découverte de *Préfixes* et de *Routeurs*. Ces trois fonctionnalités permettent à un nœud de construire automatiquement son adresse IPv6 par la procédure d'*autoconfiguration d'adresse sans état* [TNJ07]. L'automatisation de la construction de l'adresse IPv6 fait aussi appel au mécanisme NDP de *Détection d'Adresse Dupliquée*, et ce afin de ne pas assigner, sur un réseau, une adresse déjà utilisée par un autre nœud.

Ces fonctionnalités sont assurées par cinq messages de type ICMPv6 dont :

- le message *Neighbor Solicitation* (NS) permet à un nœud de demander à un voisin son adresse couche 2 (MAC). Il est notamment utilisé pendant la *résolution d'adresse*. Afin d'effectuer la résolution d'adresse en un seul échange de messages, une requête NS peut aussi contenir l'adresse couche 2 de son émetteur ;
- le message *Neighbor Advertisement* (NA) est envoyé en réponse au message *Neighbor Solicitation*. Il contient l'adresse de couche 2 du nœud dans une option ;
- le message *Router Solicitation* (RS) est émis par les hôtes (c.-à-d. les nœuds qui ne sont pas routeur) pour demander aux routeurs du lien des informations sur le réseau (préfixe sous-réseau, durée de vie du préfixe, ...) ;

– le message *Router Advertisement* (RA) est la réponse des routeurs aux requêtes *Router Solicitation*. Il transporte des informations sur le réseau local, comme le préfixe sous-réseau à utiliser, la valeur de *Maximum Transmission Unit* (MTU), ...

Afin d'optimiser l'utilisation de ses messages, les données sont enregistrées dans différents caches. Par exemple, l'association entre adresse IP et adresse couche 2 est stockée dans un cache appelé *Cache de Voisinage* (Neighbor Cache).

Chacun de ces messages est modulaire et peut être étendu par des options. Par exemple, l'option *Source Link-Layer* et l'option *Target Link-Layer* transportent les adresses de niveau 2 respectivement dans les messages NS et les messages NA. Certains messages peuvent contenir plusieurs fois la même option, c'est le cas du message Router Advertisement qui peut contenir plusieurs options *Prefix Information* si plus d'un préfixe sous-réseau est disponible localement. Bien que les messages NS/NA et RS/RA fonctionnent sur le modèle stimulus/réponse, il est possible que certains messages soient envoyés spontanément. Ainsi, des messages Router Advertisement sont émis régulièrement par les routeurs afin d'annoncer que la validité d'un préfixe est prolongée ou spontanément lorsqu'un nouveau préfixe est disponible sur le lien. Dans les messages spontanés, on notera qu'un changement d'adresse couche 2 sur une interface (remplacement d'une carte réseau) déclenche l'envoi d'un message NA qui met à jour le *Cache de Voisinage* de ses voisins.

3 Scénario d'usage du proxy Neighbor Discovery

Cette partie présente les trois scénarios d'usage du mécanisme proxy Neighbor Discovery. Le lecteur intéressé pourra aussi consulter le document [DCK09] qui offre une approche plus technique.

Dans ce document, nous qualifions de "protégé" un nœud dont un proxy est autorisé et configuré pour utiliser son adresse.

3.1 Proxy Neighbor Advertisement (RFC 4861)

Le document RFC 4861 [NNSS07] est le premier document à décrire le concept général de proxy Neighbor Discovery pour le NDP. Dans ce document, le contexte d'utilisation du proxy est assez vague, il n'est appliqué aucune différence entre un nœud protégé qui ne se trouve sur aucun lien du proxy (comme un nœud joignable uniquement à travers un tunnel par exemple) ou un nœud protégé par des annonces sur un lien du proxy et qui se trouve physiquement attaché à un autre lien du proxy. Dans la pratique, la fonctionnalité de "proxy" est assurée par des routeurs. Elle permet d'émettre des Neighbor Advertisement à la place d'un nœud dans le réseau qui se situe sur un autre lien du proxy.

Pour ce faire, les proxys écoutent les messages émanant des nœuds protégés en se joignant au groupe multicast du *Solicited-Node* (une adresse multicast spécifique seulement écoutée par le propriétaire de l'adresse). De cette façon, les proxys écoutent chaque message NS à destination d'un nœud protégé. Les proxys peuvent dès lors répondre avec des messages de type NA en spécifiant leur adresse de couche 2 dans l'option *Source Link-Layer*. Les nœuds protégés, eux, ne se situent pas sur le même lien et ne recevront donc pas les requêtes NS initiales.

Les messages modifiés par le proxy ont leur drapeau *Override* désactivé. En pratique, cela permet d'éviter qu'une entrée du Cache de Voisinage ne soit écrasé (*race condition*)

par un proxy alors que le nœud protégé se trouve sur le même lien et n'a donc plus besoin de protection.

Un proxy peut aisément déterminer qu'un nœud est connecté sur un lien autre que celui de l'émission d'une requête en consultant le Cache de Voisinage.

Cette approche forme la base des fonctionnalités de proxy utilisées dans MIPv6 [JPA04] (présenté dans la section suivante).

3.2 Mobile IPv6 (RFC 3775)

Mobile IPv6 est un protocole de mobilité pour IPv6. Un *Nœud Mobile* (MN) est associé à une entité nommée *Agent Mère* (Home Agent, HA). Quand le nœud n'est pas en déplacement, (il est dans son réseau mère), il génère et utilise une adresse IPv6 nommée *Home Address*, qui est dérivée du préfixe de sous-réseau (mère) annoncé par le *Home Agent*. Quand le MN quitte le réseau, il génère une nouvelle adresse basée sur le préfixe de sous-réseau du réseau visité. Cette adresse est nommée *Care-of-address* (CoA). Pour se rendre joignable dans ses déplacements, le MN enregistre sa CoA auprès de son HA et construit un tunnel bidirectionnel avec HA. Par la suite, le HA collecte tout le trafic destiné au MN (adressé à sa *Home Address* et le transfère au MN via le tunnel.

Pour permettre au MN de recevoir tout le trafic qui lui est destiné alors qu'il est en déplacement, MIPv6 [JPA04] impose que le HA simule la présence du MN sur le lien. Pour ce faire, le proxy répond aux messages NS destinés au MN (sans même que celui-ci soit contacté) et y place l'adresse couche 2 du proxy dans les NA associés. De la sorte, MIPv6 s'assure que le trafic émanant des nœuds voisins à destination du nœud mobile est redirigé vers le *Home Agent* pour être ensuite transmis au MN au travers du tunnel. Grâce à la signalisation de MIPv6, le HA sait exactement quand le MN se trouve dans un réseau visité et peut en déduire les moments où il doit commencer à agir en tant que proxy pour le compte du MN.

3.3 Neighbor Discovery Proxies (RFC 4389)

Le document RFC 4389 [TTP06] décrit un usage du proxy destiné à émuler les fonctionnalités d'un pont (*bridge*) au niveau 3, comme le décrit la figure 2. Ici, le proxy évite le recours à la traduction d'adresse (NAT) dans le cas où de nombreux liens ascendants se connectent sur un réseau. Il assure aussi la fonction de pont niveau 3 dans un scénario où un pont niveau 2 n'aurait pas été possible (entre deux technologies différentes), et il offre une solution transparente pour partager un sous-réseau à travers plusieurs liens.

Le proxy laisse passer les messages NDP entre plusieurs de ses liens et se contente de modifier au vol le contenu des adresses couche 2 dans les options *Source Link-Layer* et *Target Link-Layer* pour y copier sa propre adresse (voir figure 1). De cette façon, quand deux nœuds échangent des messages NDP au travers du proxy, ils apprennent uniquement les adresses couche 2 du proxy, ce qui amène le trafic échangé entre ces nœuds à transiter par le proxy.

La principale différence avec les usages des sections 3.1 et 3.2 réside dans le proxy qui peut être un routeur n'annonçant aucun préfixe (il est totalement transparent). Cela implique que le proxy peut lui même avoir à modifier des messages de type RA (en sus des messages NS, NA et Redirect) quand une option *Source Link-Layer Address* s'y trouve.

Comme nous le verrons dans la partie suivante, modifier des RA implique que le proxy dispose du même niveau d'autorisation que le routeur qui a originellement émis ce message.

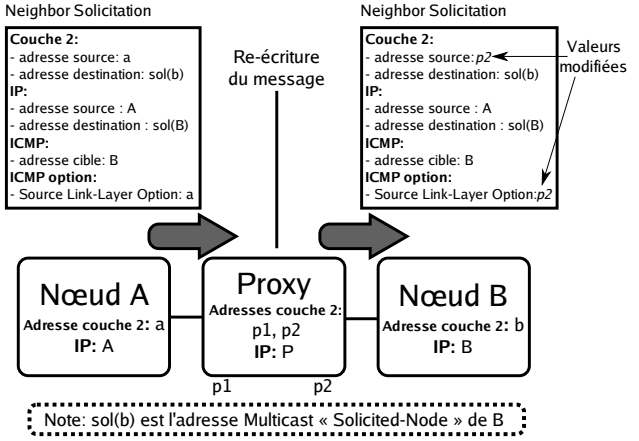

Figure 1: Intervention d'un proxy RFC 4389 durant une procédure de *Résolution d'Adresse*

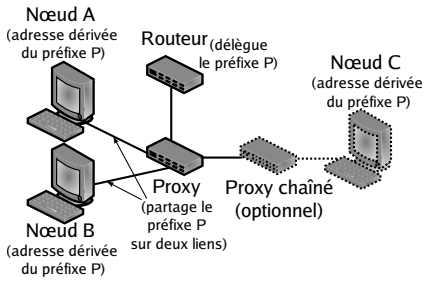

Figure 2: Architecture du proxy implémentant le document RFC 4389

4 Sécuriser la Découverte de Voisins

4.1 Les adresses CGA

Les adresses générées de manière cryptographique (Cryptographically Generated Addresses, CGA) [Aur05] sont un type particulier d'adresse Unicast IPv6 qui lie une clé Publique à la partie *Identifiant d'Interface* d'une adresse. L'algorithme de génération d'une adresse CGA consiste à calculer deux hash. Ces hashs sont fonction de la structure de données "paramètres CGA" (présentée figure 3) et de la valeur d'un paramètre SEC. Cette valeur détermine la robustesse de l'adresse à certaines attaques de type *force brute* sur l'algorithme de hachage.

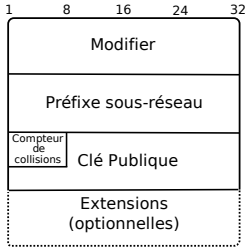

Figure 3: Structure de données "paramètres CGA"

La structure de données "paramètres CGA", présentée figure 3, est le résultat de la concaténation d'un nombre aléatoire de 128 bits nommé *Modifier*, d'un *Préfixe de sous-réseau* associé à l'adresse générée, d'un compteur de collision (pour la procédure de Détection d'Adresse Dupliquée), de la clé *Clé Publique* encodée au format DER et de champs d'extension optionnels (utilisé par la Signature Agility par exemple). Le *Modifier* améliore la sécurité de la CGA en ajoutant de l'aléa dans son processus de création (deux adresses CGA générées à partir de la même Clé Publique sont différentes).

L'algorithme de génération de CGA repose sur la fonction de hachage SHA-1 pour calculer les deux hashs suivants :

– *hash2*, calculé en premier, contient les 112 premiers bits d'un hash calculé sur la structure de données "paramètres CGA" avec un préfixe de sous-réseau et un compteur de collision à 0. Ce hash est recalculé jusqu'à ce que les $16 \times SEC$ premiers bits obtenus soient à 0. À chaque fois que cette condition n'est pas remplie, le *Modifier* est incrémenté de 1. L'objectif est d'éviter une attaque par force brute sur l'adresse

en rendant difficile le réalisation d'une collision sur *hash2* et *hash1* simultanément ;
- *hash1* contient les 64 premiers bit du hash de la structure de données "paramètres CGA". On obtient l'*Identifiant d'Interface* de l'adresse CGA nouvellement créée en remplaçant les trois premiers bits de *hash1* par la valeur de SEC et les bits *U* et *G* par 0.

L'adresse CGA est formée en concaténant le *Préfixe de sous-réseau* avec l'*Identifiant d'Interface*. Avant d'activer cette adresse CGA, le mécanisme de Détection d'Adresse Dupliquée tel que défini dans [TNJ07] est appliqué pour s'assurer qu'aucun autre équipement ne dispose de la même adresse. Si une collision est détectée, le *Compteur de Collision* est incrémenté de 1 et *hash1* est recalculé à partir de la nouvelle structure de données "paramètres CGA". La Détection d'Adresse Dupliquée est réalisée au maximum trois fois. À la quatrième collision, le nœud abandonne la création d'adresse et enregistre une erreur (précisant un problème de configuration ou une attaque sur le réseau).

4.2 Protocole de découverte de voisins sécurisée

La Découverte de Voisins Sécurisée [AKZN05] (Secure Neighbor Discovery, SEND) ajoute une couche de sécurité au NDP pour se prémunir des attaques de type Déni de Services et rejeux (répertoriées dans [NKN04]). SEND ajoute de nouvelles options aux messages NDP et définit de nouveaux messages ICMPv6 [CDG06] pour sécuriser les échanges entre les nœuds.

SEND offre une nouvelle fonctionnalité nommée Authorization Delegation Discovery (ADD) qui permet à un nœud d'authentifier un routeur avant de le charger de router ses paquets. L'idée est ainsi de se prémunir des attaques de routeurs pirates. ADD permet de distinguer une source d'information de routage valide d'une source invalide.

Deux nouveaux messages ICMPv6, *Certification Path Solicitation* (CPS) et *Advertisement* (CPA) mettent en oeuvre cette fonctionnalité ADD. Un nœud envoie un message CPS au routeur pour avoir la preuve qu'il est autorisé à se déclarer "routeur". Le routeur répond avec un CPA contenant le chemin de certification complet jusqu'à une ancre de confiance connue du nœud demandeur. Ce chemin est vérifiable par le nœud. En option, le certificat peut contenir un champ *Prefix Extensions* précisant un préfixe de sous-réseau que le routeur est autorisé à annoncer et/ou re-déléguer (à un autre routeur, plus bas dans la chaîne de confiance).

SEND introduit six nouvelles options pour sécuriser les messages ND, parmi celles-ci :
- l'*Option CGA* contient la structure de données "paramètres CGA" (contenant elle-même la clé Publique associée à l'adresse). Cette option est présente dans tous les messages sécurisés ;
- l'*Option Signature RSA* (RSA Signature Option) contient une signature RSA du message. Elle est présente dans chaque message sécurisé. C'est une option d'importance, car elle permet de grâce à la signature qu'elle contient de prouver la possession de la clé Privée associée à la clé Publique de sa CGA. Et par transitivité, la possession de son adresse CGA ;
- les options *Horodatage* et *Nonce* permettent d'éviter les rejeux. L'option *Horodatage* est présente dans tous les messages. L'option *Nonce* permet de s'assurer qu'une réponse correspond bien à la requête qui l'a déclenchée et n'est donc jointe qu'à des réponses sollicitées.

SEND s'intègre parfaitement à l'*autoconfiguration d'adresse sans état*. Quand un hôte

rejoint un réseau, il commence par envoyer un message RS afin d'obtenir des informations sur le réseau (sous forme d'un RA). Cela lui permet de découvrir le/les sous-réseaux utilisé(s) pour configurer sa (ses) nouvelles adresses. À la réception du message RA d'un routeur, l'hôte peut commencer à vérifier l'autorisation de ce dernier pour le préfixe ou les préfixes qu'il annonce. Il retient sa clé Publique (disponible dans l'*option CGA*) et déclenche le mécanisme ADD. Après un échange de messages CPS/CPA, le nœud vérifie le chemin de certification et la légitimité du routeur pour ce/ces préfixe(s) de sous-réseau. L'hôte construit alors son adresse CGA en utilisant le/les préfixe(s) annoncé(s) par le routeur et peut commencer à communiquer sur le réseau.

4.3 Incompatibilité entre la découverte de voisins sécurisée et le proxy Neighbor Discovery

D'une part, SEND est défini de manière à ce que seul le possesseur de l'adresse puisse signer les messages (cf. section 2). D'autre part, nous avons vu que le proxy génère de nouveaux messages NA (section 3.2) ou modifie des messages existants (section 3.3). Avec le modèle de sécurité actuel, ces deux actions impliquent la connaissance de la clé Privée du nœud protégé, ce qui n'est ni possible, ni réellement intéressant. De plus, l'utilisation décrite dans la section 3.3 modifie les messages émis par des routeurs. Le proxy doit donc bénéficier de la même autorisation que ces derniers.

Nous pouvons donc conclure que SEND doit être adapté pour gérer nativement la fonctionnalité de proxy Neighbor Discovery.

5 Critères de comparaison entre les différentes solutions sécurisées de proxy Neighbor Discovery

La comparaison des solutions décrites à la section 6 fait référence aux critères ci-dessous. Si la description ne fait pas mention de ce critère, cela signifie que la solution se comporte comme les mécanismes d'origine CGA [Aur05] et SEND [AKZN05].

autorisation du nœud indique si le nœud protégé donne lui même l'autorisation au proxy d'agir comme tel ou si ce choix est laissé à une autre entité hiérarchiquement supérieure au nœud. Les critères de décision d'un nœud peuvent être une simple authentification du proxy. De manière générale, le choix d'une autorisation de proxy par le nœud plutôt qu'une autorisation par le proxy lui-même permet d'amoindrir les effets d'une attaque effectuée par un proxy compromis (le nœud pourrait choisir de ne plus l'autoriser).

rétrocompatibilité indique si la modification du protocole est compatible avec les spécifications émises dans le document RFC 3972 (CGA) [AKZN05] et RFC 3971 (SEND) [Aur05]. Le critère de rétrocompatibilité est binaire, mais un degré indiquant la complexité des modifications à fournir peut-être indiqué.

modification d'un nœud tiers indique si un nœud tiers doit être modifié, autant que le nœud qui utilise la fonctionnalité de proxy, moins, ou pas du tout.

anonymat correspond à la possibilité pour un nœud extérieur de différencier les messages émis par le proxy de ceux émis par le nœud protégé. Ce critère revêt son importance. Dans le cas d'un proxy indiquant qu'un nœud a quitté son réseau mère, il est en effet

possible de tracer ce nœud, et de saturer à distance le lien entre un nœud mobile et son Home Agent.

mise en chaîne des proxys indique si un proxy peut émettre un message déjà émis par un proxy. Cela s'avère utile dans le cas de l'utilisation de proxys pour partager un même préfixe de sous-réseau [TTP06] sur plusieurs liens différents.

taille des messages est un critère qui renseigne sur la taille relative des messages par rapport à la proposition de base décrite dans [AKZN05, Aur05]. Elle est, dans beaucoup de cas, liée à la taille de la clé Publique et de la signature des messages.

autorisation après l'autoconfiguration indique que la solution repose sur un mécanisme qui ne peut être exécuté qu'après l'autoconfiguration. Cela permet de distinguer le cas des solutions où l'autorisation du proxy est effectuée par une modification de l'adresse CGA. L'inconvénient de ce genre de solution est que l'ajout de nouveaux proxy n'est pas possible par la suite sans modification de l'adresse CGA du nœud.

6 Différentes solutions de proxy Neighbor Discovery

Le tableau 1 offre une comparaison rapide des solutions décrites ci-dessous. Faute de place, nous ne proposons pas d'analyse comparative détaillée.

6.1 Support du Proxy ND sécurisé pour SEND

Le document [KLB09] propose d'étendre le protocole SEND [AKZN05] en ajoutant un nouveau type d'option de signature nommée *Proxy Signature Option* et en utilisant les attributs *Extented Key Usage* (EKU) des certificats X.509 [CSF+08] de chaque routeur SEND pour expliciter le rôle de proxy (tel que défini dans [KKA09]). Cette option n'est utilisée que par le proxy lorsqu'il protège le nœud. Ainsi, la solution ne peut pas fournir l'anonymat. Le format de l'option est similaire à l'option *RSA Signature Option*. Seul le sens du champ *Digital Signature* est modifié. Il est maintenant généré à partir de la clef Privée du proxy. Pour la vérification, le proxy agissant comme routeur sur le réseau envoie des messages RA (spontanés dans le cas d'un attachement à un lien), qui déclenchent la procédure de *Certificate Path Validation* sur les nœuds qui n'ont pas déjà enregistré la clé Publique du proxy). Par la suite, ces nœuds vérifient le certificat du proxy et y apprennent la clef Publique de ce dernier. Ils peuvent dès lors commencer à vérifier les messages protégés. Quand chacun de ses nœuds connaît la clé Publique d'un proxy qui est dans son voisinage, le chaînage des proxys est possible.

La *Proxy Signature Option* permet de s'affranchir de la vérification de l'adresse CGA (les messages protégés ne transportent pas d'option CGA). La proposition n'apporte par ailleurs aucune modification sur la spécification des CGA [Aur05]. En revanche, elle impose une modification de tous les nœuds afin qu'il puisse vérifier la signature du proxy. Le déploiement des certificats, comme pour SEND [AKZN05] n'est pas spécifié, mais ne change que par la présence d'une extension *EKU* indiquant le rôle de *proxy*.

Secure Proxy ND propose en sus une réponse au problème du partage d'adresse (assimilable à de l'adressage *anycast*) dans le protocole Proxy Mobile IPv6 (PMIPv6) [GLD+08]. Dans ce protocole, le nœud mobile n'est pas modifié pour gérer la mobilité et le réseau s'assure de présenter au nœud mobile une adresse Lien-Local du routeur d'accès constante et un même sous-réseau. Ce qui simule pour le nœud, un effet semblable à un rattachement après mouvement au même point. Après autoconfiguration de la nouvelle adresse

(utilisant [TNJ07]), le mobile réutilise la même adresse et son trafic lui est rerouté. Cette complexité est en partie absorbée au premier saut, où les routeurs d'accès, ici nommés *Mobile Access Gateway* (MAG) partagent tous la même adresse Lien-Local (pour un même nœud mobile). *Secure Proxy ND* résout ici le problème en utilisant une clé Publique et un certificat par MAG et en protégeant chaque message émis par leur adresse "commune" avec l'option de *Proxy Signature Option*.

Le désavantage de la solution en terme de sécurité est l'augmentation du rôle du routeur assurant la fonction de proxy qui devient de plus en plus central : il devient une cible de choix pour un attaquant. On notera ainsi qu'un proxy compromis peut établir une nouvelle attaque de type "homme du milieu". Le document [NKN04] décrit une attaque qui reste valide en utilisant SEND. Un routeur compromis peut siphonner le trafic du réseau en utilisant des messages RA dont les options *Prefix Information* (PIO) indiquent de ne pas communiquer sur le lien (drapeau *L* à 0). Cela a pour effet de rediriger le trafic des nœuds utilisant les préfixes annoncés vers le routeur. Seules les adresses dérivées du préfixe *Lien-Local* (fe80::/64) ne sont pas concernées par cette attaque (car les PIO ne peuvent annoncer le préfixe Lien-Local). La solution *Secure Proxy ND* autorise une nouvelle attaque complétant la précédente. Un proxy peut via son certificat être autorisé à protéger des adresses de Lien-Local. Dès lors, un proxy corrompu peut s'insérer durant la phase de résolution d'adresse de n'importe quel nœud (qu'il requiert ou non une protection) afin de rediriger le trafic vers lui même. Il peut alors monter efficacement une attaque "homme du milieu" sur le préfixe Lien-Local. Des certificats, contraints à des préfixes de sous-réseau de taille 128 (une adresse), peuvent néanmoins être utilisés plus finement pour contrer cette faiblesse.

Ce désavantage est contrebalancé par un grand avantage intrinsèque des solutions basées sur les certificats : la possibilité d'inclure une date de péremption dans le certificat et de le révoquer. Ce qui permet d'annuler les droits d'un serveur compromis.

Autre aspect de l'utilisation des certificats dans cette solution, il est possible de rajouter des proxy dans le réseau après que les nœuds aient été déployés et configurés pour peu que ceux-ci aient été fourni avec la même ancre de confiance que celle des certificats des proxy.

Cette solution est en cours de normalisation par le groupe de travail "CGA and SEND maIntenance" (CSI) au sein de l'IETF.

6.2 Délégation de droits pour l'annonce ND

Le document [DCK09] catalogue les différentes approches pour permettre le proxy ND conjointement à l'utilisation de SEND [AKZN05]. Parmi celles-ci, l'approche par "délégations d'autorisation" (Authorization Delegation) esquissée dans le document [NA02] propose à un nœud de désigner un proxy et de lui déléguer le droit de signer les messages de signalisation ND.

Cette approche peut être une simple forme de signature d'éléments publics du proxy de la part du nœud protégé ou peut être plus complète en proposant un certificat généré par le nœud et signé avec sa clef Privée. L'approche par certificat permettant de limiter la durée de vie du certificat et/ou de le révoquer (il reste encore à déterminer ici comment les nœuds pourront contacter la liste de révocation pour les certificats autosignés par le nœud). Afin de faire confiance à un proxy, on peut imaginer que celui-ci dispose d'un certificat indiquant son rôle comme pour la solution précédente [KKA09].

Cette solution, moins centralisée que la précédente, diminue l'intérêt que peut avoir un attaquant à corrompre un proxy, celui-ci ne pouvant nuire qu'aux nœuds l'ayant autorisé au préalable. Elle semble n'entraîner que des modifications sur SEND [AKZN05] et permet d'ajouter de nouveaux proxys une fois l'adresse configurée.

Le chaînage des proxys est possible lorsque l'on autorise les différents proxys à se redéléguer les droits qu'ils ont obtenus des nœuds qu'ils protègent. On peut ainsi obtenir un certificat émanant d'un nœud qui autorise un proxy qui lui même signe un certificat pour un autre proxy, lui accordant aussi les droits d'agir comme proxy pour le nœud initial.

6.3 Solution à base de signature en anneau

Les solutions présentées section 6.1 et 6.2 n'offrent pas la confidentialité de la localisation. Pour pallier ce problème, les auteurs des documents [KWRG06] et [KG05] proposent une solution basée sur l'algorithme de "signature en anneau" Rivest-Shamir-Tauman (RST) [RST01]. La "signature en anneau" diffère de la signature de groupe par l'absence de manager de groupe (dont le rôle est central car il génère puis distribue la clef) et offre l'anonymat à celui qui signe le message. C'est à notre connaissance la seule solution qui propose l'anonymat [1] de l'entité qui signe les messages ND.

Le document [KG05] explicite le côté technique de la solution : celle-ci modifie sur tous les nœuds (même ceux non protégés) le comportement par défaut de SEND [Aur05] mais aussi, bien que mineur, l'utilisation des CGA [AKZN05]. Une adresse, maintenant nommée "multi-key CGA" (ou M-CGA), est créée à partir de la clé Publique RSA du nœud à protéger et de celle(s) du ou des proxys. Cela est possible suite à un échange de CPS/CPA entre le nœud et le(s) proxy(s) où le nœud protégé apprend le certificat du/des proxys et où la clef de chaque proxy est extraite à partir du certificat. C'est en choisissant de faire confiance à un certificat pour former une M-CGA que le nœud autorise le proxy à agir comme tel. De ce point de vue, il est donc du ressort du certificat, à l'image de la solution présentée en section 6.1, de prouver que le routeur qui assure la fonction de proxy a été autorisé par une entité de confiance à agir comme tel. On notera par ailleurs que la taille de la clé Publique RST ainsi que la taille de la signature sont linéaires en fonction du nombre de membres/clés (et donc du nombre d'entités autorisées à protéger le nœud).

L'article [KWRG06] offre une comparaison des performances entre l'utilisation de la signature RSA et l'utilisation d'RST. Lorsque deux membres composent le groupe (le nœud protégé et le proxy), la génération de signature RSA et RST est de rapidité équivalente. La vérification de signature, elle, est deux fois plus lente lors de l'utilisation de l'algorithme RST. Dans le cas général, le ralentissement est peu pénalisant. Par la suite, lorsque le nombre de membres dans le groupe augmente, comme prévu, la durée nécessaire pour la signature/vérification augmente linéairement.

D'un point de vue configuration, aucune configuration n'est nécessaire à condition que le nœud soit au préalable averti qu'il doit construire une M-CGA. C'est le cas lors de l'utilisation de MIPv6 [JPA04]. Dans le cas de l'utilisation de [TTP06], une configuration manuelle du nœud sera plus probable, car celui-ci n'aura aucun moyen a priori de savoir qu'il se trouve derrière un proxy et qu'il doit générer une M-CGA. La possibilité de mise en chaîne sera problématique, puisque le nœud de manière standard n'apprend pas la clé Publique des routeurs qui sont situés à plus d'un saut.

1. anonymat toutefois relatif puisque l'entité qui signe prouve qu'elle fait partie du groupe

À noter toutefois que même si l'algorithme RST fournit l'anonymat du signataire, elle ne protège pas la fuite d'information via l'émission de messages de type ND. Ainsi, il est possible en observant un changement dans l'option *Source Link-Layer Address* et *Target Link-Layer Address*, de déterminer que le proxy commence à protéger une adresse (redirige le trafic du nœud vers lui même) et donc que le nœud mobile vient de quitter le réseau.

6.4 Utilisation de relation "symbiotique" pour protéger le Proxy ND

Le document [HN09] décrit la création d'une relation "*forte mais distante*" que les auteurs appellent relation symbiotique (*Symbiotic Relationship*). Cette relation est décrite comme une relation unidirectionnelle entre deux nœuds A et B. En créant cette relation, A permet à B de prouver "sa relation" avec A. Cette preuve permet ainsi à B de prouver qu'il a une autorisation de A à une tierce partie (ici, l'autorisation d'agir comme proxy).

D'un point de vue technique, la procédure de génération de CGA décrite dans [Aur05] est légèrement modifiée, restant compatible avec les nœuds implémentant le protocole CGA "standard" [Aur05]. Le champ *Modifier* qui est normalement un nombre de 128 bits généré de manière aléatoire, est ici généré à partir des 128 premiers bits du hash d'une ou plusieurs clés Publiques et d'un nombre aléatoire. Ces clés Publiques appartiennent aux routeurs d'accès que le nœud autorise à agir comme proxy. Elles sont apprises lorsque le nœud rejoint le lien, dans les messages de type RA sécurisés.

Par la suite, SEND [AKZN05] est modifié pour qu'un nœud souhaitant prouver sa relation signe les messages avec sa clé Privée et joint au message l'ensemble des clés Publiques et le nombre aléatoire utilisés pour la création de l'adresse. Ces derniers paramètres permettent au nœud recevant le message de re-dériver l'adresse CGA du nœud protégé et de vérifier la relation entre le proxy et le nœud protégé.

Bien que le document soit assez vague à ce sujet, il semble que la modification apportée au mécanisme de CGA n'est pas totalement transparente. En effet, comme décrit dans 4.1, le *Modifier* a initialement une valeur aléatoire par la suite incrémentée afin de construire une valeur de *hash2* respectant la valeur du paramètre *SEC*. Lorsque *SEC* vaut 0, l'approche actuelle ne pose pas de problème, mais lorsque *SEC* vaut 1, il n'est plus possible de construire l'adresse comme indiqué. L'incrémentation du *Modifier* casserait le lien avec la ou les clés Privées des proxys. Une solution que nous proposerions serait de stocker ce nouveau *Modifier* contenant la/les clés Publiques dans un champ d'extension de la structure de données "paramètres CGA". Cette solution rendrait la procédure transparente et n'occuperait que 20 octets supplémentaires.

La solution propose également une forme d'anonymat assez faible en proposant au nœud de cacher sa présence en laissant le proxy protéger son adresse, même lorsqu'il se trouve sur le lien. Cette forme d'anonymat, comme celle de la solution précédente, est vulnérable à un type de surveillance des messages ND visant à capturer des variations d'adresses de couche 2.

6.5 Solution basée sur la Signature Agility

Nous proposons ici une nouvelle approche, basée sur la Signature Agility telle que décrite dans le document [CBLM09] et dans les documents techniques [CLMSV09] et [CLSM09] [2].

2. La version des documents référencée ici s'est vu retirer quelques fonctionnalités dans la nouvelle version dans un but de simplification pour la normalisation et ne permettent plus le support tel que décrit dans ce document

L'idée motivant la Signature Agility est le besoin de pouvoir migrer vers de nouveaux algorithmes lorsque RSA et SHA-1, actuellement intégrés en dur dans CGA [Aur05] et SEND [AKZN05], ne sont pas adaptés (problèmes de sécurité récents de SHA-1, lourdeur de RSA,...). Ces travaux ont été motivés par le gain de performance obtenu en utilisant les courbes elliptiques [CBL09] dans SEND [CLSV09].

Nous pensons ici qu'il est possible de réutiliser nos travaux sur la Signature Agility. Dans ces travaux, nous avons étendu les adresses CGA afin qu'il soit possible dans la structure de données "paramètres CGA" de stocker des clés Publiques supplémentaires. La partie SEND est elle aussi modifiée : l'option de *Signature RSA* est changée en *Signature Universelle*, où il est possible d'indiquer la clé à laquelle se réfère le nœud dans la structure de données "paramètres CGA". Le comportement initial était pour un nœud de choisir la clé en testant les capacités de son interlocuteur afin d'utiliser une clé appropriée. Ici, nous réutilisons la possibilité de stocker plusieurs clés Publiques dans la CGA et nous proposons que l'une de ses clefs appartienne au nœud et qu'une ou plusieurs autres appartiennent à un ou plusieurs proxys.

Le nœud lors de l'autoconfiguration sans-état envoie un message RS à destination de tous les routeurs. Ceux-ci répondent avec un message RA. À la réception de ces messages, le nœud va déclencher la procédure de *Certificate Path Validation*. Après un bref échange CPS/CPA, le nœud peut décoder le certificat de chaque routeur où se trouve indiquée, pour les routeurs assurant également le rôle de proxy ND, l'extension EKU correspondante [KKA09]. La clé Publique de chaque proxy valide est extraite du certificat, puis est stockée dans un champ d'extension de la structure de données "paramètres CGA" [CLMSV09]. L'adresse CGA est ensuite construite selon le mécanisme standard proposé dans [Aur05]. Par la suite, le nœud peut signer normalement son adresse en utilisant la clé Publique qu'il possède et le proxy peut faire de même en utilisant la clé Privée associée à la clé Publique de son certificat.

L'intérêt principal de notre solution réside dans la réutilisation de la Signature Agility. En effet, les modifications au niveau des nœuds à protéger et des proxy sont mineures et concernent principalement la création de l'adresse. Et pour les nœuds tiers ne nécessitant pas de protection, il n'y a aucune modification à effectuer pour qu'ils puissent vérifier les messages émis par les proxys.

Les performances peuvent bénéficier de l'utilisation d'ECC, comme présenté en [CBLM09] et [CBL09]. Notre approche ne souffre pas de problème de performance quand le nombre d'entités autorisés à agir comme proxy augmente, contrairement à la solution proposée en 6.3. La limitation de notre solution réside dans le nombre maximal de proxys qui peuvent être autorisés simultanément pour un nœud. Nous avons calculé qu'une adresse CGA composée de onze clés Publiques ECC de 256 bits n'excède pas la valeur minimale de *Maximum Transmission Unit* (MTU) de 1280 octets imposée par le protocole IPv6 [DH98]. Cela implique qu'une adresse ne peut être partagée qu'avec onze nœuds au maximum. Le faible nombre de scénario nécessitant cet intense partage d'adresse nous permet néanmoins de rester confiant quant à la pertinence de notre solution.

Il n'est actuellement pas possible d'étendre l'autorisation d'agir comme proxy à de nouveau proxys après création de l'adresse. Il serait néanmoins possible, bien que plus complexe, de générer l'adresse CGA à partir de clés Publiques nouvellement crées mais inutilisées et ensuite de les distribuer aux nouveaux proxys après que ceux-ci aient été authentifiés. Mais cela entraîne une complexification du protocole que nous ne souhaitons

Solution	le nœud autorise le proxy	modifié pour la solution	modification des nœuds tiers	anonymat	mise en chaîne des proxy	autorisation après auto-configuration
Proxy ND sécurisé	non	SEND	pour les messages émis du proxy	non	possible	possible
Approche par délégation d'autorité	oui	SEND	pour les messages émis du proxy	non	possible	possible
Utilisation de la signature en anneau	oui	CGA et SEND	obligatoire	oui	difficile	impossible
Lien symbiotique	oui	CGA et SEND	pour les messages émis du proxy	oui	difficile	impossible
Basé sur la Signature Agility	oui	repose sur la Signature Agility	non quand il gère la Signature Agility	non	difficile	impossible

Table 1: Comparaison des différentes solutions

pas.

6.6 Positionnement de la solution basée sur la Signature Agility par rapport aux autres solutions

Nous proposons, comme les solutions des sections 6.2, 6.3 et 6.4, une solution décentralisée, où l'autorisation d'agir comme proxy est dépendante du nœud. Pour nous, il s'agit d'un critère important qui diminue sensiblement l'intérêt que peu avoir un attaquant à corrompre le proxy et supprime l'attaque "homme du milieu" présentée dans la section 6.1. En supposant que la Signature Agility soit déployée sur les nœuds, il n'y a pas besoin des modifications lourdes sur CGA et SEND contrairement aux solutions vues en section 6.3 et 6.4. Comme la mise en place de l'autorisation du proxy est effectuée durant l'auto-configuration, notre proposition ne requiert pas l'utilisation de messages de signalisation supplémentaires, au contraire de l'approche basée sur la délégation d'autorité (section 6.2).

Notre solution n'est pourtant pas exempte de défauts. Parmi ceux-ci, relevons qu'elle ne supporte pas l'anonymat (section 6.3), ne permet pas l'ajout de nouveau proxy (sections 6.1 et 6.2).

7 Conclusion

À la date de rédaction de cet article, l'IETF est sur le point d'achever la normalisation de la solution basée sur l'autorisation du proxy par certificat (section 6.1). Nous proposons ici une solution différente se basant sur l'autorisation du nœud et la Signature Agility. À condition que la Signature Agility soit déployée sur tous les nœuds d'un réseau, notre

solution propose une alternative simple, viable et facile à déployer en comparaison aux autres solutions déjà proposées.

Nos travaux futurs porteront sur un travail d'implémentation de la solution proposée, en nous fondant sur l'outil *Scapy6*, un binding Python de bibliothèque *NetFilter Queue* et l'implémentation *NDprotector* [3].

Références

[AKZN05] J. Arkko, J. Kempf, B. Zill, and P. Nikander. SEcure Neighbor Discovery (SEND). RFC 3971, Internet Engineering Task Force, March 2005.

[Aur05] T. Aura. Cryptographically Generated Addresses (CGA). RFC 3972, Internet Engineering Task Force, March 2005. Updated by RFCs 4581, 4982.

[CBL09] T. Cheneau, A. Boudguiga, and M. Laurent. Significantly improved performances of the cryptographically generated addresses thanks to ECC and GPGPU. *Computers & Security*, In Press, Corrected Proof, 2009.

[CBLM09] T. Cheneau, A. Boudguiga, and M. Laurent-Maknavicius. Amélioration des performances des adresses CGA et du protocole SEND : étude comparée de RSA et d'ECC/ECDSA. June 2009.

[CDG06] A. Conta, S. Deering, and M. Gupta. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443, Internet Engineering Task Force, March 2006.

[CLMSV09] T. Cheneau, M. Laurent-Maknavicius, S. Shen, and M. Vanderveen. Support for Multiple Signature Algorithms in CryptographicallyGenerated Addresses (CGAs). Internet-Draft draft-cheneau-csi-cga-pk-agility-00, Internet Engineering Task Force, October 2009. Work in progress.

[CLSM09] T. Cheneau, M. Laurent, S. Shen, and M. MichaelaVanderveen. Signature Algorithm Agility in the Secure Neighbor Discovery (SEND)Protocol. Internet-Draft draft-cheneau-csi-send-sig-agility-00, Internet Engineering Task Force, October 2009. Work in progress.

[CLSV09] T. Cheneau, M. Laurent, S. Shen, and M. Vanderveen. ECC public key and signature support in Cryptographically GeneratedAddresses (CGA) and in the Secure Neighbor Discovery (SEND). Internet-Draft draft-cheneau-csi-ecc-sig-agility-00, Internet Engineering Task Force, October 2009. Work in progress.

[CMQ87] S. Carl-Mitchell and J.S. Quarterman. Using ARP to implement transparent subnet gateways. RFC 1027, Internet Engineering Task Force, October 1987.

[CSF+08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, Internet Engineering Task Force, May 2008.

[DCK09] G. Daley, J. Combes, and S. Krishnan. Securing Neighbor Discovery Proxy Problem Statement. Internet-Draft draft-ietf-csi-sndp-prob-01, Internet Engineering Task Force, March 2009. Work in progress.

3. `http://amnesiak.org/NDprotector/`

[DH98] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, Internet Engineering Task Force, December 1998.

[GLD+08] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil. Proxy Mobile IPv6. RFC 5213, Internet Engineering Task Force, August 2008.

[HN09] W. Haddad and M. Naslund. On Secure Neighbor Discovery Proxying Using 'Symbiotic' Relationship. Internet-Draft draft-haddad-csi-symbiotic-sendproxy-01, Internet Engineering Task Force, July 2009. Work in progress.

[JPA04] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. RFC 3775, Internet Engineering Task Force, June 2004.

[KG05] J. Kempf and C. Gentry. Secure IPv6 Address Proxying using Multi-Key Cryptographically Generated Addresses (MCGAs). Internet-Draft draft-kempf-mobopts-ringsig-ndproxy-02, Internet Engineering Task Force, August 2005. Work in progress.

[KKA09] S. Krishnan, A. Kukec, and K. Ahmed. Certificate profile and certificate management for SEND. Internet-Draft draft-krishnan-cgaext-send-cert-eku-03, Internet Engineering Task Force, March 2009. Work in progress.

[KLB09] S. Krishnan, J. Laganier, and M. Bonola. Secure Proxy ND Support for SEND. Internet-Draft draft-ietf-csi-proxy-send-01, Internet Engineering Task Force, July 2009. Work in progress.

[KWRG06] J. Kempf, J. Wood, Z. Ramzan, and C. Gentry. IP Address Authorization for Secure Address Proxying Using Multi-key CGAs and Ring Signatures . In *Advances in Information and Computer Security*, pages 196–211. Springer, 2006.

[NA02] P. Nikander and J. Arkko. Delegation of signalling rights. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, *Security Protocols Workshop*, volume 2845 of *Lecture Notes in Computer Science*, pages 203–214. Springer, 2002.

[NKN04] P. Nikander, J. Kempf, and E. Nordmark. IPv6 Neighbor Discovery (ND) Trust Models and Threats. RFC 3756, Internet Engineering Task Force, May 2004.

[NNSS07] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor Discovery for IP version 6 (IPv6). RFC 4861, Internet Engineering Task Force, September 2007.

[Plu82] D. Plummer. Ethernet Address Resolution Protocol : Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 0826, Internet Engineering Task Force, November 1982.

[RST01] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT '01 : Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*, pages 552–565, London, UK, 2001. Springer-Verlag.

[TNJ07] S. Thomson, T. Narten, and T. Jinmei. IPv6 Stateless Address Autoconfiguration. RFC 4862, Internet Engineering Task Force, September 2007.

[TTP06] D. Thaler, M. Talwar, and C. Patel. Neighbor Discovery Proxies (ND Proxy). RFC 4389, Internet Engineering Task Force, April 2006.