RFC 6071 : IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 8 février 2011

Date de publication du RFC : Février 2010

http://www.bortzmeyer.org/6071.html

Le protocole de sécurité IPsec a connu plusieurs versions, et de nombreuses extensions. Le programmeur qui voudrait aujourd'hui le mettre en œuvre, l'administrateur réseaux qui voudrait le déployer, doivent souvent se gratter le crâne assez longtemps, uniquement pour identifier les normes pertinentes. Pour leur faciliter la tâche, l'IETF produit un RFC spécial, la « carte routière » qui est uniquement une liste commentée des RFC qui s'appliquent à leur cas. La précédente version de la carte d'IPsec était le RFC 2411 [1], que notre RFC 6071 remplace. Si vous voulez comprendre IPsec et IKE et que vous ne savez pas par où commencer, ce RFC 6071 pourra vous servir de guide dans la jungle des normes sur IPsec.

La section 1 commence par fournir un résumé bien utile d'IPsec et de ses protocoles auxiliaires comme IKE. IPsec vise à fournir un service de sécurité à tout service Internet, sans modifier les applications (contrairement à TLS). Si l'échange de paquets entre deux machines est protégé par IPsec, toutes leurs communications pourront être authentifiées et/ou confidentielles. On peut utiliser IPsec de bien des façons, mais les plus courantes aujourd'hui sont de créer un tunnel sécurisé entre deux sites ou bien d'établir un VPN entre un site central et des machines en promenade.

IPsec utilise la cryptographie et, pour cela, a besoin de clés. On peut les gérer de plusieurs façons (aujourd'hui, c'est souvent fait manuellement), mais la méthode standard pour la gestion dynamique des clés est IKE.

La section 2 de notre RFC classe les normes IPsec en pas moins de sept groupes : l'architecture, le protocole ESP (qui fournit authentification et confidentialité), le protocole AH (qui ne fournit que l'authentification), les algorithmes cryptographiques, les algorithmes combinés (qui indiquent comment faire

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc2411.txt

fonctionner ensemble les différentes pièces), les algorithmes de préservation de l'intégrité des paquets et enfin IKE.

Du fait de la longue histoire d'IPsec, plusieurs versions de ce protocole coexistent. La version actuelle est IPsec v3 mais IPsec v2 est encore fréquemment rencontré dans la nature. La section 2.1.1 résume leurs différences. Par exemple, le protocole AH (RFC 4302), dont l'utilité est douteuse, puisque ESP fournit les mêmes services (tels que perçus par l'application), n'est plus obligatoire. AH existait en raison des restrictions à l'exportation de logiciels cryptographiques en dehors des États-Unis et de restrictions d'usage moyenâgeuses dans certains pays comme la France. Aujourd'hui que ces restrictions sont assouplies, AH ne fait que générer de la complexité supplémentaire, comme l'exposait la fameuse critique de Niels Ferguson et Bruce Schneier, « "*A Cryptographic Evaluation of IPsec*" <http://www.schneier.com/paper-ipsec.html> ».

IKE a aussi plusieurs versions qui coexistent (section 2.3). La version actuelle est la v2, mais la v1 est toujours en service. La v2 se signale (section 2.3.1) entre autres par l'intégration d'EAP, l'utilisation d'ESP pour protéger IKE lui-même (avant, c'était un protocole spécifique qui s'en chargeait), par de meilleures protections contre les DoS, notamment en permettant au répondeur de ne pas allouer trop de ressources avant d'avoir authentifié l'appelant.

Outre les RFC sur IPsec et IKE, présentés plus loin, en section 3, il existe plusieurs registres IANA sur ces protocoles (section 2.4), notamment :

— Le registre des paramètres IKE <https://www.iana.org/assignments/ikev2-parameters>,
— Le registre des algorithmes de cryptographie <https://www.iana.org/assignments/crypto-suites>

La section 3 décrit les documents à lire si on veut programmer IPsec. Je ne mentionne ici que ceux de l'IPsec récent (IPsec v3) mais notre RFC 6071 décrit aussi (section 3.1.1) ceux de la précédente version d'IPsec, autour du RFC 2401.

Le cœur de IPsec v3 est décrit dans trois RFC (section 3.1.2), le RFC 4301 sur l'architecture générale, le RFC 4302 sur AH et le RFC 4303 sur ESP. Autour de ce cœur, on trouve plusieurs ajouts comme la possibilité d'encapsuler IPsec dans UDP (RFC 3948) ou bien la possibilité de chiffrer sans arrangement préalable entre les parties (RFC 4322), par exemple en récupérant les clés dans le DNS (avec protection par DNSSEC bien sûr).

Sans être des ajouts à IPsec, il existe plusieurs RFC qui discutent de questions générales (section 3.3) comme la traversée des équipements NAT par IPsec (RFC 3715) ou les suggestions pour évaluer si IPsec est une solution adaptée (RFC 5406, **uniquement** pour le vieil IPsec v2).

IKE est traité de la même façon en section 4 : les documents de base pour la vieille - mais encore utilisée - version et pour la version actuelle, la v2 (le principal document étant le RFC 4306, suivi du document de clarification RFC 4718) puis les ajouts.

La section 5 est entièrement consacrée au délicat problème des algorithmes de cryptographie. Comme la cryptanalyse progresse sans cesse, envoyant dans les poubelles de l'histoire des algorithmes peu sûrs, et poussant à l'introduction de petits nouveaux, il est important qu'un protocole de sécurité comme IPsec puisse suivre l'état de l'art sans avoir besoin d'une révision. En nombre de RFC, c'est de loin ce secteur qui contribue le plus à la taille des normes IPsec. Pour que, dans cette variété de protocoles, deux mises en œuvre d'IPsec aient une chance de communiquer (section 5.1), il faut qu'il y ait un ensemble d'algorithmes communs. Dans IPsec, les algorithmes sont marqués comme obligatoires (toute mise en œuvre d'IPsec doit les implémenter) ou facultatifs. Le RFC 7321 définit les exigences auxquelles doivent obéir les algorithmes utilisés par IPsec, le RFC 4307, celles pour IKE.

La section 5.2 décrit ensuite les algorithmes de chiffrement eux-mêmes, par exemple l'obligatoire NULL (qui ne chiffre rien et ne protège donc rien, RFC 2410), l'obligatoire AES-CBC (RFC 3602, le facultatif Camellia (RFC 4312), etc.

Si la section 5.2 listait les algorithmes de chiffrement, la 5.3 décrit ceux utilisés pour le contrôle d'intégrité. On y trouve HMAC-SHA1 (RFC 2404), dont la mise en œuvre est obligatoire, mais aussi la famille SHA-2 (optionnelle, cf. RFC 4868), et même MD5 (RFC 2403) pas conseillé mais pas interdit non plus, les multiples vulnérabilités de MD5 n'affectant apparemment pas (pour l'instant) son usage en HMAC (RFC 4835).

Certains algorithmes peuvent fournir à la fois le chiffrement et l'intégrité, et la section 5.4 décrit ces combinaisons. Ainsi, AES en mode CCM CBC-MAC permet d'assurer les deux tâches (RFC 4309).

Pas de cryptographie sans nombres aléatoires (section 5.5), par exemple pour générer les clés de sessions de IKE. C'est donc une nouvelle série d'algorithmes qui doit être spécifiée.

Le zoo des algorithmes de cryptographie utilisés devient donc d'une taille impressionnante et on peut se demander si deux pairs IPsec trouveront une combinaison d'algorithmes qu'ils peuvent utiliser tous les deux ! La section 5.6 expose donc les **suites**. Une suite est un ensemble cohérent d'algorithmes qui simplifie le travail du programmeur : si on implémente toute la suite, on a un groupe d'algorithmes qui interopérera avec les autres utilisateurs de la même suite. En outre, il est plus facile de se référer à une suite qu'à une longue liste d'algorithmes. C'est ainsi que le RFC 4308 décrivait deux suites, « VPN-A », qui inclut 3DES, HMAC-SHA-1, et un mode Diffie-Hellman et « VPN-B » qui inclut AES-CBC, AES-XCBC-MAC et un mode Diffie-Hellman plus long. On peut ainsi choisir tout un ensemble d'algorithmes avec un seul bouton dans la configuration (« "*Use VPN-B : Yes*" »). Le RFC 4869 a ajouté quatre autres suites, tirées de normes « Suite B » de la NSA. Elles incluent la famille SHA-2.

Tiens, puisque j'ai parlé de Diffie-Hellman, mécanisme pour permettre à deux pairs de trouver des clés secrètes communes pour pouvoir communiquer, la section 5.7 décrit les RFC qui mentionnent cette méthode dans le contexte d'IPsec. Ainsi, le RFC 5903 est la dernière norme pour l'utilisation de courbes elliptiques dans le cadre Diffie-Hellman, et le RFC 5114 définit de nouveaux groupes pour cet algorithme.

Dans le monde réel, IPsec a connu de nombreux usages qui dépassaient parfois le strict cadre de ce qui était normalisé. La section 7 décrit certains de ces usages, qui étendent et facilitent l'utilisation d'IPsec. Ainsi, les RFC 3586 et RFC 3585 se penchent sur la question de définitions formelles des politiques de sécurité IPsec, le RFC 4807 décrit une MIB pour la gestion IPsec (il semble très peu utilisé), etc.

Une extension intéressante a été BTNS ("*Better-Than-Nothing Security*", section 7.5) qui a introduit le concept de « la plus mauvaise sécurité est celle qu'on ne déploie pas » dans le monde IPsec. Celui-ci était réputé pour sa rigidité et son strict respect des bons principes techniques de sécurité. Ainsi, les protocoles IPsec imposaient une authentification des deux pairs, sans laquelle une attaque de l'intermédiaire était possible. Très bonne idée sauf que, en pratique, cette authentification nécessitait une lourde infrastructure d'identité (par exemple à base d'IGC) et qu'elle a, en pratique, pas mal freiné le déploiement d'IPsec. D'où cette évolution récente (2008) d'accepter un mode sans authentification, BTNS, qui permet d'établir une session IPsec avec des gens qu'on ne connait pas (pensez à un serveur Web, qui ne connait évidemment pas chacun de ses clients à l'avance). Les RFC 5386 et RFC 5387 décrivent BTNS.

Le talon d'Achille de tout protocole de cryptographie est souvent dans la gestion des clés. Où les stocker, comment garantir que la clé privée est en sécurité, que la clé publique est bien authentique ? Pour la seconde question, la garantie que la clé publique est bien la bonne, la section 7.8 rappelle le mécanisme

IPSECKEY qui permet de stocker les clés publiques dans le DNS (RFC 4025). Ce mécanisme est resté largement théorique à ce jour (après tout, le DNS n'offrait typiquement que peu de garanties de sécurité) mais la signature de la racine <http://www.bortzmeyer.org/la-racine-commence-signature.html> avec DNSSEC le 15 juillet 2010 lui a redonné de l'actualité. Il est désormais possible de valider une clé trouvée dans le DNS.

Le programmeur qui implémente IPsec doit se rappeler que c'est un mécanisme général, situé en desous de la couche transport. Normalement, il est invisible aux applications. Toutefois, certains protocoles intermédiaires font un usage spécifique d'IPsec et la section 8 en donne une liste. Ainsi (section 8.2) OSPF, depuis la version 3 (RFC 4552 et RFC 5340), a remplacé son ancien mécanisme d'authentification spécifique par IPsec. OSPF fonctionnant souvent en diffusion restreinte, cela ne permet pas IKE impose l'usage de clés manuelles. En pratique, ce mécanisme de sécurisation semble très peu déployé et le passage à IPsec, trop complexe par rapport à l'ancienne méthode, a donc **réduit** la sécurité de OSPF (en sécurité, comme souvent dans d'autres domaines de l'ingéniérie, le mieux est l'ennemi du bien...)

Autre protocole de routage qui aurait bien besoin de davantage de sécurité, BGP. La section 8.6 décrit ce que IPsec peut lui apporter. Le RFC 5566 décrit ainsi une méthode pour protéger les session BGP. En pratique, elle semble quasi-inutilisée, les opérateurs préférant le RFC 2385 (ou peut-être demain son successeur, le RFC 5925). À noter que le plus gros problème de BGP, en pratique, n'est pas l'authentification du pair, mais la confiance qu'on accorde à ce qu'il transmet <http://www.bortzmeyer.org/pakistan-pirate-youtube.html>. Et, là, IPsec ne peut rien.

Autre consommateur d'IPsec, HIP (section 8.3 et RFC 7401). HIP est un protocole de séparation de l'identificateur et du localisateur <http://www.bortzmeyer.org/separation-identificateur-localisat
html>. Il introduit donc des failles potentielles liées à l'indirection supplémentaire, par exemple d'un méchant essayant de rediriger un trafic important vers une victime en faisant croire que l'adresse IP de la victime est la sienne. N'ayant pas de mécanisme de sécurité propre, HIP délègue la protection de son trafic à IPsec (RFC 5202 et RFC 5206).

IPsec garantissant l'intégrité du trafic IP, il peut rentrer en conflit avec des protocoles qui modifient les paquets, pour de bonnes raisons, par exemple la compression. D'où le nombre de RFC sur l'interaction entre IPsec et ROHC (section 8.5), par exemple le RFC 5856 qui permet de comprimer les paquets dans un tunnel IPsec ou le RFC 5858 qui décrit des extensions à IPsec lui permettant de mieux s'entendre avec ROHC.

Si certains protocoles, comme dans la section précédente, utilisent IPsec, d'autres réutilisent seulement IKE pour leurs besoins propres (section 9). C'est le cas par exemple d'EAP (RFC 5106).

Voilà, ce résumé d'un long RFC (et qui se termine par une longue bibliographie) aura réussi, je l'espère, à donner une idée de la richesse et de la complexité du monde d'IPsec.

Merci à Yves Rutschle pour avoir relu et trouvé une grosse bogue.