RFC 7401 : Host Identity Protocol Version 2 (HIPv2)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 10 avril 2015

Date de publication du RFC : Avril 2015

https://www.bortzmeyer.org/7401.html

HIP, décrit dans ce RFC, est un protocole très ambitieux, puisqu'il vise à compléter IP en fournissant une séparation de l'identificateur et du localisateur <https://www.bortzmeyer.org/separation-identificateur.html>, permettant d'améliorer la sécurité (notamment la résistance aux DoS) et de mieux gérer la mobilité et le "*multi-homing*". Ce RFC décrit la version 2 de HIP, désormais norme officielle (au lieu de protocole expérimental, ce qu'était HIP v1).

L'architecture générale de HIP est décrite dans le RFC 9063 [1] (vous pouvez lire mon article de résumé de HIP <https://www.bortzmeyer.org/hip-resume.html>). Notre RFC normalise, lui, le protocole concret. HIP repose d'abord sur la séparation entre un nouvel **identificateur**, le HI ("*Host Identity*") et un **localisateur**, plus concret, qui sera simplement l'adresse IP, réduite à un rôle moins visible, sans exigence de stabilité. Par exemple, HIP permettra le changement de localisateur (d'adresse IP) en cours de connexion, sans rompre celle-ci, ce qui sera précieux pour la mobilité.

HIP est donc déployable uniquement en modifiant les machines terminales du réseau (si les coupe-feux le laissent passer), sans toucher aux routeurs. Il en existe des mises en œuvres pour FreeBSD et Linux <http://www.openhip.org/>. Le projet OpenHIP <http://www.openhip.org/> adapte également des logiciels comme Wireshark pour qu'ils aient un support HIP. InfraHIP <http://infrahip.hiit.fi/> travaille également à l'infrastructure HIP.

Si on ne veut pas lire le RFC 9063, on peut néanmoins avoir une bonne introduction à HIP en lisant les sections 1 et 2 de notre RFC (ou, si on est très pressé, mon article <https://www.bortzmeyer.org/hip-resume.html>). Elles expliquent le vocabulaire (par exemple, HIP, étant un protocole situé en haut de la couche 3 n'utilise pas le terme de connexion mais celui d'**association**), le nouvel espace de nommage et les principes du protocole.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc9063.txt

L'espace de nommage fait l'objet de la section 3. On distingue les HI ("*Host Identity*"), qui sont des **clés publiques** d'un couple clé privée / clé publique et qui sont de taille variable, et les HIT ("*Host Identity Tag*", décrits dans la section 3.1), qui sont un résumé cryptographique des HI. Ils sont de taille fixe (donc plus faciles à traiter), 128 bits, la taille d'une adresse IPv6. Un préfixe ORCHID (RFC 7343), le `2001:20::/28`, sert à éviter toute collision avec les « vraies » adresses IPv6. Avec OpenHIP, la clé peut être générée par le programme `hitgen` qui fabrique un fichier XML ressemblant à ceci :

```
<?xml version="1.0" encoding="UTF-8"?>
<my_host_identities>
  <host_identity alg="RSA" alg_id="5" length="128" anon="no" incoming="yes" r1count="10">
    <name>horcrux-1024</name>
    <N>C6EBA2894C33A1312B38853A8ECC0D7967496237A65529807EDF23C4DA753EE88F8FBF71BE38B6910181D5B75DB075B996232
    <E>010001</E>
    <D>383A51165838DBDE872611DACC94775692D09677BE87A214954843D7181D3E2C04B0905FF9721481069909AD2C497DED78B7F
    <P>FF03F93454C9C2D8EC47FE8C9DBF0D82F05E13905F304A5ACA42C45E579F917B4C8CEFEF6B06AAB9BCB7A911D5514B7AEE685
    <Q>C7B0394EB5506B2B75E19E5654262E843659BB76A465C2A7AC47A430749944378E3161FF805B4C6CB037B5CB111F0EF49FF03
    <dmp1>7426C128DEBD8EEBF2A2D004080D6F0006AF32C5FD352788B6BB3669AA0B59DE08FDE082F202755C67E25735722DB6ED65
    <dmq1>1B97DE5361FA9AD4869586ABA7351F78658A40BD443A4B8B9FE2C66D6BAF421DEB2827C2869A17156DC444FAAA83002E0D
    <iqmp>7499A27F59CA1746F1A6E5DE832592F8ACF80B814DD511C490614C44DC92B5CD1650AC944ED5751F28846487C221E8C17F
    <HIT>2001:1f:cd4:7125:2427:f77c:d1b6:e15f</HIT>
    <LSI>1.182.225.95</LSI>
  </host_identity>
</my_host_identities>
```

Notez bien que le fait d'utiliser XML est un choix de OpenHIP, qui n'est utilisé qu'en local. Il n'est pas imposé par la norme qui, sur le câble, n'utilise que du binaire. Les éléments comme `P`, `Q` ou `iqmp` sont les éléments d'une clé RSA (HIP peut utiliser d'autres algorithmes que RSA, comme ECDSA). Le HIT est représenté en utilisant la syntaxe des adresses IPv6, puisqu'il a la même taille et a été conçu pour être stocké comme une adresse par les applications.

La section 3.2 explique comment générer un HIT à partir du HI. Étant un résumé cryptographique (fait avec SHA-256 ou un équivalent), il est sûr, on ne peut pas fabriquer facilement un HI qui aurait le même HIT (cf. annexe E).

Pour signer les paquets, les deux machines utiliseront au début un échange de Diffie-Hellman.

La section 4 donne une vision générale du protocole, qui sera ensuite détaillée dans les sections ultérieures. HIP a reçu le numéro de protocole `<https://www.iana.org/assignments/protocol-numbers>` 139 (il n'a pas changé avec la version 2 de HIP).

La section 4.1 décrit comment former une **association** entre deux machines HIP. Celle qui demande l'association est nommée l'**initiateur**, celle qui l'accepte le **répondeur**. Le protocole d'association nécessite quatre paquets. En effet, avec seulement trois paquets, comme le fait TCP (RFC 793) lors de l'établissement d'une connexion ("*three-way handshake*"), on ne peut pas à la fois se protéger contre les DoS et permettre des options par connexion. Se protéger contre les DoS nécessite de ne **pas** garder d'état tant que le pair n'est pas authentifié, même faiblement. Les techniques qui permettent à TCP de ne pas garder d'état sur le « répondeur », telles que les "*SYN cookies*" du RFC 4987 sont incompatibles avec les options TCP.

Voici pourquoi les protocoles plus récents comme SCTP (RFC 3286) ou comme HIP nécessitent quatre paquets.

Dans HIP, ils sont nommés **I1**, **R1**, **I2** et **R2**. Les paquets I sont envoyés par l'initiateur et les paquets R par le répondeur.

L'établissement d'une association se passe donc comme ceci :

— L'initiateur envoie I1 au répondeur. Le paquet contient l'identificateur (le HIT) de l'initiateur et (optionnellement) celui du répondeur. Il contient également les paramètres de la session Diffie-Hellman (attention, c'est un changement par rapport à HIP v1). Aucun état n'est créé chez le répondeur.
— Le répondeur envoie R1. Ce paquet contient un « puzzle » cryptographique que l'initiateur devra résoudre. Et ce paquet est signé. (Si l'initiateur ne connaissait pas le HI du répondeur, ce qu'on nomme le « mode opportuniste », il ne peut évidemment pas vérifier cette signature, et le mode opportuniste est donc vulnérable aux attaques de l'homme du milieu lors de l'établissement de la connexion, mais pas après. Même ce mode fournit donc au moins autant de sécurité que l'IP actuel.)
— L'initiateur envoie I2, qui contient la solution du puzzle et les paramètres Diffie-Hellman pour le répondeur. Ce paquet est signé. Tant que cet I2 n'a pas été reçu, le répondeur ne garde aucun état chez lui, ce qui le protège de la plupart des attaques par déni de service.
— Le répondeur envoie R2 pour accepter l'association. Ce paquet est signé.

Le puzzle, détaillé en section 4.1.1, est un petit problème de calcul que l'initiateur doit résoudre pour montrer qu'il est prêt à « payer », à faire un effort pour que l'association soit acceptée. La difficulté du puzzle peut être réglée par le répondeur, par exemple en étant plus difficile lorsqu'une attaque dDoS est en cours. D'une manière analogue au « minage » Bitcoin, le puzzle consiste simplement à trouver un nombre J tel que, concaténé avec le nombre I envoyé par le répondeur (et imprévisible, pour empêcher les pré-calculs par un attaquant), et avec les HIT des deux pairs, le condensat comprenne un certain nombre K de zéros initiaux (voir l'annexe A pour les détails). Ce K est donc la difficulté du puzzle. À noter qu'il n'y a pas d'estampille temporelle dans les paramètres de la demande de connexion, et que les attaques par rejeu sont donc possibles (mais cela dispense d'une synchronisation d'horloges globale, cf. section 4.1.4).

En pratique, il est très difficile d'imaginer un puzzle qui soit dissuasif contre des attaquants disposant d'un "*botnet*" entier, tout en étant soluble par des appareils simples, genre téléphone portable, ne possédant guère de puissance de calcul. (La section 4.1.1 détaille ce problème et les solutions possibles.)

La section 4.1.3 détaille l'utilisation du Diffie-Hellman.

Une des nouveautés de HIP v2 est l'agilité cryptographique, c'est-à-dire le fait que les algorithmes cryptographiques utilisés sont des paramètres du protocole, pas des décisions définitives. On peut donc ainsi changer d'algorithme suivant les progrès de la cryptanalyse. Mais l'agilité a aussi des inconvénients comme le risque d'attaques par repli où, lors de la négociation entre les deux pairs, un homme du milieu réussit à faire croire à Alice que Bob ne gère pas tel algorithme, forçant ainsi Alice à se replier sur un algorithme plus faible. Une nouveauté de ce RFC 7401 est donc la section 4.1.7 sur les protections contre le repli. Contrairement à d'autres protocoles cryptographiques, HIP signe presque tous les paquets relatifs à la négociation d'algorithmes. Ils ne peuvent donc pas être modifiés. Une attaque par rejeu reste possible : Bob met à jour son logiciel, accepte désormais de meilleurs algorithmes mais Mallory a enregistré ses anciens paquets et les envoie à Alice, qui croira alors (même après la vérification de la signature) que Bob continue à ne gérer que de vieux algorithmes. HIP dispose donc d'autres protections comme le fait que les paramètres Diffie-Hellman sont aussi dans ces paquets de négociation et qu'un vieux paquet mènera donc à un échec de la négociation Diffie-Hellman.

Une des grandes forces de HIP est la possibilité de **mettre à jour** une association existante (section 4.2). À tout moment, pendant la session, un paquet de type UPDATE peut être envoyé pour changer certains paramètres de la session, comme les localisateurs (les adresses IP) utilisées. La signature des paquets permet de s'assurer que le paquet UPDATE est authentique.

Une fois l'association établie, les machines peuvent échanger des données. Si elles utilisent des protocoles comme ESP (RFC 7402), ces données sont protégées contre l'écoute et la modification (HIP lui-même ne chiffre pas).

La section 5 est longue car elle détaille le format, bit par bit, de tous les paquets échangés. Il y a huit types de paquets (section 5.3) comme I1, R1, I2, R2 ou comme UPDATE, présenté plus haut. Dans l'en-tête fixe, commun à tous les paquets, se trouve une liste de paramètres HIP, encodés en TLV, qui permettent de transporter des informations comme les caractéristiques du puzzle (dans les paquets R1), la liste des algorithmes cryptographiques, etc). La liste de tous les paramètres se trouve dans un registre IANA <https://www.iana.org/assignments/hip-parameters/hip-parameters.xml>.

Enfin, la section 6 détaille le traitement des paquets, ce qu'il faut faire en les recevant, les erreurs et la façon de les gérer (règle simple : ne pas renvoyer de paquets HIP en cas d'anomalie, seulement des ICMP, et avec un débit limité, car on ne peut pas être sûr du pair s'il y a une erreur), etc.

Notez aussi la section 7 de notre RFC, qui est consacrée à la politique de gestion des identificateurs. Elle recommande fortement que toute mise en œuvre de HIP permette de gérer au moins deux HI différents par machine. Un sera publié et servira à être contacté et au moins un autre sera « anonyme » (terme inexact mais c'est celui du RFC), non publié, utilisé uniquement pour initier des connexions et sera donc plus difficilement traçable. Un inconvénient de l'authentification forte des machines est en effet que cela risque de faire perdre de la vie privée. D'où cette idée d'identificateurs non reliables à une autre identité (si on est très prudent, on utilise un HI différent pour chaque répondeur auquel on se connecte).

La section 8 de notre RFC se penche sur les problèmes de sécurité, une plaie récurrente sur l'Internet d'aujourd'hui, et qui frappera certainement également HIP, s'il est massivement déployé. D'abord, les attaques par déni de service. Celles-ci prennent en général leur source dans une asymétrie, par exemple dans le fait que les coûts soient plus élevés pour l'une des parties. Voici pourquoi il faut quatre paquets pour établir une association : le premier envoyé par le répondeur, R1, peut être un paquet général, envoyé pour toutes les demandes d'association. Le répondeur ne stocke aucun état tant que l'initiateur n'a pas pu prouver son identité.

Comme le R1 est bien plus gros que le I1, HIP pourrait être utilisé pour une attaque par amplification (une attaque où la réponse est plus grosse que la question), en usurpant l'adresse IP source de sa victime. Comme avec le DNS et son RRL <http://www.redbarn.org/dns/ratelimits>, un bon répondeur devrait limiter le rythme d'envoi des R1 par adresse IP.

Enfin, le plus dur, quand on compte sur la cryptographie pour se protéger, c'est de combattre les Hommes du Milieu. Rien ne sert de chiffrer avec les meilleurs algorithmes de cryptographie si, en croyant parler à son correspondant, on parle en fait à un homme du milieu qui s'est fait passer pour le correspondant. HIP a donc ce problème. Certes, les paquets sont signés mais comment être sûr que le HI du correspondant, qui sert à vérifier ces signatures, est le bon ? La seule solution fiable est de valider le HI du correspondant via un tiers. Par exemple, on a trouvé ce HI dans le DNS, dans une zone sécurisée par DNSSEC. Ou bien le HI était dans un certificat X.509 qu'on a validé.

On l'a vu, ce RFC 7401 normalise la version 2 de HIP. Quels sont les principaux changements par rapport à la version 1, normalisée dans le RFC 5201 ? La section 11 répond à cette question. D'abord, on utilise pour représenter les HIT la version 2 d'ORCHID, normalisée dans le RFC 7343 et non plus la v1 du RFC 4843, qui ne permettait l'agilité cryptographique. Résultat, les HIT commenceront désormais par `2001:20...` et non plus `2001:10...`. Ensuite, les HI peuvent désormais utiliser les courbes elliptiques, via ECDSA. Le protocole d'établissement de l'association a sérieusement changé, avec les paramètres Diffie-Hellman mis plus tôt. Et il y a également d'innombrables changements moins importants, mais qui font que le protocole est différent et incompatible avec la v1. La base installée de HIP étant très faible pour l'instant, cela n'est pas trop gênant. Ces modifications viennent de l'expérience avec HIP v1, obtenue par de nombreux essais avec les différentes mises en œuvre du protocole. Cette

expérience a permis de faire passer HIP du statut « Expérimental », celui du RFC 5201, à celui de « Chemin des normes ».

Il existe plusieurs mises en œuvre de HIP v1 (cf. RFC 6538) et HIP for Linux <https://launchpad.net/hipl> ainsi que OpenHIP <http://openhip.sourceforge.net/> ont annoncé leur intention de les adapter à HIP v2.