RFC 5533 : Shim6: Level 3 Multihoming Shim Protocol for IPv6

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 18 juin 2009

Date de publication du RFC : Juin 2009

https://www.bortzmeyer.org/5533.html

Quelles sont les méthodes disponibles dans l'Internet d'aujourd'hui pour assurer une plus grande fiabilité à un site ? On peut payer plus cher son FAI mais cela ne garantit pas une meilleure fiabilité. Aujourd'hui, la seule méthode réaliste est d'avoir plusieurs FAI, le "*multihoming*". Mais cela implique d'avoir ses adresses IP à soi et de faire du BGP avec ses FAI. Si on utilise en effet des adresses « appartenant » à un des FAI, la panne de celui-ci empêchera de les utiliser, même si l'autre connexion est intacte. Comme les protocoles de transport comme TCP sont liés aux adresses IP utilisées, cela veut dire que toutes les sessions TCP existantes seront cassées. Ce n'est pas forcément génant pour HTTP, aux connexions souvent courtes, mais bien plus grave pour, par exemple, SSH. Le "*multihoming*" sans BGP permet donc d'établir de nouvelles connexions, mais pas de faire vivre les anciennes.

Il faut donc faire du BGP. Mais c'est complexe et, surtout, cela impose de mettre ses adresses IP dans la table de routage globale, qui est déjà bien chargée <http://thyme.apnic.net/>. Shim6, objet de ce RFC, propose une autre solution, lier les connexions à un groupe d'adresses IP, pas à une seule comme actuellement. On pourrait alors changer l'adresse utilisée en couche 3 sans dommage pour la couche 4. Shim6 dispense donc d'utiliser BGP.

Ce nouveau protocole dépend d'IPv6, comme son nom l'indique, car sa sécurité et même son fonctionnement de base nécessitent des mécanismes propres à IPv6. Shim6 se situe conceptuellement entre la couche 3 et la couche 4, d'où son nom de "*shim*" (mince couche entre deux vraies couches). On peut noter que SCTP (RFC 3286 [1]) part d'une idée proche (le groupe d'adresses IP) mais fonctionne, lui, dans la couche 4. Shim6, au contraire, est accessible à tous les protocoles de transport comme, par exemple, le TCP traditionnel.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3286.txt

Une autre façon de s'attaquer au même problème est de séparer complètement l'identificateur du localisateur <https://www.bortzmeyer.org/separation-identificateur-localisateur.html> comme le fait HIP, ce qui permet de traiter des problèmes supplémentaires comme la mobilité. Cette séparation est explicitement marquée comme « non-but » dans Shim6, qui ne manipule que des « localisateurs » même si l'un d'eux, baptisé « identificateur » sans l'être vraiment, a un rôle particulier dans l'association (section 1.3, qui définit les ULID - "*Upper-Layer Identifier*").

Il existe donc désormais un grand choix de protocoles, tous prometteurs mais tous aussi marginaux les uns que les autres.

La gestation de ce protocole a été longue et difficile. Au final, Shim6 est un protocole très complexe, avec un RFC de plus de 130 pages (sans la détection de panne, qui fait l'objet d'un document séparé, le RFC 5534).

La section 1 expose les principes de base de Shim6. Le protocole est mis en œuvre complètement dans les machines finales, il ne nécessite aucune participation des routeurs. Chaque machine a un jeu d'adresses IP possibles, a priori toutes liées à un FAI (Shim6, contrairement à BGP, n'impose pas d'utiliser des adresses PI - "*Provider Independant*"). À un moment donné, une paire d'adresses (source et destination) est utilisée pour la communication effective et les machines peuvent changer de paire par la suite (le mécanisme de détection des pannes, ou des autres raisons qui qui déclencheraient un changement de paire, sera spécifié dans un autre document). Pour assurer qu'une machine ne puisse pas prétendre être utilisatrice d'une adresse IP quelconque, des adresses HBA ("*Hash Based Addresses*") ou CGA (signées par la cryptographie) sont utilisées.

La section 1.1 détaille le cahier des charges et la 1.2 l'anti-cahier des charges, les questions qui étaient hors-sujet pour Shim6. Dans le cahier des charges, cette section 1 insiste notamment sur le côté « opportuniste » de Shim6 : contrairement à HIP, une association Shim6 peut commencer sans échange particulier et être établie bien après la connexion TCP. D'autre part, outre la résistance aux pannes, Shim6 doit permettre la répartition de charge, selon des paires d'adresses IP différentes. Dans l'anti-cahier des charges, se trouve la question de la rénumérotation, suite à des changements de FAI. Shim6 ne la traite pas (section 1.5) : l'ensemble des adresses IP disponibles doit être fixe, ou, en tout cas, changer lentement. Shim6 ne permet donc pas le nomadisme, même si celui-ci pourra être traité dans le futur. En attendant, au début de l'association Shim6, toutes les adresses IP possibles doivent être connues.

Shim6 est un protocole de couche « 3,5 », entre la couche de réseau et la couche de transport (section 1.6). Les protocoles situés au dessus de Shim6, comme TCP ou UDP utilisent des adresses IP habituelles, les ULID ("*Upper-Layer Identifier*", voir la section 2.1 pour une terminologie complète). En dessous d'eux, Shim6 va fabriquer des paquets IP dont les adresses sources ou destination pourront être des ULID (puisque Shim6 ne sépare pas localisateurs et identificateurs) ou bien des purs localisateurs, différents des ULID que continue à utiliser le protocole de transport. Shim6 maintient, pour chaque paire d'ULID, la liste des localisateurs possibles (les adresses IP utilisables pour la communication).

Après une section 2 de terminologie et de notations, puis une section 3 sur les pré-supposés de Shim6, voici venir le protocole lui-même en section 4. L'essentiel du fonctionnement de Shim6 peut être résumé dans les étapes suivantes :

— Une machine A en contacte une autre, B, par les moyens classiques, sans Shim6. Par exemple, elle établit une connexion TCP.
— Une des deux machines, ou les deux, décide que ce serait une bonne idée de faire du Shim6. Par exemple, plus de N paquets ont été échangés et il semble donc bien que la connexion va durer, ce qui rend intéressant de la faire résister aux pannes. Un **contexte** Shim6 (une association entre les deux machines) est alors mis en place.

— Les échanges continuent comme avant. À ce stade, bien que le contexte Shim6 aie été configuré, rien n'apparait dans les paquets échangés.
— Une panne survient, rendant impossible d'utiliser les localisateurs actuels (les adresses IP de A et B). C'est là que Shim6 sert à quelque chose, une nouvelle paire de localisateurs est choisie et un en-tête d'extension est ajouté désormais à chaque paquet, pour indiquer le contexte utilisé (section 4.1). Ce contexte servira à trouver les ULID (les identificateurs) utilisés. Shim6 réécrira alors les paquets avant de les transmettre à TCP... qui ne se sera rendu compte de rien, il utilisera les mêmes adresses IP tout le temps (section 12 pour les détails sur la réception des paquets).

Donc, on ne peut pas, en observant le réseau, savoir si un contexte Shim6 est disponible car les paquets ne sont modifiés (par l'ajout de l'en-tête d'extension Shim6) que si la première paire de localisateurs devient inutilisable.

Ces en-têtes d'extension sont une nouveauté d'IPv6 (RFC 2460, section 4). Leur présence dans les paquets IPv6 est rare aujourd'hui mais des techniques comme Shim6 pourrait la rendre plus fréquente (si les différents pare-feux acceptent de les laisser passer). La section 4.6 décrit le placement de l'en-tête Shim6.

Les informations mémorisées par chaque machine (ULID de la connexion, localisateurs possibles, contextes, etc) sont décrites dans la section 6.

Comme toute technique d'indirection, c'est-à-dire comme toutes les techniques qui découplent, si peu que ce soit, l'identificateur et le localisateur, Shim6 est vulnérable aux attaques portant sur la correspondance entre utilisateur et localisateur. Qu'est-ce qui empêche un méchant, par exemple, d'ajouter une adresse IP qu'il ne contrôle pas légitimement, au jeu des localisateurs, pour que son partenaire Shim6 y envoie des paquets ? La section 4.4 résume les techniques utilisées pour empêcher cela, qui tournent autour d'adresses cryptographiques (RFC 3972 et RFC 5535) et de tests de la connectivité effective (voir si le pair répond avant de le noyer sous les paquets, cf. section 5.13 pour le message `Probe`).

Comment se fait l'établissement du contexte Shim6 ? Par un échange de quatre paquets (comme dans HIP) et non pas de trois paquets comme avec TCP. La section 4.5 résume ces quatre paquets, I1, R1, I2 et R2, pour lesquels l'explication complète est en section 7 (et la machine à états dans la section 20). La liste des localisateurs connus est envoyée dans ces paquets mais elle peut être modifiée par la suite, Shim6 ayant d'autres messages de contrôle que ces quatre-ci (section 10).

Le format des messages sur le câble est normalisé dans la section 5. Les messages de contrôle de Shim6 ne circulent ni sur UDP, ni sur TCP mais dans son propre protocole, de numéro 140 (section 17). Le début du paquet est le même pour tous les messages Shim6 (section 5.1). Les messages de la communication en cours ont le 16ème bit à 1 et sont décrits dans la section 5.2. Les messages de contrôle ont ce 16ème bit à zéro et font l'objet de la section 5.3.

Comment se fait le premier contact ? La procédure est synthétisée dans la section 13. La méthode recommandée est d'essayer toutes les adresses disponibles, de ne pas renoncer simplement parce qu'une paire d'adresses émetteur-récepteur ne fonctionne pas. Le problème de cette méthode est que, mise en œuvre directement (boucler sur toutes les adresses de la liste de `struct addrinfo` renvoyée par `getaddrinfo()` et tenter un `connect()` sur chacune) va mener à de longs délais puisqu'il faudra attendre l'expiration du délai de garde. Idéalement, il faudrait donc tenter des connexions non-bloquantes simultanément.

Comment est établi un contexte Shim6 ? La section 7 fournit toutes les informations nécessaires. L'échange nécessite quatre paquets (section 7.3), ce qui permet (contrairement à l'échange à trois paquets de TCP) de ne pas garder d'état dès le premier paquet, tout en permettant l'usage d'options. Cette

section couvre également la vérification des adresses utilisées (section 7.2), vérification, par la cryptographie, que le « propriétaire » d'un ULID a le « droit » d'utiliser ces localisateurs et vérification que ces localisateurs fonctionnent.

Un problème classique de toute séparation, même partielle, entre l'identificateur et le localisateur, est le traitement des erreurs envoyées par ICMP. L'émetteur de ces erreurs ne connait pas forcément Shim6 et gère simplement des adresses IP. Il faut donc, sur réception d'un message ICMP, retrouver le contexte Shim6 (section 8) **et** transmettre le message ICMP à la couche de transport puisque, dans la plupart des cas, c'est elle qui doit être informée des erreurs. Cela nécessite de réécrire le message pour remplacer les adresses IP par des ULID, les seules adresses que connaisse la couche 4.

Arrivé à ce point du RFC, il est temps de voir quelques contraintes d'implémentation. Elles sont regroupées dans la section 15. D'abord, si Shim6 change la paire de localisateurs utilisée, les paquets IP passeront probablement par un chemin bien différent, et Shim6 devrait donc (section 15.1) prévenir la couche de transport que la congestion sera probablement différente.

Tout nouveau protocole qu'on tente de déployer sur l'Internet doit se battre contre les "*middleboxes*", les équipements intermédiaires qui refusent fréquemment les paquets inconnus. La section 15.2 soulève le problème. Par exemple, quoique que Shim6 soit une technique entièrement « machine », qui est gérée uniquement aux extrémités du réseau (section 15.3), il est toujours possible qu'un pare-feu bloque les paquets Shim6. Cela apparaitra à chacune des machines comme une absence de Shim6 chez le pair. Outre cette paranoïa générale des pare-feux, Shim6 soulève des questions particulières puisque l'autorisation ou l'interdiction des paquets devraient, idéalement, prendre en compte les ULID, pas les adresses IP du paquet. Shim6 aura donc les mêmes difficultés que HIP ou SCTP à s'installer dans un Internet très ossifié.

Enfin, certaines applications passent des adresses IP à leurs pairs (par exemple les téléphones SIP) et elles devraient donc s'assurer (section 15.4) qu'elles n'envoient pas uniquement l'ULID mais un nom de domaine ou bien le jeu complet de localisateurs.

Et la sécurité ? La section 16 résume les problèmes spécifiques à Shim6 (cf. RFC 4218) et les solutions possibles. Contre le risque d'usurpation d'adresses IP, les adresses cryptographiques, HBA (RFC 5535) et CGA (RFC 3972). Contre les attaques par inondation (en redirigeant du trafic vers une machine innocente), les tests d'atteignabilité (vérifier que la machine répond positivement avant de lui transmettre des gigaoctets).

En parlant d'implémentations, quelle est la situation de Shim6 de ce côté ? Une équipe coréenne avait mis en œuvre Shim6 pour le noyau Linux. Leur travail avait été décrit dans l'"*Internet-Draft*" `draft-park-shim6-implementation` mais le travail semble ne pas être allé plus loin. Plus récemment, une autre équipe, belge, avait réalisé `LinShim6`, sur le même système et documenté dans l'"*Internet-Draft*" `draft-barre-shim6-impl`. Il existe aussi une autre implémentation Linux dans l'OpenHIP `<http://sourceforge.net/project/showfiles.php?group_id=132288&package_id=266257>`. Une liste à jour des tentatives se trouve sur le site « officiel » `<http://www.shim6.org/>`.

Terminons ce résumé du RFC 5533 avec les annexes. La section 19 décrit de possibles extensions futures du protocole. Celui-ci est déjà bien assez compliqué comme cela et beaucoup d'idées ont été laissées à de futures nouvelles versions, comme la pré-vérification d'une paire de localisateurs de secours, l'interaction avec Mobile IP ou encore la normalisation d'une API. À l'heure actuelle, il n'existe pas d'API standard pour l'application qui voudrait être consciente de Shim6 et, par exemple, forcer ou, au contraire, interdire son utilisation (section 4.3).

L'excellente section 22 intéressera les étudiants en réseaux informatiques et tous les curieux qui se demandent « mais pourquoi les choses sont-elles comme elles sont ? ». Elle est en effet consacrée aux choix alternatifs, à ce qui aurait pu être. Parmi les idées envisagées et finalement abandonnées, on trouve d'autres solutions de sécurité (section 22.4) comme l'utilisation de "*cookies*" lors de l'établissement de l'association. La connaissance du "*cookie*" prouvait qu'on était toujours le même pair. Mais le "*cookie*" aurait été vulnérable lors d'une écoute du réseau. D'où le choix final des adresses cryptographiques. Dans le cas de HBA (RFC 5535), tous les localisateurs (toutes les adresses IP) sont liées et on peut donc vérifier que deux adresses appartiennent au même ensemble HBA. Par contre, HBA ne permet pas d'ajouter ou de supprimer des localisateurs. Dans le cas de CGA (RFC 3972), l'ULID est une clé publique (plus rigoureusement, un résumé d'une clé publique) et signe les localisateurs, ce qui autorisde l'ajout de localisateurs. Mais CGA dépend donc de la cryptographie à clé publique, plus coûteuse.

La création d'un contexte Shim6 (ce que j'ai baptisé association) se fait avec quatre paquets. On aurait pu en utiliser moins (section 2.5), pour accélerer l'association mais le choix a finalement été de privilégier la protection contre les attaques DoS. Une des raisons du choix est que, de toute façon, l'établissement de l'association ne bloque pas la connexion qui utilise Shim6, puisque les premiers paquets peuvent passer avant que Shim6 ne crée son contexte. Ce n'est donc pas trop grave si l'association prend du temps.