Groupe de travail Réseau
Request for Comments: 5534
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

J. Arkko, Ericsson
I. van Beijnum, IMDEA Networks
juin 2009

Protocole d'exploration de paire de localisateurs et de détection de défaillance pour multi-rattachements IPv6

Résumé
Le présent document spécifie comment le protocole Shim6 de multi-rattachements de niveau 3 (Shim6) détecte les défaillances entre deux nœuds communicants. Il spécifie aussi un protocole d'exploration pour passer à une autre paire d'interfaces et/ou d'adresses entre les mêmes nœuds si une défaillance se produit et si une paire opérationnelle peut être trouvée.

Table des matières

1. Introduction....2
2. Langage des exigences....2
3. Définitions....3
  3.1 Adresses disponibles....3
  3.2 Adresses opérationnelles localement....3
  3.3 Paires d'adresses opérationnelles....3
  3.4 Paire d'adresses principale....4
  3.5 Paire d'adresses courante....4
4. Vue d'ensemble du protocole....5
  4.1 Détection de défaillance....5
  4.2 Exploration de pleine accessibilité....6
  4.3 Ordre d'exploration....6
5. Définition du protocole....7
  5.1 Message Keepalive....7
  5.2 Message Probe....8
  5.3 Format de l'option Keepalive Timeout....10
6. Comportement....11
  6.1 Paquet de charge utile entrant....11
  6.2 Paquet de charge utile sortant....12
  6.3 Temporisation de maintien en vie....12

6.4 Fin de temporisation d'envoi....12
6.5 Retransmission....12
6.6 Réception du message Keepalive....13
6.7 Réception de l'état de message de sonde Exploring....13
6.8 Réception de l'état de message de sonde InboundOk....13
6.9 Réception de l'état de message de sonde Operational....13
6.10 Représentation graphique de l'automate à états....14
7. Constantes et variables du protocole....14
8. Considérations sur la sécurité....14
9. Considérations de fonctionnement....15
10. Références....16
10.1 Références normatives....16
10.2 Références pour information....16
Appendice A. Exemple de tours de protocole....17
Appendice B. Contributeurs....20
Appendice C. Remerciements....21
Adresse des auteurs....21

1. Introduction

Le protocole Shim6 [RFC5533] étend IPv6 pour prendre en charge le multi rattachements. C'est un mécanisme de couche IP qui cache le multi rattachements aux applications. Une partie de la solution Shim6 implique de détecter quand une paire d'adresses (ou interfaces) couramment utilisée entre deux nœuds de communication a défailli et de prendre une autre paire quand cela se produit. On appelle la première "détection de défaillance", et la dernière, "exploration de paire de localisateurs".

Le présent document spécifie les mécanismes et les messages de protocole pour réaliser à la fois la détection de défaillance et l'exploration de paire de localisateurs. Cette partie du protocole Shim6 est appelée le protocole d'accessibilité (REAP, *REAchability Protocol*).

La détection de défaillance est rendue aussi légère que possible. Le trafic de données de charge utile dans les deux directions est observé, et dans le cas où il n'y a pas de trafic parce que la communication est au repos, la détection de défaillance est aussi en repos et ne génère pas de paquets. Quand le trafic de charge utile s'écoule dans les deux directions, il n'y a pas besoin d'envoyer de paquets de détection de défaillance. C'est seulement quand il y a du trafic dans une direction que le mécanisme de détection de défaillance génère des maintiens en vie dans l'autre direction. Par suite, chaque fois qu'il y a du trafic sortant et pas de trafic de retour entrant ou de maintien en vie, il doit y avoir une défaillance, point auquel l'exploration de paire de localisateurs est effectuée pour trouver une paire d'adresses qui fonctionne pour chaque direction.

Le présent document est structuré comme suit : la Section 3 définit un ensemble de termes utiles, la Section 4 donne une vue d'ensemble de REAP, et la Section 5 donne des définitions détaillées. La Section 6 spécifie le comportement, et la Section 7 discute des constantes du protocole. La Section 8 discute les considérations sur la sécurité de REAP.

Dans cette spécification, on considère qu'une adresse est synonyme de localisateur. Les autres parties du protocole Shim6 s'assurent que les différents localisateurs utilisés par un nœud lui appartiennent réellement tous. C'est-à-dire, REAP n'est pas responsable de s'assurer que ledit nœud finit avec un localisateur légitime.

REAP a été conçu pour être utilisé avec Shim6 et est donc destiné à un environnement où il fonctionne normalement sur des hôtes, utilise des types de chemins largement variables, et est ignorant du contexte d'application. Par suite, REAP tente d'être aussi auto-configurant et non obstructif que possible. En particulier, il évite d'envoyer des paquets sauf lorsque c'est absolument nécessaire et emploie le retard exponentiel pour éviter l'encombrement. L'inconvénient est qu'il ne peut pas offrir la même granularité de détection des problèmes que les mécanismes qui ont plus de contexte d'application et la capacité de négocier ou configurer les paramètres.

De futures versions de cette spécification pourront considérer des extensions avec de telles capacités, par exemple, en héritant de mécanismes du protocole de détection de transmission bidirectionnelle (BFD, *Bidirectional Forwarding Detection*) [RFC5880].

2. Langage des exigences

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119],.

3. Définitions

Cette Section définit les termes utiles pour la discussion de la détection de défaillance et de l'exploration de paire de localisateurs.

3.1 Adresses disponibles

Les nœuds Shim6 doivent savoir quelles adresses ils ont. Si un nœud perd l'adresse qu'il utilise actuellement pour les communications, une autre adresse doit la remplacer. Et si un nœud perd une adresse que l'homologue du nœud connaît, l'homologue doit être informé. De même, quand un nœud acquiert une nouvelle adresse, il peut généralement souhaiter que l'homologue le sache.

Définition d'adresse disponible - une adresse est dite être disponible si toutes les conditions suivantes sont remplies :
- o L'adresse a été allouée à une interface du nœud.
- o La durée de validité du préfixe (paragraphe 4.6.2 de la [RFC4861]) associé à l'adresse n'est pas expirée.
- o L'adresse n'est pas une tentative au sens de la [RFC4862]. En d'autres termes, l'allocation d'adresse est complète et les communications peuvent débuter. Noter que cela permet explicitement qu'une adresse soit optimiste au sens de la détection optimiste d'adresse dupliquée (DAD, *Optimistic Duplicate Address Detection*) [RFC4429] même si les mises en œuvre peuvent préférer utiliser d'autres adresses pour autant qu'elles aient le choix.
- o L'adresse est en envoi individuel mondial ou unique locale [RFC4193]. C'est-à-dire, ce n'est pas une adresse IPv6 de site local ou de liaison locale. Avec des adresses de liaison locale, les nœuds seraient incapables de déterminer sur quelle liaison l'adresse en question est utilisable.
- o L'adresse et l'interface sont acceptables pour une utilisation en accord avec une politique locale.

Les adresses disponibles sont découvertes et surveillées par des mécanismes qui sortent du domaine d'application de Shim6. Les mises en œuvre de Shim6 DOIVENT être capables d'employer les informations fournies par la découverte de voisin IPv6 [RFC4861], par l'auto configuration d'adresse [RFC4862], et par DHCP [RFC3315] (quand DHCP est mis en œuvre). Ces informations incluent la disponibilité d'une nouvelle adresse et des changements d'état des adresses existantes (comme quand une adresse devient invalide).

3.2 Adresses opérationnelles localement

Deux niveaux de granularité différents sont nécessaires pour la détection de défaillance. Le plus gros grain est pour les adresses individuelles.

Définition de l'adresse opérationnelle localement : une adresse disponible est dite être localement opérationnelle quand il est connu que son utilisation est possible localement. En d'autres termes, quand l'interface est active, qu'il est connu qu'un routeur par défaut (si nécessaire) convenable pour cette adresse est accessible, et qu'aucune autre information locale ne désigne l'adresse comme étant inutilisable.

Les adresses opérationnelles localement sont découvertes et surveillées par des mécanismes qui sortent du domaine d'application du protocole Shim6. Les mises en œuvre de Shim6 DOIVENT être capables d'employer les informations fournies par la détection d'inaccessibilité de voisin [RFC4861]. Les mises en œuvre PEUVENT aussi employer des mécanismes supplémentaires spécifiques de la couche de liaison.

Note 1 : à part le problème de s'assurer qu'une adresse est opérationnelle, il faut s'assurer qu'après un changement de connexité de couche de liaison, on est toujours connecté au même sous-réseau IP. Des mécanismes comme ceux de la [RFC6059] peuvent être utilisés pour cela.

Note 2 : en théorie, il serait aussi possible aux nœuds d'apprendre les défaillances d'acheminement pour un préfixe de source choisi particulier, si seulement des protocoles convenables existaient pour cela. Certaines propositions ont été faites dans ce domaine (voir, par exemple [ADD-SEL] et [MULTI6]), mais aucune n'a été normalisée à ce jour.

3.3 Paires d'adresses opérationnelles

L'existence d'adresses opérationnelles en local n'est cependant pas une garantie que les communications peuvent être établies avec l'homologue. Une défaillance dans l'infrastructure d'acheminement peut empêcher les paquets d'atteindre leur destination. Pour cette raison, on a besoin de la définition d'un second niveau de granularité, qui est utilisé pour les paires d'adresses.

Définition. Paire d'adresses opérationnelle en bidirectionnel - une paire d'adresses opérationnelles en local est dite être une paire d'adresses opérationnelle en bidirectionnel quand une connexité bidirectionnelle peut être montrée entre les adresses. C'est-à-dire, un paquet envoyé avec une des adresses dans le champ Source et l'autre dans le champ Destination atteint la destination, et vice versa.

Malheureusement, il y a des scénarios où des paires d'adresses bidirectionnellement opérationnelles n'existent pas. Par exemple, le filtrage d'entrée ou des défaillances du réseau, peuvent résulter en une paire d'adresses qui est opérationnelle dans une direction tandis qu'une autre est opérationnelle dans l'autre direction. La définition qui suit retrace cette situation générale.

Définition. Paire d'adresses opérationnelle unidirectionnellement - une paire d'adresses opérationnelles en local est dite être une paire d'adresses unidirectionnellement opérationnelle quand les paquets envoyés avec la première adresse comme source et la seconde adresse comme destination atteignent la destination.

Les mises en œuvre de Shim6 DOIVENT prendre en charge la découverte des paires d'adresses opérationnelles par l'utilisation d'essais d'accessibilité explicites et de communication bidirectionnelle forcée (FBD, *Forced Bidirectional Communication*) décrits plus loin dans cette spécification. De futures extensions de Shim6 pourront spécifier des mécanismes supplémentaires. Des idées de tels mécanismes sont mentionnées ci-dessous mais ne sont pas pleinement spécifiées dans ce document :

- Un retour positif de la part des protocoles de couche supérieure. Par exemple, TCP peut indiquer à la couche IP qu'il est en cours. Ceci est similaire à la façon dont la détection d'inaccessibilité de voisin IPv6 peut, dans certains cas, être évitée quand des couches supérieures fournissent des information sur la connexité bidirectionnelle [RFC4861].
  Dans le cas de connexité unidirectionnelle, les réponses du protocole de couche supérieure reviennent en utilisant une autre paire d'adresses, mais montrent que les messages envoyés en utilisant la première paire d'adresses ont été reçus.

- Retour négatif de la part des protocoles de couche supérieure. Il est concevable que les protocoles de couche supérieure donnent une indication d'un problème à la couche de multi rattachements. Par exemple, TCP pourrait indiquer qu'il y a de l'encombrement ou un manque de connexité dans le chemin parce que il ne donne pas d'accusés de réception.

- Messages d'erreur ICMP. Étant donnée la facilité d'usurper les messages ICMP, on devrait être prudent en n'accordant pas aveuglément sa confiance. Une approche serait d'utiliser les messages d'erreur ICMP seulement comme indication d'effectuer un essai explicite d'accessibilité ou de déplacer une paire d'adresses à un rang inférieur dans la liste des paires d'adresses à sonder, mais de ne pas utiliser ces messages comme une raison d'interrompre les communications en cours sans autre indication de problème. La situation peut être différente quand certaines vérifications des messages ICMP sont effectuées, comme expliqué par Gont dans la [RFC5927]. Ces vérifications peuvent assurer que (pratiquement) seuls des attaquants dans le chemin peuvent usurper les messages.

3.4 Paire d'adresses principale

La paire d'adresses principale consiste en les adresses que les protocoles de couche supérieure utilisent dans leur interaction avec la couche Shim6. L'utilisation de la paire d'adresses principale signifie que la communication est compatible avec une communication non Shim6 régulière et qu'aucune étiquette de contexte n'a besoin d'être présente.

3.5 Paire d'adresses courante

Shim6 doit éviter d'envoyer des paquets qui appartiennent à la même connexion de transport concurremment sur des chemins différents. C'est parce que le contrôle d'encombrement dans les protocoles de transport couramment utilisés se fonde sur la notion d'un seul chemin. Bien que l'acheminement puisse aussi introduire des changements de chemin et que

les protocoles de transport aient les moyens de traiter cela, de fréquents changements causent des problèmes. Un contrôle d'encombrement efficace sur plusieurs chemins est considéré comme sujet de recherche au moment de la publication du présent document. Shim6 ne tente pas d'employer simultanément plusieurs chemins.

Note : Le protocole de transmission de commandes de flux (SCTP, *Stream Control Transmission Protocol*) et de futurs protocoles de transport multi chemins vont probablement exiger une interaction avec Shim6, au moins pour assurer qu'ils n'emploient pas Shim6 de façon inattendue.

Pour ces raisons, il est nécessaire de choisir une paire d'adresses particulière comme paire d'adresses courante qui va être utilisée jusqu'à ce que des problèmes surgissent, au moins pour la même session.

Il est théoriquement possible de prendre en charge plusieurs paires d'adresses courantes pour différentes sessions de transport ou contextes Shim6. Cependant, ceci n'est pas pris en charge par cette version du protocole Shim6.

Une paire d'adresses courante n'a pas besoin d'être opérationnelle tout le temps. Si il n'y a pas de trafic à envoyer, on ne peut pas savoir si la paire d'adresses courante est opérationnelle. Néanmoins, on peut supposer que la paire d'adresses qui fonctionnait précédemment va aussi continuer d'être opérationnelle pour de nouvelles communications.

4. Vue d'ensemble du protocole

Cette section discute de la conception des mécanismes de détection d'accessibilité et d'exploration de pleine accessibilité, et donne une vue d'ensemble du protocole REAP.

Explorer l'ensemble complet d'options de communication entre deux nœuds qui ont chacun deux adresses ou plus est une opération coûteuse car le nombre de combinaisons à explorer augmente très vite avec le nombre d'adresses. Par exemple, avec deux adresses des deux côtés, il y a quatre paires d'adresses possibles. Comme on ne peut pas supposer que l'accessibilité dans une direction signifie automatiquement l'accessibilité pour la paire complémentaire dans l'autre direction, le nombre total des combinaisons bidirectionnelles est huit. (Combinaisons = nA * nB * 2.)

Une observation importante dans le multi rattachement est que les défaillances sont relativement peu fréquentes, de sorte qu'une paire opérationnelle qui fonctionnait il y a quelques secondes va très probablement être encore opérationnelle. Donc, il y a du sens à avoir un protocole léger qui confirme l'accessibilité existante, et d'invoquer un mécanisme d'exploration plus lourd seulement quand une défaillance est suspectée.

4.1 Détection de défaillance

La détection de défaillance consiste en trois parties : retracer les informations locales, retracer l'état de l'homologue distant, et finalement vérifier l'accessibilité. Le retraçage des information locales consiste à utiliser comme entrée, par exemple, les informations d'accessibilité sur le routeur local. Les nœuds DEVRAIENT employer les techniques mentionnées aux paragraphes 3.1 et 3.2 pour retracer la situation locale. Il est aussi nécessaire de retracer les informations de l'adresse distante provenant de l'homologue. Par exemple, si l'adresse de l'homologue dans la paire d'adresses courante n'est plus localement opérationnelle, un mécanisme pour relayer cette information est nécessaire. Le message Demande de mise à jour dans le protocole Shim6 est utilisé à cette fin [RFC5533]. Finalement, quand les informations locales et distantes indiquent que la communication devrait être possible et qu'il y a des paquets de couche supérieure à envoyer, la vérification d'accessibilité est nécessaire pour s'assurer que les homologues ont réellement une paire d'adresses opérationnelle.

Une technique appelée détection bidirectionnelle forcée (FBD, *Forced Bidirectional Detection*) est employée pour la vérification d'accessibilité. L'accessibilité pour la paire d'adresses couramment utilisée dans un contexte Shim6 est déterminée en s'assurant que chaque fois qu'il y a du trafic de charge utile dans une direction, il y a aussi du trafic dans l'autre direction. Ce peut être aussi du trafic de données, ou ce peuvent être des accusés de réception de couche de transport ou des maintiens en vie d'accessibilité REAP si il n'y a pas d'autre trafic. De cette façon, il n'est plus possible d'avoir du trafic dans seulement une direction ; ainsi, chaque fois qu'il y a du trafic de charge utile sortant, mais qu'il n'y a pas de paquets de retour, il doit y avoir une défaillance, et le mécanisme d'exploration complet est lancé.

Voici une description plus détaillée du mécanisme d'évaluation de l'accessibilité de la paire courante :

1. Pour empêcher l'autre côté de conclure qu'il y a une défaillance d'accessibilité, il est nécessaire qu'un nœud mette en œuvre le mécanisme de détection de défaillance pour générer des maintiens en vie périodiques quand il n'y a pas d'autre trafic. FBD fonctionne en générant des maintiens en vie REAP si le nœud reçoit des paquets de son homologue mais

n'en envoie pas de lui-même. Les maintiens en vie sont envoyés à un certain intervalle afin que l'autre côté sache qu'il y a un problème d'accessibilité quand il ne reçoit aucun paquet entrant pendant la durée d'une temporisation d'envoi *(Send Timeout)*. Le nœud communique sa valeur de temporisation d'envoi à l'homologue dans l'option Temporisation de maintien en vie (paragraphe 5.3) dans les messages I2, I2bis, R2, UPDATE. L'homologue transpose alors cette valeur en sa valeur de temporisation de maintien en vie. L'intervalle après lequel les maintiens en vie sont envoyés est appelé Intervalle de maintien en vie. L'approche RECOMMANDÉE pour l'intervalle de maintien en vie est d'envoyer les maintiens en vie entre à la moitié et le tiers de l'intervalle de temporisation de maintien en vie, afin que plusieurs maintiens en vie soient générés et aient le temps d'atteindre l'homologue avant la fin de la temporisation.

2. Chaque fois que des paquets de charge utile sortants sont générés, un temporisateur est lancé pour refléter l'exigence que l'homologue devrait générer du trafic de retour de paquets de charge utile. La valeur de temporisation est réglée à la valeur de la temporisation d'envoi. Pour les besoins de cette spécification, "paquet de charge utile" se réfère à tout paquet qui fait partie d'un contexte Shim6, incluant les paquets de protocole de couche supérieure et les messages du protocole Shim6, sauf ceux définis dans la présente spécification. Pour ces derniers messages, la Section 6 spécifie ce qui arrive aux temporisateurs quand un message est transmis ou reçu.

3. Chaque fois que des paquets de charge utile entrants sont reçus, le temporisateur associé au trafic de retour provenant de l'homologue est arrêté, et un autre temporisateur est lancé pour refléter l'exigence que ce nœud génère du trafic de retour. Cette valeur de temporisateur est réglée à la valeur du temporisateur de maintien en vie. Ces deux temporisateurs sont mutuellement exclusifs. En d'autres termes, soit le nœud attend de voir du trafic provenant de l'homologue fondé sur le trafic que le nœud a envoyé antérieurement, soit le nœud attend de répondre à l'homologue sur la base du trafic que l'homologue a envoyé antérieurement (autrement, le nœud est dans l'état Repos).

4. La réception d'un message de maintien en vie REAP conduit à arrêter le temporisateur associé au trafic de retour provenant de l'homologue.

5. Keepalive Interval secondes après que le dernier paquet de charge utile a été reçu pour un contexte, si aucun autre paquet n'a été envoyé dans ce contexte depuis que le paquet de charge utile a été reçu, un message REAP Keepalive est généré pour le contexte en question et transmis à l'homologue. Un nœud peut envoyer le maintien en vie plus tôt que Keepalive Interval secondes si des considérations de mise en œuvre l'obligent, mais on devrait veiller à éviter d'envoyer des maintiens en vie à un taux excessif. Les messages REAP Keepalive DEVRAIENT continuer d'être envoyés à l'intervalle de maintien en vie jusqu'à ce que un paquet de charge utile dans le contexte Shim6 ait été reçu de l'homologue ou que le temporisateur de maintien en vie arrive à expiration. Les maintiens en vie ne sont pas envoyés du tout si un ou plusieurs paquets de charge utile ont été envoyés dans l'intervalle de maintien en vie.

6 Send Timeout secondes après la transmission d'un paquet de charge utile sans trafic de retour sur ce contexte, une exploration d'accessibilité complète est lancée.

La Section 7 donne des valeurs par défaut suggérées pour ces valeurs de temporisation. La valeur réelle DEVRAIT être aléatoire afin d'empêcher la synchronisation. L'expérience du déploiement du protocole Shim6 est nécessaire afin de déterminer quelles valeurs conviennent le mieux.

4.2 Exploration de pleine accessibilité

Comme expliqué dans les paragraphes précédents, la paire d'adresses actuellement utilisée peut devenir invalide, soit parce qu'une des adresses est devenue indisponible ou non opérationnelle, soit parce que la paire elle-même est déclarée non opérationnelle. Un processus d'exploration tente de trouver une autre paire opérationnelle afin que les communications puissent reprendre.

Ce qui rend ce processus difficile est l'exigence de prendre en charge des paires d'adresses unidirectionnellement opérationnelles. Il est insuffisant de sonder les paires d'adresses par un protocole de simple demande-réponse. Au lieu de cela, la partie qui détecte la première le problème commence un processus par lequel elle essaye chacune des différentes paires d'adresses tour à tour en envoyant un message à son homologue. Ces messages portent des informations sur l'état de connexité entre les homologues, comme si l'envoyeur a vu récemment du trafic provenant de l'homologue. Quand l'homologue reçoit un message qui indique un problème, il aide au processus en commençant sa propre exploration parallèle dans l'autre direction, là encore en envoyant des informations sur le trafic de charge utile ou les messages de signalisation récemment reçus.

Précisément, quand A décide qu'il a besoin d'explorer si il y a une paire d'adresses de remplacement pour B, il va initier un ensemble de messages Probe *(sonde)* à la suite, jusqu'à ce qu'il obtienne un message Probe de B indiquant que (a) B a reçu

un des messages de A et, évidemment, (b) que le message Probe de B revient à A. B utilise le même algorithme, mais commence le processus à partir de la réception du premier message Probe provenant de A.

Lorsque on change pour une nouvelle paire d'adresses, le chemin de réseau traversé a très probablement changé, donc le protocole de couche supérieure (ULP) DEVRAIT être informé. Cela peut être un signal pour que l'ULP s'adapte, du fait du changement de chemin, afin que par exemple, si l'ULP est TCP, il pourrait initier une procédure de démarrage lent. Cependant, il est probable que les circonstances qui ont conduit au choix d'un nouveau chemin ont déjà causé assez de pertes de paquets pour déclencher un démarrage lent.

REAP est conçu pour prendre en charge la récupération de défaillance même dans le cas où il y a seulement des paires d'adresses unidirectionnellement opérationnelles. Cependant, du fait des soucis de sécurité discutés à la Section 8, le processus d'exploration peut normalement seulement fonctionner pour une session déjà établie. Précisément, alors que REAP serait théoriquement capable d'exploration même durant l'établissement de connexion, son utilisation dans le protocole Shim6 ne permet pas cela.

4.3 Ordre d'exploration

Le processus d'exploration suppose la capacité de choisir des paires d'adresses à essayer. Un survol du processus de choix utilisé par REAP est comme suit :

- En entrée pour commencer le processus, le nœud a connaissance de ses propres adresses et les messages du protocole Shim6 lui ont dit quelles sont les adresses de l'homologue. Une liste des paires d'adresses possibles peut être construite en combinant les deux éléments d'information.

- En employant les règles standard de choix d'adresse IPv6, la liste est élaguée en retirant les combinaisons qui sont inappropriées, comme de tenter d'utiliser une adresse de liaison locale quand on contacte un homologue qui utilise une adresse d'envoi individuel mondiale.

- De même, les règles standard de choix d'adresse IPv6 fournissent un ordre de priorité de base pour les paires.

- Des préférences locales peuvent être appliquées pour des réglages supplémentaires de l'ordre de la liste. Les mécanismes pour les réglages de préférence locale ne sont pas spécifiées mais peuvent impliquer, par exemple, une configuration qui règle la préférence pour utiliser une interface plutôt qu'une autre.

- Par suite, le nœud a une liste de paires d'adresses prioritaires à essayer. Cependant, la liste peut encore être longue, car il peut y avoir une explosion combinatoire quand il y a de nombreuses adresses des deux côtés. REAP emploie cependant ces paires à la suite, et utilise une procédure de retard pour éviter une "tempête de signalisation". Cela assure que le processus d'exploration est relativement prudent ou "sûr". Le compromis est que trouver un chemin qui fonctionne peut prendre du temps si il y a beaucoup d'adresses des deux côtés.

Plus en détails, le processus est le suivant. Les nœuds consultent d'abord les règles de choix d'adresse par défaut de la [RFC3484] pour déterminer quelles combinaisons d'adresses sont permises d'un point de vue local, car cela réduit l'espace de recherche. La RFC 3484 donne aussi un ordre de priorité parmi les différentes paires d'adresses, rendant éventuellement la recherche plus rapide. (Des mécanismes supplémentaires pourront être définis à l'avenir pour arriver à un ordre initial des paires d'adresses avant de commencer l'essai [PAIR].) Les nœuds peuvent aussi utiliser des informations locales, comme des paramètres connus de qualité de service ou des types d'interface, pour déterminer quelles adresses sont préférées à d'autres, et essayer d'abord les paires contenant de telles adresses. Le protocole Shim6 porte aussi des informations de préférence dans ses messages.

Dans l'ensemble des paires d'adresses candidates possibles, les nœuds DEVRAIENT tenter les essais sur toutes jusqu'à ce qu'une paire opérationnelle soit trouvée, et réessayer le processus comme nécessaire. Cependant, tous les nœuds DOIVENT effectuer ce processus à la suite et avec retard exponentiel. Ce processus séquentiel est nécessaire afin d'éviter une "tempête de signalisation" quand une panne se produit (en particulier pour un site complet). Cependant, cela limite aussi le nombre d'adresses qui peuvent, en pratique, être utilisées pour le multi rattachements, considérant que les protocoles de transport et de couche d'application vont échouer si le pasage à une nouvelle paire d'adresses prend trop de temps.

La Section 7 suggère des valeurs par défaut pour les temporisateurs associés au processus d'exploration. La valeur de Temporisation initiale de sonde (0,5 seconde) spécifie l'intervalle entre les tentatives initiales d'envoi de sondes ; le nombre de sondes initiales (4) spécifie combien de sondes initiales peuvent être envoyées avant que la procédure de retard exponentiel doive être employée. Ce processus augmente le temps entre chaque sonde si il n'y a pas de réponse.

Normalement, chaque augmentation double le temps, mais la présente spécification ne rend obligatoire aucune augmentation particulière.

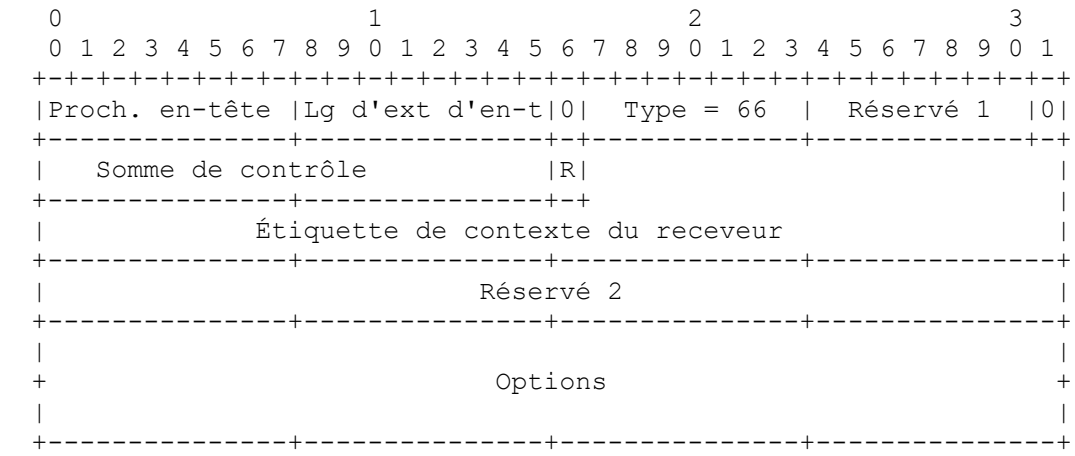
Note : La raison de l'envoi de quatre paquets à un taux fixe avant d'employer le retard exponentiel est d'éviter d'avoir à envoyer ces paquets excessivement rapidement. Sans cela, avoir 0,5 seconde entre la troisième et la quatrième sonde signifie que le temps entre la première et la seconde sonde devrait être de 0,125 seconde, ce qui donne très peu de temps pour qu'arrive une réponse au premier paquet. Aussi, cela signifie que les quatre premiers paquets sont envoyés en 0,875 seconde plutôt que 2 secondes, augmentant le potentiel d'encombrement si un grand nombre de contextes Shim6 ont besoin d'envoyer des sondes en même temps après une défaillance.

Finalement, Temporisation maximale de sonde *(Max Probe Timeout)* (60 secondes) spécifie une limite au delà de laquelle l'intervalle de sondage ne peut pas croître. Si le processus d'exploration atteint cet intervalle, il va continuer d'envoyer à ce rythme jusqu'à ce qu'une réponse convenable soit déclenchée ou que le contexte Shim6 soit mis au rebut, parce que les protocoles de couche supérieure utilisant le contexte Shim6 en question ne tentent plus d'envoyer des paquets. Atteindre Max Probe Timeout peut aussi servir d'indication au processus de mise au rebut de la collecte que le contexte n'est plus utilisable.

5. Définition du protocole

5.1 Message Keepalive

Le format du message Maintien en vie *(Keepalive)* est le suivant :



Prochain en-tête, Longueur d'extension d'en-tête, 0, 0, Somme de contrôle : ces champs sont comme spécifié au paragraphe 5.3 de la description du protocole Shim6 [RFC5533].

Type : ce champ identifie le message Maintien en vie *(Keepalive)* et DOIT être réglé à 66 (Keepalive).

Réservé 1 : champ de 7 bits réservé pour utilisation future. Il est réglé à zéro à l'émission et DOIT être ignoré à réception.

R : champ d' un bit réservé pour utilisation future. Il est réglé à zéro à l'émission et DOIT être ignoré à réception.

Étiquette de contexte du receveur : champ de 47 bits pour l'étiquette de contexte que le receveur a alloué au contexte.

Réservé 2 : champ de 32 bits réservé pour utilisation future. Il est réglé à zéro à l'émission et DOIT être ignoré à réception.

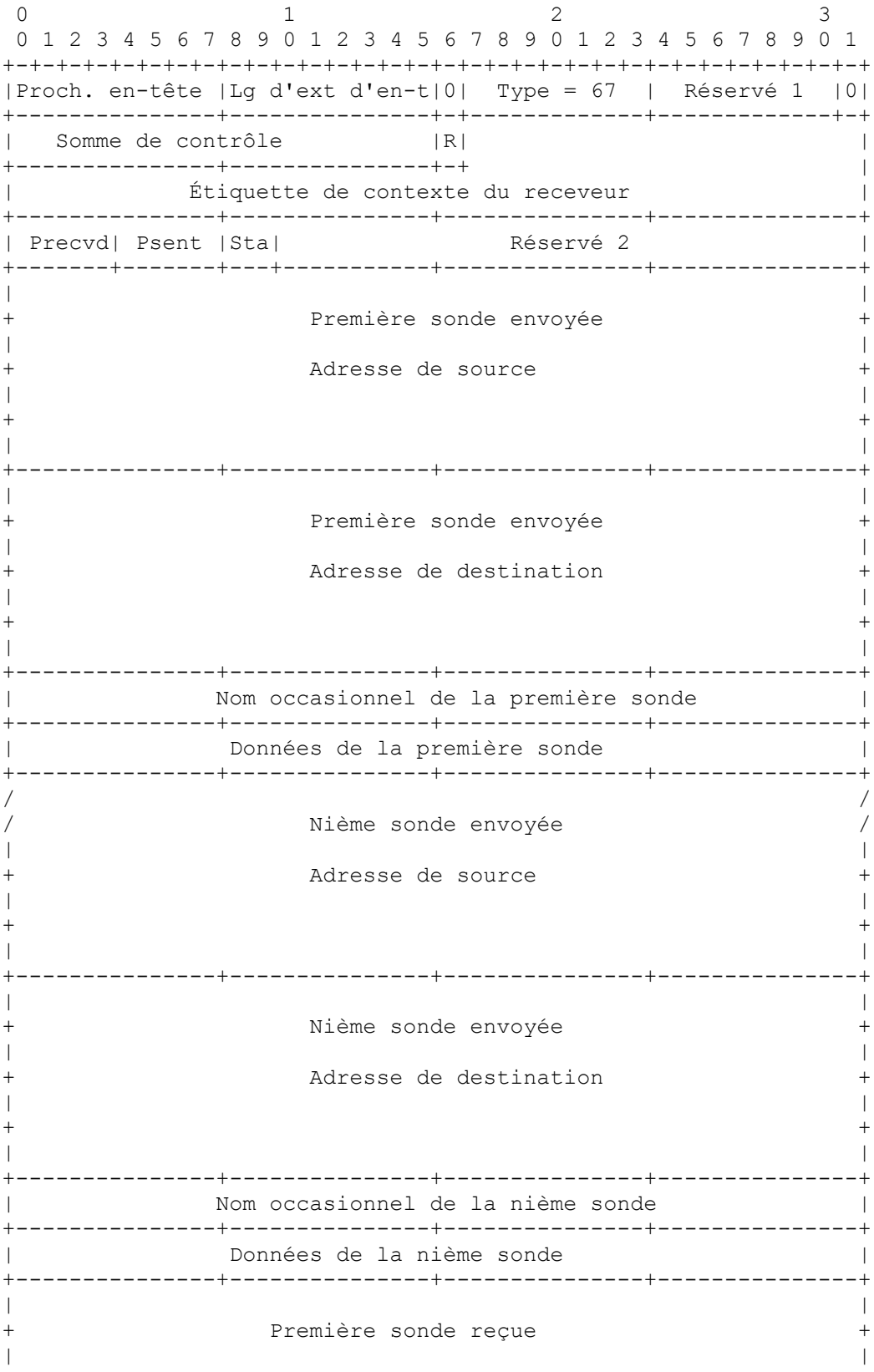
Options : ce champ PEUT contenir une ou plusieurs options Shim6. Cependant, il n'y a pas d'options actuellement définies qui soient utiles dans un message de maintien en vie. Le champ Options est fourni seulement pour des raisons de future extensibilité.

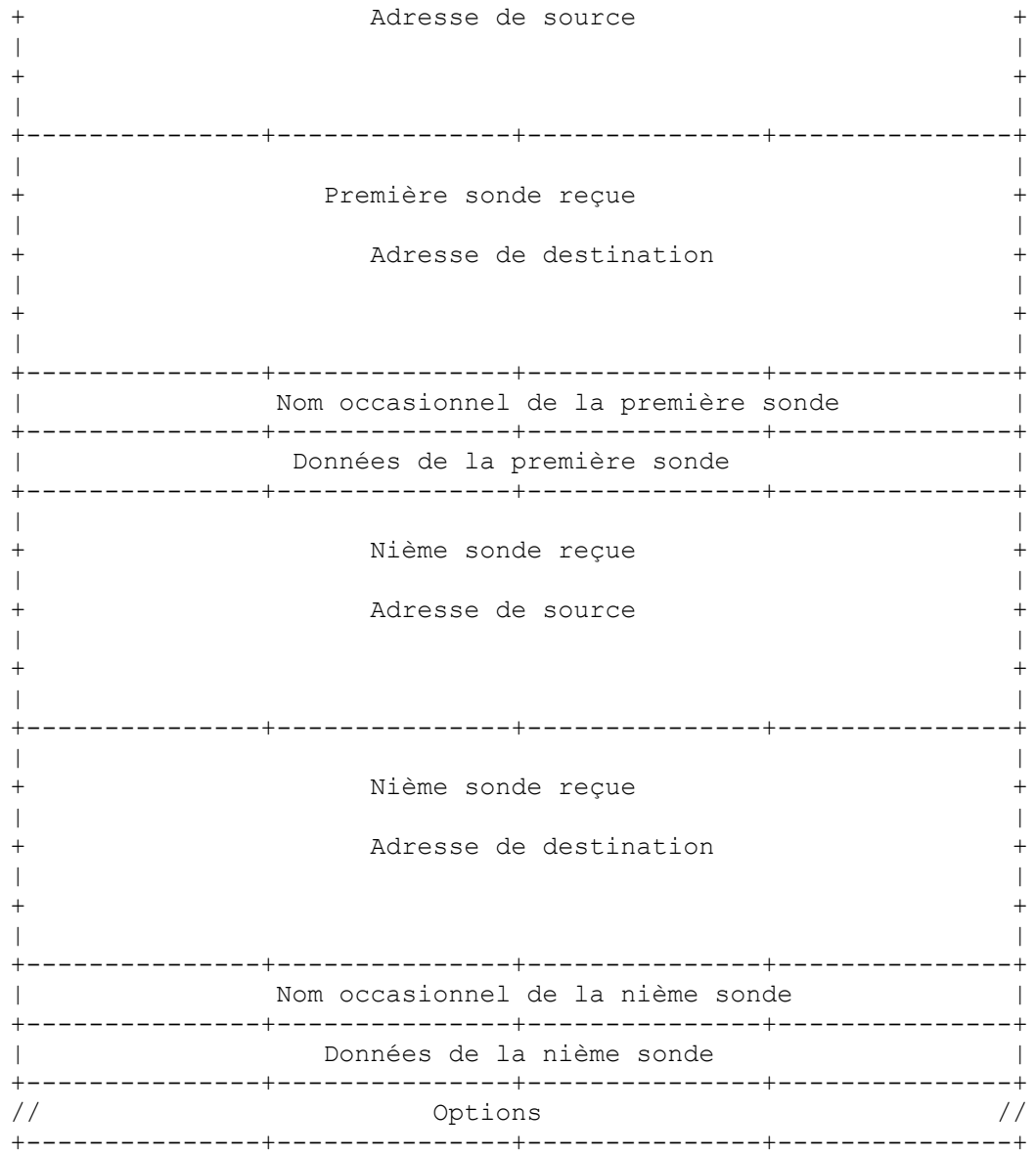
Un message valide se conforme au format ci-dessus, a une étiquette de contexte de receveur qui correspond au contexte connu du receveur, est un message de contrôle Shim6 valide comme défini au paragraphe 12.3 du protocole Shim6 [RFC5533], et a un contexte Shim6 dans l'état ESTABLISHED. Le receveur traite un message valide en inspectant ses options et en exécutant toute action spécifiée pour de telles options.

Les règles de traitement de ce message sont données plus en détail à la Section 6.

5.2 Message Probe

Ce message effectue l'exploration REAP. Son format est comme suit :





Prochain en-tête, Longueur d'extension d'en-tête, 0, 0, Somme de contrôle : ces champs sont comme spécifié au paragraphe 5.3 de la description du protocole Shim6 [RFC5533].

Type : ce champ identifie le message Probe et DOIT être réglé à 67 (Sonde).

Réservé 1 : champ de 7 bits réservé pour utilisation future. Il est réglé à zéro à l'émission et DOIT être ignoré à réception.

R : champ d' un bit réservé pour utilisation future. Il est réglé à zéro à l'émission et DOIT être ignoré à réception.

Étiquette de contexte du receveur : champ de 47 bits pour l'étiquette de contexte que le receveur a alloué au contexte.

Psent : champ de 4 bits qui indique le nombre de sondes envoyées incluses dans ce message Probe. Le premier ensemble de champs Probe appartient au message en cours et DOIT être présent, donc la valeur minimum de ce champ est 1. Des champs Probe envoyés supplémentaires sont copiés des mêmes champs envoyés (récemment) dans des sondes antérieures et peuvent être inclus ou omis selon la logique employée par la mise en œuvre.

Precvd : champ de 4 bits qui indique le nombre de sondes reçues incluses dans ce message Probe. Les champs de sondes reçues sont copiés des mêmes champs des sondes reçues antérieurement qui sont arrivées depuis la dernière transition à l'état Exploration. Quand un envoyeur est dans l'état InboundOk, il DOIT inclure des copies des champs d'au moins une des sondes entrantes. Un envoyeur PEUT inclure des ensembles supplémentaires de ces champ Sondes reçus dans tout état selon la logique employée par la mise en œuvre. Les champs Source de sonde, Destination de sonde, Nom

occasionnel de sonde, et Données de sonde peuvent être répétés, selon la valeur de Psent et Precvd.

Sta (état) : ce champ d'état de 2 bits est utilisé pour informer l'homologue de l'état de l'envoyeur. Il a trois valeurs légales :
0 (Opérationnel) implique que l'envoyeur (a) croit qu'il n'a pas de problème de communication et (b) croit que le receveur n'a pas non plus de problème de communication.
1 (Exploration) implique que l'envoyeur a un problème de communication avec le receveur, par exemple, il n'a pas vu de trafic provenant du receveur même quand il en attendait.
2 (InboundOk) implique que l'envoyeur croit qu'il n'a pas de problème de communication, c'est-à-dire, il a au moins vu des paquets du receveur mais soit le receveur a un problème, soit il n'a pas encore confirmé à l'envoyeur que le problème a été résolu.

Réservé 2 : DOIT être réglé à zéro à l'émission et DOIT être ignoré à réception.

Source de sonde : ce champ de 128 bits contient l'adresse IPv6 de source utilisée pour envoyer la sonde.

Destination de sonde : ce champ de 128 bits contient l'adresse IPv6 de destination utilisée pour envoyer la sonde.

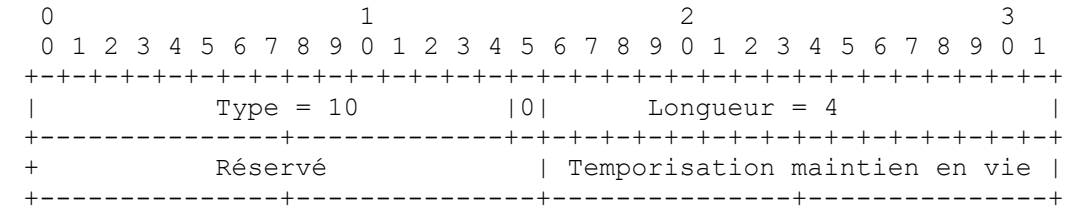
Nom occasionnel de sonde : c'est un champ de 32 bits qui est initialisé par l'envoyeur avec une valeur qui lui permet de déterminer avec quelles sondes envoyées se corrèle une sonde reçue. Il est fortement RECOMMANDÉ que le champ Nom occasionnel soit au moins modérément difficile à deviner afin que même des attaquants sur le chemin ne puissent pas déduire la prochaine valeur de nom occasionnel qui va être utilisée. Cette valeur DEVRAIT être générée en utilisant un générateur de nombres aléatoires connu pour avoir de bonnes propriétés d'aléa, comme mentionné dans la [RFC4086].

Données de sonde : champ de 32 bits sans signification fixe. Le champ Données de sonde est recopié sans changement. De futurs fanions pourront définir une utilisation pour ce champ.

Options : pour de futures extensions.

5.3 Format de l'option Temporisation de maintien en vie

L'un ou l'autre côté d'un contexte Shim6 peut notifier à l'homologue la valeur qu'il préférerait que l'homologue utilise comme valeur de temporisation de maintien en vie. Si le nœud utilise une valeur de temporisation d'envoi qui n'est pas par défaut, il DOIT communiquer cette valeur comme valeur de Temporisation de maintien en vie à l'homologue dans l'option ci-dessous. Cette option PEUT être envoyée dans les messages I2, I2bis, R2, ou UPDATE. L'option DEVRAIT seulement devoir être envoyée une fois dans une association Shim6 donnée. Si un nœud reçoit cette option, il DEVRAIT mettre à jour sa valeur de Temporisation de maintien en vie pour l'homologue.



Type : Ce champ identifie l'option et DOIT être réglé à 10 (Temporisation de maintien en vie).

Longueur : Ce champ DOIT être réglé comme spécifié au paragraphe 5.1 de la description du protocole Shim6 [RFC5533] -- c'est-à-dire, réglé à 4.

Réservé : champ de 16 bits réservé pour utilisation future. Réglé à zéro à l'émission et DOIT être ignoré à réception.

Temporisateur de maintien en vie : valeur en secondes correspondant à la valeur suggérée de temporisateur de maintien en vie pour l'homologue.

6. Comportement

Le comportement requis des nœuds REAP est spécifié ci-dessous sous la forme d'un automate à états. Le comportement

observable en externe d'une mise en œuvre DOIT se conformer à cet automate à états, mais il n'est pas exigé que la mise en œuvre emploie réellement un automate à états. Entremêlée à la description suivante, on donne aussi une description d'automate à états sous la forme de tableaux. Cependant, cette forme est seulement pour information.

Sur un contexte donné avec un certain homologue, le nœud peut être dans un des trois états suivants : Opérationnel, Exploration, ou InboundOK. Dans l'état Opérationnel, les paires d'adresses sous-jacentes sont supposées être opérationnelles. Dans l'état Exploration, ce nœud n'a pas vu de trafic provenant de l'homologue pendant une période de plus de "Temporisation d'envoi" *(Send Timer)*. Finalement, dans l'état InboundOK, ce nœud voit du trafic de l'homologue, mais l'homologue peut pas encore voir de trafic provenant de ce nœud, donc le processus d'exploration doit continuer.

Le nœud maintient aussi le temporisateur d'envoi (Send Timeout secondes) et le temporisateur de maintien en vie *(Keepalive Timer)* (Keepalive Timeout secondes). Le temporisateur d'envoi reflète l'exigence que quand ce nœud envoie un paquet de charge utile, il devrait y avoir du trafic de retour (des paquets de charge utile ou des messages de maintien en vie) dans les Send Timeout secondes. Le temporisateur de maintien en vie reflète l'exigence que quand ce nœud reçoit un paquet de charge utile, il devrait envoyer une réponse similaire à l'homologue. Le temporisateur de maintien en vie est seulement utilisé dans l'état Opérationnel, et le temporisateur d'envoi dans les états Opérationnel et InboundOK. Aucun temporisateur ne fonctionne dans l'état Exploration. Comme expliqué au paragraphe 4.1, les deux temporisateurs sont mutuellement exclusifs. C'est-à-dire, soit le temporisateur de maintien en vie, soit le temporisateur d'envoi, fonctionne, ou ni l'un ni l'autre.

Noter que l'Appendice A donne des exemples de fonctionnement normal du protocole afin d'illustrer le comportement.

6.1 Paquet de charge utile entrant

À réception d'un paquet de charge utile dans l'état Opérationnel, le nœud lance le temporisateur de maintien en vie si il ne court pas déjà, et arrête le temporisateur d'envoi si il courait.

Si le nœud est dans l'état Exploration, il passe à l'état InboundOK, envoie un message Probe, et lance le temporisateur d'envoi. Il remplit le champ Psent et les champs Adresse de source de sonde, Adresse de destination de sonde, Nom occasionnel de sonde, et Données de sonde correspondants avec les information sur les messages Sonde récents qui n'ont pas encore été rapportés comme vus par l'homologue. Il remplit aussi le Precvd et les champs Adresse de source de sonde, Adresse de destination de sonde, Nom occasionnel de sonde, et Données de sonde correspondants avec les information sur les messages Sonde récents qu'il a vus provenant de l'homologue. Lors de l'envoi d'un message Probe, le champ État DOIT être réglé à une valeur qui correspond à l'état conceptuel de l'envoyeur après l'envoi de la sonde. Dans ce cas, le nœud règle donc le champ État à 2 (InboundOk). Les adresses IP de source et de destination pour l'envoi du message Sonde sont choisies comme discuté au paragraphe 4.3.

Dans l'état InboundOK, le nœud arrête le temporisateur d'envoi si il courait, mais ne fait rien d'autre.

La réception de messages de contrôle Shim6 autre que les messages de maintien en vie et de sonde est traitée de la même façon que la réception des paquets de charge utile.

Pendant que le temporisateur de maintien en vie court, le nœud DEVRAIT envoyer des messages Maintien en vie à l'homologue à un intervalle de "Keepalive Interval" secondes. Conceptuellement, un temporisateur séparé est utilisé pour distinguer l'intervalle entre les messages de maintien en vie et l'intervalle global de temporisation de maintien en vie. Cependant,ce temporisateur séparé n'est pas modélisé dans les automates à états tabulaires ou graphiques. Quand il est envoyé, le message Maintien en vie est construit comme décrit au paragraphe 5.1. Il est envoyé en utilisant la paire d'adresses courante.

Dans les tableaux ci-dessous, "START", "RESTART", et "STOP" se réfèrent au lancement, à la relance, et à l'arrêt du temporisateur de maintien en vie ou du temporisateur d'envoi, respectivement. "GOTO" se réfère à la transition à un autre état. "SEND" se réfère à l'envoi d'un message, et "-" se réfère à "ne rien faire".

Opérationnel	**Exploration**	**InboundOk**
STOP Send	SEND Probe InboundOk	STOP Send
START Keepalive	START Send	
	GOTO InboundOk	

6.2 Paquet de charge utile sortant

À l'envoi d'un paquet de charge utile dans l'état Opérationnel, le nœud arrête le temporisateur de maintien en vie si il courait et lance le temporisateur d'envoi si il ne courait pas. Dans l'état Exploration cela n'a pas d'effet, et dans l'état InboundOK le nœud lance simplement le temporisateur d'envoi si il ne courait pas déjà. (L'envoi de messages de contrôle Shim6 est là aussi traité de la même façon.)

Opérationnel	**Exploration**	**InboundOk**
START Send STOP Keepalive	-	START Send

6.3 Temporisation de maintien en vie

Sur une fin de temporisation du temporisateur de maintien en vie, le nœud envoie un dernier message Keepalive. Cela peut seulement arriver dans l'état Opérationnel.

Le message Keepalive est construit comme décrit au paragraphe 5.1. Il est envoyé en utilisant la paire d'adresses courante.

Opérationnel	**Exploration**	**InboundOk**
SEND Keepalive	-	-

6.4 Fin de temporisation d'envoi

Sur une fin de temporisation du temporisateur d'envoi, le nœud entre dans l'état Exploration et envoie un message Probe. Le message Probe est construit comme expliqué au paragraphe 6.1, sauf que le champ État est réglé à 1 (Exploration).

Opérationnel	**Exploration**	**InboundOk**
SEND Probe Exploring GOTO Exploring	-	SEND Probe Exploring GOTO Exploring

6.5 Retransmission

Dans l'état Exploration, le nœud continue de retransmettre ses messages Probe aux différentes (ou les mêmes) adresses définies au paragraphe 4.3. Un processus similaire est employé dans l'état InboundOk, sauf que sur une telle retransmission, le temporisateur d'envoi est lancé si il ne courait pas déjà.

Les messages Probe sont construits comme expliqué au paragraphe 6.1, sauf que le champ État est réglé à 1 (Exploration) ou 2 (InboundOk) selon l'état dans lequel est l'envoyeur.

Opérationnel	**Exploration**	**InboundOk**
-	SEND Probe Exploring START Send	SEND Probe InboundOk

6.6 Réception du message Keepalive

À réception d'un message Keepalive dans l'état Opérationnel, le nœud arrête le temporisateur d'envoi si il courait. Si le nœud est dans l'état Exploration, il passe à l'état InboundOK, envoie un message Probe, et lance le temporisateur d'envoi. Le message Probe est construit comme expliqué au paragraphe 6.1.

Dans l'état InboundOK, le temporisateur d'envoi est arrêté si il courait.

Opérationnel	**Exploration**	**InboundOk**
STOP Send	SEND Probe InboundOk START Send GOTO InboundOk	STOP Send

6.7 Réception de l'état de message de sonde Exploring

À réception d'un message Probe avec État réglé à Exploration, le nœud entre dans l'état InboundOK, envoie un message Probe comme décrit au paragraphe 6.1, arrête le temporisateur de maintien en vie si il courait, et relance le temporisateur

d'envoi.

Opérationnel	**Exploration**	**InboundOk**
SEND Probe InboundOk	SEND Probe InboundOk	SEND Probe InboundOk
STOP Keepalive	START Send	RESTART Send
RESTART Send	GOTO InboundOk	
GOTO InboundOk		

6.8 Réception de l'état de message de sonde InboundOk

À réception d'un message Probe avec État réglé à InboundOk, le nœud envoie un message Probe, relance le temporisateur d'envoi, arrête le temporisateur de maintien en vie si il courait, et passe à l'état Opérationnel. Une nouvelle paire d'adresses courante est choisie pour la connexion, sur la base des rapports de sondes reçues dans le message qui vient d'être reçu. Si aucune sonde reçue n'a été rapportée, la paire d'adresses courante est inchangée.

Le message Probe est construit comme expliqué au paragraphe 6.1, sauf que le champ État est réglé à zéro (Opérationnel).

Opérationnel	**Exploration**	**InboundOk**
SEND Probe Opérationnel	SEND Probe Opérationnel	SEND Probe Opérationnel
RESTART Send	RESTART Send	RESTART Send
STOP Keepalive	GOTO Opérationnel	GOTO Opérationnel

6.9 Réception de l'état de message de sonde Operational

À réception d'un message Probe avec État réglé à Opérationnel, le nœud arrête le temporisateur d'envoi si il courait, lance le temporisateur de maintien en vie si il ne courait pas déjà, et passe à l'état Opérationnel. Le message Probe est construit comme expliqué au paragraphe 6.1, sauf que le champ État est réglé à zéro (Opérationnel).

Note : cela termine le processus d'exploration quand les deux parties sont contentes et savent que leur homologue est aussi content.

Opérationnel	**Exploration**	**InboundOk**
STOP Send	STOP Send	STOP Send
START Keepalive	START Keepalive	START Keepalive
	GOTO Opérationnel	GOTO Opérationnel

La détection d'accessibilité et le processus d'exploration n'ont pas d'effet sur les communications de charge utile jusqu'à ce qu'une nouvelle paire d'adresses opérationnelle ait été confirmée. Avant cela, les paquets de charge utile continuent d'être envoyés aux adresses précédemment utilisées.

6.10 Représentation graphique de l'automate à états

Dans la version PDF de cette spécification, un dessin pour information illustre l'automate à états. Lorsque le texte et le dessin diffèrent, le texte a la préséance.

7. Constantes et variables du protocole

Les constantes de protocole suivantes sont définies :
Temporisation initiale de sonde : 0,5 seconde
Nombre de sondes initiales : 4 sondes

Et les variables ont les valeurs par défaut suivantes :
Temporisation d'envoi : 15 secondes
Temporisation de maintien en vie : X secondes, où X est la temporisation d'envoi de l'homologue communiquée dans l'option Temporisation de maintien en vie de 15 secondes si l'homologue n'a pas envoyé d'option Temporisation de maintien en vie.
Intervalle de maintien en vie : Y secondes, où Y est entre un tiers et la moitié de la valeur de Temporisation de maintien en vie (voir le paragraphe 4.1)

D'autres valeurs de temporisation d'envoi peuvent être choisies par un nœud et communiquées à l'homologue dans l'option Temporisation de maintien en vie. Une très petite valeur de la temporisation d'envoi peut affecter la capacité d'échanger des maintiens en vie sur un chemin qui a un long délai d'aller-retour. De même, il peut causer la réaction de Shim6 à des défaillances temporaires plus souvent que nécessaire. Par suite, il est RECOMMANDÉ qu'une autre valeur de temporisation d'envoi ne soit pas en dessous de 10 secondes. Choisir une valeur plus élevée que celle recommandée ci-dessus est aussi possible, mais il y a une relation entre la temporisation d'envoi et la capacité de REAP à découvrir et corriger les erreurs dans le chemin de communication. En tous cas, afin que Shim6 soit utile, il devrait détecter et réparer les problèmes de communication longtemps avant que les couches supérieures abandonnent. Pour cette raison, il est RECOMMANDÉ que la temporisation d'envoi soit au plus de 100 secondes (temporisation TCP R2 par défaut [RFC1122]).

Note : on ne s'attend pas à ce que la temporisation d'envoi ou d'autres valeurs soient estimées sur la base des temps d'aller-retour rencontrés. Les échanges de signalisation sont effectués sur la base du retard exponentiel. Les processus de maintien en vie envoient des paquets seulement dans la condition relativement rare où tout le trafic est unidirectionnel.

8. Considérations sur la sécurité

Des attaquants peuvent falsifier diverses indications provenant des couches inférieures et du réseau afin de confondre les homologues sur les adresses qui sont ou non opérationnelles. Par exemple, des attaquants peuvent falsifier des messages d'erreur ICMP afin de faire que les parties déplacent leur trafic ailleurs ou même se déconnectent. Des attaquants peuvent aussi falsifier des informations relatives aux rattachements au réseau, à la découverte de routeur, et aux allocations d'adresse afin de faire croire aux parties qu'elles ont la connexité Internet alors qu'elles ne l'ont pas.

Cela peut causer l'utilisation d'adresses non préférées ou même de déni de service.

Ce protocole ne fournit par lui-même pas de protection pour les indications provenant d'autres parties de la pile de protocoles. Les indications non protégées NE DEVRAIENT PAS être prises comme des preuves de problèmes de connexité. Cependant, REAP a une faible résistance contre les informations incorrectes même provenant d'indications non protégées dans le sens où il effectue ses propres essais avant de prendre une nouvelle paire d'adresses. La vulnérabilité au déni de service reste cependant, comme les vulnérabilités à des attaquants sur le chemin.

Certains aspects de ces vulnérabilités peuvent être atténués par l'utilisation de techniques spécifiques des autres parties de la pile, comme de traiter correctement les erreurs ICMP [RFC5927], la sécurité de la couche de liaison, ou l'utilisation de SEND [RFC3971] pour protéger le routeur IPv6 et la découverte de voisin.

D'autres parties du protocole Shim6 assurent que l'ensemble d'adresses entre lesquelles on commute sont réellement ensemble. REAP ne donne pas par lui-même une telle assurance. De même, REAP donne une certaine protection contre les attaques d'inondation par des tiers [AURA02] ; quand REAP fonctionne, ses noms occasionnels de sonde peuvent être utilisés comme vérification d'acheminement de retour que l'adresse revendiquée veut bien recevoir du trafic. Cependant, cela doit être complété par un autre mécanisme pour assurer que l'adresse revendiquée est aussi le nœud correct. Shim6 fait cela en effectuant un lien de toutes les opérations avec les étiquettes de contexte.

Le mécanisme de maintien en vie de cette spécification est vulnérable à la falsification. Des attaquants en chemin qui peuvent voir une étiquette de contexte Shim6 peuvent envoyer des messages de maintien en vie falsifiés une fois par intervalle de temporisation d'envoi afin d'empêcher deux nœuds Shim6 d'envoyer eux-mêmes des messages de maintien en vie. Cette vulnérabilité est seulement pertinente pour les nœuds impliqués dans une communication unidirectionnelle. Le résultat de l'attaque est que les nœuds entrent sans nécessité dans la phase d'exploration, mais ils devraient être capables de confirmer la connexité sauf, bien sûr, si l'attaquant est capable d'empêcher la phase d'exploration de s'achever. Des attaquants hors chemin ne peuvent pas être capables de générer des résultats falsifiés, parce que les étiquettes de contexte sont des nombres aléatoires de 47 bits.

Pour se protéger contre les messages de maintien en vie falsifiés, un nœud qui met en œuvre Shim6 et IPsec PEUT ignorer les maintiens en vie REAP entrants si il a de bonnes raisons de penser que l'autre côté va envoyer du trafic de retour protégé par IPsec. En d'autres termes, si un nœud envoie des données de charge utile TCP, on peut raisonnablement s'attendre à recevoir des ACK TCP en retour. Si aucun ACK protégé par IPsec ne revient mais que viennent des maintiens en vie non protégés, cela pourrait être le résultat d'une attaque qui essaye de cacher la perte de connexité.

La phase d'exploration est vulnérable à des attaquants qui sont sur le chemin. Les attaquants hors chemin trouveraient difficile de deviner l'étiquette de contexte ou les identifiants de sonde corrects. Étant donné que IPsec opère au dessus de la couche Shim6, il n'est pas possible de protéger avec IPsec la phase d'exploration contre des attaquants en chemin. Ceci est similaire aux problèmes de protection des autres échanges de contrôle Shim6. Il y a des mécanismes en place pour empêcher la redirection des communications sur de mauvaises adresses, mais des attaquants en chemin peuvent causer un déni de service, déplacer les communications sur des paires d'adresses moins préférées, et ainsi de suite.

Finalement, l'exploration elle-même peut causer l'envoi d'un certain nombre de paquets. Par suite, elle peut être utilisée comme outil pour l'amplification dans des attaques d'inondation. Il est exigé que le protocole qui emploie REAP ait des mécanismes incorporés pour empêcher cela. Par exemple, les contextes Shim6 sont créés seulement après qu'un nombre relativement grand de paquets ont été échangés, ce qui a un coût qui réduit l'intérêt d'utiliser Shim6 et REAP pour des attaques d'amplification. Cependant, de telles protections ne sont normalement pas présentes au moment de l'établissement de la connexion. Quand l'exploration va être nécessaire pour que l'établissement de la connexion réussisse, son usage va résulter en une vulnérabilité à l'amplification. Par suite, Shim6 ne prend pas en charge l'utilisation de REAP dans l'étape d'établissement de connexion.

9. Considérations de fonctionnement

Quand il n'y a pas de défaillances, le mécanisme de détection de défaillance (et Shim6 en général) est léger : les maintiens en vie ne sont pas envoyés quand un contexte Shim6 est au repos ou quand il y a du trafic dans les deux directions. Donc dans des opérations normales de TCP ou de style TCP, il va seulement y avoir un ou deux maintiens en vie quand une session passe de actif à repos.

C'est seulement quand il y a des défaillances qu'il y a un trafic de détection de défaillance significatif, en particulier dans le cas où une liaison qui est partagée par de nombreuses sessions actives et plusieurs nœuds est interrompue. Quand cela arrive, un maintien en vie est envoyé et ensuite une série de sondes. Cela arrive par contexte actif (générant du trafic) dont tous vont arriver en fin de temporisation dans les 15 secondes après la défaillance. Cela fait du trafic de pointe que Shim6 génère après une défaillance environ un paquet par seconde par contexte. On peut supposer que les sessions qui fonctionnent sur ces contextes ont envoyé au moins cette quantité de trafic et très probablement plus, mais si le chemin de secours est d'une bande passante significativement inférieure à celle du chemin défaillant, cela pourrait conduire à de l'encombrement temporaire.

Cependant, on notera que dans le cas de multi rattachements en utilisant BGP, si la reprise sur défaillance est assez rapide pour que TCP ne passe pas en démarrage lent, tout le trafic de données de charge utile qui s'écoule sur le chemin défaillant est commuté sur le chemin de secours, et si ce chemin de secours est d'une capacité inférieure, il va y avoir encore plus d'encombrement.

Bien que le sondage de détection de défaillance n'effectue pas de contrôle d'encombrement en tant que tel, le retard exponentiel assure que le nombre de paquets envoyé diminue rapidement et atteint finalement un par contexte par minute, ce qui devrait être suffisamment prudent même sur les liaisons de plus faible bande passante.

La Section 7 spécifie un certain nombre de paramètres du protocole. Le réglage possible de ces paramètres et des autres qui ne sont pas obligatoires dans cette spécification peut affecter ces propriétés. On s'attend à ce que de futures révisions de la spécification donnent des informations supplémentaires après qu'une expérience suffisante de déploiement aura été obtenue de différents environnements.

Les mises en œuvre peuvent fournir des moyens de surveiller leurs performances et envoyer des alarmes sur les problèmes. Leur normalisation est, cependant, le sujet de spécifications futures. En général, Shim6 est bien applicable pour les petits sites et nœuds, et il est espéré que les exigences de surveillance sur de tels déploiements sont relativement modestes. Dans tous les cas, lorsque le nœud est associé à un système de gestion, il est RECOMMANDÉ que les défaillances détectées et les événements de reprise sur défaillance soient rapportés via des notifications asynchrones au système de gestion. De même, lorsque des mécanismes de journaux d'événements sont disponibles sur le nœud, ces événements devrait être enregistrés dans les journaux d'événements.

Shim6 utilise le même en-tête pour la signalisation et l'encapsulation de paquets de charge utile après un événement de rattachement. De cette façon, le sort est partagé entre les deux types de paquets, de sorte que la situation où les sondes d'accessibilité ou les maintiens en vie peuvent être bien transmis mais pas les paquets de charge utile, est largement évitée : soit tous les paquets Shim6 passent et donc Shim6 fonctionne comme prévu, soit aucun ne passe, et aucun état Shim6 n'est négocié. Même dans la situation où certains paquets passent et pas d'autres, Shim6 va généralement fonctionner comme

prévu ou fournir un service qui n'est pas pire que l'absence de Shim6, à part la possible génération d'une petite quantité de trafic de signalisation.

Parfois des paquets de charge utile (et éventuellement des paquets de charge utile encapsulés dans l'en-tête Shim6) ne vont pas passer, mais ceux de signalisation et de maintien en vie passent. Cette situation peut se produire quand il y a un trou noir de découverte de la MTU de chemin sur un des chemins. Si seulement de grands paquets sont envoyés à un moment, l'exploration d'accessibilité va être activée et REAP va probablement choisir un autre chemin, qui peut ou non être affecté par le trou noir de PMTUD.

10. Références

10.1 Références normatives

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC3315] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins et M. Carney, "Protocole de configuration dynamique d'hôte pour IPv6 (DHCPv6)", juillet 2003. (*MàJ par* RFC6422 et RFC6644, RFC7227 *; rendue obsolète par* RFC8415)

[RFC3484] R. Draves, "Choix d'adresse par défaut pour le protocole Internet version 6 (IPv6)", février 2003. (*Remplacée par la* RFC6724) *(P.S.)*

[RFC4086] D. Eastlake 3rd, J. Schiller, S. Crocker, "Exigences d'aléa pour la sécurité", juin 2005, DOI 10.17487/RFC4086, (*Remplace* RFC1750) (BCP0106)

[RFC4193] R. Hinden, B. Haberman, "Adresses IPv6 en envoi individuel uniques localement", octobre 2005. *(P.S.)*

[RFC4429] N. Moore, "Détection optimiste d'adresse dupliquée (DAD) pour IPv6", avril 2006. *(P.S.)*

[RFC4861] T. Narten et autres, "Découverte du voisin pour IP version 6 (IPv6)", septembre 2007. (*Remplace* RFC2461) *(D.S. ; MàJ par* RFC8028, RFC8319, RFC8425, RFC9131)

[RFC4862] S. Thomson et autres, "Auto configuration d'adresse IPv6 sans état", septembre 2007. (*Remplace* RFC2462) *(D.S.)*

[RFC5533] E. Nordmark, M. Bagnulo, "Shim6 : Protocole Shim de niveau 3 de multi rattachement pour IPv6", juin 2009. *(P. S.)*

10.2 Références pour information

[ADD-SEL] Bagnulo, M., "Address selection in multihomed environments", Travail en cours, octobre 2005.

[AURA02] Aura, T., Roe, M., et J. Arkko, "Security of Internet Location Management", Proceedings of the 18th Annual Computer Security Applications Conference, Las Vegas, Nevada, USA, décembre 2002.

[MULTI6] Huitema, C., "Address selection in multihomed environments", Travail en cours, octobre 2004.

[PAIR] Bagnulo, M., "Default locator-pair selection algorithm for the Shim6 protocol", Travail en cours, octobre 2008.

[RFC1122] R. Braden, "Exigences pour les hôtes Internet – couches de communication", STD 3, DOI 10.17487/RFC1122, octobre 1989. *(MàJ par* RFC6633, 8029, 9293)

[RFC3971] J. Arkko et autres, "Découverte de voisin sûre (SEND)", mars 2005. (*MàJ par* RFC6494) *(P.S.)*

[RFC4960] R. Stewart, éd., "Protocole de transmission de commandes de flux (SCTP)", septembre 2007. (*Remplace* RFC2960, RFC3309 ; *P.S. ; Remplacée par* RFC9260)

[RFC5206] P. Nikander et autres, "Mobilité et rattachement multiple d'hôte d'extrémité avec le protocole d'identité d'hôte", avril 2008. *(Expérimentale ; remplacée par* RFC8046)

[RFC5880] D. Katz, D. Ward, "Détection de transmission bidirectionnelle (BFD)", juin 2010. *(P. S. ; MàJ par* RFC7880)

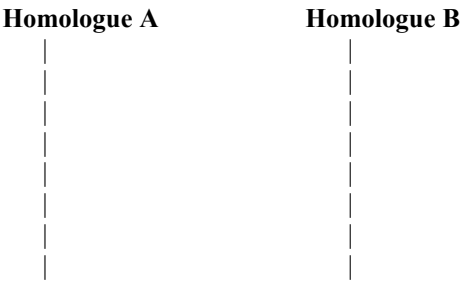
[RFC5927] F. Gont, "Attaques ICMP contre TCP", DOI 10.17487/RFC5927, juillet 2010. *(Information)*

[RFC6059] S. Krishnan, G. Daley, "Procédures simples pour détecter le rattachement au réseau en IPv6", novembre 2010. *(P.S.)*

Appendice A. Exemple de tours de protocole

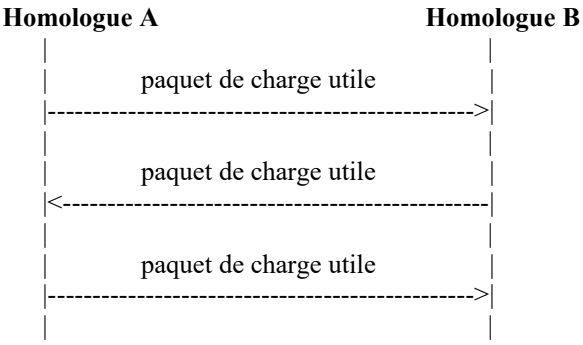
Cet appendice donne des exemples de fonctionnement du protocole REAP dans des scénarios typiques. On commence par le plus simple scénario de deux nœuds, A et B, qui ont une connexion Shim6 l'un avec l'autre mais n'envoient actuellement aucune donnée de charge utile. Comme aucun des côtés n'envoie, ils n'attendent rien en retour, donc il n'y a pas de messages du tout :

Exemple 1 : pas de communication



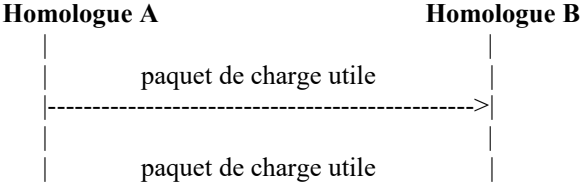
Notre second exemple implique une connexion active avec des flux bidirectionnels de paquets de charge utile. Ici, la réception des données de charge utile provenant de l'homologue est prise comme une indication d'accessibilité, donc là encore il n'y a pas de paquets supplémentaires :

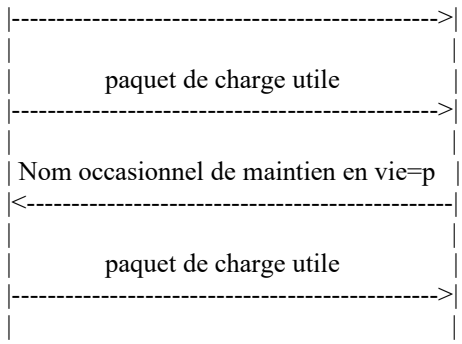
Exemple 2 : Communications bidirectionnelles



Le troisième exemple est le premier qui implique un message REAP réel. Ici, les nœuds communiquent juste dans une direction, de sorte que des messages REAP sont nécessaires pour indiquer à l'homologue qui envoie des paquets de charge utile que ses paquets passent :

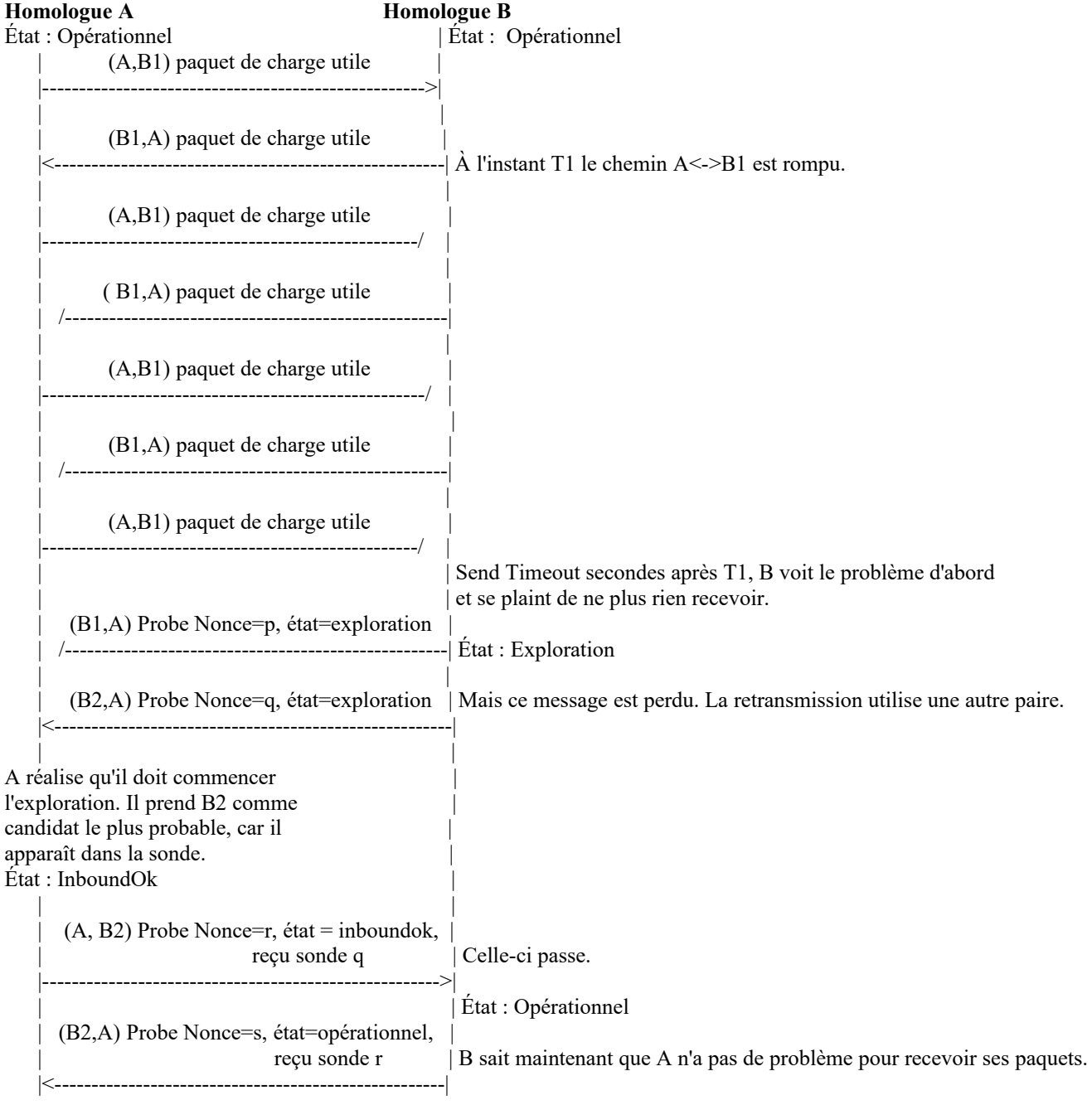
Exemple 3 : Communications unidirectionnelles

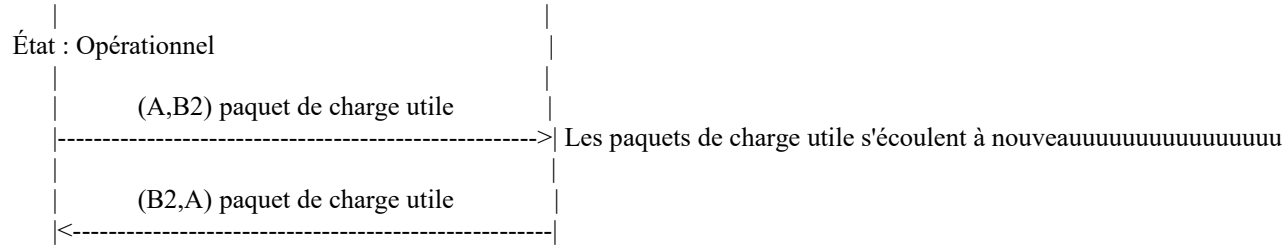




L'exemple suivant implique un scénario de défaillance. Ici, A a l'adresse A, et B a les adresses B1 et B2. Les paires d'adresses actuellement utilisées sont (A, B1) et (B1, A). Toutes les connexions via B1 sont rompues, ce qui conduit à un processus d'exploration :

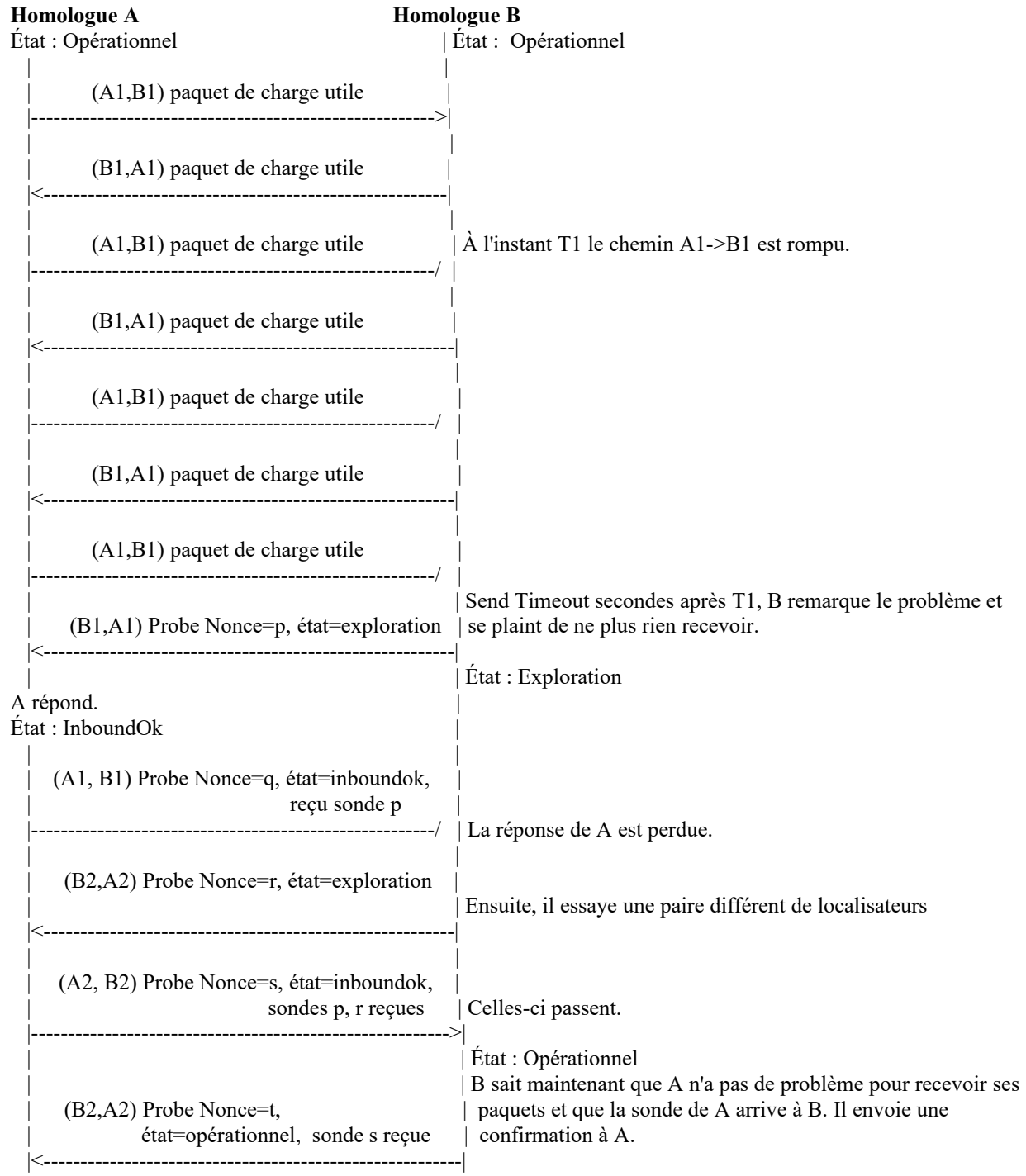
Exemple 4 : Scénario de défaillance

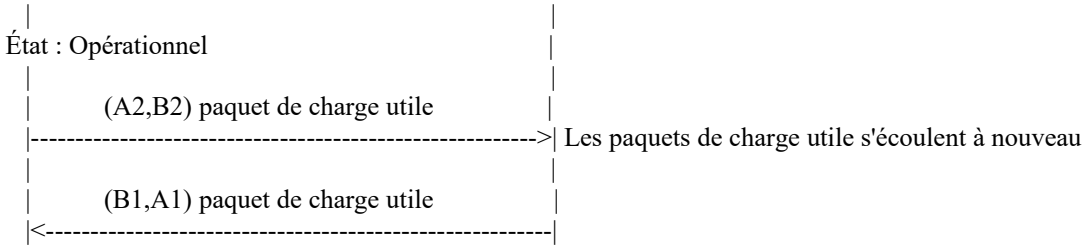




L'exemple suivant montre quand la défaillance pour la paire courante de localisateurs est seulement dans l'autre direction. A a les adresses A1 et A2, et B a les adresses B1 et B2. La communication en cours est entre A1 et B1, mais les paquets de A n'atteignent plus B en utilisant cette paire.

Exemple 5 : défaillance unidirectionnelle





Appendice B. Contributeurs

Le présent document tente de résumer les idées et les contributions non publiées de nombreuses personnes, incluant les membres de l'équipe de conception du groupe de travail MULTI6, Marcelo Bagnulo Braun, Erik Nordmark, Geoff Huston, Kurtis Lindqvist, Margaret Wasserman, et Jukka Ylitalo, les contributeurs au groupe de travail MOBIKE Pasi Eronen, Tero Kivinen, Francis Dupont, Spencer Dawkins, et James Kempf, et des contributeurs au groupe de travail HIP comme Pekka Nikander. Ce document doit aussi beaucoup au travail fait dans le contexte de SCTP [RFC4960] et de l'extension de multi ratachement et mobilité du protocole d'identité d'hôte (HIP, *Host Identity Protocol*) [RFC5206].

Appendice C. Remerciements

Les auteurs tiennent aussi à remercier Christian Huitema, Pekka Savola, John Loughney, Sam Xia, Hannes Tschofenig, Sebastien Barre, Thomas Henderson, Matthijs Mekking, Deguang Le, Eric Gray, Dan Romascanu, Stephen Kent, Alberto Garcia, Bernard Aboba, Lars Eggert, Dave Ward, et Tim Polk des discussions intéressantes dans cet espace de problème, et de leur relecture de cette spécification.

Adresse des auteurs

Jari Arkko
Ericsson
Jorvas 02420
Finlande
mél : jari.arkko@ericsson.com

Iljitsch van Beijnum
IMDEA Networks
Avda. del Mar Mediterraneo, 22
Leganes, Madrid 28918
Espagne
mél : iljitsch@muada.com