Groupe de tarvail Réseau
Request for Comments : 3706
Catégorie : Information

Traduction Claude Brière de L'Isle

G. Huang
S. Beaulieu
D. Rochefort
Cisco Systems, Inc.
février 2004

Méthode de détection des homologues d'échange de clé Internet (IKE) morts fondée sur le trafic

Statut de ce mémoire
Le présent mémoire apporte des informations pour la communauté de l'Internet. Le présent mémoire ne spécifie aucune sorte de norme de l'Internet. La distribution du présent mémoire n'est soumise à aucune restriction.

Notice de copyright
Copyright (C) The Internet Society (2004). Tous droits réservés.

Résumé
Le présent document décrit la méthode pour détecter un homologue d'échange de clé Internet (IKE, *Internet Key Exchange*) mort, qui est présentement utilisée par un certain nombre de fabricants. La méthode, appelée détection d'homologue mort (DPD, *Dead Peer Detection*) utilise le schéma de trafic IPsec pour minimiser le nombre de messages IKE nécessaires pour confirmer la vivacité. DPD, comme d'autres mécanismes de maintien en vie, est nécessaire pour déterminer quand effectuer la reprise sur défaillance d'homologue IKE, et pour réclamer les ressources perdues.

Table des Matières

1. Introduction

Lorsque deux homologues communiquent avec IKE [RFC2409] et IPsec [RFC2401], il PEUT survenir une situation dans laquelle la connexité entre les deux s'interrompt de façon inattendue. Cette situation peut arriver à cause de problèmes d'acheminement, du réamorçage d'un hôte, etc., et dans un tel cas, il n'y a souvent aucun moyen pour IKE et IPsec d'identifier la perte de la connexité de l'homologue. À ce titre, les SA peuvent rester jusqu'à ce que leur durée de vie expire naturellement, résultant en une situation de "trou noir" où les paquets sont tunnelés vers l'oubli. Il est souvent désirable de reconnaître les trous noirs aussitôt que possible afin qu'une entité puisse faire rapidement une reprise sur défaillance sur un homologue différent. De même, il est parfois nécessaire de détecter les trous noirs pour récupérer les ressources perdues.

Ce problème de détection d'un homologue IKE mort a été traité par des propositions qui exigent d'envoyer des messages

périodiques HELLO/ACK pour prouver la vivacité. Ces schémas tendent à être unidirectionnels (seulement un HELLO) ou bidirectionnels (une paire HELLO/ACK). Pour les besoins du présent document, le terme "battement de cœur" se réfère à un message unidirectionnel pour prouver la vivacité. De même, le terme "garder en vie" se réfère à un message bidirectionnel.

Le problème avec les propositions courantes de battement de cœur et de garder en vie est qu'ils s'appuient sur l'envoi des messages à des intervalles réguliers. Dans la mise en œuvre, cela se traduit par la gestion d'un temporisateur pour gérer les intervalles de messages. De même, parce que la détection rapide des homologues morts est souvent désirée, ces messages DOIVENT être envoyés à une certaine fréquence, se traduisant là encore en une redondance considérable pour le traitement des messages. Dans les mises en œuvre et les installations où la gestion de grands nombres de sessions IKE simultanées est un problème, ces battements de cœur et garder en vie réguliers se révèlent infaisables.

À cette fin, un certain nombre de fabricants ont mis en œuvre leur propre approche pour détecter la vivacité des homologues sans avoir besoin d'envoyer des messages à des intervalles réguliers. Le présent document d'information décrit la pratique actuelle de ces mises en œuvre. Ce schéma, appelé détection de l'homologue mort (DPD, *Dead Peer Detection*), s'appuie sur les messages Notify de IKE pour interroger la vivacité d'un homologue IKE.

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].

2. Objectifs du document

Comme mentionné ci-dessus, des solutions sont déjà proposées au problème de détection des homologues morts. La Section 3 expose les raisons d'utiliser un échange de message IKE pour interroger sur la vivacité d'un homologue. La Section 4 examine l'approche fondée sur garder en vie ainsi que l'approche fondée sur le battement de cœur. La Section 5 présente la proposition DPD complète, en soulignant les différences entre DPD et les schémas présentées à la Section 4 et en insistant sur les questions d'adaptabilité. La Section 6 examine les questions de sécurité autour de la répétition de messages et de fausse vivacité.

3. Raisons de l'échange périodique de messages comme preuve de vie

Comme mentionné dans l'introduction, il est souvent nécessaire de détecter qu'un homologue est injoignable aussitôt que possible. IKE ne donne pas de moyen pour le faire – à part d'attendre jusqu'à la période de changement de clés, puis de tenter le changement de clés (et d'y échouer). Il va en résulter une période de perte de connexité qui dure pour le reste de la durée de vie de l'association de sécurité (SA, *Security Association*) et dans la plupart des déploiements, ceci est inacceptable. Une méthode est donc nécessaire pour vérifier à volonté l'état d'un homologue. Différentes méthodes se sont fait jour, utilisant généralement un Notify IKE pour interroger la vivacité de l'homologue. Ces méthodes s'appuient soit sur un échange de message "garder en vie" bidirectionnel (un HELLO suivi d'un ACK) soit d'un échange de message "battement de cœur" unidirectionnel (un HELLO seul). La section suivante examine ces deux schémas.

4. Garder en vie ou battements de cœur

4.1 Garder en vie

Considérons un schéma "garder en vie" dans lequel un homologue A et un homologue B exigent l'un de l'autre des accusés de réception réguliers de leur vivacité. Les messages sont échangés au moyen d'une charge utile Notify authentifiée. Les deux homologues DOIVENT s'accorder sur l'intervalle auquel les "garder en vie" sont envoyés, ce qui signifie qu'une certaine forme de négociation est exigée durant la phase 1. Pour qu'une prompte reprise sur défaillance soit possible, le garder en vie DOIT aussi être envoyé à des intervalles assez fréquents – autour de 10 secondes à peu près. Dans ce scénario hypothétique de garder en vie, les homologues A et B s'accordent pour échanger des garder en vie toutes les 10 secondes. Essentiellement, toutes les 10 secondes, un des homologues DOIT envoyer un HELLO à l'autre. Ce HELLO sert de preuve de vie pour l'entité qui envoie. À son tour, l'autre homologue DOIT accuser réception de chaque HELLO de garder en vie. Si les 10 secondes s'écoulent, et si un côté n'a pas reçu de HELLO, il va envoyer le message HELLO lui-même, utilisant le ACK de l'homologue comme preuve de vie. La réception d'un HELLO ou d'un ACK cause la remise à zéro du temporisateur de garder en vie d'une entité. Manquer à recevoir un ACK dans un certain délai signale une erreur. Des précisions sont données ci-dessous.

Scénario 1 : Le temporisation de 10 secondes de l'homologue A expire et il envoie un HELLO à B. B répond par un ACK.

Homologue A :		**Homologue B :**
Le temporisateur de 10 secondes expire ;	------>	
Il veut savoir si B est vivant ; il envoie un HELLO.		
		Reçoit le HELLO ; accuse réception de la vie de A ;
	<------	rétablit le temporisateur garder en vie , envoie le ACK.
Reçoit le ACK comme preuve de vie de B ;		
Rétablit le temporisateur.		

Scénario 2 : le temporisation de 10 secondes de l'homologue A expire en premier, et il envoie un HELLO à B. B ne répond pas. A peut retransmettre, au cas où le HELLO initial serait perdu. Cette situation décrit comment l'homologue A détecte que son homologue est mort.

Homologue A :		**Homologue B (mort) :**
Le temporisateur de 10 secondes expire ;	------X	
Il veut savoir si B est vivant ; il envoie un HELLO.		
Le temporisateur de retransmission expire ;	------X	
Le message initial pourrait être perdu en transit ;		
A incrémente le compteur d'erreurs		
et envoie un autre HELLO.	------X	

Après un certain nombre d'erreurs, A suppose que B est mort; supprime les SA et initie éventuellement une reprise sur défaillance.

L'avantage de ce schéma est que la partie intéressée à la vie de l'autre homologue commence l'échange de messages. Dans le scénario 1, l'homologue A est intéressé à la vie de l'homologue B, et par conséquent l'homologue A envoie le HELLO. Il est concevable dans un tel schéma que l'homologue B ne soit jamais intéressé par la vie de l'homologue A. Dans un tel cas, la charge d'initier l'échange va toujours incomber à l'homologue A.

4.2 Battements de cœur

À l'opposé, considérons un schéma de preuve de vie impliquant des messages unidirectionnels (sans accusé de réception). Une entité intéressée par la vie de son homologue va s'appuyer sur l'homologue lui-même pour envoyer des messages périodiques montrant qu'il est en vie. Dans ce schéma, l'échange de messages pourrait ressembler à ceci :

Scénario 3 : l'homologue A et l'homologue B sont intéressé par la vie de l'un et l'autre. Chaque homologue dépend de l'autre pour envoyer des HELLO périodiques.

Homologue A :		**Homologue B :**
Le temporisateur de 10 secondes expire ;	------>	
Il envoie un HELLO.		Reçoit le HELLO comme preuve de vie de A
Le temporisateur signale l'attente du HELLO de B.		
	<------	Le temporisateur de 10 secondes expire ; envoi du HELLO.
Reçoit le HELLO comme preuve de vie de B		.

Scénario 4 : l'homologue A ne reçoit pas de HELLO de B et marque l'homologue comme mort. C'est comme cela qu'une entité détecte la mort de son homologue.

Homologue A :		**Homologue B (mort) :**
Le temporisateur de 10 secondes expire ;	------X	
Il envoie un HELLO.		
Le temporisateur signale aussi l'attente du HELLO de B.	------X	
Du temps passe et A suppose B mort.		

L'inconvénient de ce schéma est qu'il s'appuie sur l'homologue pour montrer qu'il est en vie. À cette fin, l'homologue B peut n'être jamais intéressé par la vie de l'homologue A. Néanmoins, si A est intéressé par la vie de B, B DOIT en être au courant, et conserver les informations d'état nécessaires pour envoyer des HELLO périodiques à A. L'inconvénient d'un tel schéma devient clair dans le scénario d'accès distant. Considérons un agrégateur VPN qui termine un grand nombre de sessions (de l'ordre de 50 000 homologues). Chaque homologue exige une reprise sur défaillance très rapide, donc exige que l'agrégateur envoie des paquets HELLO toutes les 10 secondes à peu près. Un tel schéma manque simplement d'adaptabilité, car l'agrégateur DOIT envoyer 50 000 messages toutes les quelques secondes.

Dans ces deux schémas (garder en vie et battement de cœur) des négociations d'intervalle de message DOIVENT se produire, afin que chaque entité puisse savoir à quel rythme son homologue attend un HELLO. Cela ajoute immédiatement un degré de complexité. De même, le besoin d'envoyer des messages périodiques (sans considération des autres activités IPsec/IKE) augmente aussi la redondance de calcul du système.

5. Protocole DPD

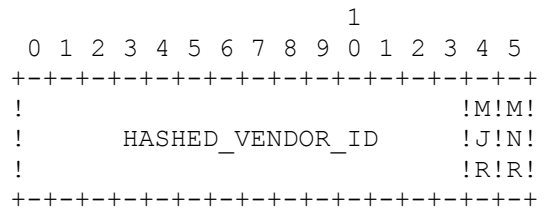
DPD traite les insuffisances des schémas garder en vie et battement de cœur de IKE en introduisant une logique plus raisonnable pour gouverner l'échange de messages. Essentiellement, garder en vie et battement de cœur rendent obligatoire l'échange de HELLO à des intervalles réguliers. À l'opposé, avec DPD, chaque état DPD d'homologue est largement indépendant de celui de l'autre. Un homologue est libre de demander une preuve de vie quand il en a besoin – et non à des intervalles obligatoires. Cette propriété asynchrone des échanges DPD permet que moins de messages soient envoyés, et c'est ainsi que DPD réalise une meilleure adaptabilité.

Par exemple, considérons deux homologues PDP A et B. Si il y a du trafic IPsec valide entre les deux, il n'y a pas besoin de preuve de vie. Le trafic IPsec lui-même sert de preuve de vie. Si, d'un autre côté, un certain temps s'écoule sans qu'aucun échange de paquet ne survienne, la vivacité de chaque homologue est questionnable. La connaissance de la vivacité de l'homologue n'est cependant nécessaire avec urgence que si il y a du trafic à envoyer. Par exemple, si l'homologue A a des paquets IPsec à envoyer après la période d'inactivité, il va avoir besoin de savoir si l'homologue B est toujours en vie. À ce point, l'homologue A peut initier l'échange DPD. À cette fin, chaque homologue PEUT avoir des exigences différentes pour détecter une preuve de vie. L'homologue A, par exemple, PEUT exiger une reprise sur défaillance rapide, tandis que les exigences de l'homologue B sur le nettoyage des ressources sont moins urgentes. Dans DPD, chaque homologue peut définir sa propre "métrique des soucis" - un intervalle qui définit l'urgence de l'échange DPD. Continuant l'exemple, l'homologue A peut définir son intervalle DPD comme étant de 10 secondes. Alors, si l'homologue A envoie du trafic IPsec sortant, mais échoue à recevoir du trafic entrant pendant 10 secondes, il peut initier un échange DPD. L'homologue B, de l'autre côté, définit ses intervalles DPD moins urgents comme étant de 5 minutes. Si la session IPsec est inactive pendant 5 minutes, l'homologue B peut initier un échange DPD la prochaine fois qu'il envoie des paquets IPsec à A.

Il est important de noter que la décision sur quand initier un échange DPD est spécifique de la mise en œuvre. Une mise en œuvre peut même définir les messages DPD comme étant à des intervalles réguliers à la suite des périodes d'inactivité. Voir au paragraphe 5.5 d'autres suggestions de mise en œuvre.

5.1 Identifiant de fabricant DPD

Pour montrer une capacité DPD, une entité DOIT envoyer l'identifiant de fabricant DPD. Les deux homologues d'une session IKE DOIVENT envoyer l'identifiant de fabricant DPD avant que les échanges DPD puissent commencer. Le format de l'identifiant de fabricant DPD est :



où HASHED_VENDOR_ID = {0xAF, 0xCA, 0xD7, 0x13, 0x68, 0xA1, 0xF1, 0xC9, 0x6B, 0x86, 0x96, 0xFC, 0x77, 0x57} et MJR et MNR correspondent à la version majeure et mineure courante de ce protocole (respectivement 1 et 0). Un homologue IKE DOIT envoyer l'identifiant de fabricant si il souhaite participer aux échanges PDP.

5.2 Échanges de messages

L'échange DPD est un message Notify bidirectionnel (HELLO/ACK). L'échange est défini comme :

Envoyeur		Répondeur
HDR*, NOTIFY(R-U-THERE), HASH	------>	
	<------	HDR*, NOTIFY(R-U-THERE-ACK), HASH

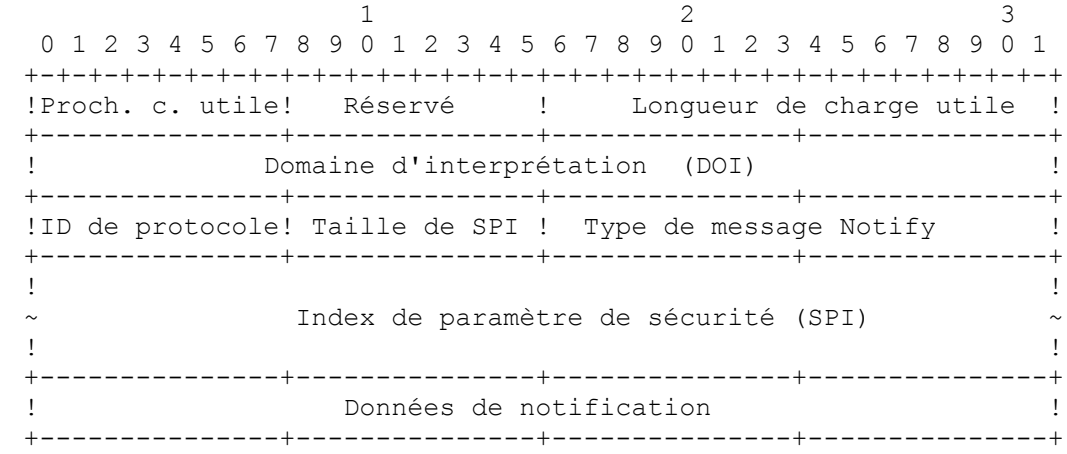
Le message R-U-THERE correspond à un "HELLO" et le R-U-THERE-ACK correspond à un "ACK". Les deux messages sont simplement des charges utiles ISAKMP Notify, et à ce titre, le présent document définit ces deux nouveaux types de message notify ISAKMP :

Notify	**Valeur de message**
R-U-THERE	36136
R-U-THERE-ACK	36137

Une entité qui a envoyé l'identifiant de fabricant DPD DOIT répondre à une interrogation R-U-THERE. De plus, une entité DOIT rejeter les messages R-U-THERE et R-U-THERE-ACK non chiffrés.

5.3 Format de message NOTIFY (R-U-THERE/R-U-THERE-ACK)

Lorsque il est envoyé, le message R-U-THERE DOIT prendre la forme suivante :



Comme ce message est un NOTIFY ISAKMP, les champs Prochaine charge utile, Réservé, et Longueur de charge utile DEVRAIENT être réglés en conséquence. Les champs restants sont réglés comme ceci :
- Domaine d'interprétation (4 octets) - DEVRAIT être réglé à IPSEC-DOI.
- Identifiant de protocole (1 octet) - DOIT être réglé à l'identifiant de protocole pour ISAKMP.
- Taille de SPI (1 octet) - DEVRAIT être réglé à seize (16), la longueur des mouchards ISAKMP.
- Type de message Notify (2 octets) - DOIT être réglé à R-U-THERE.
- Index de paramètre de sécurité (16 octets) - DEVRAIT être réglé aux mouchards d'initiateur et de répondeur de la SA IKE (dans cet ordre).
- Données de notification (4 octets) - DOIT être réglé au numéro de séquence correspondant au message.

Le format du message R-U-THERE-ACK est le même, à l'exception du type de message Notify qui DOIT être réglé à R-U-THERE-ACK. Là encore, les données de notification DOIVENT être envoyées avec le numéro de séquence correspondant au message R-U-THERE reçu.

5.4 Impulsion de l'échange DPD

Là encore, plutôt que s'appuyer sur un intervalle de temps négocié pour forcer l'échange de messages, DPD ne rend obligatoire à aucun moment l'échange des messages R-U-THERE. Un homologue IKE DEVRAIT plutôt n'envoyer une interrogation R-U-THERE à son homologue que si il s'intéresse à sa vivacité. À cette fin, si du trafic est régulièrement échangé entre deux homologues, l'un ou l'autre homologue DEVRAIT utiliser ce trafic comme preuve de vie, et les deux homologues NE DEVRAIENT PAS initier d'échange DPD.

Un homologue DOIT garder trace de l'état d'un échange DPD. C'est-à-dire que, une fois qu'il a envoyé une interrogation R-U-THERE, il attend un ACK en réponse dans un délai défini par la mise en œuvre. Une mise en œuvre DEVRAIT retransmettre les interrogations R-U-THERE lorsque elle ne reçoit pas de ACK. Après un certain nombre de messages retransmis, une mise en œuvre DEVRAIT supposer que son homologue est injoignable et supprimer les SA IPsec et IKE avec l'homologue.

5.5 Suggestion de mise en œuvre

Comme la vivacité d'un homologue ne peut être mise en question que lorsque aucun trafic n'est échangé, une mise en œuvre viable peut commencer par surveiller l'inactivité. Selon ce principe, la vivacité d'un homologue n'est importante que lorsque il y a du trafic sortant à envoyer. À cette fin, une mise en œuvre peut initier un échange DPD (c'est-à-dire, envoyer un message R-U-THERE) lorsque il y a eu un certain temps d'inactivité, suivi par le désir d'envoyer du trafic sortant. De même, une entité peut initier un échange DPD si elle a envoyé du trafic IPsec sortant, mais pas reçu de paquets IPsec entrants en réponse. Un échange PDP complet (c'est-à-dire, transmission de R-U-THERE et réception du R-U-THERE-ACK correspondant) va servir de preuve de vie jusqu'à la prochaine période inactive.

Là encore, comme DPD ne rend pas obligatoire d'intervalle, cette période "inactive" (ou "métrique de soucis") est laissée à la décision de la mise en œuvre. Ce n'est pas une valeur négociée.

5.6 Comparaisons

L'avantage en performances qu'offre DPD sur les schémas traditionnels de garder en vie et battement de cœur vient du fait que les messages réguliers n'ont pas besoin d'être envoyés. Retournant aux exemples présentés au paragraphe 4.1, une mise en œuvre de garder en vie comme celle présentée exigerait un temporisateur pour signaler quand envoyer un message HELLO et un autre temporisateur pour la limite d'attente de l'accusé de réception provenant de l'homologue (ce pourrait aussi être le temporisateur de retransmission). De même, un schéma de battement de cœur comme celui présenté au paragraphe 4.2 devrait avoir un temporisateur pour signaler quand envoyer un HELLO, ainsi qu'un autre pour signaler l'attente d'un HELLO provenant de l'homologue. À l'opposé, un schéma DPD doit conserver un horodatage pour garder trace du dernier trafic reçu de l'homologue (marquant ainsi le début de la "période d'inactivité"). Une fois qu'un message DPD R-U-THERE a été envoyé, une mise en œuvre a seulement besoin de tenir un temporisateur pour signaler la retransmission. Donc, le besoin de tenir l'état d'un temporisateur actif est réduit, résultant en une amélioration de l'adaptabilité (en supposant que la tenue d'un horodatage est moins coûteuse que celle d'un temporisateur actif). De plus, comme un échange DPD ne se produit que si une entité n'a pas récemment reçu de trafic de son homologue, le nombre de messages IKE à envoyer et traiter est aussi réduit. Par conséquent, l'adaptabilité de DPD est bien meilleure que dans garder en vie et battement de cœur.

DPD conserve le modèle de HELLO/ACK présenté par garder en vie, car il s'ensuit qu'un échange n'est initié que par une entité intéressée par la vivacité de son homologue.

6. Résistance aux attaques en répétition et fausses preuves de vie

6.1 Numéro de séquence dans les messages DPD

Pour se garder contre l'attaque de répétition de message et de fausse preuve de vie, un numéro de séquence de 32 bits DOIT être présenté avec chaque message R-U-THERE. Une réponse à un message R-U-THERE DOIT envoyer un R-U-THERE-ACK avec le même numéro de séquence. À réception du message R-U-THERE-ACK, l'envoyeur initial DEVRAIT vérifier la validité du numéro de séquence. L'envoyeur initial DEVRAIT rejeter le R-U-THERE-ACK si le numéro de séquence ne correspond pas à celui envoyé avec le message R-U-THERE.

De plus, le receveur du message R-U-THERE et celui du message R-U-THERE-ACK DEVRAIENT tous deux vérifier la validité des mouchards Initiator et Responder présentés dans le champ SPI de la charge utile.

6.2 Choix et maintenance des numéros de séquence

Comme les deux homologues DPD peuvent initier un échange DPD (c'est-à-dire que les deux homologues peuvent envoyer des messages R-U-THERE) chaque homologue DOIT tenir son propre numéro de séquence pour les messages R-U-THERE. Le premier message R-U-THERE envoyé dans une session DOIT être un nombre choisi au hasard. Pour empêcher de déborder de la frontière des 32 bits, le bit de poids fort du numéro de séquence initial DEVRAIT être réglé à zéro. Les messages R-U-THERE suivants DOIVENT incrémenter de un le numéro de séquence. Les numéros de séquence PEUVENT revenir à initialisation à l'expiration de la SA IKE, en passant à un nouveau nombre choisi au hasard. Chaque entité DEVRAIT aussi conserver le numéro de séquence de R-U-THERE de son homologue, et une entité DEVRAIT rejeter le message R-U-THERE si il ne correspond pas au numéro de séquence attendu.

Les mises en œuvre PEUVENT tenir une fenêtre des numéros de séquence acceptables, mais la présente spécification ne fait aucune hypothèse sur la façon dont ceci peut être fait. Là encore, ce détail est spécifique de la mise en œuvre.

7. Considérations pour la sécurité

Comme le souligne la section précédente, DPD utilise des numéros de séquence pour s'assurer de la vivacité. Cette section décrit les avantages de l'utilisation de numéros de séquence sur celle de noms occasionnels aléatoires pour s'assurer de la vivacité.

Bien que les numéros de séquence exigent que les entités conservent l'état par homologue, ils donnent une méthode supplémentaire de protection dans certaines attaques en répétition. Considérons un cas où un homologue A envoie à l'homologue B un message DPD R-U-THERE valide. Un attaquant C peut intercepter ce message et inonder B avec de multiples copies des messages. B devra déchiffrer et traiter chaque paquet (sans considérer si des numéros de séquence ou des noms occasionnels sont utilisés). Avec les numéros de séquence, B peut détecter que les paquets sont répétés : les numéros de séquence dans ces paquets répétés ne vont normalement pas correspondre au numéro de séquence incrémenté que B s'attend à recevoir de A. Cela empêche B de devoir construire, chiffrer, et envoyer des ACK. À l'opposé, si le protocole DPD utilisait des noms occasionnels, il ne donnerait aucun moyen à B de détecter que les messages sont répétés (sauf si B tient une liste des noms occasionnels reçus récemment).

Un autre avantage des numéros de séquence est qu'ils ajoutent une assurance supplémentaire de la vie de l'homologue. Tant qu'un receveur vérifie la validité d'un message DPD R-U-THERE (en vérifiant son numéro de séquence incrémenté) le receveur peut alors être assuré de la vie de l'homologue par le simple fait que l'envoyeur a initié l'interrogation. Les noms occasionnels, quant à eux, ne peuvent pas donner cette assurance.

8. Considérations relatives à l'IANA

Aucune action n'est demandée à l'IANA au titre du présent document. DPD utilise des numéros de notification provenant de la gamme privée.

9. Références

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

[RFC2401] S. Kent et R. Atkinson, "Architecture de sécurité pour le protocole Internet", novembre 1998. (*Obsolète, voir* RFC4301)

[RFC2409] D. Harkins et D. Carrel, "L'échange de clés Internet (IKE)", novembre 1998. *(Obsolète, voir la* RFC4306)

10. Adresse des éditeurs

Geoffrey Huang
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134
téléphone : (408) 525-5354
mél : ghuang@cisco.com

Stephane Beaulieu
Cisco Systems, Inc.
2000 Innovation Drive
Kanata, ON
Canada, K2K 3E8
téléphone : (613) 254-3678
mél : stephane@cisco.com

Dany Rochefort
Cisco Systems, Inc.
124 Grove Street, Suite 205
Franklin, MA 02038
téléphone : (508) 553-8644
mél : danyr@cisco.com

11. Déclaration complète de droits de reproduction

Copyright (C) The Internet Society (2004).

Le présent document est soumis aux droits, licences et restrictions contenus dans le BCP 78, et à www.rfc-editor.org, et sauf pour ce qui est mentionné ci-après, les auteurs conservent tous leurs droits.

Le présent document et les informations contenues sont fournis sur une base "EN L'ÉTAT" et le contributeur, l'organisation qu'il ou elle représente ou qui le/la finance (s'il en est), la INTERNET SOCIETY et la INTERNET ENGINEERING TASK

FORCE déclinent toutes garanties, exprimées ou implicites, y compris mais non limitées à toute garantie que l'utilisation des informations ci encloses ne violent aucun droit ou aucune garantie implicite de commercialisation ou d'aptitude à un objet particulier.

Propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourrait être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible ; pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur répertoire en ligne des IPR de l'IETF à http://www.ietf.org/ipr .

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf-ipr@ietf.org .

Remerciement

Le financement de la fonction d'édition des RFC est actuellement fourni par la Internet Society.