Équipe d'ingénierie de l'Internet (IETF)
Request for Comments : 6582
RFC rendue obsolète : 3782
Catégorie : En cours de normalisation
ISSN: 2070-1721
Traduction Claude Brière de L'Isle

T. Henderson, Boeing
S. Floyd, ICSI
A. Gurtov, University of Oulu
Y. Nishida, WIDE Project
avril 2012

Modification NewReno à l'algorithme de récupération rapide de TCP

Résumé
La [RFC5681] documente les quatre algorithmes imbriqués de contrôle d'encombrement de TCP : démarrage lent, évitement d'encombrement, retransmission rapide, et récupération rapide. La RFC 5681 permet explicitement certaines modifications de ces algorithmes, incluant des modifications qui utilisent l'option TCP d'accusé de réception sélectif (SACK) [RFC2883], et des modifications qui répondent aux "accusés de réception partiels" (ACK qui couvrent les nouvelles données, mais pas toutes les données en cours lorsque la perte a été détectée) en l'absence de SACK. Le présent document décrit un algorithme spécifique pour répondre aux accusés de réception partiels, appelés "NewReno". Cette réponse aux accusés de réception partiels a d'abord été proposée par Janey Hoe. Le présent document rend obsolète la RFC 3782.

1. Introduction

Pour la mise en œuvre normale de l'algorithme TCP de récupération rapide décrit dans la [RFC5681] (mis en œuvre pour la première fois en 1990 dans la version BSD Reno, et appelé depuis "algorithme Reno" dans [FF96]), l'envoyeur des données TCP ne retransmet un paquet qu'après l'expiration d'une temporisation de retransmission, ou après que trois accusés de réception dupliqués sont arrivés, déclenchant l'algorithme de retransmission rapide. Une seule temporisation de retransmission pourrait résulter en la retransmission de plusieurs paquets de données, mais chaque invocation de l'algorithme de retransmission rapide dans la RFC5681 conduit à la retransmission d'un seul paquet de données.

Deux problèmes se posent avec Reno TCP lorsque plusieurs pertes de paquets surviennent dans une seule fenêtre. D'abord,

Reno va souvent prendre une temporisation, comme cela a été documenté dans [Hoe95]. Ensuite, même si une temporisation de retransmission est évitée, plusieurs retransmissions rapides et réductions de fenêtre peuvent survenir, comme documenté dans [F94]. Lorsque surviennent plusieurs pertes de paquet, si l'option SACK [RFC2883] est disponible, l'envoyeur TCP a les informations pour prendre des décisions intelligentes sur les paquets à retransmettre et les paquets a ne pas retransmettre durant la récupération rapide.

Le présent document s'applique aux connexions TCP qui ne sont pas capables d'utiliser l'option TCP d'accusé de réception sélectif (SACK, *Selective Acknowledgment*) soit parce que l'option n'est pas prise en charge localement, soit parce que l'homologue TCP n'a pas indiqué sa volonté d'utiliser SACK.

En l'absence de SACK, il y a peu d'informations disponibles à l'envoyeur TCP pour prendre des décisions de retransmission durant la récupération rapide. Des trois accusés de réception dupliqués, l'envoyeur déduit une perte de paquet et retransmet le paquet indiqué. Après cela, l'envoyeur des données pourrait recevoir des accusés de réception dupliqués supplémentaires, car le receveur des données accuse réception de paquets de données supplémentaires qui étaient déjà en cours lorsque l'envoyeur est passé en retransmission rapide.

Dans le cas de plusieurs paquets abandonnés à partir d'une seule fenêtre de données, la première nouvelle information disponible à l'envoyeur vient lorsque il reçoit un accusé de réception pour le paquet retransmis (c'est-à-dire, le paquet retransmis lorsque la retransmission rapide a débuté). Si il y a un seul abandon de paquet et pas de réarrangement, l'accusé de réception pour ce paquet va alors accuser réception de tous les paquets transmis avant le début de la retransmission rapide. Cependant, si il y a plusieurs abandons de paquet, l'accusé de réception pour le paquet retransmis va acquitter certains, mais pas tous, des paquets transmis avant la retransmission rapide. On appelle cet accusé de réception un accusé de réception partiel.

Parmi plusieurs autres suggestions, [Hoe95] proposait que durant la récupération rapide, l'envoyeur des données TCP réponde à l'accusé de réception partiel en déduisant que le prochain paquet en séquence a été perdu et qu'il retransmette ce paquet. Le présent document décrit une modification à l'algorithme de retransmission rapide de la RFC5681 qui incorpore une réponse aux accusés de réception partiels reçus durant la récupération rapide. On appelle cela l'algorithme de récupération rapide modifié "NewReno", parce qu'il y a une variation légère mais significative du comportement qui a été historiquement désigné comme celui de Reno. Le présent document ne discute pas les autres suggestions de [Hoe95] et [Hoe96], comme de changer le paramètre ssthresh durant le démarrage lent, ou la proposition d'envoyer un nouveau paquet tous les deux accusés de réception dupliqués durant la récupération rapide. La version de NewReno de ce document ne rentre pas dans les autres discussions de NewReno dans la littérature [LM97], [Hen98].

On ne prétend pas que la version NewReno de la récupération rapide décrite ici soit une modification optimale de la récupération rapide pour répondre aux accusés de réception partiels, pour les connexions TCP qui ne sont pas capables d'utiliser SACK. Sur la base de notre expériences de la modification NewReno dans le simulateur de réseau connu sous le nom de ns-2 [NS] et avec de nombreuses mises en œuvre de NewReno, nous pensons que cette modification améliore les performances des algorithmes de retransmission et de récupération rapide dans une grande variété de scénarios. Les versions précédentes de cette RFC [RFC2582], [RFC3782], donnent des preuves fondées sur des simulations des gains possibles de performances.

2. Terminologie et définitions

Le présent document suppose que le lecteur est familier des termes taille maximum de segment d'envoyeur (SMSS, *Sender Maximum Segment Size*), fenêtre d'encombrement (cwnd, *congestion window*), et taille en cours (*FlightSize*) définis dans la [RFC5681].

Le présent document définit un variable supplémentaire d'état côté envoyeur appelée "Récupérer":

Récupérer : En récupération rapide, cette variable enregistre le numéro de séquence d'envoi qui doit être acquitté avant que la procédure de récupération rapide soit déclarée terminée.

3. Algorithmes de retransmission rapide et de récupération rapide dans NewReno

3.1. Vue d'ensemble du protocole

L'idée de base des extensions aux algorithmes de retransmission rapide et de récupération rapide décrits au paragraphe 3.2 de la [RFC5681] est la suivante. L'envoyeur TCP peut déduire, de l'arrivée d'accusés de réception dupliqués, si plusieurs

pertes sont très vraisemblablement survenues sur la même fenêtre de données, et éviter de subir une temporisation de retransmission ou de faire plusieurs réductions de fenêtre d'encombrement du fait d'un tel événement.

La modification NewReno s'applique à la procédure de récupération rapide qui commence lorsque trois ACK dupliqués sont reçus et se termine quand survient une fin de temporisation de retransmission ou qu'un ACK arrive pour acquitter toutes les données jusque et y compris les données qui étaient en cours lorsque à commencé la procédure de récupération rapide.

3.2. Spécification

Les procédures spécifiées au paragraphe 3.2 de la [RFC5681] sont suivies, avec les modifications énumérées ci-dessous. Noter que la présente spécification évite l'utilisation des mots clés définis dans la [RFC2119], car elle donne principalement des lignes directrices pour la mise en œuvre côté envoyeur pour l'amélioration des performances, et n'affecte pas l'interopérabilité.

1) Initialisation du bloc de contrôle de protocole TCP :
   Lorsque le bloc de contrôle de protocole TCP est initialisé, Récupérer est réglé au numéro de séquence d'envoi initial.

2) Trois ACK dupliqués :
   Quand le troisième ACK dupliqué est reçu, l'envoyeur TCP vérifie d'abord la valeur de Récupérer pour voir si le champ Accusé de réception cumulatif couvre plus que Récupérer. Si il en est ainsi, la valeur de Récupérer est incrémentée à la valeur du plus fort numéro de séquence transmis jusqu'alors par TCP. TCP entre alors dans la retransmission rapide (étape 2 du paragraphe 3.2 de la [RFC5681]). Sinon, TCP n'entre pas en retransmission rapide et ne rétablit pas ssthresh.

3) Réponse aux nouvelles données acquittées :
   L'étape 6 de la [RFC5681] spécifie la réponse au prochain ACK qui accuse réception des données non acquittées jusqu'alors. Quand un ACK arrive qui accuse réception de nouvelles données, cet ACK pourrait être l'accusé de réception choisi par la retransmission initiale de retransmission rapide, ou choisi par une retransmission ultérieure. Il y a deux cas :
   Accusés de réception pleins :
   Si cet ACK accuse réception de toutes les données jusque et y compris Récupérer, l'ACK accuse alors réception de tous les segments intermédiaires envoyés entre la transmission originale du segment perdu et la réception du troisième ACK dupliqué. Régler cwnd soit (1) à min(ssthresh, max(Taille en cours, SMSS) + SMSS) soit (2) à ssthresh, où ssthresh est la valeur établie lorsque la retransmission rapide a commencé, et où Taille en cours dans (1) est la quantité de données présentement en instance. C'est ce qu'on appelle "dégonfler" la fenêtre. Si on choisit la seconde option, il est recommandé que la mise en œuvre prenne des mesures pour éviter une éventuelle salve de données, au cas où la quantité de données en instance dans le réseau serait très inférieure à ce que permet la nouvelle fenêtre d'encombrement. Un mécanisme simple est de limiter le nombre de paquets de données qui peuvent être envoyés en réponse à un seul accusé de réception. Sortir de la procédure de récupération rapide.

   Accusés de réception partiels :
   Si cet ACK N'accuse PAS réception de toutes les données jusque et y compris Récupérer, c'est alors un ACK partiel. Dans ce cas, retransmettre le premier segment non acquitté. Dégonfler la fenêtre d'encombrement de la quantité de nouvelles données acquittées du champ Accusé de réception cumulatif. Si l'ACK partiel acquitte au moins un SMSS de nouvelles données, rajouter alors SMSS octets à la fenêtre d'encombrement. Cela gonfle artificiellement la fenêtre d'encombrement afin de refléter le segment supplémentaire qui a quitté le réseau. Envoyer un nouveau segment si c'est permis par la nouvelle valeur de cwnd. Ce "dégonflage partiel de fenêtre" tente d'assurer que, lorsque se termine finalement la récupération rapide, une quantité de données d'approximativement ssthresh seront en instance dans le réseau. Ne pas sortir de la procédure de récupération rapide (c'est-à-dire que si des ACK dupliqués arrivent ensuite, exécuter l'étape 4 du paragraphe 3.2 de la [RFC5681]).

   Pour le premier ACK partiel qui arrive durant la récupération rapide, remettre aussi à zéro le temporisateur de retransmission. La gestion des temporisateurs est exposée plus en détails à la Section 4.

4) Temporisations de retransmission :
   Après une temporisation de retransmission, enregistrer le plus grand numéro de séquence transmis dans la variable Récupérer, et sortir le cas échéant de la procédure de récupération rapide.

L'étape 2 ci-dessus spécifie une vérification que le champ Accusé de réception cumulatif couvre plus que Récupérer. Comme le champ Accusé de réception contient le numéro de séquence que l'envoyeur s'attend à recevoir ensuite, le

"numéro_d'ack" de l'accusé de réception couvre plus que Récupérer lorsque numéro_d'ack - 1 > Récupérer ; c'est-à-dire, au moins un octet de données de plus est acquitté au delà de l'octet le plus élevé en instance lors du début de la dernière retransmission rapide.

Noter que dans l'étape 3 ci-dessus, la fenêtre d'encombrement est dégonflées après la réception d'un accusé de réception partiel. La fenêtre d'encombrement avait vraisemblablement été considérablement gonflée lorsque l'accusé de réception partiel avait été reçu. De plus, en fonction du schéma d'origine de pertes de paquet, l'accusé de réception partiel peut accuser réception de presque une fenêtre de données. Dans ce cas, si la fenêtre d'encombrement n'était pas dégonflée, l'envoyeur des données pourrait être capable d'envoyer presque une fenêtre de données à la suite.

Le présent document ne spécifie pas la réponse de l'envoyeur aux ACK dupliqués lorsque l'algorithme de retransmission/récupération rapide n'est pas invoqué. Ceci est traité dans d'autres documents, tels que ceux qui décrivent la procédure de transmission limitée [RFC3042]. Le présent document ne traite pas non plus les questions d'ajustement du seuil d'accusés de réception dupliqués, mais suppose le seuil spécifié dans les normes de l'IETF ; la norme actuelle est la [RFC5681], qui spécifie un seuil de trois accusés de réception dupliqués.

En note finale, on observera qu'en l'absence de l'option SACK, l'envoyeur des données travaille à partir d'informations limitées. Lorsque la question de la récupération de plusieurs paquets abandonnés à partir d'une seule fenêtre de données est d'une particulière importance, la meilleure solution de remplacement serait d'utiliser l'option SACK.

4. Traitement des accusés de réception dupliqués après une fin de temporisation

Après chaque temporisation de retransmission, le plus fort numéro de séquence transmis jusqu'alors est enregistré dans la variable Récupérer. Si, après une temporisation de retransmission, l'envoyeur de données TCP retransmet trois paquets consécutifs qui ont déjà été reçus par le receveur des données, l'envoyeur des données TCP va alors recevoir trois accusés de réception dupliqués qui ne couvrent pas plus que Récupérer. Dans ce cas, les accusés de réception dupliqués ne sont pas une indication d'une nouvelle instance d'encombrement. Ils sont simplement l'indication que l'envoyeur a retransmis inutilement au moins trois paquets.

Cependant, lorsque un paquet retransmis est lui-même éliminé, l'envoyeur peut aussi recevoir trois accusés de réception dupliqués qui ne couvrent pas plus que Récupérer. Dans ce cas, l'envoyeur aurait mieux fait d'initier la retransmission rapide. Pour un envoyeur TCP qui met en œuvre l'algorithme spécifié au paragraphe 3.2 du présent document, l'envoyeur ne conclut pas à un abandon de paquet à partir de trois accusés de réception dupliqués dans ce scénario. Comme toujours, le temporisateur de retransmission est le mécanisme de repli pour conclure à une perte de paquet dans ce cas.

Il y a plusieurs heuristiques, fondées sur les horodatages ou sur la quantité d'avancement du champ Accusé de réception cumulatif, qui permettent à l'envoyeur de distinguer, dans certains cas, entre les trois accusés de réception dupliqués qui suivent un paquet retransmis qui a été abandonné, et les trois accusés de réception dupliqués d'une retransmission inutile [Gur03], [GF04]. L'envoyeur TCP peut utiliser une telle heuristique pour décider d'invoquer une retransmission rapide dans certains cas, même lorsque les trois accusés de réception dupliqués ne couvrent pas plus que Récupérer.

Par exemple, lorsque trois accusés de réception dupliqués sont causés par la retransmission inutile de trois paquets, ceci sera vraisemblablement accompagné par l'avance du champ Accusé de réception cumulatif d'au moins quatre segments. De même, une heuristique fondée sur les horodatages utilise le fait que lorsque il y a un trou dans l'espace des numéros de séquence, l'horodatage dont il est fait écho dans l'accusé de réception dupliqué est l'horodatage du plus récent paquet de données qui a avancé le champ Accusé de réception cumulatif [RFC1323]. Si les horodatages sont utilisés, et si l'envoyeur mémorise l'horodatage du dernier segment acquitté, alors l'horodatage dont il est fait écho dans les accusés de réception dupliqués peuvent être utilisés pour distinguer entre un paquet retransmis qui a été abandonné et trois accusés de réception dupliqués provenant de la retransmission inutile de trois paquets.

4.1 Heuristique de l'accusé de réception

Si on utilise l'heuristique fondée sur les ACK; en suivant alors l'avancement du champ Accusé de réception cumulatif ; l'envoyeur mémorise la valeur du précédent accusé de réception cumulatif comme prev_highest_ack, et mémorise le dernier ACK cumulatif comme highest_ack. De plus, la vérification suivante est effectuée si, à l'étape 2 du paragraphe 3.2, le champ Accusé de réception cumulatif ne couvre pas plus que Récupérer.

2*) Si le champ Accusé de réception cumulatif ne couvrait pas plus que Récupérer, vérifier pour voir si la fenêtre d'encombrement est supérieure à SMSS octets et si la différence entre highest_ack et prev_highest_ack est au plus de 4*SMSS octets. Si c'est vrai, les ACK dupliqués indiquent un segment perdu (entrer en retransmission rapide).

Autrement, les ACK dupliqués résultent vraisemblablement de retransmissions inutiles (ne pas entrer en retransmission rapide).

La vérification de la fenêtre d'encombrement sert à protéger contre la retransmission rapide immédiatement après une temporisation de retransmission.

Si plusieurs ACK sont perdus, l'envoyeur peut voir un bond des ACK cumulatifs de plus de trois segments, et l'heuristique peut échouer. La [RFC5681] recommande qu'un receveur envoie des ACK dupliqués pour chaque paquet de données déclassé, tel qu'un paquet de données reçu durant la récupération rapide. L'heuristique de ACK va plus probablement échouer si le receveur ne suit pas cet avis, parce qu'alors un plus petit nombre de pertes de ACK sont nécessaires pour produire un bond suffisant dans les ACK cumulatifs.

4.2 Heuristique de l'horodatage

Si cette heuristique est utilisée, l'envoyeur mémorise l'horodatage du dernier segment acquitté. De plus, la dernière phrase de l'étape 2 du paragraphe 3.2 de ce document est remplacée comme suit :

2**) Si le champ Accusé de réception cumulatif ne couvrait pas plus que Récupérer, vérifier pour voir si l'horodatage dont il est fait écho dans le dernier accusé de réception non dupliqué est égal à l'horodatage mémorisé. Si c'est vrai, les ACK dupliqués indiquent un segment perdu (entrer en retransmission rapide). Autrement, les ACK dupliqués résultent vraisemblablement de retransmissions inutiles (ne pas entrer en retransmission rapide).

L'heuristique d'horodatage fonctionne correctement, à la fois lorsque le receveur fait écho aux horodatages, comme spécifié par la [RFC1323], et par ses tentatives de révision. Cependant, si le receveur fait écho de façon arbitraire aux horodatages, l'heuristique peut échouer. L'heuristique peut aussi échouer si il y a une temporisation parasite et si le retour des ACK ne provient pas de segments retransmis. On peut empêcher cela par des algorithmes de détection tels que l'algorithme de détection Eifel [RFC3522].

5. Questions de mise en œuvre pour le receveur des données

La [RFC5681] spécifie que les "segments de données déclassés DEVRAIENT être acquittés immédiatement, afin d'accélérer la récupération de perte". Neal Cardwell a noté que certains receveurs de données n'envoient pas un accusé de réception immédiat lorsque ils envoient un accusé de réception partiel, mais attendent plutôt que leur temporisateur d'accusé de réception retardé arrive d'abord à expiration [C98]. Comme le note [C98], cela limite sévèrement le bénéfice potentiel de NewReno en retardant la réception de l'accusé de réception partiel chez l'envoyeur des données. En écho à la [RFC5681], notre recommandation est que le receveur des données envoie un accusé de réception immédiat pour un segment déclassé, même lorsque le segment déclassé remplit un trou dans la mémoire tampon.

6. Questions de mise en œuvre pour l'envoyeur des données

À l'étape 3 du paragraphe 3.2, ci-dessus, on a noté que les mises en œuvre devraient prendre des mesures pour éviter une possible salve de données en quittant la récupération rapide, au cas où la quantité de nouvelles données que l'envoyeur est en droit d'envoyer parce que la nouvelle valeur de la fenêtre d'encombrement est importante. Cela peut survenir pendant NewReno lorsque des ACK sont perdus ou traités comme de pures mises à jour de fenêtre, causant de ce fait la sous estimation par l'envoyeur du nombre de nouveaux segments qui peuvent être envoyés durant la procédure de récupération. Précisément, les salves peuvent survenir lorsque Taille en cours est très inférieur à la nouvelle fenêtre d'encombrement au moment de la sortie de la récupération rapide. Un mécanisme simple pour éviter une salve de données en quittant la récupération rapide est de limiter le nombre de paquets de données qui peuvent être envoyés en réponse à un seul accusé de réception. (Ceci est appelé "maxburst_" dans ns-2 [NS].) Les autres mécanismes possibles pour éviter les salves incluent le ralentissement fondé sur le taux, ou de régler le seuil de démarrage lent à la fenêtre d'encombrement résultante et ensuite de remettre la fenêtre d'encombrement à Taille en cours. Une recommandation sur le mécanisme général d'évitement des schémas d'envoi excessivement saccadés sort du domaine d'application du présent document.

Une mise en œuvre peut vouloir utiliser un fanion distinct pour noter si elle est ou non présentement dans la procédure de récupération rapide. L'utilisation à cette fin de la valeur du compteur d'accusés de réception dupliqués n'est pas fiable, parce qu'elle peut être remise à zéro lors d'une mise à jour de fenêtre et des accusés de réception décalés.

Lors de la mise à jour du champ Accusé de réception cumulatif en dehors d'une récupération rapide, la variable d'état

recover peut devoir aussi être mise à jour afin de continuer de permettre une possible entrée en récupération rapide (étape 2 du paragraphe 3.2). Cette question se pose lorsque une mise à jour du champ Accusé de réception cumulatif résulte en un retour à zéro de numéro de séquence qui affecte l'ordre entre le champ Accusé de réception cumulatif et la variable d'état recover. L'entrée en récupération rapide n'est possible que lorsque le champ Accusé de réception cumulatif couvre plus que la variable d'état recover.

Il est important que l'envoyeur réponde correctement aux ACK dupliqués reçus lorsque l'envoyeur n'est plus en récupération rapide (par exemple, à cause d'une fin de temporisation de retransmission). La procédure de la transmission limitée de la [RFC3042] décrit les réponses possibles aux premiers et seconds accusés de réception dupliqués. Lorsque sont reçus trois accusés de réception dupliqués, ou plus, le champ Accusé de réception cumulatif ne couvre pas plus que recover, et une nouvelle récupération rapide n'est pas invoquée, l'envoyeur devrait suivre les indications de la Section 4. Autrement, l'envoyeur pourrait se retrouver pris dans une chaîne de temporisations parasites. On ne mentionne cela que parce que plusieurs mises en œuvre NewReno ont eu cette bogue, y compris celle de ns-2 [NS].

Il a été observé que certaines mises en œuvre de TCP lancent l'algorithme de démarrage lent ou de mise à jour de fenêtre d'évitement d'encombrement immédiatement après que la cwnd est réglée avec l'équation qui se trouve à l'étape 3 du paragraphe 3.2, même sans qu'un nouvel événement externe génère le changement de cwnd. Noter qu'après que la cwnd est réglée sur la base de la procédure de sortie de récupération rapide (étape 3 du paragraphe 3.2) cwnd ne devrait pas être mise à jour avant que survienne un nouvel événement (par exemple, l'arrivée d'un ACK, ou une fin de temporisation) après cet ajustement.

7. Considérations pour la sécurité

La [RFC5681] expose les considérations générales sur la sécurité concernant le contrôle d'encombrement de TCP. Le présent document décrit un algorithme spécifique qui se conforme aux exigences du contrôle d'encombrement de la [RFC5681], et donc ses considérations s'appliquent aussi à cet algorithme. Il n'y a pas de problème de sécurité supplémentaire connu pour cet algorithme spécifique.

8. Conclusions

Le présent document spécifie les algorithmes NewReno de retransmission et de récupération rapide pour TCP. Cette modification NewReno à TCP peut même être importante pour les mises en œuvre TCP qui prennent en charge l'option SACK, parce que celle-ci ne peut être utilisée pour les connexions TCP que lorsque les deux nœuds d'extrémité TCP l'acceptent. NewReno fonctionne mieux que Reno dans un certain nombre de scénarios discutés dans les précédentes versions de cette RFC ([RFC2582] [RFC3782]).

Un certain nombre d'options pour les algorithmes de base présentés à la Section 3 sont aussi référencées dans l'Appendice A de ce document. Cela inclut le traitement du temporisateur de retransmission, la réponse aux accusés de réception partiels, et si l'envoyeur doit ou non conserver une variable d'état appelée recover. On pense que les différences entre ces variantes de NewReno sont petites comparées aux différences entre Reno et NewReno. C'est-à-dire que la chose importante est de mettre en œuvre NewReno plutôt que Reno pour une connexion TCP sans SACK ; il est moins important de savoir exactement quelle variante de NewReno est mise en œuvre.

9. Remerciements

Tous nos remerciements à Anil Agarwal, Mark Allman, Armando Caro, Jeffrey Hsu, Vern Paxson, Kacheong Poon, Keyur Shah, et Bernie Volz pour leurs réactions détaillées sur les RFC précurseurs 2582 et 3782. Jeffrey Hsu a fourni des précisions sur le traitement de la variable recover ; ces précisions ont été appliquées à la RFC3782 via un erratum et sont incorporées dans le texte de la Section 6 du présent document. Yoshifumi Nishida a contribué à une modification de l'algorithme de récupération rapide à prendre en compte dans le cas où Taille en cours est 0 lorsque l'envoyeur TCP quitte la récupération rapide et que le receveur TCP utilise les accusés de réception retardés. Alexander Zimmermann a fait plusieurs suggestions pour améliorer la clarté du document.

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.

[RFC5681] M. Allman, V. Paxson, E. Blanton, "Contrôle d'encombrement de TCP", septembre 2009. *(Remplace la* RFC2581*) (D. S.)*

10.2 Références pour information

[C98] Cardwell, N., "delayed ACKs for retransmitted packets: ouch!" novembre 1998, mél à la liste de diffusion tcpimpl, archivée à < http://groups.yahoo.com/group/tcp-impl/message/1428 >.

[F94] Floyd, S., "TCP and Successive Fast Retransmits", Rapport technique, mai 1995. < ftp://ftp.ee.lbl.gov/papers/fastretrans.ps >.

[FF96] Fall, K. and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno and SACK TCP", Computer Communication Review, juillet 1996. < ftp://ftp.ee.lbl.gov/papers/sacks.ps.Z >.

[GF04] Gurtov, A. and S. Floyd, "Resolving Acknowledgment Ambiguity in non-SACK TCP", NExt Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN'04), février 2004. < http://www.cs.helsinki.fi/u/gurtov/papers/heuristics.html >.

[Gur03] Gurtov, A., "[Tsvwg] resolving the problem of unnecessary fast retransmits in go-back-N", mél à la liste de diffusion tsvwg, 28 juillet 2003. < http://www.ietf.org/mail-archive/web/tsvwg/current/msg04334.html >.

[Hen98] Henderson, T., "Re: NewReno and the 2001 Revision", septembre 1998. mél à la liste de diffusion tcpimpl, archivé à < http://groups.yahoo.com/group/tcp-impl/message/1321 >.

[Hoe95] Hoe, J., "Startup Dynamics of TCP's Congestion Control and Avoidance Schemes", Thèse de maîtrise, MIT, juin 1995.

[Hoe96] Hoe, J., "Improving the Start-up Behavior of a Congestion Control Scheme for TCP", ACM SIGCOMM, août 1996. < http://ccr.sigcomm.org/archive/1996/conf/hoe.pdf >.

[LM97] Lin, D. and R. Morris, "Dynamics of Random Early Detection", SIGCOMM 97, octobre 1997.

[NS] "The Network Simulator version 2 (ns-2)", < http://www.isi.edu/nsnam/ns/ >.

[RFC1323] V. Jacobson, R. Braden et D. Borman, "Extensions TCP pour de bonnes performances", mai 1992.

[RFC2582] S. Floyd, T. Henderson, "Modification NewReno à l'algorithme de récupération rapide de TCP", avril 1999. *(Expérimentale)* (*Obsolète, voir* RFC3782)

[RFC2883] S. Floyd et autres, "Extension à l'option d'accusé de réception sélectif (SACK) pour TCP", juillet 2000. *(P.S.)*

[RFC3042] M. Allman, H. Balakrishnan, S. Floyd, "Amélioration de la récupération de perte dans TCP avec la transmission limitée", janvier 2001. *(P.S.)*

[RFC3522] R. Ludwig, M. Meyer, "Algorithme Eifel de détection pour TCP", avril 2003. *(Expérimentale)*

[RFC3782] S. Floyd, T. Henderson, A. Gurtov, "Modification NewReno à l'algorithme de récupération rapide de TCP", avril 2004. *(P.S.)*

Appendice A Informations supplémentaires

Les précédentes versions de cette RFC ([RFC2582], [RFC3782]) contenaient des informations sur les sujets suivants, qui peuvent être consultées par les lecteurs qui voudraient plus d'informations sur les possibles variantes des algorithmes et qui

peuvent vouloir des références aux simulations spécifiques [NS] qui ont fourni les cas d'essai de NewReno.

La Section 4 de la [RFC3782] expose des comportements de remplacement pour remettre à zéro le temporisateur de retransmission après un accusé de réception partiel.

La Section 5 de la [RFC3782] expose des comportements de remplacement pour effectuer une retransmission après un accusé de réception partiel.

La Section 6 de la [RFC3782] décrit plus d'informations sur la motivation de la variable d'état recover de l'envoyeur.

La Section 9 de la [RFC3782] introduit des suites d'essai de simulation NS pour NewReno. De plus, on trouvera des références aux résultats des simulations dans toute la [RFC3782].

La Section 10 de la [RFC3782] donne une comparaison de TCP Reno et NewReno.

La Section 11 de la [RFC3782] fait la liste des changements par rapport à la [RFC2582].

Appendice B Changements par rapport à la RFC 3782

Dans la [RFC3782], la cwnd après réception d'un ACK plein sera réglée à (1) min (ssthresh, Taille en cours + SMSS) ou (2) à ssthresh. Cependant, la première option comporte un risque de dégradation des performances : avec la première option, si Taille en cours est zéro, le résultat sera 1 SMSS. Cela signifie que TCP ne peut transmettre qu'un seul segment à ce moment, ce qui peut causer un retard de la transmission de l'ACK au receveur à cause de l'algorithme d'ACK retardé.

Taille en cours à réception d'un ACK plein peut être zéro dans certaines situations. Un exemple typique est quand la taille de la fenêtre d'envoi durant la récupération rapide est petite. Dans ce cas, le paquet retransmis et les paquets de nouvelles données peuvent être transmis dans un bref intervalle. Si tous ces paquets arrivent bien, le receveur peut générer un ACK plein qui accuse réception de toutes les données en instance. Même si la taille de fenêtre n'est pas petite, la perte de paquets de ACK ou une panne de mémoire tampon de réception durant la récupération rapide peuvent aussi augmenter la possibilité de tomber dans cette situation.

Le remède proposé dans le présent document, qui règle cwnd à au moins 2*SMSS si la mise en œuvre utilise l'option 1 du cas de l'ACK plein (option 1 de l'étape 3 du paragraphe 3.2) assure que l'envoyeur TCP transmet au moins deux segments à réception d'un ACK plein.

De plus, un erratum a été ajouté à la RFC3782 (une précision rédactionnelle à la Section 8) ; cet erratum a été intégré à la Section 6 du présent document.

Le texte de spécification (ici au paragraphe 3.2) a été réécrit pour suivre de plus près le paragraphe 3.2 de la [RFC5681].

Les Sections 4, 5, et 9 à 11 de la [RFC3782] ont été retirés et l'Appendice A du présent document a été ajouté à la place pour faire référence à ces informations. Quelques références qui ne sont pas citées dans le corps principal du document ont été retirées.

Adresse des auteurs

Tom Henderson
The Boeing Company
mél : thomas.r.henderson@boeing.com

Sally Floyd
International Computer Science Institute
téléphone : +1 (510) 666-2989
mél : floyd@acm.org
URL: http://www.icir.org/floyd/

Andrei Gurtov
University of Oulu
Centre for Wireless Communications CWC
P.O. Box 4500
FI-90014 University of Oulu
Finland
mél : gurtov@ee.oulu.fi

Yoshifumi Nishida
WIDE Project
Endo 5322
Fujisawa, Kanagawa 252-8520
Japan
mél : nishida@wide.ad.jp