Network Working Group
Request for Comments : 3155
BCP : 50
Catégorie : Norme de pratique recommandée
N. Vaidya
Août 2001

S.Dawkins
G.Montenegro
M. Kojo
V. Magret

Implications des liens défectueux sur les performances de bout en bout

Statut de ce document

Ce document stipule une norme de pratique Internet recommandée pour la communauté Internet et invite toute discussion et suggestion visant à son amélioration. La distribution de ce document est libre.

Droits d'auteur

Résumé

Ce document traite des mécanismes spécifiques au protocole TCP posant problème dans des environnements à taux élevés d'erreurs non corrigées, et discute de ce que l'on peut faire pour atténuer les problèmes sans introduire de périphériques intermédiaires dans la connexion.

Table des matières

1 Introduction

Actuellement en plein essor, l'Internet fait l'objet d'accès par des dispositifs de plus en plus variés, et ce via des liens tout aussi hétéroclites. Certains de ces liens n'assurent pas le degré de fiabilité exigé par les hôtes et cette multiplication de liens non fiables est responsable des faibles performances de certains protocoles Internet, notamment le protocole TCP (protocole du niveau de transport) [RFC793].

Ainsi, le contrôle de congestion TCP [RFC2581] convient aux connexions qui perdent du trafic principalement en raison de congestion et de perte de mémoire tampon mais traite difficilement les erreurs non corrigées lorsque des connexions TCP empruntent des liens présentant des taux élevés d'erreurs non corrigées. Résultat : les émetteurs TCP passent énormément de temps à attendre en vain des accusés de réception (ACK). Leur vitesse de transmission est alors considérablement réduite, bien que ces pertes ne soient pas dues à des manques de mémoire tampon liés à une congestion. Ils doivent en effet tester le réseau pour déterminer les niveaux de trafic 'sécurisés'.
Ce document ne traite pas des autres protocoles de transport tels que UDP par exemple.

Dans le domaine de l'Internet, le processus 'congestion-avoidance' (évitement de congestion) part du principe que la plupart des pertes de paquets sont dues à la congestion et interprète ainsi par conséquent l'absence d'accusés de réception comme une indication de congestion. Ce principe fonctionne correctement depuis son introduction en 1988 [VJ-DCAC]. En effet, lors d'un fonctionnement normal, la plupart des liens et des sous-réseaux présentent un taux relativement faible d'erreurs et la congestion représente la principale cause de perte dans ces environnements. Néanmoins, avec la prolifération des liens et sous-réseaux qui n'apprécient pas les erreurs non corrigées, notamment sur les liens terrestres (hertziens) et satellitaires, les utilisateurs qui comptent sur le trafic à travers ces liens risquent d'être exposés à des performances réduites. En effet, leurs connexions TCP s'éternisent sur les phases 'congestion avoidance' et/ou 'slow start' (démarrage lent) déclenchées par les pertes de paquet dues aux erreurs de transmission.

Les recommandations de ce document visent à optimiser l'utilisation de la capacité de chemin disponible à travers des liens présentant des taux d'erreurs élevés afin de ne pas compromettre la stabilité de l'Internet.

Les applications utilisent TCP de diverses manières très différentes, et celles ci interagissent avec son comportement) sont extrêmement variées et interagissent avec son comportement [RFC2861]. Il est néanmoins possible d'établir des hypothèses simples sur les flux TCP. Ainsi, les mécanismes traités ici s'appliquent à toute utilisation de TCP, bien qu'il existe différents degrés en fonction des scénarios (comme expliqué le cas échéant).

Cette recommandation se base sur l'hypothèse explicite que les modifications importantes ne sont pas forcément apportées à toute la base installée de routeurs et d'hôtes, mais uniquement aux hôtes directement affectés par les liens défectueux.

1.1 Les raisons de lire cette recommandation

Toute technologie de sous-réseau connue présente un service de sous-réseau 'imparfait', le taux d'erreur par bit ne pouvant être nul. Mais il n'existe pas de méthode permettant aux stations terminales de différencier les paquets éliminés en raison de congestion et les pertes dues à des erreurs de transmission.

Lorsqu'un sous-réseau directement connecté rapporte des erreurs de transmission à un hôte, ces rapports sont importants, mais nous ne pouvons pas compter sur des rapports d'erreurs de transmission explicite en provenance des deux hôtes.

Les mathématiques offrent une autre méthode permettant de reconnaître un sous-réseau présentant un taux d'erreurs élevé. Une formule approximative pour la fonction de réponse TCP de Reno est expliquée

dans le document [PFTK98] :

$$T = \frac{s}{RTT * \sqrt{\frac{2p}{3}} + tRTO * (3 * \sqrt{\frac{3p}{8}}) * p * (1 + 32p^2)}$$

où

- T = le débit d'émission en octets par seconde
- s = la taille du paquet en octets
- RTT = temps aller-retour en secondes
- tRTO = valeur en secondes du temporisateur de réémission TCP
- p = taux de perte de paquets en régime permanent

Imaginons qu'une personne se connecte après avoir observé un certain taux de perte de paquets, puis, en faisant le calcul, note que l'utilisation de la bande passante prévue est supérieure à la vitesse de liaison. La connexion ne pourra alors tirer parti des recommandations expliquées dans ce document car le niveau de perte de paquets n'affectera pas la capacité de TCP à utiliser cette liaison. Dans le cas contraire, si la bande passante prévue est inférieure à la vitesse de liaison, les pertes de paquets affecteront la capacité de TCP à utiliser la liaison.

En outre, si de plus amples observations font apparaître des taux d'erreurs de transmission considérables dans un sous-réseau, les recommandations contenues dans ce document permettront alors d'optimiser la capacité de TCP à utiliser la liaison.
Voici quelques avertissements, par ordre d'importance, pour procéder à ce calcul :

1. La durée aller-retour est celle de bout en bout en non celle de la liaison.
2. Il est possible de simplifier la valeur de tRTO en utilisant la formule $Max(1.0, 4 * RTT)$.
3. Les pertes peuvent se produire en paquets en rafale : le taux de perte mesuré sur un intervalle incluant différents événements de pertes de paquets en rafale peut minimiser l'impact de ces événements sur le débit d'émission.

1.2 Rapports avec les proxies améliorant les performances

Ce document traite des mécanismes de bout en bout n'exigeant pas une reconnaissance de niveau TCP via des nœuds intermédiaires, ceci limitant considérablement la capacité des nœuds terminaux à reconnaître la nature des pertes occasionnées entre eux. Les études qui ont tenté d'appliquer l'heuristique pour distinguer congestion et erreur de transmission ont été vaines [BV97, BV98, BV98a]. Cette restriction est expliquée dans un document informel sur les proxies améliorant les performances (PEP) [RFC3135]. Ces proxies pouvant être placés sur des zones intermédiaires où les caractéristiques réseau changent rapidement, ils sont les plus à même d'optimiser les performances du trafic circulant sur des liens défectueux ou présentant des erreurs non corrigées.

Néanmoins, une utilisation généralisée des ces proxies est contraire au principe de bout en bout et ne convient particulièrement pas en raison de leurs implications nuisibles, notamment un manque de distribution de destination (un proxy améliorant les performances ajoute un troisième point de panne entre les points terminaux), des diagnostiques et une fiabilité de bout en bout, la compromission de la sécurité de bout en bout (notamment la sécurité de couche réseau telle que l'IPsec), la mobilité (les transferts sont beaucoup plus complexes car l'état doit être transféré), le routage asymétrique (en règle générale, ces proxies doivent se situer sur les chemins aller et retour d'une connexion), l'évolutivité (les proxies améliorant les performances ajoutent un état supplémentaire à surveiller), les garanties et la transparence du niveau de service (QoS).

Tous les proxies améliorant les performances ne présentent pas tous ces inconvénients. Néanmoins, leur utilisation peut avoir des conséquences très lourdes qu'il ne faut pas négliger.

1.3 Rapports avec les mécanismes de couche liaison de données

Cette recommandation doit être appliquée au protocole TCP dans les technologies de sous-réseau (couches lien) d'ores et déjà déployées. Ce document, développé ou en cours de développement par le groupe PILC (Performance Implications of Link Characteristics, c.-à-d. Implications sur les Performances des Caractéristiques des Liens) WG [PILC-WEB], propose une norme de pratique Internet recommandée concernant les sous-réseaux destinés à transmettre des protocoles Internet mais qui n'ont pas fait l'objet d'une spécification complète. Ce document est destiné aux concepteurs qui sont encore à même de réduire le nombre d'erreurs non corrigées rencontrées par le protocole TCP.

2 Erreurs et interactions avec les mécanismes TCP

Un émetteur TCP adapte son utilisation de la capacité de chemin du réseau en fonction des accusés de réception qu'il reçoit du récepteur TCP. TCP n'étant pas capable de faire la distinction entre des pertes dues à une congestion et des pertes dues à des erreurs non corrigées, il ne peut pas déterminer précisément la capacité de chemin disponible en présence d'un taux d'erreurs élevé.

2.1 Processus 'slow start' et 'congestion avoidance' [RFC2581]

Les processus 'slow start' et 'congestion avoidance' [RFC2581] sont essentiels à la stabilité actuelle de l'Internet. Ces mécanismes ont été conçus pour faciliter l'utilisation de réseaux ne fournissant pas d'informations explicites sur leur congestion. Bien que des mécanismes expérimentaux comme la recommandation [RFC2481] tendent vers la notification explicite de la congestion, l'effet du mécanisme ECN (Explicit Congestion Notification) sur les implémentations TCP supportant ECN ressemble fortement à l'effet de la notification implicite de congestion : il utilise les pertes liées à une congestion. Cependant, le mécanisme ECN permet quant à lui d'obtenir ces informations avant la perte des paquets et d'éviter ainsi leur retransmission.

Les connexions TCP présentant des taux d'erreurs élevés sur leurs chemins ne conviennent pas avec les phases 'slow start' et 'congestion avoidance'. En effet, les taux d'erreurs élevés provoquent une interprétation ambiguë des pertes car ils empêchent l'émetteur de savoir si les pertes détectées sont dues à une congestion ou à une corruption de données. TCP préfère alors la sécurité et présume que les pertes sont dues à une congestion.

- Lorsque les émetteurs TCP reçoivent trois accusés de réception dans le mauvais ordre, ils présument que le réseau est légèrement congestionné et appellent les processus 'fast retransmit/fast recovery' (voir ci-dessous).

- Lorsque le temporisateur de retransmission de TCP expire, l'émetteur présume que le réseau est congestionné et invoque la procédure 'slow start'.

- Les couches liaison de données moins fiables utilisent souvent de petits MTU (Maximum Transmission Unit - unité maximum de transmission) de liens ; ce qui a pour effet de ralentir le taux d'agrandissement de la taille de la fenêtre de l'émetteur au cours de la phase 'slow start', cette fenêtre augmentant par unité d'un segment. Utilisés seuls, les petits MTU de lien ne peuvent optimiser la fiabilité. Il faut également avoir recours au processus de découverte du MTU d'un chemin [RFC1191] pour empêcher la fragmentation. Ce processus permet l'ouverture la plus rapide de la taille de la fenêtre de l'émetteur au cours de la phase 'slow start', mais un certain nombre d'allers-retours peut toujours être requis pour ouvrir complètement la fenêtre.

Recommandation : tous les TCP conformes aux normes instaureront les phases 'slow start' et 'congestion avoidance', comme stipulé par le mot-clé 'DOIT' dans le document STD 3 [RFC1122]. Les recommandations proposées dans ce document n'interfèrent nullement avec ces mécanismes.

2.2 Processus ’Fast Retransmit’ et ’Fast Recovery’ [RFC2581]

TCP permet une transmission fiable des données sous la forme d’un flux d’octets vers une application. Ainsi, lorsqu’un segment est perdu (qu’il s’agisse d’une congestion ou d’une perte de transmission), le récepteur TCP doit attendre de recevoir les informations manquantes avant de transmettre les données à l’application réceptrice. Le récepteur TCP détecte au fur et à mesure les segments manquants qui arrivent dans le mauvais ordre.

Lorsqu’il reçoit des données dans le mauvais ordre [RFC2581], TCP doit alors immédiatement envoyer un accusé de réception stipulant le numéro du prochain segment attendu, afin que l’émetteur puisse retransmettre les données requises aussi rapidement que possible et ainsi continuer la transmission de données vers l’application réceptrice. Lorsqu’un accusé de réception porte le même numéro qu’un accusé de réception précédemment envoyé pour le dernier segment ordonné reçu, on dit que ces accusés de réception sont dupliqués.

Les réseaux IP étant autorisés à réordonner les paquets, le récepteur peut envoyer des accusés de réception dupliqués pour des segments qui arrivent dans le désordre en raison de modifications de route, de retransmission de niveau couche liaison de données, etc. Lorsqu’un émetteur TCP reçoit un accusé de réception en trois exemplaires, le mécanisme ’fast retransmit’ [RFC2581] lui permet d’inférer qu’un segment est perdu. L’émetteur retransmet ce qu’il considère être le segment perdu sans attendre la temporisation de la retransmission. Ce procédé permet ainsi de gagner du temps.

Après cette phase, un émetteur réduit de moitié sa fenêtre de congestion et invoque l’algorithme ’fast recovery’ [RFC2581], par lequel il appelle la phase ’congestion avoidance’ à partir d’une fenêtre de congestion réduite. Il n’a alors pas recours à la phase ’slow start’ depuis une fenêtre de congestion d’un seul segment comme il le ferait suite à l’expiration d’un temporisateur de retransmission.. Comme l’émetteur reçoit toujours des accusés de réception dupliqués, il sait que le récepteur reçoit les paquets envoyés. Par conséquent, lorsque aucune communication n’est reçue, il n’appelle pas le processus de réduction complète suivant une temporisation. Cette optimisation relativement sécurisée permet également de gagner du temps.

Il faut être réaliste quant au débit maximum que TCP peut avoir pour une connexion traversant un lien présentant un taux d’erreurs élevé. En général, TCP agrandira sa fenêtre de congestion au delà du produit délai bande passante. La stratégie de contrôle de congestion TCP répond donc au concept ’additive-increase, multiplicative-decrease’ (accroissement additif, décroissement multiplicatif), ce qui signifie que si d’autres erreurs sont détectées avant que la fenêtre de congestion ait retrouvé sa taille initiale après une réduction de moitié, la fenêtre de congestion donne alors l’effet d’une ’spirale descendante’ avec de nouvelles réductions de 50 %. Même en utilisant les algorithmes ’Fast Retransmit/Fast Recovery’, l’émetteur réduira de moitié la taille de la fenêtre de congestion pour chaque fenêtre présentant un ou plusieurs segments perdus, puis rouvrira la fenêtre par un segment supplémentaire pour chaque accusé de réception reçu par fenêtre de congestion.

Lorsqu’un chemin de connexion traverse un lien qui perd un ou plusieurs segments au cours de la phase de restauration, une nouvelle réduction de moitié se produit, mais cette fois sur une fenêtre de congestion réduite. Cette spirale descendante continuera à maintenir la fenêtre de congestion en dessous de sa capacité de chemin jusqu’à ce que la connexion soit capable de se restaurer complètement par augmentation additive et sans perte.

Lorsque le taux d’erreurs est constamment élevé, on ne parle naturellement pas de spirale descendante et la fenêtre de congestion reste réduite. Dans ce cas, la phase d’augmentation multiplicative "slow start" sera réduite et la fenêtre de congestion reste petite pendant la connexion TCP. Dans les liens présentant un taux d’erreurs élevé, la fenêtre TCP peut rester relativement petite pendant de longs moments.

Les erreurs ne sont pas la seule cause des fenêtres réduites. Par exemple, HTTP/1.0 ferme souvent les connexions TCP pour indiquer des limites entre des ressources requises. Ainsi, ces applications ferment

constamment des connexions TCP 'instruites' et ouvrent des connexions TCP 'non instruites' qui exécuteront la phase 'slow start' en commençant avec un ou deux segments. Cette procédure peut même se produire avec HTTP/1.1 lorsque les webmestres configurent leurs serveurs HTTP/1.1 pour fermer les connexions sans attendre de savoir si la connexion sera réutilisée.

Une petite fenêtre, notamment une fenêtre constituée de moins de quatre segments, empêche effectivement l'émetteur de tirer parti du processus 'Fast Retransmit'. En outre, une restauration efficace de différentes pertes au sein d'une seule et même fenêtre nécessite l'adoption de nouvelles méthodes (NewReno [RFC2582]).

Recommandation : implémentez les algorithmes 'Fast Retransmit' et 'Fast Recovery' dès maintenant. Cette optimisation largement plébiscitée est actuellement en cours de normalisation. [RFC2488] recommande l'instauration des algorithmes 'Fast Retransmit/Fast Recovery' dans des environnements satellites.

2.3 Accusés de réception sélectifs [RFC2018, RFC2883]

Les accusés de réception sélectifs [RFC2018] permettent de pallier la perte de plusieurs segments par fenêtre sans avoir à effectuer un (ou plusieurs) aller-retour par perte.

[RFC2883] propose une petite extension aux accusés de réception sélectifs (SACK) qui permettent de recevoir TCP afin de fournir plus d'informations concernant l'ordre de livraison des segments. Cette méthode permet ainsi "un fonctionnement plus robuste dans un environnement de paquets réordonnés ou répliqués, de perte d'accusés de réception et/ou de temporisations de retransmission anticipées". Sauf indication contraire, dans ce document, 'accusé de réception sélectif' (ou 'SACK') réfère à la combinaison de [RFC2018] et de [RFC2883].

Les accusés de réception sélectifs sont particulièrement pratiques dans les réseaux dits éléphants (LFN - Long Fat Network) en raison de la longue durée d'un aller-retour dans ces environnements, tel qu'expliqué dans la section 1.1 de [RFC1323]. En outre, ces accusés de réception sont également pratiques lorsque de grandes fenêtres sont requises car la probabilité de perdre plusieurs segments par fenêtre est alors accrue.

D'autre part, lorsque des taux d'erreurs généralement bas se voient de temps en temps accrus en raison de mauvaises conditions des chaînes, TCP pourra accroître sa fenêtre à des valeurs supérieures lorsque ces conditions s'améliorent entre des pics d'erreurs. Au moment de ces pics d'erreurs, plusieurs pertes au sein d'une même fenêtre sont susceptibles de se produire. Dans ce cas, l'accusé de réception sélectif offre alors l'avantage d'accélérer la restauration et d'empêcher toute réduction inutile de la taille de la fenêtre.

Recommandation : implémentez les accusés de réception sélectifs conformément à l'explication donnée dans [RFC2018] et mise à jour dans [RFC2883], qui sont deux normes recommandées. Dans le cas où les accusés de réception sélectifs ne peuvent être activés aux deux extrémités de la connexion, les émetteurs TCP peuvent utiliser la norme NewReno [RFC2582] pour mieux manipuler ces accusés de réception et faire face à plusieurs pertes au sein d'une même fenêtre.

3 Résumé des recommandations

Obtenir des informations sur les pertes permettrait à TCP de faire la distinction entre congestion et erreur de transmission, mais l'Internet ne dispose pas à grande échelle d'un tel système. Alors que la congestion affecte tout le trafic d'un chemin, une perte de transmission n'affecte que le trafic spécifique présentant des erreurs non corrigées. C'est pourquoi il faut favoriser la suppression de la congestion plutôt qu'une réparation rapide des erreurs de transmission. Ainsi, le meilleur procédé actuel consiste à réduire le temps passé inutilement à contrôler la congestion.

Le mécanisme 'Fast Retransmit/Fast Recovery' permet de réparer rapidement les pertes sans compromettre la sécurité du contrôle de la congestion. Pour que ce mécanisme fonctionne, la taille de la fenêtre doit être assez grande pour forcer le récepteur à envoyer trois accusés de réception dupliqués avant l'expiration du délai de temporisation de retransmission, forçant ainsi la phase complète 'slow-start' TCP.

Les accusés de réception sélectifs (SACK) étendent les avantage du mécanisme 'Fast Retransmit/Fast Recovery' à des situations exigeant une réparation de plusieurs pertes de segments d'une fenêtre plus rapide que par l'exécution du processus 'Fast Retransmit' pour chaque perte de segment, qui permettrait uniquement de découvrir la perte de segment suivante.

Ces mécanismes ne se limitent pas aux environnement sans fil mais peuvent être appliqués à tous les environnements.

4 Pour approfondir ce sujet

Le document [RFC3042] intitulé 'Limited Transmit' (Émission limitée) est venu optimiser l'algorithme 'Fast Retransmit/Fast Recovery' pour les connexions TCP présentant une petite fenêtre de congestion et ne déclenchant pas trois accusés de réception dupliqués. Jugée fiable, cette spécification confère des avantages aux connexions TCP supportant un nombre important de pertes de paquets (données ou accusés de réception). Les personnes chargées de la mise en IJuvre d'applications devraient évaluer cette norme pour le protocole TCP dans de tels environnements.

Les accusés de réception dupliqués différés [MV97, VMPM99] visent à bloquer la retransmission au niveau TCP lorsque la retransmission au niveau couche est toujours en cours, ajoutant du trafic supplémentaire au réseau. Cette proposition mérite une étude approfondie, mais pour le moment, elle n'est pas encore recommandée car nous ne savons pas comment calculer le délai nécessaire pour une topologie de réseau arbitraire.

Il n'est pas possible de substituer une notification d'erreur de transmission explicite par une notification de congestion explicite [RFC2481] (même en y croyant dur comme fer !). Il serait particulièrement pratique de disposer de mécanismes permettant de notifier explicitement des erreurs de transmission. De tels mécanismes semblent notamment plus aisés à mettre en place dans un environnement PEP (proxies améliorant les performances), surtout lorsque cet environnement représente le 'premier saut' d'un chemin de connexion. En effet, les mécanismes de somme de contrôle actuels ne font pas la distinction entre erreur de transmission due à la une charge utile et erreur de transmission due à l'en-tête. En outre, lorsque l'entête est endommagé, l'envoi d'une notification d'erreur de transmission explicite à un point donné devient problématique.

Les pertes se produisant sur le flux des accusés de réception, en particulier au moment où un TCP apprend les caractéristiques d'un réseau, peuvent rendre le flux de données discontinu (occasionnant ainsi également des pertes sur le flux de données). Plusieurs solutions ont été proposées pour résoudre ce problème, notamment au niveau de la vitesse d'émission TCP chez l'émetteur et du contrôle du taux d'accusés de réception au sein du réseau.

La méthode ABC (Appropriate Byte Counting) [ALL99] a été proposée pour ouvrir la fenêtre de congestion en fonction du nombre d'octets transférés avec succès au récepteur. Elle permet ainsi un comportement plus approprié à l'application de protocoles initiant les connexions avec des paquets relativement petits. Pour SMTP [RFC2821], par exemple, le client peut envoyer un petit paquet HELO, un petit paquet MAIL, un ou plusieurs petits paquets RCPT, ainsi qu'un petit paquet DATA, le tout suivi par le corps entier du message envoyé sous forme de paquets de longueur maximale. Un émetteur TCP ABC n'utiliserait pas d'accusé de réception pour chacun de ces petits paquets afin d'augmenter la fenêtre de congestion et permettre ainsi la transmission de paquets supplémentaires de longueur maximale. Cette méthode ABC mérite une étude approfondie mais n'est pas encore recommandée car elle peut conduire à une augmentation de la

transmission en paquets en cas de perte d'accusés de réception.

4.1 Obtention et maintien de grandes fenêtres

Les recommandations présentées dans ce document aident les connexions TCP à injecter des paquets dans des connexions comportant des erreurs (ERRORed) le plus rapidement possible et sans déstabiliser l'Internet, optimisant ainsi l'utilisation de la bande passante disponible.

Outre ces recommandations au niveau TCP, il reste du travail à faire au niveau applicatif, notamment concernant le principal protocole applicatif du World Wide Web : HTTP.

HTTP/1.0 (et les versions antérieures) ferme les connexions TCP pour signaler au récepteur l'émission de toutes les ressources requises. Les objets du WWW ayant de plus en plus tendance à être petits [MOGUL], les connexions TCP qui transportent du trafic HTTP/1.0 rencontrent des difficultés lorsqu'elles 's'entraînent' sur la capacité de chemin disponible (une partie importante du transfert s'est déjà déroulée avant que TCP ne quitte la phase 'slow start').

Plusieurs modifications HTTP ont permis d'optimiser cette interaction avec TCP ('connexions persistantes' sous HTTP/1.0, avec des améliorations sous HTTP/1.1 [RFC2616]). Pour différentes raisons, de nombreuses interactions HTTP sont encore de type HTTP/1.0, avec une durée de vie relativement courte.

Des normes proposent de réutiliser les informations sur la congestion TCP au travers des connexions, notamment TCP Control Block Interdependence [RFC2140] ou la proposition plus récente Congestion Manager [BS00]. Elles permettront de faire rentrer dans le réseau plusieurs connexions parallèles sous la forme d'une seule connexion, "entraînée" suite à un seul démarrage. Ces projets sont indispensables à la stabilité à long terme de l'Internet. En effet, les internautes sont actuellement à même d'ignorer la technique de 'l'exponential backoff' de TCP en cliquant sur le bouton "Reload" (Actualiser) de leur navigateur. Cette méthode remplace ainsi les connexions qui construisent des connaissances sur la bande passante disponible par des connexions dépourvues de toutes connaissances.

5 Sécurité

L'algorithme 'Fast Retransmit/Fast Recovery' comporte un point faible potentiel (tel qu'expliqué dans le document [RFC2581]) : une attaque peut créer des files d'attentes dans des connexions TCP, voire même, ce qui est encore plus dangereux, se comporter de façon plus agressive. Cette dernière possibilité peut entraîner un effondrement dû à la congestion, du moins dans certaines zones du réseau.

On pense que les accusés de réception sélectifs ne renforcent pas les propriétés de sécurité actuelles de TCP [RFC2018] mais ne les affaiblissent pas non plus.

Les recommandations de ce document s'exécutant sur une base de bout en bout, elles s'appliquent également en présence d'Ipsec de bout en bout. Elles sont donc en parfaite opposition à des mécanismes telles que les PEP (proxies améliorant les performances) qui s'exécutent sur des nIJuds intermédiaires (section 1.2).

6 Considérations de l'IANA

Ce document se base sur les autres normes existantes de l'IETF (Internet Engineering Task Force). Il ne présente pas de nouvelles considérations de l'IANA.

7 Remerciements

Cette recommandation est issue du document RFC 2757, "Long Thin Networks - Réseaux longs et fins", lui-même basé sur une étude menée par le groupe de travail IETF TCPSAT. Les auteurs remercient particulièrement les membres actifs du groupe de travail PILC, notamment Mark Allman et Lloyd Wood qui nous ont fourni de nombreuses et précieuses informations, ainsi que Dan Grossman et Jamshid Mahdavi qui nous ont apporté les copies de textes.

Références

[ALL99] M. Allman, *TCP Byte Counting Refinements*, ACM Computer Communication Review, volume 29, numéro 3, juillet 1999. http ://www.acm.org/sigcomm/ccr/archive/ccr-toc/ccr-toc-99.html

[BS00] Balakrishnan, H. et S. Seshan, *The Congestion Manager*, FC 3124, juin 2001.

[BV97] S. Biaz et N. Vaidya, *Using End-to-end Statistics to Distinguish Congestion and Corruption Losses : A Negative Result*, Texas A&M University, rapport technique 97-009, 18 août 1997.

[BV98] S. Biaz et N. Vaidya, *Discriminating Congestion Losses from Wireless Losses using Inter-Arrival Times at the Receiver*, Texas A&M University, rapport technique 98-014, juin 1998.

[BV98a] S. Biaz et N. Vaidya, *Discriminating Congestion Losses from Wireless Losses using Inter-Arrival Times at the Receiver*, Texas A&M University, rapport technique 98-014, juin 1998.

[MOGUL] J. C. Mogul, *The Case for Persistent-Connection HTTP*, J. C. Mogul, rapport de recherche 95/4, May 1995. Disponible sur le site : //www.research.digital.com/wrl/techreports/abstracts/95.4.html

[MV97] M. Mehta et N. Vaidya, *Delayed Duplicate-Acknowledgements : A Proposal to Improve Performance of TCP on Wireless Links*, Texas A&M University, 24 décembre 1997. Disponible sur le site http ://www.cs.tamu.edu/faculty/vaidya/mobile.html

[PILC-WEB] http ://pilc.grc.nasa.gov/

[PFTK98] Padhye, J., Firoiu, V., Towsley, D. et J.Kurose, *TCP Throughput : A simple model and its empirical validation*, SIGCOMM Symposium on Communications Architectures and Protocols, août 1998.

[RFC793] Postel, J., *Transmission Control Protocol*, STD 7, RFC 793, septembre 1981.

[RFC2821] Klensin, J., Editor, *Simple Mail Transfer Protocol*, RFC 2821, avril 2001.

[RFC1122] Braden, R., *Requirements for Internet Hosts – Communication Layers*, STD 3, RFC 1122, octobre 1989.

[RFC1191] Mogul J. et S. Deering, *Path MTU Discovery*, RFC 1191, novembre 1990.

[RFC1323] Jacobson, V., Braden, R. et D. Borman. *TCP Extensions for High Performance*, RFC 1323, mai 1992.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S. et A. Romanow *TCP Selective Acknowledgment Options*, RFC 2018, octobre 1996.

[RFC2140] Touch, J., *TCP Control Block Interdependence*, RFC 2140, avril 1997.

[RFC2309] Braden B., Clark D., Crowcrfot J., Davie B., Deering S., Estrin D., Floyd S., Jacobson V., Minshall G., Partridge C., Peterson L., Ramakrishnan K., Shecker S., Wroclawski J. et L Zhang, *Recommendations on Queue Management and Congestion Avoidance in the Internet*, RFC 309, avril 1998.

[RFC2481] Ramakrishnan K. et S. Floyd, *A Proposal to add Explicit Congestion Notification (ECN) to IP*, RFC 2481, janvier 1999.

[RFC2488] Allman, M., Glover, D. et L. Sanchez. *Enhancing TCP Over Satellite Channels using Standard Mechanisms*, BCP 28, RFC 2488, janvier 1999.

[RFC2581] Allman, M., Paxson, V. et W. Stevens, *TCP Congestion Control*, RFC 2581, avril 1999.

[RFC2582] Floyd, S. et T. Henderson, *The NewReno Modification to TCP's Fast Recovery Algorithm*, RFC 2582, avril 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach P. et T. Berners-Lee, *Hypertext Transfer Protocol – HTTP/1.1*, RFC 2616, juin 1999.

[RFC2861] Handley H., Padhye J. et S. Floyd, *TCP Congestion Window Validation*, RFC 2861, juin 2000.

[RFC2883] Floyd S., Mahdavi M., Mathis M. et M. Podlosky, *An Extension to the Selective Acknowledgement (SACK) Option for TCP*, RFC 2883, août 1999.

[RFC2923] Lahey K., *TCP Problems with Path MTU Discovery*, RFC 2923, septembre 2000.

[RFC3042] Allman M., Balakrishnan H. et S. Floyd, *Enhancing TCP's Loss Recovery Using Limited Transmit*, RFC 3042, janvier 2001.

[RFC3135] Border J., Kojo M., Griner J., Montenegro G. et Z. Shelby, *Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations*, RFC 3135, juin 2001.

[VJ-DCAC] Jacobson V., *Dynamic Congestion Avoidance / Control* e-mail daté du 11 février 1988, disponible sur le site http ://www.kohala.com/ rstevens/vanj.88feb11.txt

[VMPM99] N. Vaidya, M. Mehta, C. Perkins et G. Montenegro, *Delayed Duplicate Acknowledgements : A TCP-Unaware Approach to Improve Performance of TCP over Wireless*, rapport technique 99-003, Computer Science Dept., Texas A&M University, février 1999. Doit également paraître dans le Journal of Wireless Communications and Wireless Computing (Numéro spécial sur les protocoles de transport fiables pour l'informatique mobile).

Pour contacter les auteurs

Pour toute question concernant ce document, veuillez vous adresser à :

Spencer Dawkins
Fujitsu Network Communications
2801 Telecom Parkway
Richardson, Texas 75082
Tél. : +1-972-479-3782
E-mail : spencer.dawkins@fnc.fujitsu.com

Gabriel E. Montenegro
Sun Microsystems
Laboratories, Europe
29, chemin du Vieux Chêne
38240 Meylan
FRANCE
Tél. : +33 476 18 80 45
E-mail : gab@sun.com

Markku Kojo
Department of Computer Science
University of Helsinki
P.O. Box 26 (Teollisuuskatu 23)
FIN-00014 HELSINKI
Finland
Tél. : +358-9-1914-4179
E-mail : kojo@cs.helsinki.fi

Vincent Magret
Alcatel Internetworking, Inc.
26801 W. Agoura road
Calabasas, CA, 91301
Tél. : +1 818 878 4485
E-mail : vincent.magret@alcatel.com

Nitin H. Vaiday
458 Coodinated Science Laboratory, MC-228
1308 West Main Street
Urbana, IL 61801
Tél. : 217-265-5414
E-mail : nhv@crhc.uiuc.edu

Déclaration complète des droits d'auteur

Remerciements

Les fonds nécessaires à l'édition des RFC sont actuellement fournis par l'Internet Society.