Groupe de travail Réseau
Request for Comments : 5348
RFC rendue obsolète : 3448
RFC mise à jour : 4342
Catégorie : Sur la voie de la normalisation
Traduction Claude Brière de L'Isle

S. Floyd, ICIR
M. Handley, University College London
J. Padhye, Microsoft
J. Widmer, DoCoMo
septembre 2008

Contrôle de taux favorable à FTP (TFRC) : spécification du protocole

Résumé
Le présent document spécifie le contrôle de taux favorable à TCP (TFRC, *TCP Friendly Rate Control*). TFRC est un mécanisme de contrôle d'encombrement pour les flux en envoi individuel opérant dans un environnement Internet au mieux. Il est raisonnablement équitable dans la compétition pour la bande passante avec les flux TCP, mais a une bien moindre variation de débit dans le temps comparé à TCP, le rendant très convenable pour les applications comme les supports en flux direct où un taux d'envoi relativement lissé est important.

Le présent document rend obsolète la RFC 3448 et met à jour la RFC 4342.

Table des matières

1. Introduction

Le présent document spécifie le contrôle de taux favorable à TCP (TFRC, *TCP Friendly Rate Control*). TFRC est un mécanisme de contrôle d'encombrement conçu pour les flux en envoi individuel opérant dans un environnement Internet et en concurrence avec le trafic TCP [FHPW00]. Au lieu de spécifier un protocole complet, le présent document spécifie simplement un mécanisme de contrôle d'encombrement qui pourrait être utilisé dans un protocole de transport comme le protocole de contrôle d'encombrement de datagrammes (DCCP, *Datagram Congestion Control Protocol*) [RFC4340], dans une application incorporant le contrôle d'encombrement de bout en bout au niveau application, ou dans le contexte de la gestion de l'encombrement aux points d'extrémité [BRS99]. Le présent document ne discute pas des formats de paquet ou de fiabilité. Les questions relatives à la mise en œuvre ne sont discutées que brièvement à la Section 8.

TFRC est conçu pour être raisonnablement équitable en présence de compétition pour la bande passante avec les flux TCP, où on dit qu'un flux est "raisonnablement équitable" si son taux d'envoi est généralement dans un facteur de deux du taux d'envoi d'un flux TCP dans les mêmes conditions. Cependant, TFRC a une bien plus faible variation de débit dans le temps comparée à TCP, ce qui le rend très convenable pour des applications comme la téléphonie ou les supports de flux en direct où un taux d'envoi relativement lissé est important.

L'inconvénient d'avoir un débit plus lissé que TCP tout en concourant de façon équitable pour la bande passante est que TFRC répond plus lentement que TCP aux changements de disponibilité de la bande passante. Donc, TFRC devrait n'être utilisé que quand l'application a une exigence de débit lissé, en particulier, évitant la division par deux du taux d'envoi de TCP en réponse à un seul abandon de paquet. Pour les applications qui ont simplement besoin de transférer autant de données que possible dans le plus court temps possible, on recommande d'utiliser TCP, ou si la fiabilité n'est pas exigée, d'utiliser un schéma de contrôle d'encombrement d'augmentation additive, diminution multiplicative (AIMD, *Additive-Increase, Multiplicative-Decrease*) avec des paramètres similaires à ceux utilisés par TCP.

TFRC est conçu pour de meilleures performances avec les applications qui utilisent une taille fixe de segment, et varient leur taux d'envoi en paquets par seconde en réponse à l'encombrement. TFRC peut aussi être utilisé, peut-être avec des performances moins optimales, avec des applications qui n'ont pas une taille fixe de segment, mais où la taille de segment varie selon les besoins de l'application (par exemple, applications de vidéo).

Certaines applications (par exemple, des applications audio) exigent un intervalle de temps fixe entre les paquets et font varier leur taille de segment au lieu de leur taux de paquet en réponse à l'encombrement. Le mécanisme de contrôle d'encombrement dans le présent document n'est pas conçu pour ces applications ; TFRC-SP (TFRC pour petits paquets) est une variante de TFRC pour les applications qui ont un taux d'envoi fixe en paquets par seconde mais utilisent de petits paquets ou font varier leur taille de paquet en réponse à l'encombrement. TFRC-SP est spécifié dans un document séparé [RFC4828].

Le présent document spécifie TFRC comme un mécanisme fondé sur le receveur, avec le calcul des informations de contrôle d'encombrement (c'est-à-dire, le taux d'événements de perte) chez le receveur des données plutôt que chez l'envoyeur des données. Cela convient bien à une application où l'envoyeur est un grand serveur traitant de nombreuses connexions concurrentes, et où le receveur a plus de mémoire et de cycles de CPU disponibles pour le calcul. De plus, un mécanisme fondé sur le receveur convient mieux comme bloc de construction pour le contrôle d'encombrement en diffusion groupée. Cependant, il est aussi possible de mettre en œuvre TFRC dans des variantes fondées sur l'envoyeur, comme il est permis dans l'identifiant 3 de contrôle d'encombrement de DCCP (CCID 3) [RFC4342].

Le présent document rend obsolète la RFC 3448. Dans le protocole de transport DCCP *(Datagram Congestion Control Protocol)* [RFC4340], les profils d'identifiant de contrôle d'encombrement CCID-3 [RFC4342] et CCID-4 [RFC5622] spécifient tous deux l'utilisation de TFRC à partir de la RFC 3448. Les mises en œuvre de CCID-3 et CCID-4 DEVRAIENT utiliser ce document plutôt que la RFC 3448 pour la spécification de TFRC.

La spécification normative de TFRC est dans les sections 3 à 6. La Section 7 discute les variantes fondées sur l'envoyeur, La Section 8 discute des questions de mise en œuvre, et la Section 9 donne une vue d'ensemble non normative des différences avec la RFC 3448.

2. Conventions

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].

L'Appendice A donne une liste des termes techniques utilisés dans le document.

3. Mécanisme du protocole

Pour son mécanisme de contrôle d'encombrement, TFRC utilise directement une équation de débit pour le taux d'envoi permis comme une fonction du taux d'événements de perte et du délai d'aller-retour. Afin d'exercer une concurrence équitable avec TCP, TFRC utilise l'équation de débit de TCP, qui décrit en gros le taux d'envoi de TCP comme une fonction du taux d'événements de perte, du délai d'aller retour, et de la taille de segment. On définit un événement de perte comme un ou plusieurs paquets perdus ou marqués dans une fenêtre de données, où un paquet marqué se réfère à une indication d'encombrement de notification explicite d'encombrement (ECN, *Explicit Congestion Notification*) [RFC3168].

D'une façon générale, le mécanisme de contrôle d'encombrement de TFRC fonctionne comme suit :
- o le receveur mesure le taux d'événements de perte et renvoie cette information à l'envoyeur ;
- o l'envoyeur utilise aussi des messages de rétroaction pour mesurer le délai d'aller-retour (RTT, *Round Trip Time*) ;
- o le taux d'événements de perte et le RTT sont alors introduits dans l'équation de débit de TFRC, et le taux d'envoi résultant est limité à au plus deux fois le taux de réception pour donner le taux de transmission permis X ;
- o l'envoyeur ajuste alors son taux de transmission pour correspondre au taux de transmission permis X.

La dynamique de TFRC est sensible à la façon dont les mesures sont effectuées et appliquées. On recommande ci-dessous des mécanismes spécifiques pour effectuer et appliquer ces mesures. D'autres mécanismes sont possibles, mais il est important de comprendre comment les interactions entre mécanismes affectent la dynamique de TFRC.

3.1 Équation de débit de TCP

Toute équation réaliste donnant le débit TCP comme fonction du taux d'événements de perte et du RTT devrait être convenable avec TFRC. Cependant, on note que l'équation de débit TCP utilisée doit refléter le comportement de temporisateur de retransmission de TCP, car cela domine le débit TCP aux taux de pertes importants. On note aussi que les hypothèses implicites de l'équation de débit sur le paramètre de taux d'événements de perte doivent être une correspondance raisonnable de la façon dont le taux de perte, ou le taux d'événements de perte, est en fait mesuré. Bien que cette correspondance ne soit pas parfaite pour l'équation de débit et la mesure du taux de perte données ci-dessous, en pratique les hypothèses se trouvent être à peu près vérifiées.

L'équation de débit actuellement EXIGÉE pour TFRC est une version légèrement simplifiée de l'équation de débit pour Reno TCP dans [PFTK98]. Idéalement, on préférerait une équation de débit fondée sur l'accusé de réception sélectif (SACK) TCP, mais personne n'a encore résolu l'équation de débit pour SACK TCP, et les simulations et expériences suggèrent que les différences entre les deux équations vont être relativement mineures [FF99] (Appendice B).

L'équation de débit pour X_Bps, le taux d'envoi moyen de TCP en octets par seconde, est :

$$X\_Bps = \frac{s}{R*\sqrt{(2*b*p/3)} + (t\_RTO * (3*\sqrt{(3*b*p/8)}*p*(1+32*p^2)))}$$

Où :

X_Bps : taux de transmission moyen de TCP en octets par seconde. (X_Bps est le même que X_calc dans la RFC3448.)

s : taille de segment en octets (excluant les en-têtes IP et de protocole de transport).

R : délai d'aller retour en secondes.

p : taux d'événements de perte, entre 0 et 1,0, du nombre d'événements de perte comme fraction du nombre de paquets transmis.

t_RTO : valeur du temporisateur de retransmission TCP en secondes.

b : nombre maximum de paquets acquittés par un seul accusé de réception TCP.

Réglage de la valeur du temporisateur de retransmission TCP t_RTO : les mises en œuvre DEVRAIENT régler t_RTO = 4*R. Les mises en œuvre PEUVENT choisir de mettre en œuvre un calcul plus précis de t_RTO. Les mises en œuvre PEUVENT aussi régler t_RTO à max(4*R, une seconde) pour correspondre au minimum recommandé de une seconde sur le RTO [RFC2988].

Réglage du paramètre b pour les accusés de réception retardés : certaines connexion TCP actuelles utilisent des accusés de réception retardés, envoyant un accusé de réception tous les deux paquets de données reçus. Cependant, il est aussi permis à TCP d'envoyer un accusé de réception à chaque paquet de données. Pour les mécanismes de contrôle de l'encombrement de TCP révisé la [RFC5681] spécifie actuellement que l'algorithme d'accusé de réception retardé devrait être utilisé avec TCP. Cependant, la [RFC5681] recommande d'augmenter la fenêtre d'encombrement durant l'évitement d'encombrement de un segment par RTT même en présence d'accusé de réception retardé, en cohérence avec une équation de débit TCP de b = 1. Sur une base expérimentale, la [RFC5681] permet des augmentations de la fenêtre d'encombrement durant le démarrage lent qui sont aussi cohérentes avec l' équation de débit TCP avec b = 1. Donc, l'utilisation de b = 1 est cohérente avec la [RFC5681]. L'utilisation de b = 1 est RECOMMANDÉE.

Avec t_RTO = 4*R et b = 1, l'équation de débit pour X_Bps, le taux d'envoi TCP en octets par seconde, peut être simplifié comme suit :

$$X\_Bps = \frac{s}{R * (\sqrt{(2*p/3)} + 12*\sqrt{(3*p/8)}*p*(1+32*p^2))}$$

À l'avenir, des mises à jour du présent document pourraient spécifier des équations TCP différentes à substituer à celle-ci. L'exigence est que l'équation de débit soit une approximation raisonnable du taux d'envoi de TCP pour un contrôle d'encombrement TCT conforme. L'équation de débit peut aussi être exprimée en termes de X_pps, le taux d'envoi en paquets par seconde, avec X_pps = X_Bps / s .

Les paramètres s (taille de segment), p (taux d'événements de perte), et R (RTT) doivent être mesurés ou calculés par une mise en œuvre de TFRC. La mesure de s est spécifiée au paragraphe 4.1, la mesure de R est spécifiée au paragraphe 4.3, et la mesure de p est spécifiée à la Section 5. Dans le reste du document, les taux de données sont mesurés en octets par seconde sauf mention contraire.

3.2 Contenu de paquet

Avant de spécifier les fonctions d'envoyeur et de receveur, on décrit le contenu des paquets de données envoyés par l'envoyeur et des paquets de rétroaction envoyés par le receveur. Comme TFRC va être utilisé avec un protocole de transport, on ne spécifie pas les formats de paquet, car ils dépendent des détails du protocole de transport utilisé.

3.2.1 Paquets de données

Chaque paquet de données envoyé par l'envoyeur des données contient les informations suivantes :

- Un numéro de séquence. Ce numéro DOIT être incrémenté de un pour chaque paquet de données transmis. Le champ doit être suffisamment grand pour qu'il ne revienne pas à zéro en causant la présence en même temps de deux paquets différents avec le même numéro de séquence dans l'historique de paquets récents du receveur.

- Un horodatage indiquant quand le paquet est envoyé. On note "ts_i" l'horodatage du paquet de numéro de séquence i.

La résolution de l'horodatage DEVRAIT être normalement mesurée en millisecondes.
Cet horodatage est utilisé par le receveur pour déterminer quelles pertes appartiennent au même événement de perte. Le receveur fait aussi écho à l'horodatage pour permettre à l'envoyeur d'estimer le délai d'aller-retour, pour les envoyeurs qui ne sauvegardent pas les horodatages des paquets de données transmis.
On note que, comme solution de remplacement à un horodatage incrémenté en millisecondes, un "horodatage" qui s'incrémente tous les quarts de délai d'aller-retour PEUT être utilisé pour déterminer quand des pertes appartiennent au même événement de perte, dans le contexte d'un protocole où ceci est compris par l'envoyeur et le receveur et où l'envoyeur sauvegarde les horodatages des paquets de données transmis.

- L'estimation actuelle du délai d'aller-retour par l'envoyeur. L'estimation rapportée dans le paquet i est notée R_i. L'estimation du délai d'aller-retour est utilisée par le receveur, avec l'horodatage, pour déterminer quand plusieurs pertes appartiennent au même événement de perte. L'estimation du délai d'aller-retour est aussi utilisée par le receveur pour déterminer l'intervalle à utiliser pour calculer le taux de réception et pour déterminer quand envoyer des paquets de rétroaction.
  Si l'envoyeur envoie un "horodatage" à gros grain qui s'incrémente tous les quarts de délai d'aller-retour, comme expliqué ci-dessus, l'envoyeur n'est alors pas obligé d'envoyer son estimation actuelle du délai d'aller-retour.

3.2.2 Paquets de rétroaction

Chaque paquet de rétroaction envoyé par le receveur des données contient les informations suivantes :
- L'horodatage du dernier paquet de données reçu. On le note t_recvdata. Si le dernier paquet reçu chez le receveur a le numéro de séquence i, alors t_recvdata = ts_i. Cet horodatage est utilisé par l'envoyeur pour estimer le délai d'aller-retour, et n'est nécessaire que si l'envoyeur ne sauvegarde pas les horodatages des paquets de données transmis.
- Le temps écoulé entre la réception du dernier paquet de données chez le receveur et la génération de ce rapport de rétroaction. Noté t_delay.
- Le taux auquel le receveur estime que les données ont été reçues dans le précédent délai d'aller-retour. Noté X_recv.
- L'estimation actuelle du receveur du taux d'événements de perte p.

4. Protocole d'envoyeur des données

L'envoyeur des données envoie un flux de paquets de données au receveur des données à un taux contrôlé. Quand un paquet de rétroaction est reçu du receveur des données, l'envoyeur des données change son taux d'envoi sur la base des informations contenues dans le rapport de rétroaction. Si l'envoyeur ne reçoit pas de rapport de rétroaction pendant quatre délais d'aller-retour, l'envoyeur divise son taux d'envoi par deux. Ceci est réalisé au moyen d'un temporisateur appelé le temporisateur Pas de rétroaction.

On spécifie le protocole du côté de l'envoyeur par les étapes suivantes :
- Mesure de la taille moyenne de segments envoyés.
- Initialisation de l'envoyeur.
- Le comportement de l'envoyeur quand un paquet de rétroaction est reçu.
- Le comportement de l'envoyeur quand le temporisateur Pas de rétroaction arrive à expiration.
- Prévention des oscillations (facultatif).
- Programmation de la transmission de paquets et sporadicité permise.

4.1 Mesure de la taille de segment

L'envoyeur TFRC utilise la taille de segment, s, dans l'équation de débit, dans le réglage du taux de réception maximum, le réglage des taux d'envoi minimum et initial, et le réglage du temporisateur Pas de rétroaction. Le receveur TFRC PEUT utiliser la taille de segment moyenne, s, pour initialiser l'historique de pertes après le premier événement de perte. Comme spécifié au paragraphe 6.3.1, si le receveur TFRC ne connaît pas la taille de segment, s, utilisé par l'envoyeur, le receveur TFRC PEUT utiliser à la place le taux d'arrivée en paquets par seconde lors de l'initialisation de l'historique des pertes.

La taille de segment est normalement connue d'une application. Il peut n'en être pas ainsi dans deux cas :

1) La taille de segment varie naturellement selon les données. Dans ce cas, bien que la taille de segment varie, cette variation n'est pas couplée au taux de transmission. L'envoyeur TFRC peut calculer la taille moyenne de segment ou utiliser la taille maximum de segment comme taille de segment, s.

2) L'application a besoin de changer la taille de segment plutôt que le nombre de segments par seconde pour effectuer le contrôle d'encombrement. Cela va normalement être le cas avec des applications de paquet audio où un intervalle de

temps fixe doit être représenté par chaque paquet. De telles applications ont besoin d'avoir des façons complètement différentes de mesurer les paramètres.

Pour la première classe d'applications où la taille de segment varie selon les données, l'envoyeur DEVRAIT estimer la taille de segment, s, comme la taille moyenne de segment sur les quatre derniers intervalles de perte. L'envoyeur PEUT estimer la taille moyenne de segment sur de plus longs intervalles de temps, si c'est désiré.

La seconde classe d'applications est discutée séparément dans un autre document sur TFRC-SP [RFC4828]. Dans le reste de cette section, on suppose que l'envoyeur peut estimer la taille de segment et que le contrôle d'encombrement est effectué en ajustant le nombre de paquets envoyés par seconde.

4.2 Initialisation de l'envoyeur

Les valeurs initiales pour X (le taux d'envoi permis en octets par seconde) et tld *(Time Last Doubled)* (dernier temps doublé durant le démarrage lent, en secondes) sont non définies jusqu'à ce qu'elles soient établies comme décrit ci-dessous. Si l'envoyeur est prêt à envoyer des données quand il n'a pas encore un échantillon de délai d'aller-retour, la valeur de X est réglée à s octets par seconde, pour la taille de segment s, le temporisateur Pas de rétroaction est réglé à expirer après deux secondes, et tld est réglé à 0 (ou à -1, les deux sont acceptables). À réception de la première mesure de délai d'aller-retour (par exemple, après le premier paquet de rétroaction ou l'échange SYN de l'établissement de connexion, ou provenant d'une connexion précédente [RFC2140]), tld est réglé à l'heure courante, et le taux de transmission permis, X, est réglé au taux initial, spécifié par W_init/R, pour W_init fondé sur la [RFC3390] :

taux_initial = W_init/R; W_init = min(4*MSS, max(2*MSS, 4380)).

Dans le calcul de W_init, au lieu d'utiliser la taille maximum de segment (MSS, *Maximum Segment Size*) l'envoyeur TFRC DEVRAIT utiliser la taille maximum de segment à utiliser pour le délai initial d'aller-retour des données, si il est connu par l'envoyeur TFRC quand X est initialisé.

Pour répondre au paquet de rétroaction initial, cela remplace l'étape (4) du paragraphe 4.3 ci-dessous.

L'Appendice B explique pourquoi la valeur initiale du temporisateur Pas de rétroaction de TFRC est réglée à deux secondes, au lieu de la valeur initiale recommandée de trois secondes pour le temporisateur de retransmission de TCP dans la [RFC2988].

4.3 Comportement de l'envoyeur à la réception d'un paquet de rétroaction

L'envoyeur connaît son taux d'envoi permis actuel, X, et tient une estimation du délai d'aller-retour courant R. L'envoyeur tient aussi X_recv_set qui est un petit ensemble des valeurs récentes de X_recv (normalement seulement deux valeurs).

Initialisation : X_recv_set est d'abord initialisé à contenir un seul élément, de valeur Infini. (À titre de question spécifique de mise en œuvre, X_recv_set PEUT être initialisé à un grand nombre au lieu de Infini, par exemple, au plus grand entier facilement représentable.)

Quand un paquet de rétroaction est reçu de l'envoyeur au temps t_now, l'heure actuelle en secondes, les actions suivantes DOIVENT être effectuées.

1) Calculer un nouvel échantillon de délai d'aller-retour : R_sample = (t_now - t_recvdata) - t_delay.
Comme décrit au paragraphe 3.2.2, t_delay donne le temps écoulé chez le receveur.

2) Mettre à jour l'estimation du délai d'aller-retour :
Si aucune rétroaction n'a été reçue avant { R = R_sample; }
Autrement { R = q*R + (1-q)*R_sample; }

TFRC n'est pas sensible à la valeur précise de la constante de filtre q, mais une valeur par défaut de 0,9 est RECOMMANDÉE.

3) Mettre à jour l'intervalle de fin de temporisation : RTO = max(4*R, 2*s/X)

4) Mettre à jour le taux d'envoi permis comme suit. Cette procédure utilise les variables t_mbi et recv_limit :
t_mbi : intervalle maximum de retard de 64 secondes.
recv_limit : limite du taux d'envoi calculé à partir de X_recv_set.

Cette procédure utilise aussi les procédures Maximize X_recv_set() et Update X_recv_set(), qui sont définies ci-dessous.

Procédure pour mettre à jour le taux d'envoi permis :

```
Si (l'intervalle entier couvert par le paquet de rétroaction était un intervalle limité en données) {
    Si (le paquet de rétroaction rapporte un nouvel événement de perte ou une augmentation du taux d'événements de perte
p) {
       Diviser par deux les entrées dans X_recv_set ;
       X_recv = 0,85 * X_recv ;
       Maximiser X_recv_set() ;
       recv_limit = max (X_recv_set) ;
    } Autrement {
       Maximiser X_recv_set() ;
       recv_limit = 2 * max (X_recv_set) ;
    }
} Autrement {                                          // comportement normal
    Mettre à jour X_recv_set();
    recv_limit = 2 * max (X_recv_set);
}
Si (p > 0) {                                           // phase d'évitement d'encombrement
    Calculer X_Bps en utilisant l'équation de débit TCP.
    X = max(min(X_Bps, recv_limit), s/t_mbi) ;
} Autrement si (t_now - tld ≥ R) {                     // démarrage lent initial
    X = max(min(2*X, recv_limit), initial_rate) ;
    tld = t_now ;
}
```

5) Si la réduction d'oscillation est utilisée, calculer le taux de transmission instantané, X_inst, selon le paragraphe 4.5.

6) Réinitialiser le temporisateur Pas de rétroaction à expirer après RTO secondes.

La procédure pour maximiser X_recv_set conserve une seule valeur, la plus grande de X_recv_set et de nouveau X_recv.

```
Maximiser X_recv_set() :
    Ajouter X_recv à X_recv_set;
    Supprimer la valeur initiale Infinie de X_recv_set, si elle en fait encore partie.
    Régler l'horodatage du plus grand élément à l'heure courante ;
    Supprimer tous les autres éléments.
```

La procédure pour mettre à jour X_recv_set conserve un ensemble de valeurs de X_recv avec les horodatages provenant des deux plus récents délais d'aller-retour.

```
Mise à jour de X_recv_set() :
    Ajouter X_recv à X_recv_set ;
    Supprimer de X_recv_set les valeurs plus anciennes que deux délais d'aller-retour.
```

Définition d'un intervalle limité en données : On définit un envoyeur comme limité en données chaque fois qu'il n'envoie pas autant qu'il lui est permis d'envoyer. On définit un intervalle comme "intervalle limité en données" si l'envoyeur était limité en données sur l'intervalle *entier* ; le paragraphe 8.2.1 discute les questions de mise en œuvre pour un envoyeur qui détermine si un intervalle était un intervalle limité en données. Le terme "intervalle limité en données" est utilisé dans la première condition "si" de l'étape (4), qui empêche un envoyeur d'avoir à réduire son taux d'envoi par suite d'un paquet de rétroaction rapportant le taux de réception de périodes de données limitées.

Par exemple, considérons un envoyeur qui envoie à son plein taux permis, sauf qu'il envoie les paquets en paires, plutôt que d'envoyer chaque paquet aussitôt qu'il peut. Un tel envoyeur est considéré comme limité en données une partie du temps, parce qu'il n'est pas toujours en train d'envoyer les paquets aussitôt qu'il peut. Cependant, considérant un intervalle qui couvre la transmission de cet envoyeur d'au moins deux paquets de données ; un tel intervalle ne satisfait pas la définition d'intervalle limité en données parce que l'envoyeur n'était pas limité en données *sur l'intervalle entier*.

Si le paquet de rétroaction rapporte un taux de réception X_recv de zéro (c'est-à-dire, le premier paquet de rétroaction)

l'envoyeur ne considère pas que l'intervalle entier couvert par le paquet de rétroaction était un intervalle limité en données.

X_recv_set et le premier paquet de rétroaction : parce que X_recv_set est initialisé avec un seul élément, de valeur Infini, recv_limit est réglé à Infini pour les deux premiers délais d'aller-retour de la connexion. Par suite, le taux d'envoi n'est pas limité par le taux de réception durant cette période. Cela évite le problème d'un taux d'envoi limité par la valeur de X_recv du premier paquet de rétroaction.

Intervalle couvert par un paquet de rétroaction : comment l'envoyeur détermine t-il la période couverte par un paquet de rétroaction ? Ceci est discuté plus en détails au paragraphe 8.2. En général, le receveur va envoyer un paquet de rétroaction une fois par délai d'aller-retour ; donc normalement, l'envoyeur va être capable de déterminer exactement la période couverte par le paquet de rétroaction en cours à partir du précédent paquet de rétroaction. Cependant, dans des cas où le précédent paquet de rétroaction est perdu ou quand le receveur envoie un paquet de rétroaction de façon précoce parce que il a détecté un paquet perdu ou marqué ECN, l'envoyeur va devoir estimer l'intervalle couvert par le paquet de rétroaction. Comme spécifié au paragraphe 6.2, chaque paquet de rétroaction envoyé par le receveur couvre un délai d'aller-retour, pour l'estimation du délai d'aller-retour R_m tenu par receveur R_m secondes avant l'envoi du paquet de rétroaction.

Réponse à une perte durant un intervalle limité en données : dans TFRC, après le démarrage lent initial, l'envoyeur met toujours à jour le taux de transmission calculé, X_Bps, après la réception d'un paquet de rétroaction, et le taux d'envoi permis, X, est toujours limité par X_Bps. Cependant, durant un intervalle limité en données, quand le taux d'envoi réel est généralement en dessous de X_Bps, le taux d'envoi est quand même limité par recv_limit, déduit de X_recv_set. Si l'envoyeur est limité en données, éventuellement avec un taux d'envoi variant d'un délai d'aller-retour à l'autre, et subit des pertes, alors on diminue l'entrée dans X_recv_set afin de réduire le taux d'envoi permis.

L'envoyeur peut détecter un événement de perte durant une période de données limitées soit à partir d'une rétroaction explicite du receveur, soit à partir d'une augmentation rapportée du taux d'événements de perte. Quand l'envoyeur reçoit un paquet de rétroaction rapportant un tel événement de perte dans un intervalle limité en données, l'envoyeur limite l'augmentation permise du taux d'envoi durant l'intervalle limité en données.

Phase initiale de démarrage lent : noter que quand p=0, l'envoyeur n'a pas encore appris d'événement de perte, et l'envoyeur est dans la phase initiale de démarrage lent. Dans cette phase initiale de démarrage lent, l'envoyeur peut approximativement doubler le taux d'envoi à chaque délai d'aller-retour jusqu'à ce qu'une perte survienne. Le terme initial_rate à l'éape (4) donne un minimum de taux d'envoi permis durant le démarrage lent du taux d'envoi permis initial.

On note que si l'envoyeur est limité en données durant le démarrage lent, ou si la connexion est limitée par la bande passante du chemin, l'envoyeur n'est alors pas nécessairement capable de doubler son taux d'envoi à chaque délai d'aller-retour ; le taux d'envoi de l'envoyeur est limité à au plus deux fois le taux de réception passé, ou au plus initial_rate, selon ce qui est le plus grand. Ceci est similaire au comportement de TCP, où le taux d'envoi est limité par le taux de paquets d'accusé de réception entrants ainsi que par la fenêtre d'encombrement. Donc, dans le démarrage lent de TCP, pour le cas le plus agressif de receveur TCP qui accuse réception de chaque paquet de données, le taux d'envoi de l'envoyeur TCP est limité à au plus deux fois le taux de ces paquets d'accusé de réception entrants.

Taux d'envoi minimum permis : le terme s/t_mbi assure que quand p > 0, il est permis à l'envoyeur d'envoyer au moins un paquet toutes les 64 secondes.

4.4 Expiration du temporisateur Pas de rétroaction

Ce paragraphe spécifie la réponse de l'envoyeur à un temporisateur Pas de rétroaction. Le temporisateur Pas de rétroaction pourrait expirer à cause d'une période de repos ou à cause de paquets de données ou de rétroaction éliminés dans le réseau.

On utilise la variable recover_rate *(taux de récupération)*. Si l'envoyeur TFRC a été au repos depuis que le temporisateur Pas de rétroaction a été établi, le taux d'envoi permis n'est pas réduit en dessous de recover_rate. Pour le présent document, recover_rate est réglé à initial_rate (spécifié au paragraphe 4.2). De futures mises à jour de la présente spécification pourront explorer d'autres valeurs possibles pour recover_rate.

Si le temporisateur Pas de rétroaction arrive à expiration, l'envoyeur DOIT effectuer les actions suivantes :

1) Diviser par deux le taux d'envoi permis.
Si le temporisateur Pas de rétroaction expire quand l'envoyeur a fait au moins une mesure de RTT, le taux d'envoi permis est réduit en modifiant X_recv_set comme décrit dans le pseudo code ci-dessous (incluant l'élément (2)). Dans le cas général, le taux d'envoi est limité à au plus deux fois X_recv. Modifier X_recv_set limite le taux d'envoi, mais permet quand même à l'envoyeur de faire le démarrage lent, doublant son taux d'envoi à chaque RTT, si les messages de

rétroaction ne rapportent pas des pertes.

Si l'envoyeur a été au repos depuis que ce temporisateur Pas de rétroaction a été établi et si X_recv est inférieur à recover_rate, alors le taux d'envoi permis n'est pas divisé par deux, et X_recv_set n'est pas changé. Cela assure que le taux d'envoi permis n'est pas réduit à moins de la moitié de recover_rate par suite d'une période de repos.

Dans le cas général, le taux d'envoi permis est divisé par deux en réponse à l'expiration du temporisateur Pas de rétroaction. Les détails, dans le pseudo code ci-dessous, dépendent de si l'envoyeur est en démarrage lent, en évitement d'encombrement limité par X_recv, ou est en évitement d'encombrement limité par l'équation de débit.

```
X_recv = max (X_recv_set);
Si (l'envoyeur n'a pas d'échantillon de RTT, n'a pas reçu de rétroaction du receveur, et n'a pas été au repos depuis
l'établissement du temporisateur Pas de rétroaction ) {
                                                    // On n'a pas encore de X_Bps ou de recover_rate.
                                                    // Diviser par deux le taux d'envoi permis.
   X = max(X/2, s/t_mbi) ;
} Autrement si (((p>0 && X_recv < recover_rate) ou
    (p==0 && X < 2 * recover_rate)) et
     l'envoyeur a été au repos depuis que le temporisateur Pas de rétroaction a été établi) {
// Ne pas diviser par deux le taux d'envoi permis.
  Ne rien faire.
} Autrement si (p==0) {                             // On n'a pas encore X_Bps
                                                    // Diviser par deux le taux d'envoi permis.
   X = max(X/2, s/t_mbi);
} Autrement si (X_Bps > 2*X_recv)) {
                                                    // 2*X_recv limite déjà le taux d'envoi.
                                                    // Diviser par deux le taux d'envoi permis.
   Update_Limits(X_recv;)
} Autrement {
                                                    // Le taux d'envoi était limité par X_Bps, et non par X_recv.
                                                    // Diviser par deux le taux d'envoi permis.
   Update_Limits(X_Bps/2);
}
```

Le terme s/t_mbi limite le retard d'un paquet toutes les 64 secondes.

La procédure Update_Limits() utilise la variable timer_limit pour la limite sur le taux d'envoi calculé à partir de l'expiration du temporisateur Pas de rétroaction, comme suit :

```
Update_Limits(timer_limit) :
   Si (timer_limit < s/t_mbi)
      timer_limit = s/t_mbi ;
   Remplacer le contenu de X_recv_set par le seul élément timer_limit/2 ;
       Recalculer X comme dans l'étape (4) du paragraphe 4.3 ;
```

2) Redémarrer le temporisateur Pas de rétroaction pour qu'il expire après max(4*R, 2*s/X) secondes.

Si l'envoyeur a été limité en données mais non au repos depuis l'établissement du temporisateur Pas de rétroaction, il est possible que le temporisateur Pas de rétroaction ait expiré parce que des paquets de données ou de rétroaction ont été éliminés dans le réseau. Dans ce cas, le temporisateur Pas de rétroaction est le mécanisme de reprise sur défaillance pour que l'envoyeur détecte ces pertes, de façon similaire au temporisateur de retransmission de TCP.

Noter que quand l'envoyeur arrête d'envoyer des données pendant un temps, le receveur va arrêter d'envoyer des rétroactions. Quand le temporisateur Pas de rétroaction de l'envoyeur expire, l'envoyeur pourrait utiliser la procédure ci-dessus pour limiter le taux d'envoi. Si l'envoyeur recommence ensuite à envoyer, X_recv_set va être utilisé pour limiter le taux de transmission, et le comportement de démarrage lent va se produire jusqu'à ce que le taux de transmission atteigne X_Bps.

La réduction du taux d'envoi permis de l'envoyeur TFRC après l'expiration du temporisateur Pas de rétroaction est similaire à la réduction de la fenêtre d'encombrement, cwnd, de TCP, après chaque RTO secondes d'une période de repos, pour TCP avec validation de fenêtre d'encombrement [RFC2861].

4.5 Réduction des oscillations

Pour réduire les oscillations dans le délai de mise en file d'attente et de taux d'envoi dans des environnements avec un faible degré de multiplexage statistique à la liaison encombrée, il est RECOMMANDÉ que l'envoyeur réduise le taux de transmission lorsque le délai de mise en file d'attente (et donc le RTT) augmente. Pour ce faire, l'envoyeur tient R_sqmean, une estimation à long terme de la racine carrée du RTT, et modifie son taux d'envoi selon que la racine carrée de R_sample, le plus récent échantillon de RTT, diffère de l'estimation à long terme. L'estimation à long terme R_sqmean est établie comme suit :

```
Si aucune rétroaction n'a été reçue avant {
   R_sqmean = sqrt(R_sample) ;
} Autrement {
   R_sqmean = q2*R_sqmean + (1-q2)*sqrt(R_sample);
}
```

Donc, R_sqmean donne la pondération exponentielle du déplacement de la moyenne de la racine carrée des échantillons de RTT. La constante q2 devrait être réglée comme q, la constante utilisée dans l'estimation du délai d'aller-retour R. Une valeur de 0,9 comme valeur par défaut pour q2 est RECOMMANDÉE.

Quand sqrt(R_sample) est supérieur à R_sqmean, alors le délai d'aller-retour courant est supérieur à la moyenne à long terme, ce qui implique que le délai de mise en file d'attente est probablement en augmentation. Dans ce cas, le taux de transmission est diminué pour minimiser les oscillations du délai de mise en file d'attente.

L'envoyeur obtient le taux de transmission permis de base, X, comme décrit à l'étape (4) du paragraphe 4.3. Il calcule alors un taux de transmission instantané modifié X_inst, comme suit :

```
X_inst = X * R_sqmean / sqrt(R_sample);
Si (X_inst < s/t_mbi)
   X_inst = s/t_mbi ;
```

Comme on utilise des racines carrées, il y a généralement seulement une différence modérée entre le taux de transmission instantané X_inst et le taux de transmission permis X. Par exemple, dans un cas un peu extrême, quand l'échantillon courant de RTT R_sample est deux fois la moyenne à long terme, alors sqrt(R_sample) va être en gros 1,44 fois R_sqmean, et le taux de transmission permis va être réduit d'un facteur d'à peu près 0,7.

On note que cette modification pour réduire le comportement oscillatoire n'est pas toujours nécessaire, en particulier si le degré de multiplexage statistique dans le réseau est élevé. On note aussi que la modification pour réduire le comportement oscillatoire pourrait causer des problèmes aux connexions où le délai d'aller-retour n'est pas fortement corrélé avec le délai de mise en file d'attente (par exemple, dans certaines liaisons sans fils sur des chemins avec de fréquents changements d'acheminement, etc.). Cependant, cette modification DEVRAIT être mise en œuvre parce que elle donne un meilleur comportement de TCP dans certains environnements avec un faible niveau de multiplexage. Les performances de cette modification sont illustrées au paragraphe 3.1.3 de [FHPW00]. Si elle n'est pas mise en œuvre, on DEVRAIT utiliser une valeur très faible de la pondération q pour le délai d'aller-retour moyen.

4.6 Programmation des transmissions de paquets

Comme TFRC est fondé sur le débit, et comme les systèmes d'exploitation ne peuvent normalement pas programmer avec précision les événements, il est nécessaire d'être opportuniste dans l'envoi des paquets de données afin que le taux moyen correct soit conservé en dépit d'une programmation grossière ou irrégulière du système d'exploitation. Pour aider à maintenir le taux d'envoi moyen correct, l'envoyeur TFRC PEUT envoyer des paquets avant leur temps d'envoi nominal.

De plus, la programmation des transmissions de paquets contrôle la sporadicité permise des envoyeurs après une période de repos ou de données limitées. L'envoyeur TFRC PEUT accumuler des "crédits" d'envoi pour des temps d'envoi passé non utilisés ; cela permet à l'envoyeur TFRC d'envoyer une salve de données après une période de repos ou de données limitées. Pour comparer avec TCP, TCP peut envoyer jusqu'à un délai d'aller-retour de paquets dans une seule salve, mais jamais plus. Par exemple, des salves de paquets peuvent être envoyées par TCP quand un ACK arrive pour accuser réception d'une fenêtre de données, ou quand un envoyeur limité en données a soudain une fenêtre de données à envoyer après un délai de presque un délai d'aller-retour.

Pour limiter la sporadicité, une mise en œuvre de TFRC DOIT empêcher des salves de taille arbitraire. Cette limite DOIT

être inférieure ou égale à un délai d'aller-retour de paquets. Une mise en œuvre de TFRC PEUT limiter les salves à moins d'un délai d'aller-retour de paquets. De plus, une mise en œuvre de TFRC PEUT utiliser la régulation fondée sur le débit pour lisser les salves.

Comme exemple spécifique de mise en œuvre, une boucle d'envoi pourrait calculer l'intervalle inter paquets correct, t_ipi, comme suit : t_ipi = s/X_inst ;

Soit t_now l'heure actuelle et i un entier naturel, i = 0, 1, ..., avec t_i l'heure nominale d'envoi pour le ième paquet. Alors, l'heure nominale d'envoi t_(i+1) va se déduire par récurrence comme:

t_0 = t_now,
t_(i+1) = t_i + t_ipi.

Pour les envoyeurs TFRC à qui il est permis d'accumuler des crédits d'envoi pour les temps d'envoi non utilisés sur les dernières T secondes, l'envoyeur va pouvoir utiliser les temps d'envoi nominaux non utilisés t_j pour t_j < maintenant - T, pour T réglé au délai d'aller-retour.

5. Calcul du taux d'événements de perte (p)

Obtenir une mesure précise et stable du taux d'événements de perte est de première importance pour TFRC. La mesure des taux de perte est effectuée chez le receveur, sur la base de la détection des paquets perdus ou marqués d'après les numéros de séquence des paquets arrivants. On décrit ce processus avant de décrire le reste du protocole de receveur. Si le receveur n'a pas encore détecté un paquet perdu ou marqué, alors le receveur ne calcule pas le taux d'événements de perte, mais rapporte un taux d'événements de perte de zéro.

5.1 Détection des paquets perdus ou marqués

TFRC suppose que tous les paquets contiennent un numéro de séquence qui est incrémenté de un pour chaque paquet envoyé. Pour les besoins de la présente spécification, il est EXIGÉ que si un paquet perdu est retransmis, la retransmission reçoive un nouveau numéro de séquence qui est le dernier dans la séquence de transmission, et non le même numéro de séquence que le paquet perdu. Si un protocole de transport exige qu'il soit retransmis avec le numéro de séquence original, alors le concepteur du protocole de transport doit préciser comment distinguer les paquets retardés des paquets retransmis et comment détecter les retransmissions perdues.

Le receveur tient une structure de données qui garde trace des paquets arrivés et des paquets manquants. Pour les besoins de la présente spécification, on suppose que la structure de données consiste en une liste de paquets qui sont arrivés avec l'horodatage du receveur quand chaque paquet a été reçu. En pratique, cette structure de données va normalement être mémorisée dans une représentation plus compacte, mais ceci est spécifique de la mise en œuvre.

La perte d'un paquet est détectée par l'arrivée d'au moins NDUPACK paquets avec un numéro de séquence supérieur à celui du paquet perdu, pour NDUPACK réglé à 3. L'exigence pour les NDUPACK paquets suivants est la même qu'avec TCP, et est de rendre TFRC plus robuste en présence de réarrangements. À la différence de TCP, si un paquet arrive tard (après l'arrivée de NDUPACK paquets suivants) dans TFRC, le paquet tardif peut remplir le trou dans les enregistrements de réception de TFRC, et le receveur peut recalculer le taux d'événements de perte. De futures versions de TFRC pourraient faire que l'exigence sur les NDUPACK paquets suivants soit adaptable sur la base des réarrangements de paquets subis, mais un tel mécanisme ne fait pas partie de la spécification actuelle.

Pour une connexion à capacité ECN, un paquet marqué est détecté comme un événement d'encombrement aussitôt qu'il arrive, sans avoir à attendre l'arrivée des paquets suivants.

Si un paquet marqué ECN est précédé d'un paquet qui pourrait être perdu, alors le premier événement d'encombrement détecté commence avec le paquet perdu. Par exemple, si le receveur reçoit un paquet de données avec le numéro de séquence n-1, suivi par un paquet de données non marqué avec le numéro de séquence n+1, et un paquet de données marqué avec le numéro de séquence n+2, alors le receveur détecte un événement d'encombrement quand il reçoit le paquet marqué n+2. Le premier événement d'encombrement détecté commence avec le paquet perdu n. Les lignes directrices du paragraphe 5.2 sont utilisées pour déterminer si les paquets perdus et marqués appartiennent au même événement de perte ou à des événements de perte séparés.

5.2 Traduction d'historique de perte en événements de perte

TFRC exige que la fraction de perte soit robuste à plusieurs paquets perdus ou marqués consécutifs dans le même événement de perte. Ceci est similaire à TCP, qui (normalement) n'effectue qu'une division par deux de la fenêtre d'encombrement durant un seul RTT. Donc, le receveur doit transposer l'historique des pertes de paquets en un enregistrement d'événement de perte, où un événement de perte est un ou plusieurs paquets perdus ou marqués dans un RTT. Pour effectuer cette transposition, le receveur doit savoir le RTT à utiliser, et ceci est fourni périodiquement par l'envoyeur, normalement comme informations de contrôle portées sur un paquet de données. TFRC n'est pas sensible à la façon dont sont faites les mesures de RTT envoyées au receveur, mais il est RECOMMANDÉ d'utiliser le RTT calculé par l'envoyeur, R, (voir le paragraphe 4.3) à cette fin.

Pour déterminer si un paquet perdu ou marqué devrait commencer un nouvel événement de perte ou être compté au titre d'un événement de perte existant, on doit comparer les numéros de séquence et horodatages des paquets qui arrivent chez le receveur. Pour un paquet marqué, S_nouveau, son heure de réception, T_nouveau, peut être notée directement. Pour un paquet perdu, on peut interpoler pour déduire "l'heure d'arrivée" nominale. On suppose que :
S_perte est le numéro de séquence d'un paquet perdu.
S_avant est le numéro de séquence du dernier paquet arrivé, avant toute arrivée de paquet avec un numéro de séquence au-dessus de S_perte, avec un numéro de séquence en dessous de S_perte.
S_après est le numéro de séquence du premier paquet à arriver après S_avant avec un numéro de séquence au dessus de S_perte.
S_max est le plus grand numéro de séquence.

Donc, S_avant < S_perte < S_après ≤ S_max.

T_perte est l'heure d'arrivée nominale estimée pour le paquet perdu.
T_avant est l'heure de réception de S_avant.
T_après est l'heure de réception de S_après.

Note que T_avant < T_après.

Pour un paquet perdu, S_perte, on peut interpoler son "heure d'arrivée" nominale chez le receveur à partir des heures d'arrivée de S_avant et S_après. Donc :

T_perte = T_avant + ((T_après - T_avant) * (S_perte - S_avant)/(S_après - S_avant)) ;

Pour traiter le retour à zéro du numéro de séquence, soit S_MAX = 2^b, où b est la longueur en bits des numéros de séquence dans une mise en œuvre donnée. Dans ce cas, on peut interpoler l'heure d'arrivée T_perte comme suit :

T_perte = T_avant + (T_après - T_avant) * Dist(S_perte, S_avant)/Dist(S_après, S_avant)

où

Dist(S_A, S_B) = (S_A + S_MAX - S_B) % S_MAX

Si le paquet perdu S_vieux a été déterminé comme ayant démarré le précédent événement de perte, et si on a juste déterminé que S_nouveau a été perdu, alors on interpole les heures nominales d'arrivée de S_vieux et S_nouveau, appelées T_vieux et T_nouveau, respectivement.

Si T_vieux + R ≥ T_nouveau, alors S_nouveau fait partie de l'événement de perte existant. Autrement, S_nouveau est le premier paquet dans un nouvel événement de perte.

5.3 Taille d'un intervalle de perte

Après la détection du premier événement de perte, le receveur divise l'espace de séquence en intervalles de perte. Si un intervalle de perte, A, est déterminé comme ayant commencé avec le paquet de numéro de séquence S_A et si le prochain intervalle de perte, B, a commencé par le paquet de numéro de séquence S_B, alors le nombre de paquets dans l'intervalle de perte A est donné par (S_B - S_A). Donc, l'intervalle de perte A contient tous les paquets transmis par l'envoyeur en commençant par le premier paquet transmis dans l'intervalle de perte A et se terminant par, non inclus, le premier paquet transmis dans l'intervalle de perte B.

L'intervalle de perte en cours I_0 est défini comme l'intervalle de perte contenant l'événement de perte le plus récent. Si cet événement de perte a commencé par le paquet de numéro de séquence S_A, et si S_C est le plus fort numéro de séquence

reçu jusqu'à présent, alors la taille de I_0 est S_C - S_A + 1. Par exemple, si l'intervalle de perte courant consiste en un seul paquet marqué ECN, alors S_A == S_C, et la taille de l'intervalle de perte est un.

5.4 Intervalle de perte moyen

Pour calculer le taux d'événements de perte, p, on calcule d'abord l'intervalle de perte moyen. Cela se fait en utilisant un filtre qui pondère les n plus récents intervalles d'événement de perte de telle façon que les changements de taux d'événements de perte mesurés soient lissés. Si le receveur n'a pas encore vu un paquet perdu ou marqué, le receveur ne calcule alors pas l'intervalle de perte moyen.

Les pondérations w_0 à w_(n-1) sont calculées comme :

```
Si (i < n/2) {
    w_i = 1 ;
} Autrement {
    w_i = 2 * (n-i)/(n+2) ;
}
```

Donc, si n = 8, les valeurs de w_0 à w_7 sont : 1,0, 1,0, 1,0, 1,0, 0,8, 0,6, 0,4, 0,2

La valeur n pour le nombre d'ntervalles de perte utilisé pour calculer le taux d'événements de perte détermine la vitesse de TFRC pour répondre aux changements du niveau d'encombrement. Il est RECOMMANDÉ de régler cette valeur à 8. TFRC NE DEVRAIT PAS utiliser des valeurs de n supérieures à 8 pour un trafic qui peut se trouver en compétition avec TCP dans l'Internet mondial. Au minimum, un fonctionnement sûr avec des valeurs de n supérieures à 8 exigerait un léger changement aux mécanismes de TFRC pour inclure une réponse plus stricte à deux délais d'aller-retour ou plus avec de fortes pertes de paquets.

Quand on calcule l'intervalle de perte moyen, on a besoin de décider si on inclut l'intervalle de perte actuel. On n'inclut l'intervalle de perte actuel que si il est suffisamment grand pour augmenter l'intervalle de perte moyen.

Soient les plus récents intervalles de perte I_0 à I_k, où I_0 est l'intervalle de perte courant. Si il y a eu au moins n intervalles de perte, alors k est réglé à n ; autrement, k est le nombre maximum d'intervalles de perte vus jusqu'à présent. On calcule l'intervalle de perte moyen I_moyen comme suit :

```
I_tot0 = 0 ;
I_tot1 = 0 ;
W_tot = 0 ;
pour (i = 0 à k-1) {
   I_tot0 = I_tot0 + (I_i * w_i) ;
   W_tot = W_tot + w_i ;
}
pour (i = 1 à k) {
   I_tot1 = I_tot1 + (I_i * w_(i-1)) ;
}
I_tot = max(I_tot0, I_tot1) ;
I_moyen = I_tot/W_tot ;
```

Le taux d'événements de perte, p est simplement : p = 1 / I_moyen ;

5.5 Décompte d'historique

Comme décrit au paragraphe 5.4, quand il y a eu au moins n intervalles de perte, le plus récent intervalle de perte reçoit seulement 1/(0,75*n) de la pondération totale dans le calcul de l'intervalle de perte moyen, sans considération de la taille du plus récent intervalle de perte. Ce paragraphe décrit un mécanisme FACULTATIF de décompte d'historique, exposé plus en détails dans [FHPW00a] et [W00], qui permet au receveur TFRC d'ajuster les pondérations, en se concentrant plus sur le poids relatif du plus récent intervalle de perte, quand celui-ci est plus de deux fois plus grand que l'intervalle de perte moyen calculé.

Pour porter le décompte d'historique, on associe un facteur de décompte, DF_i, à chaque intervalle de perte, L_i, pour i > 0, où chaque facteur de décompte est un nombre à virgule flottante. Le dispositif de décompte tient l'historique cumulatif des décomptes pour chaque intervalle de perte. Au début, les valeurs de DF_i dans le dispositif de décompte sont initialisées à 1 :

```
pour (i = 0 à n) {
   DF_i = 1 ;
}
```

Le compte d'historique utilise aussi un facteur général de compte, DF, qui est aussi un nombre à virgule flottante, qui est aussi initialisé à 1. On montre d'abord comment les facteurs de compte sont utilisés dans le calcul de l'intervalle de perte moyen, et on décrit ensuite, dans la suite de ce paragraphe, comment les facteurs de compte sont modifiés dans le temps.

Comme décrit au paragraphe 5.4, l'intervalle de perte moyen est calculé en utilisant les n précédents intervalles de perte I_1, ..., I_n et l'intervalle de perte courant I_0. Le calcul de l'intervalle de perte moyen en utilisant les facteurs de compte est une simple modification de la procédure du paragraphe 5.4, comme suit :

```
I_tot0 = I_0 * w_0 ;
I_tot1 = 0 ;
W_tot0 = w_0 ;
W_tot1 = 0 ;
pour (i = 1 à n-1) {
   I_tot0 = I_tot0 + (I_i * w_i * DF_i * DF);
   W_tot0 = W_tot0 + w_i * DF_i * DF;
}
pour (i = 1 à n) {
   I_tot1 = I_tot1 + (I_i * w_(i-1) * DF_i);
   W_tot1 = W_tot1 + w_(i-1) * DF_i;
}
p = min(W_tot0/I_tot0, W_tot1/I_tot1);
```

Le facteur général de compte, DF, est mis à jour sur chaque paquet arrivé comme suit. D'abord, le receveur calcule la moyenne pondérée de I_moyen des intervalles de perte I_1, ..., I_n :

```
I_tot = 0 ;
W_tot = 0 ;
pour (i = 1 à n) {
   W_tot = W_tot + w_(i-1) * DF_i;
   I_tot = I_tot + (I_i * w_(i-1) * DF_i) ;
}
I_moyen = I_tot / W_tot ;
```

Cette moyenne pondérée de I_moyen est comparée à I_0, la taille de l'intervalle de perte courant. Si I_0 est supérieur à deux fois I_moyen, alors le nouvel intervalle de perte est considérablement plus grand que l'ancien, et le facteur général de compte, DF, est mis à jour pour diminuer le poids relatif sur les vieux intervalles, comme suit :

```
si (I_0 > 2 * I_moyen) {
   DF = 2 * I_moyen/I_0 ;
   si (DF < THRESHvieux) {
      DF = THRESHvieux;
   }
} autrement {
   DF = 1 ;
}
```

Une valeur non zéro pour THRESHvieux assure que les plus vieux intervalles de perte provenant d'une période antérieure de fort encombrement ne sont pas décomptés entièrement. On recommande un THRESHvieux de 0,25. Noter qu'avec chaque nouveau paquet arrivé, I_0 va encore décroître, et le facteur de compte DF va être mis à jour.

Quand un nouvel événement de perte survient, l'intervalle courant glisse de I_0 à I_1, l'intervalle de perte I_i glisse de l'intervalle I_(i+1), et l'intervalle de perte I_n est oublié. Le facteur de compte DF précédent doit être incorporé dans le dispositif de décompte. Parce que DF_i porte le facteur de compte associé à l'intervalle de perte I_i, le dispositif DF_i doit aussi être décalé. Ceci est fait comme suit :

```
pour (i = 1 à n) {
```

```
    DF_i = DF * DF_ i;
}
pour (i = n-1 à 0 étape -1) {
    DF_(i+1) = DF_i ;
}
I_0 = 1 ;
DF_0 = 1 ;
DF = 1 ;
```

Cela complète la description du mécanisme facultatif de décompte d'historique. On souligne que ce mécanisme est FACULTATIF, dont le seul objet est de permettre à TFRC de répondre un peu plus rapidement à la soudaine absence d'encombrement, comme représenté par un long intervalle de perte courant.

6. Protocole du receveur des données

Le receveur envoie périodiquement des messages de rétroaction à l'envoyeur. Les paquets de rétroaction DEVRAIENT normalement être envoyés au moins une fois par RTT, sauf si l'envoyeur envoie à un taux de moins d'un paquet par RTT, auquel cas un paquet de rétroaction DEVRAIT être envoyé pour chaque paquet de données reçu. Un paquet de rétroaction DEVRAIT aussi être envoyé chaque fois qu'un nouvel événement de perte est détecté sans attendre la fin d'un RTT, et chaque fois qu'un paquet de données déclassé est reçu qui supprime un événement de perte de l'historique.

Si l'envoyeur transmet à un taux élevé (de nombreux paquets par RTT) il peut y avoir des avantages à envoyer des messages de rétroaction périodiques plus d'une fois par RTT car cela permet des réponses plus rapides à des mesures changeantes de RTT et plus de résilience à la perte de paquet de rétroaction.

Si le receveur envoyait k paquets de rétroaction par RTT, pour k > 1, l'étape (4) du paragraphe 6.2 va être modifiée pour régler le temporisateur de rétroaction à expirer après R_m/k secondes. Cependant, chaque paquet de rétroaction va encore rapporter le taux du receveur sur le dernier RTT, et non sur une fraction d'un RTT. Dans le présent document, on ne spécifie pas les modifications qui pourraient être exigées d'un receveur qui envoie plus d'un paquet de rétroaction par RTT. On note qu'il y a peu d'intérêt à envoyer un grand nombre de messages de rétroaction par RTT.

6.1 Comportement du receveur à la réception d'un paquet de données

Quand un paquet de données est reçu, le receveur effectue les étapes suivantes :

1) Ajouter le paquet à l'historique des paquets.

2) Vérifier si c'est fait : si le nouveau paquet résulte en la détection d'un nouvel événement de perte, ou si aucun paquet de rétroaction n'a été envoyé quand le temporisateur de rétroaction a expiré, passer à l'étape 3. Autrement, aucune action n'a besoin d'être effectuée (sauf si l'optimisation du paragraphe suivant est utilisée) de sorte qu'on quitte la procédure. Une optimisation FACULTATIVE pourrait vérifier si l'arrivée du paquet a causé un trou à boucher dans l'historique de paquets, et par conséquent, deux intervalles de perte ont été fusionnés en un seul. Si c'est le cas, le receveur pourrait aussi envoyer immédiatement une rétroaction. Les effets d'une telle optimisation sont normalement supposés être minimes.

3) Calculer p : soit p_prev la valeur précédente de p. On calcule la nouvelle valeur de p comme décrit à la Section 5.

4) Faire expirer le temporisateur de rétroaction : si p > p_prev, causer l'expiration du temporisateur de rétroaction et effectuer les actions décrites au paragraphe 6.2.

Si p ≤ p_prev et si aucun paquet de rétroaction n'a été envoyé quand le temporisateur de rétroaction a expiré, causer l'expiration du temporisateur de rétroaction et effectuer les actions décrites au paragraphe 6.2. Si p ≤ p_prev et si un paquet de rétroaction a été envoyé quand le temporisateur de rétroaction a expiré, aucune action n'a besoin d'être effectuée.

6.2 Expiration du temporisateur de rétroaction

Quand le temporisateur de rétroaction expire chez le receveur, l'action à entreprendre dépend de si des paquets de données ont été reçus depuis l'envoi des dernières rétroactions.

Pour la miène expiration du temporisateur de rétroaction, soit S_m le numéro de séquence maximum d'un paquet chez le

receveur, jusqu'à présent, et soit R_m la valeur de la mesure du RTT incluse dans le paquet S_m. Comme décrit au paragraphe 3.2.1, R_m est l'estimation la plus récente de l'envoyeur du délai d'aller-retour, comme rapporté dans les paquets de données. Si des paquets de données ont été reçus depuis l'envoi des précédentes rétroactions, le receveur effectue les étapes suivantes :

1) Calculer le taux moyen d'événements de perte en utilisant l'algorithme décrit à la Section 5.

2) Calculer le taux de réception mesuré, X_recv, sur la base des paquets reçus dans les R_(m-1) secondes précédentes. Ceci est effectué que le temporisateur de rétroaction ait expiré à son heure normale ou plus tôt à cause d'un nouveau paquet perdu ou marqué (c'est-à-dire, l'étape (3) du paragraphe 6.1). Dans le cas normal, quand le receveur envoie seulement un paquet de rétroaction par délai d'aller-retour et que le temporisateur de rétroaction n'expire pas plus tôt à cause d'un nouveau paquet perdu, alors l'intervalle de temps depuis l'expiration du temporisateur de rétroaction va être R_(m-1) secondes. On note que quand le temporisateur de rétroaction expire précocement à cause d'un nouveau paquet perdu ou marqué, l'intervalle de temps depuis la dernière expiration du temporisateur de rétroaction est probablement inférieur à R_(m-1) secondes. Pour faciliter la mise en œuvre, si l'intervalle de temps depuis la dernière expiration du temporisateur de rétroaction n'est pas R_(m-1) secondes, le taux de réception PEUT être calculé sur un intervalle plus long, l'intervalle de temps revenant à la plus récente expiration du temporisateur de rétroaction qui était au moins R_(m-1) secondes avant.

3) Préparer et envoyer un paquet de rétroaction contenant les informations décrites au paragraphe 3.2.2.

4) Redémarrer le temporisateur de rétroaction pour qu'il expire après R_m secondes.

Noter que la règle 2) ci-dessus donne une valeur minimum pour le taux de réception mesuré X_recv de un paquet par délai d'aller-retour. Si l'envoyeur est limité à un taux d'envoi de moins d'un paquet par délai d'aller-retour, cela va être dû au taux d'événements de perte, et non à une limite imposée par le taux de réception mesuré chez le receveur.

Si aucun paquet de données n'a été reçu depuis d'envoi de la dernière rétroaction, aucun paquet de rétroaction n'est alors envoyé, et le temporisateur de rétroaction est redémarré à expirer après R_m secondes.

6.3 Initialisation du receveur

Le receveur est initialisé par le premier paquet de données qui arrive chez le receveur. Soit i le numéro de séquence de ce paquet.
Quand le premier paquet est reçu :
- Régler p = 0.
- Régler X_recv = 0.
- Préparer et envoyer un paquet de rétroaction.
- Régler le temporisateur de rétroaction à expirer après R_i secondes.

Si le premier paquet de données ne contient pas une estimation R_i du délai d'aller-retour, alors le receveur envoie un paquet de rétroaction pour chaque paquet de données arrivant jusqu'à ce qu'arrive un paquet de données contenant une estimation du délai d'aller-retour.

Si l'envoyeur utilise un horodatage grossier qui s'incrémente tous les quarts de délai d'aller-retour, un temporisateur de rétroaction n'est pas nécessaire, et la procédure suivante, de la RFC 4342, est utilisée pour déterminer quand envoyer des messages de rétroaction.

- Chaque fois que le receveur envoie un message de rétroaction, le receveur règle une variable locale last_counter à la plus grande valeur reçue de compteur de fenêtre depuis que le dernier message de rétroaction a été envoyé, si des paquets de données ont été reçus depuis l'envoi du dernier message de rétroaction.

- Si le receveur reçoit un paquet de données avec une valeur de compteur de fenêtre supérieure ou égale au dernier compteur + 4, alors le receveur envoie un nouveau paquet de rétroaction. ("Supérieur" et "plus grand" sont mesurés dans un espace circulaire de compteur de fenêtre.)

6.3.1 Initialisation de l'historique de pertes après le premier événement de perte

Ce paragraphe décrit la procédure qui DOIT être utilisée pour initialiser l'historique des pertes après le premier événement de perte.

Le nombre de paquets jusqu'à la première perte ne peut pas être utilisé directement pour calculer le taux d'envoi permis, car le taux d'envoi change rapidement durant cette période. TFRC suppose que le taux de données correct après la première perte est la moitié du taux d'envoi maximum avant la survenance de la perte. TFRC fait une approximation de ce taux cible, X_cible, par la valeur maximum de X_recv jusqu'alors. (Pour le démarrage lent, pour un délai d'aller-retour particulier, le taux d'envoi de l'envoyeur est généralement deux fois le taux de réception du receveur pour les données envoyées sur le précédent délai d'aller-retour.)

Après la première perte, au lieu d'initialiser le premier intervalle de perte au nombre de paquets envoyés jusqu'à la première perte, le receveur TFRC calcule l'intervalle de perte qui serait requis pour produire le taux de données X_cible, et utilise cet intervalle de perte synthétique pour initier le mécanisme d'historique des pertes.

TFRC fait cela en trouvant une valeur, p, pour laquelle l'équation de débit du paragraphe 3.1 donne un taux d'envoi dans les 5 % de X_cible, étant donné le délai d'aller-retour R, et le premier intervalle de perte est alors réglé à 1/p. Si le receveur connaît la taille de segment, s, utilisée par l'envoyeur, alors le receveur PEUT utiliser l'équation de débit pour X ; autrement, le receveur PEUT mesurer le taux de réception en paquets par seconde au lieu d'octets par seconde pour cela, et utiliser l'équation de débit pour X_pps. (La tolérance de 5 % est introduite simplement parce que l'équation de débit est difficile à inverser, et on veut réduire les coûts du calcul numérique de p.)

Une attention particulière est nécessaire pour initialiser le premier intervalle de perte quand le premier paquet de données est perdu ou marqué. Quand le premier paquet de données est perdu dans TCP, l'envoyeur TCP retransmet le paquet après l'expiration du temporisateur de retransmission. Si le premier paquet de données de TCP est marqué ECN, l'envoyeur TCP rétablit le temporisateur de retransmission, et n'envoie un nouveau paquet de données que quand le temporisateur de retransmission expire ([RFC3168], paragraphe 6.1.2). Pour TFRC, si le premier paquet de données est perdu ou marqué ECN, alors le premier intervalle de perte consiste en l'intervalle nul sans paquet de données. Dans ce cas, la longueur de l'intervalle de perte pour cet intervalle de perte (nul) DEVRAIT être réglée à donner un taux d'envoi similaire à celui de TCP, comme spécifié dans le paragraphe suivant.

Quand le premier intervalle de perte TFRC est nul, ce qui signifie que le premier paquet de données est perdu ou marqué ECN, afin de suivre le comportement de TCP, TFRC veut que le taux d'envoi permis soit 1 paquet tous les deux délais d'aller-retour, ou de façon équivalente, 0,5 paquet par RTT. Donc, le receveur TFRC calcule l'intervalle de perte qui serait requis pour produire le taux cible X_cible de 0,5/R paquets par seconde, pour le délai d'aller-retour R, et utilise cet intervalle de perte synthétique pour le premier intervalle de perte. Le receveur TFRC utilise 0,5/R paquets par seconde comme valeur minimum pour X_cible quand il initialise le premier intervalle de perte.

On note que même si le receveur TFRC rapporte un intervalle de perte synthétique après le premier événement de perte, le receveur TFRC rapporte quand même le taux de réception mesuré X_recv, comme spécifié au paragraphe 6.2.

7. Variantes fondées sur l'envoyeur

Dans une variante fondée sur l'envoyeur de TFRC, le receveur utilise une livraison fiable pour envoyer des informations sur les pertes de paquets à l'envoyeur, et l'envoyeur calcule le taux de perte de paquets et le taux de transmission acceptable.

Le principal avantage d'une variante fondée sur l'envoyeur de TFRC est que l'envoyeur n'a pas à se fier au calcul du receveur du taux de perte de paquet. Cependant, avec l'exigence d'une livraison fiable des informations de perte du receveur à l'envoyeur, un TFRC fondé sur l'envoyeur aurait des contraintes plus strictes sur le protocole de transport dans lequel il est incorporé.

À l'opposé, la variante fondée sur le receveur de TFRC spécifiée dans le présent document est robuste à la perte de paquets de rétroaction, et n'exige donc pas la livraison fiable des paquets de rétroaction. Elle est aussi mieux adaptée pour les applications qui désirent déplacer autant que possible la charge de travail du serveur au client.

Les RFC 4340 et RFC 4342 spécifient ensemble le CCID 3 de DCCP, qui peut être utilisé comme variante fondée sur l'envoyeur de TFRC. Dans CCID 3, chaque paquet de rétroaction provenant du receveur contient une option Intervalles de perte, qui rapporte les longueurs des plus récents intervalles de perte. Les paquets de rétroaction peuvent aussi inclure l'option Vecteur d'accusé de réception, qui permet à l'envoyeur de déterminer exactement quels paquets ont été éliminés ou marqués et de vérifier les informations rapportées dans les options Intervalles de perte. L'option Vecteur d'accusé de réception peut aussi inclure des échos de nom occasionnel ECN, ce qui permet à l'envoyeur de vérifier le rapport du receveur sur la réception de paquet de données non marqués. L'option Vecteur d'accusé de réception permet à l'envoyeur de voir par lui-même quels paquets de données ont été perdus ou marqués ECN, de déterminer les intervalles de perte, et de calculer le taux d'événements de perte. La Section 9 de la RFC 4342 discute les questions de vérification par l'envoyeur des

informations rapportées par le receveur.

8. Questions de mise en œuvre

Le présent document a spécifié le mécanisme de contrôle d'encombrement de TFRC, à l'usage des protocoles d'applications et de transport. Cette section mentionne brièvement quelques questions de mise en œuvre.

8.1 Calcul de l'équation de débit

Pour t_RTO = 4*R et b = 1, l'équation de débit du paragraphe 3.1 peut être exprimée comme suit :

$$X\_Bps = \frac{s}{R * f(p)}$$

pour

$$f(p) = \sqrt{(2*p/3)} + (12*\sqrt{(3*p/8)} * p * (1+32*p^2)).$$

Une recherche de tableau pourrait être utilisée pour la fonction f(p).

Beaucoup des multiplications (par exemple, q et 1-q pour le délai moyen d'aller-retour, un facteur 4 pour intervalle de fin de temporisation) sont ou pourraient être de puissances de deux, et pourraient être mises en œuvre par une simple opération de décalage.

8.2 Comportement de l'envoyeur à la réception d'un paquet de rétroaction

Ce paragraphe discute les questions de mise en œuvre du comportement d'envoyeur quand un paquet de rétroaction est reçu, d'après le paragraphe 4.3.

8.2.1 Déterminer si un intervalle était de données limitées

Quand un paquet de rétroaction est reçu, l'envoyeur doit déterminer si l'intervalle entier couvert par ce paquet de rétroaction était d'une période de données limitées. Ce paragraphe discute une mise en œuvre possible pour que l'envoyeur détermine si l'intervalle couvert par un paquet de rétroaction était de période de données limitées.

Si les paquets de rétroaction rapportent tous l'horodatage du dernier paquet de données reçu, soit t_nouveau l'horodatage rapporté par ce paquet de rétroaction. Comme tous les paquets de rétroaction couvrent un intervalle d'au moins un délai d'aller-retour, il est suffisant à l'envoyeur de déterminer si il y a eu un instant dans la période [t_vieux, t_nouveau] où l'envoyeur n'a pas été limité en données, pour R l'estimation de l'envoyeur du délai d'aller-retour, et pour t_vieux réglé à t_nouveau - R. (Cette procédure estime l'intervalle couvert par le paquet de rétroaction, plutôt que de le calculer exactement. Elle nous paraît bonne.)

Le pseudo code pour déterminer si l'envoyeur était limité en données sur l'intervalle entier couvert dans un paquet de rétroaction est donné ci-dessous. Les variables NonLimité1 et NonLimité2 représentent toutes deux les temps où l'envoyeur n'était *pas* limité en données.

```
Initialisation :
   NonLimité1 = NonLimité2 = t_nouveau = t_prochain = 0 ;
   t_now = heure actuelle ;

Après l'envoi d'un segment :
   Si (l'envoyeur a envoyé tout ce qu'il lui est permis d'envoyer) {
// L'envoyeur n'est pas limité en données à cet instant.
      Si NonLimité1 ≤ t_nouveau
// But : NonLimité1 > t_nouveau.
         NonLimité1 = t_now ;
      Autrement si (NonLimité2 ≤ t_prochain)
// But : NonLimité2 > t_prochain.
         NonLimité2 = t_now ;
```

```
}
```

Quand un paquet de rétroaction est reçu, cet intervalle est il limité en données :

```
t_nouveau = horodatage rapporté dans le paquet de rétroaction.
t_vieux = t_nouveau - R.                    // variable locale
t_prochain = t_now ;
Si ((t_vieux < NonLimité1 ≤ t_nouveau) ou
   (t_vieux < NonLimité2 ≤ t_nouveau))
   Ce n'était pas un intervalle limité en données ;
Autrement
   C'était un intervalle limité en données.
Si (NonLimité1 ≤ t_nouveau && NonLimité2 > t_nouveau)
   NonLimité1 = NonLimité2 ;
```

Les heures de transmission se réfèrent à la transmission d'un ou de segments à la couche en dessous.

Entre les paquets de rétroaction, [t_vieux, t_nouveau] donne l'intervalle de transmission estimé être couvert par le plus récent paquet de rétroaction, et t_next donne un temps d'au moins un délai d'aller-retour supérieur à t_nouveau. Le prochain paquet de rétroaction peut être supposé couvrir en gros l'intervalle [t_nouveau, t_prochain] (sauf si le receveur envoie le paquet de rétroaction plus tôt parce qu'il rapporte un nouvel événement de perte). le but est que NonLimite1 sauvegarde un temps non limité en données dans [t_nouveau, t_prochain], si il y en a un, et que NonLimité2 sauvegarde un temps non limité en données après t_prochain.

Quand un paquet de rétroaction a été reçu, si NonLimité1 ou NonLimité2 est dans l'intervalle de temps couvert par le paquet de rétroaction, alors l'intervalle n'est pas un intervalle limité en données ; l'envoyeur n'était pas limité en données au moins une fois durant cet intervalle de temps. Si ni NonLimité1 ni NonLimité2 ne sont dans l'intervalle de temps couvert par un paquet de rétroaction, alors l'envoyeur est supposé avoir été limité en données sur cet intervalle de temps.

On note que cette procédure est une heuristique, et dans certains cas, l'envoyeur pourrait ne pas déterminer correctement si l'envoyeur était limité en données sur l'intervalle entier couvert par le paquet de rétroaction. Cette heuristique ne traite pas les possibles complications de remise en ordre.

Cela nous semble acceptable. Afin d'améliorer sa précision pour identifier si l'intervalle entier couvert par un paquet de rétroaction était un intervalle limité en données, l'envoyeur pourrait sauvegarder plus de temps NonLimité.

Dans certaines mises en œuvre de TFRC, l'envoyeur envoie des horodatages à gros grain qui s'incrémentent tous les quarts de délai d'aller-retour, et le paquet de rétroaction rapporte le plus grand numéro de séquence valide reçu jusqu'alors au lieu de rapporter l'horodatage du dernier paquet reçu. Dans ce cas, l'envoyeur peut conserver l'état par paquet pour déterminer t_nouveau (l'heure à laquelle le paquet acquitté a été envoyé) ou l'envoyeur peut estimer t_nouveau à partir de son estimation du délai d'aller-retour et du temps écoulé t_delay rapporté par le paquet de rétroaction.

8.2.2 Tenue de X_recv_set

Pour réduire la complexité de la maintenance de X_recv_set, il est suffisant de limiter X_recv_set à au plus N=3 éléments. Dans ce cas, la procédure Mise à jour de X_recv_set() va être modifiée comme suit :

```
Mise à jour de X_recv_set() :
   Ajout de X_recv à X_recv_set ;
   Suppression des valeurs de X_recv_set plus vielles que deux délais d'aller-retour.
   Garder seulement les N valeurs les plus récentes.
```

Conserver au plus *deux* éléments dans X_recv_set serait suffisant pour que l'envoyeur sauvegarde une vieille valeur de X_recv d'avant une période de données limitées, et pour permettre à l'envoyeur de n'être pas limité par le premier paquet de rétroaction après une période de repos (rapportant un taux de réception de un paquet par délai d'aller-retour). Cependant, il est *possible* que conserver au plus deux éléments dans X_recv_set ne donne pas d'aussi bonnes performances que de conserver au plus trois éléments. Conserver trois éléments dans X_recv_set permettrait à X_recv_set de contenir X_recv valeurs provenant de deux paquets de rétroaction successifs, plus une valeur plus récente de X_recv provenant d'un événement de perte.

8.3 Envoi de paquets avant leur heure d'envoi nominale

Ce paragraphe discute d'un possible mécanisme de programmation pour un envoyeur dans un système d'exploitation à la granularité grossière du temps (du paragraphe 4.6).

Soit t_gran être la granularité du temporisateur de programmation du système d'exploitation. Soit t_ipi l'intervalle inter paquets, comme spécifié au paragraphe 4.6. Si le système d'exploitation a une granularité grossière de temporisateur ou ne peut pas par ailleurs prendre en charge de courts intervalles t_ipi, alors soit l'envoyeur TFRC va être restreint à un taux d'envoi d'au plus 1 paquet toutes les t_gran secondes, soit l'envoyeur TFRC doit pouvoir envoyer de courtes salves de paquets. En plus de permettre à l'envoyeur d'accumuler des crédits d'envoi pour les temps d'envoi passés non utilisés, cela peut être utile pour permettre à l'envoyeur d'envoyer un paquet avant son temps d'envoi programmé, comme décrit au paragraphe suivant.

Un paramètre, t_delta, peut être utilisé pour permettre qu'un paquet soit envoyé avant son temps d'envoi nominal. Considérons une application qui passe au repos et demande une reprogrammation pour les temps t_i = t_(i-1) + t_ipi, pour t_(i-1) l'heure d'envoi du paquet précédent. Quand l'application est reprogrammée, elle vérifie l'heure actuelle, t_now. Si (t_now > t_i – t_delta) alors le paquet i est envoyé. Quand l'heure nominale d'envoi, t_i, du prochain paquet est calculée, il se peut que déjà t_now > t_i - t_delta. Dans un tel cas, le paquet va être envoyé immédiatement.

Afin d'envoyer au plus un paquet avant son heure d'envoi nominale, et ne jamais envoyer un paquet plus qu'un délai d'aller-retour avant son heure d'envoi nominale, le paramètre t_delta devrait être réglé comme suit :

t_delta = min(t_ipi, t_gran, rtt)/2 ;

(La granularité de programmation t_gran est de 10 ms sur certains vieux systèmes Unix.)

Par exemple, considérons un flux TFRC avec un taux d'envoi permis X de 10 paquets par délai d'aller-retour (PPR), un délai d'aller-retour de 100 ms, un système avec une granularité de programmation t_gran de 10 ms, et la capacité d'accumuler des crédits d'envoi inutilisés pour un délai d'aller-retour. Dans ce cas, t_ipi est 1 ms. Il va être permis à l'envoyeur TFRC d'envoyer des paquets 0,5 ms avant leur temps d'envoi nominal, et de sauvegarder des crédits d'envoi inutilisés pendant 100 ms. La granularité de programmation de 10 ms ne va pas significativement affecter les performances de la connexion.

Dans un exemple différent, considérons un flux TFRC avec une granularité de programmation supérieure au délai d'aller-retour, par exemple, avec un délai d'aller-retour de 0,1 ms et un système avec une granularité de programmation de 1 ms, et la capacité d'accumuler les crédits d'envoi inutilisés pendant un délai d'aller-retour. Il va être permis à l'envoyeur TFRC de sauvegarder les crédits d'envoi inutilisés pendant 0,1 ms. Si la granularité de programmation *n'a pas* affecté la réponse de l'envoyeur à un paquet de rétroaction entrant, alors l'envoyeur TFRC va être capable d'envoyer un RTT de données (comme déterminé par le taux d'envoi permis) chaque RTT, en réponse aux paquets de rétroaction entrants. Dans ce cas, la granularité de programmation grossière ne va pas réduire significativement le taux d'envoi, mais le taux d'envoi va être sporadique, avec un délai d'aller-retour des données envoyées en réponse à chaque paquet de rétroaction.

Cependant, les performances seraient différentes dans ce cas, si la granularité de programmation du système d'exploitation affectait la réponse de l'envoyeur aux paquets de rétroaction ainsi que la programmation générale de l'envoyeur. Dans ce cas, les performances de l'envoyeur seraient sévèrement limitées par la granularité de programmation supérieure au délai d'aller-retour, l'envoyeur étant capable d'envoyer un RTT de données, au taux d'envoi permis, au plus une fois par ms. Cette restriction du taux d'envoi est une conséquence inévitable de la sporadicité permise d'au plus un délai d'aller-retour de données.

8.4 Calcul de l'intervalle de perte moyen

Le calcul de l'intervalle de perte moyen du paragraphe 5.4 implique la multiplication par la pondération w_0 à w_(n-1), qui pour n = 8 est : 1,0, 1,0, 1,0, 1,0, 0,8, 0,6, 0,4, 0,2.

Avec une perte mineure de lissage, il serait possible d'utiliser des pondérations qui soient des puissances de deux ou des sommes de puissances de deux, par exemple, 1,0, 1,0, 1,0, 1,0, 0,75, 0,5, 0,25, 0,25.

8.5 Mécanisme facultatif de décompte de l'historique

Le mécanisme facultatif de décompte de l'historique décrit au paragraphe 5.5 est utilisé dans le calcul du taux de perte moyen. Le mécanisme de décompte de l'historique n'est invoqué que quand il y a eu un intervalle inhabituellement long

sans perte de paquet. Pour un fonctionnement plus efficace, le facteur de décompte, DF_i, pourrait être restreint à être une puissance de deux.

9. Changements par rapport à la RFC 3448

9.1 Vue d'ensemble des changements

Cette section résume les changements par rapport à la RFC 3448. À haut niveau, le principal changement est l'ajout de mécanismes pour traiter le cas d'un envoyeur limité en données. Le présent document permet aussi explicitement à l'envoyeur TFRC d'accumuler jusqu'à un délai d'aller-retour de crédits d'envoi inutilisés, et par suite d'envoyer une salve de paquets si des données arrivent de l'application en salve après une période de données limitées. Cette question n'était pas explicitement traitée dans la RFC 3448.

Le présent document change la RFC 3448 pour incorporer de plus forts taux d'envoi initiaux de TCP de la RFC 3390. Le présent document change aussi la RFC 3448 pour permettre l'utilisation de l'horodatage à gros grain de la RFC 4342 sur les paquets de données au lieu d'un horodatage plus fin.

Les autres changement traitent des cas marginaux impliquant le démarrage lent, la réponse quand le premier paquet de données est éliminé, et autres. Ce document incorpore aussi les éléments des errata à la RFC 3448.

Cette section n'est pas normative ; le texte normatif est dans les paragraphes cités.

9.2 Changements par paragraphe

Au paragraphe 4.1, l'estimation de la taille moyenne de segment : le paragraphe 4.1 a été modifié pour donner un algorithme spécifique qui pourrait être utilisé pour estimer la taille moyenne de segment.

Au paragraphe 4.2, mise à jour du taux d'envoi initial : dans la RFC 3448, le taux d'envoi initial était de deux paquets par délai d'aller-retour. Dans le présent document, le taux d'envoi initial peut être jusqu'à quatre paquets par délai d'aller-retour, suivant la RFC 3390. Le taux d'envoi initial a été changé pour être en termes de la taille de segment s, non en termes de MSS.

Au paragraphe 4.2 on dit maintenant que le temps doublé la dernière fois (TLD, *Time Last Doubled*) durant le démarrage lent, peut être initialisé à 0 ou -1. Le paragraphe 4.2 a aussi été précisé pour dire que les mesures de RTT ne viennent pas seulement des paquets de rétroaction ; elles pourraient aussi venir d'autres endroits, comme l'échange SYN.

Au paragraphe 4.3, réponse aux paquets de rétroaction : le paragraphe 4.3 a été modifié pour changer la façon dont le taux de réception est utilisé pour limiter le taux d'envoi permis de l'envoyeur, en utilisant l'ensemble de valeurs de taux de réception des deux derniers délais d'aller-retour, et en initialisant l'ensemble de valeurs de taux de réception à une grande valeur.

Le plus grand taux d'envoi initial au paragraphe 4.2 est de peu d'usage si le receveur envoie un paquet de rétroaction après la réception du premier paquet, et l'envoyeur, en réponse, réduit le taux d'envoi permis à au plus deux paquets par RTT, ce qui serait deux fois le taux de réception. À cause du changement du traitement par l'envoyeur du taux de réception, l'envoyeur ne réduit maintenant pas le taux d'envoi permis à deux fois le taux de réception rapporté en réponse au premier paquet de rétroaction.

Durant une période de données limitées, l'envoyeur sauvegarde le taux de réception rapporté de juste avant la période de données limitées, si il est supérieur au taux de réception durant la période de données limitées. L'envoyeur réduit aussi les valeurs sauvegardées dans X_recv_set en réponse à une perte durant une période de données limitées. L'Appendice C discute de cette réponse plus en détails.

Au paragraphe 4.4, réponse à une période de repos : suivant le paragraphe 5.1 de la [RFC4342], le présent document spécifie que quand le taux d'envoi est réduit après une période de repos qui couvre la période depuis l'établissement du temporisateur Pas de rétroaction, le taux d'envoi permis n'est pas réduit en dessous du taux d'envoi initial. (Au paragraphe 4.4, la variable recover_rate est réglée au taux d'envoi initial.)

Au paragraphe 4.4, une correction provenant de [RFC3448Err]. La RFC 3448 avait un texte contradictoire sur si l'envoyeur divise par deux son taux d'envoi après *deux* délais d'aller-retour sans recevoir de rapport de rétroaction, ou après *quatre* délais d'aller-retour. Le présent document précise que l'envoyeur divise par deux son taux d'envoi après quatre délais d'aller-retour sans recevoir de rapport de rétroaction [RFC3448Err].

Au paragraphe 4.4, précision pour le démarrage lent : le paragraphe 4.4 a été précisé pour spécifier qu'à l'expiration du temporisateur Pas de rétroaction, si p = 0, X_Bps ne peut pas être utilisé, parce que l'envoyeur n'a pas encore de valeur pour X_Bps. Le paragraphe 4.4 a aussi été précisé pour vérifier le cas où l'envoyeur n'a pas encore un échantillon de RTT, mais a envoyé un paquet depuis l'établissement du temporisateur Pas de rétroaction.

Au paragraphe 4.6 : crédits pour les temps d'envoi non utilisés : le paragraphe 4.6 a été précisé pour dire que l'envoyeur TFRC peut accumuler jusqu'à un RTT de crédits pour des temps d'envoi non utilisés. Le paragraphe 4.6 a aussi été réécrit pour préciser ce qui est spécification et ce qui est mise en œuvre.

Au paragraphe 5.4, une précision : le paragraphe 5.4 a été modifié pour préciser que le calcul du receveur de l'intervalle moyen de perte quand le receveur n'a pas encore vu d'intervalle de perte.

Au paragraphe 5.5, une correction : le paragraphe 5.5 a été corrigé pour dire que l'intervalle de perte I_0 inclut tous les paquets transmis, incluant les paquets perdus et marqués (comme défini au paragraphe 5.3 dans la définition générale des intervalles de perte).

Au paragraphe 5.5, correction provenant de [RFC3448Err] : une ligne du paragraphe 5.5 a été changée de "pour (i = 1 à n) { DF_i = 1; } " en "pour (i = 0 à n) { DF_i = 1; } [RFC3448Err].

Au paragraphe 5.5, décompte d'historique : THRESHvieux, la limite inférieure du paramètre DF, décompte d'historique, a été changée de 0,5 à 0,25, pour permettre plus de décompte d'historique quand l'intervalle courant est long.

À la Section 6, paquets de rétroaction multiples : la Section 6 contient maintenant une discussion plus longue des procédures si le receveur envoie plusieurs paquets de rétroaction à chaque délai d'aller-retour.

Au paragraphe 6.3, initialisation du temporisateur de rétroaction : le paragraphe 6.3 spécifie maintenant l'initialisation par le receveur du temporisateur de rétroaction si le premier paquet de données reçu n'a pas une estimation du délai d'aller-retour.

Au paragraphe 6.3, horodatage à gros grain : le paragraphe 6.3 a été modifié pour incorporer, en option, un horodatage à gros grain de l'envoyeur qui s'incrémente tous les quarts de délai d'aller-retour, au lieu d'un horodatage à grain plus fin. Cela suit la RFC 4342.

Au paragraphe 6.3.1, après le premier événement de perte : le paragraphe 6.3.1 dit maintenant que pour l'initialisation de l'historique des pertes après le premier événement de perte, le receveur utilise le taux de réception maximum obtenu jusqu'alors, au lieu du taux de réception du dernier délai d'aller-retour.

Au paragraphe 6.3.1, si le premier paquet de données est éliminé : le paragraphe 6.3.1 contient maintenant une spécification pour l'initialisation de l'historique des pertes si le premier paquet de données envoyé est perdu ou marqué ECN.

À la Section 7, variantes fondées sur l'envoyeur : la Section 7 discute les variantes fondées sur l'envoyeur qui ont été étendues, en référence à la RFC 4342.

10. Considérations sur la sécurité

TFRC n'est pas en soi un protocole de transport, mais un mécanisme de contrôle d'encombrement qui est destiné à être utilisé en conjonction avec un protocole de transport. Donc, la sécurité doit principalement être considérée dans le contexte d'un protocole de transport spécifique et de ses mécanismes d'authentification.

Les mécanismes de contrôle de l'encombrement peuvent éventuellement être exploités pour créer un déni de service. Cela peut arriver par de fausses rétroactions. Donc, tout protocole de transport qui utilise TFRC devrait veiller à s'assurer que les rétroactions ne sont acceptées que du receveur des données. Le mécanisme précis pour réaliser cela va cependant dépendre du protocole de transport lui-même.

De plus, les mécanismes de contrôle de l'encombrement peuvent éventuellement être manipulés par un receveur avide qui souhaite recevoir plus que sa juste part de la bande passante du réseau. Un receveur pourrait faire cela en prétendant avoir reçu des paquets qui, en fait, ont été perdus à cause de l'encombrement. Les défenses possibles contre un tel receveur incluraient normalement une forme de nom occasionnel que le receveur doit renvoyer à l'envoyeur pour prouver la réception. Cependant, les détails d'un tel nom occasionnel vont dépendre du protocole de transport, et en particulier de si le

protocole de transport est fiable ou non.

On s'attend à ce que les protocoles qui incorporent ECN dans TFRC voudront aussi incorporer la rétroaction du receveur à l'envoyeur en utilisant le nom occasionnel ECN [RFC3540]. Le nom occasionnel ECN est une modification à ECN qui protège l'envoyeur de la dissimulation accidentelle ou malveillante de paquets marqués. Là encore, les détails d'un tel nom occasionnel vont dépendre du protocole de transport, et ne sont pas traités dans le présent document.

10.1 Considérations sur la sécurité pour TFRC dans DCCP

TFRC est actuellement utilisé dans l'identifiant 3 de contrôle d'encombrement (CCID 3, *Congestion Control ID 3*) [RFC4342] du protocole de contrôle d'encombrement de datagrammes (DCCP, *Datagram Congestion Control Protocol*) [RFC4340]. La Section Considérations sur la sécurité de la [RFC4340] (Section 18) discute certaines des questions de sécurité de DCCP, incluant les vérifications de validité des numéros de séquence pour protéger contre la capture des connexions. La Section 18 de la RFC 4340 discute aussi des mécanismes de DCCP pour limiter le potentiel impact des attaques de déni de service.

La RFC 4342 spécifie l'usage de TFRC dans CCID 3. La RFC 4342 inclut des discussions extensives des mécanismes que l'envoyeur peut utiliser pour vérifier les informations envoyées par le receveur. Quand ECN est utilisé avec CCID 3, le receveur retourne les informations de nom occasionnel ECN à l'envoyeur, pour lui permettre de vérifier les informations envoyées par le receveur. Quand ECN n'est pas utilisé, la Section 9 de la RFC 4342 discute la façon dont l'envoyeur pourrait quand même utiliser diverses techniques qui pourraient saisir que le receveur a fait une erreur en rapportant l'encombrement. Cependant, comme le déclare la RFC 4342, ceci n'est pas aussi robuste ou non intrusif que la vérification fournie par le nom occasionnel ECN.

11. Remerciements

Nous tenons à remercier de leurs retours et discussions sur le contrôle d'encombrement fondés sur une équation une large gamme de personnes, incluant les membres du groupe de recherche sur la diffusion groupée fiable, le groupe de travail sur le transport de diffusion groupée fiable, et le groupe de recherche sur le bout en bout. Merci à Dado Colussi, Gorry Fairhurst, Ladan Gharai, Wim Heirman, Eddie Kohler, Ken Lofgren, Mike Luby, Ian McDonald, Vladimir Moltchanov, Colin Perkins, Michele R., Gerrit Renker, Arjuna Sathiaseelan, Vladica Stanisic, Randall Stewart, Eduardo Urzaiz, Shushan Wen, et Wendy Lee (lhh@zsu.edu.cn) pour leur réactions sur les versions antérieures de ce document, et merci à Mark Allman des ses rétroactions extensives sur l'utilisation de la [RFC3448] pour produire une mise en œuvre fonctionnelle.

Appendice A. Terminologie

Le présent document utilise les termes qui suivent. Les variables de temporisateur (par exemple, t_now, tld) sont supposées être en secondes, avec une résolution du temporisateur d'au moins une milliseconde.

Intervalle limité en données : intervalle dans lequel l'envoyeur est limité en données (n'envoyant pas autant qu'il est autorisé à envoyer) sur l'intervalle entier (paragraphe 4.3).

DF : facteur de décompte pour un intervalle de perte (paragraphe 5.5).

taux_initial : taux d'envoi initial permis.

last_counter : plus grande valeur reçue du compteur de fenêtre (paragraphe 6.3).

n : nombre d'intervalles de perte.

NDUPACK : nombre de dupacks pour déduire une perte (constante) (paragraphe 5.1).

temporisateur Pas de rétroaction : temporisateur côté envoyeur (paragraphe 4).

p : estimation du taux d'événement de perte.

p_prev : valeur précédente de p (paragraphe 6.1).

q : constante de filtre pour le RTT (constante) (paragraphe 4.3).

q2 : constante de filtre pour RTT à long terme (constante) (paragraphe 4.6).

R : délai d'aller-retour de chemin estimé.

R_m : estimation spécifique du délai d'aller-retour de chemin (paragraphe 4.3, Section 6).

R_sample : RTT mesuré de chemin (paragraphe 4.3).

R_sqmean : estimation à long terme de la racine carrée du RTT (paragraphe 4.6).

recover_rate : taux permis pour reprise après une période de repos (paragraphe 4.4).

recv_limit : limite du taux d'envoi calculé à partir de X_recv_set (paragraphe 4.3).

s : taille nominale de paquet en octets.

S : numéro de séquence.

t_delay : délai rapporté entre la réception du dernier paquet chez le receveur et la génération du paquet de rétroaction (paragraphe 3.2.2).

t_delta : paramètre de flexibilité du temps d'envoi (paragraphe 8.3).

t_gran : granularité de programmation du temporisateur du système d'exploitation (constante) (paragraphe 8.3).

t_ipi : intervalle inter paquets pour l'envoi des paquets (paragraphe 4.6).

t_mbi : valeur maximum du RTO de TCP (constante) (paragraphe 4.3).

t_recvdata : horodatage du dernier paquet de données reçu (paragraphe 3.2.2).

timer_limit : limite du taux d'envoi à partir de l'expiration du temporisateur Pas de rétroaction (paragraphe 4.4).

tld : heure du dernier doublement (paragraphe 4.2).

t_now : heure courante (paragraphe 4.3).

t_RTO : RTO estimé de TCP (paragraphe 4.3).

X : taux de transmission permis, limité par le taux de réception.

X_Bps : taux d'envoi calculé en octets par seconde (paragraphe 3.1).

X_pps : taux d'envoi calculé en paquets par seconde (paragraphe 3.1).

X_inst : taux de transmission instantané permis (paragraphe 4.6).

X_recv : taux de réception estimé chez le receveur (paragraphe 3.2.2).

X_recv_set : petit ensemble de valeurs récentes de X_recv (paragraphe 4.3).

X_target : taux d'envoi cible après le premier événement de perte (paragraphe 6.3.1).

W_init : fenêtre TCP initiale (constante) (paragraphe 4.2).

Appendice B. Valeur initiale du temporisation Pas de rétroaction

Pourquoi la valeur initiale du temporisateur Pas de rétroaction de TFRC est elle réglée à deux secondes, au lieu de la valeur initiale recommandée de trois secondes pour le temporisateur de retransmission de TCP, dans la [RFC2988] ? Il n'y a pas

de raison particulière que le temporisateur Pas de rétroaction de TFRC devrait avoir la même valeur initiale que le temporisateur de retransmission de TCP. Le temporisateur de retransmission de TCP est utilisé non seulement pour réduire le taux d'envoi en réponse à l'encombrement, mais aussi pour retransmettre un paquet qui est supposé avoir été éliminé dans le réseau. À l'opposé, le temporisateur Pas de rétroaction de TFRC est seulement utilisé pour réduire le taux d'envoi permis, et non pour déclencher l'envoi d'un nouveau paquet. Par suite, il n'y a pas de danger pour le réseau que la valeur initiale du temporisateur Pas de rétroaction de TFRC soit plus petite que la valeur initiale recommandée pour le temporisateur de retransmission de TCP.

De plus, quand le temporisateur Pas de rétroaction n'a pas encore expiré, TFRC a un mécanisme de contrôle d'encombrement à réponse plus lente que TCP, et l'utilisation par TFRC du taux de réception pour limiter le taux d'envoi est un peu moins précis que l'utilisation par TCP de fenêtre et des temporisations d'accusés de réception, de sorte que le temporisateur Pas de rétroaction est un mécanisme de sécurité particulièrement important pour TFRC. Pour toutes ces raisons, il est parfaitement raisonnable que le temporisateur Pas de rétroaction de TFRC ait une plus petite valeur initiale que le temporisateur de retransmission de TCP.

Appendice C. Réponse aux périodes de repos ou de données limitées

De futurs travaux pourraient explorer d'autres réponses en utilisant le taux de réception durant une période de données limitées, et en répondant à un événement de perte durant une période de données limitées.

En particulier, une RFC expérimentale [RFC2861] spécifie la validation de fenêtre d'encombrement (CWV, *Congestion Window Validation*) pour TCP. Pour cette discussion, on utilise le terme de "TCP standard" pour se référer aux mécanismes de contrôle de l'encombrement de TCP dans les [RFC2581] et [RFC5681]. La [RFC2861] spécifie une réponse différente aux périodes de repos ou de données limitées de celle du TCP standard. Avec la CWV, l'envoyeur TCP divise par deux la fenêtre d'encombrement après chaque RTO durant une période de repos, jusqu'à la fenêtre initiale. De façon similaire, avec CWV l'envoyeur TCP divise par deux la fenêtre d'encombrement à mi-chemin de la taille en cours après chaque RTO durant une période de données limitées.

Le présent document spécifie déjà une réponse TFRC aux périodes de repos qui est similaire à celle de TCP avec validation de fenêtre d'encombrement. Cependant, il ne spécifie pas une réponse TFRC aux périodes limitées en données similaire à celle de CWV. Ajouter un tel mécanisme à TFRC exigerait un changement d'une ligne à l'étape (4) du paragraphe 4.3. En particulier, la réponse de l'envoyeur à un paquet de rétroaction pourrait être changée de :

```
Si (l'intervalle entier couvert par le paquet de rétroaction était un intervalle limité en données) {
    Si (le paquet de rétroaction rapporte un nouvel événement de perte ou une augmentation du taux d'événements de perte
p) {
       Diviser par deux les entrées dans X_recv_set ;
       X_recv = 0,85 * X_recv ;
       Maximiser X_recv_set() ;
       recv_limit = max (X_recv_set) ;
    } Autrement {
       Maximiser X_recv_set() ;
       recv_limit = 2 * max (X_recv_set) ;
    }
}
```

en :

```
Si (l'intervalle entier couvert par le paquet de rétroaction était un intervalle limité en données) {
    Multiplier les vieilles entrées dans X_recv_set par 0,85 ;
    Si (le paquet de rétroaction rapporte un nouvel événement de perte ou une augmentation du taux d'événements de perte
p) {
       Multiplier la nouvelle valeur X_recv par 0,85.
    }
    Maximiser X_recv_set() ;
    recv_limit = 2 * max (X_recv_set) ;
}
```

En particulier, si le taux de réception d'avant une période de données limitées est sauvegardé dans X_recv_set, alors le changement de l'étape (4) ci-dessus multiplierait ce taux de réception par 0,85 chaque fois qu'un paquet de rétroaction est

reçu et que le code ci-dessus est exécuté. Par suite, après quatre délais d'aller-retour successifs d'intervalles limités en données, le taux de réception d'avant la période de données limitées serait réduit de 0,85^4 = 0,52. Donc, ce changement d'une ligne de l'étape (4) du paragraphe 4.3 résulterait en ce que le taux d'envoi permis serait divisé par deux à chaque quatre temps d'aller-retour dans lesquels l'envoyeur a été limité en données. À cause de la nature de X_recv_set, ce mécanisme ne va jamais réduire le taux d'envoi permis en dessous de deux fois le taux de réception le plus récent.

On note que dans le code suggéré ci-dessus, avec un comportement de style CWV en réponse aux intervalles limités en données, on garde : recv_limit = 2 * max (X_recv_set) ; au lieu d'utiliser : recv_limit = max (X_recv_set) à la suite des événements de perte dans les intervalles limités en données. Cette réponse plus souple à un événement de perte est permise parce que le comportement de style CWV lui-même limite les fluctuations rapides de taux d'envoi durant les périodes limitées en données.

C.1 Longues périodes de repos ou de données limitées

Le Tableau 1 résume la réponse de TCP standard [RFC2581], TCP avec validation de fenêtre d'encombrement [RFC2861], TFRC standard [RFC3448], et TFRC révisé (le présent document) en réponse aux longues périodes de repos ou de données limitées. Pour les besoins de cette section, on définit une longue période comme une période d'au moins un RTO.

Protocole	**Longues périodes de repos**	**Longues périodes limitées en données**
TCP standard :	Fenêtre-> initial.	La fenêtre augmente pour chaque cwnd de données.
TCP avec CWV :	Diviser la fenêtre par deux (pas en dessous de cwnd initial).	Réduire le fenêtre à mi chemin à la fenêtre utilisée.
TFRC standard :	Diviser le taux par deux (pas en dessous de 2 pqts/rtt). Un RTT après l'envoi de paquet, le taux est limité par X_recv.	Taux limité à deux fois le taux de réception.
TFRC révisé :	Diviser le teux par deux (pas en dessous du taux initial).	Taux limité à deux fois max (current X_recv, taux de réception avant période de données limitées).

Tableau 1 : Réponse à de longues périodes de repos ou limitée en données

TCP standard après de longues périodes de repos : pour TCP standard, la [RFC2581] spécifie que TCP DEVRAIT régler la fenêtre d'encombrement à pas plus que le fenêtre initiale après une période de repos d'au moins un RTO. (Pour être précis, la RFC 2581 spécifie que l'envoyeur TCP devrait régler cwnd à la fenêtre initiale si l'envoyeur n'a pas envoyé de données dans un intervalle excédant la temporisation de retransmission.)

TCP standard après une longue période limitée en données : TCP standard [RFC2581] ne réduit pas la fenêtre d'encombrement de TCP après une période de données limitées, quand la fenêtre d'encombrement n'est pas pleinement utilisée. TCP standard dans la [RFC2581] utilise "FlightSize", la quantité de données en cours dans le réseau, seulement dans le réglage du vieux seuil de démarrage lent après une fin de temporisation de retransmission. TCP standard n'est pas limité par le mécanisme de temporisation des accusés de réception de TCP durant les périodes de données limitées.

La réponse souple de TCP standard aux périodes de données limitées est assez différente de sa réponse stricte à une période de repos.

TCP avec validation de fenêtre d'encombrement (CWV) après de longues périodes de repos : comme solution de remplacement expérimentale, la [RFC2861] spécifie une réponse plus modérée à une période de repos que celle de TCP standard, où durant une période de repos l'envoyeur TCP divise par deux cwnd après chaque RTO, jusqu'au cwnd initial.

TCP avec validation de fenêtre d'encombrement après une longue période limitée en données : comme solution de remplacement expérimentale, la [RFC2861] spécifie une réponse plus stricte à une période de données limitées que celle de TCP standard, où après chaque RTO secondes d'une période de données limitées, la fenêtre d'encombrement est réduite à la moitié de la fenêtre qui est actuellement utilisée.

La réponse de TCP avec CWV à une période de repos est similaire à sa réponse à une période de données limitées. TCP avec CWV est moins restrictif que TCP standard en réponse à une période de repos, et plus restrictif que TCP standard en réponse à une période de données limitées.

TFRC standard après une longue période de repos : pour TFRC standard, la [RFC3448] spécifie que le taux d'envoi permis est divisé par deux après chaque RTO secondes d'une période de repos. Le taux d'envoi permis n'est pas réduit en dessous de deux paquets par RTT après une période de repos. Après une période de repos, le premier paquet de rétroaction reçu rapporte un taux de réception de un paquet par délai d'aller-retour, et ce taux de réception est utilisé pour limiter le taux

d'envoi. TFRC standard fait effectivement le démarrage lent à partir de ce taux d'envoi permis.

TFRC standard après une longue préiode limitée en données : la [RFC3448] ne distingue pas entre périodes limitées en données et non limitées en données. Par conséquent, le taux d'envoi permis est limité à au plus deux fois le taux de réception durant et après une période de données limitées. Cette réponse est très restrictive, plus que celle de TCP standard ou de TCP avec CWV.

TFRC révisé après une longue période de repos : pour TFRC révisé, le présent document spécifie que le taux d'envoi permis est divisé par deux après chaque RTO secondes d'une période de repos. Le taux d'envoi permis n'est pas réduit en dessous du taux d'envoi initial par suite d'une période de repos. Le premier paquet de rétroaction reçu après la période de repos rapporte un taux de réception de un paquet par délai d'aller-retour. Cependant, l'envoyeur de TFRC révisé n'utilise pas ce taux de réception pour limiter le taux d'envoi. Donc, TFRC révisé diffère de TFRC standard par la limite inférieure utilisée dans la réduction du taux d'envoi, et dans la meilleure réponse au premier paquet de rétroaction reçu après la période de repos.

TFRC révisé après une longue période limitée en données : pour TFRC révisé, le présent document distingue entre période limitée en données et non limitée en données. Comme spécifié au paragraphe 4.3, durant une période de données limitées, TFRC révisé se souvient du taux de réception d'avant le début des périodes de données limitées, et ne réduit pas le taux d'envoi permis en dessous de deux fois ce taux de réception. Ceci est un peu similaire à la réponse de TCP standard, et est assez différent de la réponse très restrictive de TFRC standard à une période de données limitées. Cependant, la réponse de TFRC révisé n'est pas aussi prudente que la réponse de TCP avec validation de fenêtre d'encombrement, où la fenêtre d'encombrement est graduellement réduite jusqu'à la fenêtre réellement utilisée durant une période de données limitées.

On note que pour les mises en œuvre actuelles de TCP, la fenêtre d'encombrement n'est généralement pas augmentée durant une période de données limitées (quand la fenêtre d'encombrement courante n'est pas pleinement utilisée) ([MAF05], paragraphe 5.7). On note qu'il n'y a pas de mécanisme comparable à celui-là dans TFRC révisé.

Récupération après une période de repos ou de données limitées : quand TCP réduit la fenêtre d'encombrement après une période de repos ou de données limitées, TCP peut régler le seuil de vieux démarrage lent, ssthresh, à permettre à l'envoyeur TCP de revenir dans son démarrage lent jusqu'à son vieux taux d'envoi quand la période de repos ou de données limitées est terminée. Cependant, dans TFRC, même quand le taux d'envoi de l'envoyeur TFRC est restreint de deux fois le taux de réception précédent, il en résulte que l'envoyeur est capable de doubler le taux d'envoi d'un délai d'aller-retour au prochain, si c'est permis par l'équation de débit. Donc, TFRC n'a pas besoin d'un mécanisme comme celui du réglage de ssthresh de TCP pour permettre un démarrage lent après une périodes de repos ou de données limitées.

Pour les travaux à venir, une perspective à explorer serait l'ajout des mécanismes de validation de fenêtre d'encombrement pour la réponse de TFRC aux périodes limitées en données. Actuellement, suivant TCP standard, durant les périodes limitées en données TFRC révisé ne limite pas le taux d'envoi permis en fonction du taux de réception.

C.2 Courtes périodes de repos ou de données limitées

Le Tableau 2 résume la réponse de TCP standard [RFC2581], de TCP avec validation de fenêtre d'encombrement [RFC2861], de TFRC standard [RFC3448], et de TFRC révisé (le présent document) en réponse à de courtes périodes de repos ou de données limitées. Pour les besoins de cette section, on définit une courte période comme de moins d'un RTT.

Protocole	**Courte période de repos**	**Courte période limitée en données**
TCP standard :	Envoie une salve de jusqu'à cwnd.	Envoie une salve de jusqu'à cwnd.
TCP with CWV :	Envoie une salve de jusqu'à cwnd.	Envoie une salve de jusqu'à cwnd.
TFRC standard :	?	?
TFRC révisé :	Envoie une salve (de jusqu'à un RTT de crédits d'envoi non utilisés)	Envoie une salve (de jusqu'à un RTT de crédits d'envoi non utilisés).

Tableau 2 : Réponse à une courte période de repos ou de données limitées

Le Tableau 2 montre que TFRC révisé a une réponse similaire à celle de TCP standard et TCP avec CWV à une courte période de repos ou de données limitées. Pour une courte période de repos ou de données limitées, TCP n'est limité que par la taille de la fenêtre d'encombrement non utilisée, et TFRC révisé n'est limité que par le nombre de crédits d'envoi non utilisés (jusqu'à la valeur d'un RTT). Pour TFRC standard, la [RFC3448] ne spécifie pas explicitement le comportement à l'égard des crédits d'envoi non utilisés.

C.3 Périodes de repos ou de données limitées modérées

Le Tableau 3 résume la réponse de TCP standard [RFC2581], de TCP avec validation de fenêtre d'encombrement [RFC2861], de TFRC standard [RFC3448], et de TFRC révisé (le présent document) en réponse à des périodes modérées de repos ou de données limitées. Pour les besoins de cette section, on définit une période modérée comme une période supérieure à un RTT, mais de moins qu'un RTO.

Protocole	**Période modérée de repos**	**Période modérée limitée en données**
TCP standard :	Envoie une salve jusqu'à cwnd.	Envoie une salve jusqu'à cwnd.
TCP avec CWV :	Envoie une salve jusqu'à cwnd.	Envoie une salve jusqu'à cwnd.
TFRC standard :	?	Limité par X_recv.
TFRC révisé :	Envoie une salve (jusqu'à un RTT de crédits d'envoi inutilisés).	Envoie une salve (jusqu'à un RTT de crédits d'envoi inutilisés).

Tableau 3 : Réponse à une période modérée de repos ou de données limitées

Le Tableau 3 montre que TFRC révisé a une réponse similaire à celle de TCP standard et de TCP avec CWV à une période modérée de repos ou de données limitées. Pour une période modérée de repos ou de données limitées, TCP est limité seulement par la taille de la fenêtre d'encombrement inutilisée. Pour une période modérée de repos, TFRC révisé est seulement limité par le nombre de crédits d'envoi non utilisés (jusqu'à la valeur d'un RTT). Pour une période modérée de données limitées, TFRC standard serait limité par X_recv provenant du plus récent paquet de rétroaction. À l'opposé, TFRC révisé n'est pas limité par le taux de réception provenant de périodes limitées en données qui couvrent une période entière de rétroaction d'un délai d'aller-retour. Pour TFRC standard, la [RFC3448] ne spécifie pas explicitement le comportement à l'égard des crédits d'envoi non utilisés.

C.4 Pertes durant des périodes de données limitées

Ce paragraphe discute de la réponse à une perte durant une période de données limitées.

Protocole	**Réponse à une perte durant une période de données limitées**
TCP standard :	Régler ssthresh, cwnd à FlightSize/2.
TCP avec CWV :	Le même que TCP standard.
TFRC standard :	Calculer X_Bps, envoie au plus 2*X_recv.
TFRC révisé :	Calculer X_Bps, envoie au plus recv_limit. De plus, modifie X_recv_set.

Tableau 4 : Réponse à une perte durant une période de données limitées

Dans TCP [RFC2581], la réponse à une perte durant une période de données limitées est la même que la réponse à une perte à tout autre moment dans TCP. Cette réponse est de régler la fenêtre d'encombrement à la moitié de FlightSize, où le FlightSize est la quantité réelle de données non acquittées. Donc, après une perte durant une période de données limitées, l'envoyeur TCP doit diviser par deux son taux d'envoi permis, comme il le fait normalement en réponse à une perte.

Dans TFRC standard, la réponse à une perte durant une période de données limitées est aussi la même que la réponse à une perte à tout autre moment dans TFRC standard. Le taux d'envoi est limité par X_Bps, provenant de l'équation de débit, et le taux d'envoi est aussi limité par deux fois X_recv, le taux de réception le plus récent. Par suite, après une perte dans une période de données limitées, l'envoyeur peut au plus doubler son taux d'envoi à deux fois X_recv, même si l'équation de débit X_Bps permettrait un taux d'envoi plus élevé.

Dans TFRC révisé, il y a eu des changements à l'utilisation du taux de réception X_recv durant des intervalles limités en données ; l'envoyeur est limité à envoyer au plus recv_limit, où l'envoyeur peut se souvenir du taux de réception X_recv provenant de juste avant la période de données limitées. Cela permet à l'envoyeur de plus que doubler son taux d'envoi durant les périodes limitées en données, jusqu'au taux de réception d'avant la période de données limitées (si c'est permis par l'équation de débit donnée dans X_Bps). Ceci est similaire à la pratique de TCP standard de ne pas réduire la fenêtre durant les périodes limitées en données (en l'absence de perte).

Comme avec TFRC standard, durant une période de données limitées l'envoyeur TFRC envoie moins que ce qui est permis par l'équation de débit X_Bps. Après l'événement de perte, l'envoyeur peut encore ne pas vouloir envoyer autant que ce qui est permis par la valeur recalculée de X_Bps qui prend en compte le nouvel événement de perte. TFRC révisé ajoute un mécanisme supplémentaire pour limiter graduellement le taux d'envoi de l'envoyeur après des pertes durant une période limitée en données. À la différence de la réponse de TCP de régler cwnd à la moitié de FlightSize, le mécanisme supplémentaire de TFRC révisé utilise la pratique de TFRC d'utiliser des changements de réponses ralenties pour les augmentations et diminutions du taux d'envoi permis.

Ceci est fait dans TFRC révisé (à l'étape (4) du paragraphe 4.3) en diminuant l'entrée dans X_recv_set après une perte dans un intervalle limité en données, et en permettant à l'envoyeur d'envoyer au plus max (X_recv_set) au lieu d'au plus deux fois max (X_recv_set) dans le délai d'aller-retour suivant immédiatement la perte rapportée. Donc, le "prix" pour permettre à l'envoyeur d'envoyer plus de deux fois la valeur rapportée le plus immédiatement de X_recv durant un intervalle limité en données est l'introduction d'un mécanisme supplémentaire pour réduire ce taux d'envoi permis suivant des pertes en période limitée en données.

Dans la réponse de TFRC à une perte dans un intervalle limité en données, on a considéré les exemples suivants.

Exemple 1, pertes *après* une période de données limitées : cet exemple montre que des pertes après une période de données limitées terminée sont traitées par l'équation de débit X_Bps.

Étape 1 : Non limité en données.
Envoi de 100 paquets par délai d'aller-retour (PPR).
Étape 2 : Limité en données, envoi de 10 PPR.
Étape 3 : Non limité en données.
Envoi encore de 100 PPR, comme permis par X_Bps.
Une perte de paquet dans le premier RTT de l'étape 3.
X_Bps est mis à jour.
Réponse de TFRC révisé : légère réduction du taux d'envoi permis, selon le nombre de paquets depuis le dernier événement de perte.

Tableau 5 : Exemple 1, pertes après une période de données limitées

Pour l'exemple 1, quand il y a une perte de paquet dans le premier RTT de l'étape 3, cela va être reflété dans une valeur modifiée de X_Bps, et les futurs événements de perte vont résulter en de futures réductions de l'équation de débit X_Bps. En particulier, suivant l'utilisation standard par TFRC de l'équation de débit (paragraphe A.2 de [FHPW00]) le taux d'envoi permis de TFRC va être divisé par deux après quelque chose comme cinq délais d'aller-retour successifs avec perte.

Exemple 2, un envoyeur modérément limité en données : cet exemple considère des pertes dans une période de données limitées quand, durant les périodes de données limitées, l'envoyeur envoie *presque* autant que ce qu'il lui est permis d'envoyer.

Étape 1 : Non limité en données. Envoi de 100 PPR.
Étape 2 : Limité en données, envoi de 99 PPR.
Une perte de paquet dans l'étape 2.
Réponse de TFRC révisé : légère réduction du taux d'envoi permis, jusqu'à 85 PPR ou moins, selon le nombre de paquets depuis le dernier événement de perte.

Tableau 6 : Exemple 2, envoyeur modérément limité en données

Considérons une connexion TFRC révisée où l'envoyeur a envoyé une centaine de paquets par temps d'aller retour et ensuite entre dans une période de données limitées d'envoi de seulement 99 PPR à cause de limitations de données provenant de l'application. (C'est-à-dire que chaque instance de temps durant la période de données limitées, l'envoyeur pourrait avoir envoyé un paquet de plus.) Si il y a des pertes dans la période de données limitées, le taux d'envoi permis est réduit à min(X_Bps, recv_limit) où l'équation de débit X_Bps et la limite recv_limit forcent toutes deux à une légère réduction du taux d'envoi permis.

Exemple 3, une seule perte de paquet durant une période de données limitées. Cet exemple considère la perte d'un seul paquet durant une période de données limitées, après que l'envoyeur n'a pas envoyé un paquet pour deux RTT.

Étape 1 : Non limité en données. Envoi à 100 PPR.
Étape 2 : Limité en données, envoi de 10 PPR.
Étape 3 : Limité en données, pas d'envoi de donnés pendant deux RTTs.
Étape 4 : Limité en données, envoi d'un seul paquet, qui est marqué ECN.

Réponse de TFRC révisé : une réduction du taux d'envoi permis, jusqu'à 50 PPR ou moins. Pour chaque événement de perte durant la période de données limitées, le X_recv "rappelé" d'avant la période de données limitées est effectivement divisé par deux.

Tableau 7 : Exemple 3, une seule perte de paquet

Considérons une connexion TFRC révisé où l'envoyeur a envoyé une centaine de paquets par délai d'aller-retour, et entre ensuite dans une période de données limitées d'envoi de seulement dix PPR, et ensuite n'envoie aucun paquet pendant deux RTT, puis envoie un seul paquet, qui est marqué ECN. Dans ce cas, avec TFRC révisé, pour chaque événement de perte durant la période de données limitées, l'envoyeur divise par deux son X_recv "rappelé" d'avant la période de données limitées de l'exemple 4, Pertes après augmentation du taux d'envoi durant une période de données limitées. Cet exemple considère des pertes quand l'envoyeur augmente significativement son taux d'envoi durant une période de données limitées.

Étape 1 : Non limité en données. Envoi à 100 PPR.
Étape 2 : Limité en données, envoi à 1 PPR.
Étape 3 : Limité en données, envoi à 20 PPR.
Plusieurs paquets sont perdus dans chaque RTT de l'étape 3.
Durant l'étape 3, l'envoyeur *aimerait* envoyer à 20 PPR.
Réponse de TFRC révisé : pour chaque événement de perte durant la période de données limitées, le X_recv "rappelé" d'avant la période de données limitées est effectivement divisé par deux, et le X_recv le plus récent est réduit de 0,85.

Tableau 8 : Exemple 4, Pertes après augmentation du taux d'envoi

Considérons une connexion TFRC révisé où l'envoyeur a envoyé une centaine de paquets par délai d'aller-retour, et entre ensuite dans une période de données limitées d'envoi de seulement un PPR, puis, alors qu'il est encore limité en données, augmente son taux d'envoi à vingt PPR, où il subit un certain nombre d'événements de perte successifs.

Dans ce cas, avec TFRC révisé, pour chaque événement de perte durant la période de données limitées, l'envoyeur divise par deux son X_recv "rappelé" d'avant la période de données limitées, et le X_recv le plus récent est réduit de 0,85.

C.5 Autres schémas

D'autres schéma possibles à considérer pour évaluer TFRC révisé seraient de comparer le comportement de TCP, TFRC standard, et TFRC révisé pour des connexions avec des périodes alternées d'occupation et de repos, alternant des périodes de repos et limitées en données, ou avec des périodes de repos ou de données limitées durant le démarrage lent.

C.6 Évaluation de la réponse de TFRC aux périodes de repos

Dans ce paragraphe on se concentre sur l'évaluation de la réponse de TFRC révisé aux périodes de repos ou de données limitées.

Un inconvénient de la réponse stricte de TFRC standard aux périodes de repos ou de données limitées est que cela pourrait être vu comme encourageant les applications à bourrer leur taux d'envoi durant les périodes de repos ou de données limitées, en envoyant des données factices quand il n'y a pas d'autres données à envoyer. Parce que TFRC révisé a une réponse moins stricte aux périodes limitées en données que celle de TFRC standard, TFRC révisé pourrait aussi être vu comme donnant aux applications moins d'incitation à bourrer leur taux d'envoi durant les périodes limitées en données. Des travaux en cours, comme un redémarrage plus rapide [RFC5622], peuvent aussi diminuer l'incitation des applications à bourrer leurs taux d'envoi, en permettant un démarrage plus rapide après des périodes de repos. D'autres recherches seraient utiles pour comprendre plus en détails les interactions entre les mécanismes de contrôle de l'encombrement de TCP ou TFRC, et l'incitation d'une application à bourrer son taux d'envoi durant les périodes de repos ou de données limitées.

La validation de fenêtre d'encombrement TCP, décrite à l'Appendice C.1, est un projet expérimental qui spécifie que l'envoyeur TCP réduit lentement la fenêtre d'encombrement durant une période de repos ou de données limitées [RFC2861]. Bien que les réponses de TFRC et TFRC révisé aux périodes de repos soient en gros similaires à celles de TCP avec validation de fenêtre d'encombrement, la réponse de TFRC révisé aux périodes limitées en données est moins prudente que celle de TCP avec validation de fenêtre d'encombrement (et la réponse de TFRC standard aux périodes limitées en données était considérablement *plus* conservatrice que celle de la validation de fenêtre d'encombrement). De futurs travaux pourraient inclure des modifications au présent document afin que la réponse de TFRC révisé à une période

de données limitées inclut une lente réduction du taux d'envoi permis. L'annexe C spécifie un mécanisme possible pour cela. Une telle modification serait particulièrement stimulante si la validation de fenêtre d'encombrement devenait une proposition de norme de l'IETF pour TCP.

Références normatives

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

[RFC3448] M. Handley, S. Floyd, J. Padhye, J. Widmer, "Contrôle de débit convivial sur TCP (TFRC) : Spécification du protocole", janvier 2003. (*Obsolète, voir* RFC5348) *(P.S.)*

Références pour information

[BRS99] Balakrishnan, H., Rahul, H., and Seshan, S., "An Integrated Congestion Management Architecture for Internet Hosts," Proc. ACM SIGCOMM, Cambridge, MA, septembre 1999.

[FHPW00] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", août 2000, Proc SIGCOMM 2000.

[FHPW00a] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications: the Extended Version", ICSI tech report TR-00-03, mars 2000.

[FF99] Floyd, S., and K. Fall, Promoting the Use of End-to-End Congestion Control in the Internet, IEEE/ACM Transactions on Networking, août 1999.

[MAF05] A. Medina, M. Allman, and S. Floyd, "Measuring the Evolution of Transport Protocols in the Internet", ACM Computer Communications Review, avril 2005.

[PFTK98] Padhye, J., Firoiu, V., Towsley, D. and Kurose, J., "Modeling TCP Throughput: A Simple Model and its Empirical Validation", Proc ACM SIGCOMM 1998.

[RFC2140] J. Touch, "Interdépendance de bloc de contrôle TCP", avril 1997. *(Information)*

[RFC2581] M. Alman, V. Paxson et W. Stevens, "Contrôle d'encombrement avec TCP", avril 1999. *(Obsolète, voir* RFC5681)

[RFC2861] M. Handley, J. Padhye, S. Floyd, "Validation de fenêtre d'encombrement TCP", juin 2000. *(Expérimentale, Remplacée par* RFC7661)

[RFC2988] V. Paxson, M. Allman, "Calcul du temporisateur de retransmission de TCP", novembre 2000. *(P.S.)(Obs., voir* RFC6298)

[RFC3168] K. Ramakrishnan et autres, "Ajout de la notification explicite d'encombrement (ECN) à IP", septembre 2001. *(P.S. ; MàJ par* RFC8311)

[RFC3390] M. Allman, S. Floyd, C. Partridge, "Augmentation de la fenêtre initiale de TCP", octobre 2002. *(P.S.)*

[RFC3448Err] RFC 3448 Errata, <http://www.rfc-editor.org/errata_search.php?rfc=3448 >.

[RFC3540] N. Spring, D. Wetherall, D. Ely, "Signalisation de notification robuste d'encombrement explicite (ECN) avec des noms occasionnels", juin 2003. *(Expérimentale)*

[RFC4340] E. Kohler et autres, "Protocole de contrôle d'encombrement de datagrammes (DCCP)", mars 2006. *(P.S.) (MàJ par* 6773)

[RFC4342] S. Floyd et autres, "Profil d'identifiant 3 de protocole de contrôle d'encombrement de datagrammes (DCCP) : Contrôle en douceur de débit TCP (TFRC)", mars 2006. *(P.S. ; MàJ par* RFC5348, RFC8311)

[RFC4828] S. Floyd, E. Kohler, "Contrôle de débit convivial sur TCP (TFRC) : variante du petit paquet (SP)", avril 2007. *(Exp.)*

[RFC5622] S. Floyd, E. Kohler, "Profil pour l'identifiant 4 d'encombrement du protocole de contrôle d'encombrement de datagramme (DCCP) : Contrôle de débit compatible pour les petits paquets (TFRC-SP)", août 2009. *(Expérimentale ; MàJ par* RFC8311)

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

[W00] Widmer, J., "Equation-Based Congestion Control", Thèse, Université de Mannheim, février 2000, <http://www.icir.org/tfrc/ >.

Adresse des auteurs

Sally Floyd
ICSI Center for Internet Research
1947 Center Street, Suite 600
Berkeley, CA 94708
USA
mél : floyd@icir.org

Mark Handley
University College London
London WC1E 6BT
UK
mél : M.Handley@cs.ucl.ac.uk

Jitendra Padhye
Microsoft Research
One Microsoft Way
Redmond, WA 98052
USA
mél : padhye@microsoft.com

Joerg Widmer
DoCoMo Euro-Labs
Landsberger Strasse 312
80687 Munich
Germany
mél : widmer@acm.org

Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est fourni par l'activité de soutien administratif (IASA) de l'IETF.