RFC 9959 : Convergence of Congestion Control from Retained State

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 mai 2026

Date de publication du RFC : Mai 2026

https://www.bortzmeyer.org/9959.html

Traditionnellement, les protocoles de transport comme QUIC ou TCP partaient de zéro à chaque connexion. On se connecte, on démarre prudemment (ne pas envoyer trop de données pour éviter de congestionner le réseau), puis on augmente le débit petit à petit. Mais c'est dommage de ne pas tenir compte des connexions précédentes, où on avait déjà suivi ce processus. Ne pourrait-on pas se souvenir des mesures précédentes pour aller plus vite la prochaine fois ? C'est justement ce que propose ce RFC.

Évidemment, si l'idée est simple, la réalisation soulève plein de problèmes, d'autant plus qu'on touche ici à une activité dangereuse : si on se trompe, on risque d'aggraver la congestion. Le RFC détaille donc plus précisément comment réutiliser ces mesures passées, pour ne pas faire s'écrouler le réseau. Tout protocole de transport doit utiliser un algorithme de contrôle de la congestion (RFC 2914[1]) ou bien s'auto-modérer (RFC 8085). Mais concevoir un bon algorithme de contrôle de la congestion n'est pas trivial et, par exemple, le RFC 5783 notait que les algorithmes existants marchaient mal pour les liaisons avec un BDP élevé et/ou très variable, comme les liaisons satellite.

Pour comprendre pourquoi, revenons un peu au fonctionnement typique d'un algorithme de contrôle de la congestion. Il a typiquement deux phases. D'abord, au début de la connexion, on envoie moins de données que ce qu'on pourrait, afin de ne pas congestionner le réseau. Puis on augmente le débit, jusqu'au moment où des indicateurs comme la perte de paquets ou le rythme des accusés de réception (RFC 9406) signalent qu'on a atteint la capacité maximale pour ce flux de données. La dépasser ("*overshoot*") congestionnerait le réseau ou, si les autres flux qui se partagent le réseau sont mieux élevés, entrainerait des diminutions de ressources pour ces autres, voire un écroulement du réseau sous la charge. Voilà pourquoi il faut démarrer lentement.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc2914.txt

Au passage, le RFC note que, bien sûr, la connexion TCP ou QUIC qui réutiliserait les mesures d'une précédente connexion doit s'assurer qu'on compare ce qui est comparable : mêmes adresses IP source et destination, et peut-être même DSCP ("*Differentiated Services Code Point*", cf. RFC 2474). On verra plus loin que ce n'est pas suffisant (les choses peuvent changer, le passé peut ne pas être un bon indicateur du présent), mais patientez encore un peu.

La méthode de notre RFC se nomme « reprise prudente » ("*careful resume*") ou, en plus joli « partage temporel » ("*temporal sharing*"), puisqu'on reprend des mesures passées (mesures de la capacité <https://www.bortzmeyer.org/capacite.html>, du RTT, etc). Ces mesures pouvant ne plus être d'actualité (trop vieilles et/ou le chemin suivi a changé), il faut en effet être prudent (cf. RFC 9000 et RFC 9040).

Dans quels cas cette reprise (prudente) de données d'une connexion précédente peut être utile ? La section 1.4 du RFC liste un certain nombre de cas d'usages. Entre autres, il y a le cas d'une application qui utilise plusieurs connexions, les connexions peuvent alors utiliser les données récupérées par la première d'entre elles. Ou bien lorsqu'une connexion a été violemment interrompue et repart tout de suite après. Et il y a aussi une autre utilisation, lorsque la latence <https://www.bortzmeyer.org/latence.html> du chemin utilisé est bien plus importante que dans une liaison Internet typique, ce qui est le cas des satellites lorsqu'ils ne sont pas en orbite basse. L'article « "*Google QUIC performance over a public SATCOM access* " <https://arxiv.org/abs/1810.04970> » note ainsi que sur une liaison via un satellite géostationnaire, avec l'algorithme classique, transférer 5,3 Mo prendra 9 secondes alors que le partage temporel, si on peut réutiliser les mesures d'une connexion précédente, prendra 4 secondes. (Si les 9 secondes vous semblent trop, compte-tenu de la capacité du lien, rappelez-vous que la capacité n'est pas le seul facteur limitant ; TCP, surtout au démarrage, n'utilise pas toute la capacité, et il met longtemps à converger si la latence est élevée.) Une autre présentation, à l'IETF « "*Feedback from using QUIC's 0- RTT-BDP extension over SATCOM public access*" <https://datatracker.ietf.org/meeting/111/materials/slides-111-maprg-feedback-from-using-quics-0-rtt-bdp-extension-over-sat> calcule une réduction du temps de transfert de 62 % pour faire voyager 1 Mo. Enfin, le RFC recommande la lecture de la synthèse « "*Careful Resumption of Internet Congestion Control from Retained Path State*" <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5240200> ».

Le récepteur des données peut avoir des informations que l'envoyeur n'a pas, et cela peut le pousser à vouloir désactiver la reprise que l'envoyeur croit prudente. Par exemple, le récepteur sait peut-être quelle quantité de données va être envoyée, ou bien il sait que le chemin sur le réseau a changé.

Toujours pour compliquer les choses, il faut aussi se souvenir que d'autres facteurs peuvent limiter la quantité de données qu'on envoie, par exemple le contrôle de flux, donc les fenêtres de TCP (RFC 9293, section 3.8.6) ou le système de crédit de QUIC (RFC 9000, section 4).

Prenant en compte tout cela, la section 1.5 du RFC résume les principes de la « reprise prudente » :

— D'abord, s'assurer que les conditions n'ont pas changé depuis la dernière connexion ; le chemin est-il le même ? (La mesure du RTT donne une première bonne idée.) À l'issue de cette phase de reconnaissance, si les conditions ont changé, on utilise la méthode classique, on ne tient pas compte des mesures qui ont été sauvegardées.
— Ensuite, OK, on va plus vite qu'avec la méthode classique mais pas aussi vite que ne l'indiquent les mesures sauvegardées. Après tout, la reconnaissance ne permet pas toujours d'identifier un changement dans les conditions.
— Enfin, on se prépare à faire marche arrière, et à revenir à la méthode classique, si on s'aperçoit qu'on est en train de créer de la congestion. (Le RFC note qu'il faut la détecter rapidement, avant que les autres flots de données n'aient détecté la congestion et réduit leur débit.)

Un peu de terminologie avant de continuer (section 2) : le partenaire distant ("*remote endpoint*") est l'ensemble des informations qui identifie à qui on envoie des données, typiquement l'identificateur d'une interface réseau locale couplé à l'adresse IP de la machine avec qui on parle et peut-être (c'est une décision locale, et forcément très dépendante du système d'exploitation utilisé) des informations comme DSCP. Si le partenaire distant change, on en déduit que le chemin a changé (l'inverse n'est pas vrai : le chemin peut changer alors que le partenaire distant est le même). Si on a une indication que le chemin a changé, on revient au mécanisme traditionnel, on n'utilise pas les paramètres gardés des précédentes connexions.

Le mécanisme de notre RFC a donc plusieurs phases (section 3, mais regardez aussi le joli tableau de l'annexe A, qui est peut-être plus clair) : celle de reconnaissance, où on cherche si les caractéristiques du chemin correspondent à un chemin connu, puis , selon son résultat, on passe à la phase dite normale, si on a reconnu un chemin déjà vu, ou bien à la phase non-validée (chemin inconnu, on démarre prudemment), puis à la phase de validation (y a t-il un indicateur de congestion ou bien tout est-il beau et propre ?), ou bien à celle de retraite (on jette les informations enregistrées, la situation a changé, on retourne en mode traditionnel), avant de passer à la phase normale. (Des exemples détaillés figurent dans l'annexe B.)

La section 4 du RFC fournit des détails sur la mise en œuvre des principes de ce RFC. Par exemple, la détermination pratique d'un changement du chemin. Un bon indicateur est le RTT. Cette section conseille aussi, même en l'absence d'indications que le chemin a changé, de ne pas garder les paramètres sauvegardés trop longtemps : comme la détection d'un éventuel changement n'est pas parfaite, il vaut mieux avoir une durée de vie maximale pour les paramètres enregistrés. Le RFC suggère quelques heures, voire moins si on sait que le chemin est très dynamique.

Enfin, l'annexe B détaille à l'octet près des exemples de fonctionnement de l'algorithme de reprise prudente, pour différents cas.

Vous pouvez aussi avoir une introduction aux principes de ce RFC dans l'exposé d'un des auteurs à la Journée du Conseil Scientifique de l'Afnic en 2021 <https://www.afnic.fr/observatoire-ressources/actualites/jcsa21-retour-sur-ledition-2021-de-la-journee-du-conseil-scientifique-de-lafni> (avec les supports <https://www.afnic.fr/wp-media/uploads/2021/09/afnic-jcsa2021_cnes_kuhn.pdf>).

Merci à Nicolas Kuhn pour sa relecture.