Commencer les sessions TCP plus vite ?

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 4 mai 2012

https://www.bortzmeyer.org/tcp-ouverture-rapide.html

Tout le monde a envie que les réseaux aillent plus vite. Surtout Google. Un article de 2011, par des employés de cette société, « "*TCP Fast Open*" <http://research.google.com/pubs/pub37517.html> » explique un des mécanismes qui peuvent être utilisés, notamment pour les sessions TCP de courte durée. (Cette technique a, depuis, été décrite dans le RFC 7413[1].)

Un paradoxe de TCP, connu depuis longtemps, est que le temps complet d'une session (par exemple, de récupération d'un fichier) dépend souvent essentiellement du temps d'**ouverture** de la session. En effet, beaucoup de sessions TCP sont courtes (par exemple, en HTTP, on `GET /machintruc.html`, on récupère une page de cinq kilo-octets et c'est tout). Le temps de transfert des données est négligeable par rapport au temps d'ouverture de la session, qui est causé par la latence.

Car il faut oublier les stupides publicités des FAI qui promettent toutes des capacités mirifiques (et, en prime, ils se trompent et parlent de « bande passante », voire, pire, de « débit », notez que la capacité est définie dans le RFC 5136), des Mb/s en quantité. La capacité est bien pratique quand on télécharge la saison 2 de Rome mais, pour beaucoup d'interactions sur le Web, c'est la latence qui compte. En effet, l'ouverture d'une session TCP (RFC 793) nécessite l'échange de trois paquets :

— « Je veux te parler » (paquet `SYN`),
— « Je veux te parler aussi » (paquet `SYN+ACK`),
— « Parlons-nous, alors » (paquet `ACK`).

L'acheminement de chaque paquet prend un temps certain (dixit Fernand Raynaud, mais on trouve une définition plus rigoureuse dans le RFC 7679) et le temps total d'ouverture est trois fois ce temps.

Vous ne me croyez pas ? Ce n'est pas grave, Google a mesuré pour cet article, en utilisant à la fois leurs serveurs HTTP et les mesures faites par Chrome. Première observation : la page Web moyenne fait 300 ko mais la plupart des fichiers qu'on télécharge sont petits, avec une taille moyenne de 7,3 ko et surtout une médiane de 2,4 ko. Un objet de 2,4 ko ne prend que deux paquets pour être transmis, la

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc7413.txt`

majorité des paquets seront donc pour l'ouverture de la connexion. Si le RTT est long, c'est lui, et pas la capacité, qui fixera le temps d'attente.

L'article montre ensuite que certaines solutions à ce problème, comme les connexions persistantes d'HTTP (RFC 7230, section 6.3), n'aident pas souvent, en pratique.

Alors, une solution évidente à ce problème de performances serait de permettre au premier paquet envoyé de porter des données. Le dialogue, pour un objet qui tient dans un seul paquet, deviendrait :

— « Je veux te parler, `GET /machintruc.html` » (paquet `SYN`),
— « Je veux te parler aussi, [contenu de `machintruc.html`] » (paquet `SYN+ACK`),
— « Parlons-nous, alors, tiens, c'est fini » (paquet ACK).

Mais cela ouvre une énorme faille de sécurité. Le premier paquet peut mentir sur son adresse IP source (au premier paquet, TCP n'a pas encore testé la réversibilité `<https://www.bortzmeyer.org/returnability.html>`) et forcer le serveur HTTP à envoyer des données à la victime dont il a usurpé l'adresse (une attaque par **réflexion** et **amplification**, la réponse étant plus grosse que la requête).

Donc, comme Google est très fort, comment fait-il ?

— Le demandeur fait une première session TCP normale avec le serveur, mais en incluant une option TCP, `Fast Open Cookie Request` (les options TCP sont décrites dans la section 3.1 du RFC 793, les options de "*TCP Fast Open*" sont enregistrées dans le registre IANA des options expérimentales `<https://www.iana.org/assignments/tcp-parameters/tcp-parameters.xml#tcp-exids>`),
— Le répondeur fabrique le "*cookie*" en hachant l'adresse IP du client avec une clé secrète et le transmet dans la réponse,
— Le client se souvient de ce "*cookie*" pour les connexions ultérieures.

On le voit, "*TCP Fast Open*" ne traite pas le cas de la première connexion. Mais ce n'est pas grave : on fait en général beaucoup de connexions TCP avec un serveur Web. Pour les connexions ultérieures, le client enverra le "*cookie*" dans le premier paquet, dans l'option TCP `Fast Open Cookie`. Le serveur vérifiera ledit gâteau et, s'il est correct, acceptera les données transmises dans le premier paquet (typiquement un `GET /machintruc.html` HTTP).

Naturellement, ça, ce sont les généralités. En pratique, il y a plein de détails dont il faut s'occuper. Par exemple, "*TCP Fast Open*" va devoir se battre contre les "*middleboxes*", cette plaie de l'Internet, cf. l'excellente étude de Google « "*Probing the viability of TCP extensions*" `<https://www.imperialviolet.org/binary/ecntest.pdf>` ». Et il y a la taille (limitée) dans l'en-tête TCP pour les options. Et le fait qu'un accusé de réception peut se perdre, amenant à la réémission d'un paquet `SYN` (et donc à un `GET` reçu deux fois : "*TCP Fast Open*" ne doit être utilisé qu'avec des requêtes idempotentes). Mais tout cela n'a pas arrêté les auteurs qui ont programmé cette option dans Linux (dans les 2 000 lignes de code) et Chrome. Résultat, avec une latence de 100 ms, entre 6 et 16 % de gain, montant à 11-41 % pour 200 ms de latence.

À noter que cette idée d'accélérer TCP en mettant davantage d'informations plus tôt était à la base de T/TCP (RFC 1644). Ce protocole avait de sérieux défauts (cf. RFC 4614) et a officiellement été abandonné dans le RFC 6247.

L'actuel TFO ("*TCP Fast Open*") a ensuite été décrit dans un RFC de statut « expérimental » (donc pas une « vraie » norme) en décembre 2014, dans le RFC 7413) . Mais TFO avait déjà fait l'objet de pas mal de "*buzz*", voir par exemple l'article du Monde Informatique `<http://www.lemondeinformatique.fr/actualites/lire-les-propositions-de-google-pour-accelerer-la-couche-tcp-47521.html>`, celui de 01 Net `<http://pro.01net.com/editorial/553770/google-veut-accelerer-le-proto` > ou celui de Développez `<http://www.developpez.net/forums/d1178554/systemes/autres-systemes reseaux/google-veut-augmenter-vitesse-web/>`. Ces articles (et surtout leurs commentaires) comportant pas mal d'erreurs, il vaut mieux revenir vers

l'article du blog de Google `<http://googlecode.blogspot.fr/2012/01/lets-make-tcp-faster.html>` (où "*TCP Fast Open*" n'est qu'une partie des changements proposés à TCP). L'ouverture rapide a été intégrée au noyau Linux officiel en septembre 2012, voir cet article dans LWN `<https://lwn.net/Articles/458610/>` et cet autre `<https://lwn.net/Articles/508865/>`, plus détaillé, avec exemple de code C. Le tout est apparu dans la version 3.6 de Linux.
