RFC 5594 : Report from the IETF workshop on P2P Infrastructure, May 28, 2008

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 24 août 2009

Date de publication du RFC : Juillet 2009

https://www.bortzmeyer.org/5594.html

Le développement très rapide et très spectaculaire du pair-à-pair soulève plein de questions. Parmi elles, celle de la charge que font peser ces services sur les réseaux des opérateurs, et les points sur lesquels un travail de normalisation de l'IETF pourrait aider. C'étaient les deux thèmes du colloque organisé au MIT en mai 2008 et dont ce RFC est le compte-rendu.

La principale application du pair-à-pair aujourd'hui est le transfert de fichiers. Il n'y a pas de limite à la taille des contenus, notamment vidéo et les logiciels de pair-à-pair, comme le rappelle la section 1 du RFC, sont conçus pour utiliser le réseau à fond. C'était le point de départ du colloque : peut-on améliorer les choses ? Avant cela, il faut comprendre le phénomène et donc étudier les caractéristiques des réseaux pair-à-pair existants. Puis trois propositions ont émergé du colloque, un protocole de contrôle de congestion mieux adapté aux transfert de fichiers massifs, une étude des applications qui ouvrent plusieurs connexions simultanées, et une solution qui permettrait d'améliorer la sélection des pairs avant de commencer le transfert du fichier convoité.

Pour mieux comprendre le phénomène, le colloque a vu des exposés par des opérateurs (section 3). Par exemple, deux employés de Comcast ont décrit successivement les problèmes spécifiques aux réseaux câblés utilisant la norme DOCSIS, où chaque modem a accès à son tour. Par exemple, lorsque le CMTS donne accès au câble à un modem, celui-ci peut alors envoyer autant de données qu'il veut. Une machine en train d'envoyer un gros fichier pourra alors, à nombre de transferts égaux, prendre nettement plus de capacité que les autres.

Pour éviter les problèmes, outre les évolutions de la norme DOCSIS, certains opérateurs cherchent à mesurer en temps réel l'activité réseau des clients, pour ensuite limiter dans le CMTS leur accès au réseau. Outre qu'elle pose la question de la transparence du réseau, on peut noter que l'utilisateur n'est pas informé de cette situation (communiquer vers les clients des informations concrètes n'est jamais la priorité des opérateurs).

Autre point de vue, celui des développeurs de logiciels pair-à-pair (section 4). Stanislas Shalunov, de BitTorrent, a expliqué les défis auxquels étaient confrontés les développeurs et proposé des solutions. Là encore, les réseaux où beaucoup de ressources sont partagées, comme ceux fondés sur le câble TV, sont particulièrement affectés puisqu'un seul abonné peut en gêner beaucoup d'autres.

Les solutions possibles font l'objet de la section 5. La plus évidente est d'améliorer la sélection des pairs (section 5.1). Lorsqu'un client BitTorrent veut télécharger un fichier, il peut choisir n'importe lequel (ou lesquels) dans un **essaim** qui comporte souvent des centaines ou des milliers de machines. En général, il ne tient aucun compte de la topologie du réseau ou des informations contenues dans le système de routage. L'algorithme de sélection du pair varie selon les logiciels mais aboutit souvent à choisir une machine à l'autre bout du monde, alors qu'un pair volontaire était plus « proche ». Cela induit un délai supplémentaire pour l'utilisateur, et une charge plus grande du réseau pour l'opérateur.

Une solution serait donc d'avoir quelque part un service qui pourrait informer les pairs des conditions topologiques et les aider à choisir de meilleurs pairs.

Il n'est pas certain qu'il existe une solution idéale. Par exemple, si BitTorrent utilise une part d'aléatoire dans la sélection d'un pair, c'est aussi parce que cela évite la concentration des requêtes sur les machines les mieux placées, et permet de trouver ceux qui ont des parties rares d'un fichier.

En outre, les intérêts de l'opérateur et de l'utilisateur ne sont pas identiques. On pourrait imaginer des cas où la sélection qui soit optimale pour l'opérateur ne le serait pas pour l'utilisateur... qui risquerait donc de ne plus utiliser le système de sélection. Il est donc important que le système d'aide à la sélection ne soit donc qu'une indication parmi d'autres.

Compte-tenu de tout ceci, quelles sont les solutions techniques possibles ? Par exemple, le RFC cite :

— Tenir compte des AS. Des études ont montré que, avec des essaims typiques (de 2 000 à 10 000 pairs), on pouvait obtenir la majorité des fichiers uniquement en demandant à l'intérieur de l'AS. (En revanche, un niveau de détail plus grand, par exemple chercher les pairs connectés au même DSLAM, ne rapporte quère de bénéfice, la probabilité de tout trouver dans le voisinage devient trop faible.)
— Utiliser P4P, un système où l'opérateur met à la disposition de ses clients un service leur permettant de trouver les pairs les plus proches en communiquant des informations sur la topologie du réseau et le coût des différents liens.
— Un système « multi-niveaux » où plusieurs services d'information coexisteraient, certains pour le contenu global, d'autres uniquement pour un contenu local, avec redirection des premiers vers les seconds. Comme le contenu peut être, dans certains cas, illégal, la question de la responsabilité de l'opérateur qui gère le service pourrait être engagée.
— Créer un système de sélection du pair assisté par l'opérateur. Après tout, le FAI connait plein de choses sur son propre réseau, les liens de transit et ceux de « "*peering*" », les métriques OSPF ou BGP, la capacité des différents liens, etc. Cette information permettrait sûrement au logiciel de mieux choisir ses pairs. Dans cette optique, le FAI installerait un « oracle » qui serait un service d'information qui, contrairement à P4P, ne distribuerait pas cette connaissance (beaucoup d'opérateurs la considèrent comme confidentielle) mais qui recevrait une liste d'adresses IP de pairs potentiels et la renverrait triée, les « meilleurs » en premier. Cela pose évidemment un grand problème de passage à l'échelle. Pour un essaim de 10 000 pairs, la quantité d'information à transmettre à l'oracle est impressionnante. Pour les essaims du futur, qui pourraient compter des centaines de milliers de pairs, il faudra renoncer à tout transmettre. Mais le gros avantage de cette méthode est que l'opérateur ne voit que des adresses IP, et ne connait rien du contenu transféré, ce qui décharge complètement sa responsabilité juridique.

— En informatique, des tas de problèmes de performance se résolvent en mettant des **caches**, des mémoires intermédiaires, rapides, entre le demandeur d'information et le lieu de stockage de cette information. Un FAI pourrait donc installer des caches pair-à-pair sur son réseau. Les risques juridiques sont alors à l'opposé de la solution précédente : le FAI garderait en effet sur ses propres machines un contenu dont une partie est considérée comme illégale par certains.

Une fois ces solutions énumérées, que peut faire l'IETF ? Vouloir normaliser un système pair-à-pair complet (un « BitTorrent IETF ») est clairement prématuré, dans l'état actuel de l'expérience avec le pair-à-pair, domaine très bouillonant. À la place, le RFC 5594 [1] n'envisage que de la spécification de quelques composants ponctuels d'un tel système. Une longue section 5.1.6 énumère les actions possibles pour chacune des solutions citées plus haut :

— Pour trouver le numéro d'AS d'une machine, il n'existe actuellement pas d'interface standard. L'IETF pourrait en développer une.
— Pour interroger un « oracle », un protocole standard d'interrogation (qui ne préjugerait pas de comment l'oracle prend sa décision) serait également souhaitable.
— Pour **découvrir** l'oracle, un moyen normalisé (même si ce n'est qu'une simple convention de nommage comme `oracle.example.net`) aurait bien des avantages. DHCP est un moyen possible, l'"*anycast*" un autre, un enregistrement SRV dans le DNS un troisième.
— On l'a vu plus haut, lorsque le service que vend un FAI est limité, par exemple à une certaine quantité d'octets par mois, ou bien par "*shaping*" automatique, l'approche de la limite n'est pas communiquée à l'utilisateur, ou uniquement par des mécanismes non-standard et souvent peu adaptés à une récupération automatique (par exemple via une page Web). Un protocole standard de communication aiderait les utilisateurs à mieux surveiller leur consommation. Ce ne serait évidemment pas facile de développer un tel protocole, compte-tenu de la variété des offres commerciales (pensez au côté incompréhensible des offres dans la téléphonie mobile).
— Si plusieurs services d'information sont possibles, avec redirection entre eux, un mécanisme standard pour cette redirection serait bienvenu.

Après une meilleure sélection des pairs, l'autre grand chantier que pose le pair-à-pair est le contrôle de congestion (section 5.2). La plupart des applications cherchent à transférer leurs données le plus vite possible. Avec l'abondance de contenu disponible en pair-à-pair, cela peut signifier une occupation permanente du réseau. Si les logiciels de pair-à-pair étaient plus modérés dans leur usage de la capacité réseau disponible, les choses iraient mieux. (Certains logiciels offrent des possibilités d'auto-limitation, par exemple, avec mldonkey, on peut mettre `max_hard_upload_rate = 10` dans `downloads.ini` et on limite alors la consommation « montante » à 10 ko/s.)

Un exemple d'un meilleur comportement est le client BitTorrent officiel qui mesure en permanence le délai de transmission et cherche à le minimiser, ce qui laisse de la capacité pour les applications qui veulent une faible latence (comme la VoIP).

Une autre approche serait un contrôle de congestion pondéré, pour que TCP sur la machine laisse davantage de ressources à certaines applications (actuellement, depuis l'échec de TOS dans IP, l'application n'a pas de moyen standard d'indiquer ce genre de préférences, DiffServ (RFC 2474) est plutôt prévu pour être géré par le réseau (section 5.2.2 pour un traitement plus détaillé de la pondération).

Un autre endroit où mettre en œuvre une telle inégalité de traitement est bien sûr dans le réseau lui-même. Ainsi, un FAI peut déployer une architecture souvent appelée du terme marketing de qualité de service (alors que le but est au contraire de dégrader la qualité du service de certaines applications, au profit d'autres). Pour classer les données qui circulent dans son réseau, il peut utiliser les numéros de port (443 indique de l'HTTPS, 22 du SSH, etc) ou la DPI. Comme cette classification vise à diminuer la qualité de service de certains usages, les logiciels tentent en permanence d'y échapper et, de même que le numéro de port est devenu peu à peu inutile, tous les logiciels changeant de numéro ou détournant

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc5594.txt`

un numéro de port comme 443, la DPI motive les auteurs de logiciel à utiliser de plus en plus la cryptographie pour éviter à leur trafic d'être reconnu (section 5.3).

Autre question de justice : le cas des applications qui ouvrent plusieurs connexions TCP simultanées (section 6). La tendance avait été lancée par Netscape pour imposer son navigateur contre Mosaic. En effet, TCP essayant d'attribuer des parts identiques à toutes les connexions, avoir plusieurs connexions permet une part plus grosse. Il est donc nécessaire d'étudier plus attentivement la question.

Il y a bien longtemps que l'Internet ne vit plus exclusivement des subventions de l'armée états-unienne et tous les débats techniques du monde pèsent donc peu à côté des histoires de gros sous. La section 7 détaille donc les discussions autour du coût du pair-à-pair. Les chiffres rigoureux manquent dans ce domaine car, s'il est facile de connaître le prix d'un câble, celui de la congestion est bien plus dur à fixer. Il existe des estimations allant de 10 cents à 2 dollars US pour un gigaoctet transféré. La consommation qu'entraine le pair-à-pair pousse certains à réclamer le retour au tarif au volume.

Traditionnellement, l'IETF ne travaille pas sur les questions économiques, qui sont difficiles, vue la variété des modèles d'affaires, et de toute façon pas forcément de sa compétence. Mais il faudrait peut-être que l'IRTF commence un travail sur ce sujet.

Assez parlé, il faut maintenant agir. Quelles sont les prochaines étapes ? La section 8 cite les suivantes :

— Travailler à un protocole de contrôle de congestion permettant aux applications pair-à-pair de réduire leur propre impact. À ma connaissance, ce travail n'a pas encore commencé.
— Documenter précisement les effets de l'ouverture de connexions TCP multiples (section 6). Là non plus, je ne crois pas que ce travail aie déjà démarré.
— Améliorer la séclection des pairs, par un nouveau protocole d'information (section 5.1). C'est désormais le travail du groupe Alto <https://www.bortzmeyer.org/alto-wg.html>.

Les exposés faits lors du colloque peuvent être téléchargés en <http://trac.tools.ietf.org/area/rai/trac/wiki/PeerToPeerInfrastructure>.