RFC 8899 : Packetization Layer Path MTU Discovery for Datagram Transports

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 septembre 2020

Date de publication du RFC : Septembre 2020

https://www.bortzmeyer.org/8899.html

Ce RFC qui vient d'être publié décrit une méthode pour découvrir la MTU d'un chemin sur l'Internet ("*Path MTU*", la découverte de la MTU du chemin se nommant PMTUD pour "*Path MTU Discovery*"). Il existe d'autres méthodes pour cette découverte, comme celle des RFC 1191 [1] et RFC 8201, qui utilisent ICMP. Ici, la méthode ne dépend pas d'ICMP, et se fait avec les datagrammes normaux de la couche 4 (ou PL, pour "*Packetization Layer*"). Elle étend la technique du RFC 4821 à d'autres protocoles de transport que TCP (et SCTP).

Mais, d'abord, pourquoi est-ce important de découvrir la MTU maximale du chemin ? Parce que, si on émet des paquets qui sont plus gros que cette MTU du chemin, ces paquets seront jetés, au grand dam de la communication. En théorie, le routeur qui prend cette décision devrait fragmenter le paquet (IPv4 seulement), ou bien envoyer à l'expéditeur un message ICMP "*Packet Too Big*" mais ces fragments, et ces messages ICMP sont souvent bloqués par de stupides pare-feux mal gérés (ce qu'on nomme un trou noir). On ne peut donc pas compter dessus, et ce problème est connu depuis longtemps (RFC 2923, il y a vingt ans, ce qui est une durée insuffisante pour que les administrateurs de pare-feux apprennent leur métier, malgré le RFC 4890). La seule solution fiable est de découvrir la MTU du chemin, en envoyant des paquets de plus en plus gros, jusqu'à ce que ça ne passe plus, montrant ainsi qu'on a découvert la MTU.

Le RFC donne aussi quelques raisons plus subtiles pour lesquelles la découverte classique de la MTU du chemin ne marche pas toujours. Par exemple, s'il y a un tunnel sur le trajet, c'est le point d'entrée du tunnel qui recevra le message ICMP "*Packet Too Big*", et il peut faillir à sa tâche de le faire suivre à

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc1191.txt

l'émetteur du paquet. Ou bien, en cas de routage asymétrique, le message ICMP peut être jeté car il n'y a pas de route de retour.

Et ce n'est pas tout que le message ICMP revienne à l'émetteur, encore faut-il que celui-ci l'accepte. Les messages ICMP n'étant pas authentifiés, une machine prudente va essayer de les valider, en examinant le paquet originel contenu dans le message ICMP, et en essayant de voir s'il correspond à du trafic en cours. Si le routeur qui émet le message ICMP n'inclut pas assez du paquet original (malgré ce que demande le RFC 1812) pour que cette validation soit possible, le message ICMP « Vous avez un problème de taille » sera jeté par l'émetteur du paquet original. Et il peut y avoir d'autres problèmes, par exemple avec le NAT (cf. RFC 5508). Autant de raisons supplémentaires qui font que la découverte de la MTU du chemin ne peut pas compter sur les messages ICMP.

Notre RFC utilise le terme de couche de découpage en paquets ("*Packetization Layer*", PL). Il désigne la couche où les données sont séparées en paquets qui seront soumis à IP. C'est en général la couche de transport (protocoles TCP, DCCP, SCTP, etc) mais cela peut aussi être fait au-dessus de celle-ci. Ce sigle PL donne naissance au sigle PLPMTUD, "*Packetization Layer Path MTU Discovery*". Contrairement à la classique PMTUD ("*Path MTU Discovery*") des RFC 1191 et RFC 8201, la PLPMTUD ne dépend pas des messages ICMP. C'est la PL, la couche de découpage en paquets qui se charge de trouver la bonne taille, en envoyant des données et en regardant lesquelles arrivent. La PLPMTUD est plus solide que la traditionnelle PMTUD, qui est handicapée par le blocage fréquent d'ICMP (cf. RFC 4821, qui a introduit ce terme de PLPMTUD, et RFC 8085).

La découverte de la MTU du chemin par le PL est ancienne pour TCP (RFC 4821). Et pour les autres protocoles de transport ? C'est l'objet de notre RFC. Pour UDP, le RFC 8085 (section 3.2) recommande d'utiliser une telle procédure (sauf si les couches inférieures fournissent l'information). Et, pour SCTP, la section 10.2 du RFC 4821 le recommandait également mais sans donner de détails, désormais fournis par notre nouveau RFC.

Un peu de vocabulaire avant de continuer la lecture du RFC (section 2) : un trou noir est un endroit du réseau d'où des paquets ne peuvent pas sortir. Par exemple, parce que les paquets trop gros sont jetés, sans émission de messages ICMP, ou bien avec des messages ICMP filtrés. Un paquet-sonde est un paquet dont l'un des buts est de tester la MTU du chemin. Il est donc de grande taille, et, en IPv4, a le bit DF ("*Don't Fragment*"), qui empêchera la fragmentation par les routeurs intermédiaires. Ceci étant défini, attaquons-nous au cahier des charges (section 3 du RFC). Le RFC 4821 était très lié à TCP, mais la généralisation de la PLPMTUD qui est dans notre nouveau RFC va nécessiter quelques fonctions supplémentaires :

— Il faut évidemment qu'on puisse envoyer des paquets-sonde, donc mettre le bit DF à 1 (en IPv4).
— Surtout, il faut un mécanisme de notification de la bonne arrivée du paquet. En TCP (ou SCTP, ou QUIC `<https://www.bortzmeyer.org/quic.html>`), il est inclus dans le protocole, ce sont les accusés de réception (`ACK`). Mais pour UDP et des protocoles similaires, il faut ajouter ce mécanisme de notification, ce qui implique en général une coopération avec l'application (par exemple, pour le DNS, le client DNS peut notifier le PL qu'il a reçu une réponse à sa question, prouvant ainsi que le datagramme est passé).
— Les paquets-sonde peuvent se perdre pour d'autres raisons que la MTU, la congestion, par exemple. La méthode de PLPMTUD doit donc savoir traiter ces cas, par exemple en réessayant. (D'autant plus que le paquet-sonde ne sert pas forcément qu'à la PLPMTUD, il peut aussi porter des données utiles.)
— Et, même si la PLPMTUD ne dépend pas d'ICMP, il est quand même recommandé d'utiliser les messages PTB ("*Packet Too Big*") si on en reçoit (cf. section 4.6). Attention toutefois à faire des efforts pour les valider : rien n'est plus facile pour un attaquant que d'envoyer de faux PTB pour réduire la MTU et donc les performances. Une solution sûre est de ne pas changer la MTU en recevant un PTB, mais d'utiliser ce PTB comme indication qu'il faut re-tester la MTU du chemin tout de suite.

En parlant de validation, cela vaut aussi pour les paquets-sonde (section 9 du RFC) : il faut empêcher un attaquant situé hors du chemin d'injecter de faux paquets. Cela peut se faire par exemple en faisant varier le port source (cf. section 5.1 du RFC 8085), comme le fait le DNS.

La section 4 du RFC passe ensuite aux mécanismes concrets à utiliser. D'abord, les paquets-sonde. Que doit-on mettre dedans, sachant qu'ils doivent être de grande taille, le but étant d'explorer les limites du chemin ? On peut utiliser des données utiles (mais on n'en a pas forcément assez, par exemple les requêtes DNS sont toujours de petite taille), on peut utiliser uniquement des octets bidons, du remplissage (mais c'est ennuyeux pour les protocoles qui doivent avoir une faible latence <https://www.bortzmeyer.org/latence.html>, comme le DNS, qui n'ont pas envie d'attendre pour envoyer les vraies données), ou bien on peut combiner les deux (des vraies données, plus du remplissage). Et, si on utilise des données réelles, il faut gérer le risque de perte, et pouvoir réémettre. Le RFC ne donne pas de consigne particulière aux mises en œuvre de PLPMTUD, les trois stratégies sont autorisées.

Ensuite, il faut évidemment un mécanisme permettant de savoir si le paquet-sonde est bien arrivé, si le protocole de transport ne le fournit pas, ce qui est le cas d'UDP, ce qui nécessite une collaboration avec l'application.

Dans le cas où le paquet-sonde n'arrive pas, il faut détecter la perte. C'est relativement facile si on a reçu un message ICMP PTB mais on ne veut pas dépendre uniquement de ces messages, vu les problèmes d'ICMP dans l'Internet actuel. Le PL doit garder trace des tailles des paquets envoyés et des pertes (cf. le paragraphe précédent) pour détecter qu'un trou noir avale les paquets de taille supérieure à une certaine valeur. Il n'est pas facile de déterminer les seuils de détection. Si on réduit la MTU du chemin au premier paquet perdu, on risque de la réduire trop, alors que le paquet avait peut-être été perdu pour une tout autre raison. Si on attend d'avoir perdu plusieurs paquets, on risque au contraire de ne pas réagir assez vite à un changement de la MTU du chemin (changement en général dû à une modification de la route suivie).

(J'ai parlé de MTU du chemin mais PLMTUD détecte en fait une valeur plus petite que la MTU, puisqu'il y a les en-têtes IP.)

Si vous aimez les détails des protocoles, les machines à état et la liste de toutes les variables nécessaires, la section 5 du RFC est pour vous, elle spécifie complètement la PLMTUD pour des protocoles utilisant des datagrammes. Cette section 5 est générique, et la section 6 décrit les détails spécifiques à un protocole de transport donné.

Ainsi, pour UDP (RFC 768) et UDP-Lite (RFC 3828), le protocole de transport n'a tout simplement pas les mécanismes qu'il faut pour faire de la PLPMTUD ; cette découverte de la MTU du chemin doit être faite par l'application, dans l'esprit du RFC 8085. L'idéal serait que cette PLPMTUD soit mise en œuvre dans une bibliothèque partagée, pour éviter que chaque application ne la réinvente mal. Mais je ne connais pas actuellement de telle bibliothèque.

Le RFC insiste sur le fait que l'application doit, pour effectuer cette tâche, pouvoir se souvenir de quels paquets ont été envoyés, donc mettre dans chaque paquet un identificateur, comme le "*Query ID*" du DNS.

Pour SCTP (RFC 9260), c'est un peu plus facile, puisque SCTP, comme TCP, a un système d'accusé de réception. Et les "*chunks*" de SCTP fournissent un moyen propre d'ajouter des octets au paquet-sonde pour atteindre la taille souhaitée (cf. RFC 4820), sans se mélanger avec les données des applications.

Pour le protocole QUIC <https://www.bortzmeyer.org/quic.html>, la façon de faire de la PLMTUD est spécifiée dans le RFC 9000. DCCP, lui, n'est pas spécifiquement cité dans cette section.

Ah, et quelles mises en œuvre de protocoles font déjà comme décrit dans ce RFC ? À part divers tests, il parait (mais je n'ai pas vérifié personnellement) que c'est le cas pour SCTP dans FreeBSD, et dans certains navigateurs Web pour WebRTC (WebRTC tourne sur UDP et rappelez-vous qu'en UDP, il faut une sérieuse coopération par l'application pour faire de la PLPMTUD). Côté QUIC, il y a lsquic <https://github.com/litespeedtech/lsquic>, qui gère les techniques de notre RFC.