RFC 2680 : A One-way Packet Loss Metric for IPPM

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 3 décembre 2010

Date de publication du RFC : Septembre 1999

https://www.bortzmeyer.org/2680.html

Une des tristes réalités de l'Internet d'aujourd'hui est que les paquets se perdent. Ils quittent la machine émettrice et ne sont jamais reçus par la machine réceptrice. Il y a de nombreuses causes à cela (abandon du paquet par un routeur surchargé, par exemple), mais ce RFC 2680[1], comme les autres documents du groupe de travail IPPM <http://tools.ietf.org/wg/ippm/> se focalise sur la mesure du phénomène, pas sur ses causes. Il définissait donc une **métrique** « perte de paquet » permettant de comparer des mesures entre elles en sachant qu'on parle bien de la même chose. (Il a depuis été remplacé par le RFC 7680.)

Comme les autres RFC décrivant des métriques, des grandeurs rigoureusement définies et qu'on va mesurer, il s'appuie sur les définitions et le vocabulaire du RFC 2330. Par ailleurs, il suit de très près le plan du RFC 2679, qui spécifiait la mesure du délai d'acheminement d'un paquet. Cette fois, ce qui est défini est une mesure **binaire** (un paquet est perdu ou bien ne l'est pas), `Type-P-One-way-Loss`, puis une statistique pour le cas où il y a plusieurs paquets, le taux de perte moyen. (Petit rappel : `Type-P` signifie que le rapport de mesure **doit** indiquer le type du paquet - protocole de transport, port, etc - car le résultat peut en dépendre. Cf. section 2.8.1.)

Pourquoi cette métrique est-elle utile ? La section 1.1 rappelle l'intérêt de connaître les pertes :

— Certaines applications, notamment interactives, se comportent mal (ou pas du tout) si le taux de pertes dépasse un certain seuil.
— Les applications plus ou moins temps-réel aiment encore moins les pertes de paquets que les autres applications.
— Les protocoles de transport comme TCP compensent les pertes en réémettant les paquets mais un taux de pertes trop élevé les empêchera d'atteindre leur débit maximum.

Mais pourquoi mesurer les pertes sur un chemin aller-simple ("*one-way*") plutôt que aller-retour ("*two-way*") ? La célébrissime commande ping affiche en effet des pertes après un aller-retour (ici 57 %) :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc2680.txt`

```
% ping -c 19 198.51.100.80
PING 198.51.100.80 (198.51.100.80) 1450(1478) bytes of data.
1458 bytes from 198.51.100.80: icmp_seq=1 ttl=46 time=168 ms
1458 bytes from 198.51.100.80: icmp_seq=5 ttl=46 time=167 ms
1458 bytes from 198.51.100.80: icmp_seq=6 ttl=46 time=167 ms
1458 bytes from 198.51.100.80: icmp_seq=9 ttl=46 time=169 ms
1458 bytes from 198.51.100.80: icmp_seq=10 ttl=46 time=167 ms
1458 bytes from 198.51.100.80: icmp_seq=13 ttl=46 time=168 ms
1458 bytes from 198.51.100.80: icmp_seq=15 ttl=46 time=168 ms
1458 bytes from 198.51.100.80: icmp_seq=18 ttl=46 time=167 ms
--- 198.51.100.80 ping statistics ---
19 packets transmitted, 8 received, 57% packet loss, time 18013ms
rtt min/avg/max/mdev = 167.407/168.034/169.066/0.639 ms
```

Mais les mesures aller-retour ont bien des limites :

— Si le chemin est asymétrique, on mesure en fait les performances de deux chemins, l'aller et le retour, qui n'ont pas forcément les mêmes caractéristiques. Même si le chemin est symétrique (passage par les mêmes routeurs à l'aller et au retour), rien ne dit que les résultats soient les mêmes dans les deux sens : files d'attente différentes, QoS peut-être réglée différemment, etc.
— Beaucoup d'applications, par exemple les transferts de fichiers, voient leurs performances dépendre essentiellement d'un seul chemin (pour un transfert de fichiers, celui que suivent les données, pas le chemin inverse par lequel ne transitent que les petits accusés de réception).

Mais les mesures aller-simple sont plus difficiles à effectuer entre autres parce qu'elles ont souvent besoin d'horloges synchronisées (section 1.2). Le principe de la mesure de notre métrique est en effet d'émettre un paquet depuis la machine source à un temps T et de l'attendre à la machine destination jusqu'au temps T + t (où t est le délai qu'on accepte d'attendre). Si les deux machines ne sont pas synchronisées, leurs mesures de T vont différer, faussant ainsi les résultats. La section 1.2 rappelle donc le vocabulaire à utiliser pour évaluer la synchronisation. Les gourous de l'horlogerie verront qu'il est différent de celui des documents UIT comme le G.810 <http://www.itu.int/rec/T-REC-G.810/e>, « "*Definitions and terminology for synchronization networks*" ».

— Synchronisation ("*synchronization*") signifie que deux horloges sont d'accord entre elles sur l'heure qu'il est ("*time error*" pour l'UIT).
— Correction ("*accuracy*") désigne le degré d'accord entre une horloge et la vraie heure UTC ("*time error from UTC*" pour l'UIT). Deux horloges peuvent donc être synchronisées et néanmoins incorrectes.
— Résolution ("*resolution*") est la précision de l'horloge. Certains vieux Unix n'avancent ainsi l'horloge que toutes les dix ms et sa résolution est donc de 10 ms (cela se voyait bien avec la commande ping, qui n'affichait que des RTT multiples de 10). L'UIT dit "*sampling period*".
— Décalage ("*skew*") est le changement dans la synchronisation ou la correction. Il se produit lorsque l'horloge va plus ou moins vite qu'elle ne le devrait. L'UIT appelle cela "*time drift*".

Une fois ces préliminaires achevés, la section 2 décrit la métrique principale de notre RFC, Type-P-One-way-Pack
Sa valeur est simplement 0 lorsque le paquet est arrivé et 1 autrement.

Il y a bien sûr davantage de choses à dire sur cette métrique. Par exemple (section 2.5), faut-il distinguer le cas où un paquet a vraiment été perdu et le cas où il est simplement arrivé en retard, après l'expiration du délai ? En théorie, on devrait attendre 255 secondes, la durée de vie maximale d'un paquet IP (RFC 791, section 3.2). En pratique, on attendra moins longtemps : après tout, pour beaucoup d'applications, un paquet en retard n'a aucun intérêt, on peut aussi bien le considérer comme perdu. C'est l'approche retenue ici.

Et si le paquet arrive corrompu, le considère-t-on comme perdu ? Là encore, oui, pas de distinction. En effet, si le paquet est corrompu, on ne peut même pas être sûr qu'il était bien le paquet attendu, puisque les bits qui permettent de le reconnaître sont peut-être ceux qui ont été changés.

Même chose si le paquet est fragmenté et que certains des fragments n'arrivent pas à tout. On ne peut pas reconstituer le paquet, on le considère comme perdu. En revanche, la duplication, elle, n'est pas considérée comme une perte.

Notre RFC 2680 décrit une métrique (une grandeur définie rigoureusement), pas une méthodologie de mesure, encore moins un protocole. Toutefois, la section 2.6 donne des indications sur ce que pourrait être une telle méthodologie. Le mécanisme recommandé est de mettre une estampille temporelle dans le paquet émis, et de regarder à l'arrivée si on détecte le paquet au bout d'un temps « raisonnable ». À noter que cette méthode n'implique pas une stricte synchronisation des horloges entre les deux machines. On est loin d'un protocole complet (je n'ai pas l'impression qu'il ait jamais été mis au point) et, par exemple, on n'indique pas comment la destination sait qu'elle doit s'attendre à voir arriver un paquet.

Toute mesure implique des erreurs et des incertitudes et la section 2.7 les analyse. D'abord, si les horloges ne sont pas synchronisées du tout, un paquet peut être déclaré comme perdu à tort (si l'émetteur a une horloge qui retarde, le paquet arrivera tard et le destinataire aura pu s'impatienter et le considéré perdu). Même problème si le délai d'attente n'est pas raisonnable, si le destinataire renonce trop vite. Ces deux problèmes peuvent être évités en synchronisant à peu près les horloges (il suffit que leur écart soit petit par rapport au délai d'attente) et en choisissant bien le délai (par exemple, sur une liaison utilisant un satellite géostationnaire, la finitude de la vitesse de la lumière impose un délai d'attente **minimum** de 240 ms - 2 * 35 786 / 300 000).

Une troisième source d'erreur est plus subtile : le paquet peut arriver jusqu'à la machine de destination (donc le réseau fonctionne bien) mais celle-ci le rejeter car **ses** ressources (par exemple les tampons d'entrée/sortie) sont pleines. Pour éviter de compter à tort des paquets comme perdus, il faut s'assurer que la machine de mesure a des ressources suffisantes pour traiter tous les paquets.

La métrique présentée en section 2 était pour **un** paquet. La section 3 définit une métrique supplémentaires, `Type-P-One-way-Packet-Loss-Poisson-Stream` pour le cas où on utilise plusieurs paquets. Et la section 4 s'en sert pour définir une statistique utile. `Type-P-One-way-Packet-Loss-Average` (section 4.1) est le taux de pertes moyen. Si on envoie cinq paquets et que quatre arrivent, elle vaut 0,2 (c'est ce qu'affiche ping sous l'intitulé "*% packet loss*").

Cette moyenne n'est pas toujours facile à évaluer. Ainsi, sur un lien Internet typique, le taux de pertes est bas (nettement moins de 1 %). Pour obtenir une valeur statistiquement significative, il faut souvent tester avec des centaines de paquets. Comme le note la section 5, consacrée à la sécurité, c'est un problème courant des mesures actives : elles peuvent perturber le réseau qu'elle observent.

Ce RFC 2680 a par la suite été évalué dans le RFC 7290, avec des résultats positifs. Cela a mené à son évolution en RFC 7680