RFC 9097 : Metrics and Methods for One-Way IP Capacity

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 10 novembre 2021

Date de publication du RFC : Novembre 2021

https://www.bortzmeyer.org/9097.html

Ce RFC revisite la notion de capacité <https://www.bortzmeyer.org/capacite.html> du RFC 5136[1] et spécifie une nouvelle métrique, Type-P-One-way-Max-IP-Capacity.

Il y a déjà plusieurs RFC qui spécifient rigoureusement des métriques pour déterminer la capacité <https://www.bortzmeyer.org/capacite.html> réseau. (Attention, les publicités des FAI et les articles dans les médias parlent souvent à tort de débit pour désigner ce qui est en fait la capacité, c'est-à-dire le débit maximum.) Une définition rigoureuse est en effet nécessaire car il peut y avoir, pour une même connexion et un même chemin, des capacités différentes selon ce qu'on mesure. Par exemple, est-ce qu'on compte les bits/s à la couche 3 ou à la couche 7 ? Compte-tenu de l'encapsulation et des retransmissions, la seconde est forcément plus basse. Les définitions actuelles sont surtout dans le RFC 5136 et le RFC 3148 mais le travail n'est jamais terminé et ce nouveau RFC vient donc compléter le RFC 5136. Vu l'importance de cette notion de capacité dans les publicités (« Nouveau : accès Premium Gold Plus à 10 Gb/s ! »), il est crucial que des associations de consommateurs ou des régulateurs puissent mesurer et vérifier les annonces.

Une des raisons pour lesquelles le travail sur la définition des métriques ne cesse jamais est que l'Internet évolue. Ainsi, note le RFC, depuis quelques années :

— Ce n'est plus forcément le premier kilomètre qui limite la capacité du chemin,
— le protocole UDP prend plus d'importance, grâce à WebRTC et QUIC <https://www.bortzmeyer.org/quic.html>,
— avec les CDN, le contenu se rapproche des utilisateurs (et franchit moins souvent des frontières entre AS).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5136.txt

Première métrique défnie dans notre RFC (section 5), et venant s'ajouter à celles du RFC 5136, `Type-P-One-way-IP-Capacity`. C'est le nombre de bits par seconde, mesurés en couche 3 et au-dessus (donc incluant l'en-tête IP, par exemple). Le `Type-P` est là pour rappeler que cette métrique n'a de sens que pour un certain type de trafic, car des équipements réseau « intelligents » peuvent faire que, par exemple, les paquets vers le port 22 passent plus vite que ceux pour un autre port. `Type-P` est donc la description complète des paquets utilisés. À noter que le RFC recommande d'indiquer la valeur de cette métrique en mégabits par seconde (et pas en mébibits, on utilise plutôt les préfixes des télécoms que ceux de l'informatique).

Deuxième métrique (section 6), `Type-P-One-way-Max-IP-Capacity`, qui indique la capacité maximum (la différence avec la métrique précédente vient du fait que certains réseaux peuvent avoir une capacité variable, par exemple les réseaux radio, oui, je sais, c'est compliqué).

Troisième métrique (section 7), `Type-P-IP-Sender-Bit-Rate`. Elle désigne la capacité de l'émetteur à envoyer des bits. En effet, lors d'une mesure, le goulet d'étranglement peut être l'expéditeur et on croit alors que le réseau a une capacité inférieure à ce qu'elle est réellement.

La section 8 se penche sur la mesure effective de ces valeurs. Il faut une mesure active (envoi de bits sur le réseau uniquement pour faire la mesure), et elle possiblement très perturbatrice puisqu'on va chercher à remplir les tuyaux le plus possible. Le RFC inclut un algorithme d'ajustement du trafic mais qui n'est **pas** un vrai algorithme de contrôle de congestion. Le pseudo-code de cet algorithme est dans l'annexe A.

La mesure est bidirectionnelle (envoyeur et récepteur doivent coopérer) même si la métrique est unidirectionnelle (le `One-Way` dans le nom). Le RFC recommande de la faire sur UDP (attention au RFC 8085, UDP n'ayant pas de contrôle de congestion propre, c'est à l'application de mesure de faire attention à ne pas écrouler le réseau).

Jouons maintenant un peu avec une mise en œuvre de ce RFC, le programme udpst `<https://github.com/BroadbandForum/obudpst>`, sur deux machines Arch Linux (un Raspberry Pi 1, donc avec un réseau très lent) et Debian, sur le même `commutateur`. On l'installe :

```
% git clone https://github.com/BroadbandForum/obudpst.git
% cd obudpst
% cmake .
% make
```

On peut alors lancer le serveur :

```
% ./udpst
UDP Speed Test
Software Ver: 7.2.1, Protocol Ver: 8, Built: Sep 28 2021 15:46:40
Mode: Server, Jumbo Datagrams: Enabled, Authentication: Available, sendmmsg syscall: Available
```

Et le client, le Raspberry Pi :

```
% ./udpst -u  2001:db8:fafa:35::1
UDP Speed Test
Software Ver: 7.2.1, Protocol Ver: 8, Built: Sep 28 2021 18:25:23
Mode: Client, Jumbo Datagrams: Enabled, Authentication: Available, sendmmsg syscall: Available
Upstream Test Interval(sec): 10, DelayVar Thresholds(ms): 30-90 [RTT], Trial Interval(ms): 50, Ignore OoO/Dup: D
    SendingRate Index: <Auto>, Congestion Threshold: 3, High-Speed Delta: 10, SeqError Threshold: 10, IPv6 TClas
...
Sub-Interval[10](sec): 10, Delivered(%): 100.00, Loss/OoO/Dup: 0/0/0, OWDVar(ms): 2/8/18, RTTVar(ms): 2-16, Mbps
Upstream Summary Delivered(%): 100.00, Loss/OoO/Dup: 0/0/0, OWDVar(ms): 0/3/23, RTTVar(ms): 0-16, Mbps(L3/IP): 4
Upstream Minimum One-Way Delay(ms): 2 [w/clock difference], Round-Trip Time(ms): 1
Upstream Maximum Mbps(L3/IP): 67.29, Mbps(L2/Eth): 67.45, Mbps(L1/Eth): 67.63, Mbps(L1/Eth+VLAN): 67.67
```

En sens inverse, avec l'option `-d`, où le Raspberry Pi va envoyer des données :

```
Downstream Summary Delivered(%):  84.90, Loss/OoO/Dup: 8576/0/0, OWDVar(ms): 0/641/956, RTTVar(ms): 0-39, Mbps(L
Downstream Minimum One-Way Delay(ms): -927 [w/clock difference], Round-Trip Time(ms): 1
Downstream Maximum Mbps(L3/IP): 46.98, Mbps(L2/Eth): 47.68, Mbps(L1/Eth): 48.46, Mbps(L1/Eth+VLAN): 48.62
```

(Notez le taux de pertes élevé, la pauvre machine n'arrive pas à suivre.)