RFC 5357 : A Two-way Active Measurement Protocol (TWAMP)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 7 octobre 2008

Date de publication du RFC : Octobre 2008

https://www.bortzmeyer.org/5357.html

Aujourd'hui, l'administrateur réseau qui teste ses problèmes de performance utilise en général ping pour mesurer le RTT, le temps d'aller-retour avec la machine visée. Demain, utilisera t-il un programme nommé `twping` et le vieux `ping` sera t-il dépassé ? C'est en tout cas ce que visent les auteurs de ce protocole de mesure de RTT. (ping restera utile pour les tests de bon fonctionnement.)

ping, qui repose sur l'envoi de paquets ICMP `echo request` et `echo reply` (cf. RFC 792 [1]) a en effet des limites, décrites dans le RFC 2681, section 2.6 : notamment, il est difficile de savoir si les paquets ICMP ont été traités rapidement ou pas par la machine qui répond (un routeur Cisco renvoie la réponse très lentement, car elle doit passer par le processeur généraliste du routeur). En outre, il n'est pas possible de réserver l'usage de ces paquets `echo` à certains, sauf à filtrer par adresse IP (TWAMP, lui, permet l'authentification).

TWAMP ("*Two-way Active Measurement Protocol*") hérite directement d'OWAMP ("*One-way Active Measurement Protocol*", RFC 4656). Il reprend beaucoup de ses concepts, notamment la séparation en deux protocoles, celui de contrôle et celui de test. Notre RFC 5357 est d'ailleurs largement écrit sous forme de différences entre OWAMP et TWAMP, on ne peut donc pas implémenter ce dernier sans avoir lu le RFC 4656.

TWAMP met donc en œuvre la métrique décrite dans le RFC 2681 pour les mesures bidirectionnelles. Ces mesures ont l'avantage de ne pas dépendre de la présence d'une horloge correcte sur les deux machines et l'inconvénient de ne pas pouvoir séparer les contributions du trajet aller et du trajet retour. Un des deux est peut-être bien plus lent que l'autre mais les mesures bidirectionnelles ne pourront pas le détecter.

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc792.txt`

Comme l'explique la section 1.2, un système TWAMP peut comprendre jusqu'à quatre machines mais on peut supposer que, la plupart du temps, il n'y en aura que deux, la "*Control-Client/Session-Sender*" et la "*Server/Session-Reflector*". Le programme sera lancé sur la première, qui émettra les paquets qui seront renvoyés par la seconde, permettant de calculer le RTT.

La section 2 décrit le protocole : le "*Control-Client*" établit une connexion avec le "*Server*" sur le port 862, ils se mettent d'accord sur le test, puis le "*Session-Sender*" envoie les paquets au "*Session-Reflector*".

La section 3 décrit le protocole de contrôle, très proche de celui d'OWAMP. La section 4 est consacrée au protocole de test, dont la différence avec son prédécesseur est plus marquée, puisqu'il faut désormais renvoyer les paquets reçus. La procédure exacte suivie par le "*Reflector*" est décrite en 4.2. Le réflecteur renvoie la date d'arrivée, celle d'émission de la réponse et d'autres informations (section 4.2.1). Comme la réponse contient le TTL tel qu'il était à l'arrivée, une implémentation de TWAMP doit pouvoir accéder aux en-têtes IP, ce qui n'est pas pas toujours simple <https://www.bortzmeyer.org/ip-header-set.html>. Comme avec OWAMP, le diable est dans les détails et l'implémentation doit être faite très soigneusement pour limiter les erreurs et imprécisions.

TWAMP a depuis, bénéficié d'extensions au protocole de base, décrites dans le RFC 6038.

À noter que ce RFC est, je crois, le premier RFC qui normalise la prononciation du sigle du protocole (section 1.3) : « ti-ouampe ».

Il n'existe pas actuellement de mise en œuvre de TWAMP en logiciel libre. Compte-tenu de la proximité du protocole avec OWAMP, le meilleur moyen d'en écrire une serait sans doute de partir du programme owamp <http://e2epi.internet2.edu/owamp/>.

Pour une critique vigoureuse de ce RFC (pas du protocole, mais de la rédaction du RFC), voir « "*How NOT to Write a Protocol Specification*" <http://blog.dustintrammell.com/2008/11/17/how-not-to-write-a-protocol-specification/> », de Dustin D. Trammell.