RFC 3489 : STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 juin 2007

Date de publication du RFC : Mars 2003

https://www.bortzmeyer.org/3489.html

Le NAT a toujours été une des plaies de l'Internet, entre autres parce qu'il perturbe les applications qui veulent transmettre une référence à leur adresse IP. STUN, décrit dans ce RFC (depuis mis à jour dans le RFC 5389 [1]), est un des protocoles qui permet de limiter les dégâts.

Pour plusieurs raisons, dont le manque d'adresses IPv4, de nombreuses machines sont connectées à l'Internet derrière un routeur qui fait du NAT. Leur adresse étant alors privée, elles ne peuvent pas la transmettre à l'extérieur, ce qui est une gêne considérable pour tous les protocoles qui reposent sur la référence à des adresses IP, comme SIP. SIP lui-même (RFC 3261) marche à travers le NAT mais les données audio ou vidéo transmises (typiquement par RTP) le sont à une adresse IP que l'appelant donne à l'appelé et, si cette adresse est privée, le flux n'arrivera jamais.

La bonne solution serait de déployer IPv6, qui fournit des adresses en quantité illimitée, ou à la rigueur de permettre un meilleur contrôle du routeur NAT avec MIDCOM (RFC 3303) mais, en attendant, STUN est une solution simple et qui marche souvent.

STUN résout partiellement le problème de la manière suivante : un serveur STUN doit être installé quelque part sur l'Internet public (il existe des serveurs STUN publics <http://www.vovida.org/applications/downloads/stun/>) et reçoit des demandes envoyées en UDP au port 3478. Le serveur STUN peut alors connaitre l'adresse publique, celle mise par le routeur NAT et, en répondant au client, il pourra informer celui-ci « Voilà à quoi tes paquets ressemblent, vus de l'extérieur ».

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5389.txt

Le nom du serveur STUN est typiquement configuré dans le client, puis le serveur recherché dans le DNS via les enregistrements SRV (RFC 2782).

Le principe est simple, mais le RFC est plus compliqué que cela, notamment en raison des problèmes de sécurité (la section 12 du RFC est très détaillée sur ce point). Par exemple, le client STUN doit en fait commencer par une connexion TCP pour obtenir un mot de passe temporaire.

Un autre problème est qu'il y a en fait plusieurs sortes de NAT (voir par exemple le RFC 2663). Notre RFC en rappelle certaines dans la section 5 avec sa propre terminologie (il n'existe pas réellement de norme pour le NAT). Par exemple, certains routeurs NAT ("*Symmetric NAT*" pour notre RFC) n'attribuent pas l'adresse externe uniquement en fonction de la **source** mais aussi en fonction de la **destination**. Ceux-ci ne fonctionnent pas avec STUN, qui a besoin de NAT "*cone*", c'est-à-dire où les paquets d'une même machine auront toujours la même adresse IP externe.

Voici une démonstration d'un client STUN <http://sourceforge.net/projects/stun/> qui, en se connectant à un serveur STUN public <http://www.voip-info.org/wiki-STUN> va apprendre son adresse IP publique :

```
% ifconfig -a
hme0: [...] inet 172.19.1.2
(Adresse privée, probablement NATée...)

% ./client  stun.ekiga.net -v|& more
STUN client version 0.96
...
MappedAddress = 213.41.181.9:32143
(Voici mon adresse telle que vue de l'extérieur)
...
Primary: Indepndent Mapping, Port Dependent Filter, preserves ports, no hairpin
(Et le type de NAT, à noter que cette partie disparaitra du successeur de STUN,
actuellement en cours d'élaboration, car il existait trop de types de NAT, et
leur détection n'était pas toujours fiable.)
```

STUN s'inscrit dans la catégorie des protocoles de « réparation » du NAT, catégorie décrite dans le RFC 3424. Il a été depuis asse profondément modifié dans le RFC 5389.