RFC 6967 : Analysis of Solution Candidates to Reveal a Host Identifier (HOST_ID) in Shared Address Deployments

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 juin 2013

Date de publication du RFC : Juin 2013

https://www.bortzmeyer.org/6967.html

Lorsque plusieurs machines sont situées derrière un routeur NAT, leurs pairs sur l'Internet ne peuvent pas facilement les distinguer. Tout le trafic vient de la même adresse IP, l'adresse publique allouée au routeur NAT, et c'est gênant pour certaines applications. Par contre, cela est positif d'un autre point de vue : cela fournit un certain niveau de masquage des machines individuelles, qui peut aider à protéger sa vie privée. Pris entre ces deux exigences contradictoires, l'IETF propose dans ce RFC une analyse des techniques permettant de révéler la machine individuelle derrière le routeur NAT, sans prendre position sur le fait de savoir si cette révélation est une bonne ou une mauvaise chose.

L'idée est que chaque machine a une identité unique (notée "*HOST_ID*") et que les techniques étudiées ici doivent permettre de communiquer ce "*HOST_ID*" au pair avec qui on parle sur l'Internet. Autrefois, l'adresse IPv4 pouvait servir de "*HOST_ID*" mais cela ne marche plus désormais en raison de la prévalence du NAT. Le cas du petit routeur NAT à la maison n'est pas forcément le plus intéressant (toutes les machines appartiennent au même foyer) mais le problème est plus crucial avec les CGN, où des machines n'ayant aucun lien entre elles se partagent la même adresse IP, ce qui a des tas d'effets néfastes (RFC 6269 [1]) : impossible de mettre en liste noire la machine coupable, par exemple, sans risquer de gêner des utilisateurs n'ayant commis d'autre délit que de partager l'adresse IP d'un méchant. Même problème avec des techniques comme l'A+P du RFC 6346.

J'ai dit plus haut que le but était d'avoir des "*HOST_ID*" uniques. En fait, il doivent être uniques par adresse IP publique. Deux routeurs NAT qui utilisent des adresses IPv4 publiques différentes peuvent générer des "*HOST_ID*" identiques, seul le couple {adresse IPv4 publique, "*HOST_ID*"} a besoin d'être unique au niveau mondial (section 2 du RFC).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc6269.txt

À ce stade, rappelez-vous que ce RFC présente un problème en terme très général et expose des solutions possibles. Par exemple, le "*HOST_ID*" peut figurer dans chaque paquet IP ou bien uniquement au début d'une session. Et il peut être ajouté par l'engin qui gère le partage d'adresse (le routeur NAT ou CGN) ou bien par la machine originale (qui pourra donc mentir, mais c'est pareil avec l'adresse IP source). Enfin, bien des identificateurs peuvent être utilisés comme "*HOST_ID*" (l'adresse IP à la source, les seize bits de poids le plus faible de l'adresse IP, un identifiant opaque compris uniquement par le FAI de départ, un VLAN ID, etc).

Une telle fonction de révélation des identités soulève évidemment des problèmes liés à la protection de la vie privée. Certains voient en effet dans le NAT des avantages en terme de sécurité <https://www.bortzmeyer.org/nat-et-securite.html> comme, via le partage d'adresses, le fait que l'observateur extérieur ne puisse pas savoir quelle machine exactement, parmi toutes les machines situées derrière le routeur NAT, il est en train d'observer. En fournissant un "*HOST_ID*" à l'extérieur, on perdrait cette intimité. Notez qu'"*HOST_ID*" ramènerait juste au cas où on utilise une adresse IP publique sur sa machine, ni plus, ni moins. Notamment, un "*HOST_ID*", comme une adresse IP publique, ne serait pas forcément permanent (lisez le RFC 8981 pour un exemple d'adresses publiques temporaires préservant la vie privée). Ainsi, une machine en redémarrant pourrait acquérir un nouvel "*HOST_ID*".

Pour limiter les risques, ce RFC suggère deux règles : que les "*HOST_ID*" ne soient uniques que localement, pas globalement. Et qu'ils soient toujours temporaires. Cette question de la vie privée avait été la plus chaude lors des discussions à l'IETF sur le futur RFC, et le document a beaucoup évolué vers davantage de prise en compte du problème de la protection de la vie privée. Alissa Cooper, du CDT et auteure du RFC 6462 avait été particulièrement critique sur les premières versions de ce document, réclamant des modifications comme le fait que le "*HOST_ID*" ne soit pas forcément unique mondialement.

D'autre part, les logiciels qui visent à mettre en œuvre des mécanismes de dissimulation, afin de protéger la vie privée, comme Tor, supprimeront évidemment le "*HOST_ID*", comme ils suppriment aujourd'hui d'autres informations bien plus révélatrices. Regardez le Panopticlick <https://panopticlick.eff.org/> pour voir qu'on est facilement identifié de manière unique lorsqu'on navigue sur le Web, même derrière un routeur NAT.

La section 4 de notre RFC présente ensuite toutes les solutions étudiées pour transmettre l'"*HOST_ID*". Certaines sont clairement mauvaises et ne sont mentionnées que pour documenter les raisons de leur rejet. D'autres méritent davantage d'étude.

D'abord, utiliser le champ Identification des paquets IP. Ce champ est surtout connu pour servir lors du réassemblage des datagrammes mais on peut théoriquement s'en servir pour « exfiltrer » de l'information sur une machine. Ce champ fait seize bits, ce qui est suffisant pour identifier une machine. Cela empêche évidemment de s'en servir pour le réassemblage après fragmentation (voir l'excellent RFC 6864 si vous voulez tout comprendre sur ce champ). Comme il n'est pas toujours facile pour le routeur NAT de garantir que les datagrammes ne seront jamais fragmentés, ni avant, ni après, cette solution est déconseillée par notre RFC.

On peut alors imaginer de définir une nouvelle option IP, conçue uniquement pour notre but d'identification, et dont l'usage n'entrerait pas en conflit avec des usages existants. Elle permettrait de transporter l'information qu'on veut. (Cela avait été documenté dans l'"*Internet-Draft*" draft-chen-intarea-v4-uid-hea abandonné depuis.) Cette solution, comme la précédente, a l'avantage d'être indépendante du protocole de transport. Elle pose quelques problèmes techniques (que faire si le paquet a une taille proche de la MTU et qu'on veut y ajouter cette option ? que faire si l'espace pour les options, qui a une taille maximale en IPv4, est plein ?) Mais elle a surtout comme défaut que les options IP sont souvent bloquées sur l'Internet <https://www.bortzmeyer.org/options-interdites.html> (voir aussi « "*Measuring interactions between transport protocols and middleboxes*" <http://conferences.sigcomm.org/imc/

2004/papers/p336-medina.pdf> » par Alberto Medina, Mark Allman et Sally Floyd). Le RFC la regarde donc comme non viable.

On peut alors monter d'une couche et choisir une nouvelle option TCP, qui indiquerait le "*HOST_ID*". (Documenté dans l'"*Internet-Draft*" draft-wing-nat-reveal-option qui a été abandonné par la suite.) Cela aurait l'avantage qu'on n'aurait pas à la répéter dans chaque paquet mais uniquement dans le paquet SYN. Cela aiderait à résoudre les problèmes de MTU (le paquet SYN est petit). Et les mesures indiquent que les options TCP sont bien mieux respectées que les options IP sur l'Internet (voir l'article de Medina, Allman et Floyd cité plus haut, ainsi que l'"*Internet-Draft*" draft-abdo-hostid-tcpopt-implementation). Mais cette solution ne marcherait évidemment pas pour les autres protocoles de transport. Et il y a quelques problèmes techniques (une vision générale est dans l'article de Honda, M., Nishida, Y., Raiciu, C., Greenhalgh, A., Handley, M. et H. Tokuda, « "*Is it still possible to extend TCP?*" <http://nrg.cs.ucl.ac.uk/mjh/tmp/mboxes.pdf> »). Par exemple, l'espace pour les options a une taille maximum dans TCP.

Si on n'a pas trouvé son bonheur dans la couche 3 (champ ID d'IP ou option IP), ni dans la couche 4 (option TCP), les couches supérieures seront-elles plus accueillantes ? Pourquoi ne pas transmettre l'information dans les applications ? Par exemple, pour HTTP, l'en-tête Forwarded: (RFC 7239) pourrait être l'endroit idéal pour cela. L'engin qui fait le partage d'adresses pourrait ainsi ajouter à la requête HTTP :

```
Forwarded: for=192.168.13.1
```

Il existait avant un en-tête non-standard, X-Forwarded-For:, dit aussi XFF, qui faisait quelque chose de ce genre. Comme les autres solutions au problème de la révélation du "*HOST_ID*", elle permet la triche. C'est pour cela que Wikipédia maintient une liste <http://meta.wikimedia.org/wiki/XFF_project#Trusted_XFF_list> des FAI à qui on peut faire confiance pour mettre de l'information XFF correcte. Wikipédia peut alors utiliser l'information dans ce champ pour attribuer les responsabilités (de vandalisme, par exemple, ou d'introduction délibérée d'informations fausses) à la bonne adresse IP. La principale limite de cette solution est qu'elle ne marche que pour certains protocoles, comme HTTP. Pour les autres, il faudrait au fur et à mesure concevoir une extension permettant de diffuser cette information. Même pour HTTP, elle nécessite de modifier les requêtes en cours de route, ce qui est impossible avec HTTPS.

Pour éviter de devoir développer une extension ou un en-tête spécifique par protocole applicatif, la solution PROXY a été proposée <http://haproxy.1wt.eu/download/1.5/doc/proxy-protocol.txt>. Elle consiste à ajouter l'information juste avant les données de la couche Application. L'information a cette allure (au moment du partage d'adresses, 192.168.13.1:56324 voulait parler à 192.0.2.15:443) :

```
PROXY TCP4 192.168.13.1 192.0.2.15 56324 443\r\n
```

En la recevant, le serveur a juste à retirer cette ligne et à passer ce qui suit à l'application. En pratique, cette solution n'est pas réaliste car elle casse la compatibilité : un client qui a PROXY ne peut pas parler à un serveur qui ne l'a pas (l'opposé est vrai, par contre). Sans même parler des pare-feux qui n'accepteront pas ce qui leur semblera une requête invalide. Bref, elle n'est envisageable que dans des environnements fermés.

Le NAT tel que déployé aujourd'hui dans le monde IPv4 ne mérite pas réellement son nom. Il ne traduit pas une adresse pour une autre mais un couple {adresse, port} pour un autre. Il devrait plutôt

être qualifié de NAPT ("*Network Address and Port Translation*"). Peut-on utiliser les ports alloués pour transporter l'information sur le "*HOST_ID*" ? L'idée est d'allouer un jeu de ports à chaque machine située derrière le routeur NAPT. Par exemple, imaginons trois machines derrière le routeur, celui-ci utilise les ports publics de 1 024 à 20 000 pour la première machine, de 20 001 à 40 000 pour la seconde et de 40 001 à 65 535 pour la troisième (voir l'"*Internet-Draft*" `draft-donley-behave-deterministic-cgn`). Il ne reste plus ensuite qu'à publier cette information à l'extérieur (le RFC est muet sur ce point). Le RFC 6346 propose une solution de ce genre. Malheureusement, cela contredit partiellement les exigences de sécurité des RFC 6056 et RFC 6269 qui demandent des ports sources imprévisibles.

Plus disruptive est la solution qui consiste à utiliser un protocole nouveau, au dessus d'IP, le protocole HIP <`https://www.bortzmeyer.org/hip-resume.html`>. HIP a déjà cette notion de "*HOST_ID*" (l'identité d'une machine est une clé cryptographique, ce qui résout bien des problèmes de sécurité). Mais cela nécessite que le client et le serveur parlent HIP tous les deux, ce qui parait irréaliste. À la rigueur, si tous les serveurs parlaient HIP, le routeur NAT pourrait se contenter de prendre des requêtes non-HIP et de les traduire en HIP. Mais tous les serveurs ne parlent pas HIP, loin de là.

Bon, si on n'arrive pas à mettre le "*HOST_ID*" dans le canal normal de communication, peut-on le mettre à l'extérieur ? Par exemple, peut-on utiliser ICMP pour signaler l'adresse originale d'une communication en cours ? Dans cette idée, le routeur NAT, en même temps que le paquet TCP initial vers le serveur, enverrait une requête ICMP contenant l'information utile. Cela avait été documenté dans l'"*Internet-Draft*" `draft-yourtchenko-nat-reveal-ping`, qui a expiré depuis. Cela marcherait pour tous les protocoles ayant la notion de port (puisque c'est ce qu'on mettrait dans le paquet ICMP pour indiquer de quelle session on parle). Et un problème éventuel sur cette fonction de révélation de l'"*HOST_ID*" (par exemple de l'ICMP bloqué) ne gênerait pas la session normale (contrairement aux options IP ou TCP). Par contre, on n'est pas sûr que ce soit la même machine qui reçoive l'ICMP et qui reçoive les autres paquets (en cas d'utilisation de répartiteurs de charge). Et cette technique rend plus difficile l'examen de l'"*HOST_ID*" par des intermédiaires, genre IDS.

Toujours dans la série « communications hors bande », l'idée de ressortir le vieux protocole IDENT (RFC 1413), qui récupérerait le "*HOST_ID*" par une connexion TCP vers la source d'une session. Il faudra donc un serveur IDENT dans la machine faisant le partage d'adresses et publier cette information (dans le DNS ?) pour éviter que les serveurs ne bombardent les pauvres CPE de requêtes IDENT. Il y a aussi des craintes quant aux performances de cette solution. Et quant à sa sécurité la requête IDENT pourrait être émise par un méchant.

Voilà, on a fait le tour de toutes les solutions envisageables. La section 5 de notre RFC présente une synthèse sous forme d'un tableau dont les lignes sont les solutions présentées plus haut et les colonnes sont les critères d'évaluation (« nécessite une modification de TCP/IP dans les machines », « a un impact sur les performances », « déployable aujourd'hui (sans refaire l'Internet de zéro) », etc).