RFC 8981 : Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 mars 2021

Date de publication du RFC : Février 2021

https://www.bortzmeyer.org/8981.html

Une des particularités d'IPv6 est de disposer d'un mécanisme, l'**autoconfiguration sans état** qui permet à une machine de se fabriquer une adresse IP **globale** sans serveur DHCP. Autrefois, ce mécanisme créait souvent l'adresse IP à partir de l'adresse MAC de la machine et une même machine avait donc toujours le même identifiant (IID, "*Interface IDentifier*", les 64 bits les plus à droite de l'adresse IPv6), même si elle changeait de réseau et donc de préfixe. Il était donc possible de « suivre à la trace » une machine, ce qui pouvait poser des problèmes de protection de la vie privée. Notre RFC, qui remplace le RFC 4941 [1] fournit une solution, sous forme d'un mécanisme de création d'adresses **temporaires**, partiellement aléatoires (ou en tout cas imprévisibles). Les changements sont assez importants depuis le RFC 4941, qui traitait un peu les adresses IPv6 temporaires comme des citoyennes de deuxième classe. Désormais, elles sont au contraire le choix préféré.

L'autoconfiguration sans état (SLAAC, pour "*Stateless Address Autoconfiguration*"), normalisée dans le RFC 4862, est souvent présentée comme un des gros avantages d'IPv6. Sans nécessiter de serveur central (contrairement à DHCP), ce mécanisme permet à chaque machine d'obtenir une adresse globale (IPv4, via le protocole du RFC 3927, ne permet que des adresses purement locales) et unique. Cela se fait en concaténant un **préfixe** (par exemple `2001:db8:32:aa12::`), annoncé par le routeur, à un **identifiant d'interface** (par exemple `213:e8ff:fe69:590d`, l'identifiant de mon PC portable - les machines individuelles, non partagées, comme les ordiphones sont particulièrement sensibles puisque la connaissance de la machine implique celle de son maître), typiquement dérivé de l'adresse MAC.

Partir de l'adresse MAC présente des avantages (quasiment toute machine en a une, et, comme elle est relativement unique, l'unicité de l'adresse IPv6 est obtenue facilement) mais aussi un inconvénient :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc4941.txt`

comme l'identifiant d'interface est toujours le même, cela permet de reconnaître une machine, même lorsqu'elle change de réseau. Dès qu'on voit une machine XXXX:213:e8ff:fe69:590d (où XXXX est le préfixe), on sait que c'est mon PC. Un tiers qui écoute le réseau peut ainsi suivre « à la trace » une machine, et c'est pareil pour les machines avec lesquelles on correspond sur l'Internet <https://arstechnica.com/information-technology/2016/02/using-ipv6-with-linux-youve-likely-beer>. Or, on ne peut pas cacher son adresse IP, il faut l'exposer pour communiquer. Le RFC 7721 analyse le problème plus en détail (voir aussi le RFC 7707).

Les sections 1.2 et 2 de notre RFC décrivent la question à résoudre. Elles notent entre autre que le problème n'est pas aussi crucial que l'avaient prétendu certains. En effet, il existe bien d'autres façons, parfois plus faciles, de suivre une machine à la trace, par exemple par les fameux petits gâteaux de HTTP (RFC 6265) mais aussi par des moyens de plus haute technologie comme les métadonnées de la communication (taille des paquets, écart entre les paquets) ou les caractéristiques matérielles <http://www.caida.org/publications/papers/2005/fingerprinting/> de l'ordinateur, que l'on peut observer sur le réseau. Parmi les autres méthodes, notons aussi que si on utilise les adresses temporaires de notre RFC 8981 mais qu'on configure sa machine pour mettre dynamiquement à jour un serveur DNS avec cette adresse, le nom de domaine suffirait alors à suivre l'utilisateur à la trace !

La section 2.2 commence à discuter des solutions possibles. DHCPv6 (RFC 3315, notamment la section 12) serait évidemment une solution, mais qui nécessite l'administration d'un serveur. L'idéal serait une solution qui permette, puisque IPv6 facilite l'existence de plusieurs adresses IP par machine, d'avoir une adresse stable pour les connexions entrantes et une adresse temporaire, non liée à l'adresse MAC, pour les connexions sortantes, celles qui « trahissent » la machine. C'est ce que propose notre RFC, en générant les adresses temporaires selon un mécanisme pseudo-aléatoire (ou en tout cas imprévisible à un observateur extérieur).

Enfin, la section 3 décrit le mécanisme de protection lui-même. Deux algorithmes sont proposés mais il faut bien noter qu'une machine est toujours libre d'en ajouter un autre, le mécanisme est unilatéral et ne nécessite pas que les machines avec qui on correspond le comprennent. Il suffit que l'algorithme respecte ces principes :

— Les adresses IP temporaires sont utilisées pour les connexions **sortantes** (pour les entrantes, on a besoin d'un identificateur stable, un serveur n'a pas de vie privée),
— Elles sont. . .temporaires (typiquement quelques heures),
— Cette durée de vie limitée suit les durées de vie indiquées par SLAAC, et, en prime, on renouvelle les IID (identifiants d'interface) tout de suite lorsque la machine change de réseau, pour éviter toute corrélation,
— L'identifiant d'interface est bien sûr différent pour chaque interface, pour éviter qu'un observateur ne puisse détecter que l'adresse IP utilisée en 4G et celle utilisée en WiFi désignent la même machine,
— Ces adresses temporaires ne doivent pas avoir de sémantique visible (cf. RFC 7136) et ne pas être prévisibles par un observateur.

Le premier algorithme est détaillé dans la section 3.3.1 et nécessite l'usage d'une source aléatoire, selon le RFC 4086. On génère un identifiant d'interface avec cette source (attention, certains bits sont réservés, cf. RFC 7136), on vérifie qu'il ne fait pas partie des identifiants réservés <http://www.iana.org/assignments/ipv6-interface-ids> du RFC 5453, puis on la préfixe avec le préfixe du réseau.

Second algorithme possible, en section 3.3.2, une génération à partir de l'algorithme du RFC 7217, ce qui permet d'utiliser le même algorithme pour ces deux catégories d'adresses, qui concernent des cas d'usage différents. Le principe est de passer les informations utilisées pour les adresses stables du RFC 7217 (dont un secret, pour éviter qu'un observateur ne puisse déterminer si deux adresses IP appartiennent à la même machine), en y ajoutant le temps (pour éviter la stabilité), à travers une PRF comme SHA-256. Plus besoin de générateur aléatoire dans ce cas.

Une fois l'adresse temporaire générée (les détails sont dans la section 3.4), et testée (DAD), il faut la renouveler de temps en temps (sections 3.5 et 3.6, cette dernière expliquant pourquoi renouveler une fois par jour est plus que suffisant). L'ancienne adresse peut rester active pour les connexions en cours mais plus pour de nouvelles connexions.

Notez que le RFC impose de fournir un moyen pour activer ou débrayer ces adresses temporaires, et que cela soit par préfixe IP, par exemple pour débrayer les adresses temporaires pour les adresses locales du RFC 4193.

Maintenant que l'algorithme est spécifié, la section 4 du RFC reprend de la hauteur et examine les conséquences de l'utilisation des adresses temporaires. C'est l'occasion de rappeler un principe de base de la sécurité : il n'y a pas de solution idéale, seulement des compromis. Si les adresses temporaires protègent davantage contre le « flicage », elles ont aussi des inconvénients comme de rendre le débogage des réseaux plus difficile. Par exemple, si on note un comportement bizarre associé à certaines adresses IP, il sera plus difficile de savoir s'il s'agit d'une seule machine ou de plusieurs. (Le RFC 7217 vise justement à traiter ce cas.)

Et puis les adresses temporaires de notre RFC ne concernent que l'identifiant d'interface, le préfixe IP reste, et il peut être révélateur, surtout s'il y a peu de machines dans le réseau. Si on veut vraiment éviter le traçage par adresse IP, il faut utiliser Tor ou une technique équivalente (cf. section 9).

L'utilisation des adresses temporaires peut également poser des problèmes avec certaines pratiques prétendument de sécurité comme le fait, pour un serveur, de refuser les connexions depuis une machine sans enregistrement DNS inverse (un nom correspondant à l'adresse, via un enregistrement PTR). Cette technique est assez ridicule mais néanmoins largement utilisée.

Un autre conflit se produira, note la section 8, si des mécanismes de validation des adresses IP source sont en place dans le réseau. Il peut en effet être difficile de distinguer une machine qui génère des adresses IP temporaires pour se protéger contre les indiscrets d'une machine qui se fabrique de fausses adresses IP source pour mener une attaque.

Quelles sont les différences entre le précédent RFC et celui-ci ? La section 5 du RFC résume les importants changements qu'il y a eu depuis le RFC 4941. Notamment :

- Abandon de MD5 (cf. RFC 6151),
- Autorisation de n'avoir que des adresses temporaires (le RFC 4941 imposait qu'une adresse stable reste présente),
- Autoriser les adresses temporaires à être utilisées par défaut, ce qui était déjà le cas en pratique sur la plupart des systèmes d'exploitation (depuis l'époque du RFC 4941, la gravité de la surveillance de masse est mieux perçue),
- Réduction de la durée de vie recommandée, et recommandation qu'elle soit partiellement choisie au hasard, pour éviter que toutes les adresses soient refaites en même temps,
- Un algorithme de génération des identifiants d'interface, reposant sur une mémoire stable de la machine, a été supprimé,
- Un autre a été ajouté, se fondant sur le RFC 7217,
- Intégration des analyses plus détaillées qui ont été faites de la sécurité d'IPv6 (RFC 7707, RFC 7721 et RFC 7217),
- Correction des bogues <https://www.rfc-editor.org/errata/rfc4941>.

Passons maintenant aux mises en œuvre. L'ancienne norme, le RFC 4941, est appliqué par presque tous les systèmes d'exploitation, souvent de manière plus protectrice de la vie privée, par exemple en étant activé par défaut. Les particularités de notre nouveau RFC ne sont pas toujours d'ores et déjà présentes. Ainsi, il existe des "*patches*" pour FreeBSD <https://www.gont.com.ar/code/fgont-patch-freebsd-rtxt> et pour Linux <https://patchwork.ozlabs.org/project/netdev/patch/20200501035147.GA1587@archlinux-current.localdomain/> mais pas forcément intégrés aux versions officielles. Sur Linux, le paramètre sysctl qui contrôle ce protocole est `net.ipv6.conf.XXX.use_tempaddr` où XXX est le nom de l'interface réseau, par exemple `eth0`. En mettant dans le fichier `/etc/sysctl.conf` :

```
# Adresses temporaires du RFC 8981
net.ipv6.conf.default.use_tempaddr = 2
```

On met en service les adresses temporaires, non « pistables » pour toutes les interfaces (c'est le sens de la valeur `default`). Pour s'assurer que les réglages soient bien pris en compte, il vaut mieux faire en sorte que le module `ipv6` soit chargé tout de suite (sur Debian, le mettre dans `/etc/modules`) et que les interfaces fixes aient leur propre réglage. Par exemple, sur Debian, on peut mettre dans `/etc/network/interfaces` :

```
iface eth2 inet dhcp
   pre-up sysctl -w net.ipv6.conf.eth2.use_tempaddr=2
```

Ou alors il faut nommer explicitement ses interfaces :

```
net.ipv6.conf.eth0.use_tempaddr = 2
```

Notez aussi qu'`ifconfig` n'affiche pas quelles adresses sont les temporaires (mais `ip addr show` le fait).

Notons que l'adresse « pistable » est toujours présente mais vient s'y ajouter une adresse temporaire choisie au hasard. Selon les règles du RFC 6724, rappelées dans la section 3.1 de notre RFC, c'est cette adresse temporaire qui sera choisie, en théorie, pour les connexions sortantes (voir aussi le RFC 5014 pour une API permettant de contrôler ce choix). Avec Linux, il faut pour cela que `use_tempaddr` vale plus que un (sinon, l'adresse temporaire est bien configurée mais pas utilisée par défaut). ifconfig affichera donc :

```
wlan0     Link encap:Ethernet  HWaddr 00:13:e8:69:59:0d
...
          inet6 addr: 2001:db8:32:aa12:615a:c7ba:73fb:e2b7/64 Scope:Global
          inet6 addr: 2001:db8:32:aa12:213:e8ff:fe69:590d/64 Scope:Global
```

puis, au démarrage suivant, l'adresse temporaire (la première ci-dessus) changera :

```
wlan0     Link encap:Ethernet  HWaddr 00:13:e8:69:59:0d
...
          inet6 addr: 2001:db8:32:aa12:48a9:bf44:5167:463e/64 Scope:Global
          inet6 addr: 2001:db8:32:aa12:213:e8ff:fe69:590d/64 Scope:Global
```

Sur NetBSD, il n'y a qu'une seule variable syscvtl pour toutes les interfaces. Il suffit donc de mettre dans `/etc/sysctl.conf` :

```
net.inet6.ip6.use_tempaddr=1
```

Pour que l'adresse temporaire soit utilisée par défaut, c'est `net.inet6.ip6.prefer_tempaddr`. Sur FreeBSD, je n'ai pas essayé, mais je suppose que les variables sysctl au nom bien parlant `net.inet6.ip6.use_tempaddr` et `net.inet6.ip6.prefer_tempaddr` sont là pour cela.