RFC 7239 : Forwarded HTTP Extension

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 17 juin 2014

Date de publication du RFC : Juin 2014

https://www.bortzmeyer.org/7239.html

Lorsqu'un **client** HTTP se connecte directement à un **serveur** HTTP, ce dernier voit l'adresse IP du client et peut la journaliser pour la traçabilité, faire du contrôle d'accès, des statistiques, etc. Mais si le client est passé par un relais ("*proxy*") ? La solution proposée par notre RFC est que le relais ajoute un en-tête, `Forwarded:`, à la requête, reprenant l'information que le passage par le relais a fait perdre. Il existait déjà des en-têtes non standards pour ce rôle, ce RFC est le premier à normaliser.

Notez que, non seulement les relais sont d'un usage très fréquent dans le Web d'aujourd'hui, mais qu'ils sont souvent inconnus du client, par exemple parce qu'ils s'insèrent automatiquement dans la requête HTTP, sans configuration explicite. Même s'il voulait indiquer son adresse IP, le client ne pourrait pas. Il est donc nécessaire que ce soit le relais qui le fasse. La plupart du temps, la perte d'information due au relais (on n'a plus accès à l'adresse IP du client) n'est pas le but principal du relais, ni même un effet souhaité. Le relais vise par exemple à diminuer l'usage de la liaison Internet en gardant les ressources Web dans son cache. Dans ce cas, la perte d'information est un effet de bord non désiré et on peut souhaiter que le relais répare le problème qu'il a causé.

À noter qu'il existe aussi souvent des relais situés, non pas à proximité du client, mais proche du serveur, par exemple pour lui ajouter une fonction de cache des données souvent accédées (le logiciel le plus connu pour cette tâche est Varnish). Ces relais (parfois appelés « relais inverses ») posent le même problème (de perte d'information) et peuvent utiliser la même solution.

Il existe déjà plusieurs en-têtes pour transporter l'information jusqu'au serveur (comme `X-Forwarded-For:`) mais aucun n'avait encore été normalisé, ce qui fait que l'interopérabilité ne pouvait être garantie. `Forwarded:` est désormais dans le registre des en-têtes <https://www.iana.org/assignments/message-headers/perm-headers.html> (section 9).

Parfois, le relais a au contraire pour but de dissimuler les clients (par exemple pour permettre l'accès anonyme au Web) et, dans ce cas, il ne va évidemment pas utiliser la technique de ce RFC : `Forwarded:` n'est pas obligatoire.

Notez que les mêmes problèmes de perte d'information (parfois souhaitée, parfois non) se produisent avec le NAT : voir le RFC 6269 [1].

L'ancien système non normalisé faisait en général appel à trois en-têtes, `X-Forwarded-For:`, `X-Forwarded-By:` et `X-Forwarded-Proto:`. Un des gros inconvénients de ce système (et qui explique la décision de n'avoir plus qu'un en-tête, avec plusieurs types d'information) était qu'on ne pouvait pas savoir quels en-têtes étaient liés : rien n'indiquait si un `X-Forwarded-For:` et un `X-Forwarded-By:` avaient été mis par le même relais.

La section 4 donne la syntaxe complète de cet en-tête `Forwarded:`. Il est bien sûr facultatif et doit être débrayé par défaut (en raison des problèmes de protection de la vie privée qu'il pose). Il comporte une liste de couples {paramètre, valeur} qui permettent d'indiquer les changements (par exemple d'adresse IP) au passage du relais. Si plusieurs relais ont été successivement utilisés, certains paramètres peuvent être répétés. Premier exemple, où seule l'adresse IP du « vrai » client est indiquée :

```
Forwarded: For="[2001:db8:cafe::17]"
```

Autre exemple, où on met l'adresse IP et le port originels :

```
Forwarded: For="[2001:db8:cafe::17]:4711"
```

Bien plus bavard, un en-tête qui indique l'adresse IP du client mais aussi celle du relais, ainsi que le protocole original (un relais peut transformer du HTTP en HTTPS et réciproquement, par exemple si un relais termine les sessions TLS avant de passer au vrai serveur) :

```
Forwarded: for=192.0.2.60;proto=http;by=203.0.113.43
```

(Je vous laisse chercher pourquoi seules les adresses IPv6 sont entre guillemets. C'est expliqué plus loin.) Et enfin un exemple où deux relais ont été successivement traversés, ils sont alors séparés par des virgules :

```
Forwarded: for=192.0.2.43, for="[2001:db8:cafe::17]"
```

Ces paramètres (dans les exemples ci-dessus, `for`, `by` et `proto`) sont spécifiés en détail en section 5. `by` identifie le relais (utile, si on en a plusieurs, ou simplement si un relais écoute sur plusieurs adresses IP et veut indiquer laquelle a reçu la requête originale). Ce n'est pas forcément une adresse IP. `for` identifie le client original. Là encore, ce n'est pas forcément une adresse IP, cela peut être un identificateur arbitraire.

Si ce n'est pas une adresse IP, qu'est-ce que c'est ? La section 6 décrit le concept d'identificateur, utilisé par les paramètres `for` et `by`. Pour des raisons de sécurité, mais aussi parce qu'ils peuvent être plus pratiques, un relais peut toujours utiliser des identificateurs à lui pour désigner le client (l'identificateur spécial `unknown` indique que le relais ne sait pas). On peut donc avoir :

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

```
Forwarded: for=unknown;by=203.0.113.43
```

où le relais s'identifie mais ne veut pas ou ne peut pas dire qui est son client (il pourrait aussi ne pas mettre `for` du tout mais le `unknown` indique que c'est une décision délibérée, pas un oubli).

Autre possibilité, utiliser des identificateurs attribués localement (précédés par un trait bas), qui ont un sens pour le relais mais pas forcément pour le serveur. Cela permet la traçabilité, sans trop révéler à l'extérieur. Exemple :

```
Forwarded: by=_eth1
```

ou bien :

```
Forwarded: for=_97470ea54d581fc676f6b781b811296e
```

Des nouveaux paramètres pourront être ajoutés dans le futur et seront stockés dans le nouveau registre des paramètres <https://www.iana.org/assignments/http-parameters/http-parameters.xml#forwarded>.

Enfin, les identificateurs peuvent être des adresses IP, comme dans les précédents exemples. Dans ce cas, les adresses IPv6 doivent être entre guillemets, car le deux-points a une signification particulière dans les en-têtes HTTP.

Petit point à garder en tête, l'en-tête `Via:`, normalisé dans la section 5.7.1 du RFC 7230 ne donne, lui, d'informations que sur le relais, pas sur le client originel. En général, on ne peut pas espérer retrouver toute l'information en combinant `Via:` et `Forwarded:` car certains relais mettront uniquement un `Via:`, certains uniquement un `Forwarded:` et d'autres les deux.

La section 8 couvre toutes les questions de sécurité liées à l'en-tête `Forwarded:`. D'abord, il faut se rappeler qu'il n'est absolument pas authentifié. Un client plaisantin a pu en mettre un bidon au départ. Si la requête n'est pas en HTTPS, quelqu'un sur le trajet a pu ajouter, retirer ou modifier cet en-tête n'importe comment. Il faut donc traiter cette information avec prudence, sauf si on est sûr que le précédent relais était une machine de confiance.

Mais les plus gros problèmes de sécurité posés par ce mécanisme concernent évidemment la fuite d'information. L'en-tête `Forwarded:` peut révéler la structure interne d'un réseau privé (`Forwarded: for=192.168.33.165` et on sait quelle plage du RFC 1918 est utilisée en interne...). Plus drôle, si un serveur renvoie cet en-tête dans la **réponse** (ce que le RFC interdit), il révèle au client les relais éventuellement non connus de ce dernier (notez qu'il existe des sites Web pour cela <https://www.bortzmeyer.org/tests-clients-http.html>).

Mais le risque principal de cette fuite d'information est évidemment celui de compromission de la vie privée : l'en-tête `Forwarded:` transmet au serveur des informations que le client peut considérer comme privées, notamment une donnée nominative, l'adresse IP. D'où le rappel qu'un relais qui vise l'anonymisation des requêtes ne doit évidemment pas utiliser cet en-tête. (Le RFC 8165 cite d'ailleurs cet en-tête `Forwarded:` comme un exemple de mauvaise pratique.)

Le RFC demande donc que, par défaut, les identificateurs utilisés dans les paramètres `for` et `by` soient opaques, et générés plus ou moins aléatoirement, **pour chaque requête**. (Mon avis personnel est que, dans ce cas, autant ne pas mettre de `Forwarded:` du tout...)

Le RFC rappelle aussi que le problème de l'anonymat sur le Web est bien plus vaste que cela : si on n'utilise pas de relais anonymisant, on divulgue déjà son adresse IP, et il existe un million d'autres moyens pour un serveur de suivre à la trace un client, comme le montre le Panopticlick <`https://panopticlick.eff.org/`>. Les gens qui se plaignent de la menace que `Forwarded:` (ou ses prédécesseurs comme `X-Forwarded-For:`) font courir à leur vie privée ne sont pas toujours cohérents avec eux-mêmes (par exemple, il est rare qu'ils débrayent les "*cookies*", bien plus efficaces pour le suivi des visiteurs).

Question mises en œuvre, plusieurs logiciels HTTP géraient déjà les en-têtes non officiels (par exemple, Squid documente comment le configurer <`http://www.squid-cache.org/Doc/config/forwarded_for/`>, même chose avec Varnish dans sa FAQ <`https://www.varnish-cache.org/docs/2.1/faq/http.html`>). Pour le `Forwarded:`, il va falloir attendre un peu.