RFC 7739 : Security Implications of Predictable Fragment Identification Values

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 7 février 2016

Date de publication du RFC : Février 2016

https://www.bortzmeyer.org/7739.html

Le protocole IPv6 a un mécanisme de fragmentation des paquets. L'en-tête d'un fragment contient un champ « Identification » (RFC 2460[1], section 4.5) qui, avec les deux adresses Source et Destination, identifie de manière unique un paquet fragmenté (tous les fragments de ce paquet ont le même identificateur), et permet donc le réassemblage à la destination. La façon la plus simple d'affecter ces identificateurs est d'avoir, dans chaque machine, un compteur global incrémenté à chaque paquet qu'on fragmente (section 1 du RFC). Une telle façon de faire mène à des identificateurs **prévisibles** : un observateur qui se connecte à cette machine peut prévoir quel sera l'identificateur du paquet suivant. Cela a de sérieuses conséquences en terme de sécurité. Ce RFC explore ces conséquences, et propose des solutions.

Petit rappel au passage sur la fragmentation IPv6 : contrairement à la fragmentation IPv4, elle ne peut se faire que dans la machine émettrice, pas dans les routeurs intermédiaires. Voici un exemple de paquet fragmenté, vu par tcpdump, la réponse DNS, trop grosse, a été fragmentée en deux :

```
10:45:24.818912 IP6 (hlim 56, next-header Fragment (44) payload length: 432) 2001:4b98:dc2:45:216:3eff:fe4b:8c5b
10:45:24.819008 IP6 (hlim 56, next-header Fragment (44) payload length: 1458) 2001:4b98:dc2:45:216:3eff:fe4b:8c5
```

Le paquet a l'identificateur 0x000079cc, seul le premier fragment (de l'octet 0 à l'octet 423) a pu être analysé (il avait le début de la réponse DNS). Et voici les deux fragments, analysés par tshark :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc2460.txt

```
    Type: IPv6 (0x86dd)
      Internet Protocol Version 6, Src: 2001:4b98:dc2:45:216:3eff:fe4b:8c5b, Dst: 2001:67c:1348:7::86:133
    Next header: Fragment Header for IPv6 (44)
    Fragment Header
        Next header: UDP (17)
        Reserved octet: 0x0000
        0000 0000 0000 0... = Offset: 0 (0 bytes)
        .... .... .... .00. = Reserved bits: 0
        .... .... .... ...1 = More Fragments: Yes
        Identification: 0x000079cc

    Type: IPv6 (0x86dd)
Internet Protocol Version 6, Src: 2001:4b98:dc2:45:216:3eff:fe4b:8c5b, Dst: 2001:67c:1348:7::86:133
    Next header: Fragment Header for IPv6 (44)
    Fragment Header
        Next header: UDP (17)
        Reserved octet: 0x0000
        0000 0001 1010 1... = Offset: 53 (424 bytes)
        .... .... .... .00. = Reserved bits: 0
        .... .... .... ...0 = More Fragments: No
        Identification: 0x000079cc

  [2 IPv6 Fragments (1874 bytes): #2(424), #3(1450)]
        [Frame: 2, payload: 0-423 (424 bytes)]
        [Frame: 3, payload: 424-1873 (1450 bytes)]
        [Fragment count: 2]
        [Reassembled IPv6 length: 1874]
        [Reassembled IPv6 data: 0035921f0752901c6216850000010018000000010000ff00...]
```

Le RFC 2460, section 4.5, indique une seule obligation à la machine fragmenteuse : l'identificateur de paquet doit être unique parmi tous les paquets émis récemment par cette machine, pour éviter toute collision. Ce RFC 2460 donne même le conseil malheureux d'utiliser un compteur global, incrémenté à chaque paquet, pour cela.

Pourquoi, « malheureux » ? Parce que cela rend l'identificateur de paquet prévisible pour un observateur extérieur, faisant ainsi fuir de l'information qui peut être utilisée pour certaines attaques (section 3) :

— L'observateur peut découvrir le débit de la machine émettrice,
— Il peut procéder à des balayages de ports très discrets, en utilisant la machine aux identificateurs prévisibles (détails plus loin),
— Il peut compter le nombre de machines situées derrière un routeur NAT,
— Et surtout, il peut faire des attaques par déni de service (en empêchant le réassemblage, par exemple en envoyant des faux fragments recouvrant en partie des fragments légitimes) ou, encore pire, injecter du contenu dans un paquet (comme dans l'attaque Shulman contre le DNS <http://u.cs.biu.ac.il/~herzbea/security/13-03-frag.pdf>).

Ces attaques n'ont rien de spécifique à IPv6 : des identificateurs de paquet prévisibles ont les mêmes conséquences en IPv4 (voir les articles pionniers de Sanfilippo ici <http://diswww.mit.edu/menelaus.mit.edu/bt/8704> ou là <http://diswww.mit.edu/menelaus.mit.edu/bt/8736>, puis de nombreux autres papiers cités dans la bibliographie de notre RFC, en section 9). Le RFC 6274 décrit en détail ces attaques pour IPv4.

En IPv4, le champ Identification fait partie de chaque paquet (RFC 791, section 3.1) puisque tout routeur pourra fragmenter ce paquet. En IPv6, ce champ n'est présent que dans les paquets fragmentés, dans un « en-tête d'extension » optionnel. Cela rend les attaques un peu plus compliquées en IPv6 : l'attaquant doit d'abord forcer une fragmentation. Une des possibilités pour cela est d'envoyer un paquet ICMP "*Packet Too Big*". Si, en plus, le paquet ICMP contient une MTU inférieure à 1 280 octets, les paquets ne seront pas fragmentés mais porteront quand même l'en-tête d'extension "*Fragment Header*"

(c'est ce qu'on appellait des « fragments atomiques », cf. RFC 6946, fragments atomiques dont l'usage est désormais abandonné, voir le RFC 8021).

En théorie, rien de plus simple que d'envoyer un faux paquet ICMP "*Packet Too Big*". En pratique, il y a quelques limitations pour l'attaquant :

— Les mises en œuvre d'IPv6 mémorisent une PMTU par destination. Il faudra donc envoyer un paquet ICMP contenant l'adresse IP vers laquelle on veut que des fragments soient générés.
— Comme les paquets ICMP ne sont pas signés, les mises en œuvre d'IP savent bien qu'elles doivent s'en méfier, et déploient souvent les précautions du RFC 5927.

Pour éviter ces attaques, que faudrait-il faire ? On souhaite des identificateurs de paquets imprévisibles, mais en même temps ils ne doivent pas être réutilisés, du moins tant qu'on n'est pas sûr que les paquets portant précédemment cet identificateur n'ont pas terminé leur voyage dans le réseau. Des identificateurs complètement aléatoires, par exemple, ne sont pas si idéaux qu'ils le paraissent : il y a un risque de réutilisation accidentelle. Autre contrainte sur les algorithmes de génération de ces identificateurs : ils doivent être très rapides, puisqu'il faut les faire tourner pour chaque paquet fragmenté.

La section 5 de notre RFC décrit les algorithmes possibles. On a vu que l'algorithme « compteur global » était exclu, car il produit des identificateurs prévisibles. Que reste-t-il ?

Le premier algorithme possible (section 5.1) est « un compteur par destination ». On maintient un compteur par adresse IP de destination utilisée, qu'on initialise avec une valeur aléatoire. Simple à mettre en œuvre (il faut juste un générateur de nombres aléatoires), sans réutilisation (pour qu'il y ait collision, il faudrait qu'on émette tellement de paquets vers une destination donnée que le compteur, de 32 bits, déborde), et très rapide (une fois la valeur initiale choisie). Par contre, le prochain identificateur de fragment reste prévisible, mais uniquement pour la machine de destination. Cela peut être utilisé dans certaines attaques. Et il consomme pas mal de mémoire (une entrée par destination), mémoire qu'on ne peut pas libérer, sauf à risquer des collisions d'identificateurs.

Deuxième algorithme envisageable, « identificateur aléatoire » (section 5.2). Idéal pour la sécurité, mais il fait courir des risques de réutilisation des identificateurs, et il est plus coûteux en temps de calcul.

Enfin, un troisième algorithme possible est « condensation de variables » (section 5.3). On prend des variables comme les adresses source et destination, un compteur (pas forcément global), une valeur secrète et on condense le tout. Non, je simplifie trop. En fait, pour être le plus rapide possible tout en étant imprévisible, on a un tableau de compteurs (pas un par destination, mais pas non plus un seul), la condensation des adresses et d'un secret donne l'index dans ce tableau, et on concatène ce compteur avec un autre condensat, fait avec un autre secret. Le résultat est imprévisible de l'extérieur (mais prévisible par le destinataire) et la réutilisation est plus lente qu'avec un compteur global, et cet algorithme peut être mis en œuvre de façon à être très rapide.

L'annexe B de notre RFC est le résultat d'une étude de la prédictabilité par système d'exploitation, en utilisant l'outil `frag6`, qui vient du toolkit SI6 <`http://www.si6networks.com/tools/ipv6toolkit/`>. On peut refaire la même étude chez soi :

```
% sudo frag6  --frag-id-policy -d www.example
Identifying the 'Fragment ID' generation policy of the target node....
Fragment ID policy: Per-destination IDs with increments of 2 (sdev: 1.112134)
```

La destination est un FreeBSD, qui utilise des identificateurs aléatoires depuis sa version 9 :

```
%  sudo frag6   --frag-id-policy -d www.freebsd.org
Identifying the 'Fragment ID' generation policy of the target node....
Fragment ID policy: Randomized IDs (Avg. inc.: 2101165500, sdev: 1531579889.681898)
```

Ici, un Linux récent, avec des compteurs par destination :

```
%  sudo frag6   --frag-id-policy -d 2001:db8:9321:8bb0:9da8:f055:5381:d54c
Identifying the 'Fragment ID' generation policy of the target node....
Fragment ID policy: Per-destination IDs with increments of 1 (sdev: 1.504380)
```

On voit, après cet essai sur trois machines différentes, que les différents algorithmes sont toujours utilisés.