RFC 9715 : IP Fragmentation Avoidance in DNS over UDP

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 28 janvier 2025

Date de publication du RFC : Janvier 2025

https://www.bortzmeyer.org/9715.html

Les requêtes et réponses DNS peuvent voyager sur divers transports, mais le plus fréquent est UDP. Une réponse de grande taille sur UDP peut dépasser la MTU et donc devoir être fragmentée. La fragmentation crée divers problèmes et ce RFC décrit les problèmes, et comment éviter la fragmentation.

Un peu de rappel : avec UDP (RFC 768 [1]), il n'y a pas de négociation de la taille des paquets et un émetteur UDP peut donc, en théorie, envoyer ce qu'il veut, jusqu'à la limite maximum de 65 536 octets. Bon, en pratique, on voit rarement un paquet aussi grand. Déjà, la norme originelle du DNS mettait une limite à 512 octets. Cette limite a disparu depuis longtemps, grâce à EDNS (RFC 6891) et, désormais, le logiciel DNS peut indiquer la taille maximale qu'il est prêt à recevoir. Pendant longtemps, on voyait souvent une taille de 4 096 octets, ce qui excède la MTU de la plupart des liens Internet. Un paquet aussi grand devait donc être fragmenté (que cela soit en IPv4 ou en IPv6). Voici un exemple de réponse plus grande que la MTU typique :

```
% dig oracle.com TXT
...
;; MSG SIZE  rcvd: 3242
```

Si cette réponse était transmise dans un seul datagramme UDP, elle serait fragmentée, ce qui ne marcherait pas toujours, comme l'explique le RFC 8900 et créerait des risques de sécurité <https://www.bortzmeyer.org/dns-attaques-shulman.html>.

TCP (RFC 9293) évite le problème grâce à la négociation de la MSS ("*Maximum Segment Size*", section 3.7.1 du RFC 9293) et au découpage en paquets plus petits que la MTU. La requête DNS ci-dessus ne poserait aucun problèmes avec TCP :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc768.txt

```
% dig +tcp oracle.com TXT
...
;; SERVER: 192.168.2.254#53(192.168.2.254) (TCP)
;; MSG SIZE  rcvd: 3242
```

Mais UDP, plus rigide, n'a pas un tel mécanisme. On pourrait tout faire en TCP (le RFC 7766 rappelle opportunément qu'il n'est pas en option : tout serveur DNS doit l'accepter) mais cela a d'autres inconvénients donc restons avec UDP pour l'instant.

Donc, il est recommandé d'éviter la fragmentation. La section 3 traduit cela en recommandations concrètes : ne pas fragmenter au départ (IPv6) et mettre le bit DF ("*Don't fragment*", pour IPv4, cf. RFC 791) car, sinon, un paquet IPv4 pourra être fragmenté en route (cf. section 7.1). Malheureusement, tous les systèmes d'exploitation ne donnent pas un accès facile à ce bit. Sur Linux, par exemple, il faut passer (ce qui n'est pas très intuitif) par l'option `IP_MTU_DISCOVER`, ce qui mène à d'autres problèmes (cf. annexe C du RFC).

Le logiciel qui répond à une requête DNS doit la maintenir sous une taille qui est le **miminum** de ces quatre tailles :

— la taille indiquée par le demandeur, via EDNS,
— la MTU de l'interface réseau par laquelle partira la réponse,
— la MTU du chemin (si on la connait),
— 1 400 octets (la valeur sûre : aller au-delà serait risqué).

Si on ne peut pas fabriquer une réponse de taille inférieure à ce minimum, il faut mettre le bit TC ("*TrunCated*") dans la réponse DNS.

Ça, c'était pour les répondeurs (les serveurs DNS). Et pour les demandeurs (les clients) ? Ils ne devraient pas indiquer en EDNS une taille supérieure au minimum des trois dernières valeurs citées plus haut. Ici, vue par tshark, la requête faite par dig avec l'option `+dnssec` mais sans autre mention, notamment de taille. dig indique une taille maximale de 1 232 octets, conforme aux recommandations du RFC :

```
Domain Name System (query)
    Flags: 0x0120 Standard query
        0... .... .... .... = Response: Message is a query
...
    Queries
        re: type NS, class IN
            Name: re
            [Name Length: 2]
            [Label Count: 1]
            Type: NS (2) (authoritative Name Server)
            Class: IN (0x0001)
    Additional records
        <Root>: type OPT
            Name: <Root>
            Type: OPT (41)
            UDP payload size: 1232
            Higher bits in extended RCODE: 0x00
            EDNS0 version: 0
            Z: 0x8000
                1... .... .... .... = DO bit: Accepts DNSSEC security RRs
                .000 0000 0000 0000 = Reserved: 0x0000
            Data length: 12
            Option: COOKIE
                Option Code: COOKIE (10)
```

```
Option Length: 8
Option Data: 83d7d3e78b76b45d
Client Cookie: 83d7d3e78b76b45d
Server Cookie: <MISSING>
```

Le RFC demande également aux clients DNS de refuser les réponses fragmentées, compte tenu de certains risques de sécurité <https://www.bortzmeyer.org/dns-attaques-shulman.html> (risque de ne pas pouvoir valider les réponses), et de ne pas hésiter à essayer avec d'autres protocoles de transport, comme TCP.

La section 4, elle, rassemble les recommandations pour les hébergeurs DNS. Comme ils contrôlent les zones publiées, ils peuvent faire en sorte que les réponses ne soient pas trop longues, rendant la fragmentation inutile. Par exemple, ils devraient faire attention à ne pas lister « trop » de serveurs de noms, ne pas mettre « trop » d'enregistrements d'adresses à ces serveurs, etc. Un conseil DNSSEC : certains algorithmes cryptographiques ont des clés et des signatures plus courtes que d'autres. Utilisez-les. (Et, donc, ECDSA ou EdDSA, plutôt que RSA.)

Le RFC note aussi que certains logiciels ne sont pas conformes à la norme. Par exemple, la taille maximale indiquée dans la requête est ignorée et une réponse plus grande que cette taille renvoyée <https://indico.dns-oarc.net/event/31/contributions/692/>. Et, bien sûr, on trouve encore des serveurs qui n'acceptent pas TCP.

Mais quel était le problème de sécurité avec la fragmentation, au juste ? Comme détaillé en section 7.3, qui cite le papier « "*Fragmentation Considered Poisonous*" <https://arxiv.org/abs/1205.4011> », de Amir Herzberg et Haya Shulman, l'empoisonnement de la mémoire d'un résolveur <https://www.bortzmeyer.org/resolveur-dns.html> est plus facile lorsqu'il y a fragmentation, puisque certains des éléments dans le paquet qui peuvent être utilisés pour juger de la légitimité d'une réponse peuvent se retrouver dans un autre fragment. (Voir aussi cet exposé au RIPE <https://ripe67.ripe.net/presentations/240-ipfragattack.pdf>.) Ces attaques permettent ensuite plein d'autres attaques, par exemple contre les autorités de certification <https://dl.acm.org/doi/10.1145/3243734.3243790>.

DNSSEC résout évidemment ces problèmes mais attention, les délégations ne sont pas signées donc DNSSEC ne protège que si les victimes ont signé leur zone et, en prime, DNSSEC permet de détecter la tentative d'empoisonnement et de la refuser, mais pas d'obtenir la bonne réponse.

L'annexe A du RFC détaille les observations quantitatives sur la taille des réponses et les raisons qui peuvent guider le choix des tailles maximum. La section 5 du RFC 8200 garantit que 1 280 octets passeront partout en IPv6, c'est la MTU minimale (mais elle est bien plus basse dans l'ancienne version d'IP). Le RFC 4035 (section 3) impose 1 220 octets minimums pour DNSSEC. Le "*DNS flag day 2020*" <http://www.dnsflagday.net/2020/> proposait 1 232 octets comme taille maximale acceptée (MTU de 1 280 octets, moins 48 octets d'en-têtes). La MTU typique dans l'Internet est de 1 500 octets et, au moins dans le cœur, il n'y a pratiquement pas de liens avec une MTU plus basse. Une taille de 1 452 octets (la MTU moins les en-têtes) est donc réaliste mais, pour tenir compte de certaines techniques qui abaissent la MTU effective, comme les tunnels, notre RFC arrive finalement aux 1 400 octets cités plus haut (section 3).

L'annexe B, elle, décrit le concept de « réponses minimisées ». Certaines mises en œuvre du DNS ont une option qui permet de dire « essaie de réduire la taille de la réponse, en n'incluant pas certaines informations non indispensables ». Attention, des informations comme la colle (les adresses IP des serveurs nommés dans la zone qu'ils servent) sont indispensables (RFC 9471).

Par exemple, pour un serveur faisant autorité <https://www.bortzmeyer.org/serveur-dns-faisant-au
html> qui fait tourner BIND, qui envoie des réponses minimales par défaut, si on lui demande de ne pas être si économe :

```
options {
minimal-responses no;
};
```

Il va alors renvoyer des informations qu'on n'avait pas demandé et dont l'utilité se discute (ici, les deux enregistrements NS) :

```
% dig @::1 -p 8053 under.michu.example
...
;; ANSWER SECTION:
under.michu.example. 600 IN A 192.0.2.56

;; AUTHORITY SECTION:
michu.example. 600 IN NS ns2.nic.fr.
michu.example. 600 IN NS ns1.nic.fr.
...
```

Par défaut, BIND, conformément aux recommandations de notre RFC, n'enverra pas ces deux NS. Ici, il s'agissait d'un serveur faisant autorité mais cette option s'applique aussi aux résolveurs. (Notez que, pour d'autres logiciels, le fait d'envoyer des réponses minimales est par défaut, et pas modifiable.)

En parlant de logiciels, l'annexe C liste un certain nombre de serveurs DNS libres (BIND, Knot, Knot resolver <https://www.knot-resolver.cz/>, Unbound, PowerDNS, PowerDNS recursor <https://www.powerdns.com/powerdns-recursor> et dnsdist <https://dnsdist.org/>) en regardant quelles recommandations de ce RFC sont appliquées.