RFC 9210 : DNS Transport over TCP - Operational Requirements

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 15 avril 2022

Date de publication du RFC : Mars 2022

https://www.bortzmeyer.org/9210.html

Ce nouveau RFC décrit les exigences opérationnelles pour le DNS, plus précisement pour son protocole de transport TCP. Le RFC 7766 [1] décrivait la norme technique, et s'imposait donc aux programmeur·ses, ce RFC 9210 est plus opérationnel et concerne donc ceux et celles qui installent et configurent les serveurs DNS et les réseaux qui y mènent.

Les messages DNS sont historiquement surtout transportés sur UDP. Mais le DNS a toujours permis TCP et, de nos jours, il est fréquemment utilisé notamment en conjonction avec TLS (cf. RFC 7858). Il est d'autant plus important que TCP marche bien qu'il est parfois nécessaire pour certains usages, par exemple pour des enregistrements de grande taille (ce qui arrive souvent avec DNSSEC). Une mise en œuvre du DNS **doit** donc inclure TCP, comme clarifié par le RFC 7766 (les RFC précédents n'étaient pas sans ambiguité). Mais cela ne signifiait pas forcément que les serveurs DNS avaient activé TCP ou que celui-ci marchait bien (par exemple, une stupide "*middlebox*" pouvait avoir bloqué TCP). Ce nouveau RFC 9210 ajoute donc que l'obligation d'avoir TCP s'applique aussi aux serveurs effectifs, pas juste au logiciel.

DNS sur TCP a une histoire compliquée (section 2 du RFC). Franchement, le fait que le DNS marche sur TCP aussi bien que sur UDP n'a pas toujours été clairement formulé, même dans les RFC. Et, en dehors de l'IETF, beaucoup de gens ont raconté n'importe quoi dans des articles et des documentations, par exemple que TCP n'était utile qu'aux transferts de zone, ceux normalisés dans le RFC 5936. Cette légende a été propagée par certains auteurs (comme Cheswick et Bellovin dans leur livre "*Firewalls and Internet Security : Repelling the Wily Hacker*") ou par des gens qui ne mesuraient pas les limites de leurs compétences DNS comme Dan Bernstein <https://cr.yp.to/djbdns/tcp.html#why>.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc7766.txt

Pourtant, TCP a toujours été nécessaire, par exemple pour transporter des données des grande taille (que DNSSEC a rendu bien plus fréquentes). Le RFC 1123, en 1989, insistait déjà sur ce rôle. Certes, EDNS (RFC 6891) permettait déjà des données de taille supérieure aux 512 octets de la norme originelle. Mais il ne dispense pas de TCP. Par exemple, des données de taille supérieure à 1 460 octets (le maximum qui peut tenir avec le MTU typique) seront fragmentées et les fragments, hélas, ne passent pas partout sur l'Internet. (Cf. le « "*DNS Flag Day*" <https://dnsflagday.net/2020/> » de 2020 et lire « "*DNS XL*" <https://labs.apnic.net/?p=1380> » et « "*Dealing with IPv6 fragmentation in the DNS*" <https://blog.apnic.net/2017/08/22/dealing-ipv6-fragmentation-dns/> ».) Et la fragmentation pose également des problèmes de sécurité, voir par exemple « "*Fragmentation Considered Poisonous*" <https://arxiv.org/pdf/1205.4011.pdf> ».

Bon, en pratique, la part de TCP reste faible mais elle augmente. Mais, trop souvent, le serveur ne répond pas en TCP (ou bien ce protocole est bloqué par le réseau), ce qui entraine des tas de problèmes, voir par exemple le récit dans « "*DNS Anomalies and Their Impacts on DNS Cache Servers*" <http://newnog.net/meetings/nanog32/presentations/toyama.pdf> ».

Notons enfin que TCP a toujours permis que plusieurs requêtes se succèdent sur une seule connexion TCP, même si les premières normes n'étaient pas aussi claires qu'il aurait fallu. L'ordre des réponses n'est pas forcément préservé (cf. RFC 5966), ce qui évite aux requêtes rapides d'attendre le résultat des lentes. Et un client peut envoyer plusieurs requêtes sans attendre les réponses (RFC 7766). Par contre, la perte d'un paquet va entrainer un ralentissement de toute la connexion, et donc des autres requêtes (QUIC <https://www.bortzmeyer.org/quic.html> résout ce problème et il y a un projet de DNS sur QUIC).

Vous pouvez tester qu'un serveur DNS accepte de faire passer plusieurs requêtes sur une même connexion TCP avec dig et son option `+keepopen` (qui n'est pas activée par défaut). Ici, on demande à `ns4.bortzmeyer.org` les adresses IP de `www.bortzmeyer.org` et `mail.bortzmeyer.org` :

```
% dig +keepopen +tcp @ns4.bortzmeyer.org www.bortzmeyer.org AAAA mail.bortzmeyer.org AAAA
```

Vous pouvez vérifier avec tcpdump qu'il n'y a bien eu qu'une seule connexion TCP, ce qui ne serait pas le cas sans `+keepopen`.

Les exigences opérationnelles pour le DNS sur TCP figurent dans la section 3 du RFC. Il est désormais obligatoire non seulement d'avoir la possibilité de TCP dans le code mais en outre que cela soit activé partout. (En termes du RFC 2119, on est passé de "*SHOULD*" à "*MUST*".)

La section 4 discute de certaines considérations opérationnelles que cela pourrait poser. D'abord, certains serveurs ne sont pas joignables en TCP. C'était déjà mal avant notre RFC mais c'est encore pire maintenant. Mais cela arrive (ce n'est pas forcément la faute du serveur, cela peut être dû à une "*middlebox*" boguée sur le trajet). Les clients doivent donc être préparés à ce que TCP échoue (de toute façon, sur l'Internet, tout peut toujours échouer). D'autre part, diverses attaques par déni de service sont possibles, aussi bien contre le serveur ("*SYN flooding*", contre lequel la meilleure protection est l'utilisation de "*cookies*", cf. RFC 4987), que contre des machines tierces, le serveur étant utilisé comme réflecteur.

Aussi bien le client DNS que le serveur n'ont pas des ressources illimitées et doivent donc gérer les connexions TCP. Dit plus brutalement, il faut être prêt à couper les connexions qui semblent inutilisées. Bien sûr, il faut aussi laisser les connexions ouvertes suffisamment longtemps pour amortir leur création

sur le plus grand nombre de requêtes possibles, mais il y a des limites à tout. (Le RFC suggère une durée maximale dans les dizaines de secondes, pouvant être abaissée à quelques secondes pour les serveurs très chargés.)

TCP est particulièrement intéressant pour le DNS quand TLS est inséré (RFC 7858). Mais il va alors consommer davantage de temps de CPU et, dans certains cas, l'établissement de connexion sera plus lent.

Le RFC donne quelques conseils quantitatifs (à ne pas appliquer aveuglément, bien sûr). Un maximum de 150 connexions TCP simultanées semble raisonnable pour un serveur ordinaire, avec un maximum de 25 par adresse IP source. Un délai de garde après inactivité de 10 secondes est suggéré. Par contre, le RFC ne propose pas de valeur maximale au nombre de requêtes par connexion TCP, ni de durée maximale à une connexion.

Les fanas des questions de sécurité noteront que les systèmes de journalisation et de surveillance peuvent ne pas être adaptés à TCP. Par exemple, un méchant pourrait mettre la requête DNS dans plusieurs segments TCP et donc plusieurs paquets IP. Un système de surveillance qui considérerait qu'une requête = un paquet serait alors aveugle. Notez qu'un logiciel comme dnscap <https://www.dns-oarc.net/tools/dnscap> a pensé à cela, et réassemble les paquets. Mais il est sans doute préférable, de nos jours, de se brancher sur le serveur, par exemple avec dnstap <https://dnstap.info>, notamment pour éviter de faire le réassemblage deux fois. (Ceux qui lisent mes articles depuis longtemps savent que je donnais autrefois le conseil inverse. Mais le déploiement de plus en plus important de TCP et surtout de TLS impose de changer de tactique.)