RFC 5966 : DNS Transport over TCP - Implementation Requirements

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 24 août 2010

Date de publication du RFC : Août 2010

https://www.bortzmeyer.org/5966.html

La vague mention des "*virtual circuits*" (connexions TCP) dans la section 4.2 du RFC 1035 [1] a suscité d'innombrables polémiques, que ce RFC 5966 a tranché. En deux mots, un serveur DNS doit-il fournir son service avec les protocoles de transport TCP et UDP ou bien peut-il se contenter d'UDP ? La réponse était clairement « TCP **et** UDP » et le remplaçant de ce document, le RFC 7766, a été depuis encore plus net.

La discussion était d'autant plus vive que certains registres procèdent à des tests techniques obligatoires avant l'enregistrement d'un nom de domaine et que ces tests peuvent inclure la disponibilité du service sur TCP. Par exemple, l'outil Zonecheck <https://www.bortzmeyer.org/zonecheck-3-0.html> dispose d'un tel test (qui se configure avec <check name="tcp" severity="f ou bien w" category="connectivity:l4"/>) et l'AFNIC impose le succès de ce test pour un enregistrement dans .fr (contrairement à ce qu'on lit parfois, il est faux de dire « Zonecheck impose TCP », Zonecheck est configurable et c'est l'AFNIC qui choisit d'activer ce test, et décide de son caractère bloquant ou non). On a vu ainsi des discussions sur ces tests, même si les opposants ont rarement fait l'effort de prendre le clavier pour écrire une argumentation raisonnée (on les comprend quand on voit que toutes les discussions publiques sur le sujet indiquent un consensus des experts sur l'importance du service TCP, voir par exemple <http://www.circleid.com/posts/afnic_dns_server_redelegation/>).

Mais c'est aussi que l'approche « légaliste » de la discussion est vouée à tourner en rond. Le texte du RFC 1035 (ou celui de la section 6.1.3.2 du RFC 1123) est vague, et peut s'interpréter de différentes

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc1035.txt

manières. Il faut donc revenir aux bases du DNS, pour décider si TCP est important ou pas, et pas essayer vainement de trouver un texte sacré qui trancherait la question de manière claire.

Revenons donc à ces bases (section 1 du RFC). Le DNS peut s'utiliser sur UDP et TCP. Par exemple, l'outil dig, par défaut, utilise UDP, mais l'option `+tcp` (ou `+vc` pour "*virtual circuit*") lui fait utiliser TCP. TCP est obligatoire pour les transferts de zone (cf. RFC 5936) mais, contrairement à une légende répandue, n'est pas réservé à ces transferts et peut être utilisé pour des requêtes ordinaires. Il est notamment obligatoire si la réponse arrive tronquée (bit TC mis à un) car elle ne tenait pas dans le paquet UDP (dont la taille était autrefois limitée à 512 octets). Depuis la création du DNS, la taille des réponses a beaucoup augmenté (IDN et IPv6 mais surtout DNSSEC y ont largement contribué) et, malgré la suppression de la limite de 512 (cf. RFC 6891), TCP est donc encore plus nécessaire que dans le passé.

Arrivé là, il faut faire une distinction importante entre ce que peut le logiciel et ce qu'a activé l'administrateur système. Ainsi, le logiciel djbdns permet parfaitement TCP mais il n'est pas activé par défaut <http://cr.yp.to/djbdns/tcp.html>. De même, BIND ou NSD ont TCP par défaut mais un pare-feu situé devant le serveur de noms peut bloquer les accès TCP. Notre RFC 5966 sépare donc **protocole** et **mise en œuvre** et ne traite que le cas du logiciel : il précise que tout logiciel DNS **doit** avoir la possibilité de faire du TCP mais il ne tranche pas (délibérement) la question de savoir si TCP doit être disponible sur un serveur de noms en activité. Il note simplement que l'absence de TCP peut planter le processus de résolution de noms.

La section 3 du RFC discute ensuite les différentes questions liées à ce choix. Le principal problème est celui de la taille des réponses <https://www.bortzmeyer.org/dns-size.html>. Autrefois limitée à 512 octets, elle peut prendre des valeurs plus grandes (jusqu'à 65536 octets) avec EDNS0. Mais la MTU de 1500 octets est hélas une limite pratique fréquente (cf. RFC 5625, du même auteur), en raison de pare-feux mal configurés. Cela peut poser des problèmes, par exemple lors du déploiement de DNSSEC <https://www.bortzmeyer.org/risques-reels-dns-limite.html>. Un simple `NXDOMAIN` depuis `.org` dépasse les 512 octets :

```
% dig +dnssec SOA certainlydoesnotexist.org

; <<>> DiG 9.6-ESV-R1 <<>> +dnssec SOA certainlydoesnotexist.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 64046
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;certainlydoesnotexist.org. IN SOA

;; AUTHORITY SECTION:
org. 0 IN SOA a0.org.afilias-nst.info. noc.afilias-nst.info. 2009281221 1800 900 604800 86400
org. 0 IN RRSIG SOA 7 1 900 20100907065203 20100824055203 52197 org. m3DPnEo+Ibd8W0d/cVW7sMZb8UooI6F6mOn/mQS
h9p7u7tr2u91d0v0ljs9l1gidnp90u3h.org. 86400 IN NSEC3 1 1 1 D399EAAB H9RSFB7FPF2L8HG35CMPC765TDK23RP6 NS SOA
h9p7u7tr2u91d0v0ljs9l1gidnp90u3h.org. 86400 IN RRSIG NSEC3 7 2 86400 20100907065203 20100824055203 52197 org
b42oorh0vfd9ble13e1him76im76qsl6.org. 86400 IN NSEC3 1 1 1 D399EAAB B49TR2SSRPRC2FF6FIVQ25UFDMRL7Q63 NS DS R
b42oorh0vfd9ble13e1him76im76qsl6.org. 86400 IN RRSIG NSEC3 7 2 86400 20100906200816 20100823190816 52197 org
vae6tvink0oqnd756037uoir56fokhtd.org. 86400 IN NSEC3 1 1 1 D399EAAB VB1V404HEINQ7A8TQTJ9OASALN2IS19G A RRSIG
vae6tvink0oqnd756037uoir56fokhtd.org. 86400 IN RRSIG NSEC3 7 2 86400 20100907011155 20100824001155 52197 org

;; Query time: 43 msec
;; SERVER: ::1#53(::1)
;; WHEN: Tue Aug 24 08:54:12 2010
;; MSG SIZE  rcvd: 1019
```

Dans tous ces cas, TCP est la seule solution fiable. À l'ère de YouTube et de ses giga-octets de vidéo, il serait curieux d'en rester au DNS de 1990 et de ses limites archaïques à 512 octets. Un serveur de noms doit donc pouvoir utiliser TCP.

Mais, lorsque le serveur de noms a le choix, quel protocole de transport doit-il utiliser ? C'est l'objet de la section 4. Elle modifie légèrement le RFC 1123 dont la section 6.1.3.2 imposait à un résolveur de tenter UDP d'abord (et de passer en TCP si la réponse arrive tronquée). Désormais, le résolveur a le droit de poser sa question en TCP d'abord, s'il a de bonnes raisons de penser (par exemple parce qu'il a déjà parlé à ce serveur) que la réponse arrivera tronquée.

Les sections 5 et 6 sont consacrées à des problèmes pratiques avec la mise en œuvre de TCP. Par exemple, comme le DNS sur TCP peut faire passer plusieurs requêtes dans une connexion TCP, notre RFC précise que les réponses ont parfaitement le droit d'arriver dans un ordre différent de celui des questions.

Restent les questions de sécurité et autres craintes qui sont mentionnées par certains objecteurs (on peut citer <http://cr.yp.to/djbdns/tcp.html#why> comme très bel exemple de catalogue d'erreurs et d'énormités). En effet, programmer TCP dans le serveur de noms n'est pas très difficile. Ma propre implémentation, dans Grong <https://www.bortzmeyer.org/dnsserver-en-go.html>, fut triviale, car le langage Go, avec son parallélisme natif facilite beaucoup les choses. Mais, même en C, si le serveur utilise plusieurs "*sockets*", par exemple pour gérer IPv4 et IPv6, ajouter TCP en prime ne changera pas beaucoup la boucle principale autour de `select()`. L'obligation de gérer TCP ne gênera donc qu'une petite minorité de programmeurs, ceux qui essayaient de faire un serveur DNS basé sur le traitement séquentiel des paquets.

En revanche, l'obligation de gérer TCP est parfois critiquée pour des raisons de sécurité. La section 7 discute ce problème des DoS : TCP nécessite un **état** sur le serveur et consomme donc des ressources. En théorie, cela rendrait les serveurs DNS plus sensibles aux DoS. Toutefois, presque tous les serveurs de noms de la racine ont TCP depuis longtemps, ainsi que la grande majorité des serveurs des grands TLD et on ne voit pas d'attaques pour autant. (Le RFC se limite au cas du DNS mais on peut aussi, en sortant du petit monde DNS, noter que l'écrasante majorité des serveurs Internet utilise exclusivement TCP... En outre, UDP a ses propres problèmes de sécurité, notamment la facilité à tricher sur l'adresse IP source, facilité qui est à la base de l'attaque Kaminsky <https://www.bortzmeyer.org/comment-fonctionne-la-faille-kaminsky.html>.) Le RFC recommande toutefois la lecture de bons textes comme « "*CPNI technical note 3/2009 \ Security assessment of the Transmission Control Protocol (TCP)* " <http://www.cpni.gov.uk/Docs/tn-03-09-security-assessment-TCP.pdf> ».

Et la charge du serveur ? Le RFC n'en parle pas mais il y avait eu des inquiétudes à ce sujet, basées sur le fait que les études <https://www.dns-oarc.net/node/199> montrent une augmentation relative très importante du trafic TCP lorsqu'on active DNSSEC. Ce trafic peut-il épuiser le serveur. Notons que, si un passage de 0,2 requête/s à 50 peut sembler énorme, cela reste ridicule en valeur absolue, à l'heure où le plus petit serveur HTTP en gère bien davantage.

Par contre, une autre objection contre TCP n'est pas citée, ses possibles problèmes avec l'"*anycast*". Désolé, mais je manque de temps pour la commenter ici.

Ah, me demanderez-vous, mon opinion personnelle ? Je trouve qu'aujourd'hui, TCP est à la fois indispensable pour ne pas limiter à des valeurs ridiculement basses la taille des réponses, et facile à déployer, comme le montre l'expérience de tous les gros TLD. EDNS0 permettrait de résoudre une bonne partie des problèmes de taille (et je veux donc bien entendre les objecteurs qui diraient « le test technique devrait exiger TCP **ou** EDNS0 ») mais je note que les serveurs qui n'ont pas TCP n'ont pratiquement jamais EDNS0 non plus... Il n'y a donc guère de raisons valables, en 2010, d'avoir des serveurs de noms inaccessibles en TCP. (En 2016, notre RFC a été remplacé par le RFC 7766, qui augmente encore le rôle de TCP.)