RFC 9103 : DNS Zone Transfer over TLS

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 septembre 2021

Date de publication du RFC : Août 2021

https://www.bortzmeyer.org/9103.html

Traditionnellement, le transfert d'une zone DNS depuis le serveur maitre vers ses esclaves se fait en clair. Si l'authentification et l'intégrité sont protégées, par exemple par TSIG, le transfert en clair ne fournit pas de confidentialité. Or on n'a pas forcément envie qu'un tiers puisse lire l'entièreté de la zone. Ce RFC normalise un transfert de zones sur TLS, XoT ("*zone transfer over TLS*"). Il en profite pour faire quelques ajustement dans l'utilisation de TCP pour les transferts de zone.

Le RFC fait quarante-deux pages mais le principe est très simple. Au lieu de faire le transfert de zones (aussi nommé AXFR, RFC 5936 [1]) directement sur TCP, on le fait sur du TLS, comme le fait DoT (RFC 7858). Le reste du RFC est consacré à des détails pratiques, mais le concept tient en une seule phrase.

Le DNS sur TLS est normalisé depuis cinq ans, mais il ne couvre officiellement que la communication entre la machine terminale et le résolveur <https://www.bortzmeyer.org/resolveur-dns.html>. Bien que des essais aient été faits pour communiquer en TLS avec des serveurs faisant autorité <https://www.bortzmeyer.org/serveur-dns-faisant-autorite.html>, rien n'est encore normalisé. Mais, au fait, est-ce que les transferts de zone ont vraiment besoin de confidentialité ? Le RFC 9076 ne traite pas le problème, le renvoyant aux RFC 5936 (la norme du transfert de zones, qui parle de la possibilité de restreindre ce transfert à des machines autorisées) et au RFC 5155 (qui parle des risques d'énumération de tous les noms d'une zone). Certes, le DNS étant public, tout le monde peut interroger les serveurs faisant autorité, et connaitre ainsi les données associées à un nom de domaine, mais il faut pour cela connaitre les noms. Si les petites zones n'ont rien de secret (elles ne contiennent que l'apex et www), si certaines zones sont publiques (la racine, par exemple), pour les autres zones, leur responsable ne souhaite pas forcément qu'on ait accès à tous les noms. Certains peuvent être le nom d'une personne (le-pc-de-henri.zone.example), et donc être des données personnelles, à protéger (le RFC note

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5936.txt

bien sûr que c'est une mauvaise pratique, mais elle existe), certains noms peuvent renseigner sur les activités et les projets d'une entreprise. Bref, il peut être tout à fait raisonnable de ne pas vouloir transmettre la zone en clair.

À noter qu'une surveillance du réseau pendant un transfert de zones fait en clair n'est pas la seule façon de trouver tous les noms. Une autre technique est l'énumération, utilisant par exemple les enregistrement NSEC de DNSSEC. Les enregistrement NSEC3 du RFC 5155 ont été conçus pour limiter ce risque (mais pas le supprimer <https://www.cs.bu.edu/~goldbe/papers/nsec3attacks.pdf>; le projet NSEC5 va essayer de résoudre le problème mais c'est loin d'être garanti).

Comment sécurise-t-on les transferts de zone aujourd'hui ? Les TSIG du RFC 8945 protègent l'intégrité mais pas la confidentialité. Vous pouvez trouver des détails pratiques sur la sécurisation des transferts de zone, combinant TSIG et ACL, dans un document du NIST en anglais <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-81-2.pdf> et un document de l'ANSSI en français <https://www.ssi.gouv.fr/administration/guide/bonnes-pratiques-pour-lacquisiti>. On peut aussi en théorie utiliser IPsec entre maitre et esclaves mais cela semble irréaliste. D'où l'idée de base de ce nouveau RFC : mettre le transfert de zones sur TLS. Après tout, on utilise déjà TLS pour sécuriser la communication entre le client final et le résolveur <https://www.bortzmeyer.org/resolveur-dns.html> : le protocole DoT, normalisé dans le RFC 7858.

Notre RFC introduit quelques nouveaux acronymes rigolos :

— XFR : couvre à la fois AXFR (RFC 5936) et IXFR (transferts incrémentaux, RFC 1995). Donc, "*XFR-over-TLS*", c'est un transfert de zones, complet ou incrémental, sur TLS.
— XoT : acronyme pour "*XFR-over-TLS*".
— AXoT : AXFR sur TLS.
— IXoT : IXFR sur TLS.

Pour la terminologie liée à la vie privée, voir le RFC 6973.

La section 3 du RFC spécifie le modèle de menace : on veut protéger le transfert de zones contre l'espionnage par un tiers, mais on n'essaie pas de cacher l'existence de la zone, ou le fait que telle machine soit serveur faisant autorité <https://www.bortzmeyer.org/serveur-dns-faisant-autorite.html> pour la zone. En effet, ces informations peuvent être obtenues facilement en interrogeant le DNS ou, dans le cas d'un maitre caché, en regardant le trafic réseau, sans même en décrypter le contenu.

Les principes de conception de XoT ? Fournir de la confidentialité et de l'authentification grâce à TLS, mais aussi en profiter pour améliorer les performances. Un certain nombre de programmes DNS ne peuvent pas, par exemple, utiliser la même connexion TCP pour plusieurs transferts de zone (même si le RFC 5936 le recommande). Il est difficile de résoudre ce problème sans casser les programmes existants et pas encore mis à jour. Par contre, XoT étant nouveau, il peut spécifier dès le début certains comportements qui améliorent les performances, comme l'utilisation d'une seule connexion pour plusieurs transferts.

Petit rappel de comment fonctionne le transfert de zones (RFC 5936, si vous voulez tous les détails). D'abord, le fonctionnement le plus courant : le serveur maitre (ou primaire) envoie un NOTIFY (RFC 1996) à ses esclaves (ou secondaires). Ceux-ci envoient une requête de type SOA au maitre puis, si le numéro de série récupéré est supérieur au leur, ils lancent l'AXFR sur TCP. Il y a des variations possibles : si le maitre n'envoie pas de NOTIFY ou bien s'ils sont perdus, les esclaves testent quand même de temps en temps le SOA (paramètre "*Refresh*" du SOA). Autre variante possible : les esclaves peuvent utiliser IXFR (RFC 1995) au lieu de AXFR pour ne transférer que la partie de la zone qui a changé. Et ils peuvent avoir à se rabattre en AXFR si le maitre ne gérait pas IXFR. Et les messages IXFR peuvent être sur UDP ou sur TCP (en pratique, BIND, NSD et bien d'autres font toujours l'IXFR sur TCP). Bref, il y a

plusieurs cas. Les messages NOTIFY et SOA ne sont pas considérés comme sensibles du point de vue de la confidentialité et XoT ne les protège pas.

Maintenant, avec XoT, comment ça se passera (section 6 du RFC) ? Une fois le transfert de zones décidé (via SOA, souvent précédé du NOTIFY), le client (qui est donc le serveur secondaire) doit établir une connexion TLS (1.3 minimum), en utilisant ALPN, avec le nom `dot` (on notera que c'est donc le même ALPN pour XoT et pour DoT, l'annexe A du RFC explique ce choix). Le port est par défaut le classique 853 de DoT. Le client authentifie le serveur via les techniques du RFC 8310. Le serveur vérifie le client, soit avec des mécanismes TLS (plus sûrs mais plus compliqués à mettre en œuvre), soit avec les classiques ACL. Le client TLS (le serveur DNS esclave) fait ensuite de l'IXFR ou du AXFR classique sur ce canal TLS. Le serveur doit être capable d'utiliser les codes d'erreur étendus du RFC 8914.

Voilà, c'est tout. Maintenant, quelques détails pratiques. D'abord, rappelez-vous qu'il n'existe pas à l'heure actuelle de norme pour du DoT entre résolveur et serveur faisant autorité. Mais il y a des expériences en cours. Ici, par exemple, je fais du DoT avec un serveur faisant autorité pour `facebook.com` :

```
% kdig +tls @a.ns.facebook.com.  SOA facebook.com
;; TLS session (TLS1.3)-(ECDHE-X25519)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)
...
;; ANSWER SECTION:
facebook.com.        3600 IN SOA a.ns.facebook.com. dns.facebook.com. 1631273358 14400 1800 604800 300

;; Received 86 B
;; Time 2021-09-10 13:35:44 CEST
;; From 2a03:2880:f0fc:c:face:b00c:0:35@853(TCP) in 199.0 ms
```

Mais on pourrait imaginer des serveurs faisant autorité qui ne fournissent pas ce service, pas encore normalisé, mais qui veulent faire du XoT. Est-ce possible de réserver TLS aux transferts de zone, mais sans l'utiliser pour les requêtes « normales » ? Oui, et dans ce cas le serveur doit répondre REFUSED à ces requêtes normales, avec le code d'erreur étendu 21 ("*Not supported*" <`https://www.iana.org/assignments/dns-parameters/dns-parameters.xml#extended-dns-error-codes`>). Ceci dit, il est difficile de dire comment réagiront les clients ; ils risquent d'ignorer le code d'erreur étendu et de recommencer bêtement (ou au contraire d'arrêter complètement d'utiliser le serveur).

Le but de XoT est de protéger le contenu de la zone contre les regards indiscrets. Mais comme vous le savez, TLS ne dissimule pas la taille des données qui passent entre ses mains et l'observation de cette taille peut donner des informations à un observateur, par exemple la taille des mises à jour en IXFR indique l'activité d'une zone. Le RFC suggère donc de remplir les données. Cela peut nécessiter d'envoyer des réponses vides, ce qui était interdit par les précédents RFC mais est maintenant autorisé.

Et les différentes formes de parallélisme des requêtes que permet le DNS sur TCP (et donc DoT) ? Elles sont décrites dans le RFC 7766. Si elles sont nécessaires pour exploiter le DNS de manière satisfaisante, elles ne sont pas forcément utiles pour XoT et notre RFC explique qu'on n'est pas forcé, par exemple, de permettre le "*pipelining*" (envoi d'une nouvelle requête quand la réponse à la précédente n'est pas encore arrivée) quand on fait du XoT. Par contre, comme avec tous les cas où on utilise du DNS sur TCP, il est recommandé de garder les connexions ouvertes un certain temps, par exemple en suivant le RFC 7828.

J'ai parlé plus haut de l'authentification dans une session XoT. La section 9 du RFC revient en détail sur cette question. Quels sont les mécanismes existants pour authentifier un transfert de zones ? Ils diffèrent par les services qu'ils rendent :

— Authentification de la source des données DNS.

— Authentification du correspondant (qui n'est pas forcément la source des données).
— Confidentialité.

Les mécanismes possibles sont :

— TSIG (RFC 8945) ; utiisant des clés partagées (ce qui n'est pas pratique), il permet d'authentifier la source des données.
— SIG(0) (RFC 2931) ; utilisant de la cryptographie asymétrique, il permet d'authentifier la source des données. Ceci dit, il n'est quasiment jamais mis en œuvre dans les programmes DNS.
— TLS en mode « opportuniste » (sans authentification) ; il ne garantit rien, sauf peut-être contre un observateur passif.
— TLS en mode strict (obligation d'authentification du serveur par le client, RFC 8310) ; cela permet l'authentification du correspondant, et la confidentialité (mais ne garantit pas que le correspondant est la vraie source des données).
— TLS mutuel ; le client s'authentifie également.
— ACL fondées sur l'adresse IP ; très répandu, ce mécanisme permet au serveur d'« authentifier » le client. Une faiblesse pour le cas de XoT : le serveur doit accepter l'établissement de la session TLS avant de savoir s'il s'agit de XoT ou de DNS ordinaire sur TLS.
— ZONEMD (RFC 8976) ; ce n'est pas vraiment une technique d'authentification mais elle peut être utile, en combinaison avec XoT.

Quelles sont les mises en œuvres actuelles de XoT ? Une liste existe <https://dnsprivacy.org/implementation_status/#xfrxot-implementation-status>. Actuellement, il n'y a guère que NSD qui permet le XoT mais le travail est en cours pour d'autres. Voyons comment faire du XoT avec NSD. On télécharge la dernière version (XoT est apparu en 4.3.7), on la compile (pas besoin d'options particulières) et on la configure, par exemple ainsi :

```
server:
       port: 7053
       ...
       tls-service-key: "server.key"
       tls-service-pem: "server.pem"
       # No TLS service if the port is not explicit in an interface definition
       ip-address: 127.0.0.1@7153
       tls-port: 7153

zone:
        name: "fx"
        zonefile: "fx"
# NSD 4.3.7 (september 2021) : not yet a way to restrict XoT
        provide-xfr: 127.0.0.1 NOKEY
```

Ainsi configuré, NSD va écouter sur le port 7153 en TLS. Notons qu'il n'y a pas de moyen de configurer finement : il accepte forcément XoT et les requêtes « normales ». De même, on ne peut pas obliger un client DNS à n'utiliser que XoT. Testons avec kdig, un client DNS qui accepte TLS, d'abord une requête normale :

```
% kdig +tls @localhost -p 7153 SOA fx
;; TLS session (TLS1.3)-(ECDHE-SECP256R1)-(RSA-PSS-RSAE-SHA256)-(AES-256-GCM)
;; ->>HEADER<<- opcode: QUERY; status: NOERROR; id: 11413
...
;; ANSWER SECTION:
fx.                  600 IN SOA ns1.fx. root.fx. 2021091201 604800 86400 2419200 86400
...
;; Time 2021-09-12 11:06:39 CEST
;; From 127.0.0.1@7153(TCP) in 0.2 ms
```

Parfait, tout marche. Et le transfert de zones ?

```
% kdig +tls @localhost -p 7153 AXFR fx
;; AXFR for fx.
fx.                  600 IN SOA ns1.fx. root.fx. 2021091201 604800 86400 2419200 86400
...
;; Received 262 B (1 messages, 9 records)
;; Time 2021-09-12 11:07:25 CEST
;; From 127.0.0.1@7153(TCP) in 2.3 ms
```

Excellent, nous faisons du XoT. Wireshark nous le confirme :

```
% tshark -d tcp.port==7153,tls -i lo port 7153
...
4 0.001200185    127.0.0.1 → 127.0.0.1    TLSv1 439 Client Hello
...
6 0.006305031    127.0.0.1 → 127.0.0.1    TLSv1.3 1372 Server Hello, Change Cipher Spec, Application Data, Appli
...
10 0.007385823    127.0.0.1 → 127.0.0.1    TLSv1.3 140 Application Data
```

Dans ce cas, on utilisait NSD comme serveur primaire, recevant les clients XoT. Il peut aussi fonctionner comme serveur secondaire et il est alors client XoT. Je n'ai pas testé mais John Shaft l'a fait et en gros la configuration ressemble à :

```
tls-auth:
   name: un-nom-pour-identifier-ce-bloc
   auth-domain-name: nom-du-serveur-principal
zone:
   ...
   request-xfr: AXFR 2001:db8::1 NOKEY un-nom-pour-identifier-ce-bloc
```