Quad9, résolveur DNS public, et sécurisé par TLS

Posté par **Stephane Bortzmeyer** (**page perso**) le 17/11/17 à 23:49. Édité par **4 contributeurs**. Modéré par **tankey**.
Tags : **quad9**, **dns**, **résolveur**, **tls**, **dnssec**

Le résolveur DNS Quad9 (prononcer « quoi de neuf » en français) a été annoncé aujourd'hui. C'est un résolveur DNS public, mais dont l'originalité est d'être accessible de manière sécurisée, avec TLS (DNS sur TLS est décrit dans le RFC 7858).

Alors, le lectorat de *LinuxFr.org* étant super au courant, va dire « mais des résolveurs DNS publics, il y en a plein ! Pourquoi un de plus ? ». Le plus connu est Google Public DNS, mais il en existe beaucoup d'autres, avec des politiques et des caractéristiques techniques diverses. Notamment, tous (à l'exception de Cisco OpenDNS) sont non sécurisés : le lien entre vous et le résolveur est en clair, tout le monde peut écouter, et il n'est pas authentifié, donc vous croyez parler à Google Public DNS mais, en fait, vous parlez au tricheur que votre FAI a annoncé dans ses réseaux locaux.

- Journal à l'origine de la dépêche (63 clics)
- Le site de référence Quad9 (69 clics)
- Politique de vie privée de Quad9 (25 clics)
- La FAQ de Quad9 (29 clics)
- Subby (19 clics)
- Le projet DNS privacy (22 clics)

Et Quad9, c'est mieux, alors ? D'abord, c'est géré par l'organisme sans but lucratif bien connu PCH, qui gère une bonne partie de l'infrastructure du DNS (et qui sont des copains, oui, je suis subjectif).

Quad9, lui, sécurise par TLS (RFC 7858). Cela permet d'éviter l'écoute par un tiers, et cela permet d'authentifier le résolveur (mais, attention, je n'ai pas encore testé ce point, Quad9 ne semble pas distribuer de manière authentifiée ses clés publiques).

Question politique, des points à noter :

- Quad9 s'engage à ne pas stocker votre adresse IP ;
- leur résolveur est un résolveur menteur : il ne répond pas (délibérément) pour les noms de domaines qu'il estime liés à des activités néfastes comme la distribution de logiciels malveillants.

L'adresse IPv4 de Quad9, comme son nom l'indique, est 9.9.9.9. Son adresse IPv6 est 2620:fe::fe. D'abord, un accès classique en UDP en clair, sur votre distribution GNU/Linux favorite :

```
% dig +nodnssec @9.9.9.9 AAAA irtf.org

; <<>> DiG 9.10.3-P4-Ubuntu <<>> +nodnssec @9.9.9.9 AAAA irtf.org
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11544
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;irtf.org.      IN AAAA

;; ANSWER SECTION:
irtf.org.       1325 IN AAAA 2001:1900:3001:11::2c

;; Query time: 4 msec
;; SERVER: 9.9.9.9#53(9.9.9.9)
;; WHEN: Thu Nov 16 09:49:41 +08 2017
;; MSG SIZE  rcvd: 65
```

On y voit que Quad9 valide avec DNSSEC (la réponse a bien le bit AD — *Authentic Data*).

Maintenant, testons la nouveauté importante de ce service : DNS sur TLS. C'est du TLS, donc on peut y aller avec `openssl` :

```
% openssl s_client -connect \[2620:fe::fe\]:853 -showcerts
```

On voit que Quad9 répond bien en TLS et a un certificat *Let's Encrypt*.

Testons ensuite avec un client DNS, le programme `getdns_query` distribué avec getdns (l'option `-l L` lui dit d'utiliser DNS sur TLS) :

```
% getdns_query @9.9.9.9 -s -l L www.afnic.fr AAAA
{
  "answer_type": GETDNS_NAMETYPE_DNS,
  "canonical_name": <bindata for lb01-1.nic.fr.>,
  "just_address_answers":
  [
    {
      "address_data": <bindata for 2001:67c:2218:30::24>,
      "address_type": <bindata of "IPv6">
    }
 ...
```

On peut utiliser `tshark` pour vérifier qu'on est bien en TLS :

```
% tshark -n -i eth0  -d tcp.port==853,ssl host 9.9.9.9
```

Le `-d tcp.port==853,ssl` était là pour dire à tshark d'interpréter ce qui passe sur le port 853 (celui de DNS-sur-TLS) comme étant du TLS. On voit bien le dialogue TLS, mais évidemment pas les questions et réponses DNS puisque tout est chiffré.

Bien, maintenant que les tests se passent bien, comment utiliser Quad9 pour la vraie résolution de noms ? On va utiliser Stubby pour parler à Quad9. Le fichier de configuration Stubby sera du genre :

```
listen_addresses:
  - 0::1@8053

dns_transport_list:
  - GETDNS_TRANSPORT_TLS

upstream_recursive_servers:
# Quad9
   - address_data: 9.9.9.9
     tls_auth_name: "dns.quad9.net"
   - address_data: 2620:fe::fe
     tls_auth_name: "dns.quad9.net"
```

On indique à stubby d'écouter sur l'adresse locale `::1`, port 8053, et de faire suivre les requêtes en DNS sur TLS à `9.9.9.9` ou `2620:fe::fe`. On lance stubby :

```
% stubby
```

Et on peut le tester, en utilisant `dig` pour interroger à l'adresse et au port indiqué :

```
% dig @::1 -p 8053 A www.catstuff.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @::1 -p 8053 A www.catstuff.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20910
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 65535
;; QUESTION SECTION:
;www.catstuff.com.  IN A

;; ANSWER SECTION:
www.catstuff.com.   600 IN A 216.157.88.24

;; Query time: 974 msec
;; SERVER: ::1#8053(::1)
;; WHEN: Thu Nov 16 20:29:26 +08 2017
;; MSG SIZE  rcvd: 77
```

Et on peut vérifier avec tshark que Stubby parle bien avec Quad9, et en utilisant TLS.

Stubby a l'avantage de bien gérer TCP, notamment en réutilisant les connexions (il serait très coûteux d'établir une connexion TCP pour chaque requête DNS, surtout avec TLS par dessus). Mais il n'a pas de cache des réponses, ce qui peut être ennuyeux si on est loin de Quad9. Pour cela, le plus simple est d'ajouter un vrai résolveur, ici Unbound. On le configure ainsi :

```
server:
   interface: 127.0.0.1
   do-not-query-localhost:  no
forward-zone:
  name: "."
    forward-addr: ::1@8053
```

Avec cette configuration, Unbound va écouter sur l'adresse de bouclage `127.0.0.1` (sur le port par défaut, 53, le port du DNS) et relayer les requêtes pour lesquelles il n'a pas déjà une réponse dans son cache vers Stubby (`::1`, port 8053). Interrogeons Unbound :

```
% dig @127.0.0.1 A mastodon.gougere.fr

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @127.0.0.1 A mastodon.gougere.fr
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40668
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;mastodon.gougere.fr.   IN A

;; ANSWER SECTION:
mastodon.gougere.fr.    600 IN A 185.167.17.10

;; Query time: 2662 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Nov 16 20:36:09 +08 2017
;; MSG SIZE  rcvd: 64
```

Unbound a une mémoire (le cache). Donc, si l'on recommence la requête aussitôt, la réponse arrivera bien plus vite et on verra le TTL diminué.

Engagement

Posté par **Sytoka Modon** (**page perso**) le 18/11/17 à 10:12. Évalué à 5 (+3/-0).

> *Quad9 s'engage à ne pas stocker votre adresse IP ;*
> *leur résolveur est un résolveur menteur : il ne répond pas (délibérément) pour les noms de domaines qu'il estime liés à des activités néfastes comme la distribution de logiciel malveillant.*

Il me semble que la loi française oblige la conservation des IP un an mais que la loi européenne ne soit pas aussi stricte (cf bras de fer entre FDN et la justice). Quad9 stocke actuellement combien de temps les IP dans les log par exemple ? En effet, il y a conservation des IP a des fins de stockage pour traçage / revente… et les log qui sont principalement là en cas de problème (et parfois pour faire des camemberts couleurs pour ceux qui ont du temps). De même, la géolocalisation se fait jusqu'à quel niveau (National, Régional, Communal…) ?

Est-il possible d'avoir la liste ou les listes du menteur comme le fait l'université de Toulouse. C'est bien pratique de pouvoir ré-utiliser ces listes bien faites sans que chacun refasse le monde dans son coin, quitte à participer et proposer des noms à ajouter dans ces listes (cela m'est déjà arrivé pour Toulouse par exemple).

Re: Engagement

Posté par **Guillaume Rousse** (**page perso**) le 18/11/17 à 13:55. Évalué à 3 (+2/-0).

Les obligations légales de conservation de traces en France s'appliquent aux hébergeurs de contenu en ligne, et aux opérateur de télécommunication (les fournisseurs d'accès, en clair). Un DNS public me parait difficilement assimilable à l'une ou l'autre de ces catégories.

Re: Engagement

Posté par **Pol' uX** (**page perso**) le 18/11/17 à 16:55. Évalué à 2 (+0/-0).

On n'est pourtant plus très loin (après dkim, sshfp, dane, et autres joyeusetés) de partager des photos d'Estelle Halliday avec le DNS. :)

--
Adhérer à l'April, ça vous tente ?

Re: Engagement

Posté par **SRL** le 18/11/17 à 14:39. Évalué à 2 (+2/-0).

J'ai l'impression de me connecter à une instance située à Londres. Nous pouvons donc nous asseoir sur les lois française et faire confiance aux five eyes[W].
D'après la politique de vie privé, la géolocalisation va aussi loin qu'elle le peut et est conservé aussi longtemps que possible.

D'après le about. Les mensonges sont fournis par IBM X-Force.

rfc7858

Posté par **karim67** le 18/11/17 à 14:16. Évalué à 3 (+3/-0).

Quoi de neuf ?

On ne peut que se réjouir de l'arrivée d'un acteur qui devrait favoriser la démocratisation de l'usage de DNS sur TLS (rfc7858).
Dans cette quête de "privacy" autour de DNS, il serait malheureux que ce soit une solution quasi propriétaire (DNSCrypt) ou promue par Google (dns-over-https) qui l'emporte.
On regrettera évidemment le choix délibéré de filtrer les réponses mais on ne va pas faire la fine bouche, c'est préférable à tout ce que l'on a vu jusqu'à présent.
Espérons que d'autres initiatives plus neutres viennent à l'avenir concurrencer celle-ci.

Une (micro) contribution en complément à cet excellent journal

De nos jours, devoir se passer de l'authentification TLS du résolveur est, pour certains, rédhibitoire.
Pour pallier ce manque, voici une horreur permettant de "découvrir" la valeur "tls_pubkey_pinset" voulue par Stubby :

```
ip=9.9.9.9
port=853

openssl s_client -showcerts -connect $ip:$port </dev/null 2>/dev/null | openssl x509 -pubkey -noout | openssl pkey -pubin -
outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

Cela permet d'obtenir, à l'heure qu'il est, la configuration Stubby suivante [1] :

```
dns_transport_list:
  - GETDNS_TRANSPORT_TLS
tls_authentication: GETDNS_AUTHENTICATION_REQUIRED
upstream_recursive_servers:
   - address_data: 9.9.9.9
     tls_auth_name: "dns.quad9.net"
     tls_pubkey_pinset:
       - digest: "sha256"
         value: MujBQ+U0p2eZLTnQ2KGEqs+fPLYV/1DnpZDjBDPwUqQ=
```

Un (petit) retour d'expérience de DNS sur TLS

J'ai mené ces dernières semaines quelques expérimentations autour de solutions permettant de mettre en oeuvre le rfc7858 chez soi.

Mon retour d'utilisation de Stubby est mitigé.
Heureux possesseur d'un routeur Turris, j'ai configuré le résolveur Knot pour rediriger les requêtes DNS vers un container LXC qui héberge une instance Stubby (ouf).
J'utilise les trois premiers résolveurs de la configuration par défaut de Stubby qui, depuis ma connexion Orange, offrent une latence acceptable [2] . Cette configuration fonctionne parfaitement… jusqu'à ce qu'elle tombe en panne.
Il faut régulièrement relancer Stubby qui se fige au bout de quelques heures.
Le diagnostic n'est pas aisé car le logging de stubby est, pour l'instant, assez rudimentaire.
Il faudrait effectuer une analyse de trafic réseau.
TLS masquant la couche application, cela pourrait être fastidieux.
Pour ne rien arranger, j'ai autour de moi des utilisateurs exigeants qui n'arrivent pas à comprendre que, sous prétexte d'échapper au regard de la NSA, twitch.tv ou vente-privee.com ne soient régulièrement plus accessibles [3] …

Pour rétablir la paix dans mon foyer, j'ai du me rabattre sur un mode fort bien expliqué ici (et là) qui couple un tunnel TLS à destination du résolveur public (réalisé avec stunnel) avec une instance du résolveur Unbound chargée d'établir un "pont" TCP/UDP.
Cette configuration est la plus stable que j'ai trouvée jusqu'à présent.
Elle présente cependant le défaut majeur d'établir une session TCP/TLS à chaque requête DNS [4] . En plus de ne pas être performant, c'est assez peu respectueux des ressources sollicitées.
Si une bonne âme avait une idée pour configurer stunnel (ou socat) afin de "persister" la session TCP client, je lui serais éternellement reconnaissant :)
En attendant, je vais tenter un retour à Stubby avec Quad9 pour tester si la stabilité est au rendez-vous.

À suivre © …

[1] Les lecteurs attentifs ne manqueront pas d'observer que :

- ce hack permettrait tout aussi bien de récupérer la signature du certificat public d'un attaquant déjà en place,
- une nouvelle valeur devra être appliquée à chaque émission d'un nouveau certificat, ce qui en utilisant une CA comme Let's Encrypt se produira fréquemment. Idéalement, Quad9 devrait utiliser un certificat autosigné (attention, troll inside) avec un délai d'expiration suffisamment long et, comme dit dans le journal, communiquer la clef publique par différents moyens sécurisés.

[2]
À propos de latence (qui est critique en matière de DNS), un traceroute lancé ce jour depuis le réseau Orange montre que le dernier saut avant 9.9.9.9 est l'adresse IP 83.231.233.182 qui s'avère géolocalisée en Grande Bretagne.
Cela laisse penser qu'il n'y a pas (encore) d'instance Quad9 en France.
Dommage avec un nom pareil :(

[3]
Que diraient-ils s'ils savaient que, alors que j'argumente autour de la nécessaire protection de leur vie privée, je m'abstiens de dire que le seul chiffrement du trafic DNS ne les protège pas des regards indiscrets du fait du transport en clair du champ SNI à l'initialisation d'HTTPS…
(Quand on lit ceci qui n'est qu'à l'état de proposition, on a la désagréable impression que ce n'est pas demain que l'on aura un niveau de "privacy" acceptable même avec le fameux cadenas vert).

[4]
L'auteur de la page sus-citée a commis un outil qui cherche à résoudre le problème.

Écriture

Posté par **Wawet76** (**page perso**) le 18/11/17 à 16:57. Évalué à 2 (+0/-0).

> *Alors, le lectorat de LinuxFr.org étant super au courant, va dire « mais des résolveurs DNS publics, il y en a plein !*

La phrase est mal tournée. Et je trouve "le lectorat" un peu impersonnel :)