Compas'2023 : Parallélisme/ Architecture/ Système
LISTIC/USMB - Annecy France, 4-7 juillet 2023

Effets de l'utilisation de transports sécurisés sur les performances d'un resolveur DNS

Etienne Le Louët, Antoine Blin, Julien Sopena, Ahmed Amaou, Kamel Hadadou

Résumé
Des contraintes de performances et de passage à l'échelle ont orienté les choix de conception initiaux de DNS vers l'utilisation de protocoles non chiffrés, le rendant vulnérable à certaines attaques. Des protocoles visant à sécuriser les échanges DNS ont récemment été standardisés, mais la question de leur passage à l'échelle ainsi que de leur soutenabilité reste ouverte. Les travaux présentés dans cet article ont pour but d'étudier le coût de passage du protocole DNS historique à ceux plus récents, en étudiant le surcoût associé à chaque protocole. En comparant différents transports DNS, nous observons que le passage du protocole DNS historique à un plus sécurisé peut mener à une diminution importante des performances.

Mots-clés : DNS, chiffrement, resolver

1. Introduction

Les choix de conception initiaux de DNS a ont été orientés par des problématiques de performance et de passage à l'échelle. L'utilisation du protocole de transport UDP a permis d'obtenir des temps de latence ainsi qu'une charge charge de serveur les plus faibles possibles. Étant donné que ce protocole n'est pas chiffré, les messages DNS sont vulnérables à de l'espionnage ou à la modification de leur contenu. De nouvelles normes, DNS-over-TLS (DoT) [7] et DNS-over-HTTPS (DoH) [5], ont été proposées au sein de l'IETF afin d'y remédier.
Bien que ces nouvelles normes garantissent à la fois la confidentialité et l'intégrité des messages DNS, la question de leur coût reste ouverte : nous ne savons pas si la transition, vers ces nouveaux protocoles de l'ensemble du trafic DNS est viable d'un point de vue énergétique ou environnemental. Ce travail propose une estimation des ressources serveur supplémentaires nécessaires pour passer d'un protocole DNS non chiffré et non connecté à un protocole sécurisé mais coûteux, en mesurant, dans un environnement contrôlé, le coût ajouté par ces protocoles.
Nous avons effectués plusieurs mesures des performances de deux résolveurs mettant en oeuvre ces protocoles sécurisés : une première dans laquelle tous les messages sont envoyés et reçus en utilisant le protocole UDP, afin d'obtenir une base de comparaison, puis d'autres visant à isoler les différences étapes de chacun de ces nouveaux protocoles afin de déterminer leur coût.
Le passage d'UDP à DoH peut entraîner une diminution de 70% des performances (pour les connexions TCP à durée de vie relativement longue). Cette diminution est due au coût de l'échange de clés TLS, mais à la mise en oeuvre du protocole HTTP, ce qui entraîne une baisse des performances. Le reste du document est organisé comme suit : les sections 2 et 3 décrivent le contexte technique et les travaux connexes, la section 4 propose une analyse des performances côté serveur et dans la section 5, nous concluons.

2. Contexte technique

DNS peut être vu comme un registre interrogé par un client afin de trouver (entre autres) l'adresse IP correspondant à un nom de domaine. Il a été mis en œuvre sous la forme d'une base de données arborescente, dans laquelle l'information est distribuée sur plusieurs serveurs ne détenant chacun qu'une fraction de l'information : la recherche d'informations dans DNS est donc un parcours en profondeur de cet arbre, commençant par la racine. Cependant, si chaque client cherchant à traduire un nom de domaine en adresse IP réalisait un tel processus, cela résulterait des temps de latence prohibitifs et une surcharge des serveurs les plus proches de la racine ; c'est pourquoi le parcours de l'arbre est déléguée aux résolveurs, des serveurs qui, lorsqu'ils reçoivent une requête d'un client, y répondent soit à partir de leur cache, soit en effectuant la résolution eux-mêmes.

Contrairement aux autres protocoles web, historiquement basés sur des messages contenant du texte brut, le protocole DNS a été mis en œuvre en utilisant un format de message binaire afin de viser les performances les plus élevées. Les protocoles UDP et TCP ont été choisis pour acheminer les messages DNS. Le premier protocole, UDP, ne requiert aucun établissement de connexion (figure 1 a, annexe), et fournit des performances élevées, mais n'offre aucune garantie de remise et limite la taille de la charge utile des messages. C'est donc le protocole recommandé pour transporter les requêtes DNS standard (qui représentent la majeure partie du trafic DNS). Le second, TCP, nécessite l'échange de trois messages (flèches 1 à 3 sur les figures 1 b c d) pour établir une connexion, avant de pouvoir échanger des message DNS. Il a été principalement utilisé pour transporter des messages DNS dont la taille est supérieure à la charge utile maximale d'un datagramme UDP.

Ces anciens protocoles de transport n'offrent aucune garantie de sécurité. Plusieurs protocoles ont été proposés pour remédier aux failles de sécurité introduites par l'absence de ces garanties. Le premier de ces protocoles sécurisé, DoT [7], utilise une connexion TLS [10] pour assurer l'authentification des pairs ainsi que la confidentialité et l'intégrité des messages. Il s'appuie sur une connexion TCP pour établir une session TLS entre le client et le serveur. Deux messages (flèches 3 et 4 de la figure) sont échangés entre les pairs pour dériver, à partir de leurs paires respectives de clés asymétriques, une clé symétrique utilisée pour chiffrer les messages binaires DNS. Au cours de ce processus, le client valide également l'identité du résolveur en utilisant le certificat numérique de ce dernier. Cependant, comme le protocole utilise un port TCP spécifique (853), il peut facilement être bloqué, forçant un l'utilisation d'un transport non sécurisé ou empêchant la résolution DNS.

Le protocole DoH a été proposé comme alternative pour offrir un transport DNS sécurisé. Il s'appuie sur HTTP/2 [2] pour transporter les messages DNS, contournant ainsi les blocages par pare-feu basés sur le port. DoH peut être considéré comme une couche supplémentaire construite au-dessus d'une session TLS. Une fois cette session établie les flux multiples du protocole HTTP/2 sont utilisés pour échanger les requêtes DNS, soit directement dans leur format binaire en tant que corps d'une requête HTTP POST (flèches 5, 6, 7 et 8 de la figure 1 d), soit encodés en base64url et envoyés en tant que paramètre URL d'une requête GET (étant donné que cette méthode est est moins courante et donc non pris en compte dans le présent document, elle n'est pas présenté sur la figure 1).

3. Travaux connexes

Nous séparons les travaux relatifs à la sécurité de DNS en deux catégories : les travaux qui visent à comparer le coût côté client des protocoles DNS sécurisés, et enfin, les travaux qui se

concentrent sur l'évaluation de l'ampleur de l'adoption de ces protocoles.

Performance côté client

Diverses études se concentrent sur le coût, pour les clients, du passage à DoT ou DoH : Hounsel et al. [6] comparent les temps de chargement de pages web en utilisant différentes combinaisons de transports DNS, de types de réseaux et de résolveurs publics. Boettger et al. [3] comparent également les temps de résolution et la surcharge de protocole de différents transports DNS sécurisés lors de l'utilisation de connexions persistantes ou non persistantes. Ces études montrent que la réutilisation des connexions est bénéfique pour le client et que le DNS sécurisé n'ajoute aucun coût notable en matière de latence pour les clients, sauf sur certains réseaux cellulaires.

Adoption des protocoles

Garcìa et al [4] analysent à la fois le nombre de résolveurs DNS compatible avec les protocoles chiffrés disponibles ainsi que l'utilisation du DNS chiffré par divers utilisateurs. Malgré que le volume du trafic DoH soit resté stationnaire, représentant environ 1% du trafic DNS actuel, le nombre de serveurs DoH disponibles augmente régulièrement, laissant ouverte la question de la durabilité énergétique, si leur utilisation est généralisée.

4. Performances serveur

Le passage d'UDP à des protocoles connectés plus complexes peut entraîner une augmentation de la consommation de ressources matérielles du résolveur. Alors que le traitement par le résolveur des requêtes DNS transportées dans un datagramme UDP nécessite simplement de recevoir le datagramme et d'en envoyer un autre contenant la réponse une fois la résolution terminée, l'utilisation de protocoles basés sur des sessions est plus complexe, car ils nécessitent, afin de recevoir des requêtes et d'émettre des réponses, l'établissement d'une session et la gestion de l'état qui lui est associé.
Le coût de ces étapes supplémentaires soulève des questions sur le passage à l'échelle et la consommation de ressources. Afin d'évaluer leur coût, nous avons réalisé une série de benchmarks synthétiques, en utilisant d'abord DNS over UDP (DoUDP), puis DNS sur TCP (DoTCP), DNS sur TLS (DoT), et DNS sur HTTPS (DoH). Bien qu'il n'offre aucune garantie de sécurité, la mesure des performances de DoTCP reste intéressante car, comme nous l'avons dit dans la section 2, les deux protocoles sécurisés ont été construits sur la base de TCP. Ainsi, la comparaison entre DoUDP et DoTCP est un bon indicateur du coût ajouté par la connexion TCP. En utilisant la même approche, la comparaison entre DoTCP et DoT nous permet de mesurer le coût de performance de l'établissement de la session TLS et du chiffrement du trafic, et la comparaison entre DoT et DoH nous donne des indications sur le coût des couches HTTP/2.
Dans la section 4.1, nous décrivons l'environnement dans lequel ont été effectué nos expériences. La section 4.2 présente les résultats d'un benchmark du protocole UDP, tandis que les sections 4.3 et 4.4 décrivent les multiples benchmarks synthétiques que nous avons réalisées pour caractériser les coûts des différentes étapes des protocoles basés sur les connexions.

4.1. Configuration expérimentale

Nous avons déployé une architecture DNS sur la plateforme Grid5000 [1]. Notre banc d'essai est composé de 22 Dell PowerEdge R640, chacun d'entre eux ayant un CPU à 18 cœurs avec une fréquence de base de 2,2 GHz et une fréquence «turbo» de 3,9 GHz, 96 GiB de RAM et une carte réseau de 25 Gbps, tous reliés ensemble par le même commutateur. De ces 22 machines,

nous exécutons notre résolveur sur l'une d'entre elles, en utilisons vingt comme clients et la dernière comme moniteur d'expérience chargé de déployer et d'exécuter les différents acteurs et outils de mesure sur leurs machines respectives.
Nous avons sélectionné deux résolveurs à tester : knot-resolver, car il est utilisé par des acteurs influents de l'industrie (notamment Cloudflare) et dnsdist, qui n'est pas un résolveur en soi, mais agit comme un relais et un équilibreur de charge entre un client et un autre résolveur, qui répond soit à partir de son cache, soit en transmettant la requête à un autre résolveur. Comme il est compatible avec DoH et DoT, il permet de moderniser une infrastructure DNS existante en ajoutant la prise en charge de ces protocoles sans modifier les logiciels existant. Étant donné que nous avons besoin d'un grand nombre de clients pour atteindre une charge de 100% sur un seul cœur dans notre configuration, nous configurons les deux résolveurs pour qu'ils ne fonctionnent que sur un seul cœur en utilisant les *cgroups*.
Comme nous voulons nous concentrer sur le coût induit par l'utilisation de différents protocoles de transport, nous avons décidé d'exclure de nos mesures le coût du processus de résolution, afin d'éviter le bruit de mesure qui pourrait se produire lors de l'interrogation de serveurs de noms externes non contrôlés. Pour ce faire, nous remplissons, au début de chaque expérience, le cache de knot ou de dnsdist avec les enregistrements DNS qui seront demandés par les clients. Pour réduire autant que possible la variabilité expérimentale, tous les noms de domaine interrogés ont tous la même longueur et le même faux TLD.
Les requêtes DNS sont générées à l'aide de Flamethrower [8], un utilitaire capable de générer des messages DNS compatible avec les protocoles dont nous mesurons les performances. Nous avons corrigé son code pour qu'il puisse garder les connexions sous-jacentes ouvertes pendant une durée configurable, car le comportement par défaut était de les fermer une fois un lot de requêtes envoyé. Un fork de Flamethrower incluant ces changements est disponible sur github [9]. Lors des benchmarks concernant DoH, nous envoyons nos requêtes dans le corps d'une requête POST HTTP/2, car c'est cette méthode qui est utilisée par les clients que nous avons testé (voir 7).

4.2. Base de comparaison

Comme il s'agit du transport le plus ancien, le plus largement utilisé et le plus efficace, la mesure des performances d'UDP nous donne une base de référence. Par conséquent, nous évaluons nos résolveurs en envoyant autant de requêtes que possible, ne nous arrêtant que lorsque nous commençons à enregistrer des pertes systématiques. Au mieux, knot a répondu à 115 000 qps sur les 120 000 qps envoyés par nos clients, tandis que dnsdist a répondu à 225 000 qps, sur les 240 000 envoyés. Nous avons étudié la raison de ces pertes et avons observé qu'elles étaient dues à une saturation du processeur, les deux résolveurs étant incapables de traiter les requêtes à un tel rythme, causant un remplissage du tampon de réception UDP côté noyau, et donc un rejet des paquets. Nous expliquons la différence de performances de presque 50% entre knot et dnsdist par le fait que dnsdist est un proxy et un équilibreur de charge dont le but est de transmettre les requêtes à un serveur en amont aussi efficacement que possible, n'ayant rien d'autre à faire lors de la réception d'une requête que de la transférer ou bien d'y répondre à partir de son cache, alors que knot doit, même dans le cas d'un cache hit, effectuer plus de traitements (comme la vérification et l'application de la politique de requête, le remplissage de la réponse).

4.3. Coût du traitement des requêtes

L'utilisation des protocoles sécurisés peut induire une utilisation plus importante du processeur en raison des étapes supplémentaires requises pour traiter une requête. Ces étapes supplé-

mentaires peuvent être décomposées en deux parties. Tout d'abord, l'ouverture de la connexion, puis le traitement des requêtes individuelles, dont la complexité dépend du protocole utilisé (voir la section 2). Dans cette section, nous nous concentrons sur le coût de cette dernière étape (traitement des requêtes individuelles).

Afin de mesurer le coût supplémentaire par requête, nous devons prendre en considération le nombre de connexions ouvertes simultanément sur lesquelles les requêtes sont envoyées. Pour ce faire, nous avons envoyé une quantité fixe de trafic sur un nombre variable de connexions déjà ouvertes. La quantité fixe totale de requêtes envoyées, ainsi que le nombre minimum de connexions, ont été choisis de manière à ce que l'utilisation du processeur par le résolveur, ainsi que la fréquence du cœur sur lequel il s'exécute soient aussi élevées que possible. Nous avons rencontré un problème lors de l'utilisation de Flamethrower, car nous ne pouvions pas dépasser un certain nombre de requêtes par seconde avec DoH, de sorte que le nombre de requêtes par seconde envoyées n'est pas le même entre DoH et DoT / DoTCP. Cependant, toutes les configurations testées nous ont permis d'atteindre 100% d'utilisation du CPU, ce qui signifie que ce problème n'affecte pas la validité de nos résultats.

Chaque point de la figure 2 (présentée en annexe) présente le nombre moyen de requêtes par seconde auxquelles le résolveur testé a répondu avec succès, les barres présentant les valeurs minimale et maximale atteintes, pour un protocole spécifique et un nombre spécifique de connexions. Nous avons également représenté le trafic maximum traité avec UDP à des fins de comparaison. Pour chaque protocole connecté, il y a une baisse de performance par rapport à UDP, due à la gestion de l'état associé aux connexions. Nous observons une différence notable entre DoTCP et DoT, pour dnsdist seulement alors que pour knot-resolver les performances sont les mêmes. Étant donné que knot-resolver doit exécuter plus de tâches que dnsdist lors de la réception d'une requête DNS (comme indiqué dans la section 4.2), le coût supplémentaire du chiffrement symétrique est absorbé par le coût du traitement des requêtes DNS, et comme dnsdist a moins de travail à effectuer lors de la réception d'une requête, le coût par message ajouté par le chiffrement symétrique a un impact plus important. En comparant DoH à d'autres protocoles, nous constatons, pour les deux résolveurs, une baisse considérable des performances (d'un facteur de deux), expliquée par le coût supplémentaire des couches du protocole HTTP/2. Pour chaque protocole, nous observons que les performances ont tendance à baisser lorsque le nombre de connexions augmente (jusqu'à une baisse de 40% lorsque l'on considère dnsdist sur DoH), à l'exception de dnsdist sur TCP.

4.4. Surcout d'établissement des connexions

Dans l'expérience suivante, nous mesurons le coût d'ouverture et de fermeture des connexions pour chaque protocole. Nous utilisons les mêmes paramètres expérimentaux que ceux utilisés dans l'expérience précédente (figure 2, annexe) pour tracer le point correspondant à 1000 connexions, en tenant compte de l'établissement et de la fermeture des connexions. Ainsi, en comparant cette expérience et la précédente, nous pouvons déduire le coût induit par l'établissement des connexions (établissement de la connexion TCP et/ou échange de clés TLS). Nous effectuons deux séries d'expériences, l'une dans laquelle les connexions durent 30 secondes, l'autre dans laquelle les connexions durent 1 seconde. Cela signifie que, dans la première expérience, 33 connexions sont établies toutes les secondes, tandis que dans la seconde, 1000 connexions sont établies toutes les secondes. Nous effectuons une série d'expériences supplémentaire ou les connexions sont utilisées pour une seule requête, et rouvertes immédiatement, afin de mesurer les performances liées uniquement à l'ouverture de la connexion.

Les résultats de l'expérience sont présentés dans la Figure 3 (annexe) avec le nombre de requêtes par seconde traitées par le résolveur pour chaque combinaison de résolveur, de pro-

tocole et de durée de connexion. Nous avons également représenté le nombre maximum de requêtes par seconde atteint lors de l'expérience précédente (figure 2) à titre de référence pour la comparaison. En général, pour knot-resolver et dnsdist, nous observons les mêmes variations de performance entre les protocoles, mais avec des ordres de grandeur différents :

- Lorsque peu de connexions sont établies (les connexions durent 30 secondes), nous observons une diminution de 20% des performances pour TCP et DoT, par rapport à une situation où aucune connexion n'est établie. Cependant, nous n'observons pas cette perte de performance pour DoH, car le coût CPU de la gestion des requêtes reste le goulot d'étranglement.
- Lorsque la fréquence d'établissement des connexions augmente (les connexions durent 1 seconde), les performances de TCP ne changent pas, tandis que les deux protocoles cryptés voient leurs performances diminuer en raison du coût supplémentaire de l'échange de clés TLS.
- Les performances de DoH sont très proches quel que soit le résolveur lorsque les connexions durent une seconde : dans ce cas, le coût de l'établissement des connexions TLS ainsi que le coût de gestion d'HTTP/2 sont si importants qu'ils masquent totalement la différence de performance des deux résolveurs en matière de traitement des messages.
- Lorsque les connexions sont utilisées pour une seule requête, le coût d'établissement des connexions TCP induit un effondrement des performances pour tous les protocoles.

5. Conclusion

DNS est toujours au cœur de l'internet d'aujourd'hui. Conçu à l'origine pour les performances, à l'aide d'un protocole non chiffré et sans connexion, les préoccupations croissantes en matière de sécurité ont conduit à la normalisation de protocoles sécurisés. Dans cet article, nous avons étudié le coût de la transition vers de tels protocoles du côté du résolveur, en mesurant le coût de chaque étape ajoutée par les protocoles connectés (DNS over TCP, DoT, DoH). Nous avons observé que le passage de l'ancien protocole à un protocole sécurisé entraînait, au minimum, une division des performances par deux, en raison de l'établissement de la connexion TCP, de l'échange de clés TLS et du chiffrement des messages, et que la perte de performance était encore plus importante en ce qui concernait le passage à DoH en raison des couches protocolaires ajoutées par HTTP/2, ce qui, compte tenu du fait que l'utilisation de ce protocole semble être poussée par l'industrie, est contradictoire avec les objectifs initiaux d'efficacité du DNS. De plus, dans le cas où les connexions sont courtes, les performances sont encore plus affectées, ce qui peut conduire à une forte augmentation du nombre de résolveurs ainsi que de leur consommation d'énergie, entraînant à son tour un coût d'exploitation et environnemental plus élevé.

En l'état actuel des choses, le transfert de 100% du trafic DNS vers DoH ne parait pas viable. Pour réaliser cette transition, il faudra veiller à ce que les clients puissent maintenir leurs connexions en vie autant que possible, et utiliser des protocoles moins coûteux que HTTP/2, qui conservent leur capacité à traverser les pare-feu. Il pourrait donc être intéressant à l'avenir de se pencher sur le protocole HTTP/3, basé sur le QUIC, qui est toujours en cours de développement.

6. Annexe 1 - Figures

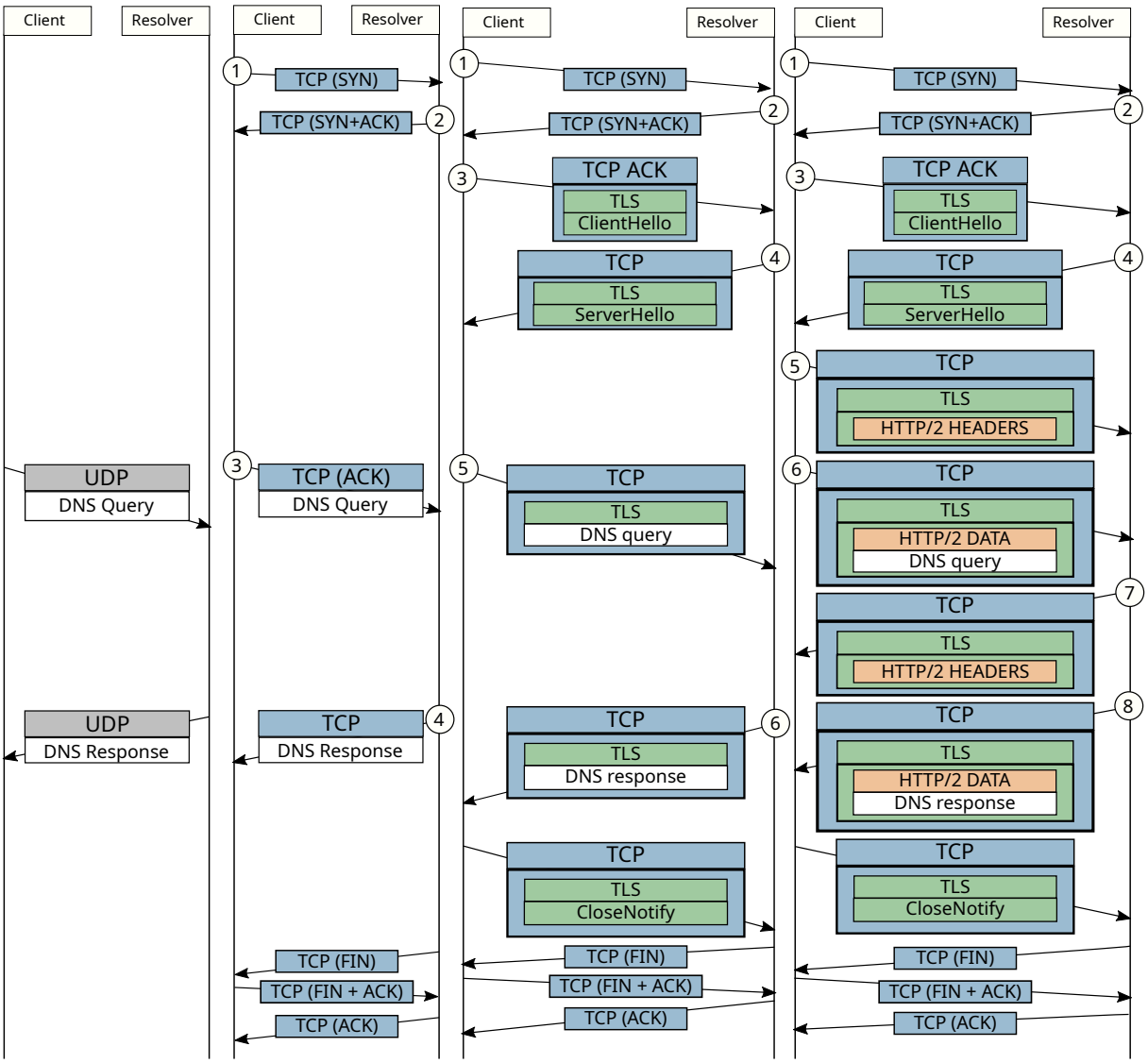

FIGURE 1 – Comparison de DNS over UDP, TCP, TLS et HTTP/2

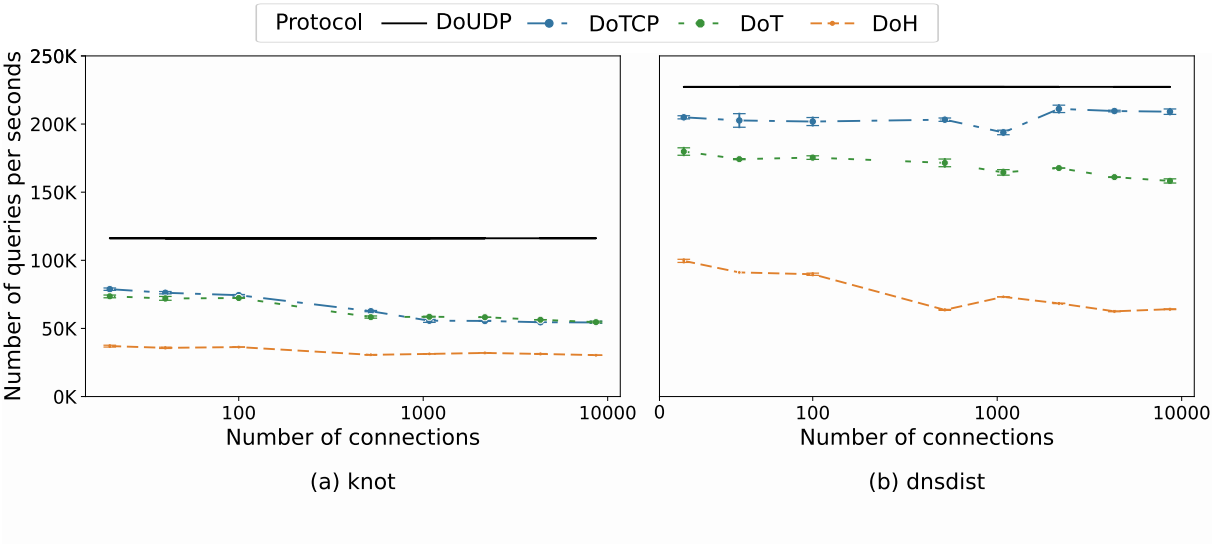

FIGURE 2 – Requetes par seconde traitées par les deux résolveurs lorsque les connexions durent toute l'expérience

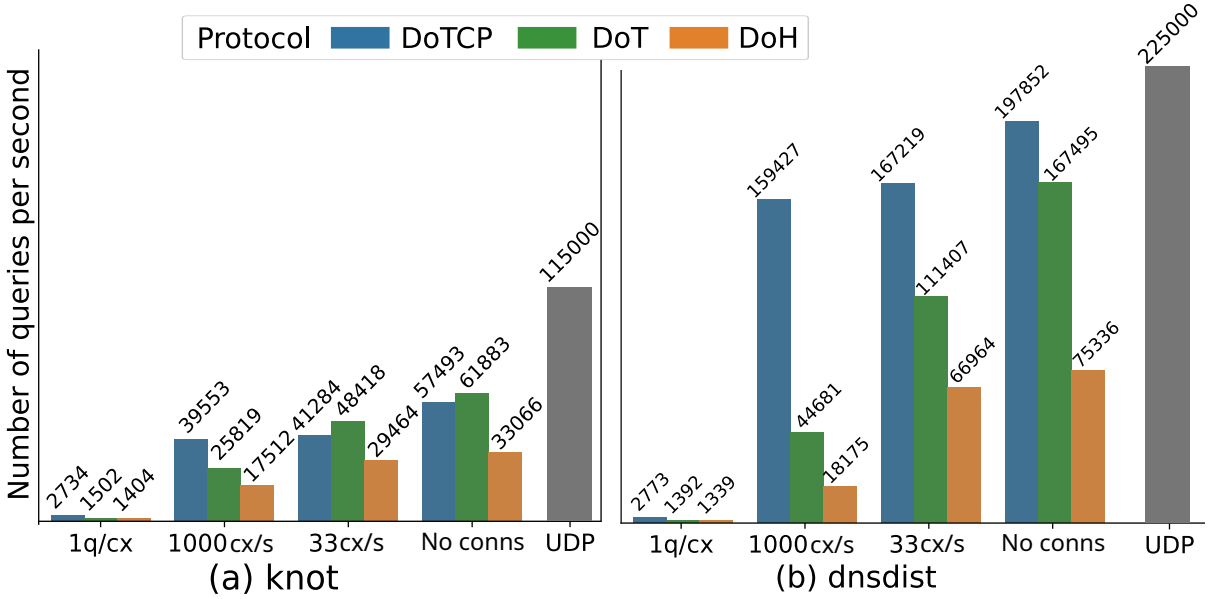

FIGURE 3 – Requêtes par seconde traitées par résolveur et par protocole en fonction du nombre de connexions par seconde

7. Annexe 2 - Etude comportement clients

Étant donné que les nouveaux transports DNS sont basés sur des connexions, de nouvelles questions se posent : si le protocole définit la manière d'interroger le service, il ne précise pas comment les connexions sous-jacentes doivent être gérées par le client et le résolveur. La séquence d'ouverture des connexions et d'envoi des requêtes est principalement contrôlée par le client, mais se concentrer uniquement sur le comportement du client n'est pas suffisant, car le serveur a le choix d'accepter, de rejeter, de fermer ou de maintenir les connexions ouvertes.
L'objectif de cette expérience est de caractériser la forme du trafic entre les clients existants et les résolveurs accessibles au public, afin de pouvoir générer un trafic similaire lors de la mesure des performances côté serveur. La section 7.1 décrit le dispositif expérimental utilisé pour les mesures, tandis que la section 7.2 contient une analyse des différents comportements observés chez les clients et les résolveurs.

7.1. Configuration expérimentale

Nous avons choisi comme clients deux navigateurs web : Firefox v91.5 et Chromium v101.01, car ils annoncent l'utilisation de protocoles sécurisés en proposant un résolveur interne compatible avec DoH. Nous avons également testé DNScrypt-proxy v2.0, un client qui agit comme un démon et remplace le résolveur interne d'un système, permettant l'utilisation transparente des protocoles DNS sécurisés pour tous les logiciels présents sur le système, ainsi que le partage de la connexion TLS sous-jacente. Nous avons choisi de concentrer cette analyse sur le protocole DoH qui a gagné plus de terrain que DoT, et qui est donc intégré dans les navigateurs. Pour les résolveurs publics, nous avons choisi trois acteurs majeurs très utilisés par le public : Quad9, Google et Cloudflare. Nous rassemblons la liste des sites web utilisés pour la résolution à travers une liste publique de noms de domaines [18], filtrée pour garder ceux qui ont encore un enregistrement A correspondant à un serveur acceptant le trafic HTTP. Afin de générer le trafic approprié, nous configurons le navigateur avec le résolveur DoH sélectionné, puis nous chargeons un script JavaScript qui effectue des requêtes HTTP à différents rythmes (une toutes les 50 ms, 1000 ms et 60 000 ms) pendant une période de 30 minutes. Pour générer du trafic à l'aide de DNScrypt-proxy, nous utilisons un programme C qui effectue des résolutions UDP aux mêmes taux que ceux configurés pour le navigateur, en utilisant le résolveur configuré du système, qui, dans ce cas, est DNSCrypt-proxy. Nous caractérisons la forme du trafic en mesurant le nombre de connexions et, pour chaque connexion, sa durée, le nombre de requêtes qui y ont été envoyées, ainsi que l'origine (client ou serveur) et la méthode (FIN ou RST) de fermeture de la connexion.

7.2. Résultats

La figure 4 présente, pour chaque logiciel et combinaison de délai inter-requête et de résolveur testés, l'utilisation des connexions : chaque rectangle représente une connexion entre un client et un résolveur, sa longueur représentant la durée de la connexion. Les connexions inférieures à une seconde sont représentées par une croix. Par manque de place, nous avons choisi de ne présenter que le profil de Chrome, celui de Firefox étant similaire.

DNScrypt-proxy

De tous les logiciels utilisés, DNScrypt-proxy est celui qui génère la charge la moins agressive envers le serveur. En effet, son comportement principal est d'ouvrir et de maintenir ouverte une seule connexion TCP qu'il utilisera pour effectuer toutes les requêtes, quelle que soit l'intensité du trafic généré. De plus, un timer interne est mis en place pour déclencher la fermeture

de la connexion TCP lorsqu'elle est inutilisée, libérant ainsi les ressources du serveur (ligne inférieure de la figure 4b).

Navigateurs web

Les navigateurs utilisent les ressources DNS de manière plus agressive : au début des sessions, ils tentent de maximiser la probabilité d'une connexion réussie au serveur DNS en ouvrant plusieurs connexions en parallèle au même serveur, ce qui accélère probablement les premières résolutions que les navigateurs effectuent habituellement (figure 4a), entraînant une augmentation de l'utilisation des ressources du serveur. L'utilisation ultérieure de ces connexions ouvertes dépend de l'intensité du trafic. Lorsque l'intensité du trafic est faible, avec une fréquence de requête inférieure à 1 requête par seconde (voir les deux lignes inférieures de la figure 4a), une seule connexion est principalement utilisée pour gérer le trafic, les connexions restantes étant éventuellement fermées. Nous observons parfois des fermetures de connexion, forçant une réouverture (ligne de délai de 1000 ms sur la figure 4a), ou plusieurs connexions en même temps (lignes de délai de 1000 ms et 60 000 ms sur la figure 4a), mais ces événements ne sont pas assez nombreux pendant la durée de vie d'une expérience pour être significatifs. En cas de trafic DNS intense (plus d'une requête par seconde), le schéma de connexion des navigateurs web change radicalement. Non seulement le navigateur ne parvient pas à générer le trafic que nous demandons, mais nous observons également que les connexions sont ouvertes et fermées en séquence, chaque fermeture de connexion provenant du client (voir la ligne supérieure de la figure 4a) et chaque connexion étant utilisée pour acheminer peu ou pas de demandes. Du point de vue du serveur, ce comportement représente le pire des cas, car chaque ouverture de connexion étant coûteuse, il en résulte une énorme consommation de ressources.

Resolvers

Cependant, les clients ne sont pas les seuls responsables des schémas de connexion. Les résolveurs ont le choix d'accepter ou de refuser les connexions et le trafic provenant des clients. Nous avons observé que Google a la configuration de résolveur la plus permissive de celles que nous avons testées, car nous n'avons observé aucune limitation en termes de nombre de connexions, de durée ou de nombre de requêtes par seconde (qps) par connexion. Cloudflare n'impose aucune restriction sur la durée de la connexion, alors que le nombre maximum de requêtes par connexion est limité à 10000 (figure 4b, graphique en haut à gauche). Contrairement à Cloudflare, quad9 n'impose aucune limite au nombre de requetes par connexion, mais ferme les connexions inutilisées après environ 20 secondes d'inactivité (figure 4a, graphique en bas à droite).
Le comportement voulu des clients et des résolveurs semble être de maintenir une connexion TCP/TLS en vie pendant qu'elle est en cours d'utilisation.

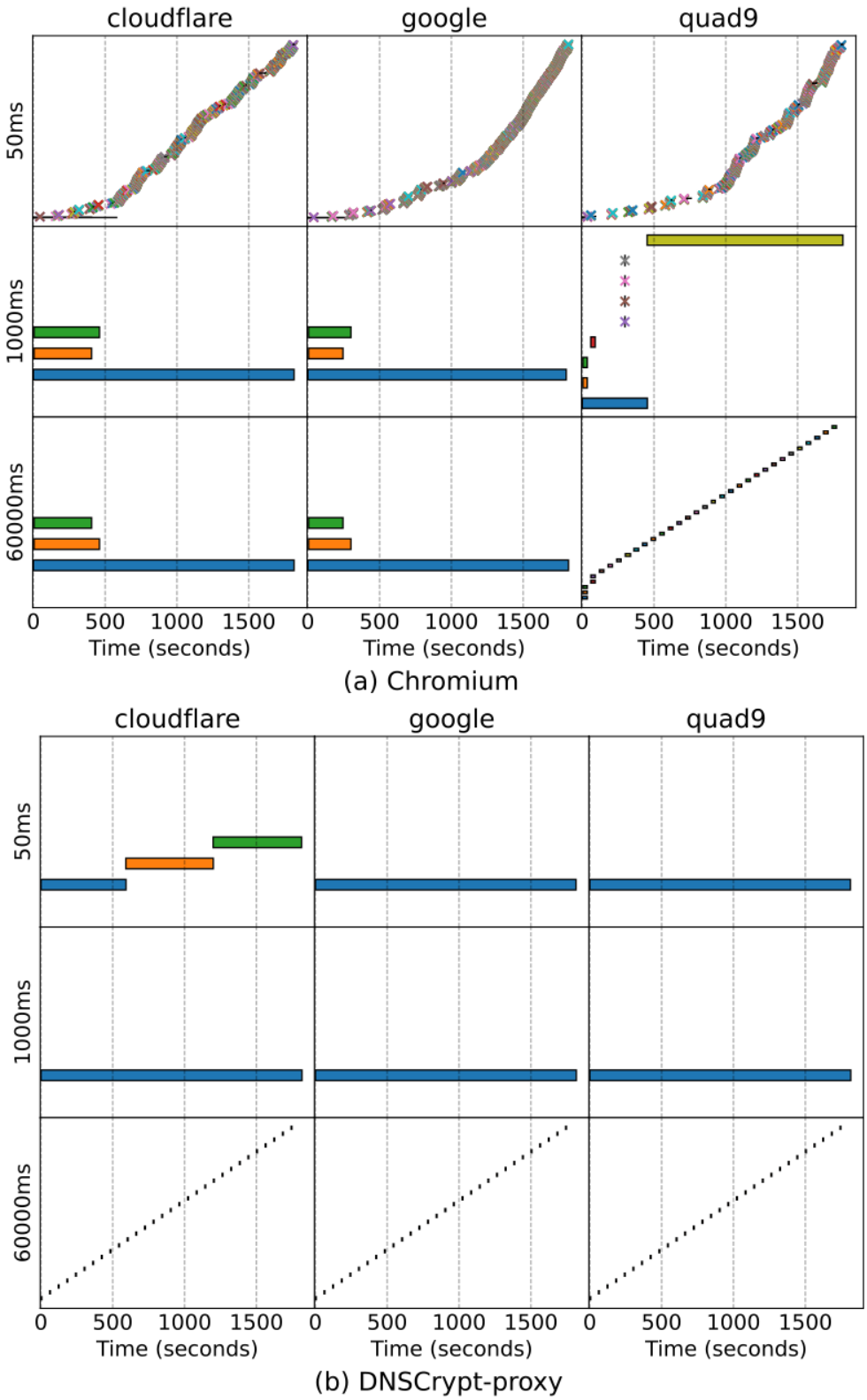

FIGURE 4 – Connexion utilisée par Chromium et DNSCrypt-proxy pour un ensemble de délais de résolution et de requête

Bibliographie

1. Balouek (D.), Carpen Amarie (A.), Charrier (G.), Desprez (F.), Jeannot (E.), Jeanvoine (E.), Lèbre (A.), Margery (D.), Niclausse (N.), Nussbaum (L.), Richard (O.), Pérez (C.), Quesnel (F.), Rohr (C.) et Sarzyniec (L.). – Adding virtualization capabilities to the Grid'5000 testbed. *In : Cloud Computing and Services Science*. – 2013.
2. Belshe (M.), Peon (R.) et Thomson (M.). – *Hypertext Transfer Protocol Version 2 (HTTP/2)*. – RFC n7540, RFC Editor, May 2015.
3. Böttger (T.), Cuadrado (F.), Antichi (G.), Fernandes (E. L. a.), Tyson (G.), Castro (I.) et Uhlig (S.). – An empirical study of the cost of dns-over-https (imc '19). – In *Internet Measurement Conference*, 2019.
4. Garcia (S.), Hynek (K.), Vekshin (D.), Cejka (T.) et Wasicek (A.). – Large scale measurement on the adoption of encrypted dns. *arXiv preprint arXiv :2107.04436*, 2021.
5. Hoffman (P.) et McManus (P.). – *DNS Queries over HTTPS (DoH)*. – RFC n8484, RFC Editor, October 2018.
6. Hounsel (A.), Borgolte (K.), Schmitt (P.), Holland (J.) et Feamster (N.). – *Comparing the Effects of DNS, DoT, and DoH on Web Performance*. – 2020.
7. Hu (Z.), Zhu (L.), Heidemann (J.), Mankin (A.), Wessels (D.) et Hoffman (P.). – *Specification for DNS over Transport Layer Security (TLS)*. – RFC n7858, RFC Editor, May 2016.
8. ns1labs. – flamethrower.
9. ns1labs (E. L. L.). – flamethrower.
10. Rescorla (E.). – *The Transport Layer Security (TLS) Protocol Version 1.3*. – RFC n8446, RFC Editor, August 2018.