Encrypted DNS → Privacy ? A Traffic Analysis Perspective

1. Résumé :

Toutes les connections Internet sont précédées d'une recherche DNS sans protection du trafic, ce qui implique la possibilité pour des entités de surveiller et tracker les utilisateurs, voire de les censurer. Pour remédier à cela, de grandes entreprises telles que Google, Cloudflare et Mozilla ont standardisé des protocoles permettant de chiffrer le trafic DNS : DNS-over-TLS (DoT) et DNS-over-HTTPS (DoH). Ils permettent une protection contre l'analyse du trafic par bourrage du trafic ou multiplexage du trafic DNS avec d'autres trafics. Cependant, même si les communications sont chiffrées, certaines caractéristiques du trafic tel que le volume et le temps peuvent révéler des informations sur son contenu.

Dans ce document, les auteurs ont étudié l'efficacité des attaques par analyse du trafic en se concentrant sur le protocole DoH, et ils ont comparé la protection offerte par DoH par rapport à DoT. Pour cela, ils ont considéré un adversaire passif dont le but est d'identifier quelles pages web l'utilisateur visite, ou de pratiquer de la surveillance ou de la censure. Les caractéristiques utilisées pour une attaque sur le trafic web n'étant pas applicables sur le trafic DNS, il ont conçu un nouvel ensemble de fonctionnalités permettant des attaques efficaces pour identifier des sites web visités sur un trafic DNS encrypté. Le trafic DNS étant composé de petits paquets, leur technique nécessite 124x moins de données que celle utilisée pour analyser le trafic web, et est tout autant efficace, voire plus. Les fonctionnalités permettent également à l'adversaire de connaître l'environnement sur lequel l'attaque sera déployée, contrairement à la plupart des anciens travaux sur le sujet, ce qui permet une attaque de haute performance sur tous les environnements.

Ils ont alors testé leur attaque sur des défenses existantes du trafic, et ont remarqué que la solution de bourrage EDNS0 (par Google et Cloudflare) ne pouvait pas complètement empêcher leur attaque. En revanche, bien que Tor ne témoigne pas d'une grande protection contre l'empreinte de l'utilisateur sur des sites web sur le trafic web, il est extrêmement efficace contre celle sur le trafic DNS encrypté.

Ils ont ensuite montré que même en encryptant le trafic DNS, il est possible pour l'attaquant de trouver le paquet contenant la recherche DNS pour le 1er domaine. De plus ils ont quantifié le compromis entre le nombre de données d'un site blacklisté qu'un utilisateur peut télécharger, et le nombre de sites non-blacklistés qui sont censurés par effet secondaire du blocage basé sur l'analyse du trafic. Ils ont également rassemblé un premier ensemble de données de DNS encrypté collectés dans une grande gamme d'environnements.

2. Protocoles techniques & vocabulaire :

Le DNS (Domain Name System) : Permet de translater des domaines faciles à lire en adresses IP numériques (= résolution de domaine). Pour cela le client envoie une requête DNS à un résolveur récursif (un serveur avec des capacités de résolutions et de cache), et si la résolution de domaine n'est pas dans le cache du résolveur récursif, il contacte un certain nombre de serveurs de noms faisant autorité contenant une base de données du mappage entre noms de domaines et adresses IP. Le résolveur récursif va alors parcourir cette base de données jusqu'à trouver la réponse à la requête DNS qu'il a reçu. Il va alors la transmettre au client et ce dernier pour utiliser cette adresse IP pour se connecter à l'hôte de destination.

Le DNS fingerprinting : C'est l'utilisation de l'analyse du trafic de la page web qui a généré une trace DNS chiffrée.

Open-world : L'adversaire n'a accès qu'aux sites surveillés et l'utilisateur peut avoir accès à des sites en dehors de cette liste.

Closed-world : L'adversaire connaît l'ensemble des pages web que l'utilisateur peut visiter.

Le DoT (DNS-over-TLS) (standardisé en 2016) : Le client établit une session TLS avec un résolveur récursif en utilisant un port TCP dédié pour le trafic DNS (facile à tracker et bloquer), et échange des requêtes et réponses DNS sur une connexion chiffrée.

« DNS over TLS est un protocole de sécurité pour le chiffrement et l'encapsulation des requêtes et des réponses DNS via le protocole TLS. Le but de la méthode est d'augmenter la confidentialité et la sécurité des utilisateurs en empêchant les écoutes et la manipulation des données DNS via des attaques man-in-the-middle. » (Wikipédia)

Le DoH (DNS-over-HTTPS) (standardisé en 2018) : Le résolveur DNS local établit une connexion HTTPS avec le résolveur récursif et encode les requêtes DNS dans le corps des requêtes HTTP (pousse les réponses DNS qui vont avoir tendance à suivre une recherche DNS, ce qui réduit la latence).

« DNS over HTTPS (DoH) est un protocole permettant d'effectuer une résolution DNS (Domain Name System) à distance via le protocole HTTPS. L'un des objectifs de la méthode est d'accroître la confidentialité et la sécurité des utilisateurs en empêchant les écoutes clandestines et la manipulation des données DNS par des attaques de type man-in-the-middle. » (Wikipédia)

L'attaque de l'homme du milieu ou man-in-the-middle attack, parfois appelée attaque de l'intercepteur, est une attaque qui a pour but d'intercepter les communications entre deux parties, sans que ni l'une ni l'autre ne puisse se douter que le canal de communication entre elles a été compromis.

3. Expérimentations :

Expérience 1 : Savoir s'il est possible de déduire les sites web qu'un utilisateur visite en observant le trafic DNS chiffré.

La collecte de données :

- On lance des machines virtuelles sur Ubuntu avec des clients DoH qui lancent des requêtes DNS à un résolveur DoH public
- On visite une page web de la liste, déclenchant des recherches DNS (avec Sélénium)
- On restart le navigateur entre chaque visite pour être sûr que le cache et le profil n'affectent pas la collecte
- On capture le trafic du réseau entre le client DoH et le résolveur (avec tcpdump)
- On filtre le trafic avec le port de destination et l'IP pour obtenir la trace du trafic DoH final.

La sélection des données :

- Suppression des effets de fausses erreurs (ex : pages qui ont shutdown pendant la période de collecte).
- Elimination des comportements de sites web qui génèrent des erreurs de classification non relatées aux caractéristiques du trafic DNS (ex : pages qui dirigent vers d'autres pages/la même ressource ou serveurs web qui retournent les mêmes erreurs (404)).

L'analyse de données :

- Pour collecter requêtes et réponses HTTP (pas les réponses DNS), on utilise le moteur d'exploration Chrome sur Sélénium. Il y a 2 checks que l'on va répéter plusieurs fois sur nos collections :
    - On regarde le statut de la réponse HTTP du « top level domain » (ex : .fr …) (domaine de haut niveau) (ex : l'URL demandée par le client). On identifie alors les pages web qui n'ont pas un statut HTTP OK (celles qui ont des erreurs 404 etc) et on marque ces domaines comme « conflicting » (en conflit)
    - On checke que le « top level domain » soit bien présent dans la liste des requêtes et des réponses pour être sûrs que la page demandée par le client ne renvoie pas le navigateur vers d'autres URLs. (Cela peut déclencher des fausses alarmes (ex : pages qui renvoie vers des URL de la version française du site (.com vers .fr) etc) -> on ne considère pas ces alarmes comme des anomalies).

Expérience 2 : Tester l'hypothèse selon laquelle la performance des attaques de l'état de l'art du trafic web diminue de 20% quand elles sont appliquées sur des traces DoH.

Expérience 2.0 : Création d'un nouveau set de caractéristiques

Introduction d'un nouveau set de caractéristiques composé de n-grammes de longueur d'enregistrement TLS dans une trace :

- On représente le trafic comme une séquence d'entiers où la valeur absolue est la taille de l'enregistrement TLS et où le signe indique la direction (+ pour les paquets du client vers le résolveur (outgoing) et – pour les paquets du résolveur vers le client (incoming)).
  (ex : en uni-grammes : (-64),(88),(33),(-33), en bi-grammes : (-64, 88)(88, 33)(33, -33))
- Pour créer les fonctionnalités, on prend des tuples de n longueur d'enregistrements TLS consécutifs dans la trace du trafic DoH et on compte leur nombre d'apparition dans chaque trace.

- On étend la représentation en n-grammes aux éclatements de trafic (traffic bursts) (=séquences consécutives de paquets dans la même direction). Pour cela on ajoute des longueurs sur les paquets d'une direction dans le tuple.
  (ex : uni-gramme (-64, 121,-33), bi-grammes (-64, 121)(121, -33))
- On expérimente avec des uni, bi, et tri-grammes pour chaque type (web et DoH). Puis avec le temps des paquets

Sélection de l'algorithme d'analyse

- On utilise l'algorithme RF (Random Forest : ensembles d'arbres de décisions qui utilisent une structure de données en arbre où un nœud = une condition et une branche = une décision par rapport à la condition. La décision finale du RF est une fonction qui fait le total des décisions de ses arbres).
- On utilise 100 arbres.

Expérience 2.1 : Evaluation de l'efficacité de notre classificateur (Precision, Recall & F1-Score)

- Un site est positif s'il a bien été identifié, négatif si on l'a classifié dans le mauvais label
  Precision = total des vrais positifs/nombre de sites qu'on avait id positifs (faux + vrais positifs)
  Recall = total des vrais positifs/total des positifs (vrais positifs + faux négatifs)
  F1-Score = moyenne harmonisée de Precision et Recall
  On va utiliser une validation croisée 10-fold pour éliminer les informations biaisées dues à la superposition des expérimentations.

Evaluation des caractéristiques n-grammes :

- Dans les closed & open worlds
  - **Closed world** : avec le data-set LOC1 on utilise 3 réglages :
    - Un adversaire qui attaque le datas-set entier de 1500 sites web
    - Un qui attaque le data-set après la sélection des données de 1414 sites web
    - Un qui attaque le data-set entier mais qui considère les versions de pays des sites comme équivalentes (si sur le .fr et .uk il y a erreur commune classifiée, on considère le site comme vrai positif).

    On utilise un graphe de confusion (confusion graph) qui permet une représentation intuitive des erreurs de classification permettant de visualiser les collisions entre sites.

    On étudie ensuite la performance du classificateur pour des sites individuels : On utilise un scatterplot où chaque point=un site web, et leur couleur représente la différence absolue entre la Precision et le Recall (bleu si |Precison – Recall|= 0, rouge si |Precision – Recall| = 1).

    - Si basse précision (d'autres sites sont souvent id à ce site) mais haut Recall (souvent id par classificateur) -> good privacy
    - Si bas Recall (casi jamais id) mais haute précision (si id, l'adversaire est absolument sûr que c'est correct) -> permet au censeur de bloquer sans peur de dommages collatéraux.
  - **Open world** : On considère deux sets de pages web, les surveillées (monitored) et les non-surveillées (unmonitored). Le but de l'adversaire est de déterminer si une trace test correspond à une page du set surveillé.

- On créé des samples non-surveillés avec les traces de 5000 pages web formées par un mix de LOC1 et OW data-sets.
- On a 1% des classes qui sont dans le set surveillé, et 10% utilisées pour l'entraînement (le set d'entrainement contient autant de samples surveillés que non surveillés). Le set de test contiendra autant de samples venant du set d'entrainement que de samples jamais vus par le classificateur.
- Pour savoir si le sample est surveillé ou pas, on utilise une méthode de Stolerman et al. : on assigne un sample à la classe surveillée ssi le classificateur prédit cette classe avec une proba > seuil t, sinon il va dans la classe non surveillée.

Comparaison avec le fingerprinting du trafic web

- On va comparer nos résultats avec le fingerprinting du trafic web en utilisant des attaques de l'état de l'art : the k-Fingerprinting Attack par Hayes et Danezis, et le Deep Fingerprinting (DF), en considérant un closed-world de 700 sites web, et en utilisant un algo de RF avec les mêmes paramètres que notre classificateur. On évalue les performances :
  - Seulement sur le trafic DoH
  - Seulement sur le trafic HTTPS correspondant au trafic du contenu web
  - Sur un mélange des 2

Expérience 2.2 : Test de la robustesse du fingerprinting sur DNS

On va tester la robustesse selon 3 facteurs : le temps, l'espace, et l'infrastructure.

- **Le temps :** Les traces DNS varient à cause du dynamisme du contenu des pages web et des variations dans les réponses DNS. Pour comprendre l'impact de ces variations sur le classificateur, il faut :
  - Collecter des données de LOC1 pour 10 semaines. Diviser cette période en 5 intervalles, contenant chacun 2 semaines consécutives.
  - Reporter le F1-score du classificateur dans un tableau quand on entraîne le classificateur sur des données d'un seul intervalle, et utiliser les autres intervalles comme des données de test.
- **L'espace :** Les traces DNS peuvent varier selon la location. En effet, les recherches DNS varient quand les sites web adaptent leur contenu à des régions spécifiques, les ressources populaires des résolveurs varient selon les régions, et les résolveurs et CDNs utilisent des méthodes de géo-location pour équilibrer leur charge de requêtes. Nous allons donc :
  - Collecter des datas dans 3 locations : 2 en Europe (LOC1 et 2) et une en Asie (LOC3)
  - Nous allons reporter le F1-score dans un tableau lorsque nous entraînons le classificateur
- **L'infrastructure :** On étudie comment le résolveur DoH, le client et la plateforme utilisateur affecte la performance de l'attaque.
  - <u>Influence du résolveur DoH</u>
    On étudie 2 résolveurs DoH commerciaux : celui de CloudFlare et celui de Google.
    - On instrumente une nouvelle collection de paramètres en utilisant Firefox dans sa configuration « trusted recursive resolver » avec les deux résolveurs DoH.
    - On compare ensuite les résultats des deux résolveurs. Nous devons observer que le classificateur performe de la même manière sur les 2, et que pour le paramètre de la localisation, il y a une diminution asymétrique dans une des directions : s'entraîner sur le dataset de Google and attaquer Cloudflare doit

donner un F1-Score de 13% alors qu'attaquer Google sur un classificateur entraîné par Cloudflare donne des résultats similaires à un entraînement par Google lui-même.
        - Pour comprendre cette asymétrie, on peut comparer les 15 plus importantes caractéristiques pour les datasets de CloudFlare et ceux de Google dans un diagramme. Nous devons observer qu'il manque des caractéristiques importantes pour Google dans les datasets de Cloudflare ce qui fausse les arbres et donc explique la baisse de performance. (Il manque aussi des caractéristiques de Cloudflare dans les datasets de Google, mais ceux-ci sont de faible importance donc négligeables.)
    - Influence de la plateforme utilisateur
        - On collecte des traces pour le top 700 des pages web Alexa sur une Raspberry Pi (RPI dataset) et sur un ordinateur Ubuntu (DESKTOP dataset) venant tous les deux du LOC1.
        - On observe les performances du classificateur (bonnes quand on s'entraine sur la même plateforme, très mauvaises quand on croise les datasets).
        - On observe les tailles d'enregistrement TLS sur les deux plateformes (DESKTOP plus longues que RPI).
        - On répète la classification croisée après avoir ajouté 8 octets à tous les enregistrements TLS de la RPI. On observe à nouveau les performances du classificateur (grande amélioration du F1-Score)
    - Influence du client DNS
        - On considère différents setups de clients : le « trusted recursive resolver de Firefox ou TRR (CLOUD), le client DoH de Cloudflare avec Firefox (CL-FF), et le client Cloudflare avec Chrome (LOC2).
        - On collecte ces datasets dans la location LOC2 avec le résolveur de Cloudflare.
        - On observe les résultats du classificateur (bons sur le même setup client, mauvais quand le setup change, tend vers 0 quand on utilise différents navigateurs -> cela est dû aux différences d'implémentations entre les clients DoH de Firefox et de Cloudflare).
        - Observer le trafic chiffré : Utiliser un proxy pour man-in-the-middle la connexion DoH entre le client et le résolveur, en obtenant les clés de sessions TLS OpenSSL avec les scripts de Lekensteyn (https://git.lekensteyn.nl/peter/wireshark-notes). Avec ce proxy on peut déchiffrer les captures DoH pour Firefox configuré pour utiliser le résolveur Cloudflare, mais ça ne marche pas pour Google. Pour Google, il faut man-in-the-middle un client curl-doh (https://github.com/curl/doh) qui a aussi des traces sur courtes que Firefox.

Expérience 2.3 : Détermination de quels sites sont les survivants et quels sont les proies faciles

On va déterminer quels sites sont bons et lesquels sont mauvais pour éluder le fingerprinting sous toutes les configurations.

- On calcule la moyenne des F1-Score de toutes les configurations comme mesure agrégée de la performance globale de l'attaque.
- On trace le CDF (Cumulative Distribution Function) de la distribution des F1-Score moyens sur le site web.

- On classe les sites par plus petit F1-Score moyen et par plus petite déviation standard. Au top du classement se trouve les survivants aux attaques dans toutes les configurations. En bas on aura les sites avec un long nom de domaine, avec peu de ressources et qui ont l'air de peu varier.

Expérience 3 : Evaluation de l'efficacité des techniques existantes to protéger le trafic web de l'analyse de trafic

Nous allons considérer 2 différentes défenses :

- Le EDNS Padding (Extension mechanisms for DNS) qui améliore les fonctionnalités du protocole DNS. Il spécifie comment ajouter du padding (bourrage) sur les clients DoH et sur les résolveurs. Pour les clients, il est recommandé de bourrer les requêtes DNS par le multiple le plus proche de 128 octets. Pour les résolveurs, de 468 octets.
  Le client DoH de Cloudflare permet l'utilisation de l'EDNS padding :
  - On modifie le code source du client pour suivre la stratégie de bourrage décrite ci-dessus.

  La spécification de Google mentionne le padding EDNS mais il n'a pas été trouvé d'option permettant son activation donc on ne l'analysera pas.

  On veut aussi voir si des mesures simples à utiliser côté utilisateur qui altèrent le motif de requêtes, comme un bloqueur de pub, pouvaient être une contre-mesure efficace.

  - On conduit donc une expérience où l'on utilise Sélénium avec une instance Chrome ayant l'extension AdBlock Plus. On la fait avec des requêtes et réponses DoH bourrées par des multiples de 128 octets.
  - Afin d'évaluer les recommandations de padding, on utilise un proxy HTTPS (mitmproxy) entre le client DoH et le résolveur Cloudflare. Celui-ci interceptera les réponses du résolveur DoH Cloudflare, des bandes du padding existant, et les réponses du bourrage par le plus proche multiple de 468 octets.
  - On observe les résultats : aucune stratégie de padding n'est 100% efficace.
  - Pour comprendre les limites du padding, on simule un paramétrage où le padding est parfait : tous les enregistrements ont la même taille et le classificateur ne peut pas exploiter la taille de l'enregistrement TLS. Pour cela, nous mettons artificiellement la taille de tous les paquets du dataset à 825, soit la taille maximale observée dans le dataset.

- Le DNS-over-Tor. Tor est un réseau de communication anonyme. Cloudflare permet son utilisation pour protéger la privacité de ses utilisateurs par la mise en place d'un résolveur DoH pouvant être accédé en utilisant Tor. Cela permet de ne pas révéler leur IP au résolveur lors de recherches DNS. Nous savons que les mesures de Tor ne sont pas efficaces pour protéger le trafic web, et nous allons voir si elles le sont pour protéger le trafic DNS.

Nous comparons alors l'efficacité de ces deux défenses.

Puis, nous allons évaluer le coût de communication des défenses. Pour chaque contre-mesure, on collecte 10 échantillons de 50 pages web, avec et sans contre-mesures, et nous mesurons la différence du volume de données totale échangées entre le client et le résolveur.

Expérience 4 : Comparaison entre DoH et DoT

- On utilisera le dataset DOT.

- Comme pour DoH, le résolveur DoT de Cloudflare implémente le EDNS0 padding des réponses DNS par un multiple de 128 octets. Mais le client DoT cloudflare ne supporte pas le trafic DoT. On utilisera donc un client Stubby pour questionner le DoT Cloudflare bourré par un multiple de 128 octets.
- On observe les résultats (DoT meilleur que DoH car moins de variance que DoH et le trafic est donc moins unique). On peut faire un histogramme des tailles des enregistrements TLS reçus et envoyés pour 100 pages web à la fois sur DoT et sur DoH pour comparer les résultats.

Expérience 5 : Etude de la faisabilité pour un censeur d'utiliser l'analyse de trafic pour identifier le domaine qui est recherché.

Pour bloquer un site, le censeur doit non seulement identifier la recherche du trafic chiffré mais doit aussi le faire le plus rapidement possible afin d'empêcher l'utilisateur de télécharger du contenu.

Unicité des traces DoH

Pour que le censeur puisse identifier les domaines d'un trafic DoH donné, il faut que les traces DoH soient uniques. Pour un blocage le plus tôt possible, il faut que les premiers paquets de la trace soient uniques.

- Pour étudier l'unicité d'un trafic DoH, on modélise l'ensemble de pages web dans le monde comme une variable W avec un espace échantillon ΩW; et l'ensemble des traces réseau possibles générées par ces sites web comme une variable aléatoire S avec un espace d'échantillonnage Ω S. La trace W d'un site web est une séquence d'entiers non nuls : $(s_i)_{i=1}^{n}, s_i \in \mathbb{Z} \setminus \{0\}, n \in \mathbb{N},$ où si représente la taille (en octets) du i-ème enregistrement TLS dans la trace du trafic. Rappelez-vous que son signe représente la direction - négatif pour les entrants (DNS au client) et positif dans le cas contraire. Nous voulons des traces partielles, c'est-à-dire seuls les *l* premiers enregistrements TLS, que l'on nomme S*l*.
- Nous mesurons l'unicité des traces partielles en utilisant l'entropie conditionnelle H (W | S*l*), définie comme :

$$H(W \mid S_l) = \sum_{\forall o \in \Omega_{S_l}} \Pr[S_l = o] H(W \mid S_l = o)\,.$$

Ici, H (W | S*l* = o) est l'entropie de Shannon de la distribution de la probabilité Pr[W | S*l* = o]. Ceci décrit la probabilité que l'adversaire devine les sites web en W compte tenu de l'observation o. L'entropie conditionnelle H (W | S*l*) mesure à quel point les sites web dans ΩW sont distinguables lorsque l'adversaire a seulement observé *l* enregistrements TLS. Lorsque cette entropie est nulle, les sites sont parfaitement distincts. Par exemple, si le premier paquet de chaque trace DoH avait une taille différente, alors l'entropie H (W | S1) serait 0.

- Nous calculons l'entropie conditionnelle pour différentes tailles de monde | ΩW | et longueurs partielles *l*, en utilisant les traces du jeu de données OW. Chaque point est une moyenne de 3 échantillons de |ΩW | webs. Ces sites web sont sélectionnés uniformément au hasard dans l'ensemble de données. Les nuances représentent l'écart type sur les 3 échantillons.
- On observe les résultats (diminution lente de l'entropie, + la taille du monde augmente, plus les samples ΩW contiennent des sites web plus communs). On observe la même chose avec 5000 pages web (entropie diminue après le 15^{e} paquet).

On va vérifier que la diminution forte du 4^{e} paquet est causée par les enregistrements contenant les recherches DNS.

- Pour cela on va comparer la fréquence d'apparition de la longueur du domaine et du 4e enregistrement sortant. On jette les tailles d'enregistrements TLS correspondant aux messages de contrôle HTTPS2 (ex : taille 33) MAIS on garde la taille 88 même si elle apparaît trop souvent car ça pourrait être causé par les requêtes de nom de domaines de 37 caractères.
- On représente la distribution de fréquence dans un histogramme.
- On observe les résultats -> le 4e paquet contient bien la requête DoH de première partie.

La stratégie de censure DNS-blocking

On va étudier les dégâts collatéraux : combien de sites sont affectés lorsqu'un censeur bloque un site après une analyse de trafic. On fait l'hypothèse que le censeur utilise des techniques standard pour bloquer la connexion : High-confidence blocking, Block on first DoH query.

➔ Block on first DoH query :

- Afin de comprendre les dommages collatéraux encourus par blocage basé sur la longueur de domaine en se basant sur le quatrième paquet, nous comparons la distribution des longueurs de nom de domaine dans le Classement Alexa top 1M avec la distribution de noms de domaine susceptibles d'être censurés dans différents pays.
- Pour le premier, nous prenons le classement mondial selon les classements par pays. Il ne fournit que 500 sites web de premier ordre, ce qui n'est pas suffisant pour nos expériences. Pour certains pays, les listes ne sont pas disponibles.
- Nous prenons alors les listes de tests fournies par Citizen Laboratoires. Ces listes contiennent des domaines que les experts régionaux identifient comme susceptibles d'être censurés. Alors que l'apparence dans ces listes ne garantit pas que les domaines sont réellement censurés, nous pensons qu'ils capturent un comportement de censure potentiel.
- Nous prenons comme exemples cinq pays sujets à la censure: le Turkménistan, l'Iran, l'Arabie saoudite, le Vietnam et la Chine. Comme la liste Alexa ne contient que des domaines, nous extrayons les domaines des URL apparaissant dans les listes de test de Citizen Labs.
- On établit alors un tableau du nombre de domaines dans chaque liste de test de censure et leur présence dans le top 1M d'Alexa.
- On observe les résultats.

DATASETS : https://github.com/spring-epfl/doh_traffic_analysis