Contournez restrictions et censure avec l'IP over DNS !

Vous êtes sur un réseau très restrictif comme un point d'accès Wi-Fi public, une bibliothèque, une école, et vous souhaitez utiliser certains services comme la messagerie instantanée mais le réseau ne laisse passer que le web ? Faites sauter les barrières ! Dans ce cours, vous allez apprendre à faire transiter vos communications rien qu'avec des requêtes DNS. Oui, ces requêtes qui permettent de traduire les adresses comme www.siteduzero.com en adresses compréhensibles par les ordinateurs !

Pour suivre ce tuto, il est recommandé d'avoir quelques notions en réseau, notamment savoir ce qu'est une adresse IP, un protocole et connaitre un minimum le fonctionnement des noms de domaine. Quelques connaissances en Linux seront aussi utiles, notamment le système d'interfaces réseau (ifconfig...). Vous aurez aussi besoin d'un serveur que vous pouvez contrôler (obligatoirement) ainsi que d'un nom de domaine (facultatif).

C'est parti !

Sommaire du chapitre :

L'arbre à DNS

Pour comprendre ce cours, vous devez savoir que les DNS utilisent une structure arborescente et qu'il existe différents types de requêtes. Si ce n'est pas le cas, je vous renvoie à cet excellent tuto de Mathieu Nebra.

Côté client

Quand nous voulons accéder à www.siteduzero.com, que se passe-t-il ? Premièrement, notre machine envoie une requête DNS à un serveur DNS pour savoir à quelle adresse IP s'adresser. On le voit sur cette capture effectuée avec Wireshark :

207	25.232984	192.168.1.53	192.168.1.1	DNS	78 Standard query A www.siteduzero.com
208	25.279558	192.168.1.1	192.168.1.53	DNS	94 Standard query response A 80.248.210.229

On demande l'adresse IP (type A) correspondant à www.siteduzero.com et le serveur nous répond par l'adresse IP en question.

Mais le serveur DNS questionné ne connait pas forcément l'adresse IP demandée ! S'il connait l'adresse de siteduzero.com, il va lui demander l'adresse du sous-domaine www. S'il ne la connait pas, il va demander au serveur qui gère les .com quelle est l'adresse de siteduzero.com pour pouvoir faire la demande précédente.

En tant que client, on ne visualise pas bien comment tout cela se passe. Regardons alors comment le serveur voit les choses.

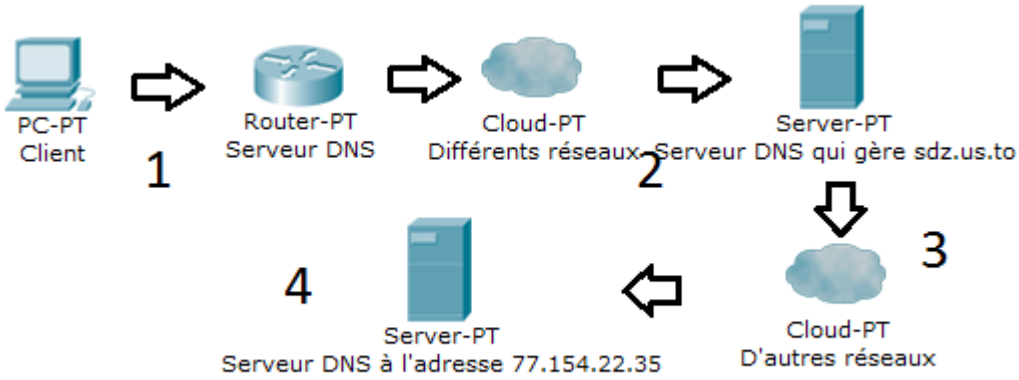
Côté serveur

Voici un exemple concret de configuration d'un domaine.

Sous-domaine	Type	Adresse pointée
a.sdz.us.to	A	77.154.22.35
b.sdz.us.to	NS	a.sdz.us.to

Le type A signifie "ce (sous-)domaine correspond à cette adresse IP". Le type NS signifie "pour tout sous-domaine de ce (sous-)domaine, il faut voir avec tel serveur". En pratique, cela signifie que si on demande à quelle adresse correspond truc.b.sdz.us.to, le serveur DNS va contacter un autre serveur DNS à l'adresse a.sdz.us.to pour avoir la réponse et la transmettre au client qui l'a demandée.

En image, c'est plus clair :



1. Le client demande l'adresse de truc.b.sdz.us.to à son serveur DNS défini dans la configuration de sa machine.
2. Ce serveur DNS transmet la requête au serveur qui gère sdz.us.to, car il connait l'adresse de celui-ci, mais c'est tout.
3. Ce serveur est configuré pour transmettre toute requête concernant les sous-domaines de b.sdz.us.to vers un autre serveur DNS à l'adresse a.sdz.us.to. Il la lui transmet donc.
4. Ce dernier serveur renvoie la réponse au client.

C'est clair jusque là ? Il vaut mieux, car on va rentrer dans le vif du sujet maintenant !

Et si on truquait les requêtes ?

Rien ne nous empêche d'avoir notre propre serveur DNS à l'étape 4 configuré comme on l'entend. Rien ne nous empêche d'envoyer les requêtes qu'on veut à notre serveur. Ainsi, le client et le serveur peuvent utiliser un même "code" pour s'envoyer des messages cachés à travers les requêtes DNS ! Imaginez : le client demande l'adresse de loto.b.sdz.us.to. C'est notre serveur qui va répondre, et on peut parfaitement le configurer de manière qu'il réponde à cette requête particulière par les numéros du dernier tirage du loto ! On aura donc transmis des données d'une nature quelconque *over DNS*.

Et c'est le jackpot ! On peut donc transmettre n'importe quoi à travers des requêtes DNS. On peut transmettre de l'*IP over DNS*. En effet, toute communication que vous effectuez sur Internet utilise le protocole IP (Internet Protocol).
Attention, ce n'est pas non plus la panacée. Cela peut dépanner pour transmettre des données en contournant des restrictions, mais n'espérez pas rejoindre une partie de League of Legends, Need for Speed ou tout autre jeu qui demande à envoyer et recevoir en permanence des données. DNS s'appuie sur UDP qui est un protocole non fiable : on n'est jamais sûr qu'un paquet transmis avec UDP arrive bien à destination. Il est possible qu'un paquet ait à être envoyé plusieurs fois pour qu'il arrive intègre à destination.

Toutefois, comme cela peut quand même être bien utile, nous allons apprendre à mettre en place un tel système.

Configurons notre propre sous-domaine

Vous possédez un nom de domaine ? Parfait ! Sinon, ce n'est pas grave, nous allons en emprunter pour créer des sous-domaines.

Vous possédez un nom de domaine

Si vous possédez un nom de domaine, vous êtes censé savoir le configurer ! 
Dans les zones DNS, créez un sous-domaine de type A que vous appelez comme vous voulez ("a" par exemple, parce que c'est court ) et faites-le pointer vers l'adresse IP du serveur que vous allez utiliser pour l'IP over DNS. C'est ce serveur qui recevra les requêtes DNS "trafiquées" et qui renverra les paquets de manière normale.

Si vous préférez utiliser une adresse IPv6, choisissez le type AAAA. Le type A ne permet de faire pointer que vers une adresse IPv4.

Ensuite, créez un autre sous-domaine de type NS. Donnez-lui un nom court ("b" par exemple), car il va falloir le retaper quand on va mettre en place le serveur. Faites-le pointer vers l'autre sous-domaine précédemment créé mais rajoutez un point au bout. Si votre domaine est **dom.tld**, saisissez "a.dom.tld.". Je ne peux que vous renvoyer au cours de Mathieu Nebra pour de plus amples informations à ce sujet.

Votre zone DNS est prête, vous pouvez passer à la sous-partie suivante. La suite de celle-ci concerne ceux qui n'ont pas de nom de domaine.

Emprunter des noms de domaine

Il y a des gens très gentils qui proposent à tous d'utiliser leur propre nom de domaine pour créer des sous-domaines. Le service FreeDNS est simple d'utilisation et permet à n'importe qui de créer des sous-domaines sur des milliers de noms de domaine différents. Inscrivez-vous donc à ce service gratuit si vous ne possédez pas de nom de domaine à vous !

Une fois inscrit et identifié, allez dans la section "Subdomains" et cliquez sur "Add". Choisissez un domaine qui vous plait dans la liste, sélectionnez le type A et rentrez l'adresse IP de votre serveur dans le champ "Destination" (ou sélectionnez le type AAAA pour pouvoir rentrer l'adresse IPv6). Dans le champ "Subdomain", mettez ce que vous voulez (le tout est d'en trouver un qui ne soit pas déjà pris, essayez des valeurs comme "moi.sdz" ou "truc.sdz"). Cliquez sur "Save!" et hop ! Vous avez un sous-domaine qui pointe vers votre serveur. Par la suite, je considérerai qu'il s'agit de "a.sdz.us.to". Vous adapterez les manipulations en utilisant le vôtre, évidemment.

Créons encore un sous-domaine. Encore une fois, choisissez le domaine qui vous plait, idem pour le sous-domaine. Sélectionnez le type NS et renseignez comme destination le sous-domaine précédemment créé (pas besoin de rajouter un point). Par la suite, je considérerai que ce sous-domaine de type NS est "b.sdz.us.to".

Cela nous donne une configuration qui ressemble à ceci :

Sous-domaine	Type	Adresse pointée
a.sdz.us.to	A	77.154.22.35
b.sdz.us.to	NS	a.sdz.us.to

Voilà, la configuration des DNS est terminée ! Il n'y a plus qu'à mettre le serveur en place !

---

53 protons !

Pour gérer le trafic qui va arriver jusqu'à notre serveur, nous utiliserons le logiciel **iodine**. Il est disponible sous Linux et sous Windows. Toutefois, ne l'ayant jamais utilisé sous Windows, je vous montrerai comment l'utiliser sous Linux. Vous pouvez demander à Google comment faire sur votre système d'exploitation s'il est différent.

Côté serveur

Installez iodine s'il n'est pas déjà présent sur votre serveur (`apt-get install iodine` sous Debian et dérivés). Lorsque vous allez lancer iodine, une nouvelle interface réseau sera créée. Vous devrez spécifier une adresse IP pour cette interface, assurez-vous qu'elle n'entrera pas en conflit avec votre configuration actuelle. Pour cela, relevez les adresses IP de vos cartes réseau avec `ifconfig` ou `ipconfig` sous Windows. Vous choisirez une adresse privée qui n'appartient pas à un réseau déjà configuré.

Si vous n'avez rien compris, vous prendrez l'adresse **192.168.53.1**. Assurez-vous juste que vous n'avez aucune adresse commençant par "192.168.53" dans la configuration de vos cartes réseau.

Lançons donc le serveur iodine (adaptez la commande avec vos propres valeurs).

Code : console

```
iodined -f 192.168.53.1 b.sdz.us.to
```

Le `-f` est facultatif, il sert juste à ne pas lancer le processus en arrière-plan pour qu'on puisse à tout moment l'arrêter en tapant CTRL C.

Vous êtes invité à rentrer un mot de passe, choisissez celui que vous voulez. Il servira juste à initier la connexion quand vous vous connecterez avec votre client.

Votre serveur IP over DNS est maintenant lancé. Il faut maintenant s'y connecter !

Côté client

Depuis votre client, lancez simplement la commande suivante :

Code : console

```
iodine -f -P mot_de_passe b.sdz.us.to
```

La connexion peut être longue à s'établir. Une fois connecté, vous verrez un message comme suit :

Code : console

```
Connection setup complete, transmitting data.
```

La connexion est établie ! Maintenant, testez si cela fonctionne, en lançant une analyse de trames avec Wireshark, par exemple. Si vous ne voyez que des requêtes DNS, c'est que ça fonctionne ! Si cela ne fonctionne pas, ce n'est pas grave, on passe au plan B.

Même si cela fonctionne chez vous, lisez quand même ce qui suit : nous allons établir un tunnel SSH, ce qui permettra de crypter les informations transmises. Cela est fortement recommandé si vous résidez dans un pays où la liberté d'expression est réduite et que vous voulez communiquer des informations délicates.

Le plan B

Je dois vous avouer quelque chose que je ne m'explique pas. La toute première fois que je me suis connecté sur un serveur iodine, tout était transmis *over DNS*, cela fonctionnait. Ça n'a plus jamais refonctionné comme cela depuis. Tout était transmis de manière normale, sans passer par des requêtes DNS. Alors j'ai eu recours à une astuce que je partage avec vous.

Logiquement, nous sommes dans le même réseau que notre serveur iodine avec l'interface dns0. Pourtant, toutes nos communications vers l'extérieur passent par une autre interface (wlan0, eth0, ...). Si on tente de contacter notre serveur via son adresse 192.168.53.1, les communications se feront forcément over DNS. Établissons alors une connexion SSH avec notre serveur.

Code : console

```
ssh 192.168.53.1
```

Nous sommes connectés à notre serveur over DNS. Maintenant, si nous ouvrions un tunnel SSH entre nous et le serveur pour faire transiter nos données ? Comme la connexion SSH fonctionne sur IP over DNS, tout ce qu'on transmet à travers fait de même !

Ouvrons donc cette fois une connexion SSH avec un tunnel dynamique.

Code : console

```
ssh -D 30000 192.168.53.1
```

J'ai choisi le port 30000 mais vous pouvez utiliser celui que vous voulez, tant qu'aucun autre programme sur votre client n'écoute dessus.

Comment fonctionne ce tunnel ? Comment je peux faire passer ma navigation sur le web, mes e-mails, etc. dessus ?

Ce genre de tunnel agit comme un proxy SOCKS. Il vous faut donc configurer les applications

que vous souhaitez utiliser pour qu'ils utilisent un proxy SOCKS à l'adresse 127.0.0.1 (la boucle locale, car c'est sur votre client que SSH écoute pour transmettre les données) et au port 30000 (ou celui que vous avez choisi).

Voilà, il faut juste que la connexion IP over DNS soit suffisamment solide pour pouvoir maintenir la connexion SSH, et roulez jeunesse !

Informations légales

Cette technique permet de contourner des restrictions, elle ne vous soustrait toutefois pas à la loi. Par exemple, ce n'est pas parce que vous allez utiliser un serveur dans un pays où le téléchargement d'œuvres protégées par le droit d'auteur n'est pas interdit, que vous avez le droit de rapatrier les-dites œuvres sur votre machine si vous résidez dans un pays où cela est interdit.

De plus, les hotspots ne filtrant généralement pas les requêtes DNS, il est possible d'utiliser l'IP over DNS pour utiliser sans avoir de compte les SFR Wi-Fi, Belgacom FON et autres points d'accès réservés aux utilisateurs enregistrés. N'en abusez pas, c'est à la limite de la légalité suivant les pays.

Bien entendu, si vous êtes en Iran ou en Chine et que vous souhaitez transmettre des informations sensibles de manière discrète, personne ne vous reprochera d'avoir recours à de telles techniques.

"Tout ça pour ça" ! Il est vrai que l'utilité de l'IP over DNS est relative. Quand on peut, on préfèrera utiliser des tunnels SSH : c'est beaucoup plus facile à mettre en place ! Mais il arrive que l'on tombe sur des réseaux qui ne laissent presque rien passer, dans ce genre de cas, une telle technique peut s'avérer utile.

L'icone de ce cours a été réalisée par p@lex et est sous licence