Persée et la Gorgone : attaques par déni de service utilisant le DNS, et les contre-mesures

Stéphane Bortzmeyer
AFNIC
Immeuble International
78181 Saint-Quentin-en-Yvelines
bortzmeyer(à)nic.fr

Résumé

Les années récentes ont vu une brusque augmentation du nombre d'attaques par déni de service (DoS) utilisant le DNS. Les attaques ainsi menées dépassent désormais couramment les 20 Gb/s cumulés. Ces attaques utilisent en général la réflexion (un serveur tiers relaie le paquet IP) et l'amplification (la réponse est plus grande que la question, aidant l'attaquant à atteindre son objectif).

Il existe plusieurs contre-mesures possibles contre ces attaques : militer pour le déploiement effectif de BCP 38 (c'est-à-dire que les FAI empêchent l'usurpation d'adresses IP par leurs usagers), modifier le protocole DNS et imposer une limitation de trafic, refusant de répondre au-delà d'un certain rythme.

Mots clefs

DNS, sécurité, DoS, dDoS, rate-limiting, BIND, NSD, Netfilter

1 Introduction

Les attaques par déni de service sont une des principales plaies de l'Internet. Elles sont faciles et relativement peu coûteuses à monter, et il est très difficile de se défendre, sauf à dépenser des sommes considérables et, souvent, à brider des activités légitimes.

La variété « attaque par déni de service utilisant le DNS [1] » est très ancienne (la première description date de 2006 [3]). Mais elle était restée relativement rare, jusqu'à la vague de 2011-2012, qui a été mise sur le devant de la scène par un fil de discussion sur la liste dns-operations gérée par l'OARC [2] : « *Abnormal activity from chinanet ?* » en décembre 2011.

Depuis, de nombreuses réunions de l'OARC ou d'organisations similaires ont été consacrées à ce problème.

2 Description de l'attaque

2.1 Principe

Le principe d'une attaque par réflexion est simple : l'attaquant utilise un tiers, le réflecteur. Il écrit au réflecteur en *mentant* sur son adresse IP. Le paquet arrivant au réflecteur aura donc une adresse IP source qui est celle de la victime. Le réflecteur va donc répondre... et enverra la réponse à la victime. C'est en l'honneur du héros mythologique qui tua la gorgone Méduse, grâce à une attaque par réflexion (son bouclier poli renvoya vers Méduse le regard pétrifiant de celle-ci), que le titre de cet exposé et le logiciel d'attaque présenté plus loin font référence à Persée.

La réflexion seule n'est pas très utile pour l'attaquant. Elle ne le dissimule pas vraiment (puisqu'il peut tricher sur son adresse). Elle brouille un peu les pistes (le trafic entre par un autre chemin que les paquets qui seraient venus directement de l'attaquant). Ceci dit, elle est surtout intéressante quand on la couple avec l'amplification. Ce phénomène se produit lorsque la réponse est plus grande que la question, ce qui est fréquent avec le DNS [3] L'attaquant obtient ainsi plus de Mb/s

1. Les attaques contre le service DNS, visant à perturber ce service, sont une autre question [1] [2].
2. https://www.dns-oarc.net/
3. D'autres protocoles ont cette propriété, comme SNMP [4] mais aussi Call of Duty [5].

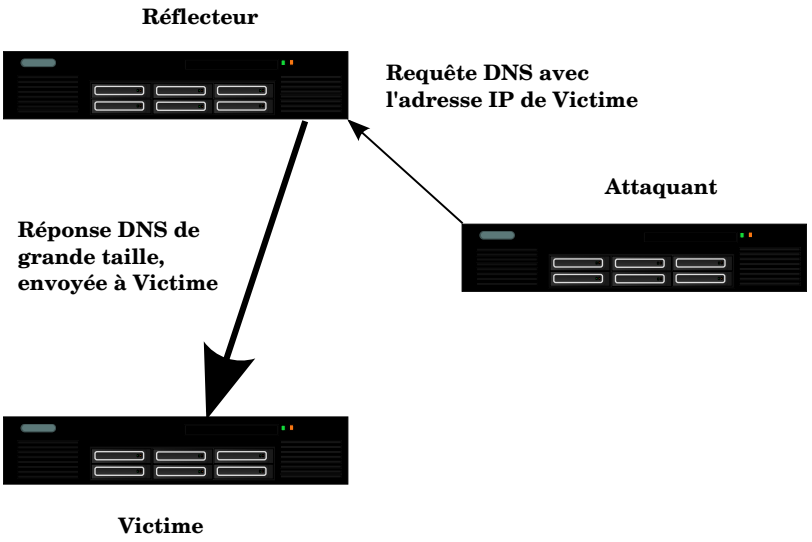

Figure 1 - Schéma de l'attaque par réflexion

qu'il n'en avait dépensé. Voici un exemple de requête et de réponse DNS vu avec tcpdump :

```
15:05:49.973266 length 85 proto UDP (17), length 71
15:05:49.973468 length 1996 proto UDP (17), length 1982
```

On voit que la réponse était 23 fois plus grande que la requête. Cette attaque est schématisée sur la figure 1.

On notera que cette attaque est possible car IP permet de tricher sur l'adresse IP source (d'où les contre-mesures présentées en section 5.1) et parce que UDP, le protocole de transport principal du DNS, n'a pas de notion de connexion.

Une des conséquences de l'usurpation d'adresses est que le réflecteur peut se tromper sur l'origine de l'attaque. Voyant les adresses IP source, il risque de prendre la victime pour l'agresseur…

2.2 Quel réflecteur ?

Quels serveurs DNS utiliser comme réflecteurs ? L'attaquant par réflexion+amplification a deux possibilités :

1. Il peut se servir de résolveurs ouverts, des serveurs DNS récursifs qui, contrairement aux bonnes pratiques, répondent à tout l'Internet,
2. Il peut se servir de serveurs faisant autorité, par exemple ceux d'un gros TLD [4].

Avec des résolveurs ouverts, l'attaquant doit :

1. Mettre un enregistrement DNS de grande taille (le type est quelconque, c'est souvent un TXT mais cela peut aussi être un A) sur un nom qu'il contrôle, comme `mechant.attaque.quelquepart.example`. Le contenu est indifférent et on trouve des textes comme SONYMOUS, XXXXX…
2. Envoyer des requêtes pour ce nom et ce type à de nombreux résolveurs ouverts (il en existe des millions, voir la section 5.2).

Les résolveurs vont alors répondre et bombarder la victime.

Le seul risque pour l'attaquant est qu'il soit retrouvé via l'enregistrement du domaine. Ce risque est faible en pratique. La plupart des résolveurs ouverts sont des machines lentes et mal connectées mais elles sont très nombreuses. N'étant la plupart du temps pas gérées du tout, l'attaquant pourra travailler à son aise.

Dans le second cas (utilisation de serveurs faisant autorité), l'attaquant va :

4. *Top-Level Domain* ou domaine de premier niveau.

1. Trouver un domaine ayant une bonne amplification et de bons serveurs. Un domaine signé avec DNSSEC fournit une meilleure amplification. Prenons le TLD `example` pour la suite.
2. Envoyer des requêtes, par exemple de type ANY [5] aux serveurs faisant autorité pour `example` en usurpant l'adresse IP.

Ces serveurs bombardent alors la victime.

Les serveurs faisant autorité sont moins nombreux que les résolveurs ouverts mais souvent plus puissants (bonnes machines, et bien connectées). Par contre, ils sont plus fréquemment gérés, supervisés et donc relativement bien protégés.

Maintenant, en pratique, quelles sont les attaques les plus courantes ? Faut-il concentrer ses efforts sur les attaques DNS ou bien sur celles faites avec SNMP ? Et, si on se focalise sur le DNS, le problème le plus grave est-il celui des résolveurs ouverts ou bien celui des serveurs faisant autorité, notamment avec DNSSEC ? Le problème est qu'on ne sait pas. Comme le notait Florent Chabaud [1], contrairement à ce qui se passe pour les accidents d'avions, les attaques par déni de service ne font pas l'objet d'une enquête indépendante, avec résultats publiés. Les décideurs sont donc dans le noir. On voit parfois sur des forums des affirmations fortes sur le danger de telle ou telle attaque mais elles ne s'appuient pas sur des faits.

À noter que les attaques sont rarement menées par une machine isolée, mais plutôt par un *botnet*, un réseau de machines piratées et qui obéissent désormais à un maître qui les loue [6] pour envoyer du spam, faire des attaques par déni de service, etc.

3 L'attaque dans le monde réel

On l'a vu plus haut, peu d'attaques sont documentées publiquement [7]. Félicitons donc Lars Noring (du FAI norvégien PowerTech) qui, à l'atelier OARC de 2012 à Teddington, a présenté « *A deep dive into heaps of Spoofed DNS Traffic* », racontant l'attaque dont son employeur avait été victime. Des paquets DNS étaient envoyés au résolveur du FAI, avec une adresse IP usurpée, appartenant au réseau local du FAI, et demandant des noms qui appartenaient à ChinaNetCenter, un fournisseur de CDN en Chine.

Le dialogue avec les Chinois a été très difficile (ils prétendaient ne pas comprendre) mais l'attaque a fini par cesser.

Suite à cette mésaventure, l'auteur a voulu savoir si d'autres opérateurs en Norvège avaient la même faiblesse. Il a envoyé à leurs résolveurs des requêtes avec une adresse IP source usurpée, pour un domaine unique qu'il contrôlait. tcpdump sur le serveur faisant autorité permettait de voir si la requête mensongère avait été reçue. Résultat, 50 % des FAI norvégiens ne filtraient pas en entrée leurs propres adresses IP…

Même transparence de la part de Jérémy Martin (First Heberg) à la réunion FRnog 21 de 2013 lorsqu'il a décrit en détail l'attaque dont avait été victime son employeur (les réflecteurs étaient 80 % de résolveurs ouverts et 20 % de serveurs faisant autorité).

Autre exemple, vue sur les serveurs de l'AFNIC (figure 2), une attaque réelle, affichée par le logiciel de supervision DSC. Les requêtes de type ANY sont normalement en nombre infime (ANY ne sert guère qu'au déboguage). Le trafic ANY intense est donc presque toujours une attaque.

4 Tester les contre-mesures

Pour étudier les attaques, on utilise le logiciel SOP [6]. Il permet d'envoyer des requêtes DNS répétées à un réflecteur, en trichant sur l'adresse source. Ses principales fonctions :

- Fonctionne en IPv4 et IPv6 [7],
- Permet de choisir l'adresse source au hasard dans un préfixe donné, pour chaque requête (cela perturbe certains limiteurs de trafic),

5. Requête qui demande l'envoi de tous les enregistrements disponibles.
6. *Shield of Perseus*. Il n'est pas disponible en France, en raison de l'article 323-3-1 du Code Pénal qui interdit « d'importer, *de détenir*, d'offrir, de céder ou de mettre à disposition un équipement, un instrument, un programme informatique ou toute donnée conçus ou spécialement adaptés pour commettre une ou plusieurs des infractions prévues par les articles 323-1 à 323-3 », sauf « motif légitime ». On peut penser que la recherche en sécurité fait partie des motifs légitimes.
7. Même si aucune attaque utilisant IPv6 n'a encore été rencontrée dans la nature.

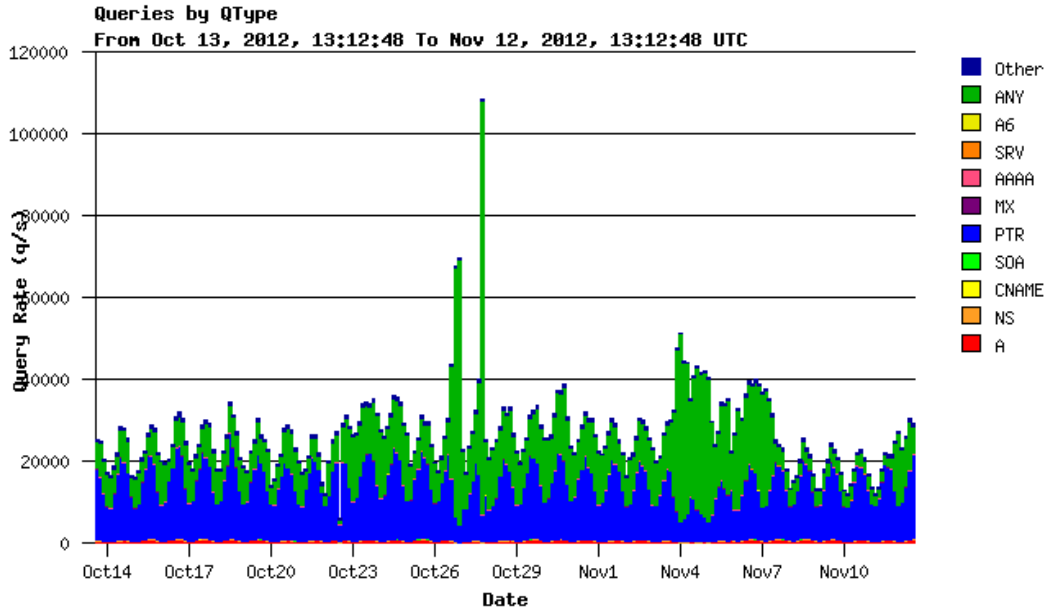

Figure 2 - Attaque réelle, requêtes ANY, sur un serveur faisant autorité

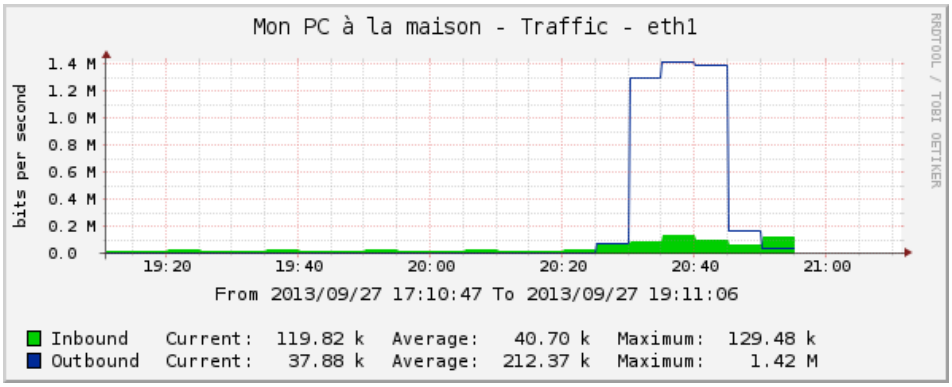

Figure 3 - Attaque simple, sans contre-mesure, en laboratoire (avec SOP), vue du réflecteur

- Permet d'utiliser EDNS et DNSSEC,
- Permet de choisir un certain nombre de champs, comme le *query type* de la requête (A, MX, ANY…).

D'abord, regardons une attaque menée avec SOP. La ligne de commande était :

```
sop --qnames example --source 192.168.2.4 --delay 10000 --count 90000 \
    --size 4096  --dport 9053 192.168.2.1
```

Ce qui fait un rythme de cent paquets par seconde (10 000 µs entre deux paquets) pendant un quart d'heure. Le type des requêtes est la valeur par défaut, ANY. Le réflecteur est `192.168.2.1`, et la victime `192.168.2.4`. La commande dig équivalente est :

```
dig +dnssec +bufsize=4096 -p 9053 @192.168.2.1 ANY example
```

Et elle produit ici une réponse de 1 716 octets. Sur le réflecteur, une machine NSD sans contre-mesures déployées, on voit bien (figure 3) la différence entre le trafic entrant, qui plafonne à 130 kb/s et le trafic sortant, plus de dix fois plus important. Naturellement, lors d'une vraie attaque, on observe des débits bien plus élevés. Celle-ci n'était qu'une expérience de laboratoire.

5 Changer l'Internet

Comme toujours en sécurité (ou en santé publique), la question n'est pas trouver *la* bonne technique. Il n'existe pas de panacée. Il faut fermer les résolveurs ouverts *et* il faut sécuriser les serveurs faisant autorité. Si on étudie les secteurs qui ont ce genre de problème de déploiement depuis des siècles (lutte contre la délinquance, santé publique), on voit qu'il faut attaquer sur plusieurs fronts à la fois.

5.1 Empêcher l'usurpation d'adresse

Comme le point de départ d'une attaque par réflexion est l'usurpation d'adresse IP, si on supprime cette possibilité d'usurpation, on résoud le problème. Le jeu de documents connu sous le nom de « BCP [8] 38 », qui comprend actuellement deux RFC [8] [9], demande exactement cela : que les FAI ne permettent pas à leurs clients d'envoyer des paquets avec une adresse IP source usurpée.

Aujourd'hui, le moins qu'on puisse dire est que le déploiement de BCP 38 n'est pas généralisé [9]. Le fait que la majorité des réseaux ne permette pas la triche ne suffit pas, il faudrait qu'une *grande* majorité le fasse. Autrement, les attaquants iront simplement sur les réseaux les plus laxistes. Les facteurs économiques jouent contre BCP 38 : le déployer, pour un FAI, c'est dépenser de l'argent pour gêner ses utilisateurs et protéger ses concurrents. Les statistiques du *Spoofer project* [10] ne permettent pas d'être optimiste. On est là dans un cas typique où la somme des intérêts individuels ne correspond pas à l'intérêt collectif.

Certains experts estiment que l'essentiel des efforts de lutte contre les attaques par réflexion devrait être consacré à promouvoir et illustrer BCP 38. D'autres pensent que c'est une cause perdue.

Un projet plus général, SAVI [11], vise à généraliser BCP 38 mais il en est à ses tous débuts.

5.2 Fermer les résolveurs ouverts

Un autre cas où il faut une campagne longue et patiente de conviction, pour faire bouger un grand nombre d'acteurs, est celui de la lutte contre les résolveurs ouverts. Ils facilitent beaucoup l'attaque par réflexion+amplification puisqu'ils permettent à l'attaquant de choisir exactement la taille et le type des paquets, qu'il va héberger sur son serveur, avant d'interroger ces résolveurs ouverts.

Le danger de ces résolveurs est connu depuis longtemps, l'AFNIC avait alerté à ce sujet dès 2006 [10] et le RFC qui recommande officiellement leur fermeture [11] date de 2008. Comme le montrent les études du *Open Resolver Project* [12], ce n'est pas gagné. Les résolveurs ouverts sont nombreux, et en général non gérés (machines installées en vitesse et pas administrées par la suite). Prévenir un par un leurs responsables, devoir tout leur expliquer, et sans garantie qu'ils agissent, est une tâche épuisante.

6 Changer le DNS

Une autre possibilité serait de changer le DNS, pour rendre les usurpations d'adresse IP source plus difficiles. Pour cela, on peut par exemple utiliser TCP plutôt que UDP. Cela résoud complètement la question [13] mais cela soulève deux problèmes :

- Les performances sont moins bonnes en TCP [13], même si, aujourd'hui, des serveurs affichant un grand nombre de connexions TCP par seconde sont bien plus courants, grâce au Web.
- Hélas, bien des serveurs DNS ne répondent pas en TCP, en général en raison d'un pare-feu mal configuré situé devant le serveur. Le remplacement d'UDP par TCP nécessiterait donc une campagne de sensibilisation, comme les solutions de la section précédente.

8. *Best Current Practices*
9. hping a une option `--spoof` si vous voulez tester depuis le réseau de votre FAI…
10. http://spoofer.cmand.org/
11. *Source Address Validation Improvement*
12. http://openresolverproject.org/
13. TCP n'est pas vulnérable aux usurpations d'adresse menées en aveugle [12].

Autre changement possible du DNS, plus radical : modifier le protocole pour avoir un système de *cookies*. Ce sont des nombres imprévisibles, générés par le serveur, et que le client doit transmettre dans ses futures requêtes, pour prouver qu'il a bien reçu le *cookie* [14] Comme tout changement de protocole, on peut s'inquiéter des chances réelles de déploiement, dans un Internet où il est très difficile de faire bouger les choses.

Naturellement, on n'est pas obligé de déployer ces techniques seules. On peut les combiner avec la limitation du trafic, présentée dans la section suivante, par exemple en ne limitant que les clients qui ne présentent pas de *cookies*. Cela fournirait une incitation au déploiement de logiciel plus récent, gérant ces *cookies*.

7 Limiter le trafic

7.1 Diverses méthodes

Les attaques utilisant les serveurs faisant autorité s'appuient souvent sur des requêtes de type ANY, pour obtenir le maximum d'amplification. Il a parfois été proposé de refuser de répondre à ces requêtes. Neustar avait choisi cette voie pour `us` et `biz` mais a fait machine arrière depuis.

L'approche la plus courante à l'heure actuelle, chez les opérateurs de serveurs DNS faisant autorité, est de limiter le trafic sortant, pour empêcher que leurs machines servent d'amplificateurs. Ce n'est pas que les autres méthodes soient mauvaises, mais leur déploiement dépend d'autres acteurs. Première idée, diminuer la taille maximale des réponses, par exemple à 1 460 octets [15]. Cette technique ne semble avoir été déployée, pour l'instant, que par Verisign, sur les serveurs de `com` et `net`. Elle diminue le facteur d'amplification mais celle-ci demeure quand même non négligeable.

On peut déjà utiliser les services de base du système d'exploitation pour limiter le trafic, et cela a été fait au début, avant que les logiciels serveurs de noms ne soient dotés de cette fonction. Ainsi, le serveur racine F, qui tourne sur FreeBSD, avait dans sa configuration :

```
add     pipe 1          udp     from any to any 53 in
pipe 1  config  mask src-ip 0xffffffff buckets 1024 bw 400Kbit/s queue 3
add     pipe 2          tcp     from any to any 53 in
pipe 2  config  mask src-ip 0xffffffff buckets 1024 bw 100Kbit/s queue 3
```

Sur Linux, le système de pare-feu se nomme Netfilter [16]. Netfilter est modulaire et deux modules nous intéressent particulièrement ici :

- hashlimit permet de faire de la limitation de trafic par préfixe IP [17],
- m32 permet de tester des champs quelconques du paquet entrant, par exemple pour ne filtrer que les requêtes concernant un domaine donné.

m32 utilise un langage plutôt compliqué [15]. Pour ne pas avoir à écrire directement dans ce langage, on passe par un script en Python [18] qui prend en argument les caractéristiques de la requête (nom demandé, type demandé) et génère le code m32.

On lance SOP avec les mêmes options qu'en section 4. Mais cette fois, on a installé une limitation du trafic « ANY example » avec Netfilter (figure 4). En regardant le trafic sur la victime (figure 5), on voit qu'il a en effet été divisé par cinq, comme on s'y attendait (SOP attaque à 100 p/s, le limiteur coupe à 20 p/s) :

À noter que le test aurait aussi pu être fait avec queryperf [19] ou bien avec une commande comme :

```
% repeat 100 dig +short +tries=1 +time=1 @SERVER ANY example
```

14. L'attaquant qui usurpe une adresse IP source travaille en aveugle, puisqu'il ne peut pas recevoir les réponses.
15. Avec NSD, c'est la directive `ipvX-edns-size: 1460`, avec BIND, c'est `max-udp-size 1460`.
16. Mais est parfois désigné par le nom de la commande de configuration, iptables.
17. Plusieurs autres modules de limitation de trafic existent mais ont des limites sérieuses [14].
18. http://www.bortzmeyer.org/files/generate-netfilter-u32-dns-rule.py
19. Distribué dans le répertoire `contrib/` de BIND.

```
iptables -A RATELIMITER -m hashlimit \
        --hashlimit-name DNS --hashlimit-above 20/second \
        --hashlimit-mode srcip
        --hashlimit-burst 100 --hashlimit-srcmask 28 -j DROP

iptables -A INPUT -p udp --dport 9053 -m u32 \
     --u32 $(python generate-netfilter-u32-dns-rule.py --qname example --qtype ANY) \
     -j RATELIMITER
```

Figure 4 - Commandes Netfilter pour limiter le trafic

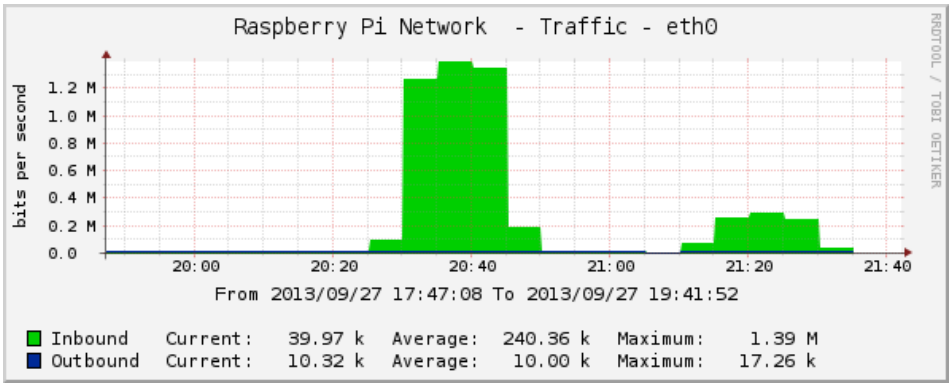

Figure 5 - Attaque simple, Netfilter - figure 4 - limite le trafic ANY (trafic vers la victime, sans et avec Netfilter sur le réflecteur)

7.2 RRL

Une des techniques les plus déployées, en raison de son efficacité, est la RRL (*Response Rate Limiting*). Le terme de RRL est en général utilisé pour désigner les solutions de limitation de trafic incluses dans le serveur DNS, et reposant sur l'idée qu'un client n'est pas censé demander la même chose cent fois par seconde. L'inconvénient principal de la solution Netfilter présentée en section 7.1 est qu'il faut l'installer à la main lorsqu'on détecte une attaque, puisque la règle est spécifique d'un nom de domaine, et d'un type d'enregistrement. Au contraire, la RRL [16] se configure automatiquement : dès que le rythme des réponses identiques dépasse un certain seuil, la RRL se déclenche et limite le trafic.

RRL est disponible dans NSD depuis la version 3.2.15 [17], dans Knot depuis la 1.2.0 et dans BIND depuis la 9.9.4. Il peut être nécessaire de compiler le serveur avec une option spécifique [20].

L'inconvénient classique de la limitation de trafic est qu'on arrête aussi les requêtes légitimes. Si la victime de l'attaque par déni de service est également un client normal (un résolveur interrogeant le serveur faisant autorité), cette contre-mesure le gênera. En pratique, la victime n'est en général pas un client DNS et ce n'est donc pas un problème. Pour les cas où la victime est un client DNS, les mises en œuvre du concept de RRL utilisent SLIP. Son principe est de répondre de temps en temps aux requêtes, mais avec le bit TC [21] mis à 1. Une telle réponse est plus petite que la question (et ne contribue donc pas à l'amplification). Le client DNS qui la recevra réessaiera sa question en TCP et pourra donc recevoir sa réponse. On peut configurer quel pourcentage des requêtes pour lesquelles le serveur ne veut pas répondre déclencheront l'envoi de ce paquet tronqué.

Un exemple de configuration de la RRL dans NSD, le premier serveur DNS qui a été livré avec cette fonction :

```
server:
       ...
       # 20 identical queries per second, maximum
       rrl-ratelimit: 20
```

20. `--enable-ratelimit` pour NSD, et `--enable-rrl` pour BIND
21. Qui indique une réponse tronquée

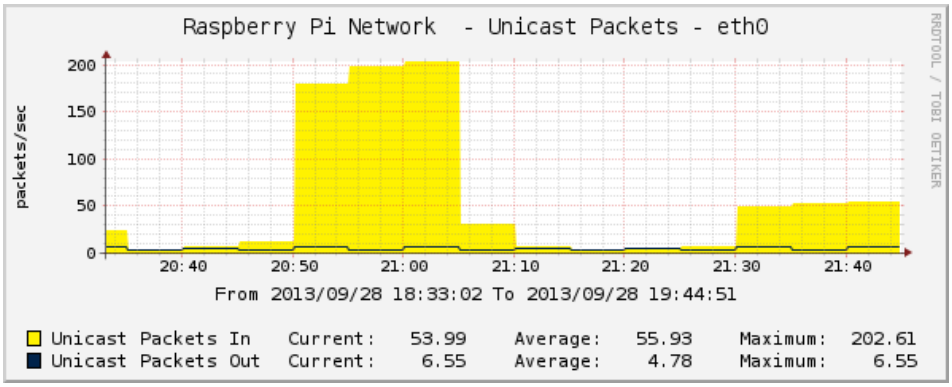

Figure 6 - Le trafic sur la victime, sans et avec RRL

La figure 6 montre le nombre de paquets par seconde reçus par la victime (et non pas le nombre de bits par seconde comme sur les graphes précédents). L'attaque est la même. Le premier pic correspond à un réflecteur NSD 3.2.16 sans RRL. Le second au même NSD avec RRL à 20 réponses par seconde. L'attaquant envoie 100 p/s et la victime en reçoit 200 p/s, car la réponse, plus grosse que la MTU d'Ethernet, a été fragmentée en deux paquets. Avec la RRL, on ne reçoit que 50 p/s. Il y a les 20 réponses par seconde autorisées (40 paquets avec la fragmentation) et le SLIP pour les réponses refusées. En nombre de paquets, l'effet de RRL peut donc sembler faible, mais il est plus important en nombre d'octets (les paquets SLIP sont très petits).

NSD journalise le déclenchement de la limitation de trafic [22] :

```
[1380484720] nsd[31551]: info: ratelimit block example. \
     type any target 192.168.2.0/24 query 192.168.2.4 TYPE255
```

Et pour BIND ? La RRL se configure ainsi [18] :

```
options {
        ...
        rate-limit {
          responses-per-second 20;
        };
};
```

Et le résultat est journalisé :

```
29-Sep-2013 15:52:47.929 client 192.168.2.4#5353 (example): \
    rate limit drop response to 192.168.2.0/24 for example IN ANY  (05a919a3)
```

Et visible graphiquement sur la figure 7.

Knot permet aussi le RRL :

```
system {
  ...
  rate-limit 20;
}
```

Avec ses paramètres par défaut [23], Knot est nettement plus modéré quand il limite le trafic.

Une évaluation tierce de l'efficacité de la RRL dans BIND est décrite dans une excellente étude pratique [19]. Cette étude note qu'une des faiblesses potentielles de la RRL est dans le cas où le serveur DNS réflecteur fait autorité pour beaucoup

22. Le type 255 est ANY, le type envoyé par l'attaquant.
23. Paramètre « SLIP » mis à 1

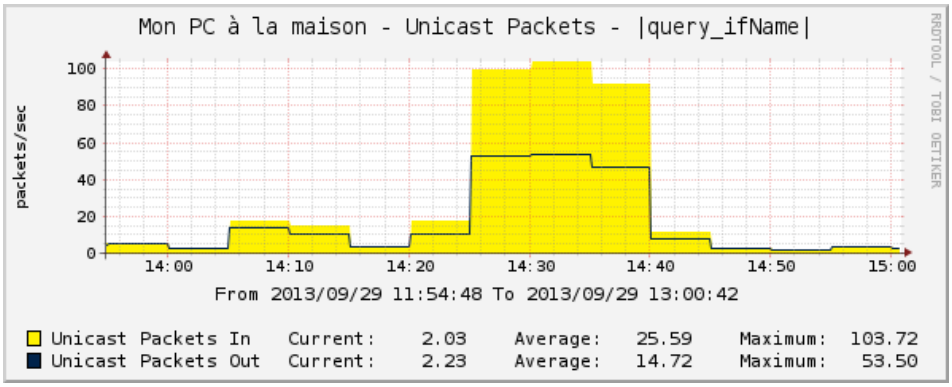

Figure 7 - Le trafic sur un réflecteur BIND, avec RRL

de domaines [24]. Si l'attaquant fait varier le domaine demandé [25], il peut contourner la RRL. L'option `--qnames` de SOP peut prendre comme valeur un nom de fichier, contenant la liste de domaines hébergés par le serveur, afin de réaliser cette attaque.

Une autre limite de la RRL est qu'elle ne s'applique pas forcément bien aux serveurs récursifs qui peuvent, eux aussi, servir de réflecteurs. Un client DNS d'un serveur faisant autorité n'est pas censé répéter la même question des dizaines de fois par seconde, parce qu'il est censé avoir un cache, une mémoire des réponses, pour la durée du TTL. Au contraire, un client DNS d'un serveur récursif a le droit de ne pas avoir de mémoire, et il peut donc émettre légitimement de nombreuses requêtes. La question de l'adaptabilité de la RRL aux serveurs récursifs est encore ouverte aujourd'hui.

Enfin, comme toute technique de sécurité, la RRL apporte de nouveaux problèmes. C'est ainsi que son utilisation peut favoriser un autre type d'attaques, les attaques par empoisonnement [20] [21]. Dans ces attaques, l'attaquant va tenter de répondre aux requêtes d'un client *avant* le serveur légitime. Il y a donc une course entre le serveur légitime et l'attaquant. Si le serveur légitime utilise RRL, il renonce délibérement à répondre à certaines requêtes, se retirant de lui-même de la course. Une solution possible est de toujours répondre, ne serait-ce que par un paquet SLIP [26]. Mais le problème ne semble pas aujourd'hui très présent en pratique.

Dans le futur, on verra sans doute de nouvelles attaques. Certaines sont prévues mais d'autres pas. Par exemple, il peut être tentant d'essayer de tromper le limiteur de trafic en essayant plein de noms au hasard (la réponse NXDOMAIN, dans un domaine signé, peut être assez grosse quoique pas autant que lorsque le nom existe). SOP met en œuvre cette technique avec l'option `--qnames random` :

```
sop --qnames random --suffix example ...
```

Mais le RRL de NSD ne s'y laisse pas prendre. Il considère les réponses NXDOMAIN comme une seule réponse :

```
[1380567519] nsd[5195]: info: ratelimit block example. \
   type nxdomain target 192.168.2.0/24 query 192.168.2.4 TYPE255
```

Et on voit sur le graphique une amplification en octets qui est négative (moins de données émises par le réflecteur qu'envoyées par l'attaquant).

8 Conclusion

En 2013, le nombre d'attaques utilisant le DNS semble avoir baissé, en tout cas sur les serveurs faisant autorité ayant déployé des contre-mesures. Peut-être ont-elles simplement migré vers des serveurs moins protégés ? De toute façon, il est aujourd'hui nécessaire de mettre en œuvre des contre-mesures. Outre la sensibilisation à BCP 38 et à la fermeture des résolveurs ouverts, il est nécessaire d'installer et de configurer un mécanisme de limitation de trafic sur les serveurs faisant autorité.

24. C'est le cas d'un hébergeur DNS, où des serveurs servant des centaines de milliers de domaines sont courants.
25. Cela s'est déjà produit dans certaines attaques.
26. Cela se fait avec `rrl-slip: 1` sur NSD et `slip 1` sur BIND.

Bibliographie

[1] Stéphane Bortzmeyer. Journée du conseil scientifique AFNIC sous le signe de la résilience, 2012. http://www.afnic.fr/fr/l-afnic-en-bref/actualites/actualites-generales/6132/show/succes-pour-la-journee-du-conseil-scientifique-sous-le-signe-de-la-resilience-1.html.

[2] Stéphane Bortzmeyer. La journée du 31 mars sur les serveurs racine du DNS, 2012. http://www.bortzmeyer.org/racine-dns-opblackout.html.

[3] Randy Vaughn et Gadi Evron. DNS amplification attacks, 1996. http://www.isotf.org/news/DNS-Amplification-Attacks.pdf.

[4] BITAG. SNMP DDoS attacks, 2012. http://www.bitag.org/report-snmp-ddos-attacks.php.

[5] Foxxz. Call of duty 5 : Stuxnet, 2010. http://www.ar15.com/archive/topic.html?b=1&f=5&t=1131291.

[6] Gunter Ollmann. Want to rent an 80-120k DDoS botnet ?, 2009. https://blog.damballa.com/archives/330.

[7] Kevin Shortt. DNS ANY request cannon - need more packets, 2012. https://isc.sans.edu/diary/DNS+ANY+Request+Cannon+-+Need+More+Packets/13261.

[8] P. Ferguson et D. Senie. RFC 2827 : Network ingress filtering : Defeating denial of service attacks which employ ip source address spoofing, 2000. http://www.rfc-editor.org/rfc/rfc2827.txt.

[9] F. Baker et P. Savola. RFC 3704 : Ingress filtering for multihomed networks, 2006. http://www.rfc-editor.org/rfc/rfc3704.txt.

[10] AFNIC. Avertissement concernant les serveurs DNS récursifs ouverts, 2006. https://www.afnic.fr/fr/l-afnic-en-bref/actualites/actualites-generales/2692/show/avertissement-concernant-les-serveurs-dns-recursifs-ouverts.html.

[11] Joao Damas et Federico Neves. RFC 5358 : Preventing use of recursive nameservers in reflector attacks, 2008. http://www.rfc-editor.org/rfc/rfc5358.txt.

[12] M. Dalal A. Ramaiah, R. Stewart. RFC 5961 : Improving TCP's robustness to blind in-window attacks, 2010. http://www.rfc-editor.org/rfc/rfc5961.txt.

[13] Francis Dupont. a TCP DNS performance test tool, 2013. https://indico.dns-oarc.net/indico/contributionDisplay.py?contribId=21&confId=0.

[14] Stéphane Bortzmeyer. Diminuer une attaque DoS avec Netfilter sur Linux, 2012. http://www.bortzmeyer.org/rate-limiting-dos.html.

[15] Stéphane Bortzmeyer. On ne peut pas analyser tous les protocoles avec Netfilter, 2012. http://www.bortzmeyer.org/dns-netfilter-u32.html.

[16] Paul Vixie et Vernon Schryver. DNS response rate limiting (DNS RRL), 2012. http://ss.vix.com/~vixie/isc-tn-2012-1.txt.

[17] Wouter Wijngaards. DNS response rate limiting as implemented in NSD, 2012. http://www.nlnetlabs.nl/blog/2012/10/11/nsd-ratelimit/.

[18] ISC. A quick introduction to response rate limiting, 2013. https://kb.isc.org/article/AA-01000/0.

[19] Matthijs Mekking Thijs Rozekrans, Javy de Koning. Defending against DNS reflection amplification attacks, 2013. http://www.nlnetlabs.nl/downloads/publications/report-rrl-dekoning-rozekrans.pdf.

[20] CERTA. Vulnérabilité dans DNS response rate limiting, 2013. http://www.certa.ssi.gouv.fr/site/CERTA-2013-AVI-506/index.html.

[21] ANSSI. Démonstration d'un détournement possible de technologies anti-déni de service distribué (DDoS), 2013. http://www.ssi.gouv.fr/fr/anssi/publications/publications-scientifiques/articles-de-conferences/demonstration-d-un-detournement-possible-de-technologies-anti-deni-de-service.html.

[22] Roland Van Rijswijk. DNSSEC : what every sysadmin should know to keep things working, 2012. https://www.usenix.org/sites/default/files/conference/protected-files/vanrisjwik_lisa12_slides.pdf.

[23] Matthew Prince. How to launch a 65gbps DDoS, and how to stop one, 2013. http://blog.cloudflare.com/65gbps-ddos-no-problem.

[24] Verisign. Anatomy of recent DNS reflector attacks from the victim and reflector point of view, 2006. http://www.verisign.com/static/037903.pdf.