RFC 8612 : DDoS Open Threat Signaling (DOTS) Requirements

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 22 août 2019

Date de publication du RFC : Mai 2019

https://www.bortzmeyer.org/8612.html

Les attaques par déni de service, et notamment les dDoS ("*distributed Denial of Service*"), sont une des principales plaies de l'Internet. Le projet DOTS ("*DDoS Open Threat Signaling*") à l'IETF vise à développer un mécanisme de **signalisation** permettant à des acteurs de la lutte anti-dDoS d'échanger des informations et de se coordonner, même lorsque l'attaque fait rage. Par exemple, un mécanisme DOTS permettra à un client d'un service de traitement des attaques de demander à son fournisseur d'activer le filtrage anti-dDoS. Ce RFC est le premier du projet : il décrit le cahier des charges.

Ces attaques par déni de service (décrites dans le RFC 4732 [1]) peuvent être utilisées à des fins financières (racket), lors d'affrontements inter-étatiques (comme dans le cas estonien souvent cité), à des fins de censure contre des opposants politiques. Le risque est particulièrement élevé pour les « petits ». En effet, beaucoup d'attaques par déni de service reposent sur la taille : par exemple, l'attaquant envoie tellement d'octets qu'il sature la ou les connexions Internet de sa victime. La seule solution est alors de louer un tuyau plus gros, ce qui n'est pas toujours financièrement possible. Les attaques dDoS favorisent donc les plus riches. Aujourd'hui, par exemple, un petit hébergeur Web a le plus grand mal à faire face à d'éventuelles attaques, ce qui rend difficile l'hébergement associatif et/ou décentralisé. Les attaques par déni de service ont donc des conséquences bien au-delà des quelques heures d'indisponibilité du service : elles encouragent la centralisation des services, puisqu'il faut être gros pour encaisser le choc. C'est ainsi qu'aujourd'hui beaucoup d'organisations sont chez Cloudflare, dépendant de cette société privée étatsunienne pour leur « protection ». On est dans l'équivalent moderne de la relation féodale au Moyen-Âge : le paysan seul, ou même le village, est trop vulnérable, il doit chercher la protection d'un seigneur, en échange de sa soumission.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4732.txt

Il est très difficile de se protéger contre les attaques par déni de service. Et le projet DOTS ne va pas proposer de solution magique, uniquement des mécanismes de cooordination et d'échange en cas d'attaque. La réponse à une attaque dDoS consiste typiquement à examiner les paquets entrants, et à jeter ceux qui semblent faire partie de l'attaque. (Voir par exemple mon article sur le filtrage <https://www.bortzmeyer.org/rate-limiting-dos.html>.) Il faut bien sûr le faire le plus tôt possible. Si vous êtes connecté à l'Internet par un lien de capacité <https://www.bortzmeyer.org/capacite.html> 1 Gb/s, et que l'attaquant arrive à le saturer par les paquets qu'il envoie, trier les paquets de votre côté ne servira à rien, cela serait trop tard ; ils doivent être triés en amont, par exemple chez votre opérateur. Et, évidemment, trier n'est pas trivial, les paquets ne sont pas marqués comme participant à l'attaque (sauf si on utilise le RFC 3514, mais regardez sa date de publication). Il y aura donc toujours des faux positifs, des paquets innocents jetés. (Pour un exemple de solution anti-dDoS, voir le VAC <https://www.ovh.com/fr/anti-ddos/mitigation.xml> d'OVH, et les nombreux articles qui lui ont été consacrés.) En 2019, beaucoup d'organisations ne font plus ce tri elles-mêmes (par manque de moyens financiers, et surtout humains) mais sous-traitent à un fournisseur spécialisé (comme Arbor, pour lequel travaille un des auteurs du RFC). On envoie le trafic vers ce fournisseur, par des astuces DNS ou BGP, il le trie, et vous renvoie ce qui lui semble inoffensif. Ce tri se nomme en anglais "*scrubbing*". Ces fournisseurs sont donc un élement critique, par exemple parce qu'ils voient passer tout votre trafic. En général, on active ce service à la demande, et cette activation est un des scénarios d'utilisation de DOTS les plus cités dans le RFC.

Actuellement, l'activation du service de "*scrubbing*" se fait via des interfaces privatrices, fournies par le « protecteur », ce qui contribue à enfermer le client dans sa relation avec le fournisseur. Et puis, parfois, il faut que plusieurs acteurs participent à la réponse à attaque. D'où l'idée du projet DOTS ("*dDoS Open Threat Signaling*") qui va développer une interface normalisée, au sein du groupe de travail du même nom <https://datatracker.ietf.org/wg/dots/> à l'IETF.

La section 1.2 du RFC précise le terminologie employée : DOTS sera client/serveur, le client DOTS étant chez la victime, qui cherche une solution, et le serveur DOTS étant chez le protecteur ("*mitigator*" dans le RFC). Rappelez-vous que DOTS ne normalise pas les méthodes de protection (elles évoluent vite, même si le motif « tri puis poubellisation des paquets » reste dominant), mais uniquement la communication entre les acteurs impliqués. Les différents acteurs communiquent avec deux sortes de canaux, les canaux de signalisation et les canaux de données. Les premiers sont prévus pour des messages assez courts (« jette tous les paquets à destination du port NNN ») mais qui doivent arriver à tout prix, même en cas d'attaque intense ; ils sont le cœur du système DOTS, et privilégient la survivabilité. Les seconds, les canaux de données, sont prévus pour de grandes quantités de données, par exemple pour envoyer des informations de configuration, comme la liste des préfixes IP à protéger.

L'essentiel du RFC est la section 2, qui décrit les exigences auxquelles devra se soumettre le futur protocole DOTS. (Notez que le travail est déjà bien avancé. Depuis, un RFC d'architecture générale du systéme, le RFC 8811, a été publié, et les RFC 8782 et RFC 8783 ont normalisé le protocole.) Il s'agit d'exigences techniques : ce RFC ne spécifie pas d'exigences "*business*" ou de politique. Par exemple, il ne dit pas à partir de quand un client DOTS a le droit de demander une action au serveur, ni dans quels cas le client a le droit d'annuler une demande.

Le protocole DOTS a des exigences difficiles ; compte-tenu du caractère très sensible des échanges entre le client et le serveur, il faut absolument fournir authentification, intégrité, confidentialité et protection contre le rejeu par un tiers. Autrement, le protocole DOTS, comme la plupart des techniques de sécurité, pourrait en fait fournir un nouveau moyen d'attaque. Mais, d'un autre côté, le protocole doit être très robuste, puisqu'il est précisément conçu pour fonctionner face à un hostile, qui va essayer de perturber les communications. Combiner toutes ces demandes n'est pas trivial. DOTS fournira de la robustesse en utilisant des messages de petite taille (qui auront donc davantage de chances de passer), asynchrones, et qui pourront être répétés sans dommage (en cas de doute, les acteurs DOTS n'hésiteront pas à envoyer de nouveau un message).

Je ne vais pas répéter ici la longue liste des exigences, vous les trouverez dans la section 2. Elles sont réparties en plusieurs catégories. Il y a d'abord les exigences générales :

— Le protocole doit être extensible, car les attaques par déni de service vont évoluer, ainsi que les solutions (la lutte de l'épée et de la cuirasse est éternelle),
— Comme rappelé ci-dessus, le protocole doit être robuste, la **survivabilité** doit être sa principale qualité puisqu'il est prévu pour fonctionner en situation très perturbée,
— Un message qui a du mal à passer ne ne doit pas bloquer le suivant (pas de "*head-of-line blocking*"),
— Le client doit pouvoir donner des indications sur les actions souhaitées, puisqu'il dispose parfois d'informations, par exemple issues du renseignement sur les menaces.

Il y a ensuite les exigences sur le canal de signalisation :

— Il doit utiliser des protocoles existants comme UDP (TCP est possible mais, en cas d'attaque, il peut être difficile, voir impossible, d'établir une connexion),
— La taille des messages doit être faible, à la fois pour augmenter les chances de passer malgré l'attaque, et pour éviter la fragmentation,
— Le canal doit être bidirectionnel, entre autres pour détecter une éventuelle coupure du lien (un serveur peut être configuré pour activer les solutions anti-dDoS précisément quand il ne peut plus parler au client, des messages de type battement de cœur sont donc nécessaires, mais pas évidents à faire correctement, cela avait été une des plus grosses discussions à l'IETF),
— Le client doit pouvoir demander au serveur des actions (c'est le but principal du protocole), et doit pouvoir aussi solliciter des informations sur ce que voit et fait le serveur DOTS (« j'ai jeté 98 % des paquets » ou « je jette actuellement 3,5 Gb/s »),
— Le client doit pouvoir tenir le serveur au courant de l'efficacité perçue des actions effectuées (« ça marche pas, je reçois toujours autant »),
— Le client doit pouvoir indiquer une durée pour les actions du serveur, y compris une durée infinie (si le client préfère que tout son trafic soit examiné et filtré en permanence),
— Le client doit pouvoir demander un filtrage fin, en indiquant une portée (« uniquement ce qui vient de `192.0.2.0/24` » ou « seulement les paquets à destination de `2001:db8:a:b::/64` », voire « seulement les paquets pour `www.example.com` », et le serveur DOTS doit alors faire la résolution de nom).

Il y a aussi des exigences pour l'autre canal, celui des données. Rappelons que, contrairement au canal de signalisation, il n'est pas indispensable qu'il puisse fonctionner pendant l'attaque. La principale exigence est la transmission fiable des données.

Vu le contexte de DOTS, il y a évidemment des exigences de sécurité :

— Authentification mutuelle (du serveur par le client et du client par le serveur), un faux serveur ou un faux client pourraient faire des catastrophes, cf. section 4 du RFC,
— Confidentialité et intégrité, vu le caractère critique des données (imaginez si un attaquant pouvait modifier les messages DOTS…)

La section 3 du RFC se penche sur le problème de la congestion. Le protocole DOTS ne doit pas ajouter à l'attaque en noyant tout le monde sous les données, alors qu'il utilisera sans doute un transport qui ne gère pas lui-même la congestion, UDP (au moins pour le canal de signalisation). Il faudra donc bien suivre les règles du RFC 8085.

À noter qu'Arbor a un brevet sur les mécanismes analogues à DOTS (brevet 20130055374 <`http://patft.uspto.gov/netacgi/nph-Parser?Sect1=PTO2&Sect2=HITOFF&p=1&u=%2Fnetahtml%2FPTO%2Fsearch-bool.html&r=1&f=G&l=50&co1=AND&d=PTXT&s1=20130055374&OS=20130055374&RS=20130055374`>, signalé à l'IETF ici <`https://datatracker.ietf.org/ipr/2744/`>.) Arbor promet des licences RF et RAND. Même les attaques créent du business…