Situphen

Zeste de savoir

L'histoire du premier (petit) DDOS subi par Zeste de Savoir

9 avril 2021

Table des matières

1. Houston, we've had a problem.

Dimanche 28 mars 2021 dans la soirée, plusieurs membres déclarent sur notre serveur Discord non-officiel ne pas avoir pu accéder au site web pendant plusieurs secondes:

— «Y avait une maintenance de prévu? Le site a été down plusieurs secondes là.» et «Je me prends des erreurs 500 en essayant de joindre le site.» aux alentours de 20h45;
— «y'a une erreur serveur sur ZdS?» aux alentours de 21h30.

Quand je vois ces messages un peu plus tard, le site web est de nouveau disponible. Il ne s'agit donc pas d'une urgence, mais il faut néanmoins investiguer et trouver la cause de l'inaccessibilité du site web! Un des messages mentionne une erreur 500, c'est-à-dire une erreur interne au serveur, généralement causée par par un dysfonctionnement de notre site web écrit en Python avec Django. Lorsqu'un tel dysfonctionnement se produit, les détails du dysfonctionnement sont remontées sur notre Sentry, un outil qui permet d'avoir un aperçu des différents dysfonctionnements qui se sont produit. Mais ce soir là, aucun nouveau dysfonctionnement n'apparaît!

? **Que se passe-t-il quand je visite Zeste de Savoir?**

Afin que vous puissiez correctement comprendre la suite, je fais un petit détour par le fonctionnement interne de Zeste de Savoir. Lorsque vous visitez notre site web, votre navigateur web (Google Chrome, Mozilla Firefox, Microsoft Edge, etc.) envoie une requête à notre serveur web NGINX qui est chargé de réceptionner les requêtes et de faire un premier tri. Si la requête concerne un fichier statique (une image, un PDF, un fichier CSS, etc.), NGINX s'en charge tout seul. Si la requête concerne une page web, alors il la transmet à notre serveur applicatif Gunicorn qui est chargé de répartir les requêtes entre différents processus qui traitent les requêtes du site web.
On a donc ce schéma: Vous NGINX Gunicorn Site web

Un peu plus tard je pense à regarder l'historique de journalisation du serveur web NGINX, c'est-à-dire l'historique des différents événements importants qui se sont produit dans le serveur web. Et là, je vois s'afficher des centaines de lignes d'erreurs semblables à celle-ci:

```
2021/03/28 21:30:33 [error] 2351#2351: *12699791 connect() to
    unix:/opt/zds/run/gunicorn.sock failed (11: Resource
    temporarily unavailable) while connecting to upstream, client:
    xxx.xxx.xxx.xxx, server: zestedesavoir.com, request: "GET
    /articles/232/les-failles-xss/ HTTP/1.1", upstream:
    "http://unix:/opt/zds/run/gunicorn.sock:/articles/232/les-
    failles-xss/", host: "zestedesavoir.com", referrer:
    "https://google.com"
```

Je lis donc que notre serveur web NGINX n'a pas réussi à communiquer avec notre serveur applicatif Gunicorn car celui-ci est temporairement indisponible (`connect() to unix:/opt/zds/run/gunicorn.sock failed (11: Resource temporarily unavailable) while connecting to upstream`). Je remarque aussi que la plupart de ces requêtes sont dirigées vers une seule et même page (`request: "GET /articles/232/les-failles-xss/ HTTP/1.1"`). On vient de subir la première attaque *DDOS* dans l'histoire de Zeste de Savoir!

Qu'est-ce qu'une attaque DDOS?

Une attaque *DDOS*, de l'anglais *Distributed Denial-Of-Service*, est une attaque distribuée par déni de service . L'attaquant envoie un très grand nombre de requêtes sur un serveur qui se retrouve submergé et ne peut plus répondre aux demandes des utilisateurs légitimes. Le serveur n'est plus disponible, il y a donc *déni de service*. Lorsque les requêtes proviennent de milliers d'ordinateurs différents, on dit qu'elle est *distribuée*.

Grâce à l'historique de journalisation de NGINX j'ai pu analyser un peu le déroulement de l'attaque:

- 1ère vague aux alentours de 20h45
  - environ 15 700 requêtes sont reçues sur une durée d'environ 130 secondes soit environ 120 requêtes par seconde
  - environ 3 600 requêtes sont traitées par Gunicorn
  - environ 12 100 requêtes échouent et sont renvoyées avec une erreur interne
- 2ème vague aux alentours de 21h30
  - environ 8 900 requêtes sont reçues sur une durée d'environ 75 secondes soit environ 120 requêtes par seconde
  - environ 2 070 requêtes sont traitées par Gunicorn
  - environ 6 830 requêtes échouent et sont renvoyées avec une erreur interne

Ces requêtes proviennent de plus de 4 000 adresses IP différentes dont la très grande majorité n'ont envoyé qu'une poignée de requêtes. L'analyse de quelques adresses IP ne montre pas de liens entre elles. Cela confirme que l'attaque était bien distribuée et cela rend difficile l'identification de sa source.

Cela peut paraître énorme, et c'est effectivement beaucoup pour Zeste de Savoir, mais on reste très loin des attaques *DDOS* auxquelles peuvent être confrontés des organismes plus prestigieux!

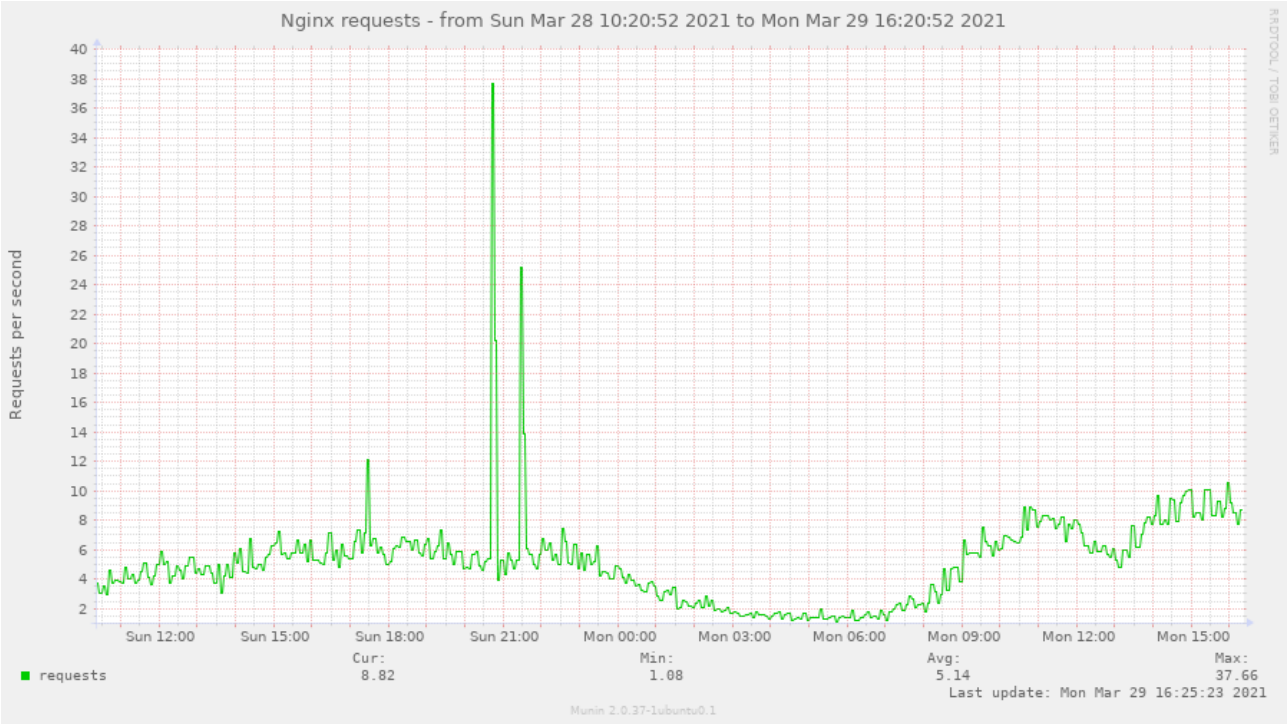

FIGURE 1.1. – Sur ce graphique qui montre le nombre de requêtes par seconde reçues par notre serveur web NGINX, on voit bien les deux pics à 20h45 et 21h30.

? Pourquoi les deux pics sont respectivement à 38 et 25 requêtes par seconde au lieu de 120 requêtes par seconde?

Pour calculer les 120 requêtes par seconde, j'ai divisé le nombre de requêtes (15 700 pour la 1ère vague) par la durée (130 secondes). Ce graphique, généré par notre Munin qui permet de garder un œil sur les différentes constantes vitales du serveur, n'est pas aussi précis! Les valeurs affichées sont des moyennes sur environ 5 minutes. Par exemple pour la première vague, 38 requêtes par seconde est la moyenne de la 1ère vague à 120 requêtes par seconde pendant 2 minutes et le trafic normal du site web pendant 3 minutes.

2. Un problème ? Une solution !

Soyons sincère, l'équipe technique de Zeste de Savoir n'est pas assez compétente pour se prémunir entièrement des différentes attaques par déni de service tellement elles sont variées et peuvent être complexes. Néanmoins, il serait quand même bien de pouvoir résister à des attaques simples:

— un grand nombre de requêtes provenant d'une seule adresse IP;
— un grand nombre de requêtes provenant de plusieurs adresses IP visant une page en particulier (comme l'attaque du 28 mars).

Pour cela, j'ai configuré NGINX de telle sorte à limiter:

— le nombre de requêtes par seconde par adresse IP;

— le nombre de requêtes par seconde par URL.

Tant que vous restez en dessous des limites, rien ne se passe. Si les limites sont légèrement dépassées à un instant donné, par exemple lors d'un pic de visites, alors une partie des requêtes sont légèrement retardées pour étaler le pic de visites. Au-delà, les requêtes sont rejetées et une belle page d'erreur s'affichera!

Il faut trouver le juste milieu entre une limite trop forte du nombre de requêtes par seconde (auquel cas on risque de bloquer des utilisateurs avec un usage normal) et une limite trop faible (auquel cas on risque de ne pas bloquer efficacement les attaques simples). Ces limitations ont été testées sur le serveur de bêta et peaufinées. Elles continueront d'être peaufinées si besoin sur le serveur de production. N'hésitez pas à nous contacter si vous rencontrez régulièrement la page d'erreur, ce serait le signe qu'il y a besoin de mieux régler ces limitations!

FIGURE 2.2. – Voici la page d'erreur que vous allez peut-être rencontrer un jour, même si on n'espère pas. Elle est belle hein !

J'en profite pour rappeler que de telles attaques par déni de service sont punies par la loi française.

> En France, l'attaque par déni de service est punie pénalement à l'article 323–2 du Code Pénal: «Le fait d'entraver ou de fausser le fonctionnement d'un système de traitement automatisé de données est puni de cinq ans d'emprisonnement et de 75000 euros d'amende».
>
> *Attaque par déni de service - Wikipédia*

Voici quelques liens sur les possibilités offertes par NGINX pour résister à des attaques simples:

— Mitigating DDoS Attacks with NGINX and NGINX Plus
— Rate Limiting with NGINX and NGINX Plus
— NGINX rate-limiting in a nutshell
— Limiting Access to Proxied HTTP Resources

Je remercie @Amaury pour la relecture!