# Détection d'attaques DDoS dans les réseaux

Gilles Roudière
LAAS-CNRS
Email: gilles.roudiere@laas.fr

***Résumé*—La menace que représente la recrudescence des attaques par déni de service distribuées (DDoS) reste une préoccupation majeure pour de nombreux acteurs de l'Internet. Par l'envoi de nombreuses requêtes parasites, ces attaques cherchent à épuiser les ressources de la victime. Elle peuvent alors paralyser les systèmes ciblés, les empêchant de répondre aux requêtes légitimes.**

**Avant de pouvoir être neutralisées, ces attaques doivent être identifiées l'aide de détecteurs capables de traiter le trafic en temps réel, aussi autonomes que possible et devant fournir les informations nécessaires à une décision appropriée. Bien que les méthodes d'apprentissage automatique récentes ont permis la création de détecteurs plus autonomes, les nombreux faux-positifs qu'ils produisent et les ressources de calcul conséquentes qu'ils nécessitent ont rendu ces détecteurs peu compétitifs. Ainsi, malgré l'engouement durable de la communauté scientifique pour ce sujet, aucun détecteur ne semble faire véritablement consensus.**

**Pour répondre à cette problématique, nous proposons AATAC, un nouveau détecteur de DDoS autonome et temps réel, nécessitant peu de ressources de calcul. De part sa nature non supervisée, il peut également détecter des anomalies ou attaques inconnues jusqu'alors.**

## I. Introduction

La multiplication des attaques, ou plus largement des anomalies réseau, reste une préoccupation majeure des acteurs de l'Internet. Ces anomalies, généralement définies comme des évènements ne correspondant pas à l'état normal du trafic, peuvent avoir un impact conséquent sur la qualité du service rendu.

Parmi l'ensemble des anomalies se trouvent les attaques par déni de service distribuée (DDoS). Ce type d'attaque consiste à utiliser un ensemble de machines compromises pour envoyer des requêtes illégitimes à un système cible. Ces requêtes inutiles épuisent alors certaines ressources de la victime (bande passante, capacité de calcul...) provoquant jusqu'à l'incapacité totale de la victime à répondre aux requêtes légitimes. Considérant le danger qu'elles représentent, se protéger contre de telles attaques est une priorité pour de nombreux acteurs de l'Internet.

Lutter efficacement contre ces attaques nécessite néanmoins des outils adaptés, permettant une réponse rapide et nécessitant peu de travail de l'administrateur. La construction de détecteurs temps-réel, autonomes, et capables de fournir un maximum d'information sur la nature des anomalies est donc nécessaire.

Parmi les propositions récentes proposées dans la littérature [1]–[4], nous n'avons pas trouvé de détecteur à la fois complètement autonome et fournissant plus de détails sur les anomalies détectées. Malgré l'intérêt que porte la communauté scientifique au sujet, aucune méthode de détection particulière ne semble faire consensus. Bien que les approches récentes basées sur l'apprentissage automatique permettent une meilleure autonomie de la détection, elles produisent généralement de nombreux faux-positifs et souffrent souvent d'une complexité algorithmique importante. Elles nécessitent donc une capacité de calcul conséquente pour traiter le trafic en temps réel.

Pour répondre à cette problématique, nous proposons AATAC, un nouvel algorithme de détection d'anomalies nécessitant peu de ressources pour opérer en temps réel, non supervisé et conçu pour détecter et caractériser les attaques DDoS. Il permet également de détecter d'autres types d'anomalies.

Nos travaux sont réalisés dans le cadre d'un projet en collaboration avec Border 6, un éditeur de logiciel fournissant une solution d'optimisation du plan de contrôle BGP (dit BGP-SDN). Leurs clients, souvent des sites de vente en ligne, souffrent régulièrement d'attaques DDoS.

## II. AATAC

Dans cette section, nous décrivons notre algorithme de détection d'anomalies appelé AATAC (Autonomous Algorithm for Traffic Anomalies Characterization). Il se base sur l'hypothèse qu'une attaque DDoS impacte considérablement le profil du trafic au moment où elle à lieu. L'approche utilisée consiste alors à construire un modèle du trafic de manière non supervisée. Ce modèle inclut une forte composante temporelle, permettant de détecter les éventuelles déviations au fil du temps.

Pour construire ce modèle, un traitement en deux partie est effectué. Il est illustré par la figure 1. Une partie dite "en ligne" exécute un traitement rapide - de complexité linéaire - sur le trafic. Elle alimente différentes structures de données qui serviront à construire ce que nous appelons des "instantanés" du trafic. Ces instantanés contiennent un ensemble de métriques caractérisant le trafic à un moment donné. La deuxième partie de l'algorithme, dite "hors ligne" produit effectivement cet instantané à intervalles réguliers. Pour simplifier l'analyse, les instantanés produits peuvent être tracés en un ensemble de graphiques en deux dimensions.

Lorsqu'un nouvel instantané est produit, il est alors comparé à ceux précédemment créés. Une méthode basée sur du clustering nous permet alors d'identifier si l'instantané est statistiquement anormal, et si tel est le cas de lever une alerte.

Notre approche répond à plusieurs problématiques. En premier lieu, la séparation du traitement en deux parties de complexités différentes rend le détecteur robuste aux augmentations soudaines de trafic, et permet son opération en temps

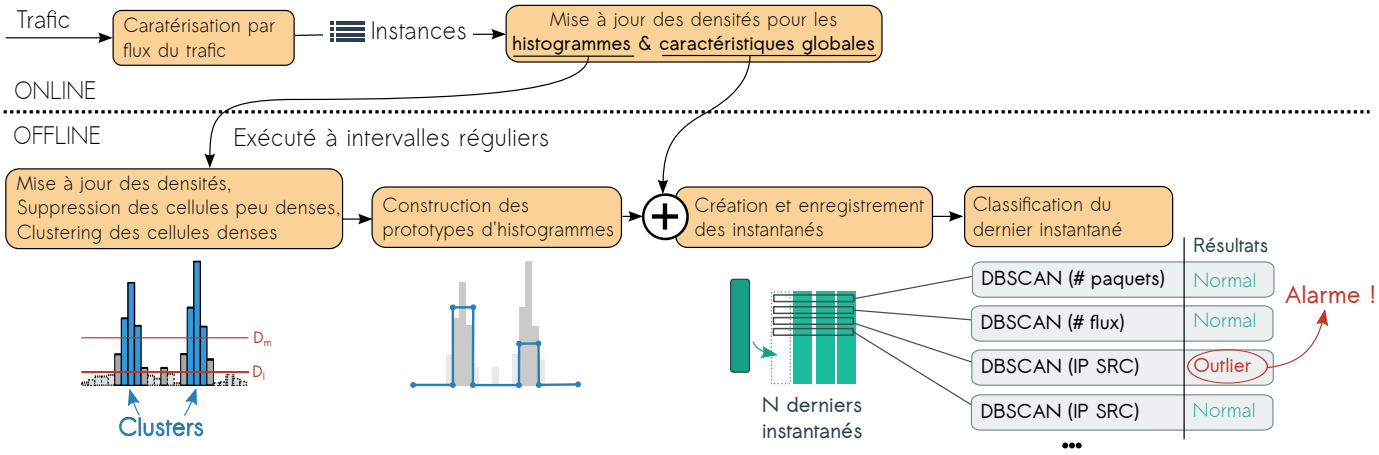


FIGURE 1. Architecture de l'algorithme AATAC

réel. De plus, étant un algorithme non-supervisé (il ne repose sur aucune base de connaissance), AATAC est entièrement autonome, et ne nécessite que peu de configuration pour opérer. Enfin, la représentation graphique des instantanés du trafic donne à l'administrateur une vue exhaustive de l'état du trafic lorsqu'un une anomalie est détectée.

Bien que la modélisation a l'aide d'histogrammes ait été utilisée dans la détection d'anomalies en général [5]–[7], c'est moins le cas pour la détection d'attaques DDoS. Néanmoins, Kind et al. [8], proposent un détecteur similaire, basé lui aussi sur du clustering d'histogramme. AATAC s'en distingue par le fait qu'il n'a pas besoin d'un entrainement préalable et réduit au minimum ses besoins en temps de calcul. De nombreuses approches, basées sur l'entropie [4], [9], [10], ont aussi été proposées. Elles apportent néanmoins peu d'information sur les anomalies détectées. Notons que AATAC, répond aussi aux limites de notre précédent algorithme, UNADA [11], qui nécessitait des ressources de calcul beaucoup plus conséquentes car non focalisé sur la détection des DDoS.

### A. Traitement en ligne

En entrée de notre algorithme, nous utilisons des données agrégées par flux. La partie en ligne de l'algorithme (partie supérieure de la figure 1) les utilise pour alimenter un ensemble de compteurs génériques (pourcentage de paquets SYN, bande passante...), et construit une structure de données permettant de caractériser les distributions de plusieurs attributs du trafic (adresses IP, ports, taille des paquets...). Ces données seront ensuite utilisées par la partie hors ligne pour construire un instantané du trafic.

Pour caractériser les distributions, nous utilisons une méthode inspirée de l'algorithme de clustering D-Stream [12]. Cet algorithme divise l'espace de valeurs en $p$ partitions de taille égale. A chacune de ces partitions peut être associée une cellule, caractérisée ici par deux variables : une date de mise à jour $t_u$ et une densité $d$.

Lorsqu'une instance est ajoutée au modèle, suivant la partition à laquelle elle appartient, deux situations sont possibles : si la cellule associée à la partition n'existe pas, elle est créée et initialisée avec une densité $d$ à 1. Si elle existe, sa densité $d$ est mise à jour à partir de la formule suivante :

$$d_{new} = \lambda^{(t_n - t_u)} d_{old} + 1 \quad (1)$$

Avec $t_n$ l'estampille actuelle et $t_u$ l'estampille de dernière mise à jour de la cellule. La constante $\lambda$, appelée facteur de dégradation, est un paramètre de l'algorithme.

Ainsi une cellule recevant fréquemment des instances aura une densité élevée, alors qu'une cellule en recevant rarement aura une densité faible. Les espaces de valeurs utilisés étant mono-dimensionnels, cette structure de données s'assimile donc à un histogramme.

Cette méthode a l'avantage d'être de complexité linéaire avec le nombre d'instances et ne nécessite pas de stocker les instances pour leur analyse.

### B. Traitement hors-ligne

Le traitement hors-ligne, exécuté à intervalles réguliers, peut être divisé en plusieurs parties. Comme illustré dans la partie inférieure de la figure 1, un premier traitement est effectué sur les histogrammes utilisés pour décrire les distributions du trafic (II-B1). Le résultat de ce traitement est ensuite utilisé pour construire un instantané du trafic (II-B2), qui sera enfin soumis à une détection d'anomalie (II-B3).

*1) Suppression des cellules peu denses et construction des prototypes:* La première étape du traitement hors ligne vise à mettre à jour la densité de chaque cellule. Considérant $t_h$ la date à laquelle est effectué le traitement hors-ligne, la densité de chaque cellule est mise à jour de la manière suivante :

$$d_{new} = \lambda^{(t_h - t_u)} d_{old} \quad (2)$$

Ensuite, pour simplifier les structures utilisées par la partie en ligne, les cellules ayant une densité très faible sont supprimées. Ayant reçu peu d'instances récemment, elles ne sont plus utiles à la représentation du trafic actuel. Les supprimer permet alors d'accélérer la recherche de la cellule correspondant à chaque instance et limite le nombre de cellules nécessaires à

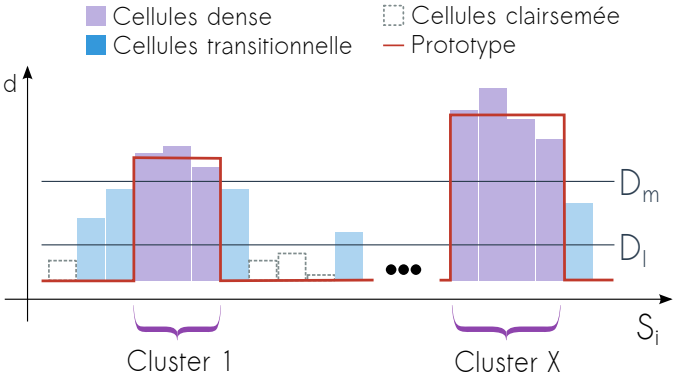


FIGURE 2. Suppression des cellules clairsemées et construction du prototype d'histogramme

l'exécution de l'algorithme. Le seuil de densité minimale est nommé $D_l$.

Dans un second temps, les cellules ayant une densité supérieure à un seuil $D_m$, dites denses, sont utilisées pour construire un prototype résumant la distribution à l'instant de la création de l'instantané. Les cellules adjacentes sont ainsi groupées en clusters dont plusieurs caractéristiques sont enregistrées, à savoir : les extrémités du clusters ($min_c$ et $max_c$) sur la dimension considérée, ainsi que la densité moyenne des cellules incluses dans le cluster ($avg_c$).

Ces données sont utilisées pour produire une représentation simplifiée de la distribution appelée prototype d'histogramme. La courbe est produite en ajoutant, pour chaque cluster, les 4 points $(min_c, 0)$, $(min_c, avg_c)$, $(max_c, avg_c)$ et $(max_c, 0)$, à l'ensemble des points constituant le prototype. Cette construction est illustrée par la figure 2.

*2) Construction des instantanés du trafic:* L'ensemble des prototypes produits est ensuite associé à plusieurs métriques, issues de compteurs alimentés par la partie en ligne. En effet, les prototypes de distributions ne suffisent pas à décrire précisément le trafic à un instant précis, des informations comme le nombre de paquets ou le nombre de flux enregistrés depuis le dernier instantané sont des informations permettant de mieux caractériser le trafic. L'ensemble des prototypes de distribution associé à ces métriques constitue alors un instantané du trafic.

*3) Détection des instantanés anormaux:* Afin de détecter les instantanés anormaux, nous considérons les $N$ derniers instantanés enregistrés.

Grâce à l'application de l'algorithme de clustering DBSCAN, les instantanés similaires sont regroupés en clusters et les instantanés anormaux isolés (outliers). Afin d'éviter la malédiction de la dimension, DBSCAN est appliqué plusieurs fois, considérant à chaque fois une composante différente des instantanés. Ainsi, pour les métriques simples, la distance euclidienne est utilisée alors que pour les prototypes de distribution, c'est l'aire entre les deux courbes qui est utilisée.

Le dernier instantané créé est considéré comme anormal s'il apparait comme outlier dans au moins une des itération de DBSCAN. Auquel cas une alerte est levée.

En cas d'anomalie, les $N$ derniers instantanés créés peuvent alors être enregistrés créant une image dynamique du trafic. Les changement au sein des caractéristiques du trafic apparaissent alors clairement, donnant à l'administrateur réseau la possibilité de prendre une décision appropriée quant à l'anomalie détectée.

## III. EVALUATION

Dans le cadre du projet Border 6/LAAS, l'opportunité nous a été donnée de capturer le trafic à l'entrée du réseau d'une entreprise dont l'activité principale est l'hébergement de sites de vente en ligne. Ces traces serviront à évaluer la capacité de notre algorithme à opérer en temps réel sur du trafic réaliste. L'exactitude de la détection sera évaluée grâce à d'autres ensembles de données, incluant au moins un ensemble de données mis à disposition par la communauté scientifique (UNB ISCX [13]) ainsi qu'un ensemble que nous avons construit par émulation.

## IV. BIOGRAPHIE DE L'AUTEUR

J'ai débuté en 2009 mes études supérieures à l'Institut National des Sciences Appliquées (INSA) de Toulouse, pour y obtenir en 2014 mon diplôme d'ingénieur. J'y ai suivi la filière Informatique, Réseaux et Télécommunications, conjointement avec l'option sécurité. J'ai par la suite eu l'opportunité de démarrer une thèse au Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS-CNRS), dont j'entame la deuxième année. Cette thèse est effectuée dans le cadre d'un projet avec Border 6, et vise à proposer de nouveaux algorithmes autonomes pour la détection des anomalies sur réseaux Internet.

## RÉFÉRENCES

[1] S. T. Nezhad, M. Nazari, and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks," vol. 20, no. 4, pp. 700–703, 2016.

[2] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using Artificial Neural Networks," *Neurocomputing*, vol. 172, pp. 385–393, 2016. [Online]. Available : http ://dx.doi.org/10.1016/j.neucom.2015.04.101

[3] D. van der Steeg, R. Hofstede, A. Sperotto, and A. Pras, "Real-time DDoS Attack Detection for Cisco IOS using NetFlow," *IFIP/IEEE Int. Symp. Integr. Netw. Manag. Exp. Sess. Pap.*, pp. 972–977, 2015.

[4] J. David and C. Thomas, "DDoS Attack Detection Using Fast Entropy Approach on Flow- Based Network Traffic," *Procedia Comput. Sci.*, vol. 50, pp. 30–36, 2015. [Online]. Available : http ://linkinghub.elsevier.com/retrieve/pii/S1877050915005086

[5] D. Anderson, T. Frivold, and A. Valdes, "Next-generation Intrusion Detection Expert System (NIDES) : A summary," Tech. Rep. May 1995, 1995.

[6] K. Yamanishi, J.-i. Takeuchi, G. Williams, and P. Milne, "On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms," *Data Min. Knowl. Discov.*, vol. 8, no. 3, pp. 275–300, 2004. [Online]. Available : http

[7] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," *19th Annu. Comput. Secur. Appl. Conf. 2003 Proc.*, no. Acsac, pp. 14–23, 2003. [Online]. Available : http ://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm ?arnumber=1254306

[8] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, "Histogram-Based Traffic Anomaly Detection," vol. 6, no. 2, pp. 110–121, 2009.

[9] G. No and I. Ra, "Adaptive DDoS detector design using fast entropy computation method," *Proc. - 2011 5th Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput. IMIS 2011*, pp. 86–93, 2011.

[10] J.-h. Jun, D. Lee, and S.-h. Kim, "DDoS Attack Detection Using Flow Entropy and Packet Sampling on Huge Networks," *Thirteen. Int. Conf. Networks.*, no. c, pp. 185–190, 2014.

[11] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised Network Intrusion Detection Systems : Detecting the Unknown without Knowledge," *Comput. Commun.*, vol. 35, no. 7, pp. 772–783, 2012. [Online]. Available : http ://dx.doi.org/10.1016/j.comcom.2012.01.016

[12] Y. Chen and L. Tu, "Density-Based Clustering for Real-Time Stream Data," *KDD '07 Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, vol. d, pp. 133–142, 2007.

[13] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2011. [Online]. Available : http ://dx.doi.org/10.1016/j.cose.2011.12.012