Détection et localisation décentralisées d'anomalies dans le trafic internet

Alexandre LUNG-YUT-FONG, Olivier CAPPÉ, Céline LÉVY-LEDUC, François ROUEFF

Institut Télécom & CNRS, Télécom ParisTech, LTCI
46, Rue Barrault, 75634 Paris Cédex 13, France
{lung,cappe,levyledu,roueff}@telecom-paristech.fr

Résumé – Nous proposons une méthode de détection et localisation décentralisée d'anomalies en faisant coopérer des sondes n'ayant accès qu'à une partie du trafic réseau et en minimisant la quantité d'information échangée dans le réseau.

Cette méthode se décompose en trois étapes : au niveau de chaque sonde disséminée dans le réseau est effectué un filtrage par records suivi d'un test de rang non-paramétrique pour données censurées. Chaque sonde transmet alors au collecteur central les séries temporelles censurées des machines les plus suspectes. Les données remontées par chaque sonde sont alors fusionnées au niveau du collecteur central qui fournit une liste de machines attaquées en appliquant le même test de rang sur ces données agrégées.

La méthode que nous proposons a été testée avec succès sur des données d'opérateur et permet de détecter en temps réel les anomalies réseaux et d'identifier les machines attaquées. Elle obtient de meilleures performances que les méthodes usuelles pour des attaques très distribuées c'est à dire difficiles à détecter localement (au niveau des sondes).

Abstract – We propose an efficient and decentralised method for anomaly detection in the Internet traffic. Local sensors cooperate with each other while having only access to a part of the global trafic. This method can be described in three steps. A record filtering followed by a nonparametric rank test for censored data are done by each sensor in the network. Then the sensors transmit to the central collector the censored time series that corresponds to the most suspicious addresses. This data is then aggregated at the fusion centre that then applies the same change-point detection test as in the local sensors so as to output a list of potentially attacked computers.

The method that we propose has been succesfully tested on data from a major network operator, and has been able to detect in real-time network anomalies and to locate the attacked addresses. Compared to some common methods for distributed attacks, the *Distributed TopRank* yields promising results for very distributed attacks, *i.e.*, attacks that are difficult to detect locally.

1 Introduction

Les effets des attaques réseaux pouvant être désastreux, arriver à s'en prémunir est une préoccupation majeure des fournisseurs d'accès et de services en ligne. Parmi ces comportements malveillants, on peut citer les attaques de type déni de service (*DoS*) ou leur version distribuée (*DDoS*) qui sont des attaques par saturation. Pour un opérateur de réseau, il peut être important de détecter une telle attaque et de localiser les machines victimes afin de pouvoir prendre les mesures adéquates. Différentes méthodes peuvent être utilisées à cet effet. Une première approche utilisée par les systèmes dits à signatures consiste à comparer le trafic observé avec des modèles d'attaques connues. Une autre méthode consiste à utiliser une approche statistique pour détecter les anomalies réseaux, celle-ci ne nécessitant pas une base de données d'attaques. Dans ce cas, la détection d'anomalies se fait via un test de détection de ruptures qui est une problématique classique en statistique. En effet, les attaques de type déni de service sont connues pour provoquer des changements abrupts dans le trafic réseau.

Une méthode statistique couramment utilisée en détection d'anomalies réseau est l'algorithme CUSUM [1]. Cet algorithme a été, entre autres, utilisé par [9] pour la détection d'attaques de type *TCP/SYN flooding*, qui est un exemple d'attaque de type *DoS*. Celle-ci exploite les caractéristiques du protocole TCP (*Transmission Control Protocol*) en engorgeant de paquets de synchronisation (SYN) la machine destination attaquée, qui doit tenir à jour une table de demandes de connexion en attente d'un message d'acquittement ACK (*ACKnowledgement*) de la part de la machine source. Les ressources de la machine et la taille de la table étant limitées, l'attaque peut conduire à une saturation et à une interruption du service fourni par la machine. Les machines victimes d'une attaque de type *TCP/SYN flooding* pourraient ainsi être détectées en appliquant un test de détection de ruptures aux séries temporelles correspondant au nombre de paquets SYN reçus par chaque machine destination présente dans le trafic. Cependant, un réseau typique d'opérateur fait circuler plusieurs dizaines de milliers de connexions par seconde ; les analyser toutes devient rapidement un problème difficile, au vu de la quantité massive de données à traiter. C'est pourquoi, une méthode de réduction de dimension doit être préalablement employée, par exemple avec une analyse en composantes principales [7], une agrégation aléatoire (*sketch*) [6], [3] ou un filtrage par records [8].

Par ailleurs, on s'intéresse à un cadre plus large dans lequel de multiples sondes collectent les données à différents nœuds du réseau. Une manière naïve de détecter des attaques serait que les sondes envoient les données qu'elles reçoivent à un collecteur central qui appliquerait alors un algorithme de détection de ruptures sur les données agrégées. C'est une approche que l'on pourrait qualifier de *centralisée*. Une telle méthode appelée *TopRank* a été proposée dans [8]. Avec l'explosion des débits et un nombre important de sondes, les méthodes centralisées ont cependant un problème de passage à l'échelle : la quantité de données échangées par les sondes peut devenir une charge supplémentaire importante pour le réseau ainsi que pour le collecteur. L'idée pour limiter ces quantités de données est de transférer une partie de l'intelligence au sein du réseau en mettant à contribution les sondes qui traiteront leurs données localement afin d'envoyer au collecteur central les informations les plus pertinentes pour le problème à résoudre. On parle alors de méthode *distribuée*, ou encore *décentralisée*. Dans [5], une façon de décentraliser l'approche [7] a été proposée mais celle-ci ne permet pas de localiser l'attaque en trouvant la machine attaquée. Nous proposons dans cette contribution une manière de décentraliser l'algorithme du *TopRank* qui, lui, permet de faire de la localisation.

2 Principe

Le *TopRank distribué*, que nous proposons, utilise une approche rétrospective, les données sont traitées dans une fenêtre d'observation divisée en P segments, chacun de longueur Δ secondes. Les données traitées sont ensuite effacées après leur traitement à la fin de chaque fenêtre d'observation. Une première analyse est d'abord effectuée localement dans les sondes, puis une synthèse est effectuée au sein du collecteur central. Les K sondes dont on dispose sont notées dans la suite : $M_1, \ldots, M_K$.

On note $N_i^{(k)}(t)$ le nombre de paquets en destination de la machine d'adresse IP i vus par le moniteur M_k dans l'intervalle t de longueur Δ de la fenêtre d'observation. Par exemple, pour détecter une attaque d'engorgement par paquets TCP/SYN, $(N_i^{(k)}(t))_{1\leq t\leq P}$ représente la série temporelle du nombre de paquets SYN reçus par l'adresse IP i.

2.1 Traitement local dans les sondes

Chaque sonde M_k effectue un premier traitement sur les données qu'elle recueille ; il se déroule en quatre étapes et correspond à l'algorithme *TopRank* détaillé dans [8] dont on rappelle ici le principe (pour alléger les notation, on omet l'exposant $^{(k)}$ désignant le numéro de moniteur dans les nombres de paquets) :

Filtrage par records Dans chaque sous-intervalle indexé par $t \in \{1, \ldots, P\}$ de longueur Δ secondes de la fenêtre d'observation, on garde les adresses IP i des M plus grands $N_i(t)$, que l'on note $i_1(t), \ldots, i_M(t)$ et tels que : $N_{i_1(t)}(t) \geq N_{i_2(t)}(t) \geq \cdots \geq N_{i_M(t)}(t)$. On note $\mathcal{T}(t) = \{i_1(t), \ldots, i_M(t)\}$ ce classement d'adresses IP. Notons que l'on ne garde pour la suite que les éléments de $\mathcal{T}(t)$ ainsi que les valeurs correspondantes $\{N_i(t),\, i \in \mathcal{T}(t),\, t = 1, \ldots, P\}$.

Création des séries temporelles censurées Pour chaque adresse IP i sélectionnée à l'étape précédente ($i \in \bigcup_{t=1}^{P} \mathcal{T}(t)$), on construit la série temporelle $(X_i(t))_{1\leq t\leq P}$. Cette série est censurée, puisqu'il se peut qu'à un instant t, i ne soit pas dans l'ensemble $\mathcal{T}(t)$ et que l'on ne dispose donc plus de la valeur $N_i(t)$ correspondante. Dans ce cas, $X_i(t)$ prend alors la valeur $N_{i_M(t)}(t)$. On note par ailleurs $x_i(t)$ l'état de censure de $X_i(t)$:

$$(X_i(t), x_i(t)) = \begin{cases} (N_i(t), 1), & \text{si } i \in \mathcal{T}_M(t) \\ (\min\limits_{j\in\mathcal{T}_M(t)} N_j(t), 0), & \text{sinon.} \end{cases}$$

On définit ensuite les bornes supérieure $\overline{X_i(t)} = X_i(t)$ et inférieure $\underline{X_i(t)} = X_i(t)x_i(t)$ de $X_i(t)$.

Test de détection de changement On utilise un test non-paramétrique de détection de changement pour données censurées inspiré par celui proposé par [4]. C'est un test de rang non-paramétrique utilisant une fonction de score (que l'on note A) qui généralise le test de rang de Wilcoxon aux données censurées.

On veut tester :

(H_0) : «$(N_i(t))_{1\leq t\leq P}$ sont des variables aléatoires i.i.d » contre

(H_1) : « il existe un entier r tel que $(N_i(1), \ldots, N_i(r))$ et $(N_i(r+1), \ldots, N_i(P))$ ont une distribution différente. »

Pour tout $s, t \in 1, \ldots, P$, on définit les quantités suivantes :

$$A_{s,t} = \mathbb{1}\left(\underline{X_i(s)} > \overline{X_i(t)}\right) - \mathbb{1}\left(\underline{X_i(t)} > \overline{X_i(s)}\right) \;; \quad (1)$$

$$U_s = \sum_{t=1}^{P} A_{s,t},\ s = 1, \ldots, P; \quad (2)$$

$$S_t = \left(\sum_{s=1}^{t} U_s\right) / \left(\sum_{s=1}^{P} U_s^2\right)^{1/2},\ t = 1, \ldots, P\,. \quad (3)$$

On utilise alors la quantité

$$W_P = \max_{1\leq t\leq P} |S_t| \quad (4)$$

comme statistique de test.

Sous (H_0), il est montré par exemple dans le Théorème 1 de [4] que sous certaines conditions,

$$W_P \xrightarrow{\mathcal{D}} B^\star = \sup_{0<t<1} |B(t)|, \text{ quand } P \to \infty, \quad (5)$$

où $\{B(t),\ t \in [0,1]\}$ est un pont Brownien et $\mathcal{D}$ la convergence en loi.

On associe à cette statistique de test une p-valeur : $Pval(W_P)$, où, voir par exemple [2],

$$Pval(b) = 2\sum_{j\geq 1} (-1)^{j-1} e^{-2j^2b^2} \text{ pour tout } b > 0. \quad (6)$$

Sélection des données à envoyer au collecteur central La sonde M_k choisit les d séries temporelles censurées ayant les plus petites p-valeurs et les transmet au collecteur central.

2.2 Agrégation dans le collecteur central

Le collecteur central construit alors les séries temporelles suivantes à partir des données envoyées par les sondes :

$$\underline{Z_i(t)} = \sum_{k=1}^{K} \underline{X_i^{(k)}(t)} \text{ et } \overline{Z_i(t)} = \sum_{k=1}^{K} \overline{X_i^{(k)}(t)} \,,$$

où $(\underline{X_i^{(k)}(t)}, t = 1, \ldots, P)$ et $(\overline{X_i^{(k)}(t)}, t = 1, \ldots, P)$ sont les séries temporelles associées à l'adresse IP i créées dans la sonde M_k. Le même test de détection de ruptures que celui utilisé dans les sondes est alors appliqué aux séries temporelles $\underline{Z_i}$ et $\overline{Z_i}$, l'adresse IP i étant alors déclarée comme attaquée au niveau $\alpha \in (0, 1)$ lorsque : $Pval(W_P) < \alpha$.

3 Résultats

3.1 Description des données

Pour évaluer l'approche proposée nous avons utilisé une trace de trafic réel d'opérateur réseau (connexions sur une plage de 118 minutes auxquelles une vingtaine d'attaques d'engorgement par paquets TCP/SYN vers quatre adresses IP ont été ajoutées), correspondant à du trafic pair à pair (*P2P*) ainsi qu'à des connexions ADSL. Des méthodes de sous-échantillonnage ont été utilisées afin de moduler l'intensité des attaques, et les données ont été réparties sur $K = 15$ sondes virtuelles en attribuant à chaque paire (IP source, IP destination) une sonde au hasard.

Le profil du trafic contenu dans les données utilisées est représenté sur la Figure 1. À gauche est mis en évidence le nombre total de paquets de type TCP/SYN reçus à chaque seconde par l'ensemble des adresses IP sollicitées. Le nombre de paquets reçus par les adresses IP attaquées est représenté à droite. On peut voir que les attaques se produisent autour des instants 2000s, 4000s, 6000s et 6500s. Les deux figures du haut correspondent au trafic originel, alors que celles du bas montrent le trafic dans l'une des sondes virtuelles.

En comparant la quantité de données reçues par les adresses IP attaquées (figures de droite) au trafic global (figures de gauche), on peut constater que l'on a affaire à une très grande quantité de données et que les attaques sont complètement masquées dans le trafic global et sont donc difficilement détectables.

Remarquons par ailleurs qu'une étape de réduction des données comme le filtrage par records dans les sondes locales est bien nécessaire : les données contiennent un total d'environ 1006000 adresses IP destination, soit une moyenne de 15000 IP par minute dans chacune des 118 fenêtres d'observation. Appliquer directement le test de détection de ruptures sur chacune des adresses IP destination serait difficilement réalisable dans le cadre d'un traitement en temps réel.

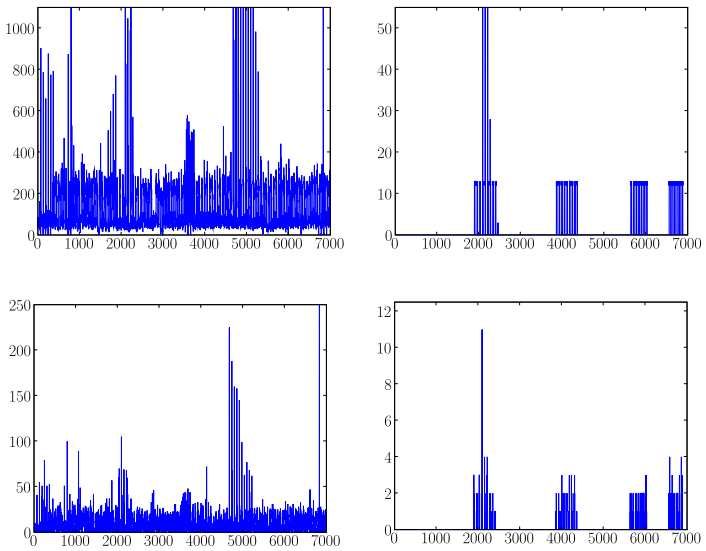

FIG. 1 – Nombre de paquets TCP/SYN globalement échangés (gauche) et reçues par les 4 adresses IP attaquées (droite) pour les données originelles (haut) et au sein d'un moniteur (bas). Notons bien que l'échelle des figures de droite s'étend sur une plage vingt fois moindre que celle des figures de gauche.

3.2 Performances

Les graphes de la Figure 2 illustrent les apports de la méthode d'agrégation utilisée dans le *TopRank distribué*. Les figures (a), (b) et (c) sont un exemple de séries temporelles correspondant à une attaque vue par trois sondes différentes au cours d'une fenêtre d'observation. La figure (d) correspond à la série qui est l'agrégation des séries provenant des 11 moniteurs ayant vu l'attaque. La p-valeur calculée sur la série temporelle agrégée dans le collecteur central est beaucoup plus faible que les p-valeurs individuellement calculées dans les sondes, ce qui permet de potentiellement déceler des attaques difficilement détectables localement *i.e.* au niveau des sondes.

Nous avons comparé le *TopRank distribué* à une méthode plus simple de décision : une série temporelle est déclarée comme attaquée si au moins une sonde a renvoyé une p-valeur plus petite que le seuil corrigé par la méthode de Bonferroni. Avec cette méthode, une adresse IP est déclarée comme attaquée au niveau $\alpha \in (0, 1)$ si au moins une des sondes a calculé une p-valeur inférieure à α/K, c'est à dire si $K(\inf_{1 \le k \le K} \text{Pval}_k) < \alpha$, Pval_k étant la p-valeur calculée dans la sonde k.

La Figure 3 représente les courbes de Caractéristiques Opérationnelles du Récepteur (COR) obtenues à partir de 50 réplications Monte Carlo pour des attaques d'intensité 12.5 SYN/s observées sur 15 sondes, pour ces deux méthodes ainsi que pour l'approche centralisée du *TopRank*. On remarque que la méthode du *TopRank distribué* a de meilleures performances que la méthode utilisant la correction de Bonferroni et qu'elle a des performances proches de celles du *TopRank* qui, rappelons-le est une approche centralisée donc plus coûteuse en termes d'informations échangées au sein du réseau.

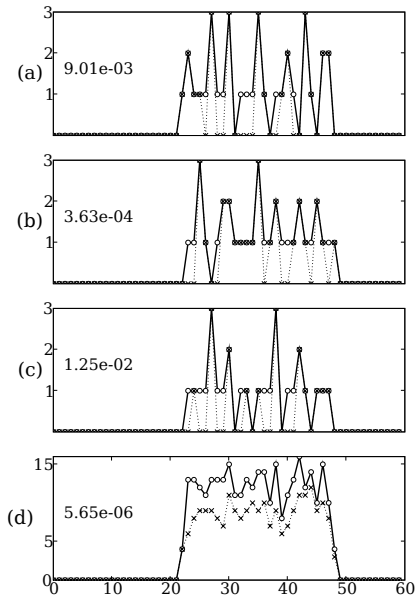

FIG. 2 – (a), (b), (c) : séries temporelles $(\underline{X_i^{(k)}}(t), 1 \leq t \leq 60)$, $(\overline{X_i^{(k)}}(t), 1 \leq t \leq 60)$ représentées avec ('x') et ('$\circ$') respectivement, pour trois valeurs de k parmi 15, (d) : série temporelle agrégée dans le collecteur $(\underline{Z_i}(t), 1 \leq t \leq 60)$, $(\overline{Z_i}(t), 1 \leq t \leq 60)$ représentées avec ('x') et ('$\circ$') respectivement. Les p-valeurs sont données dans chacun des cas.

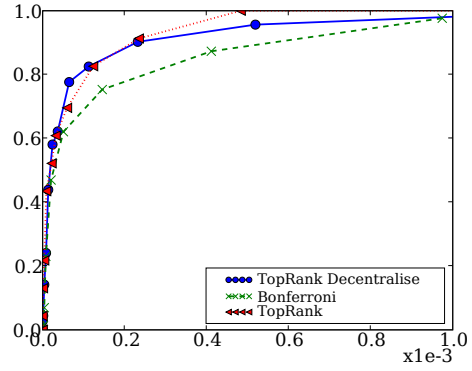

FIG. 3 – Courbes COR pour TopRank distribué ('$\circ$'), Bonferroni ('x'), TopRank ('$\triangleleft$').

4 Conclusion

Dans cette contribution, nous avons proposé avec le *TopRank distribué* un algorithme de détection et de localisation d'attaques de type *DDoS* sur le réseau internet, capable de traiter une grande quantité de données à la volée. Cette méthode, distribuée, met en œuvre un traitement local dans les sondes d'un réseau de capteur qui est basé sur une méthode de filtrage par records, suivie d'un test de rang non-paramétrique, un collecteur central faisant finalement la synthèse des tests effectués dans chacune des sondes. Ce système a pour avantage, par rapport à un schéma centralisé, de réduire la quantité de données échangées consacrées à cette tâche sans pour autant sacrifier les performances de détection. De plus, l'étape d'agrégation permet la détection d'anomalies qui seraient difficiles à détecter localement.

Références

[1] M. Basseville and I. V. Nikiforov. *Detection of Abrupt Changes : Theory and Applications*. Prentice-Hall, 1993.

[2] P. Billingsley. *Convergence of probability measures*. Wiley, New York, 1968.

[3] P. Borgnat, G. Dewaele, and P. Abry. Identification d'anomalies statistiques dans le trafic internet par projections aléatoires multirésolutions. Colloque GRETSI-2007, 2007.

[4] E. Gombay and S. Liu. A nonparametric test for change in randomly censored data. *The Canadian Journal of Statistics*, 28(1) :113–121, 2000.

[5] L. Huang, X. Nguyen, M. Garofalakis, M. Jordan, A. Joseph, and N. Taft. Distributed PCA and network anomaly detection. NIPS, 2006.

[6] B. Krishnamurthy, S. Subhabrata, and Y. Zhang. Sketch-based change detection : methods, evaluation and applications. Proceedings of IMC, 2003.

[7] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. Proceedings of SIGCOMM, 2004.

[8] C. Lévy-Leduc and F. Roueff. Detection and localization of change-points in high-dimensional network traffic data. *Annals of Applied Statistics*, 2009. To appear.

[9] A. Siris and F. Papagalou. Application of anomaly detection algorithms for detecting SYN flooding attacks. Proceedings of ICON 2004, 2004.