Détection non supervisée d'anomalies du trafic

Johan Mazel, Pedro Casas, Yann Labit, Philippe Owezarski

Détection non supervisée d'anomalies du trafic

Johan Mazel[*,**] — **Pedro Casas**[*,**] — **Yann Labit**[*,**] — **Philippe Owezarski**[*,**]

[*] *CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France*

[**] *Université de Toulouse ; UPS, INSA, INP, ISAE ; UT1, UTM, LAAS ; F-31077 Toulouse Cedex 4, France*

{jmazel,pcasashe,ylabit,owe}@laas.fr

RÉSUMÉ. *La détection d'anomalies est une tâche critique de l'administration des réseaux. Les méthodes existantes de détection d'anomalies s'appuient soit sur des signatures crées à partir d'anomalies connues, soit sur le résultat d'un apprentissage effectué sur un échantillon de trafic, deux méthodes compliquées et coûteuses en termes de ressources humaines et inefficace contre les nouvelles anomalies. Un système de détection d'anomalies doit donc être capable de s'adapter rapidement et de manière autonome à l'évolution du trafic, i.e. en évitant une coûteuse intervention humaine. Dans cet article, nous présentons une approche non-supervisée qui permet de détecter et caractériser les anomalies réseaux de façon autonome. Notre approche utilise des techniques de partitionnement robuste afin d'identifier les flux anormaux et construire les signatures correspondantes qui peuvent alors être déployées dans des outils classiques de sécurité. Les techniques de partitionnement utilisées sont parallélisables ce qui permet d'envisager un fonctionnement temps-réel. Nous évaluons les performances de notre système sur des traces de trafic réel. Les résultats obtenus mettent en évidence la possibilité de mettre en place des systèmes de détection et caractérisation d'anomalies autonomes et fonctionnant sans connaissance préalable.*

ABSTRACT. *The unsupervised detection of network anomalies represents an extremely challenging goal. Current methods rely on either very specialized signatures of previously seen anomalies, or on expensive and difficult to produce labeled traffic data-sets for profiling and training. We think that a modern anomaly detection system needs to be able to autonomously cope with the changing nature of both network traffic and anomalies and avoid any human involvement. In this paper we present a completely unsupervised approach to detect anomalies, without relying on signatures, or training on labeled traffic. The method uses robust clustering techniques to detect anomalous traffic flows. The structure of the anomaly identified by the clustering algorithms is used to construct signatures which can be exported towards standard security devices. The clustering algorithms are highly adapted for parallel computation, which permits the unsupervised detection and characterization in an on-line basis. We evaluate the performance of this new approach using real network traces. Results show that knowledge-independent autonomous detection and characterization of network anomalies is possible.*

MOTS-CLÉS : *Détection et caractérisation non-supervisée d'anomalies, Partitionnement robuste, Génération automatique de signatures, Sécurité autonome.*

KEY WORDS: *Unsupervised Anomaly Detection & Characterization, Robust & Distributed Clustering, Automatic Generation of Signatures, Autonomous Security.*

1. Introduction

La détection d'anomalies dans l'internet actuel est une tâche aussi compliquée que fastidieuse. Les attaques de déni de service (Denial of Service, DoS), de déni de service distribué (Distributed Denial of Service, DDoS) et les scans menacent l'intégrité et le fonctionnement quotidien du réseau. La principale difficulté liée à la détection et à l'analyse de ces différentes anomalies dans les réseaux est leur capacité à évoluer et à s'amplifier [HUN 03].

Deux approches existent dans la littérature et dans les outils existants : la détection par signature et la détection par apprentissage. Les systèmes basés sur des signatures sont très efficaces pour détecter les anomalies dont les signatures sont connues. Cependant, ils ne peuvent défendre les réseaux contre des anomalies inconnues. De plus, la construction et la mise à jour des signatures sont coûteuses et fastidieuses car nécessitant une intervention humaine. La détection par apprentissage utilise un trafic dépourvu d'anomalie afin de construire un modèle de trafic normal. Ce modèle permet alors d'isoler les anomalies en tant qu'instances déviant du modèle précédemment construit. Ce type de détection est capable de mettre en évidence de nouvelles anomalies. Cependant, la construction du profil de trafic normal peut être compliquée et elle suppose que l'on ait accès à du trafic dépourvu d'anomalie. De plus, l'évolution continue des caractéristiques du trafic complique la mise à jour du profil de trafic normal. Ces deux méthodes - construction de signature et apprentissage d'un modèle de trafic normal - sont donc coûteuses et lentes du fait de l'intervention humaine dans leur processus et de la nature évolutive de toutes les composantes du trafic réseau (normales et anormales).

L'autonomie du processus de détection et de caractérisation nous semble cruciale pour la mise en place de systèmes de détection d'anomalies faciles à déployer et utiliser. Les systèmes de détection d'anomalies modernes doivent donc proposer des capacités de détection et de caractérisation ne nécessitant pas d'intervention humaine et capables de s'adapter automatiquement à l'évolution de toutes les composantes du trafic, normales et anormales. Nous proposons pour cela d'utiliser des techniques d'apprentissage non-supervisées ne requérant aucune intervention humaine.

Dans cet article, nous présentons une méthode non-supervisée pour détecter et caractériser les anomalies réseaux. Cette méthode n'utilise ni signature, ni apprentissage, ni trafic documenté. Notre approche est basée sur un *clustering* (partitionnement) robuste pour détecter des anomalies connues et inconnues et construire des signatures associées, aisément compréhensibles, le tout en temps réel. L'algorithme de détection et caractérisation est constitué de 3 phases. La première phase consiste à détecter une fenêtre temporelle dans laquelle une anomalie pourrait être cachée. Cette détection est basé sur l'analyse de séries temporelles [BAR 02, BRU 00, KRI 03, SOU 05, COR 05]. Toute anomalie détectée lors cette phase est transmise à l'algorithme de détection non supervisées qui constitue la seconde étape. Cet algorithme utilise en entrée l'ensemble des flux capturés dans l'intervalle anormal. Notre méthode utilise une technique de partitionnement robuste basée sur le *sub-space clustering* (partitionnement en sous-espace) [PAR 04], le *density-based clustering* (partitionnement basé sur la densité) [EST 96], et l'*evidence accumulation* (accumulation de preuve) [FRE 05] pour extraire les flux suspects qui composent l'anomalie. Lors de la troisième phase de l'algorithme, le résultat de la deuxième phase est utilisé afin de construire des règles qui caractérisent l'anomalie constatée et simplifie son analyse. La caractérisation d'une anomalie peut être en effet très compliquée et fastidieuse surtout si l'on considère des anomalies inconnues. Pour régler ce problème, les règles obtenues les plus pertinentes sont combinées afin de construire une nouvelle signature

qui caractérise l'anomalie de façon simple et concise. Cette signature peut ensuite être utilisée dans des outils de sécurité standards, ce qui constitue un progrès majeur vers l'apparition de systèmes de détection d'anomalies autonomes. Pour résumer, notre algorithme produit automatiquement de nouvelles signatures sans connaissance préalable sur le trafic ou l'anomalie considérée.

L'article est organisé de la façon suivante. La section 2 présente un court état de l'art des techniques de détection d'anomalies et nos contributions principales. La section 3 décrit l'algorithme de détection non-supervisé que nous avons développé. La section 4 présente l'algorithme de génération automatique de règles qui permet de construire des signatures simples et concises pour les anomalies détectées. La section 5 présente la validation de notre proposition via la découverte et la caractérisation d'anomalies dans des traces de trafic réel issues d'un cœur de réseau. La section 6 évalue le temps de calcul de l'approche non-supervisée en prenant en compte la parallélisation de l'algorithme de *clustering*. Enfin, la section 7 conclut ce travail.

2. État de l'art et contributions

La problématique de la détection d'anomalies a donné lieu a un grand nombre de publications au cours de la dernière décennie. La plupart des approches existantes analysent les variations statistiques de métriques simples (e.g. : nombre de paquets, d'octets, ou de flux) et/ou d'autres attributs du trafic (e.g. : distribution des adresses IP et des ports) et ce, sur des données issues d'un seul lien ou de l'ensemble d'un réseau. Parmi les outils utilisés, nous pouvons citer les techniques d'analyse du signal (e.g. : Auto-Regressive Integrated Moving Average, ondelettes) sur des données issues d'un seul lien [BAR 02, BRU 00], la PCA [LAK 04, LAK 05] et les filtres de Kalman [SOU 05] pour la détection sur l'ensemble d'un réseau et les sketches appliqués aux flux IP [KRI 03, DEW 07].

Notre approche peut être classifiée dans le champ de la détection non-supervisée. La plupart des travaux existants utilisant des techniques non-supervisées ont traité le domaine de la détection d'intrusions à travers la base de données KDD'99. La plupart des méthodes de détection non-supervisées utilisent le *clustering* et la détection d'outliers (points isolés) [POR 01, ESK 02, LEU 05]. Dans [POR 01], les auteurs utilisent du *single-linkage hierarchical clustering* pour partitionner les données KDD'99 en utilisant une distance euclidienne. [ESK 02] présente des résultats améliorés sur les mêmes données obtenues avec trois algorithmes de partitionnement : un *fixed-width clustering*, une version optimisée de k-NN, et un one class *support vector machine* (machine à vecteurs de support). [LEU 05] utilise un algorithme de *density-grid-based clustering* (partionnement basé sur la densité et sur les grilles) pour améliorer l'extensibilité du *clustering* et obtient des résultats équivalents.

Notre algorithme non-supervisé présente plusieurs avantages par rapport à l'ensemble des travaux existants : (i) il fonctionne d'une manière complètement non-supervisée, ce qui signifie qu'il peut être associé à n'importe quel système de monitoring et être utilisé directement, sans configuration ou connaissance préalable. (ii) Il combine des techniques de *clustering* robuste afin d'éviter les problèmes habituels des algorithmes de partitionnement, e.g. : la sensibilité à l'initialisation, l'indication préalable du nombre de *clusters* ou la perturbation des résultats par l'utilisation d'attributs non pertinents. (iii) Il construit automatiquement des signatures simples et concises qui caractérisent les anomalies qui peuvent alors être utilisées dans un outil de sécurité. (iv) Il est conçu pour fonctionner en temps-réel en exploitant le parallélisme de notre approche de *clustering*.

3. Détection non-supervisée des anomalies

La première phase de l'analyse est la détection de changement. Pour ce faire, le trafic est agrégés suivants 9 niveaux l_i : *adresse IP source* ($l_1 : \mathrm{IPsrc}$), *adresse IP destination* ($l_2 : \mathrm{IPdst}$), *préfixe réseau source* ($l_{3,4,5} : \mathrm{IPsrc}/24, /16, /8$), *préfixe réseau destination* ($l_{6,7,8} : \mathrm{IPdst}/24, /16, /8$), et *trafic par fenêtre* ($l_9 : \mathrm{tpTS}$). Des séries temporelles $Z_t^{l_i}$ sont alors construites sur des métriques simples (le nombre de paquets, le nombre d'octets et le nombre de nouveaux flux) et ce, en considérant les 9 niveaux d'agrégation $l_{1...9}$. Un algorithme de détection d'anomalies $\mathcal{F}(.)$ basé sur l'analyse de séries temporelles [BAR 02, BRU 00, KRI 03, SOU 05, COR 05] est utilisé sur les $Z_t^{l_i}$ afin d'identifier une fenêtre anormale. Une fenêtre t_j est déclarée anormale si $\mathcal{F}(Z_{t_j}^{l_i})$ déclenche une alarme pour n'importe lequel des l_i niveaux d'agrégation. Repérer les anomalies à différents niveaux d'agrégation augmente la fiabilité de la détection et permet de détecter des anomalies composées d'un seul couple source-destination mais aussi des anomalies distribuées d'intensités variables.

La deuxième étape de l'algorithme consiste à détecter et caractériser de manière non-supervisée les flux anormaux. L'algorithme utilise pour cela l'ensemble $\mathbf{Y} = \{\mathbf{y}_1, .., \mathbf{y}_F\}$ des F flux capturés dans l' intervalle anormal, agrégés suivant l'un des niveaux d'agrégation l_i utilisés dans la première phase. Chaque flux $\mathbf{y}_f \in \mathbf{Y}$ est décrit par un ensemble de A attributs de trafic sur lesquels l'analyse est faite. $\mathbf{x}_f = (x_f(1), .., x_f(A)) \in \mathbb{R}^A$ est le vecteur des attributs décrivant le flux de trafic $\mathbf{y}_f$, et $\mathbf{X} = \{\mathbf{x}_1; ..; \mathbf{x}_F\}$ la matrice des attributs appelée aussi espace d'attributs. Les $A = 9$ attributs utilisés sont : nombre d'adresses IP source/destination et nombre de ports sources et destination, ratio du nombre de sources au nombre de destinations, nombre de paquets par seconde, ratio du nombre de paquets au nombre de destinations, et proportion de paquets ICMP et SYN. La sélection de ces attributs est une question clé de n'importe quel algorithme de détection d'anomalies. Dans cet article, nous limitons le champ de notre détection à des anomalies connues, nous utilisons donc quelques attributs déjà utilisés dans d'autres travaux pour détecter plusieurs types d'anomalies connues comme les DoS, DDoS, scan et vers en cours de propagation [FER 09]. Le choix des attributs n'a cependant pas d'influence sur la caractérisation d'un type d'anomalie. En effet, c'est la troisième phase d'analyse, dite de caractérisation (cf section 4), qui détermine la pertinence de chaque attributs en fonction de l'anomalie détectée. La limitation du nombre d'attributs est de plus tout à fait surmontable via une augmentation de A. Cela permet alors une détection et une caractérisation plus fiables.

Une fois l'espace d'attributs construit, un algorithme de partitionnement est utilisé afin d'identifier les flux anormaux au sein de $\mathbf{Y}$. Pour cela, nous faisons l'hypothèse que l'anomalie est contenue dans un petit nombre de flux. Ainsi, une anomalie peut apparaître comme un outlier (i.e. un flux isolé) ou un *cluster* de petite taille (quelques flux similaires), et ce, selon le type d'agrégation et le masque réseau utilisés. Le tableau 1 détaille les caractéristiques de différentes anomalies : nature distribuée, type d'agrégation et masque de réseau utilisés et signature construite. Une anomalie de type SYN DDos (SYN Distributed Denial of Service ou attaque de déni de service distribuée) qui vise une seule cible depuis un grand nombre d'adresses IP situées dans plusieurs domaines /24 formera un *cluster* si le trafic est agrégé avec une IP source. En effet, chacun des domaines /24 représentera un flux avec des valeurs d'attributs très éloignés de celles de la majorité du trafic : un nombre important de paquets envoyés, une seule destination et une très forte proportion de paquet SYN. C'est l'ensemble de ces flux qui formera un *cluster*. Si le trafic est agrégé via l'adresse IP destination, quelque soit le masque, l'unique adresse destination (i.e. la cible) formera un outlier qui se caractérisera notamment par un grand nombre de sources et une forte proportion de paquets SYN.

Tableau 1. *Attributs utilisés pour la détection de DoS, DDoS, balayage de ports et de réseaux (port scan and network scan), et propagation de vers. (NB : les anomalies de nature distribuée impliquent ici plusieurs @IP/24, elles-mêmes contenues dans une seule @IP/16)*

Exemple d'anomalie	Nature distribuée	Type d'agrégat. /masque réseau	Résultat de clustering	Impact sur les attributs de trafic
DoS (ICMP∨SYN)	1-vers-1	$\mathrm{IPsrc}/*$	Outlier	$\mathrm{nSrcs} = \mathrm{nDsts} = 1, \mathrm{nPkts/sec} > \lambda_1, \mathrm{avgPktsSize} < \lambda_2,$
		$\mathrm{IPdst}/*$	Outlier	$(\mathrm{nICMP/nPkts} > \lambda_3 \vee \mathrm{nSYN/nPkts} > \lambda_4)$.
DDoS (ICMP ∨ SYN) depuis plus. @IP/24	N-vers-1	$\mathrm{IPsrc}/24$ (l_3)	Cluster	$\mathrm{nDsts} = 1, \mathrm{nSrcs} > \alpha_1, \mathrm{nPkts/sec} > \alpha_2,$
		$\mathrm{IPsrc}/16$ (l_4)	Outlier	$\mathrm{avgPktsSize} < \alpha_3,$
		$\mathrm{IPdst}/*$	Outlier	$(\mathrm{nICMP/nPkts} > \alpha_4 \vee \mathrm{nSYN/nPkts} > \alpha_5)$.
Port scan	1-vers-1	$\mathrm{IPsrc}/*$	Outlier	$\mathrm{nSrcs} = \mathrm{nDsts} = 1, \mathrm{nDstPorts} > \beta_1,$
		$\mathrm{IPdst}/*$	Outlier	$\mathrm{avgPktsSize} < \beta_2, \mathrm{nSYN/nPkts} > \beta_3.$
Network scan vers plus. @IP/24	1-vers-1	$\mathrm{IPsrc}/*$	Outlier	$\mathrm{nSrcs} = 1, \mathrm{nDsts} > \delta_1,$
		$\mathrm{IPdst}/24$ (l_6)	Cluster	$\mathrm{nDstPorts} > \delta_2, \mathrm{avgPktsSize} < \delta_3,$
		$\mathrm{IPdst}/16$ (l_7)	Outlier	$\mathrm{nSYN/nPkts} > \delta_4.$
Propagation de ver sur plus. @IP/24	1-vers-N	$\mathrm{IPsrc}/*$	Outlier	$\mathrm{nSrcs} = 1, \mathrm{nDsts} > \eta_1, \mathrm{nDstPorts} < \eta_2,$ $\mathrm{avgPktsSize} < \eta_3, \mathrm{nSYN/nPkts} > \eta_4.$
		$\mathrm{IPdst}/24$ (l_6)	Cluster	
		$\mathrm{IPdst}/16$ (l_7)	Outlier	

L'algorithme de partitionnement est appliqué à $\mathbf{X}$ afin de grouper certains éléments de $\mathbf{Y}$ en sous-ensembles homogènes dont les flux sont similaires entre eux selon un indice choisi. Il existe un grand nombre d'algorithmes de partitionnement, mais ces algorithmes ne sont pas toujours capables d'identifier toutes les tailles et formes de *clusters* possibles. De plus, ces algorithmes peuvent être sensibles à la valeur des paramètres utilisés pour l'initialisation (e.g. : positions des centres de *cluster* dans l'algorithme k-means) et aux paramètres utilisés (e.g. : nombre de *clusters* dans l'algorithme k-means). Tous ces aspects illustrent le problème de manque de robustesse des algorithmes de partitionnement.

Pour éviter ces problèmes, nous avons développé une approche de *clustering* de type « division et conquête » qui se base les notions de *clustering ensemble* [STR 03] et de *clustering combination*. Cette approche combine le *density-based clustering* [EST 96], le *sub-space clustering* (SSC)[PAR 04] et l'*evidence accumulation* (EA) [FRE 05]. Les détails de l'algorithme sont exposés dans [CAS 10]. Nous présentons ici l'idée générale de l'approche.

Au lieu de directement partitionner l'espace des attributs $\mathbf{X}$ en entier en utilisant une distance classique comme la distance euclidienne, l'algorithme partitionne N sous-espaces différents $\mathbf{X}_n \subset \mathbf{X}$, ce qui permet ainsi d'obtenir N résultats de partitionnement P_n des flux de $\mathbf{Y}$. Chaque sous-espace $\mathbf{X}_n$ est construit à partir de $R < A$ attributs. Cela permet d'analyser la structure de $\mathbf{X}$ depuis R-parmi-A différents point de vue tout en utilisant une bien meilleure résolution. Pour déterminer le nombre R de dimensions utilisées par sous-espace, nous utilisons la propriété monotone des ensembles de résultats de partitionnement appelée *downard closure property* : "si un ensemble de points forme un *cluster* dans un espace de dimension d, alors, ce même ensemble de points forme aussi un *cluster* dans chacune des projections de dimension $d-1$ de ce même espace". Cela implique que, s'il existe des *clusters* dans $\mathbf{X}$, alors, ils existeront dans des sous-espaces de dimensions inférieures. L'utilisation de petites valeurs de e (ou R dans notre cas) présente alors plusieurs avantages : premièrement, le partitionnement est plus rapide dans des espaces de faibles dimensions. Deuxièmement, les algorithmes de partitionnement basés sur la densité sont plus efficaces dans des espaces de dimensions réduites [AGR 98]. Enfin, les résultats de partitionnement obtenus sont plus faciles à visualiser. Cet ensemble de contraintes nous a amené à choisir un nombre R de dimensions utilisées égal à 2, et ainsi $N(m) = C_R^A = A(A-1)/2$.

L'information obtenue entre les différents résultats de partitionnement construits pour $\mathbf{X}_n$ est ensuite combinée afin de produire une nouvelle mesure de similarité entre les flux de $\mathbf{Y}$. Ces valeurs sont stockées dans une matrice S de taille $F \times F$ qui sert à détecter les petits *clusters* et un vecteur D qui est utilisé pour classer les outliers. L'élément $S(o, p)$ représente le degré de similitude entre les flux o et p. Cette valeur reflète le nombre de fois où les flux o et p considérés sont dans le même *cluster*. Elle tient aussi compte de la taille du *cluster* en question afin de privilégier les petits *clusters*. L'élément $D(d)$ reflète quant à lui l'anormalité d'un outlier. Cette anormalité tiens compte du nombre de fois où le flux a été classé outlier et de la séparation entre cet outlier et le reste du trafic. Cela permet de séparer les *outliers* de flux et les *clusters* de flux simultanément identifiés dans différents sous-espaces du reste du trafic. Ces nouvelles mesures de similarité permettent donc d'extraire les flux anormaux de l'ensemble du trafic. En d'autres termes, si nous pouvons extraire des flux qui se démarquent de la majorité du trafic, alors, ces flux forment une anomalie, sinon, cela signifie que l'alarme sur la fenêtre temporelle considérée $Z_{t_j}^{l_i}$ était une fausse alarme.

4. Caractérisation automatique des anomalies

A la fin de l'étape précédente, l'algorithme non supervisé a identifié un ou plusieurs flux similaires dans $\mathbf{Y}$ et éloignés de la majorité des flux du trafic. La tâche suivante consiste à produire automatiquement un ensemble de règles de filtrage pour isoler et caractériser ce(s) flux. Ces règles de filtrage permettent deux choses : obtenir une image précise de la nature de l'anomalie pour faciliter l'analyse par l'opérateur réseau et construire une signature de l'anomalie par combinaison de règles. La signature ainsi produite peut ensuite être utilisée pour détecter l'anomalie via un très classique système de détection par signature. Pour produire les règles de filtrage, l'algorithme conserve les sous-espaces $\mathbf{X}_n$ dans lesquels le ou les flux anormaux sont éloignés du reste du trafic. Nous définissons deux classes de filtrage différentes : les règles absolues notées $AFR(\mathbf{Y})$ et les règles relatives notées $RFR(\mathbf{Y})$. Les règles absolues ne dépendent pas de la séparation entre *clusters*, et correspondent à la présence de valeurs dominantes pour les attributs des flux anormaux. Une règle absolue sur l'attribut a qui caractérise un ensemble de flux $\mathbf{Y}_g \subset \mathbf{Y}$ avec $\mathbf{Y}_g$ contenant une anomalie est de la forme $AFR_a(\mathbf{Y}_g) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(a) == \lambda\}$. Par exemple, dans le cas d'une attaque ICMP flooding, la majorité des flux comportent uniquement des paquets ICMP. La règle de filtrage absolue $\{\mathrm{nICMP/nPkts} == 1\}$ s'applique. Les règles de filtrage relatives dépendent de la séparation entre les flux normaux et anormaux. Si les flux anormaux sont séparés du reste du trafic dans un résultat de partitionnement P_n, alors les attributs du sous-espace correspondant $\mathbf{X}_n$ peuvent être utilisés pour définir une règle relative. Une règle relative sur l'attribut a qui caractérise un ensemble de flux $\mathbf{Y}_g \subset \mathbf{Y}$ avec $\mathbf{Y}_g$ contenant une anomalie est de la forme $RFR_a(\mathbf{Y}_g) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(a) < \lambda || x_f(a) > \lambda\}$. Une relation de couverture entre deux règles relatives est définie dans le cas où deux règles énoncent une relation sur le même attribut et selon le même position par rapport au seuil (inférieure ou supérieure). E.g. : soient les règle $f_a(\mathbf{Y}_g) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(a) < \lambda_1\}$ et $f_a(\mathbf{Y}_g) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(a) < \lambda_2\}$, la règle $f_a(\mathbf{Y}_g)$ couvre la règle $f_a(\mathbf{Y}_g)$ si et seulement si $\lambda_1 < \lambda_2$. Si deux règles ou plus se couvrent, l'algorithme conserve celle qui couvre les autres. Pour construire une signature concise de l'anomalie, la règle de filtrage la plus discriminante est choisie. Les règles absolues sont importantes, car elles définissent des caractéristiques inhérentes aux anomalies. Les règles relatives ont un intérêt qui dépend du degré de séparation des flux anormaux entre eux. Dans le cas d'outliers, nous choisissons

les K attributs pour lesquels la distance normalisée au trafic normal (représenté par le *cluster* le plus grand dans chaque sous-espace) est parmi les K distances les plus importantes. Dans le cas de *clusters* de petites tailles, le degré de séparation avec le reste des *clusters* est ordonné en utilisant le score de Fisher (FS) [JAA 99], et les K règles les mieux classées sont sélectionnées. Le score de Fisher mesure la séparation entre *clusters*, en fonction de la variance totale à l'intérieur de chaque *cluster*. Enfin, pour construire une signature, les règles absolues et les K principales règles relatives sont combinées en un seul prédicat, en utilisant la règle de couverture en cas de superposition. Cet ensemble de méthodes pour construire les règles et composer une signature permet au système de caractériser de façon efficace les anomalies détectées et de fournir une mise en forme simple, concise et facile à comprendre.

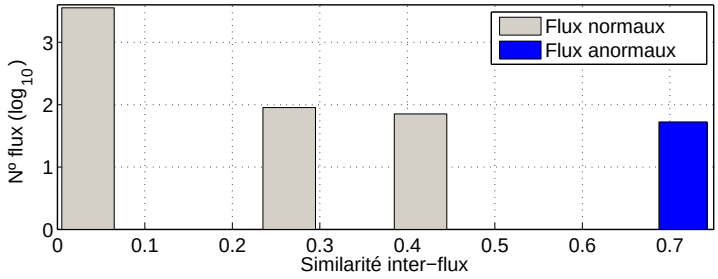
5. Evaluation experimentale

Nous évaluons ici la capacité de notre algorithme non supervisé à détecter et caractériser plusieurs anomalies contenues dans des traces de trafic réel de la base de traces publique du projet WIDE [CHO 00]. Le réseau opérationnel WIDE interconnecte des institutions de recherche au Japon, des fournisseurs d'accès internet et des universités aux Etats-Unis. Cette base contient des traces de paquets de 15 minutes collectées depuis 1999 et documentées de manière parcellaire [FON 10].

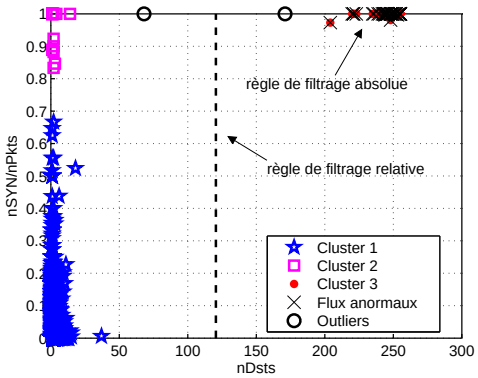
Tout d'abord, nous avons analysé manuellement une trace WIDE choisie arbitrairement. Nous avons détecté et caractérisé un scan réseau de type SYN en direction de plusieurs machines appartenant au même réseau $/16$. Les paquets sont agrégés en flux $\mathrm{IPdst}/24$ dans $\mathbf{Y}$. La longueur de la fenêtre temporelle est ΔT=20 secondes. Comme nous l'avons expliqué dans la section 3, l'algorithme SSC-EA construit alors une mesure de similarité entre les flux contenus dans $\mathbf{Y}$. La figure 1.(a) montre un histogramme sur les valeurs de S. La structure des flux fournie par S met en évidence la présence d'un petit *cluster* isolé dans plusieurs sous-espaces. Sélectionner les flux les plus similaires selon S permet d'obtenir un *cluster* très compact de 53 flux ; une analyse supplémentaire de ces flux révèle plusieurs sous-flux de paquets SYN avec la même adresse source, correspondant à une machine initiant un scan.

Si on s'intéresse aux règles de filtrage et à la signature associée à l'anomalie, les figures 1.(b, c) montrent certains résultats de *clustering* P_n pour lesquels les règles absolues et les K règles relatives ont été produites. L'anomalie est ici représentée par un *cluster*. Les règles sont construites sur les attributs suivants : nombre de sources et de destinations et proportion de paquets SYN. En les combinant, on obtient la signature $(\mathrm{nSrcs} == 1) \wedge (\mathrm{nDsts} > \lambda_1) \wedge (\mathrm{nSYN}/\mathrm{nPkts} > \lambda_2)$ où λ_1 et λ_2 sont deux seuils obtenus en séparant les *clusters* à mi distance de la moyenne de chaque *cluster* pour l'attribut considéré. L'utilisation de la moyenne permet d'avoir une approximation des valeurs du *cluster* pour l'attribut en question. Cette signature et le résultat du partitionnement sont parfaitement cohérents avec ce qu'annonce le tableau 1 : le scan réseau forme un *cluster*, il émet des paquets d'une source unique vers un grand nombre de cibles et la majorité des paquets sont de type SYN. L'immense nouveauté avec cette approche se situe au niveau de la génération de cette signature sans aucune connaissance préalable sur l'anomalie et le trafic de fond.

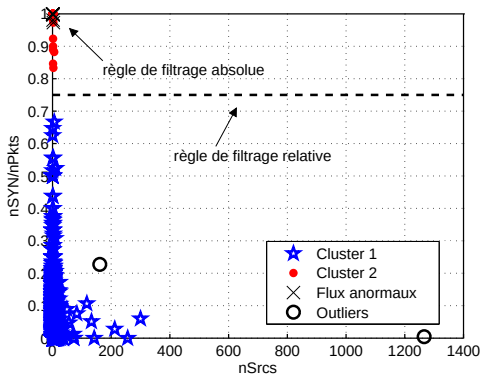
Les figures 1.(d,e) décrivent les règles obtenues lors de la détection d'une anomalie de type SYN DDoS. Les flux IP sont maintenant agrégés selon les IPsrc. L'analyse de la distribution de la simi-



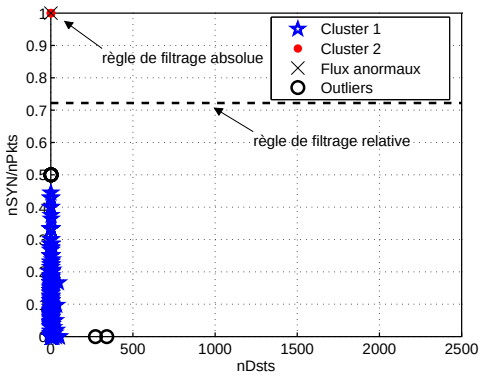
(a) Distribution des valeurs de S pour la détection d'un SYN network scan.



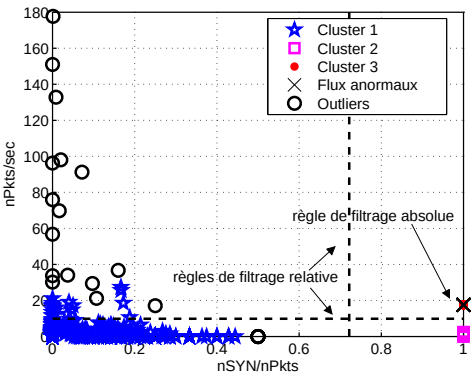
(b) SYN Network Scan (1/2)



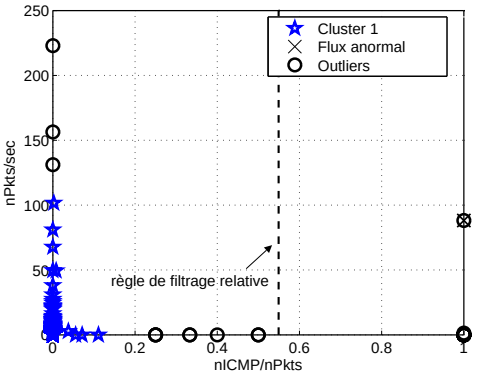
(c) SYN Network Scan (2/2)



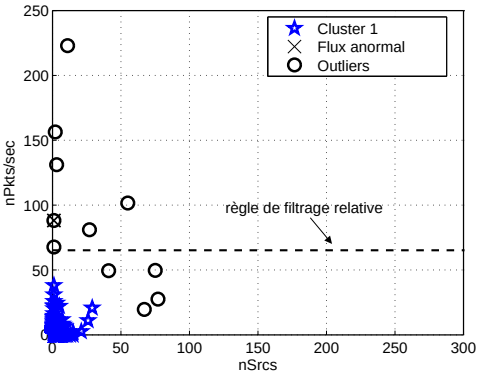
(d) SYN DDoS (1/2)



(e) SYN DDoS (2/2)



(f) ICMP flooding DoS (1/2)



(g) ICMP flooding DoS (2/2)

Figure 1. *Règles de filtrage pour la caractérisation des attaques de la trace WIDE.*

larité inter-flux contenue dans S révèle un petit *cluster* correspondant à l'ensemble des machines attaquantes. Comme pour l'anomalie précédente, la présence de ce *cluster* est cohérente par rapport à ce que spécifie le tableau 1 pour un SYN DDoS. La signature obtenue est $(\mathrm{nDsts} == 1) \wedge (\mathrm{nSYN}/\mathrm{nPkts} > \lambda_3) \wedge (\mathrm{nPkts}/\mathrm{sec} > \lambda_4)$. Le grand nombre de sources observées $(\mathrm{nSrcs} > \lambda_5)$ confirme la nature distribuée de l'anomalie constatée : un SYN DDoS.

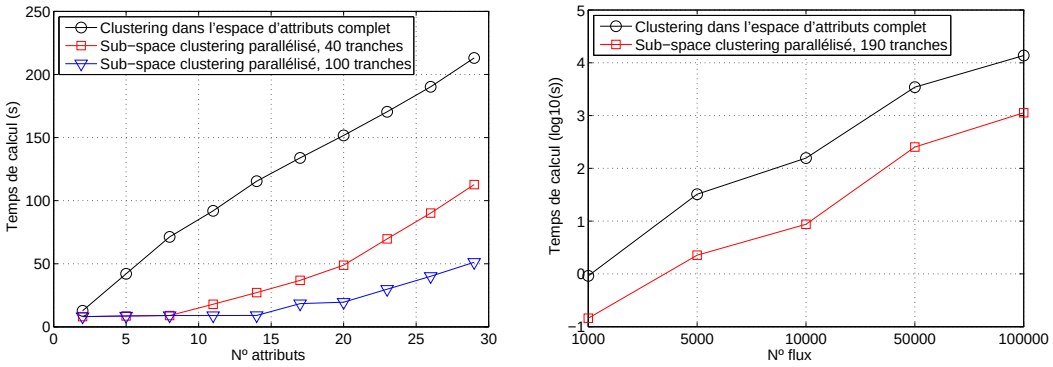
Les figures 1.(f,g) montrent la détection d'une anomalie de type DoS ICMP. Le trafic est agrégé selon les flux IPdst ; l'anomalie détectée l'est donc en tant qu'outlier et non via un *cluster* de petite taille. Le tableau 1 associe une anomalie de type DoS ICMP avec un outlier, ce qui est cohérent avec le résultat obtenu. Les règles absolues ne sont pas construites pour les outliers. Les règles relatives correspondent quant à elles à la séparation entre l'outlier et le plus gros *cluster* dans chaque sous-espace qui représente le trafic normal. En plus de mettre en évidence les caractéristiques de l'anomalie considérée, comme le grand nombre de paquets ICMP par seconde issus de la même source, le résultat du *clustering* montre que l'anomalie détectée n'englobe pas les flux éléphants de la fenêtre temporelle considérée. Cela met en évidence la capacité de l'algorithme à détecter des anomalies dont certaines caractéristiques sont proches de celles du trafic normal en termes de volume mais éloignées au niveau d'autres caractéristiques. La signature obtenue est ici $(\mathrm{nICMP}/\mathrm{nPkts} > \lambda_6) \wedge (\mathrm{nPkts}/\mathrm{sec} > \lambda_7)$.

Une évaluation supplémentaire peut être trouvée dans [CAS 10]. Cette évaluation contient une évaluation du taux de vrais/faux positifs et une comparaison avec d'autres méthodes d'apprentissage non-supervisé. Ces résultats confirment la supériorité de notre approche dans le domaine de la détection d'anomalies non-supervisée.

6. Temps de calcul et parallélisation

Le dernier sujet que nous abordons est le temps de calcul (TC) de l'algorithme. Notre algorithme réalise plusieurs partitionnement sur $N(m)$ sous-espaces $\mathbf{X}_n \subset \mathbf{X}$ de petite dimension. Ces multiples calculs posent le problème de l'extensibilité pour une détection temps-réel sur des cœurs de réseaux à très hauts débits. Deux caractéristiques de notre algorithme sont utilisées pour améliorer ces problèmes d'extensibilité vis-à-vis du nombre d'attributs m et du nombre de flux agrégés n à analyser. D'une part, le *clustering* est réalisé dans des sous-espaces de très petites dimensions, $\mathbf{X}_n \in \mathbb{R}^R$ avec ici $R = 2$ (cf. section 3), ce qui est plus rapide qu'un *clustering* réalisé dans un espace de grande dimension [JAI 10]. D'autre part, chaque sous-espace peut être partitionné indépendamment des autres sous-espaces, ce qui rend l'algorithme de *subspace* clustering particulièrement adapté à la parallélisation. Cette parallélisation peut être réalisée de différentes façons : via une machine multi-cœurs et/ou multi-processeurs, via une ou des carte(s) réseaux avec processeurs de traitement intégré, via un ou des GPU(s) (Graphic Processor Unit), via plusieurs machines ou via une combinaison de n'importe lesquelles de ces techniques. Nous utiliserons désormais le terme tranche pour nommer une entité de calcul.

La figure 2 représente le temps de calcul de notre algorithme, en fonction du nombre d'attributs m utilisés pour décrire les flux (a) et en fonction du nombre n de flux à analyser (b). La figure 2.(a) compare le temps de calcul obtenu lorsque l'on partitionne l'espace d'attributs $\mathbf{X}$, noté $\mathrm{TC}(\mathbf{X})$ avec le temps de calcul nécessaire au *sub-space clustering*, en faisant varier A de 2 à 29 attributs. Un grand



(a) Temps vs. n attributs. (b) Temps vs. n flux.

Figure 2. *Temps de calcul en fonction du nombre d'attributs et de flux à analyser. Le nombre de flux dans (a) est* $n = 10000$*. Le nombre d'attributs et de tranches pour (b) est* $m = 20$ *et* $M = 190$*.*

nombre de flux N, ici 10^4, est analysé en utilisant deux nombres de tranches $T = 40$ et $T = 100$. Pour estimer le temps de calcul de notre algorithme de *sub-space clustering* avec des valeurs données de m et T, nous procédons de la façon suivante : premièrement, nous partitionnons chacun des $N = m(m-1)/2$ sous-espaces $\mathbf{X}_n$ et le plus mauvais résultat en terme de temps de calcul pour un seul sous-espace $\text{TC}(\mathbf{X}_{\text{SSCwc}}) = \max_n \text{TC}(\mathbf{X}_n)$ est conservé en tant que référence. Si $N \leqslant T$, le nombre de tranches couvre la totalité des partitionnements à réaliser, l'algorithme va donc s'exécuter de manière complètement parallèle. Le temps de calcul total correspond au plus mauvais résultat $\text{TC}(\mathbf{X}_{\text{SSCwc}})$. Par contre, si $N > T$, certaines tranches doivent partitionner plusieurs sous-espaces, le temps de calcul devient égal à $(N\%T + 1)$ fois le plus mauvais cas $\text{TC}(\mathbf{X}_{\text{SSCwc}})$, où % représente la division entière. La première observation intéressante sur la figure 2.(a) concerne l'augmentation de $\text{TC}(\mathbf{X})$ quand m augmente ; TC vaut 8 secondes pour $m = 2$ et plus de 200 secondes for $m = 29$. La diminution du TC pour des petites valeurs de m permet de compenser en partie le grand nombre de partitionnements réalisés. La seconde observation concerne le parallélisme : si l'algorithme est déployé dans une architecture parallèle, il peut être utilisé pour analyser de grandes quantités de trafic en utilisant un grand nombre d'attributs en temps-réel. Par exemple, l'analyse de 20 attributs sur une architecture avec 100 tranches permet de traiter 10000 flux en moins de 20 secondes.

La figure 2.(b) compare $\text{TC}(\mathbf{X})$ et $\text{TC}(\mathbf{X}_{\text{SSCwc}})$ en fonction du nombre de flux n à analyser en utilisant $m = 20$ attributs et $M = N = 190$ tranches (i.e. notre algorithme est complètement parallélisé). La différence en termes de temps de calcul entre le *clustering* dans l'espace entier et le *clustering* dans les sous-espaces est toujours présente : le gain est de plus d'un ordre de magnitude, indépendamment du nombre de flux à analyser. En ce qui concerne la quantité de trafic analysable dans une configuration complètement parallèle, l'algorithme est capable de traiter plus de 50000 flux avec un temps de calcul relativement limité : 4 minutes environ. Ces évaluations ont été réalisées avec un nombre de flux agrégés égal à environ 2500 dans une fenêtre temporelle de durée $\Delta T = 20s$, ce qui représente un temps de calcul de $\text{TC}(\mathbf{X}_{\text{SSCwc}}) \approx 0.4$ secondes. Si l'on considère les $m = 9$ attributs utilisés précédemment ($N = 36$), le temps de calcul total sans parallélisation est de $N \times \text{TC}(\mathbf{X}_{\text{SSCwc}}) \approx 14.4$ secondes.

7. Conclusion

Notre algorithme non supervisé de détection et de caractérisation d'anomalies supplante les méthodes précédemment proposées dans ce domaine, et ce, sans nécessiter de connaissance préalable des anomalies et du trafic de fond. L'algorithme proposé parvient, grâce au *sub-space clustering* et à l'*evidence accumulation*, à être plus robuste que les approches de *clustering* classiques. Il génère aussi des signatures d'anomalies de façon automatique sans aucune connaissance préalable. Ces signatures peuvent alors être traduites dans le langage des équipements de sécurité (systèmes de détection d'intrusion, firewalls, ...) et y être utilisées automatiquement. Notre algorithme peut donc être implanté à l'intérieur de systèmes de sécurité autonomes dans lequel la détection/caractérisation non-supervisée fonctionne en parallèle avec un système à base de signature afin d'identifier les événements anormaux inconnus et être capable de les détecter immédiatement après.

Enfin, et contrairement à la plupart des travaux précédents, nous avons évalué le temps de calcul de notre algorithme. Les résultats confirment que l'utilisation de notre algorithme pour la détection non-supervisée et la génération automatique de signature en temps-réel est possible pour les volumes de trafic considérés. De plus, l'évaluation montre que l'algorithme, une fois déployé sur une architecture parallèle, est utilisable sur des réseau à très hauts débits tout en utilisant un nombre accru d'attributs pour caractériser le trafic et ainsi les anomalies.

Remerciements

Ce travail a été réalisé dans le cadre du projet ECODE FP7-ICT-2007-2/223936 financé par la Commission Européenne.

8. Bibliographie

[AGR 98] AGRAWAL R., GEHRKE J., GUNOPULOS D., RAGHAVAN P., « Automatic subspace clustering of high dimensional data for data mining applications », *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, SIGMOD '98, New York, NY, USA, 1998, ACM, p. 94–105.

[BAR 02] BARFORD P., KLINE J., PLONKA D., RON A., « A signal analysis of network traffic anomalies », *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, IMW '02, New York, NY, USA, 2002, ACM, p. 71–82.

[BRU 00] BRUTLAG J. D., « Aberrant Behavior Detection in Time Series for Network Monitoring », *Proceedings of the 14th USENIX conference on System administration*, Berkeley, CA, USA, 2000, USENIX Association, p. 139–146.

[CAS 10] CASAS P., MAZEL J., OWEZARSKI P., LABIT Y., « Sub-Space Clustering and Evidence Accumulation for Unsupervised Anomaly Detection in IP Networks », Rapport technique LAAS-CNRS 10713, 2010.

[CHO 00] CHO K., MITSUYA K., KATO A., « Traffic data repository at the WIDE project », *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '00, Berkeley, CA, USA, 2000, USENIX Association, p. 51–51.

[COR 05] CORMODE G., MUTHUKRISHNAN S., « What's new : finding significant differences in network data streams », *Networking, IEEE/ACM Transactions on*, vol. 13, n° 6, 2005, p. 1219 - 1232.

[DEW 07] DEWAELE G., FUKUDA K., BORGNAT P., ABRY P., CHO K., « Extracting hidden anomalies using sketch and non Gaussian multiresolution statistical detection procedures », *Proceedings of the 2007 workshop on Large scale attack defense*, LSAD '07, New York, NY, USA, 2007, ACM, p. 145–152.

[ESK 02] ESKIN E., ARNOLD A., PRERAU M., PORTNOY L., STOLFO S., « A Geometric Framework for Unsupervised Anomaly Detection : Detecting Intrusions in Unlabeled Data », *Applications of Data Mining in Computer Security*, Kluwer, 2002.

[EST 96] ESTER M., PETER KRIEGEL H., S J., XU X., « A density-based algorithm for discovering clusters in large spatial databases with noise », AAAI Press, 1996, p. 226–231.

[FER 09] FERNANDES G., OWEZARSKI P., « Automated Classification of Network Traffic Anomalies », *Proceedings SecurComm'09*, 2009.

[FON 10] FONTUGNE R., BORGNAT P., ABRY P., FUKUDA K., « MAWILab : combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking », *Proceedings of the 6th International COnference*, Co-NEXT '10, New York, NY, USA, 2010, ACM, p. 8 :1–8 :12.

[FRE 05] FRED A. L. N., JAIN A. K., « Combining Multiple Clusterings Using Evidence Accumulation », *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, 2005, p. 835–850, IEEE Computer Society.

[HUN 03] HUNT P. R., HANSMAN S., « A Taxonomy of Network and Computer Attack Methodologies », 2003.

[JAA 99] JAAKKOLA T. S., HAUSSLER D., « Exploiting generative models in discriminative classifiers », *Proceedings of the 1998 conference on Advances in neural information processing systems II*, Cambridge, MA, USA, 1999, MIT Press, p. 487–493.

[JAI 10] JAIN A. K., « Data clustering : 50 years beyond K-means », *Pattern Recognition Letters*, vol. 31, n° 8, 2010, p. 651-666, Elsevier B.V.

[KRI 03] KRISHNAMURTHY B., SEN S., ZHANG Y., CHEN Y., « Sketch-based change detection : methods, evaluation, and applications », *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, IMC '03, New York, NY, USA, 2003, ACM, p. 234–247.

[LAK 04] LAKHINA A., CROVELLA M., DIOT C., « Diagnosing network-wide traffic anomalies », *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '04, New York, NY, USA, 2004, ACM, p. 219–230.

[LAK 05] LAKHINA A., CROVELLA M., DIOT C., « Mining anomalies using traffic feature distributions », *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '05, New York, NY, USA, 2005, ACM, p. 217–228.

[LEU 05] LEUNG K., LECKIE C., « Unsupervised anomaly detection in network intrusion detection using clusters », *Proceedings of the Twenty-eighth Australasian conference on Computer Science - Volume 38*, ACSC '05, Darlinghurst, Australia, Australia, 2005, Australian Computer Society, Inc., p. 333–342.

[PAR 04] PARSONS L., HAQUE E., LIU H., « Subspace clustering for high dimensional data : a review », *SIGKDD Explor. Newsl.*, vol. 6, 2004, p. 90–105, ACM.

[POR 01] PORTNOY L., ESKIN E., STOLFO S., « Intrusion detection with unlabeled data using clustering », *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*, 2001, p. 5–8.

[SOU 05] SOULE A., SALAMATIAN K., TAFT N., « Combining filtering and statistical methods for anomaly detection », *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, IMC '05, Berkeley, CA, USA, 2005, USENIX Association, p. 331–344.

[STR 03] STREHL A., GHOSH J., « Cluster ensembles — a knowledge reuse framework for combining multiple partitions », *J. Mach. Learn. Res.*, vol. 3, 2003, p. 583–617, JMLR.org.