Hunting attacks in the dark: clustering and correlation analysis for unsupervised anomaly detection

Johan Mazel, Pedro Casas, Romain Fontugne, Kensuke Fukuda, Philippe Owezarski

J. Mazel, P. Casas, R. Fontugne, K. Fukuda, P. Owezarski-
Hunting Attacks in the Dark
[2]National Institute of Informatics (NII), Tokyo, Japan
[3]The Telecommunications Research Center Vienna (FTW), Vienna, Austria
[4]CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse Cedex 4, France
[5]Université de Toulouse; UPS, INSA, INP, ISAE ; UT1, UTM, LAAS; F-31077 Toulouse Cedex 4, France
The Telecommunications Research Center Vienna, Donau-City-Straβe 1, A-1220 Vienna, Austria, Tel. +43 1 5052830-25. E-mail: casas@ftw.at

***Abstract*—Network anomalies and attacks represent a serious challenge to ISPs, who need to cope with an increasing number of unknown events that put their networks' integrity at risk. Most of the network anomaly detection systems proposed so far employ a supervised strategy to accomplish their task, using either signature-based detection methods or supervised-learning techniques. The former fails to detect unknown anomalies, exposing the network to severe consequences; the latter requires labeled traffic, which is difficult and expensive to produce. In this paper we introduce a powerful unsupervised approach to detect and characterize network anomalies in the dark, i.e., without relying on signatures or labeled traffic. Unsupervised detection is accomplished by means of robust clustering techniques, combining sub-space clustering with correlation analysis to blindly identify anomalies. To alleviate network operator's post-processing tasks and to speed-up the deployment of effective countermeasures, anomaly ranking and characterization are automatically performed on the detected events. The system is extensively tested with real traffic from the WIDE backbone network, spanning six years of flows captured from a trans-pacific link between Japan and the US, using the MAWILab framework for ground-truth generation. We additionally evaluate the proposed approach with synthetic data, consisting of traffic from an operational network with synthetic attacks. Finally, we compare the performance of the unsupervised detection against different previously used unsupervised detection techniques, as well as against multiple anomaly detectors used in MAWILab.**

***Index Terms*—nsupervised Anomaly Detection & Characterization, Clustering, Outliers Detection, Anomaly Correlation, Filtering Rules, MAWILab.nsupervised Anomaly Detection & Characterization, Clustering, Outliers Detection, Anomaly Correlation, Filtering Rules, MAWILab.U**

Hunting Attacks in the Dark: Clustering and Correlation Analysis for Unsupervised Anomaly Detection

Johan Mazel[2], Pedro Casas[3], Romain Fontugne[2],
Kensuke Fukuda[2], Philippe Owezarski[4][5]

I. Introduction

Network anomaly detection has become a vital component of any network in today's Internet. Ranging from non-malicious unexpected events such as flash-crowds and failures, to network attacks and intrusions such as denials-of-service, network scans, worms propagation, botnets activity, etc., network traffic anomalies can have serious detrimental effects on the performance and integrity of the network. The principal challenge in automatically detecting and characterizing traffic anomalies is that these are moving targets. It is difficult to precisely and continuously define the set of possible anomalies that may arise, especially in the case of network attacks, because new attacks as well as new variants to already known attacks are continuously emerging. A general anomaly detection system should therefore be able to detect a wide range of anomalies with diverse structures, without relying exclusively on previous knowledge and information.

The problem of network anomaly detection has been extensively studied during the last decade. Two different approaches are by far dominant in current research literature and commercial detection systems: signature-based detection and supervised-learning-based detection. Both approaches require some kind of guidance to work, hence they are generally referred to as supervised-detection approaches. Signature-based detection systems are highly effective to detect those anomalies which are programmed to alert on. When a new anomaly is discovered and clearly described by drilling down on its characteristics, the associated signature is coded by human experts, which is then used to detect a new occurrence of the same anomaly. Such a detection approach is powerful and very easy to understand, because the operator can directly relate the detected anomaly to its specific signature. However, these systems cannot defend the network against new attacks, simply because they cannot recognize what they do not know. Furthermore, building new signatures represents an expensive task, as it involves manual inspection by human experts.

On the other hand, supervised-learning-based detection uses labeled traffic data to train a baseline model for normal-operation traffic, detecting anomalies as patterns that deviate from this model. Such methods can detect new kinds of anomalies and network attacks not seen before, because they will naturally deviate from the baseline. Nevertheless, supervised-learning requires training, which is time consuming and depends on the availability of purely anomaly-free traffic data-sets. Labeling traffic as anomaly-free is expensive and hard to achieve in practice, since it is difficult to guarantee that no anomalies are hidden inside the collected traffic. Additionally, it is not easy to maintain an accurate and up-to-date model for anomaly-free traffic, particularly when new services and applications are constantly emerging.

We think that modern anomaly detection systems should not rely exclusively on previously acquired knowledge, but shall be able to autonomously detect and characterize traffic deviating from normal operation. Autonomous security is a strong requirement in current networks. Indeed, hand made analysis of anomalies and attacks is slow, inefficient, costly, and generally lets the network unprotected for several days. This work proposes an unsupervised approach to detect and characterize network anomalies without relying on signatures, training, or labeled traffic. The approach uses unsupervised machine learning techniques to discover anomalous patterns in traffic flows. More precisely, it combines robust clustering techniques with correlation analysis to detect and characterize anomalies, reducing the intervention of a human network operator. The proposed approach permits to equip routers and security components (e.g., IDS, firewall, etc.) with analysis capabilities, easing automatic and adaptive configuration, therefore providing first steps towards autonomous network security. While the ultimate goal is to come up with a fully unsupervised system, we do not claim that the proposed approach can operate without human intervention. Our system is capable of finding meaningful clusters and deriving a signature for the corresponding traffic flows based on the absolute values of selected traffic features, but it still requires the intervention of a human expert to label these clusters in a consistent way.

A first release of this unsupervised anomaly detection technique was initially presented at [1] and [2]. This initial approach proved to be highly successful in detecting isolated anomalies, identified by manual inspection on some limited traffic traces. In this paper we introduce an extended approach for unsupervised anomaly detection and characterization; in particular, we include the following specific novel contributions w.r.t. [1] and [2]: (i) the new system increases the robustness of the detection by correlating results from multiple independent analyses of the monitored traffic flows. In a nutshell, by analyzing flows using different flow-aggregations (e.g., by source IP address, by destination subnetwork, etc.), one adds redundancy to the overall traffic analysis, alleviating the potential misclassification done by a single snapshot analysis; (ii) we include an anomaly volume-based ranking approach to the overall system, which allows the operator to focus on the most important anomalies in case of multiple consecutive alarms; (iii) the evaluation of the system is extended to the analysis of a substantial subset spanning six years of the public MAWI network traffic repository at the WIDE project [3], using the MAWILab repository [4] as ground-truth. The WIDE operational network provides interconnection between different research institutions in Japan, as well as connection to different commercial ISPs and universities in the US; (iv) the detection performance is compared to the one obtained by other relevant state-of-the-art approaches, including four anomaly detectors [5]–[8] used in MAWILab and three unsupervised anomaly detectors based on clustering [9]–[11] and Principal Components Analysis (PCA) [12], [13]. This analysis is done with synthetic data, consisting of traffic from an operational network with annotated synthetic attacks added; (v) last but not least, we additionally provide an evaluation of the computational time of the core clustering process of the system, demonstrating that unsupervised traffic analysis can be done online in operational networks when appropriate distributed and scalable parallel data analysis frameworks are employed, such as Apache Hadoop[1] or Apache Spark[2] engines.

The remainder of the paper is organized as follows. Section II presents the state of the art in the supervised and unsupervised anomaly detection fields, additionally describing our main contributions. Section III describes the clustering techniques and analysis algorithms used by the unsupervised anomaly detection system. Section IV presents the anomaly post-processing techniques, designed to improve the robustness of the detection and to ease the tasks of the network operator.

Section V presents a complete evaluation and validation of the system, detecting and characterizing different anomalies on six years of real network traffic traces from the MAWI trace repository. Evaluations also include a detection performance comparison to other state-of-the-art proposals, using both MAWI traffic and traffic traces from the METROSEC project [14]. In addition, we perform a sensitivity analysis of the detection techniques and an evaluation of the computational time of the underlying clustering algorithms. Finally, Section VI presents the concluding remarks of the paper, pointing to future research directions.

II. Related Work

Traditional approaches for network anomaly detection analyze statistical variations of traffic volume metrics (e.g., number of bytes, packets, or flows) and/or other specific traffic features (e.g. distribution of IP addresses and ports), using either single-link measurements or network-wide data. A non-exhaustive list of methods includes the use of signal processing techniques (e.g., wavelets) on single-link traffic measurements [15]–[17], PCA [12], [13] and Kalman filters [18] for network-wide anomaly detection, Hough Transform [5], sketches [6], [19] or a combination of PCA, sketches [7] and equilibrium properties [20] on IP-flows, as well as traffic related distribution analysis [8]. A comprehensive summary of general anomaly detection techniques is presented in [21], and a specific survey on anomaly detection and diagnosis in Internet traffic is available at [22].

Lakhina et al. [13] revisit their PCA-based method [12] but use several entropy metrics based on source and destination IP address distributions and source and destination port distributions. They propose to reuse these entropy metrics to classify anomalies. Xu et al. [23] apply clustering to entropy metrics similar to [13] to build a traffic model and then classify anomalous events. Fernandes et al. [24] present NADA, a signature-based tool that classifies anomalies into different categories. Similarly, Silveira et al. [25] propose URCA, a method to identify the root causes of anomalous events. URCA follows a hybrid approach which relies on both signatures and supervised learning. The tool uses as input the result of any anomaly detection system, and it is able to classify anomalies by associating them with previously manually built signatures through hierarchical clustering. Using their previous work on entropy-based Traffic Entropy Spectrum (TES) anomaly detection, Tellenbach et al. [26] present the entropy telescope, which allows both to detect and classify network anomalies. Their classification scheme uses simulated anomalies and a support vector machine (SVM) model. Brownlee [27] and Glatz et al. [28] analyze one-way traffic in the context of darknet traffic.

While the majority of the main contributions to the anomaly detection field date back to almost a decade, some newer references address the detection of anomalies in Content Delivery Networks (CDN) [29], the application of empirical and estimated feature probability distribution to detect anomalies [30], the collaborative detection of network attacks [31], as well as the usage of SDN-based technology to simplify traffic processing for anomaly detection [32].

Besides detection and classification of network anomalies, there have also been several papers proposing taxonomies for network attacks. Mirkovic et al. [33] propose a classification of DDoS attacks according to several criteria (e.g., IP address spoofing, exploited weakness, etc.). Barnett et al. [34] present a taxonomy of scanning events, while Plonka et al. [35] present a taxonomy that covers a wide range of attacks. In addition, few works have been published on longitudinal studies of anomalies, meaning the analysis of long-in-time traffic traces. Borgnat et al. [36] study seven years of WIDE traffic and analyze its long range dependency (LRD). They provide an embryonic analysis of anomalies in the MAWI dataset. Allman et al. [37] study 13 years of scanning activity.

Other network anomaly detection proposals use machine learning techniques. Shon et al. [38] apply Self-Organizing Feature Map, Genetic Algorithm and Support Vector Machines (SVM) to build a profile of normal traffic and then detect anomalies as events deviating from this profile. Duffield et al. [39] demonstrate that a system using flow features and trained with Snort alarms can be as efficient as Snort itself without the expensive payload inspection. These approaches are supervised, i.e. they rely on training to build a profile, either normal [38] or abnormal [39] that is then used to find anomalies.

Our proposal falls within the unsupervised machine learning-based anomaly detection domain. Most work has been devoted to the Intrusion Detection field, focused on the well known KDD'99 data-set. The vast majority of the unsupervised detection schemes proposed in the literature are based on clustering and outliers detection, being [9]–[11] some relevant examples. In [9], authors use a single-linkage hierarchical clustering method to cluster data from the KDD'99 data-set, based on the standard Euclidean distance for inter-pattern similarity. Clusters are considered as normal-operation activity, and patterns lying outside a cluster are flagged as anomalies. Based on the same ideas, [10] reports improved results in the same data-set, using three different clustering algorithms: Fixed-Width clustering, an optimized version of k-Nearest Neighbors, and one class SVM. Finally, [11] presents a combined density-grid-based clustering algorithm to improve computational complexity, obtaining similar detection results. Some newer references include a nice overview on the usage of clustering for the unsupervised detection of anomalies [40], as well as more complex techniques based on hybrid approaches [41].

Our unsupervised algorithm presents several advantages w.r.t. current state of the art. First and most important, it works in a completely unsupervised fashion, which means that it can be directly plugged-in to any monitoring system and start to detect anomalies from scratch, without any kind of calibration. Secondly, it avoids the lack of robustness of general clustering techniques used in current unsupervised anomaly detection algorithms; in particular, it is immune to general clustering problems such as sensitivity to initialization, specification of number of clusters, or structure-masking by irrelevant features. Thirdly, it uses a new technique to correlate results from the unsupervised anomaly detection and improve its reliability, performing anomaly detection based not only on outliers detection, but also by identifying connected relevant clusters. This is achieved by exploring different levels of traffic aggregation, both at the source and destination of the traffic. Finally, it includes an anomaly post-processing step which automatically builds compact and easy-to-interpret signatures to isolate attacks, which can be directly integrated into any traditional security device.

III. Unsupervised Anomaly Detection

The proposed unsupervised anomaly detection system consists of three consecutive traffic analysis steps, working on top of small-time scale contiguous batches of data. Figure 1 depicts a

[1]http://hadoop.apache.org/

[2]https://spark.apache.org/

Anomaly	Nature	Aggregation	Clustering type	Impact on traffic features
DoS (ICMP $\vee$ SYN)	1-to-1	IPsrc/$*$	Outlier	$\text{nSrcs} = \text{nDsts} = 1, \text{nPkts/sec} > \lambda_1$, av
		IPdst/$*$	Outlier	$(\text{nICMP/nPkts} > \lambda_3 \vee \text{nSYN/nPkts} >$
DDoS (ICMP $\vee$ SYN) to several @IP/24	N-to-1	IPsrc/24 (l_3)	Cluster	$\text{nDsts} = 1, \text{nSrcs} > \alpha_1, \text{nPkts/sec} > \alpha$
		IPsrc/16 (l_4)	Outlier	$(\text{nICMP/nPkts} > \alpha_4 \vee \text{nSYN/nPkts} >$
		IPdst/$*$	Outlier	
Port scan	1-to-1	IPsrc/$*$	Outlier	$\text{nSrcs} = \text{nDsts} = 1, \text{nDstPorts} > \beta_1$, av
		IPdst/$*$	Outlier	$\text{nSYN/nPkts} > \beta_3$.
Network scan to several @IP/24	1-to-N	IPsrc/$*$	Outlier	$\text{nSrcs} = 1, \text{nDsts} > \delta_1, \text{nDstPorts} > \delta_2$
		IPdst/24 (l_6)	Cluster	$\text{nSYN/nPkts} > \delta_4$.
		IPdst/16 (l_7)	Outlier	
Spreading worms to several @IP/24	1-to-N	IPsrc/$*$	Outlier	$\text{nSrcs} = 1, \text{nDsts} > \eta_1, \text{nDstPorts} < \eta_2$
		IPdst/24 (l_6)	Cluster	$\text{nSYN/nPkts} > \eta_4$.
		IPdst/16 (l_7)	Outlier	

TABLE I: Features used for the detection of DoS, DDoS, network/port scans, and spreading worms. Anomalies of distributed nature 1-to-N or N-to-1 involve several /24 (source or destinations) addresses contained in a single /16 address.

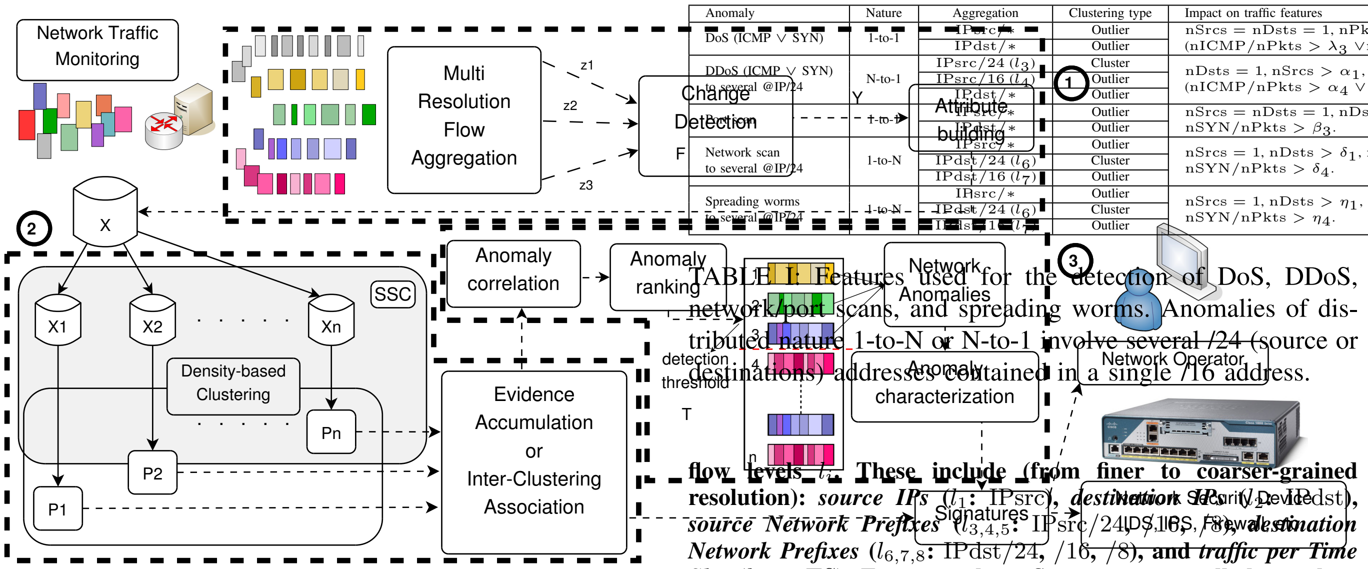


Fig. 1: High-level description of the unsupervised anomaly detection approach.

high-level diagram of the complete approach. In the first step, a batch of traffic measurements is aggregated into multi-resolution flows (e.g., all packets coming from the same IP address, from the same $/24$ subnet, etc.) to construct different views of the traffic. Several simple metrics such as number of bytes, packets or flows per batch are computed on top of the captured traffic, and a time series based abrupt change detection algorithm is applied on these metrics to detect an abrupt change, flagging an anomalous batch.

The second step consists of the unsupervised detection algorithm. It uses as input the set of flows captured in the batch flagged as anomalous by the abrupt change detector. Sub-Space Clustering (SSC) [42] techniques are applied to a set of features describing this set of flows, and results are then combined through multiple Evidence Accumulation (EA) [43] or Inter-Clustering Results Association (ICRA) to blindly identify the suspicious traffic flows that compose the anomaly responsible for the abrupt change.

In the third step, the identified flows go through a post-processing analysis to improve the robustness of the detection and to characterize the detected anomalies. Anomalous flows identified in different features are firstly correlated and merged into single sets representing the same anomalies. Then, detected anomalies are ranked according to their relevance, using different criteria to assess their potential impacts on the network. Finally, the evidence of traffic structure obtained in the clustering step is further used to produce filtering rules that characterize the most relevant detected anomalies, which are ultimately combined into a new anomaly signature. The final output of the complete post-processing analysis is a list of ranked anomalies with their corresponding signatures that provide a simple and easy-to-interpret description of the problem. This eases network operator tasks in terms of time prioritization and speed of diagnosis. The following paragraphs provide further details on each of these steps.

A. First Step: Traffic Aggregation and Abrupt Change Detection

The anomaly detection system works on single-link packet-level traffic captured in consecutive batches or time slots of fixed length Δ_T. At each time slot, packets are aggregated in 9 different flow levels l_i. These include (from finer to coarser-grained resolution): *source IPs* (l_1: IPsrc), *destination IPs* (l_2: IPdst), *source Network Prefixes* ($l_{3,4,5}$: $\text{IPsrc}/24, /16, /8$), *destination Network Prefixes* ($l_{6,7,8}$: $\text{IPdst}/24, /16, /8$), and *traffic per Time Slot* (l_9: tpTS). For example, a flow aggregates all the packets targeting the same destination IP address when flow level l_2 is considered. Time series $Z_t^{l_i}$ are built for basic traffic metrics such as number of bytes, packets, and 5-tuple flows per time slot, using the 9 flow resolutions $l_{1...9}$. Any generic anomaly-detection algorithm $\mathcal{F}(.)$ based on time series analysis [15], [16], [18], [19], [44] is then used on $Z_t^{l_i}$ to identify an anomalous slot. In our case and for the sake of simplicity, we use the *absolute deltoids* approach [44] (i.e., basically a change-detector based on mean and variance of a time series), based on volume metric time series ($\#packets$, $\#bytes$ and $\#syn$ – number of SYN packets). Time slot t_j is flagged as anomalous if $\mathcal{F}(Z_{t_j}^{l_i})$ triggers an alarm for any of the l_i flow aggregation levels. Tracking anomalies at multiple aggregation levels provides additional reliability to the anomaly detector, and permits to detect both single source-destination and distributed attacks of very different intensities.

B. Second Step: Clustering for Unsupervised Anomaly Identification

The unsupervised anomaly detection step takes as input all the flows in the time slot flagged as anomalous, aggregated according to one of the different levels used in the first stage. An anomaly will generally be detected in different aggregation levels, and there are many ways to select a particular aggregation to use in the unsupervised step; for the sake of simplicity, we shall skip this issue, and use any of the aggregation levels in which the anomaly was detected. Without loss of generality, let $\mathbf{Y} = \{\mathbf{y}_1, .., \mathbf{y}_F\}$ be the set of F flows in the flagged time slot, referred to as *patterns* in more general terms. Each flow $\mathbf{y}_f \in \mathbf{Y}$ is described by a set of A traffic attributes or *features*. We use a list of traffic features widely used in literature, which includes $A = 9$ traffic features: number of source/destination IP addresses and ports, ratio of number of sources to number of destinations, packet rate, ratio of packets to number of destinations, and fraction of ICMP and SYN packets. According to our previous work on signature-based anomaly analysis [24], such simple traffic descriptors permit to characterize general traffic anomalies in easy-to-interpret terms. The list is by no means exhaustive, and more features can be easily plugged-in to improve results. Let $\mathbf{x}_f = (x_f(1), .., x_f(A)) \in \mathbb{R}^A$ be the corresponding vector of traffic features describing flow $\mathbf{y}_f$, and $\mathbf{X} = (\mathbf{x}_1, .., \mathbf{x}_F)$ the complete matrix of features, referred to as the *feature space*.

The reader should note that, even if many of the considered features could potentially yield no useful information (at the expense of higher processing costs), it is not possible to apply standard feature selection algorithms to identify the smallest necessary set, as we are considering an unsupervised classification

problem, where labels are not available. Feature selection in unsupervised analysis is an open and very challenging problem.

The unsupervised detection algorithm is based on clustering techniques applied to $\mathbf{X}$. The objective of clustering is to partition a set of unlabeled patterns into homogeneous groups of similar characteristics, based on some measure of similarity. Table I explains the characteristics of different classes of anomalies in terms of distributed nature, aggregation type, clustering nature and impact on traffic features. For example, a SYN DDoS which targets one machine from a high number of hosts located in several /24 addresses will consist of a cluster if flows are aggregated in l_3 (cf., Section III-A). In fact, each of these /24 addresses will have traffic attributes values different from the ones of normal traffic: a high packet rate, a single destination and many SYN packets. The complete set of these flows will most probably create a cluster. However, if flows are aggregated in l_6, the single destination address will be represented as an outlier, characterized by many sources and a high fraction of SYN packets.

Our particular goal is to identify and to isolate the different flows that compose the anomaly flagged in the first stage, both in a robust way. Unfortunately, even if hundreds of clustering algorithms exist [45], it is very difficult to find a single one that can handle all types of cluster shapes and sizes, or even decide which algorithm would be the best to our particular problem. Different clustering algorithms produce different partitions of data, and even the same clustering algorithm provides different results when using different initializations and/or different algorithm parameters. This is in fact one of the major drawbacks in current cluster analysis techniques: the lack of robustness.

To avoid such a limitation, we have developed a divide and conquer clustering approach, using the notions of clustering ensemble [46] and multiple clusterings combination. A clustering ensemble $\mathbf{P}$ consists of a set of N partitions $P_{n,n=1,..,N}$ produced for the same data. Each of these partitions provides a different and independent evidence of data structure, which can be combined to construct a global clustering result for the whole feature space. There are different ways to produce a clustering ensemble. We use Sub-Space Clustering (SSC) [42] to produce multiple data partitions, applying the same clustering algorithm to N different sub-spaces $\mathbf{X}_n \subset \mathbf{X}$ of the original space. In particular, we use the well-known DBSCAN density-based clustering approach [47]. DBSCAN is a powerful clustering algorithm that discovers clusters of arbitrary shapes and sizes, relying on a density-based notion of clusters: clusters are high-density regions of the space, separated by low-density areas. This algorithm perfectly fits our unsupervised traffic analysis, because it is not necessary to specify a-priori difficult to set parameters such as the number of clusters to identify, which is the case in other clustering algorithms such as k-means.

1) Clustering Ensemble and Sub-Space Clustering: Each of the N sub-spaces $\mathbf{X}_n \subset \mathbf{X}$ is obtained by selecting R features from the complete set of A attributes. The number of sub-spaces N is therefore equal to R-combinations-obtained-from-A. To set the sub-space dimension R, we take a very useful property of monotonicity in clustering sets, known as the downward closure property: "if a collection of points is a cluster in a d-dimensional space, then it is also part of a cluster in any $(d-1)$ projections of this space" [48]. This directly implies that, if there exists any evidence of density in $\mathbf{X}$, it will certainly be present in its lowest-dimensional sub-spaces. Using small values for R provides several advantages: firstly, doing clustering in low-dimensional spaces is more efficient and faster than clustering in bigger dimensions. Secondly, density-based clustering algorithms such as DBSCAN provide better results in low-dimensional spaces [48], because high-dimensional spaces are usually sparse, making it difficult to distinguish between high and low density regions. We shall therefore use $R = 2$ in our SSC algorithm, which gives $N = C_R^A = A(A-1)/2$ data partitions, each of them obtained from each of the resulting sub-spaces.

Having produced the N partitions, we now explore different methods to combine these partitions in order to build a single partition where anomalous flows are easily distinguishable from normal-operation traffic: the Evidence Accumulation (EA) approach and the Inter-Clustering Result Association (ICRA) approach.

2) Combining Multiple Partitions using Evidence Accumulation: A possible answer is provided in [43], where authors introduced the idea of multiple-clusterings Evidence Accumulation (EA). By simple definition of what it is, an anomaly may consist of either outliers or small-size clusters, depending on the aggregation level of flows in $\mathbf{Y}$ (cf Table I). We use EA on top of $\mathbf{P}$ to build two inter-pattern similarity measures between the flows in $\mathbf{Y}$: a similarity matrix S to detect small clusters and a vector D to rank outliers. $S(p,q)$ represents the similarity between flows p and q. This value increases when the flows p and q are located in the same cluster in multiple partitions, and when the sizes of the resulting clusters are small. The rationale of these two conditions is to spot out very similar flows which are different from the majority. $D(o)$ represents the abnormality of the outlier o. This value increases when the outlier has been classified as such in several partitions and when the separation between the outlier and the normal traffic is bigger. As we are only interested in finding the smallest-size clusters and the most dissimilar outliers, the detection consists in finding the flows with the biggest similarity in S and the biggest dissimilarity in D. Any clustering algorithm can then be applied on the similarity matrix S to obtain a final partition of $\mathbf{X}$ that isolates small-size clusters of high intra-similarity values. Concerning the most dissimilar outliers, they can be isolated through a threshold applied to the values of D.

3) Combining Multiple Partitions using Inter-Clustering Result Association: A more robust approach to identify relevant clusters and outliers is provided by the ICRA approach. The idea of ICRA is to consider inter-cluster similarity rather than inter-pattern similarity, reducing the chances of merging together anomalous and normal flows. The ICRA aggregation shifts the similarity measure from the patterns to the clustering results, adding an additional analysis level. The problem solved by ICRA is split in two sub-problems: clusters' correlation through Inter-CLuster Association (ICLA), and outliers' correlation through Inter-Outlier Association (IOA).

In each case, a graph is used to express the similarity between either clusters or outliers. Each vertex is a cluster/outlier from any sub-space $\mathbf{X}_n$ (i.e., the complete set of clusters and outliers $\in \mathbf{P}$) and each edge represents the fact that two connected vertices are similar. The underlying idea is straightforward: identify clusters or outliers present in different sub-spaces that contain the same flows. To do so, we first define a Cluster Similarity measure CS between two clusters C_r and C_s: $CS(C_r, C_s) = \frac{card(C_r \cap C_s)}{max(card(C_r), card(C_s))}$, $card$ being the function that associates a cluster with its cardinality, and $C_r \cap C_s$ the intersection of C_r and C_s. Each edge in the ***cluster similarity graph*** between two clusters C_r and C_s means $CS(C_r, C_s) > 0.9$, being this an empirically chosen value[3]. IOA uses an ***outlier similarity graph*** built by linking outliers in different sub-spaces which represent the same pattern – aggregated flow (note that the same aggregated flow can result in multiple different outliers in $\mathbf{P}$, depending on the specific sub-spaces where those outliers are identified). Once these graphs are built, we need to find cluster sets where every cluster contains the same flows. In terms of vertices, we need to find vertex sets where every vertex is linked to every other vertex. In graph theory, such vertex set is called

[3]The value 0.9 guarantees that the vast majority of patterns are located in both clusters with a small margin of error.

a *clique*. The clique search problem is a NP-hard problem. Most existing solutions use exhaustive search inside the vertex set which is too slow for our application. We then make the hypothesis that a vertex can only be part of a single clique. A greedy algorithm is then used to build each clique. Relevant flow sets are finally identified as the intersection of all the aggregated flows present in the clusters or outliers within each clique. We thus yield a global partition of every aggregated flow.

4) Identification of Anomalies: Once the global partition over aggregated flows is built, we need to determine what is the nature of each of these flows. In other words, what is the nature of each cluster or outlier from the global partition. To do so, we make the hypothesis that a large proportion of the traffic in the analyzed time slot is normal. The normal traffic will thus be identified as a single big cluster which shall contain more than 50% of the aggregated traffic flows. This fraction comes from our operational experience when analyzing MAWI traffic, but is also related to the fact that outliers are, by definition, less numerous than clustered patterns [21]. If such cluster does not exist in the partition built upon the considered time slot, we consider that the approach is not able to reliably separate anomalous and normal traffic. We then discard the current time slot and restart the process at step 1 on Figure 1. When the condition is actually met, the time slot is valid, and every aggregated flow outside the normal traffic is considered as anomalous.

IV. Anomaly post-processing

The unsupervised clustering step presented in Section III allows to extract anomalies from a feature space $\mathbf{X}$ describing F aggregated flows according to an aggregation level l_i. However the potential number of detected anomalies can be high and thus overwhelm the operator. A post-processing step is thus crucial to help the operator to prioritize its own work towards the most dangerous anomalies. A more general help shall also be provided to the operator concerning tedious tasks such as characterization of the anomaly nature. We thus propose a three steps post-processing technique that reduces the amount of work required from the operator and increases the robustness of the overall approach, by successively correlating and reducing the overall number of found anomalies, ranking anomalies by dangerousness, and building anomaly signatures. This post-processing step corresponds to the last part of the unsupervised system.

A. Correlating Anomalies from Multiple Aggregations

Using the clustering and analysis algorithms previously introduced on single flow aggregation levels would generally (and naturally) generate false alarms. A standard approach to reduce the number of such false alarms is to combine and correlate the anomalies detected by analyzing flows at different aggregation levels l_i. In a nutshell, instead of relying on a single snapshot analysis of the data, why not combining multiple outlooks of the same dataset, using different flow aggregations? We actually follow this rationale to correlate anomalies extracted from the clustering results obtained for different flow aggregation levels.

To do so, we define two unique characteristics of an anomaly: its source IP address (source IP address set) and its destination IP address (destination IP address set). We then define the similarity between two IP address sets as the ratio between the sets' intersection cardinality and the maximum cardinality of each IP address set (cf., Section III-B3). If the similarity of the two IP address sets (source and destination) for two different anomalies are over a specific threshold, it guarantees that these anomalies have a very similar source and destination IP address sets. In this work, we chose to only correlate anomalies detected from aggregation levels source and destination (i.e., l_1 and l_2), in order to avoid correlating anomalies located in the same aggregation level type (e.g. l_3 and l_4), that would be potentially contained in each other. Finally, correlated anomalies are then built from each couple of similar anomalies.

B. Anomaly ranking

To increase its own efficiency, a network operator analyzing anomalies needs to prioritize its work towards the most dangerous anomalies. In the field of network anomaly detection, the more an anomaly is dangerous, the more its mitigation is critical. Thus, task prioritization shall be realized by ranking anomalies according to their dangerousness. In this paper, we adopted the operator point of view, i.e. we mainly care about the network performance and try to detect any unexpected traffic variation which could impact the throughput or the delay of the Internet service provided to customers. Among the anomalies which are dangerous for the network performance are the flooding based attacks and the flash crowd like sudden increases of the traffic. From this point of view, we define a ranking formula based on the amount of traffic to be transmitted at any time in the network. We introduce two anomaly characteristics to rank anomalies: its number of packets and its number of bytes. We build a dangerousness index that uses these two characteristics. We actually do not use the absolute number of packets and bytes but the fraction of the traffic belonging to this anomaly in terms of number of packets and bytes. Let DI be the Dangerousness Index built according to the two aforementioned criteria: fraction of number of packets and fraction of number of bytes:

$$DI(anomaly) = P(\#pkts) + P(\#bytes) \tag{1}$$

$$\text{where}: \begin{cases} P(\#pkts) = \frac{\#\text{pkts in anomaly}}{\#\text{pkts in timeslot}} \\ P(\#bytes) = \frac{\#\text{bytes in anomaly}}{\#\text{bytes in timeslot}} \end{cases} \tag{2}$$

This formula defines the dangerousness criterion as the mean between the proportion of traffic belonging to the considered anomaly in terms of number of packets and the proportion of traffic belonging to the considered anomaly in terms of number of bytes. This step is designed to reflect the potential of dangerousness of an anomaly for the normal behavior of the network. From a general point of view, the bigger $DI(anomaly)$, the more dangerous the anomaly $anomaly$ is. The T_h threshold in Figure 1 is a threshold applied on the DI value for the considered anomaly. This dangerousness index is then used for flagging the most urgent anomalies to the network administrator.

C. Automatic Characterization of Anomalies

At this point, the unsupervised algorithm has identified anomalies containing a set of aggregated traffic flows $\in \mathbf{Y}$ far from the rest of the traffic (far in the sense of the used clustering algorithms, i.e., identifying outliers). Some of these anomalies are correlated and they are all ranked in terms of dangerousness. The following and final post-processing step is to produce filtering rules to correctly isolate and characterize each of these anomalies. Such a signature could eventually be compared against well-known signatures to automatically classify the anomaly, or be integrated into a list of new signatures when no matches are found (e.g., in the case of detecting a 0-day anomaly, i.e., a previously unknown one).

To produce such filtering rules, the algorithm selects those sub-spaces $\mathbf{X}_n$ where the separation between the considered anomalous flows and the rest of the traffic is the biggest. We define two different classes of filtering rule: ***absolute*** rules $FR_A(\mathbf{Y})$ and ***relative*** rules $FR_R(\mathbf{Y})$. Absolute rules do not depend on the separation between flows, and correspond to the presence of dominant features in the considered flows. An absolute rule for a certain feature j characterizing a certain flow set $\mathbf{Y}_g$ has the form $FR_A(\mathbf{Y}_g, j) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(j) == \lambda\}$, where λ is in this case the value of the dominant feature, obtained directly from the data. For example, in the case of an ICMP flooding attack, the vast majority of the associated flows use only ICMP packets, hence the absolute filtering rule $\{\text{nICMP}/\text{nPkts} == 1\}$ makes sense (i.e., $\lambda == 1$). On the contrary, relative filtering rules depend on the relative separation between anomalous and normal-operation flows. Basically, if the anomalous flows are well separated from the normal cluster in a certain partition P_n, then the features of the corresponding sub-space $\mathbf{X}_n$ are good candidates to define a relative filtering rule. A relative rule for feature j has the form $FR_R(\mathbf{Y}_g, j) = \{\forall \mathbf{y}_f \in \mathbf{Y}_g \subset \mathbf{Y} : x_f(j) < \lambda \vee x_f(j) > \lambda\}$. Here λ represents a separation threshold, obtained once again from the data, as explained in Figure 4. We also define a *covering relation* between filtering rules: we say that rule F_1 *covers* rule $F_2 \Leftrightarrow F_2(\mathbf{Y}) \subset F_1(\mathbf{Y})$. If two or more rules overlap (i.e., they are associated to the same feature), the algorithm keeps the one that covers the rest.

In order to construct a compact signature of the anomaly, we have to devise a procedure to select the most discriminant filtering rules. Absolute rules are important, because they define inherent characteristics of the anomaly. As regards relative rules, their relevance is directly tied to the degree of separation between anomalous and normal flows. In the case of outliers, we select the K features for which the Mahalanobis distance to the normal-operation traffic is among the top-K biggest distances. We use the Mahalanobis distance to consider the variance of the different features within the distance computation. In the case of small-size clusters, we rank relative rules according to the degree of separation to the normal traffic (represented by the biggest cluster) using the well-known Fisher Score (FS), which also uses the intra-distance variance within each cluster (normal and anomalous). To finally construct the signature, the absolute rules and the top-K relative rules are combined into a single inclusive predicate, using the covering relation in case of overlapping rules.

V. Experimental Evaluation in Real Traffic

We focus now on the evaluation of the unsupervised detection algorithm's ability to detect anomalies located in real traffic trace from the public MAWI repository of the WIDE project [3]. The WIDE operational network provides interconnection between different research institutions in Japan, as well as connections to different commercial ISPs and universities in the US. The traffic repository consists of 15 minutes-long raw packet traces collected daily since 1999. These traces were not originally labeled, but during the past years, the MAWILab initiative led by Prof. Kensuke Fukuda[4] has been working on the anomaly-related documentation of these traces. MAWILab [4] uses four anomaly detectors to analyze the MAWI traffic: a PCA and sketch-based detector [7], a detector relying on sketching and multi-resolution gamma modeling [6], a Hough Transform-based detector [5] and a detector relying on traffic feature distribution changes [8]. Anomaly labels are obtained by combining detection results from the previously presented approaches. The evaluation is performed in three steps: firstly, we provide a characterization of the types of attacks which are present at the MAWI data repository, using the labels provided at the MAWILab dataset [4]. Then, we present an example of the execution of the complete system over a single batch of measurements to help the reader get a better feeling of the rationales and principles of our technique. Finally, we perform a more thorough analysis on a subset of the MAWI repository covering six consecutive years of traffic traces, going from January 2001 to December 2006. While we acknowledge that these traces are rather old, we decided to use them because the associated ground truth is already mature and stable, which is paramount to conduct a proper evaluation. In any case, the rationale behind this evaluation is to test the capabilities of our system with real traffic, rather than discovering new types of anomalies in todays' Internet traffic. This long-term analysis considers also an evaluation of the sensitivity of the detection approach to the clustering parameters, as well as an assessment of the computational time of the analysis.

[4]www.fukuda-lab.org/mawilab/

A. Attacks in the MAWI Dataset

Figure 2 depicts the occurrence of anomalies and attacks over six years of MAWI traces, from January 2001 to December 2006. Figure 2a classifies the traffic in terms of attack events, i.e., all the traffic related to the occurrence of a specific attack or anomaly, whereas Figure 2b complements the picture in terms of packet counts. Events are classified in the following macro-categories: scans, Denial of Service (DoS), normal, and unknown.

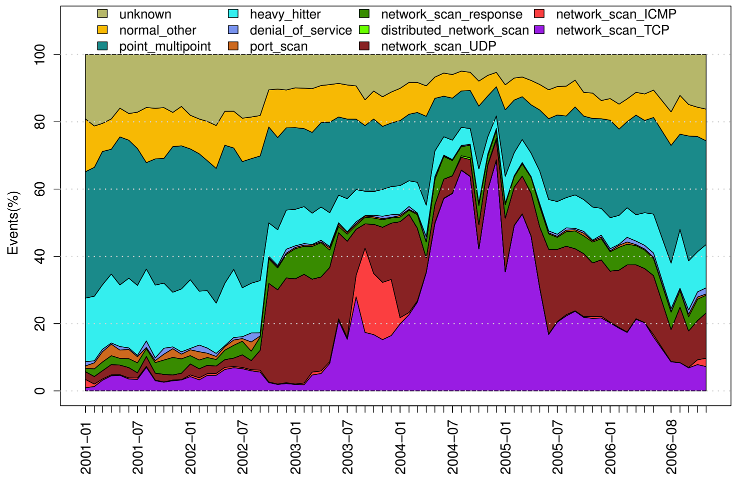
The largest majority of the attacking events correspond to scans. A very well known backdoor worm called Sasser has a very active period from May 2004 to June 2005, characterized as a surge of TCP network scans. It is interesting to note that a very small number of distributed scans account for a significant quantity of probing packets at the end of 2004: 27% in August, 29% in September, 13% in November, and 25% in December. There is also a surge of ping-based ICMP network scans that starts in September 2003 and lasts until December 2003. Anomalous yet no attacking events are also present in the traces, including (in decreasing proportion): point-multipoint events, heavy hitter and other events (which, in fact, are mainly "light hitters", i.e., point-to-point traffic of less than 1000 packets). The number of unknown events is always bounded to less than 20%, but these represent a large share of the overall packet counts, close to 40%.

B. Case Study: one Time Slot in one MAWI Trace

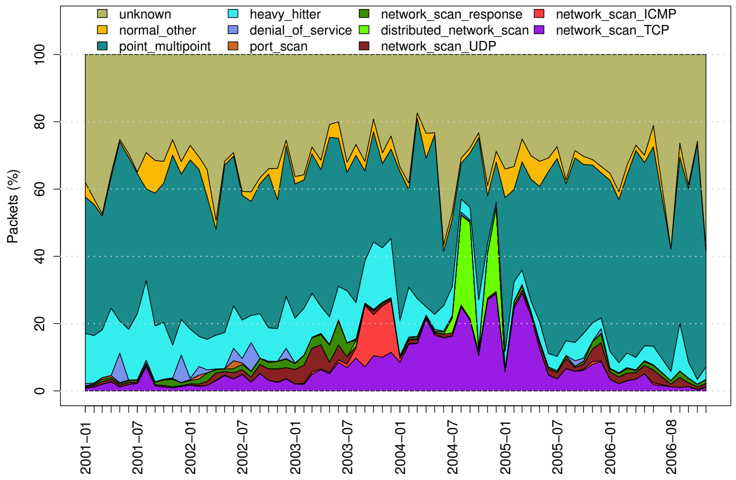
We present now several results associated to the execution of the unsupervised detection system over a single, 15 seconds long time slot. The choice of 15 seconds is motivated by the target of performing the analysis in short-time batches, close to real-time. The same slot length shall be considered in the rest of the evaluations. We use the MAWILab trace on April 1st 2004. This time slot lasts from the 405th to the 420th second in the trace.

Figure 3 depicts the similarity graphs obtained by the Inter-Clustering Result Association approach, as described in Section III-B1. The ICLA cluster similarity graphs obtained for flows aggregated on a destination IP address $/24$ basis are presented in Figure 3a. Each vertex is a cluster found in any of the generated sub-spaces. Each vertex number is the index of a cluster among the whole cluster set within the clustering ensemble $\mathbf{P}$. The normal traffic is here circled and it is represented by the vertex group with the highest number of vertices. Every other group of vertices is a clique and potentially contains an anomaly. Similarly, Figure 3b depicts the outlier similarity graph obtained for flow aggregation at the destination IP address $/24$ level using the IOA approach. Each vertex number is the index of the associated outlier among the complete outliers set within the clustering ensemble $\mathbf{P}$. Every edge means that the linked clusters or outliers are similar according to the criteria defined in Section III-B3. Finally, the same type of ICLA and IOA graphs are built over the source IP address $/24$ flow aggregation.

Every connected component which is also a clique is treated as a potential anomaly. Each potential anomaly is assigned an index. This index has the following meaning: values between 0 and 99 represent anomalies from clusters found in source IP address aggregated flows; values between 100 and 199 represent anomalies from outliers found in source IP address aggregated flows;, values between 200 and 299 represent clusters found in destination IP address aggregated flows, and values between 300 and 399 represent anomalies from outliers found in destination IP address aggregated flows. The choice of 100 anomalies for

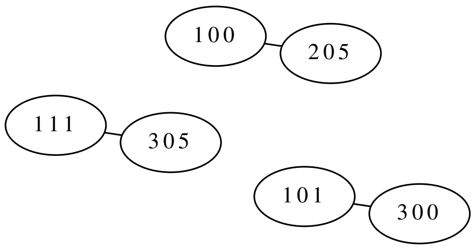


(a) Attacking events.



(b) Packet counts.

Fig. 2: Attack events occurrence and packet counts from the MAWILab repository.



(a) Cluster similarity graph (b) Outlier similarity graph

(c) Anomaly similarity graph

Fig. 3: Cluster similarity graph, outlier similarity graph and anomaly similarity graph for destination aggregated data. Anomalies are easily identified as small cliques.

Anomaly type	Source traffic segment index	Destination traffic segment index	Source signature	Destina
Few ICMP pkts	111	305	nSrcs $= 1$, nICMP/nPkts $> \lambda_1$	nS nICMP
Few ICMP pkts	101	300	nSrcs $= 1$, nICMP/nPkts $> \alpha_1$	nS nICMP
SYN Net Scan	100	205	nSrcs $= 1$, nDsts $> \beta_1$, nSYN/nPkts $> \beta_2$	nSrcs = nSYN

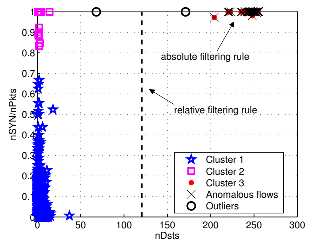
TABLE II: Signatures of anomalies found.

each type of anomaly source is made under the assumption that there are less than 100 cliques in each graph, but the indexing can be simply overruled in case of more detected anomalies.

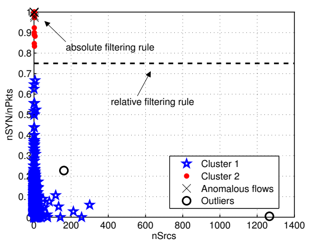
The next step consists of applying anomaly correlation on the potential anomalies, to finally identify those flows that are different from the normal ones and that share the same source IP address and destination IP address sets. Recall the definitions introduced in Section IV-A to compute a similarity metric among potential anomalies. Figure 3c depicts the resulting anomaly similarities as a graph. In this specific analyzed time slot, the anomaly correlation results in three edges from the graph. Each edge represents the link between two similar anomalies, here anomalies 101 and 300 (outlier in both source and destination IP address aggregation, i.e., a point-to-point anomaly), 111 and 305 (the same as before), and finally 100 and 205 (outlier in source IP address aggregation and cluster in destination IP address aggregation, i.e., a point-multipoint anomaly).

Table II details each of the identified anomalies along with its type, the segment indexes extracted from ICLA and IOA and the two signatures detected from both source and destination aggregated flows. The term "Few ICMP packets" actually means that these two anomalies were containing just a few harmless ICMP packets. Both of these anomalies could have easily been discarded by an impact estimation based on nPkts/second**.**

To better understand how the signatures are generated, Figures 4a and 4b depict the results of the characterization phase for the SYN network scan anomaly. Each sub-figure represents a partition P_n for which filtering rules were found. They involve the number of IP sources and destinations, and the fraction of



(a) SYN Network Scan (1/2)



(b) SYN Network Scan (2/2)

Fig. 4: Filtering rules for characterization of the found network scan in MAWI.

(a) ROC curves of attacking events. (b) Sensitivity curves of alarms.

Fig. 5: ROC curve for attacking events and sensitivity curve of alarms from 2001 to 2006 with 4 traces by month.

SYN packets. Combining them produces a signature that can be expressed as $(\text{nSrcs} = 1) \wedge (\text{nDsts} > \beta_1) \wedge (\text{nSYN/nPkts} > \beta_2)$, where β_1 and β_2 are the two thresholds obtained by separating normal and anomalous clusters at half distance, as depicted in Figure 4. This signature makes perfect sense: the network scan uses SYN packets from a single attacking host to a large number of victims. The main advantage of the unsupervised approach relies on the fact that this new signature has been produced without any previous information about the attack or the baseline traffic.

C. Global Evaluation and Benchmarking on the MAWI Dataset

We present now a deeper and more reliable view of the global detection results obtained with the unsupervised detection approach on the MAWI dataset. In particular, we run the detection system on a large time-span subset of the MAWI dataset, consisting of four traces randomly selected per month during six years, from January 2001 to December 2006. The detection accuracy is evaluated through ROC curves and alarm sensitivity curves. ROC curves evaluation is the standard statistical approach used to evaluate a detector or binary classifier. ROC curves allow the visualization of the trade-off between the True Positives Rate (TPR) and False Positives Rate (FPR) in a single curve, as a function of the detection threshold T_h, as described in Section IV-B.

The ROC curve analysis considers the detection of all the attacking events in the corresponding time span, using the labels of the MAWILab dataset as ground truth. An attack is assumed to be correctly detected if at least one of its flows is flagged as anomalous. The alarm sensitivity evaluation accounts for the variations of the TPR when changing the anomaly detection threshold. Figure 5 reports the obtained results. The ROC curve depicted in Figure 5a shows that the unsupervised detection system is able to detect more than 95% of the anomalous events documented by MAWILab. The curve also evidences a constant and very low rate of false positives across statistical tests, with a FPR below 1%. This is especially important since it reduces the risk of alarms saturation for the network operator. In terms of the TPR sensitivity to the detection threshold, we see that, as expected, smaller detection thresholds are able to capture more of the anomalous events, additionally showing that the largest majority of the attacks are composed by small flows in terms of packets and transmitted bytes. This is also coherent with the fact that most of the attacks in the MAWILab dataset are labeled as scans.

Detector	Total # alarms	# alarms also in MAWILab	% events w.r.t MAWILab
Hough [5]	275419	134920	71%
Gamma [6]	608714	142872	75%
PCA [7]	1375118	145834	77%
KL [8]	24781	24777	13%

TABLE III: Table of alarms reported by each detector and their respective contributions to MAWILab. MAWILab contains 190375 anomalies from 2001 to 2006.

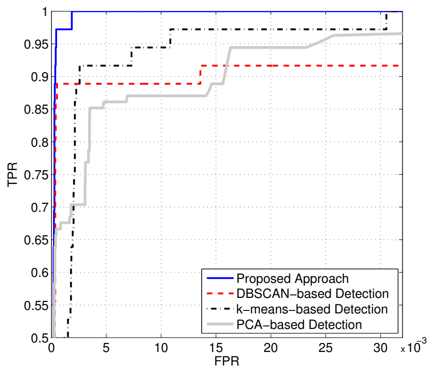
As a comparison with other papers analyzing MAWI traffic, we describe the detection results of the detectors [5]–[8] used in MAWILab [4] in Table III. It is important here to remind that the MAWILab labels are obtained through consensus among detectors. This means that every alarm from each detectors is not necessary considered as abnormal in MAWILab, only a fraction of alarms are actually classified as anomalies. Table III thus provides the absolute and relative number of anomalies for each detector that contributes to MAWILab from 2001 to 2006. The detector that exhibits the biggest contributions is PCA [7]: 77% of the documented anomalies in MAWILab have been reported by this detector. Since our unsupervised anomaly detection system is able to detect at least one flow in more than 95% of the anomalies of MAWILab, it has better performance than any of the detectors used in MAWILab in terms of number of detected anomalous events. This demonstrates the high accuracy of our detection approach: it can detect a wide range of anomalies and obtain results close to the ones obtained in MAWILab through a combination of multiple anomaly detectors.

D. Comparison to Other Unsupervised Approaches

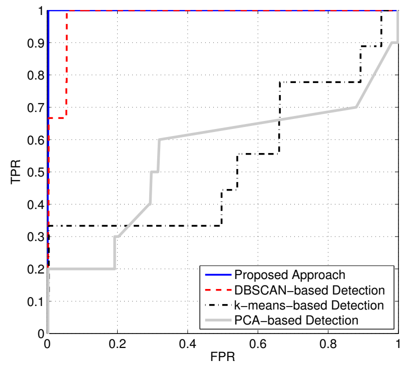
In this Section we propose to benchmark the detection capabilities of the proposed system against some approaches proposed in the past for unsupervised detection of network attacks and anomalies: DBSCAN-based, k-means-based, and PCA-based outliers detection. The first two consist of applying either DBSCAN or k-means to the complete feature space X, identify the largest cluster $C_{\max}$, and compute the Mahalanobis distance of all the flows lying outside $C_{\max}$ to its centroid. The ROC curve is generated by comparing the sorted distances to a detection threshold. These approaches are similar to those used in previous work [9]–[11]. In the PCA-based approach, PCA and the sub-space methods [12], [13] are applied to the complete matrix X, and the attacks are detected by comparing the residuals to a variable threshold. Both the k-means and the PCA-based approaches require fine tuning: in k-means, we repeat the clustering for different values of clusters k, and take the average results. In the case of PCA we present the best performance obtained for each evaluation scenario.

The comparison is conducted using once again MAWI traffic, but now including real traffic traces from the METROSEC project [14]. These traces consist of real traffic collected on the French RENATER network, containing simulated attacks performed with well-known DDoS attack tools. Traces were collected between 2004 and 2006, and contain DDoS attacks that range from very low intensity (i.e., less than 4% of the overall traffic volume) to massive attacks (i.e., more than 80% of the overall traffic volume).

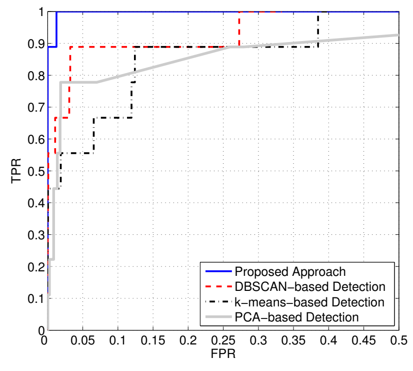
Figure 6 depicts the detection performance and false alarm rates obtained in the detection of attacks in MAWI and METROSEC traffic. Figure 6a corresponds to the detection of 36 anomalies in MAWI traffic, using l_1 flow aggregation level (cf., Section III-A). These anomalies include network and port scans, worm scanning activities (Sasser and Dabber variants), and some anomalous flows consisting on very high volumes of NNTP traffic.



(a) MAWI, l_1 flow aggregation.



(b) MAWI, l_2 flow aggregation.



(c) METROSEC, l_2 flow aggregation.

Fig. 6: True positives rate vs false alarms in MAWI and METROSEC. Comparison to state-of-the-art unsupervised approaches for anomaly detection.

Figure 6b also corresponds to anomalies in MAWI traffic, but using l_2 aggregation level. In this case, there are 9 anomalies, including different kinds of flooding DoS/DDoS attacks. Finally, Figure 6c corresponds to the detection of 9 DDoS attacks in the METROSEC dataset. From these, 5 correspond to massive attacks (more than 70% of traffic), 1 to a high intensity attack (about 40%), 2 are low intensity attacks (about 10%), and 1 is a very-low intensity attack (about 4%). The detection is performed using l_2 aggregation.

Obtained results permit to evidence the great advantage of using the proposed approach with respect to previous ones. In particular, all the approaches used in the comparison generally fail to detect all the attacks with a reasonable false alarm rate. Both the DBSCAN-based and the k-means-based algorithms get confused by masking features when analyzing the complete feature space X. The PCA approach shows to be not sensitive enough to discriminate different kinds of attacks of very different intensities, using the same representation for normal-operation traffic.

E. Sensitivity Analysis of the Clustering Algorithm

The performance of the detector is directly linked to the goodness of the clustering step. By goodness we refer to the ability of the algorithm to produce homogeneous clusters. Remember that we use DBSCAN to cluster each of the subspaces of the feature space. As any other clustering algorithm, DBSCAN has some parameters which constrain its performance; in particular, DBSCAN uses two parameters which define its notion of cluster density: the minimum number of patterns which define a cluster ($nbpDBS$ from now on) and the intra-patterns maximum neighboring distance ϵ (or $epsDBS$ from now on). We therefore evaluate the sensitivity of the detection accuracy of the complete system to variations on $nbpDBS$ and $epsDBS$. Sensitivity analysis is critical as it permits to verify that an algorithm is able to perform well when used with a wide range of settings. We conduct the sensitivity analysis on a subset of the MAWI dataset generated by randomly selecting one trace for each month between January 2001 and December 2006. As mentioned before, the ground-truth is provided by the MAWILab dataset.

The parameter ϵ defines the neighborhood of a pattern. In our case, the feature space is normalized for each attribute j, and the dimension of each sub-space X_n is 2; ϵ is thus bounded between 0 and $\sqrt{2}$. When $\epsilon = 0$, each point is an outlier. On the contrary, $\epsilon = \sqrt{2}$ guarantees that each subspace contains a single cluster and no outliers. Figure 7a shows the obtained ROC curves of the detection accuracy when changing $epsDBS$ between 0.1

(a) ROC curves of events. (b) Sensitivity curves of alarms.

Fig. 7: ROC curve for attacking events and sensitivity curve of alarms regarding ϵ ($epsDBS$).

(a) ROC curves of events. (b) Sensitivity curves of alarms.

Fig. 8: ROC curve for attacking events and sensitivity curve of alarms regarding the number of points by cluster ($nbpDBS$).

and 0.3. The neighboring distance $epsDBS = 0.15$ provides the best detection performance; $epsDBS = 0.2$ exhibits a slightly degraded performance compared to $epsDBS = 0.15$, and both $epsDBS = 0.1$ and $epsDBS = 0.3$ display much worse results. Figure 7b shows similar results in terms of TPR variations.

The minimum number of patterns by cluster $nbpDBS$ represents the minimum number of patterns in the neighborhood of a considered pattern needed to form a cluster (this neighborhood being defined by $epsDBS$); $epsDBS$ is bounded between 2 and the total number of patterns in the feature space. However, if we want to correctly find anomalous clusters, we shall use a small enough value to find small cluster. We therefore choose to test values between 2 and 20. Figure 8 evidences the lack of influence of $nbpDBS$ on the detection performance and sensitivity, since all curves are overlapping. There are several reasons that explain this behavior. First, the value of $nbpDBS$ has no impact on the ability of the system to find a normal cluster in each subspace X_n. Such impact would be noticeable if the selected values would be high enough to cause the absence of a normal cluster, as defined in Section III-B4. The chosen values never lead to the absence of a normal cluster since there is always at least one cluster that contains more than 20 aggregated flows. Second, the considered parameter has no influence on whether an anomaly is considered as detected in the MAWILab dataset. Section III explains that anomalies can be represented either as clusters or outliers. When the minimum number of patterns by clusters is too high, an anomaly does not contain enough flows to form a cluster. In this case, the anomaly is represented by several outliers. But, from the point of view of MAWILab, both cases are identical since the same flows are tagged as anomalous. These two issues explain the lack of influence of $nbpDBS$ on the detection performance.

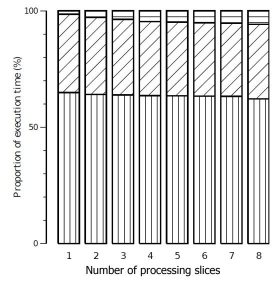
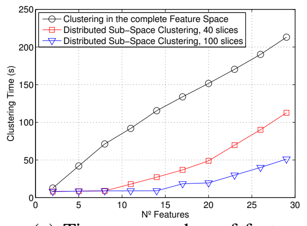
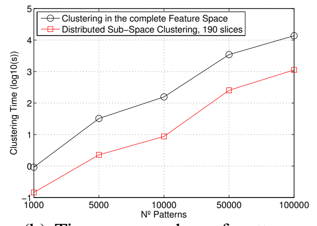


Fig. 9: Proportion of execution time for source aggregated flows clustering (vertical stripes), destination aggregated flows clustering (diagonal stripes) and other processing time (horizontal stripes), according to an increasing number of processing slices.



(a) Time vs. number of features.



(b) Time vs. number of patterns.

Fig. 10: Clustering Time as a function of number of features and number of flows to analyze. The number of aggregated flows in (a) is $F = 10000$. The number of features and slices in (b) is $A = 20$ and $S = 190$ respectively.

F. *Computational Time of the Unsupervised Analysis*

The last element that we analyze is the computational time of the unsupervised analysis algorithm. The SSC-based clustering algorithm performs multiple clusterings in $N(A)$ low-dimensional sub-spaces $\mathbf{X}_n \subset \mathbf{X}$. This multiple computation imposes scalability issues for on-line detection of attacks in very-high-speed networks. Two key features of the algorithm are exploited to reduce scalability problems in the number of features A and the number of aggregated flows F to analyze. Firstly, clustering is performed in very-low-dimensional sub-spaces, $\mathbf{X}_n \in \mathbb{R}^2$, which is faster than clustering in high-dimensional spaces [45]. Secondly, each sub-space can be clustered independently of the other sub-spaces, which is perfectly adapted for parallel computing architectures. Parallelization can be achieved in different ways: using a single multi-processor and multi-core machine, using network-processor cards and/or GPU (Graphic Processor Unit) capabilities, using a distributed cluster of machines, or combining all these techniques. In particular, the novel parallel processing programming model MapReduce [49] and the different big data parallel analysis frameworks implementing it such as Hadoop and Spark are perfectly fit to perform the proposed SSC algorithm in parallel. From now on, we shall use the term ”slice” as a reference to a single computational entity, independently of the specific implementation. In the following evaluations, algorithms run on a desktop machine consisting of an Intel i7 860 CPU with 4 cores and Hyperthreading, and 8GBs of RAM.

Figure 9 shows the share of the total execution time of the system between source and destination aggregated flows clustering time (vertical and diagonal stripes) and other processing time (horizontal stripes), according to an increasing number of slices. Just as a reference, the total execution time considers the analysis of the time slot from the 1st to the 15 th second of the MAWI trace on January 1st 2009. The clustering time of source aggregated flows is much greater than the clustering time of destination aggregated flows for any number of used slices. This is due to the fact that the number of source aggregated flows is much bigger than the number of destination aggregated flows: almost 9×10^3 to about 6×10^3. The evaluation clearly evidences that clustering is the most time consuming task of the overall process. It is therefore pertinent to focus our computational time analysis on the clustering time.

Figure 10 depicts the Clustering Time (CT) of the SSC-based clustering algorithm, both (a) as a function of the number of features A used to describe the aggregated flows and (b) as a function of the number of flows F to analyze. Figure 10a compares the CT obtained when clustering the complete feature space X, referred to as CT($\mathbf{X}$), against the CT obtained with SSC, varying A from 2 to 29 features. We analyze a large number of aggregated flows, $F = 10^4$, and use two different number of slices, $S = 40$ and $S = 100$. The analysis is once again done with traffic from the MAWI dataset, combining multiple traces to attain the desired number of flows. To estimate the CT of SSC for a given value of A and S, we proceed as follows: first, we separately cluster each of the $N = A(A-1)/2$ sub-spaces $\mathbf{X}_i$, and take the worst-case of the obtained clustering time as a representative measure of the CT in a single sub-space, i.e., CT$(\mathbf{X}_{\text{SSCwc}}) = \max_n$ CT$(\mathbf{X}_n)$. Then, if $N \leqslant S$, we have enough slices to completely parallelize the SSC algorithm, and the total CT corresponds to the worst-case (or it is bounded by the worst-case), CT$(\mathbf{X}_{\text{SSCwc}})$. On the contrary, if $N > S$, some slices have to cluster various sub-spaces, one after the other, and the total CT becomes $(N\%S + 1)$ times the worst-case CT$(\mathbf{X}_{\text{SSCwc}})$, where $\%$ represents integer division. The first interesting observation from Figure 10a regards the increase of CT$(\mathbf{X})$ when A increases, going from about 8 seconds for $A = 2$ to more than 200 seconds for $A = 29$. As we said before, clustering in low-dimensional spaces is faster, which reduces the overhead of multiple clusterings computation. The second paramount observation is about parallelization: if the algorithm is implemented in a parallel computing framework, it can be used to analyze large volumes of traffic using many traffic descriptors in an on-line basis; for example, if we use 20 traffic features and a parallel framework with 100 slices, we can analyze 10000 aggregated flows in less than 20 seconds. Recall that depending on the aggregation level used in the analysis, this might even correspond to millions of standard 5-tuple IP flows per second.

Figure 10b compares CT$(\mathbf{X})$ against CT$(\mathbf{X}_{\text{SSCwc}})$ for an increasing number of flows F to analyze, using $A = 20$ traffic features and $S = N = 190$ slices (i.e., a completely parallelized implementation of the SSC-based algorithm). As before, we can appreciate the difference in CT when clustering the complete feature space vs. using low-dimensional sub-spaces: the difference is more than one order of magnitude, independently of the number of flows to analyze. Regarding the volume of traffic that can be analyzed with this 100% parallel configuration, the SSC-based algorithm can analyze up to 50000 aggregated flows with a reasonable CT, about 4 minutes in this experience. In the MAWI traffic analysis corresponding to traces between 2001 and 2006, the average number of aggregated flows in a time slot of $\Delta T = 20$ seconds rounds the 2500 flows, which represents a value of CT$(\mathbf{X}_{\text{SSCwc}}) \approx 0.4$ seconds. For the $m = 9$ features that we have used ($N = 36$), and even without doing parallelization, the total CT is $N \times$ CT$(\mathbf{X}_{\text{SSCwc}}) \approx 14.4$ seconds.

VI. Concluding Remarks

The completely unsupervised anomaly detection system that we have presented in this paper has many interesting advantages w.r.t. previous proposals in the field. It uses exclusively unlabeled data to detect and characterize network anomalies, without assuming any kind of signature, particular model, or canonical data distribution. This allows to detect new previously unseen

anomalies, even without using statistical-learning or human analysis. Despite using ordinary clustering techniques, the algorithm avoids the lack of robustness of general clustering approaches by applying the notions of Sub-Space Clustering. The post-processing approach permits to construct easy-to-interpret-and-to-visualize ranked results, providing insights and explanations about the detected anomalies to the network operator. This allows to prioritize his time and reduce his overall workload.

We have verified the effectiveness of the overall system to detect distributed and non-distributed network anomalies in a completely blind fashion on real traffic from the WIDE network, without assuming any particular traffic model, significant clustering parameters, or even clusters structure beyond a basic definition of what an anomaly is. The evaluation on six years of traffic traces and the sensitivity analysis provide strong evidence of the accuracy and robustness of the employed techniques to detect network anomalies. In addition, we have shown detection results that outperform state-of-the-art approaches for unsupervised anomaly detection.

We have evaluated the computational time of the proposed clustering algorithm. Results confirm that the system can be used for on-line unsupervised detection and characterization of network attacks for the volumes of traffic that we have analyzed. Even more, they show that if run in a parallel framework, as it is dictated by the big data analytics tendency today, the analysis can reasonably scale-up to run in high-speed networks, using more traffic descriptors to characterize network attacks.

Still, we believe that there are several aspects of the system which deserve some discussion. The first of these aspects is the selection of the traffic features for the traffic analysis. The used traffic features set in this paper is relatively small and indeed carefully chosen. This may lead the reader to think that the proposed setup is rather ”ad hoc” and lacks generality. However, the evaluation results clearly show that our algorithm is able to discriminate irrelevant and relevant traffic features. In fact, our approach extracts clusters and outliers that are far from normal traffic regarding very diverse features. One also needs to consider the fact that the automatically constructed signatures (cf. Table I) do not use the same features. This suggests that adding new features, and among them, irrelevant ones, will not impact anomaly mining performance. We intend to greatly increase the number of used traffic features in future work to introduce generality and verify that the discrimination power is still preserved.

The clustering techniques also deserve some thoughts. While the evaluation clearly shows that the detector is efficient in terms of detection ability, the clustering technique itself looks perfectible in terms of execution time. SSC combined with ICRA is indeed able to reliably separate anomalies with different properties and, in our case, isolate anomalous traffic from normal flows. However, the computational time analysis clearly shows that clustering is the most time consuming task. This leads us to think that the trade-off between accuracy and analysis latency leans towards the former at the expense of the latter. State-of-the-art clustering algorithms such as those evaluated in [50] and [51] might be interesting in our particular approach. These algorithms would potentially allow to greatly improve the response time of the system while maintaining, and maybe improving, the detection accuracy. However, such integration must be carefully realized in order to preserve the anomaly mining abilities of the system as proposed in this paper.

This work has been done in the framework of the ECODE project, funded by the European commission under grant FP7-ICT-2007-2/223936. The work has been supported by the Strategic International Collaborative R&D Promotion Project, of the Ministry of Internal Affairs and Communication, Japan, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 608533 (NECOMA).

References

[1] Casas P, Mazel J, Owezarski P. Steps towards autonomous network security: Unsupervised detection of network attacks. *Proceedings of the 4th IFIP International Conference on New Technologies, Mobility and Security*, 2011; 1–5.

[2] Mazel J, Casas P, Labit Y, Owezarski P. Sub-space clustering, inter-clustering results association & anomaly correlation for unsupervised network anomaly detection. *Proceedings of the 7th International Conference on Network and Service Management*, 2011; 1–8.

[3] Cho K, Mitsuya K, Kato A. Traffic data repository at the wide project. *Proceedings of the annual conference on USENIX Annual Technical Conference*, 2000; 51–56.

[4] Fontugne R, Borgnat P, Abry P, Fukuda K. MAWILab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. *Proceedings of the 10th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2010.

[5] Fontugne R, Fukuda K. A Hough-transform-based anomaly detector with an adaptive time interval. *ACM SIGAPP Applied Computing Review* 2011; 11(3):41–51.

[6] 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)*, 2007; 145–152.

[7] Kanda Y, Fontugne R, Fukuda K, Sugawara T. ADMIRE: Anomaly detection method using entropy-based pca with three-step sketches. *Computer Communications* 2013; 36(5):575–588.

[8] Brauckhoff D, Dimitropoulos X, Wagner A, Salamatian K. Anomaly extraction in backbone networks using association rules. *IEEE/ACM Transactions on Networking* 2012; 20(6):1788–1799.

[9] 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*, 2001.

[10] 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*, 2002.

[11] Leung K, Leckie C. Unsupervised anomaly detection in network intrusion detection using clusters. *Proceedings of the Twenty-eighth Australasian conference on Computer Science*, 2005; 333–342.

[12] Lakhina A, Crovella M, Diot C. Diagnosing network-wide traffic anomalies. *Proceedings of the 4th conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2004.

[13] 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)*, 2005; 217–228.

[14] METROlogy for SECurity and qos 2007. `http://laas.fr/METROSEC`.

[15] 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 (IMC)*, 2002; 71–82.

[16] Brutlag JD. Aberrant behavior detection in time series for network monitoring. *Proceedings of the 14th USENIX conference on System administration*, 2000; 139–146.

[17] Fontugne R, Abry P, Fukuda K, Borgnat P, Mazel J, Wendt H, Veitch D. Random projection and multiscale wavelet leader based anomaly detection and address identification in internet traffic. *Proceedings of the 40th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[18] 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)*, 2005; 331–344.

[19] 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)*, 2003; 234–247.

[20] Silveira F, Diot C, Taft N, Govindan R. ASTUTE: detecting a different class of traffic anomalies. *Proceedings of the ACM SIGCOMM 2010 conference*, 2010; 267–278.

[21] Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. *ACM Comput. Surv.* Jul 2009; 41(3):15:1–15:58, doi:10.1145/1541880.1541882. URL http://doi.acm.org/10.1145/1541880.1541882.

[22] Marnerides A, Schaeffer-Filho A, Mauthe A. Traffic anomaly diagnosis in internet backbone networks: A survey. *Computer Networks* 2014; 73(0):224 – 243, doi:http://dx.doi.org/10.1016/j.comnet.2014.08.007. URL http://www.sciencedirect.com/science/article/pii/S1389128614002850.

[23] Xu K, Zhang ZL, Bhattacharyya S. Internet traffic behavior profiling for network security monitoring. *IEEE/ACM Transactions on Networking* 2008; 16(6):1241–1252.

[24] Fernandes G, Owezarski P. Automated classification of network traffic anomalies. *Proceedings of 5th Conference on Security and Privacy in Communication Networks (SecurComm)*, 2009; 91–100.

[25] Silveira F, Diot C. URCA: Pulling out anomalies by their root causes. *Proceedings of the 29th Conference on Information Communications (INFOCOM)*, 2010; 1–9.

[26] Tellenbach B, Burkhart M, Schatzmann D, Gugelmann D, Sornette D. Accurate network anomaly classification with generalized entropy metrics. *Computer Networks* 2011; 55(15):3485–3502.

[27] Brownlee N. One-way traffic monitoring with iatmon. *Proceedings of the 13th International Conference on Passive and Active Measurement*, 2012; 179–188.

[28] Glatz E, Dimitropoulos X. Classifying internet one-way traffic. *Proceedings of the 12th ACM Conference on Internet Measurement Conference*, 2012; 37–50.

[29] Fiadino P, D'Alconzo A, Bar A, Finamore A, Casas P. On the detection of network traffic anomalies in content delivery network services. *Teletraffic Congress (ITC), 2014 26th International*, 2014; 1–9, doi:10.1109/ITC.2014.6932930.

[30] Coluccia A, D'alconzo A, Ricciato F. Distribution-based anomaly detection via generalized likelihood ratio test: A general maximum entropy approach. *Comput. Netw.* Dec 2013; 57(17):3446–3462, doi:10.1016/j.comnet.2013.07.028. URL http://dx.doi.org/10.1016/j.comnet.2013.07.028.

[31] Gamer T. Collaborative anomaly-based detection of large-scale internet attacks. *Comput. Netw.* Jan 2012; 56(1):169–185, doi:10.1016/j.comnet.2011.08.015. URL http://dx.doi.org/10.1016/j.comnet.2011.08.015.

[32] Zhang Y. An adaptive flow counting method for anomaly detection in sdn. *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, ACM: New York, NY, USA, 2013; 25–30, doi:10.1145/2535372.2535411. URL http://doi.acm.org/10.1145/2535372.2535411.

[33] Mirkovic J, Reiher P. A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review* 2004; 34(2):39–53.

[34] Barnett RJ, Irwin B. Towards a taxonomy of network scanning techniques. *Proceedings of the 2008 South African Institute of Computer Scientists and Information Technologists on IT (SAICSIT)*, 2008; 1–7.

[35] Plonka D, Barford P. Network anomaly confirmation, diagnosis and remediation. *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing (CCC)*, 2009; 128–135.

[36] Borgnat P, Dewaele G, Fukuda K, Abry P, Cho K. Seven years and one day: Sketching the evolution of internet traffic. *Proceedings of the 28th Conference on Information Communications (INFOCOM)*, 2009; 711 –719.

[37] Allman M, Paxson V, Terrell J. A brief history of scanning. *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement (IMC)*, 2007; 77–82.

[38] Shon T, Moon J. A hybrid machine learning approach to network anomaly detection. *Inf. Sci.* September 2007; 177:3799–3821.

[39] Duffield N, Haffner P, Krishnamurthy B, Ringberg H. Rule-Based Anomaly Detection on IP Flows. *Proceedings of the 28th Conference on Information Communications (INFOCOM)*, 2009; 424–432.

[40] Bhuyan MH, Bhattacharyya DK, Kalita JK. Towards an unsupervised method for network anomaly detection in large datasets. *Computing and Informatics* 2014; 33(1):1–34. URL http://www.cai.sk/ojs/index.php/cai/article/view/909.

[41] Novakov S, Lung CH, Lambadaris I, Seddigh N. A hybrid technique using pca and wavelets in network traffic anomaly detection. *Int. J. Mob. Comput. Multimed. Commun.* Jan 2014; 6(1):17–53, doi:10.4018/ijmcmc.2014010102. URL http://dx.doi.org/10.4018/ijmcmc.2014010102.

[42] Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.* June 2004; 6:90–105.

[43] Fred ALN, Jain AK. Combining multiple clusterings using evidence accumulation. *IEEE Trans. Pattern Anal. Mach. Intell.* June 2005; 27:835–850.

[44] Cormode G, Muthukrishnan S. What's new: finding significant differences in network data streams. *Networking, IEEE/ACM Transactions on* 2005; 13:1219–1232.

[45] Jain AK. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters* 2010; 31:651–666.

[46] Strehl A, Ghosh J. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* March 2003; 3:583–617.

[47] Ester M, Kriegel HP, S J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, 1996.

[48] Agrawal R, Gehrke J, Gunopulos D, Raghavan P. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.* Jun 1998; 27(2):94–105.

[49] Dean J, Ghemawat S. Mapreduce: Simplified data processing on large clusters. *Commun. ACM* January 2008; 51(1):107–113.

[50] Müller E, Günnemann S, Assent I, Seidl T. Evaluating clustering in subspace projections of high dimensional data. *Proc. VLDB Endow.* August 2009; 2:1270–1281.

[51] Moise G, Zimek A, Kröger P, Kriegel HP, Sander J. Subspace and projected clustering: experimental evaluation and analysis. *Knowl. Inf. Syst.* November 2009; 21:299–326.