Unsupervised Detection of Network Attacks in the dark

Philippe Owezarski[1,2], Pedro Casas[3], and Johan Mazel[4]

[1]CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[2]Univ de Toulouse, LAAS; F-31400 Toulouse, France
[3]FTW-Telecommunication Research Center, Vienna, Austria
[4]NII-JFLI, Tokyo, Japan
Email: owe@laas.fr,casas@ftw.at,johanmazel@nii.ac.jp

Abstract. The unsupervised detection of network attacks represents an extremely challenging goal. Current methods rely on either very specialized signatures of previously seen attacks, or on expensive and difficult to produce labeled traffic data-sets for profiling and training. In this paper we present a completely unsupervised approach to detect attacks, without relying on signatures, labeled traffic, or training. The method uses robust clustering techniques to detect anomalous traffic flows. The structure of the anomaly identified by the clustering algorithms is used to automatically construct specific filtering rules that characterize its nature, providing easy-to-interpret information to the network operator. In addition, these rules are combined to create an anomaly signature, which can be directly exported towards standard security devices like IDSs, IPSs, and/or Firewalls. The clustering algorithms are highly adapted for parallel computation, which permits to perform the unsupervised detection and construction of signatures in an on-line basis. We evaluate the performance of this new approach to discover and to build signatures for different network attacks without any previous knowledge, using real traffic traces.

Key words: Anomaly Detection & Characterization, Robust Clustering, Automatic Generation of Signatures, Autonomous Security.

1 Introduction

The detection of network attacks is a paramount task for network operators in today's Internet. Denial of Service attacks (DoS), Distributed DoS (DDoS), network/host scans, and spreading worms or viruses are examples of the different attacks that daily threaten the integrity and normal operation of the network. The principal challenge in automatically detecting and analyzing network attacks is that these are a moving and ever-growing target.

Two different approaches are by far dominant in the literature and commercial security devices: signature-based detection and anomaly detection. Signature-based detection systems are highly effective to detect those attacks which they are programmed to alert on. However, they cannot defend the network against

unknown attacks. Even more, building new signatures is expensive and time-consuming, as it involves manual inspection by human experts. Anomaly detection uses labeled data to build normal-operation-traffic profiles, detecting anomalies as activities that deviate from this baseline. Such methods can detect new kinds of network attacks not seen before. Nevertheless, anomaly detection requires training to construct normal-operation profiles, which is time-consuming and depends on the availability of purely anomaly-free traffic data-sets. In addition, it is not easy to maintain an accurate and up-to-date normal-operation profile.

In this paper we present a completely unsupervised method to detect and characterize network attacks, without relying on signatures, training, or labeled traffic of any kind. Our approach relies on robust clustering algorithms to detect both well-known as well as completely unknown attacks, and to automatically produce easy-to-interpret signatures to characterize them, both in an on-line basis. The analysis is performed on packet-level traffic, captured in consecutive time slots of fixed length ΔT and aggregated in IP flows (standard *5-tuples*). IP flows are additionally aggregated at 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* ($l_{3,4,5}$: IPsrc/24, /16, /8), *destination Network Prefixes* ($l_{6,7,8}$: IPdst/24, /16, /8), and *traffic per Time Slot* (l_9: tpTS).

The complete detection and characterization algorithm runs in three successive stages. The first step consists in detecting an anomalous time slot where an attack might be hidden. For doing so, time series $Z_t^{l_i}$ are built for basic traffic metrics such as number of bytes, packets, and IP flows per time slot, using the 9 flow resolutions $l_{1...9}$. Any generic anomaly-detection algorithm $\mathcal{F}(.)$ based on time-series analysis [1–5] is then used on $Z_t^{l_i}$ to identify an anomalous slot. Time slot t_0 is flagged as anomalous if $\mathcal{F}(Z_{t_0}^{l_i})$ 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.

The unsupervised detection and characterization algorithm begins in the second stage, using as input the set of IP flows captured in the flagged time slot. The method uses robust clustering techniques based on Sub-Space Clustering (SSC) [11], Density-based Clustering [10], and Evidence Accumulation (EA) [12] to blindly extract the suspicious flows that compose the attack. In the third stage, the evidence of traffic structure provided by the clustering algorithms is used to produce filtering rules that characterize the detected attack and simplify its analysis. The characterization of an attack can be a hard and time-consuming task, particularly when dealing with unknown attacks. Even expert operators can be quickly overwhelmed if simple and easy-to-interpret information is not provided to prioritize the time spent in the analysis. To alleviate this issue, the most relevant filtering rules are combined into a new traffic signature that characterizes the attack in simple terms. This signature can ultimately be integrated to any standard security device to detect the attack in the future, which constitutes a major step towards autonomous security.

The remainder of the paper is organized as follows. Section 2 presents a short state of the art in the unsupervised anomaly detection field and describes our main contributions. Section 3 briefly describes the unsupervised detection algorithm that we have developed. Section 4 presents the automatic characterization algorithm, which builds easy-to-interpret signatures for the detected attacks. Section 5 presents the validation of our proposals, discovering and characterizing single source/destination and distributed network attacks in traffic traces from an operational backbone network. Section 6 evaluates the computational time of the unsupervised detection approach, considering the parallelization of the clustering algorithms. Finally, section 7 concludes the paper.

2 Related Work & Contributions

The problem of network attacks and anomaly detection has been extensively studied in the last decade. Most approaches analyze statistical variations of traffic volume-metrics (e.g., number of bytes, packets, or flows) and/or other 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., ARIMA, wavelets) on single-link traffic measurements [1, 2], PCA [7, 8] and Kalman filters [4] for network-wide anomaly detection, and sketches applied to IP-flows [3, 6].

Our approach falls within the unsupervised anomaly detection domain. Most work has been devoted to the Intrusion Detection field, targeting 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 [15–17] some relevant examples.

Our unsupervised algorithm has several advantages w.r.t. the state of the art: (i) 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 work from scratch, without any kind of calibration or previous knowledge. (ii) It combines robust clustering techniques to avoid general clustering problems such as sensitivity to initialization, specification of number of clusters, or structure-masking by irrelevant features. (iii) It automatically builds compact and easy-to-interpret signatures to characterize attacks, which can be directly integrated into any traditional security device. (iv) It is designed to work on-line, using the parallel structure of the proposed clustering approach.

3 Unsupervised Detection of Attacks

The unsupervised detection stage takes as input all the IP flows in the anomalous time slot, aggregated according to one of the different aggregation levels used in the first stage. Let $\mathbf{Y} = \{\mathbf{y}_1, .., \mathbf{y}_n\}$ be the set of n flows in the flagged time slot. Each flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic attributes or *features* on which the analysis is performed. The selection of these features is a key issue to any anomaly detection algorithm, and it becomes critical in the case of unsupervised

detection, because there is no additional information to select the most relevant set. In this paper we shall limit our study to detect and characterize well-known attacks, using a set of standard traffic features widely used in the literature. However, the reader should note that the approach can be easily extended to detect other types of attacks, considering different sets of traffic features. In fact, more features can be added to any standard list to improve detection and characterization results.

The set that we shall use here includes the following $m = 9$ 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 previous work on signature-based anomaly characterization [9], such simple traffic descriptors permit to describe standard network attacks such as DoS, DDoS, scans, and spreading worms/virus. Let $\mathbf{x}_i = (x_i(1), .., x_i(m)) \in \mathbb{R}^m$ be the corresponding vector of traffic features describing flow $\mathbf{y}_i$, and $\mathbf{X} = \{\mathbf{x}_1, .., \mathbf{x}_n\}$ the complete matrix of features, refereed to as the *feature space*.

The algorithm is based on clustering techniques applied to $\mathbf{X}$. The objective of clustering is to partition a set of unlabeled elements into homogeneous groups of similar characteristics, based on some measure of similarity. Our goal is to identify in $\mathbf{Y}$ the different aggregated flows that may compose the attack. For doing so, the reader should note that an attack may consist of either outliers (i.e., single isolated flows) or compact small-size clusters, depending on the aggregation level of flows in $\mathbf{Y}$. For example, a DDoS attack is represented as an outlier flow if the aggregation is done for IPdst, consisting of all the attacking IP flows sent towards the same victim. On the contrary, the attack is represented as a cluster if we use IPsrc flow-resolution. To avoid the lack of robustness of general clustering techniques, we have developed a parallel-multi-clustering approach, combining the notions of Density-based Clustering [10], Sub-Space Clustering [11], and Evidence Accumulation [12]. in what follows, we shall present the general idea behind the approach.

Instead of directly partitioning the complete feature space $\mathbf{X}$ using a traditional inter-flow similarity measure (i.e., the Euclidean distance), we do parallel clustering in N different sub-spaces $\mathbf{X}_i \subset \mathbf{X}$ of smaller dimensions, obtaining N different partitions P_i of the flows in $\mathbf{Y}$. Each sub-space $\mathbf{X}_i$ is constructed using only $r < m$ traffic features; this permits to analyze the structure of $\mathbf{X}$ from $N(m, r)$ different perspectives, using a finer-grained resolution. In particular, we do clustering in very-low dimensional sub-spaces, using $r = 2$. To deeply explore the complete feature space, we analyze all the r-combinations-obtained-from-m sub-spaces; hence, $N(m) = m(m-1)/2$. The information provided by the multiple partitions P_i is then combined to produce a new similarity measure between the flows in $\mathbf{Y}$, which has the paramount advantage of clearly highlighting both those outliers and small-size clusters that were simultaneously identified in different sub-spaces. This new similarity measure is finally used to easily extract the anomalous flows from the rest of the traffic.

4 Automatic Characterization of Attacks

The following task after the detection of a group of anomalous flows is to automatically produce a set of K filtering rules $f_k(\mathbf{Y})$, $k = 1, .., K$ to characterize them. In the one hand, such filtering rules provide useful insights on the nature of the anomaly, easing the analysis task of the network operator. On the other hand, different rules can be combined to construct a signature of the anomaly, which can be used to easily detect its occurrence in the future. To produce filtering rules $f_k(\mathbf{Y})$, the algorithm selects those sub-spaces $\mathbf{X}_i$ where the separation between the anomalous flows and the rest of the traffic is the biggest. We define two different classes of filtering rule: *absolute* rules $f_A(\mathbf{Y})$ and *relative* rules $f_R(\mathbf{Y})$. Absolute rules are only used in the characterization of small-size clusters, and correspond to the presence of dominant features in the flows of the anomalous cluster. An absolute rule for feature j has the form $f_A(\mathbf{Y}) = \{\mathbf{y}_i \in \mathbf{Y} : x_i(j) == \lambda\}$. 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 $\{\text{nICMP/nPkts} == 1\}$ makes sense (nICMP/nPkts corresponds to the fraction of ICMP packets).

On the other hand, relative filtering rules depend on the relative separation between anomalous and normal-operation flows. Basically, if the anomalous flows are well separated from the rest of the traffic in a certain partition P_i, then the features of the corresponding sub-space $\mathbf{X}_i$ are good candidates to define a relative filtering rule. A relative rule defined for feature j has the form $f_R(\mathbf{Y}) = \{\mathbf{y}_i \in \mathbf{Y} : x_i(j) < \lambda \text{ or } x_i(j) > \lambda\}$. We shall 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. Regarding relatives rules, their relevance is directly tied to the degree of separation between flows. In the case of outliers, we select the K features for which the normalized distance to the normal-operation traffic (statistically represented by the biggest cluster in each sub-space) is among the top-K biggest distances. In the case of small-size clusters, we rank the degree of separation to the rest of the clusters using the well-known Fisher Score (FS) [14], and select the top-K ranked rules. The FS basically measures the separation between clusters, relative to the total variance within each cluster. 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.

5 Experimental Evaluation

We evaluate the ability of the unsupervised algorithm to detect and to automatically construct a signature for different attacks in real traffic from the WIDE

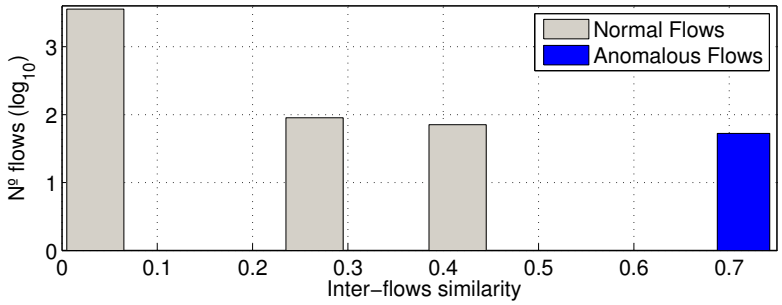
project data repository [18]. The WIDE network provides interconnection between different research institutions in Japan, as well as connection to different commercial ISPs and universities in the U.S.. Traffic consists of 15 minutes-long raw packet traces; the traces we shall work with consist of packets captured at one of the trans-pacific links between Japan and the U.S.. Traces are not labeled, thus our analysis will be limited to show how the unsupervised approach can detect and characterize different network attacks without using signatures, labels, or learning.

We shall begin by detecting and characterizing a distributed SYN network scan directed to many victim hosts under the same /16 destination network. Packets in $\mathbf{Y}$ are aggregated using IPdst/24 flow resolution, thus the attack is detected as a small-size cluster. The length of each time slot is $\Delta T = 20$ seconds. As we explained in section 3, the SSC-EA-based clustering algorithm constructs a new similarity measure between flows in $\mathbf{Y}$, using the multiple clustering results obtained from the different sub-spaces. Let us express this new similarity measure as a $n \times n$ matrix S, in which element $S(i, j)$ represents the degree of similarity between flows i and j. Figure 1.(a) depicts a histogram on the distribution of inter-flows similarity, according to S. The structure of flows in $\mathbf{Y}$ provided by S evidences the presence of a small isolated cluster in multiple subspaces. Selecting this cluster results in 53 anomalous IPdst/24 flows; a further analysis of the packets in these flows reveals multiple IP flows of SYN packets with the same IPsrc address and sequential IPdst addresses, scanning primary the same TCP port. Such a behavior is characteristic of a worm in the spreading phase.

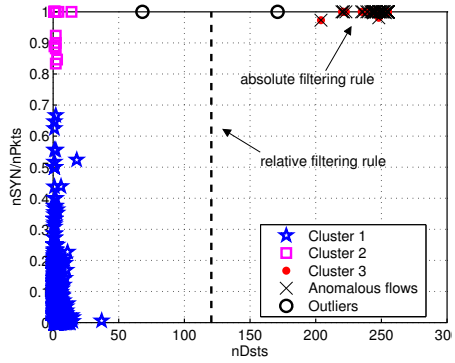
Regarding filtering rules, figures 1.(b,c) depict some of the partitions P_i where both absolute and top-K relative rules were produced. These involve the number of sources and destinations, and the fraction of SYN packets. Combining them produces a signature that can be expressed as $(\text{nSrcs} == 1) \wedge (\text{nDsts} > \lambda_1) \wedge (\text{nSYN/nPkts} > \lambda_2)$, where both λ_1 and λ_2 are obtained by separating clusters at half distance. Surprisingly enough, the extracted signature matches quite closely the standard signature used to detect such an attack in current signature-based systems [9]. The beauty and main advantage of our unsupervised approach relies on the fact that this new signature has been produced without any previous information about the attack or baseline traffic, and now it can be directly exported towards any security device to rapidly detect the same attack in the future.

Figures 1.(d,e) depict different rules obtained in the detection of a SYN DDoS attack. IP flows are now aggregated according to IPsrc resolution. The distribution analysis of inter-flows similarity w.r.t. S selects a compact cluster with the most similar flows, corresponding to the set of attacking hosts. The obtained signature can be expressed as $(\text{nDsts} == 1) \wedge (\text{nSYN/nPkts} > \lambda_3) \wedge (\text{nPkts/sec} > \lambda_4)$, which combined with the large number of identified sources $(\text{nSrcs} > \lambda_5)$ confirms the nature of a SYN DDoS attack. This signature is able to correctly isolate the most aggressive hosts of the DDoS attack, i.e., those with highest packet rate.

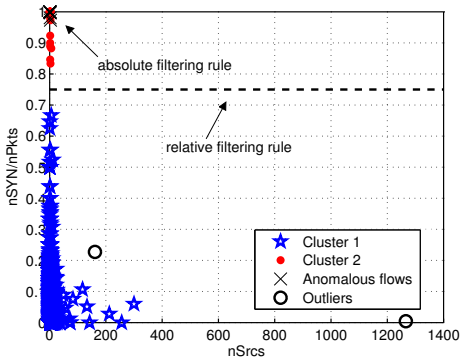
Figures 1.(f,g) depict the detection of an ICMP flooding DoS attack. Traffic is aggregated in IPdst flows, thus the attack is now detected as an outlier rather



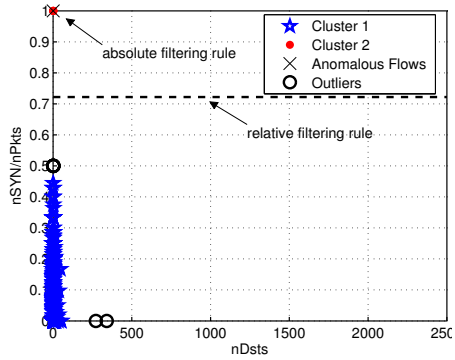
(a) Detecting a distributed SYN network scan using S.



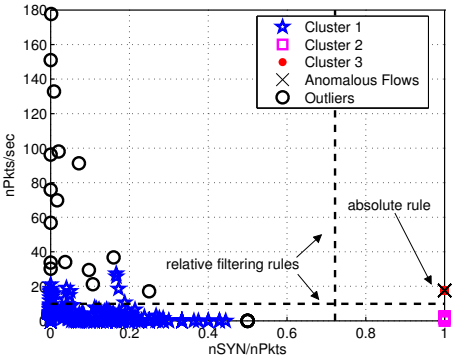
(b) SYN Network Scan (1/2)



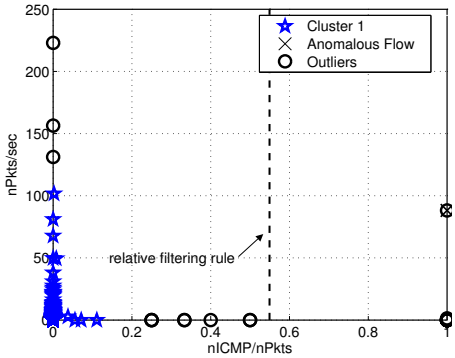
(c) SYN Network Scan (2/2)



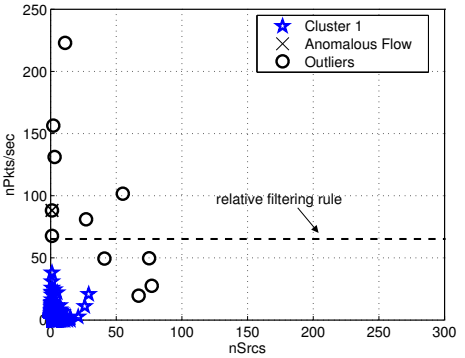
(d) SYN DDoS (1/2)



(e) SYN DDoS (2/2)



(f) ICMP flooding DoS (1/2)



(g) ICMP flooding DoS (2/2)

Fig. 1. Filtering rules for characterization of attacks in WIDE.

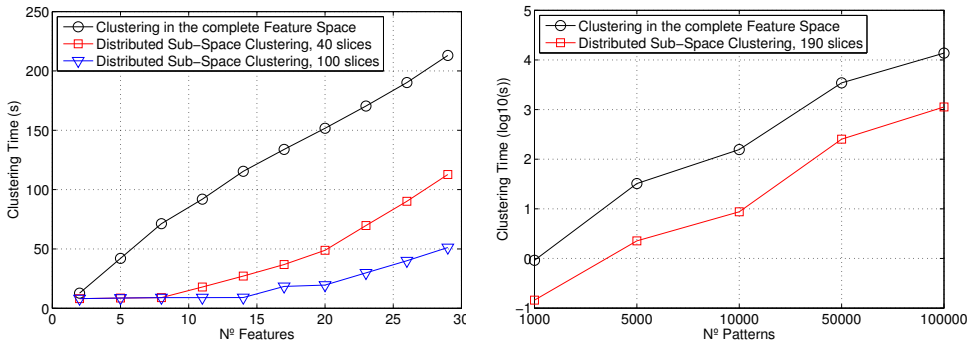
than as a small-size cluster. Absolute rules are not applicable in the case of outliers detection. Relative rules correspond to the separation of the outlier from the biggest cluster in each sub-space, which statistically represents normal-operation traffic. Besides showing typical characteristics of this attack, such as a high packet rate of exclusively ICMP packets from the same source host, both partitions show that the detected attack does not involve the largest elephant flows in the time slot. This emphasizes the ability of the algorithm to detect attacks that are not necessarily different from normal-operation traffic in terms of volume, but that they differ in other, less evident characteristics. The obtained signature can be expressed as $(\text{nICMP/nPkts} > \lambda_6) \wedge (\text{nPkts/sec} > \lambda_7)$.

6 Computational Time and Parallelization

The last issue that we analyze is the Computational Time (CT) of the algorithm. The SSC-EA-based algorithm performs multiple clusterings in $N(m)$ low-dimensional sub-spaces $\mathbf{X}_i \subset \mathbf{X}$. 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 number of features m and the number of aggregated flows n to analyze. Firstly, clustering is performed in very-low-dimensional sub-spaces, $\mathbf{X}_i \in \mathbb{R}^2$, which is faster than clustering in high-dimensional spaces [13]. 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: We shall use the term "slice" as a reference to a single computational entity.

Figure 2 depicts the CT of the SSC-EA-based algorithm, both (a) as a function of the number of features m used to describe traffic flows and (b) as a function of the number of flows n to analyze. Figure 2.(a) compares the CT obtained when clustering the complete feature space $\mathbf{X}$, referred to as $\text{CT}(\mathbf{X})$, against the CT obtained with SSC, varying m from 2 to 29 features. We analyze a large number of aggregated flows, $n = 10^4$, and use two different number of slices, $M = 40$ and $M = 100$. The analysis is done with traffic from the WIDE network, combining different traces to attain the desired number of flows. To estimate the CT of SSC for a given value of m and M, we proceed as follows: first, we separately cluster each of the $N = m(m-1)/2$ 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., $\text{CT}(\mathbf{X}_{\text{SSCwc}}) = \max_i \text{CT}(\mathbf{X}_i)$. Then, if $N \leqslant M$, we have enough slices to completely parallelize the SSC algorithm, and the total CT corresponds to the worst-case, $\text{CT}(\mathbf{X}_{\text{SSCwc}})$. On the contrary, if $N > M$, some slices have to cluster various sub-spaces, one after the other, and the total CT becomes $(N\%M + 1)$ times the worst-case $\text{CT}(\mathbf{X}_{\text{SSCwc}})$, where % represents integer division. The first interesting observation from figure 2.(a) regards the increase of $\text{CT}(\mathbf{X})$ when m increases, going from about 8 seconds for $m = 2$ to more than 200 seconds for $m = 29$. 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 architecture, 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 architecture with 100 slices, we can analyze 10000 aggregated flows in less than 20 seconds.



(a) Time vs. nï¿½ features. (b) Time vs. nï¿½ patterns.

Fig. 2. Computational Time as a function of nï¿½ of features and nï¿½ of flows to analyze. The number of aggregated flows in (a) is $n = 10000$. The number of features and slices in (b) is $m = 20$ and $M = 190$ respectively.

Figure 2.(b) compares CT($\mathbf{X}$) against CT($\mathbf{X}_{\text{SSCwc}}$) for an increasing number of flows n to analyze, using $m = 20$ traffic features and $M = N = 190$ slices (i.e., a completely parallelized implementation of the SSC-EA-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-EA-based algorithm can analyze up to 50000 flows with a reasonable CT, about 4 minutes in this experience. In the presented evaluations, the number of aggregated flows in a time slot of $\Delta T = 20$ seconds rounds the 2500 flows, which represents a value of CT($\mathbf{X}_{\text{SSCwc}}$) ≈ 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($\mathbf{X}_{\text{SSCwc}}$) ≈ 14.4 seconds.

7 Conclusions

The completely unsupervised algorithm for detection of network attacks that we have presented has many interesting advantages w.r.t. previous proposals. It uses exclusively unlabeled data to detect and characterize network attacks, without assuming any kind of signature, particular model, or canonical data distribution.

This allows to detect new previously unseen network attacks, even without using statistical-learning. By combining the notions of Sub-Space Clustering and multiple Evidence Accumulation, the algorithm avoids the lack of robustness of general clustering approaches, improving the power of discrimination between normal-operation and anomalous traffic. We have shown how to use the algorithm to automatically construct signatures of network attacks without relying on any kind of previous information.

Finally, we have evaluated the computational time of our algorithm. Results confirm that the use of the algorithm for on-line unsupervised detection and automatic generation of signatures is possible and easy to achieve for the volumes of traffic that we have analyzed. Even more, they show that if run in a parallel architecture, the algorithm can reasonably scale-up to run in high-speed networks.

Acknowledgments

This work is part of the ECODE and ONTIC-619633 projects, funded by the European Commission under the 7th Framework Programme.

References

1. P. Barford, J. Kline, D. Plonka, A. Ron, "A Signal Analysis of Network Traffic Anomalies", in *Proc. ACM IMW*, 2002.
2. J. Brutlag, "Aberrant Behavior Detection in Time Series for Network Monitoring", in *Proc. 14th Systems Administration Conference*, 2000.
3. B. Krishnamurthy et al., "Sketch-based Change Detection: Methods, Evaluation, and Applications", in *Proc. ACM IMC*, 2003.
4. A. Soule et al., "Combining Filtering and Statistical Methods for Anomaly Detection", in in *Proc. ACM IMC*, 2005.
5. G. Cormode, S. Muthukrishnan, "What's New: Finding Significant Differences in Network Data Streams", in *IEEE Trans. on Net.*, vol. 13 (6), pp. 1219-1232, 2005.
6. G. Dewaele et al., "Extracting Hidden Anomalies using Sketch and non Gaussian Multi-resolution Statistical Detection Procedures", in *Proc. SIGCOMM LSAD*, 2007.
7. A. Lakhina, M. Crovella, C. Diot, "Diagnosing Network-Wide Traffic Anomalies", in *Proc. ACM SIGCOMM*, 2004.
8. A. Lakhina, M. Crovella, C. Diot, "Mining Anomalies Using Traffic Feature Distributions", in *Proc. ACM SIGCOMM*, 2005.
9. G. Fernandes, P. Owezarski, "Automated Classification of Network Traffic Anomalies", in *Proc. SecureComm'09*, 2009.
10. M. Ester et al., "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", in *Proc. ACM SIGKDD*, 1996.
11. L. Parsons et al., "Subspace Clustering for High Dimensional Data: a Review", in *ACM SIGKDD Expl. Newsletter*, vol. 6 (1), pp. 90-105, 2004.
12. A. Fred, A. Jain, "Combining Multiple Clusterings Using Evidence Accumulation", in *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 27 (6), pp. 835-850, 2005.
13. A. Jain, "Data Clustering: 50 Years Beyond K-Means", in *Pattern Recognition Letters*, vol. 31 (8), pp. 651-666, 2010.
14. T. Jaakkola and D. Haussler, "Exploiting Generative Models in Discriminative Classifiers.", in *Advances in Neural Inf. Processing Sys. II*, pp. 487-493, 1998.
15. L. Portnoy, E. Eskin, S. Stolfo, "Intrusion Detection with Unlabeled Data Using Clustering", in *Proc. ACM DMSA Workshop*, 2001.
16. E. Eskin et al., "A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data", in *Applications of Data Mining in Computer Security*, Kluwer Publisher, 2002.
17. K. Leung, C. Leckie, "Unsupervised Anomaly Detection in Network Intrusion Detection Using Clustering", in *Proc. ACSC05*, 2005.
18. K. Cho, K. Mitsuya, A. Kato, "Traffic Data Repository at the WIDE Project", in *USENIX Annual Technical Conference*, 2000.