Sub-Space Clustering and Evidence Accumulation for Unsupervised Network Anomaly Detection

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

[1] CNRS; LAAS; 7 avenue du colonel Roche,
F-31077 Toulouse Cedex 4, France
[2] Universite de Toulouse; UPS, INSA, INP, ISAE; UT1, UTM, LAAS;
F-31077 Toulouse Cedex 4, France
{jmazel,pcasashe,owe}@laas.fr

Abstract. Network anomaly detection has been a hot research topic for many years. Most detection systems proposed so far employ a supervised strategy to accomplish the task, using either signature-based detection methods or supervised-learning techniques. However, both approaches present major limitations: the former fails to detect unknown anomalies, the latter requires training and labeled traffic, which is difficult and expensive to produce. Such limitations impose a serious bottleneck to the development of novel and applicable methods in the near future network scenario, characterized by emerging applications and new variants of network attacks. This work introduces and evaluates an unsupervised approach to detect and characterize network anomalies, without relying on signatures, statistical training, or labeled traffic. Unsupervised detection is accomplished by means of robust data-clustering techniques, combining Sub-Space Clustering and multiple Evidence Accumulation algorithms to blindly identify anomalous traffic flows. Unsupervised characterization is achieved by exploring inter-flows structure from multiple outlooks, building filtering rules to describe a detected anomaly. Detection and characterization performance of the unsupervised approach is extensively evaluated with real traffic from two different data-sets: the public MAWI traffic repository, and the METROSEC project data-set. Obtained results show the viability of unsupervised network anomaly detection and characterization, an ambitious goal so far unmet.

Keywords: Unsupervised Anomaly Detection & Characterization, Clustering, Clusters Isolation, Outliers Detection, Filtering Rules.

1 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 such as denials-of-service and worms spreading, 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 a moving target. It is difficult to precisely and permanently define the set of possible anomalies that may arise, especially in the case of network attacks, because new attacks as well a 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 structure, using the least amount of previous knowledge and information, ideally no information at all.

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 they are programmed to alert on. However, these systems cannot defend the network against new attacks, simply because they cannot recognize what they do not know. Furthermore, building new signatures is expensive, 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 these will naturally deviate from the baseline. Nevertheless, supervised-learning requires training, which is time-consuming and depends on the availability of labeled traffic data-sets.

Apart from detection, operators need to analyze and characterize network anomalies to take accurate countermeasures. The characterization of an anomaly is a hard and time-consuming task. The analysis may become a particular bottleneck when new anomalies are detected, because the network operator has to manually dig into many traffic descriptors to understand its nature. Even expert operators can be quickly overwhelmed if further information is not provided to prioritize the time spent in the analysis.

Contrary to current supervised approaches, we develop in this work a completely unsupervised method to detect and characterize network anomalies, without relying on signatures, training, or labeled traffic of any kind. The proposed approach permits to detect both well-known as well as completely unknown anomalies, and to automatically produce easy-to-interpret signatures that characterize them. The algorithm runs in three consecutive stages. Firstly, traffic is captured in consecutive time slots of fixed length ΔT and aggregated in IP flows (standard *5-tuples*). IP flows are additionally aggregated at different *flow-resolution* levels, using {IPaddress/netmask} as aggregation key. Aggregation is done either for IPsrc or IPdst. To detect an anomalous time slot, time-series Z_t are constructed for simple traffic metrics such as number of bytes, packets, and IP flows per time slot, using the different flow-resolutions. Any generic change-detection algorithm $\mathcal{F}(.)$ based on time-series analysis [1–5] is then applied to Z_t: at each new time slot, $\mathcal{F}(.)$ analyses the different time-series associated with each aggregation level, going from coarser to finer-grained resolution. Time slot t_0 is flagged as anomalous if $\mathcal{F}(Z_{t_0})$ triggers an alarm for any of the traffic metrics at any flow-resolution. Tracking anomalies from multiple metrics and at multiple aggregation levels (i.e. /8, /16, /24, /32 netmask) provides additional reliability to the change-detection algorithm, and permits to detect both single source-destination and distributed anomalies of very different characteristics.

The unsupervised detection and characterization algorithm begins in the second stage, using as input the set of flows in the flagged time slot. Our method uses robust and efficient clustering techniques based on Sub-Space Clustering (SSC) [8] and multiple Evidence Accumulation (EA) [9] to blindly extract the suspicious traffic flows that compose the anomaly. As we shall see, the simultaneous use of SSC and EA improves the power of discrimination to properly detect traffic anomalies. In the third stage of the algorithm, the evidence of traffic structure provided by the SSC and EA algorithms is further used to produce filtering rules that characterize the detected anomaly, which are ultimately combined into a new anomaly signature. This signature provides a simple and easy-to-interpret description of the problem, easing network operator tasks.

The remainder of the paper is organized as follows. Section 2 presents a very brief state of the art in the supervised and unsupervised anomaly detection fields, additionally describing our main contributions. Section 3 introduces the core of the proposal, presenting an in-depth description of the clustering techniques and detection algorithms that we use. Section 4 presents the automatic anomaly characterization algorithm, which builds easy-to-interpret signatures for the detected anomalies. Section 5 presents an exhaustive validation of our proposals, discovering and characterizing single source/destination and distributed network attacks in real network traffic from two different data-sets: the public MAWI traffic repository of the WIDE project [17], and the METROSEC project data-set [18]. Finally, section 6 concludes this paper.

2 Related Work and Contributions

The problem of network anomaly detection has been extensively studied during the last decade. Traditional approaches 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., ARIMA, wavelets) on single-link traffic measurements [1, 2], PCA [7] and Kalman filters [4] for network-wide anomaly detection, and Sketches applied to IP-flows [3, 6].

Our proposal falls within the unsupervised 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 [11–13] some relevant examples. In [11], 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, [12] reports improved results in the same data-set, using three different clustering algorithms: Fixed-Width clustering, an optimized version of k-NN, and one class SVM. Finally, [13] presents a combined density-grid-based clustering algorithm to improve computational complexity, obtaining similar detection results.

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 performs anomaly detection based not only on outliers detection, but also by identifying small-size clusters. This is achieved by exploring different levels of traffic aggregation, both at the source and destination of the traffic, which additionally permits to discover low-intensity and distributed attacks. Thirdly, 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. Fourthly, the algorithm performs clustering in low-dimensional spaces, using simple traffic descriptors such as number of source IP addresses or fraction of SYN packets. This simplifies the analysis and characterization of the detected anomalies, and avoids well-known problems of sparse spaces when working with high-dimensional data. Finally, the method combines the multiple evidence of an anomaly detected in different sub-spaces to produce an easy-to-interpret traffic signature that characterizes the problem. This permits to reduce the time spent by the network operator to understand the nature of the detected anomaly.

3 Unsupervised Anomaly Detection

The unsupervised anomaly detection stage 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 stage; 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 $\mathbf{Y} = \{\mathbf{y}_1, .., \mathbf{y}_n\}$ be the set of n flows in the flagged time slot, referred to as *patterns* in more general terms. Each flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic attributes or *features*. 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, referred 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 patterns into homogeneous groups of similar characteristics, based on some measure of similarity. Our particular goal is to identify and to isolate the different flows that compose the anomaly flagged in the first stage. Unfortunately, even if hundreds of clustering algorithms exist [10], it is very difficult to find a single one that can handle all types of cluster shapes and sizes. 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 & conquer* clustering approach, using the notions of clustering ensemble [15] and multiple clusterings combination. The idea is novel and appealing: why not taking advantage of the information provided by multiple partitions of $\mathbf{X}$ to improve clustering robustness and detection results? A clustering ensemble $\mathbf{P}$ consists of a set of multiple partitions P_i 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 new measure of similarity that better reflects natural groupings. There are different ways to produce a clustering ensemble. We use Sub-Space Clustering (SSC) [8] to produce multiple data partitions, applying the same clustering algorithm to N different sub-spaces $\mathbf{X}_i \subset \mathbf{X}$ of the original space.

3.1 Clustering Ensemble and Sub-Space Clustering

Each of the N sub-spaces $\mathbf{X}_i \subset \mathbf{X}$ is obtained by selecting k features from the complete set of m attributes. To deeply explore the complete feature space, the number of sub-spaces N that are analyzed corresponds to the number of k-combinations-obtained-from-m. Each partition P_i is obtained by applying DBSCAN [16] to sub-space $\mathbf{X}_i$. DBSCAN is a powerful density-based clustering algorithm that discovers clusters of arbitrary shapes and sizes [10], and it is probably one of the most common clustering algorithms along with the widely known k-means. DBSCAN 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. The clustering result provided by DBSCAN is twofold: a set of p clusters $\{C_1, C_2, .., C_p\}$ and a set of q outliers $\{o_1, o_2, .., o_q\}$. To set the number of dimensions k of each sub-space, 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 k-dimensional space, then it is also part of a cluster in any $(k-1)$ projections of this space". 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 k 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 [10], because high-dimensional spaces are usually sparse, making it difficult to distinguish between high and low density regions. Finally, results provided by low-dimensional clustering are more easy to visualize, which improves the interpretation of results by the network operator. We shall therefore use $k = 2$ in our SSC algorithm, which gives $N = m(m-1)/2$ partitions.

3.2 Combining Multiple Partitions Using Evidence Accumulation

Having produced the N partitions, the question now is how to use the information provided by the obtained clusters and outliers to isolate anomalies from normal-operation traffic. An interesting answer is provided in [9], where authors introduced the idea of multiple-clusterings Evidence Accumulation (EA). EA uses the clustering results of multiple partitions P_i to produce a new inter-patterns similarity measure which better reflects natural groupings. The algorithm follows a split-combine-merge approach to discover the underlying structure of data. In the **split** step, the N partitions P_i are generated, which in our case they correspond to the SSC results. In the **combine** step, a new measure of similarity between patterns is produced, using a *weighting* mechanism to combine the multiple clustering results. The underlying assumption in EA is that patterns belonging to a "natural" cluster are likely to be co-located in the same cluster in different partitions. Taking the membership of pairs of patterns to the same cluster as weights for their association, the N partitions are mapped into a $n \times n$ similarity matrix S, such that $S(i, j) = n_{ij}/N$. The value n_{ij} corresponds to the number of times that pair $\{\mathbf{x}_i, \mathbf{x}_j\}$ was assigned to the same cluster in the N partitions. Note that if a pair of patterns $\{\mathbf{x}_i, \mathbf{x}_j\}$ is assigned to the same cluster in each of the N partitions then $S(i, j) = 1$, which corresponds to maximum similarity.

We adapt the EA algorithm for our particular problem of unsupervised anomaly detection. For doing so, let us think about the particular structure of any general anomaly. 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 $\mathbf{Y}$. Let us take a flooding attack as an example; in the case of a DoS, all the packets of the attack will be aggregated into a single flow $\mathbf{y}_i$ targeting the victim, which will be represented as an outlier in $\mathbf{X}$. If we now consider a DDoS launched from β attackers towards a single victim, then the anomaly will be represented as a cluster of β flows if the aggregation is done for IPsrc/32, or as an outlier if the aggregation is done for IPdst/32. Taking into account that the number of flows in $\mathbf{Y}$ can reach a couple of thousands even for small time slots, the value of β would have to be too large to violate the assumption of small-size cluster.

We have developed two different EA methods to isolate small-size clusters and outliers: EA for small-clusters identification, EA4C, and EA for outliers identification, EA4O. Algorithm 1 presents the pseudo-code for both methods. In EA4C, we assign a stronger similarity weight when patterns are assigned to small-size clusters. The weighting function $w_k(n_t(k))$ used to update $S(i, j)$ at each iteration t takes bigger values for small values of $n_t(k)$, and goes to zero for big values of $n_t(k)$, being $n_t(k)$ the number of flows inside the co-assigned cluster for pair $\{\mathbf{x}_i, \mathbf{x}_j\}$. Parameters $n_{\min}$ and ρ specify the minimum number of flows that can be classified as a cluster and the neighborhood-density distance used by DBSCAN respectively. The parameter γ permits to set the slope of $w_k(n_t(k))$. Even tunable, we shall work with fixed values for $n_{\min}$, ρ, and γ, $n_{\min} = 20$, $\rho = 0.1$, and $\gamma = 5$, all empirically obtained.

Algorithm 1. EA4C & EA4O for Unsupervised Anomaly Detection

```
1: Initialization:
2:     Set similarity matrix $S$ to a null $n \times n$ matrix.
3:     Set dissimilarity vector $D$ to a null $n \times 1$ vector.
4: for $t = 1 : N$ do
5:     $P_t$ = DBSCAN $(\mathbf{X}_t, n_{\min,\rho})$
6:     Update $S(i,j)$, $\forall$ pair $\{\mathbf{x}_i, \mathbf{x}_j\} \in C_k$ and $\forall C_k \in P_t$:
7:         $w_k \leftarrow e^{-\gamma \frac{(n_t(k) - n_{\min})}{n}}$
8:         $S(i,j) \leftarrow S(i,j) + \frac{w_k}{N}$
9:     Update $D(i)$, $\forall$ outlier $o_i \in P_t$:
10:        $w_t \leftarrow \frac{n}{(n - n_{\max_t}) + \epsilon}$
11:        $D(i) \leftarrow D(i) + \mathrm{d_M}(o_i, C_{\max_t})\, w_t$
12: end for
```

In the case of EA4O, we define a dissimilarity vector D where the distances from all the different outliers to the centroid of the biggest cluster identified in each partition P_t are accumulated. We shall use $C_{\max_t}$ as a reference to this cluster. The idea is to clearly highlight those outliers that are far from the normal-operation traffic in the different partitions, statistically represented by $C_{\max_t}$. The weighting factor w_t takes bigger values when the size $n_{\max_t}$ of $C_{\max_t}$ is closer to the total number of patterns n, meaning that outliers are more rare and become more important as a consequence. The parameter ϵ is simply introduced to avoid numerical errors ($\epsilon = 1e^{-3}$). Finally, instead of using a simple Euclidean distance, we compute the Mahalanobis distance $\mathrm{d_M}(o_i, C_{\max_t})$ between the outlier and the centroid of $C_{\max_t}$, which is an independent-of-features-scaling measure of similarity.

In the final **merge** step, any clustering algorithm can be applied to matrix S or to vector D to obtain a final partition of $\mathbf{X}$ that isolates both small-size clusters and outliers. 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. This is simply achieved by comparing the values in S and D to a variable detection threshold.

4 Automatic Characterization of Anomalies

At this stage, the algorithm has identified a set of traffic flows in $\mathbf{Y}$ far out the majority of the traffic. The following task is to automatically produce a set of K filtering rules $f_k(\mathbf{Y})$, $k = 1, .., K$ to correctly isolate and characterize these flows. 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 detect its occurrence in the future, using a traditional signature-based detection system. Even more, this signature could eventually be compared against well-known signatures to automatically classify the anomaly.

In order 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. These rules do not depend on the separation between flows, and correspond to the presence of dominant features in the flows of the anomalous cluster. An absolute rule for a certain 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.

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 rest of the clusters 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. As regards 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 Mahalanobis distance to the normal-operation traffic 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), and select the top-K ranked rules. The FS measures the separation between clusters, relative to the total variance within each cluster. Given two clusters C_1 and C_2, the Fisher Score for feature i can be computed as:

$$F(i) = \frac{(\bar{x}_1(i) - \bar{x}_2(i))^2}{\sigma_1(i)^2 + \sigma_2(i)^2}$$

where $\bar{x}_j(i)$ and $\sigma_j(i)^2$ are the mean and variance of feature i in cluster C_j. In order to select the top-K relative rules, we take the K features i with biggest $F(i)$ value. 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 in Real Traffic

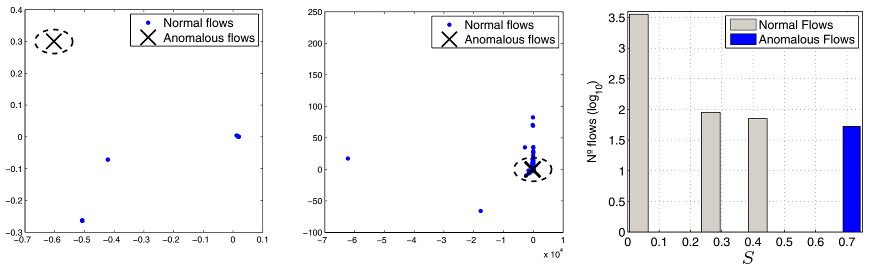
We evaluate the ability of the unsupervised algorithm to detect and to construct a signature for different attacks in real traffic from the public MAWI repository of the WIDE project [17]. The WIDE operational network provides interconnection between different research institutions in Japan, as well as connection to different commercial ISPs and universities in the U.S.. The traces we shall work with consist of 15 minutes-long raw packet traces collected at one of the trans-pacific links between Japan and the U.S.. Traces are not labeled, thus our analysis is limited to show the detection and characterization of different network attacks found by manual inspection in randomly selected traces, such as ICMP DoSs, SYN network scans, and SYN DDoS. Whenever possible, we refer to results obtained in [6], where some of these attacks have already been identified.

We shall also test the true positive and false positive rates obtained with annotated attacks, using different traffic traces from the METROSEC project [18]. 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). Additionally, we shall compare the performance of the algorithm against some traditional methods for unsupervised outliers detection presented in section 2, and also against the very well-known PCA and the sub-space approach [7].

In these evaluations we use the following list of $m = 9$ traffic features: number of source/destination IP addresses and ports (nSrcs, nDsts, nSrcPorts, nDstPorts), ratio of number of sources to number of destinations, packet rate (nPkts/sec), fraction of ICMP and SYN packets (nICMP/nPkts, nSYN/nPkts), and ratio of packets to number of destinations. According to previous work on signature-based anomaly characterization [14], such simple traffic descriptors permit to describe standard attacks such as DDoS, scans, and spreading worms. The list is by no means exhaustive, and more features can be easily plugged-in to improve results. In fact, a paramount advantage of our approach is that it is not tied to any particular set of features, and can therefore be generalized to any kind of traffic descriptors. For $m = 9$ features, we get $N = 36$ sub-spaces to analyze, a pretty small clustering ensemble that can be computed very fast.

5.1 Detecting Attacks in the Wild: MAWI Traffic

We begin by detecting and characterizing a distributed SYN network scan directed to many victim hosts under the same /16 destination network. The trace consists of traffic captured the 01/04/01. Traffic in $\mathbf{Y}$ is aggregated in IPdst/24 flows, thus we shall detect the attack as a small-size cluster. To appreciate the great advantage of using the SSC-EA-based algorithm w.r.t. a traditional approach, based on directly clustering the complete feature space, we shall compute a "traditional" similarity matrix S_{tra} for the n flows in $\mathbf{Y}$. Each element $S_{\text{tra}}(i, j)$ represents inter-flows similarity by means of the Euclidean distance. Figures 1.(a,b) depict the discrimination provided by both similarity matrices S and S_{tra}, using a Multi-Dimensional Scaling (MDS) analysis. The anomalous flows are mixed-up with the normal ones w.r.t. S_{tra}, and the discrimination using all the features at the same time becomes difficult. In the case of S, the flows



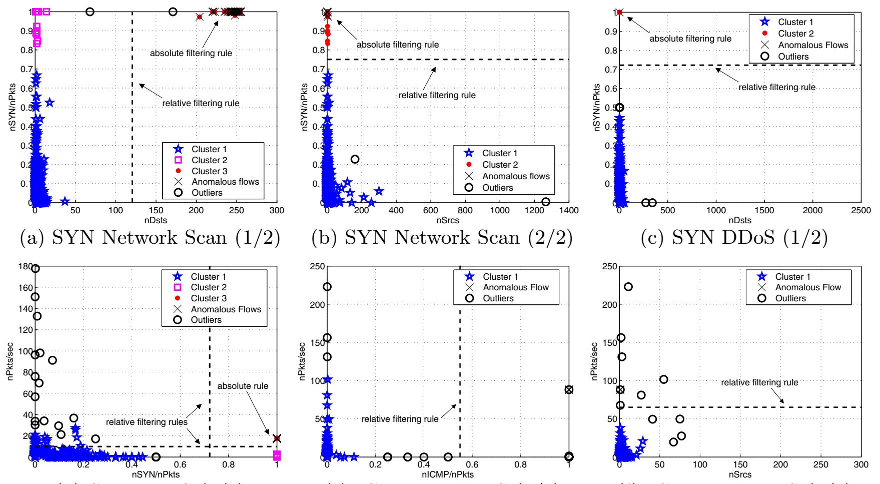
(a) SSC-EA Similarity Matrix (b) Traditional Similarity Matrix (c) SYN Network Scan in S

Fig. 1. MDS for traditional and SSC-EA-based clustering. A SYN network scan can be easily detected using the SSC-EA similarity measure.

that compose the attack are perfectly isolated from the rest, providing a powerful discrimination. As we explained before, the detection of the attack consists in identifying the most similar flows in S. Figure 1.(c) depicts a histogram on the distribution of inter-flows similarity, according to S. Selecting the most similar flows results in a compact cluster of 53 flows. A further analysis of the traffic that compose each of these flows reveals different IPdst/32 sub-flows of SYN packets with the same origin IP address, corresponding to the attacker.

Regarding filtering rules and the characterization of the attack, figures 2.(a,b) depict some of the partitions P_i where both absolute and relative filtering rules where found, corresponding to those with biggest Fisher Score. These rules involve the number of IP sources and destinations, and the fraction of SYN packets. Combining them produces a signature that can be expressed as (nSrcs == 1) $\wedge$ (nDsts $> \lambda_1$) $\wedge$ (nSYN/nPkts $> \lambda_2$), where λ_1 and λ_2 are two thresholds obtained by separating the clusters at half distance. This signature makes perfect sense, since the network scan uses SYN packets from a single attacking host to a large number victims. The signature permits to correctly identify all the flows of the attack. The beauty and 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.

The next two case-studies correspond to flooding attacks. For practical issues, traffic corresponds to different combined traces (14/10/03, 13/04/04, and 23/05/06). Figures 2.(c,d) depict different rules obtained in the detection of a SYN DDoS attack. Traffic is now aggregated in IPsrc/32 flows. The distribution analysis of inter-flows similarity w.r.t. S selects a compact cluster with the most similar flows, corresponding to traffic from the set of attacking hosts. The obtained signature can be expressed as (nDsts == 1) $\wedge$ (nSYN/nPkts $> \lambda_3$) $\wedge$ (nPkts/sec $> \lambda_4$), which combined with the large number of identified sources (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, namely those with highest packet rate. Figures 2.(e,f) depict the detection of an ICMP flooding DoS attack. Traffic is aggregated using aggregation index IPdst/32, thus the attack is detected as an outlier rather than as a small cluster. Besides showing



(a) SYN Network Scan (1/2) (b) SYN Network Scan (2/2) (c) SYN DDoS (1/2)

(d) SYN DDoS (2/2) (e) ICMP Echo DoS (1/2) (f) ICMP Echo DoS (2/2)

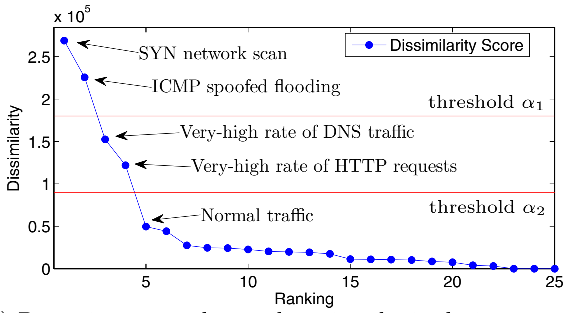
Fig. 2. Filtering rules for characterization of attacks in MAWI

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 represent the largest elephant flow in the time slot. This emphasizes the ability of the algorithm to detect low volume attacks, even of lower intensity than normal traffic. The obtained signature can be expressed as $(\text{nICMP/nPkts} > \lambda_6) \wedge (\text{nPkts/sec} > \lambda_7)$.

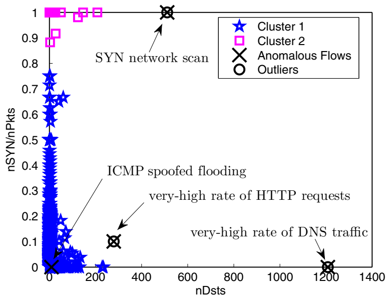
To conclude, we present the detection of two different attacks in one of the traces previously analyzed in [6], captured the 18/03/03. Traffic is aggregated in IPsrc/32 flows, and both attacks are detected as outliers. Figure 3.(a) shows the ordered dissimilarity values in D obtained by the EA4O method, along with their corresponding label. The first two most distant flows correspond to a highly distributed SYN network scan (more than 500 destination hosts) and an ICMP spoofed flooding attack directed to a small number of victims (ICMP redirect packets towards port 0). The following two flows correspond to unusual large rates of DNS traffic and HTTP requests; from there on, flows correspond to normal-operation traffic. The ICMP flooding attack and the two unusual flows are also detected in [6], but the SYN scan was missed by their method. Note that both attacks can be easily detected and isolated from the anomalous but yet legitimate traffic without false alarms, using for example the threshold α_1 on D. Figures 3.(b,c) depict the corresponding four flows in two of the N partitions produced by EA4O. As before, we verify the ability of the algorithm to detect network attacks that are not necessary the biggest elephant flows.

5.2 Detecting Attacks with Ground Truth: METROSEC Traffic

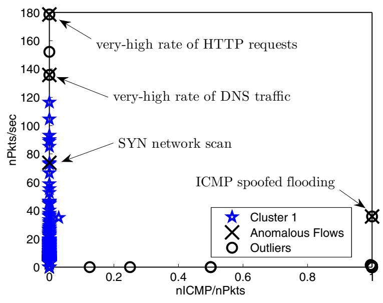
Figure 4 depicts the True Positives Rate (TPR) vs. the False Positives Rates (FTR) in the detection of 9 DDoS attacks in the METROSEC data-set.



(a) Detecting network attacks using dissimilarity vector D



(b) Scan and flooding attacks (1/2)



(c) Scan and flooding attacks (2/2)

Fig. 3. Detection and analysis of network attacks in MAWI

Detection is performed with traffic aggregated at the IPdst/32 level. Traffic corresponds to different combined traces (24/11/04, 07-09-10/12/04, and 11/04/06). The ROC plot is obtained by comparing the sorted dissimilarities in D to a variable detection threshold. From the 9 documented attacks, 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%). EA4O correctly detects 8 out of the 9 attacks without false alarms. The detection of the very-low intensity attack is more difficult; however, the 9 attacks are correctly detected with a very low FPR, about 1.2%.

We compare the performance of our approach against three "traditional" approaches: DBSCAN-based, k-means-based, and PCA-based outliers detection. The first two consist in applying either DBSCAN or k-means to the complete feature space $\mathbf{X}$, identify the largest cluster $C_{\max}$, and compute the Mahalanobis distance of all the flows lying outside $C_{\max}$ to its centroid. The ROC is finally obtained by comparing the sorted distances to a variable detection threshold. These approaches are similar to those used in previous work [11–13]. In the PCA-based approach, PCA and the sub-space methods [7] are applied to the complete matrix $\mathbf{X}$, 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 2 principal components to describe the normal sub-space.

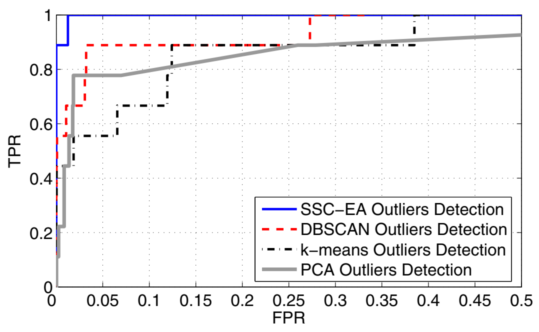


Fig. 4. DDoS detection in METROSEC. EA4O detects even low intensity attacks with a very-low false alarm rate, which is not achieved by traditional approaches.

Obtained results permit to evidence the great advantage of using the SSC-EA-based algorithm in the clustering step w.r.t. to traditional approaches. In particular, all the approaches used in the comparison fail to detect the smallest attack with a reasonable false alarm rate. Both clustering algorithms based either on DBSCAN or k-means get confused by masking features when analyzing the complete feature space $\mathbf{X}$. The PCA approach shows to be not sensitive enough to discriminate both low-intensity and high-intensity attacks, using the same representation for normal traffic.

6 Concluding Remarks

The completely unsupervised anomaly detection algorithm that we have presented 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. Despite using ordinary clustering techniques, the algorithm avoids the lack of robustness of general clustering approaches, by combining the notions of Sub-Space Clustering and multiple Evidence Accumulation. The Sub-Space Clustering approach also permits to obtain easy-to-interpret results, providing insights and explanations about the detected anomalies to the network operator. Even more, clustering in low-dimensional feature spaces provides results that can be visualized by standard techniques, which improves the assimilation of results.

We have verified the effectiveness of our proposal to detect and isolate real single source/destination and distributed network attacks in real traffic traces from different networks, all in a completely blind fashion, without assuming any particular traffic model, significant clustering parameters, or even clusters structure beyond a basic definition of what an anomaly is. Additionally, we have shown detection results that outperform traditional approaches for outliers detection, providing a stronger evidence of the accuracy of the SSC-EA-based method to detect network anomalies.

Acknowledgments

This work has been done in the framework of the ECODE project, funded by the European commission under grant FP7-ICT-2007-2/223936.

References

1. Barford, P., et al.: A Signal Analysis of Network Traffic Anomalies. In: Proc. ACM IMW (2002)
2. Brutlag, J.: Aberrant Behavior Detection in Time Series for Network Monitoring. In: Proc. 14th Systems Administration Conference (2000)
3. Krishnamurthy, B., et al.: Sketch-based Change Detection: Methods, Evaluation, and Applications. In: Proc. ACM IMC (2003)
4. Soule, A., et al.: Combining Filtering and Statistical Methods for Anomaly Detection. In: Proc. ACM IMC (2005)
5. Cormode, G., et al.: What's New: Finding Significant Differences in Network Data Streams. IEEE Trans. on Networking 13(6), 1219–1232 (2005)
6. Dewaele, G., et al.: Extracting Hidden Anomalies using Sketch and non Gaussian Multiresolution Statistical Detection Procedures. In: Proc. SIGCOMM LSAD (2007)
7. Lakhina, A., et al.: Diagnosing Network-Wide Traffic Anomalies. In: Proc. ACM SIGCOMM (2004)
8. Parsons, L., et al.: Subspace Clustering for High Dimensional Data: a Review. ACM SIGKDD Expl. Newsletter 6(1), 90–105 (2004)
9. Fred, A., et al.: Combining Multiple Clusterings Using Evidence Accumulation. IEEE Trans. Pattern Anal. and Machine Intel. 27(6), 835–850 (2005)
10. Jain, A.K.: Data Clustering: 50 Years Beyond K-Means. Pattern Recognition Letters 31(8), 651–666 (2010)
11. Portnoy, L., et al.: Intrusion Detection with Unlabeled Data Using Clustering. In: Proc. ACM DMSA Workshop (2001)
12. Eskin, E., et al.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. In: Apps. of Data Mining in Comp. Sec., Kluwer Publisher, Dordrecht (2002)
13. Leung, K., et al.: Unsupervised Anomaly Detection in Network Intrusion Detection Using Clustering. In: Proc. ACSC 2005 (2005)
14. Fernandes, G., et al.: Automated Classification of Network Traffic Anomalies. In: Proc. SecureComm 2009 (2009)
15. Strehl, A., et al.: Cluster Ensembles - A Knowledge Reuse Framework For Combining Multiple Partitions. Jour. Mach. Learn. Res. 3, 583–617 (2002)
16. Ester, M., et al.: A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. ACM SIGKDD (1996)
17. Cho, K., et al.: Data Repository at the WIDE Project. In: USENIX ATC (2000)
18. METROlogy for SECurity and QoS, `http://laas.fr/METROSEC`