Coping with 0-Day Attacks through Unsupervised Network Intrusion Detection

Pedro Casas, Johan Mazel, Philippe Owezarski

Coping with 0-Day Attacks through Unsupervised Network Intrusion Detection

Pedro Casas
FTW - Telecommunications Research Center
Vienna, Austria
casas@ftw.at

Johan Mazel
NII/JFLI
Tokyo, Japan
johanmazel@nii.ac.jp

Philippe Owezarski
CNRS/LAAS; Université de Toulouse
Toulouse, France
owe@laas.fr

***Abstract*—Traditional Network Intrusion Detection Systems (NIDSs) rely on either specialized signatures of previously seen attacks, or on expensive and difficult to produce labeled traffic datasets for profiling and training. Both approaches share a common downside: they require the knowledge provided by an external agent, either in terms of signatures or as normal-operation profiles. In this paper we describe UNIDS, an Unsupervised NIDS capable of detecting 0-day attacks, i.e., network attacks for which no signature is yet available, without using any kind of signatures, labeled traffic, or training. UNIDS uses a novel unsupervised outliers detection approach based on Sub-Space Clustering and Multiple Evidence Accumulation techniques to pin-point different kinds of network intrusions and attacks such as DoS/DDoS, probing attacks, propagation of worms, buffer overflows, illegal access to network resources, etc. In this paper we make the strong point that the de-facto approach for NIDS, namely the application of rule-based detection techniques, can be highly harmful for the protected network in case of 0-day attacks. In contrast, we show how UNIDS can work as a complementary system to current NIDS to detect the occurrence of previously unseen attacks. For doing so, we compare the performance of a standard rule-based NIDS against UNIDS to detect 0-day attacks in the well-known KDD99 dataset. In addition, we also compare the performance of UNIDS against other popular unsupervised detection techniques to detect attacks in traces collected at two operation networks.**

***Keywords*—NIDS, Unsupervised Machine Learning, Clustering, C4.5 Decision Trees, DDoS, Buffer Overflow attacks, Probing attacks, KDD99 Dataset.**

I. Introduction

The detection of network attacks is a paramount task for network operators in today's Internet. Distributed Denial of Service attacks (DDoS), buffer overflow attacks, network/host probing/scanning activities, and spreading worms or viruses are examples of the different threats that daily compromise the integrity and normal operation of the network. The principal challenge in automatically detecting network attacks is that these are a moving and ever-growing target [2].

Network Intrusion Detection Systems (NIDS) are the battle-horse of network-based security. Two different approaches are by far dominant in the research literature and commercial security devices: signatures-based or **misuse detection** (*detect what I know*) and **anomaly detection** (*detect what it is different from what I know*). Misuse detection is the de-facto approach used in most IDSs. When an attack is discovered, generally after its occurrence during a diagnosis phase, the associated malicious pattern is coded as a *signature* by human experts, which is then used to detect a new occurrence of the same attack. To avoid costly and time-consuming human intervention, signatures can also be constructed by supervised machine-learning techniques, using instances of the discovered attack to build a detection model for it. A standard technique for constructing such signatures-based or *rule-based* NIDS is to apply decision tree learning [6], [7]. Misuse detection systems are highly effective to detect those attacks which they are programmed to alert on. However, they cannot defend the network against new attacks, because they cannot recognize those attacks which do not match its list of signatures.

Anomaly detection uses instances of normal-operation traffic to build normal-operation profiles, detecting anomalies as activities that deviate from this baseline. Such methods can detect new kinds of network attacks not seen before. Nevertheless, they require training to construct profiles, which is time-consuming and depends on the availability of anomaly-free traffic instances. In addition, it is not easy to maintain an accurate and up-to-date normal-operation profile.

In our previous work we introduced UNADA [1], an unsupervised network anomaly detection system capable of detecting general network anomalies without relying on signatures, training, or labeled traffic instances of any kind. In this paper we extend UNADA to the NIDS domain, specifically targeting the detection of 0-day network attacks. The term "0-day attack" is taken from the software security domain; in software applications, a 0-day attack describes a software vulnerability which was not originally identified by the developer, and which is exploited by the attacker to launch an attack before the software fix or patch is applied on the involved systems. In this paper, we refer to 0-day attacks as those attacks for which there is no signature already available to detect it.

The system we describe and evaluate is referred to as UNIDS, an Unsupervised NIDS. Based on the observation that attacks, and particularly the most difficult ones to detect, are contained in a small fraction of traffic instances with respect to normal-operation traffic [3] (we show that this hypothesis can always be verified by using traffic aggregation), their unsupervised detection consists in identifying *outliers*, i.e. instances that are remarkably different from the majority. UNIDS relies on robust clustering techniques based on Sub-Space Clustering (SSC) [10], Density-based Clustering [9], and Evidence Accumulation (EA) [11] to blindly extract the instances that compose the attack.

The main point we make on this paper is that solely using a standard rule-based NIDS can be highly harmful for the protected network in case of 0-day attacks, as these will most probably go unnoticed and reach their target before the security system might react. UNIDS provides a proactive approach for network security, and can complement standard NIDS to identify the occurrence of new attacks. To support this claim, we compare the ability of UNIDS to detect 0-day attacks with respect to a rule-based NIDS based on decision trees. Decision trees permit to construct comprehensive signatures for network attacks in the form of multiple filtering-rules, using a graph structure. The comparison is done in the well known KDD99 network attacks dataset[1]. KDD99 is an artificially generated dataset containing different types of intrusions, and even if it has been highly questioned in the past as not being representative of real network attacks, it serves the point of this paper by providing two different datasets, each of them containing different types of attacks. This allows to build the NIDS on the basis of the first dataset, and use the second one as a set containing multiple 0-day attacks.

The remainder of the paper is organized as follows. Section II presents a brief state of the art in the supervised and unsupervised attacks detection field. Section III describes the main components of UNIDS, whereas section IV briefly describes a standard NIDS based on decision trees. Section V evaluates the ability of UNIDS to detect 0-day attacks in the well-known KDD99 network attacks dataset, comparing its performance against an extensively investigated NIDS based on decision trees. Section VI compares the performance of UNIDS against previously proposed unsupervised approaches. Finally, section VII concludes the paper.

II. Related Work

The problem of network attacks and intrusions detection has been extensively studied in the last decade. Most NIDS are based on rule-based misuse detection techniques, being Bro[2] and SNORT[3] the two most celebrated open-source systems in the literature. Most of the different techniques used in the NIDS field are summarized in [4], [5]. A particularly studied approach in recent years for misuse-detection that we shall use in the paper consists in the use of decision trees [6], [7]. Decision trees are one of the most powerful and simple data mining methods for decision-making; a decision tree uses a tree-like graph consisting of branch nodes that represent a choice among different alternatives, and leaves representing a class for the evaluated data (i.e. *attack* or *normal traffic*).

Our approach falls within the unsupervised attacks detection domain. The vast majority of the unsupervised detection schemes proposed in the literature are based on clustering and outliers detection, being [13]–[15] some relevant examples. In [13], authors use a single-linkage hierarchical clustering method to cluster data from the KDD99 dataset, based on the standard Euclidean distance for inter-patterns similarity. Reference [14] reports improved results in the same dataset, using three different clustering algorithms: Fixed-Width clustering, an optimized version of k-NN, and one class SVM. Reference [15] presents a combined density-grid-based clustering algorithm to improve computational complexity, obtaining similar detection results. PCA and the sub-space approach is another well-known unsupervised anomaly detection technique, used in [8] to detect network-wide traffic anomalies in highly aggregated traffic flows.

UNIDS consists of the same techniques we introduced in UNADA [1]. The main contribution of this paper is that of extending UNADA to the network security field, and more specifically, showing tangible evidence that it can provide reliable results in the event of 0-day attacks, in which we show that standard NIDS fail.

III. The UNIDS Approach

UNIDS works on packets captured on a single link and aggregated into traffic "flows". The definition of a flow is not fixed and any kind of traffic flows can be analyzed by the system: for example, a flow may consist of traditional 5-tuples, packets aggregated at source or destination IP address, etc. Without loss of generality, let $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ be the set of n flows under analysis. Each flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic attributes or *features*, like number of packets, packet rate, flow duration, etc. Let $\mathbf{x}_i \in \mathbb{R}^m$ be the vector of traffic features describing flow $\mathbf{y}_i$, and $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{n \times m}$ the complete matrix of features, referred to as the *feature space*.

UNIDS ranks the degree of abnormality of each of the flows by applying clustering and outliers analysis techniques on $\mathbf{X}$. Samples that do not belong to any of the obtained clusters are classified as *outliers*. Our particular goal is to identify those outliers that are remarkably different from the rest of the samples, additionally ranking how much different these are. The most appropriate approach to find outliers is to properly identify clusters. Unfortunately, even if hundreds of clustering algorithms exist [12], 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.

To overcome this limitation, we developed in [1] a novel clustering approach, based on the concept of Sub-Space Clustering (SSC) [10]. In a nutshell, the approach combines the results obtained from clustering multiple sub-spaces $\mathbf{X}_i$ of the original feature space $\mathbf{X}$ to improve clustering robustness and reduce the effects of noisy data. For the sake of completeness, we describe the main components of UNIDS next.

A. 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 the standard DBSCAN [9] clustering algorithm to sub-space $\mathbf{X}_i$. The results provided by applying DBSCAN to sub-space $\mathbf{X}_i$ are twofold: a set of $p(i)$ clusters $\{C_1^i, C_2^i, .., C_{p(i)}^i\}$ and a set of $q(i)$ outliers $\{o_1^i, o_2^i, .., o_{q(i)}^i\}$. To set the number of dimensions k of each sub-space, we take a

[1]https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[2]http://www.bro-ids.org
[3]http://www.snort.org

Algorithm 1 Evidence Accumulation for Ranking Outliers

Initialization:
  Set dissimilarity vector D to a null $n \times 1$ vector
  Set smallest cluster-size $n_{\min} = \alpha \, . \, n$
for $i = 1 : N$ **do**
  Set density neighborhood δ_i for DBSCAN
  P_i = DBSCAN $(\mathbf{X}_i, \delta_i, n_{\min})$
  Update $D(j)$, $\forall$ outlier $o_j^i \in P_i$:
    $w_i \leftarrow \dfrac{n}{(n - n_{\max_i}) + \epsilon}$
    $D(j) \leftarrow D(j) + \mathrm{d_M}(o_j^i, C_{\max}^i)\, w_i$
end for
Rank flows: D_{rank} = sort(D)

very useful property of monotonicity in clustering sets, known as the downward closure property: if a collection of elements 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 interesting 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 [12], because high-dimensional spaces are usually sparse. Finally, clustering multiple low-dimensional sub-spaces provides a finer-grained analysis, which improves the ability of UNIDS to detect attacks of very different characteristics. We shall therefore use $k = 2$ for SSC, which gives $N = m(m-1)/2$ partitions.

B. Ranking Outliers using Evidence Accumulation

Having produced the N partitions, we use the notions of Evidence Accumulation Clustering (EAC) [11] to combine the obtained results in a meaningful way. EAC uses the clustering results of multiple partitions P_i to produce a new inter-samples similarity measure that better reflects their natural groupings. UNIDS implements a particular algorithm for Evidence Accumulation, called Evidence Accumulation for Ranking Outliers (EA4RO) [1]. EA4RO constructs a dissimilarity vector $D \in \mathbb{R}^n$ in which it accumulates the distance between the different outliers o_j^i found in each sub-space $i = 1, .., N$ and the centroid of the corresponding sub-space-biggest-cluster $C_{\max}^i$. The idea is to clearly highlight those flows that are far from the normal-operation traffic at each of the different sub-spaces, statistically represented by $C_{\max}^i$.

Algorithm 1 presents a pseudo-code for EA4RO. The different parameters used by EA4RO are automatically set by the algorithm itself. The first two parameters are used by the density-based clustering algorithm: $n_{\min}$ specifies the minimum number of flows that can be classified as a cluster, while δ_i indicates the maximum neighborhood distance of a sample to identify dense regions. $n_{\min}$ is set at the initialization of the algorithm, simply as a fraction α of the total number of flows n to analyze (we take $\alpha = 5\%$ of n). δ_i is set as a fraction of the average distance between flows in sub-space $\mathbf{X}_i$ (we take a fraction $1/10$), which is estimated from 10% of the flows, randomly selected. This permits to fast-up computations. The weighting factor w_i is used as an outlier-boosting parameter, as it gives more relevance to those outliers

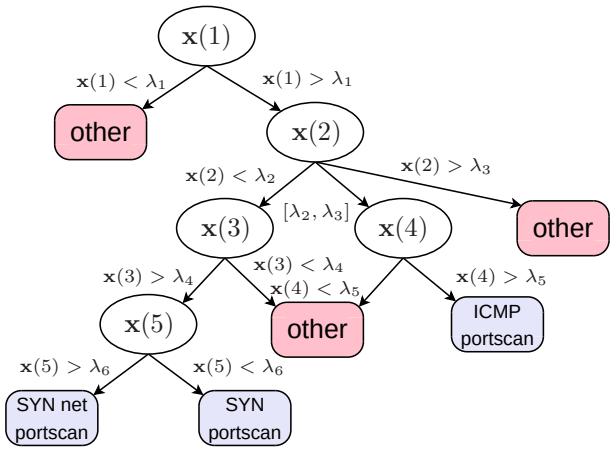


Figure 1. A decision tree for standard rule-based network intrusion detection. The tree targets the detection of SYN/ICMP scan activity.

that are "less probable": w_i takes bigger values when the size $n_{\max_i}$ of cluster $C_{\max}^i$ is closer to the total number of flows n. Finally, instead of using a simple Euclidean distance as a measure of dissimilarity, we compute the Mahalanobis distance $\mathrm{d_M}$ between outliers and the centroid of the biggest cluster. The Mahalanobis distance takes into account the correlation between samples, dividing the standard Euclidean distance by the variance of the samples. This permits to boost the degree of abnormality of an outlier when the variance of the samples is smaller. In the last part of EA4RO, flows are ranked according to the dissimilarity obtained in D, and detection is finally done as a binary thresholding operation: if $D_{rank} > T_h$, the system flags an anomaly in flow $\mathbf{y}_i$. The computation of the detection threshold T_h is simply achieved by finding the value for which the slope of the sorted dissimilarity values in D_{rank} presents a major change.

IV. Rule-based NIDS using C4.5 Decision Trees

Rule-based NIDS consists of a set of signatures defined for the set of attacks it is programmed to detect. The system functioning is straightforward: it compares the evaluated instance against the set of stored signatures, and outputs the name of the attack in case it finds a matching signature. However, if the set of stored signatures does not contain a specific one matching the evaluated instance, the instance is classified as normal. In a nutshell, rule-based NIDS are classification system, where the null class is assumed as normal operation.

An intuitive and efficient way of representing and building rule-based NIDS is through the usage of decision trees [6], [7]. A decision tree is a classification algorithm that classifies instances by repeatedly partitioning the feature space $\mathbf{X}$, so as to build a tree whose nodes are as pure as possible (i.e., they contain instances of a single class). The classification of a new instance is achieved by moving from top to bottom along the branches of the tree, starting from the root node, until a terminal node is reached. Figure 1 depicts an example tree for detecting SYN/ICMP scan activity. Note that the tree is incomplete and is only used for exemplifying the approach. If we define the features vector $\mathbf{x} = \{\mathbf{x}(1), \mathbf{x}(2), \mathbf{x}(3), \mathbf{x}(4), \mathbf{x}(5)\}$ describing a set of flows, where $\mathbf{x}(1)$ is the number of different dst. ports, $\mathbf{x}(2)$ the number of different src IPs, $\mathbf{x}(3)$ the fraction of SYN packets, $\mathbf{x}(4)$ the fraction of ICMP packets

and $\mathbf{x}(5)$ the number of different dst /24 subnets, then a signature or rule to detect SYN network scans can be written as: $\mathbf{x}(1) > \lambda_1 \wedge \mathbf{x}(2) < \lambda_2 \wedge \mathbf{x}(3) > \lambda_4 \wedge \mathbf{x}(5) > \lambda_6$, where the thresholds λ_i are obtained from a training set of labeled instances.

The learning procedure for building a decision tree iteratively selects the features that minimize the impurity of the immediate descendant nodes. The most popular impurity measure is the entropy impurity:

$$\begin{aligned} i(N) &= -\sum_{j=1}^{c} P(w_j)\log_2 P(w_j),\ \ i(N) \in [0, \log_2(c)] \\ P(w_j) &= \%\text{ instances at node } N \in w_j \end{aligned}$$

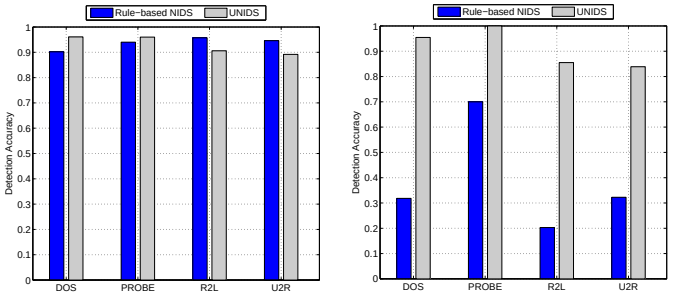
where w_j is the j-th class of the classification problem (e.g., ICMP portscan, SYN netscan, etc.) and c the number of classes. At each step, the learning procedure creates a new node by taking the feature that maximizes the $\Delta i(N)$. Growing the tree to the minimum impurity may cause over-fitting. For this reason, a post-pruning algoruthm is run at the end of the learning to reduce over-fitting.

Decision trees are simple yet very fast and effective. They are easy to interpret, and directly provide filtering rules. In addition, decision trees explicitly show the importance of different features, as the learning algorithm automatically chooses the most discriminating features. The C4.5 decision tree is the most frequently used algorithm [6], [7], so we shall design the benchmarking NIDS based on such trees.

V. UNIDS vs Rules-based NIDS

To show tangible evidence on how UNIDS outperforms standard NIDSs in detecting previously unseen attacks, we evaluate the performance of UNIDS to detect network attacks in the well-known and widely used KDD99 network attacks dataset. We compare its performance with the one obtained by an extensively investigated rule-based NIDS based on decision trees [6], [7].

The KDD99 dataset contains a wide variety of intrusions simulated in a military network environment. Traffic consists of packets aggregated into connections, being a connection a flow of TCP packets between a source and a destination IP address. Simulated attacks include DoS attacks, unauthorized access from a remote machine - R2L attacks (e.g. password guessing), unauthorized access to super-user privileges - U2R attacks (e.g. buffer overflows), and probing attacks (e.g. port scanning). Each connection or flow is described by a set of $m = 41$ features (e.g. number of bytes, TCP flags, failed remote-login attemps, etc.) and a label indicating either the name of the attack or if the flow corresponds to normal-operation traffic. The KDD99 dataset consists of two independent datasets: a training dataset and a testing dataset. The testing dataset is not drawn from the same probability distributions as the training dataset, and it includes specific attack types not present in the training data. As such, these new attacks can be considered as 0-day attacks in our evaluations. The datasets contain a total of 24 training attack types split in the aforementioned four categories, and 14 attack types present in the testing dataset only.



(a) Training dataset (b) Testing Dataset

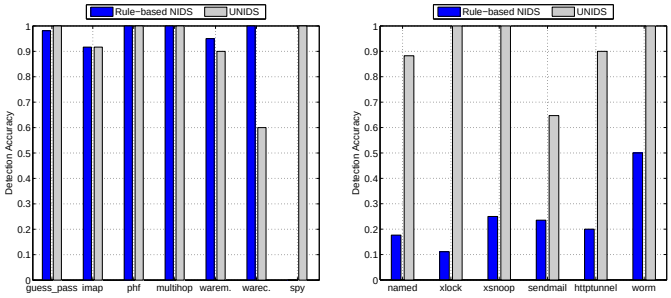
Figure 2. UNIDS vs Rule-based NIDS in KDD99. The rule-based NIDS is not capable of detecting most of the attacks not present in the training dataset. On the other hand, UNIDS can detect a major share of such 0-day attacks.

DoS and probing attacks in KDD99 are represented by a large number of flows, in some cases even more flows than those corresponding to normal-operation traffic. While this issue limits the use of the outliers detection technique (i.e. the attack is not an outlier in those cases), we show in the next section that real network traffic can be aggregated either at source or destination IP address to dramatically reduce the number of flows that compose a highly distributed and/or large volume attack. In order to avoid this limitation in the already processed flows of KDD99 (which can not be aggregated because we do not have the corresponding IP addresses), we shall select only a small fraction of flows for both DoS and probing attacks in the training and testing datasets. The training dataset has 950 normal flows and 255 attacks, while the testing dataset consists of 950 normal flows and 162 attacks.

Regarding the implementation of the rule-based NIDS, we build a different decision tree for each of the four different categories of attacks (DoS, probe, R2L, and U2R), using the training dataset and C4.5 decision trees. To train each of the trees for each specific category of attacks, we consider that the flows belonging to the rest of the categories of attacks as well as the normal operation flows correspond to the "other" class (there is no attack of the corresponding category in this case). For example, let us suppose that we want to build a decision tree to detect R2L attacks; in that case, all the flows in the trainig dataset which belong to the R2L category belong to the "R2L attack" class (there is a R2L attack), while the normal-operation flows as well as the DoS, probe, and U2R flows compose the other class.

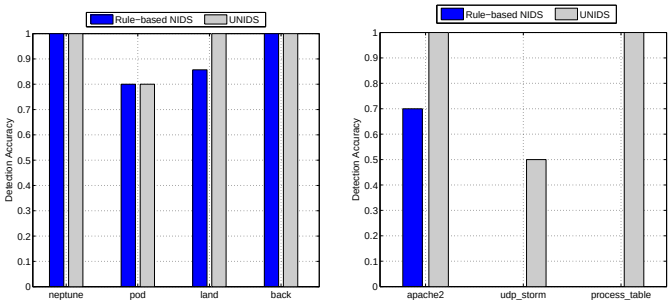
Figure 2 presents the detection accuracy (number of correctly detected attacks) obtained both with UNIDS and the decision-tree-based NIDS previously described in both the training and testing datasets. Results are individually presented for each of the four categories of attacks. Figure 2(a) shows that the results obtained by both systems in the training dataset are very similar: in the case of UNIDS, more than 90% of the attacks can be correctly detected in the four categories; in the case of the NIDS, it is quite obvious that using the system to detect the attacks which has been programmed to alert on shall provide good performance results. The detection accuracy of the NIDS in the training dataset is not 100% due to the post decision tree pruning step, which is performed to avoid over-fitting the model to the specific training dataset.

The interesting part of the comparison comes when we evaluate both systems in the testing dataset. Figure 2(b)



(a) Training dataset (b) Testing Dataset

Figure 3. UNIDS vs Rule-based NIDS - R2L attacks.



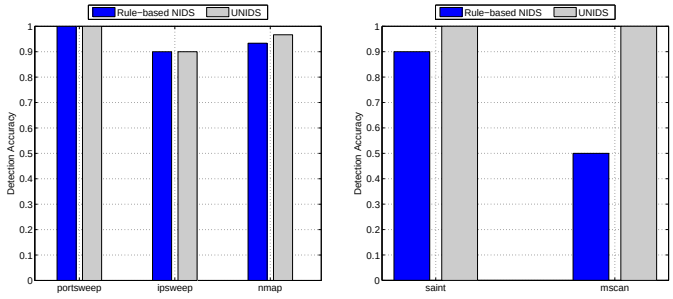
(a) Training dataset (b) Testing Dataset

Figure 4. UNIDS vs Rule-based NIDS - DoS attacks.

evidences the aforementioned limitations of rule-based NIDS to detect 0-day attacks, as well as the paramount advantage of using UNIDS in these cases. The NIDS is able to detect only 30% of all the DoS attacks in the testing dataset, and as few as 20% of the R2L attacks, whereas in both cases, UNIDS can detect more than 95% of these 0-day attacks. Figure 4 shows the same comparison, focusing exclusively in the DoS attacks category. Similar results are obtained for the remaining three categories, which are also individually evaluated in figures figures 3, 5, and 6. Finally, figure 7 shows the ROC curves obtained with UNIDS in detecting the aforementioned attacks when changing the detection threshold T_h, both in the training and testing datasets. Figure 7(a) shows the results obtained in the training dataset, while figure 7(b) shows the results obtained in testing dataset. In both cases we can appreciate that UNIDS is able to detect more than 90% of the attacks with very low false positive rates, less than 1% and 3.5% in both datasets respectively. As a general conclusion from these evaluations, we can see how useful can be UNIDS in pro-actively detecting unknown attacks, with remarkably low false alarm rates for being a fully unsupervised approach.

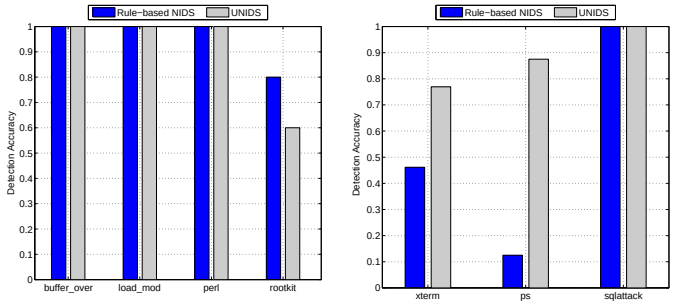
VI. UNIDS vs Other Unsupervised Approaches

We compare now the performance of UNIDS against three previous approaches for unsupervised anomaly detection proposed in the literature: DBSCAN-based, k-means-based, and PCA-based outliers detection. The first two consist in applying either DBSCAN or the k-means clustering algorithm [12] 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



(a) Training dataset (b) Testing Dataset

Figure 5. UNIDS vs Rule-based NIDS - Probing attacks.



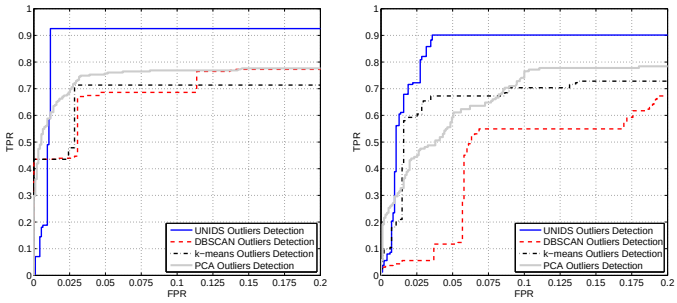
(a) Training dataset (b) Testing Dataset

Figure 6. UNIDS vs Rule-based NIDS - U2R attacks.

in previous work [13]–[15]. In the PCA-based approach, PCA and the sub-space method [8] 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 each evaluation scenario. Figure 7 shows the ROC curves obtained for UNIDS and the aforementioned unsupervised approaches in the detection of attacks in KDD99, including both the training and testing datasets. In both cases we can appreciate the out-performance of UNIDS w.r.t. these traditional approaches, which fail to detect as many attacks as UNIDS with a reasonable false alarms rate.

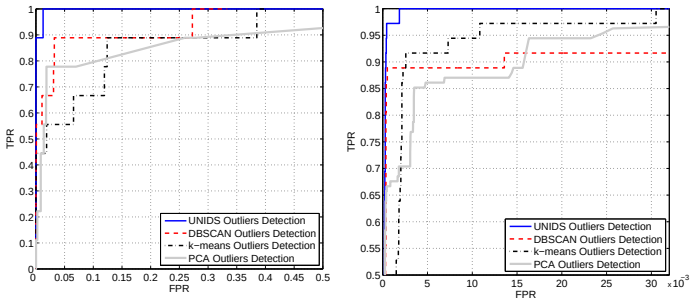
To conclude with this study, we finally evaluate the ability of UNIDS to detect different attacks in real traffic traces from two different networks: the WIDE network [18] and the French RENATER network. These results were already reported in [1], but we include them in here for the sake of completeness of the performance evaluation of UNIDS. 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. The dataset we use consists of raw packet traces collected at one of the trans-pacific links between Japan and the US. WIDE traces are not labeled, but some previous work on anomaly detection has been done on them [16], [17]. We use the combination of results obtained in both works as our *ground truth* for WIDE traffic. In the case of RENATER traffic, we consider traffic traces collected in the METROSEC project[4]. These

[4]"METROlogy for SECurity and QoS", at http://laas.fr/METROSEC



(a) Training dataset (b) Testing Dataset

Figure 7. True Positives Rate vs False Alarms in KDD99.



(a) METROSEC, IP dest. flows. (b) WIDE, IP source flows.

Figure 8. True Positives Rate vs False Alarms in WIDE and METROSEC.

traces consist of real traffic, complemented by the injection of synthetic flooding attacks performed with well-known DDoS attack tools. Traces 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). In both cases, the performance of UNIDS is compared against the previously described unsupervised approaches.

In these evaluations, we aggregate traffic either at source or destination IP address to show that the hypothesis of having a reduced number of flows as attacking flows is valid even in the case of distributed attacks. For example, if we define a flow as all the packets that are directed towards a single destination IP address, then this flow will be an outlier in the case of a DDoS, in which many different attackers send traffic to the same victim. As traffic features, we shall use in these evaluations basic traffic descriptors such as number of sources and destinations, number of source and destination ports, packet rate, fraction of SYN and ICMP packets, etc.

Figure 8 depicts the ROC curves obtained in the detection of different attacks in METROSEC and WIDE. Figure 8(a) corresponds to the detection of 9 DDoS attacks in the METROSEC dataset, using packets aggregated at destination IP address. 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%). Figure 8(b) corresponds to the detection of 36 anomalies in WIDE traffic, using packets aggregated at source IP address. 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. As before, obtained results permit to evidence the great advantage of using the proposed clustering and outliers ranking algorithms w.r.t. previously proposed approaches. In particular, these approaches 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 $\mathbf{X}$. 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.

VII. Conclusions

UNIDS has many interesting advantages w.r.t. standard NIDS. It uses exclusively unlabeled data to detect network attacks, without assuming any kind of signature, particular model, or canonical data distribution. This allows to detect new previously unseen network attacks, outperforming current misuse-based NIDS in copying with 0-day attacks. Despite using ordinary clustering techniques to identify network attacks, we have verified the effectiveness of UNIDS to detect both synthetic and real network attacks of very different nature (DoS, DDoS, scans, worms, buffer overflows, etc.) in multiple traffic traces, without assuming any particular traffic model, clustering parameters, or even clusters structure. UNIDS also outperforms traditional approaches for outliers detection, providing a stronger evidence of its accuracy to detect 0-day attacks.

References

[1] P. Casas et al., "UNADA: Unsupervised Network Anomaly Detection using Sub-Space Outliers Ranking", in *Proc. IFIP Networking*, 2011.

[2] S. Hansman, R. Hunt "A Taxonomy of Network and Computer Attacks", in *Computers and Security*, vol. 24 (1), pp. 31-43, 2005.

[3] G. Androulidakis, V. Chatzigiannakis, and S. Papavassiliou, "Network Anomaly Detection and Classification via Opportunistic Sampling", in *IEEE Network*, vol. 23 (1), 2009.

[4] R. Kemmerer and G. Vigna, "Intrusion Detection: A Brief History and Overview", in *IEEE Security & Privacy Mag.*, vol. 35 (4), 2002.

[5] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection", in *IEEE Communications Surveys & Tutorials*, vol. 12 (3), 2010.

[6] C. Kruegel and T. Toth, "Using Decision Trees to Improve Signature-Based Intrusion Detection", in *Proc. RAID*, 2003.

[7] J.-H. Lee et al., "Effective Value of Decision Tree with KDD99 Intrusion Detection Datasets for IDS", in *Proc. ICACT*, 2008.

[8] A. Lakhina, M. Crovella, C. Diot, "Diagnosing Network-Wide Traffic Anomalies", in *Proc. ACM SIGCOMM*, 2004.

[9] M. Ester et al., "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", in *Proc. ACM SIGKDD*, 1996.

[10] L. Parsons et al., "Subspace Clustering for High Dimensional Data: a Review", in *ACM SIGKDD Expl. Newsletter*, vol. 6 (1), pp. 90-105, 2004.

[11] 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.

[12] A. Jain, "Data Clustering: 50 Years Beyond K-Means", in *Pattern Recognition Letters*, vol. 31 (8), pp. 651-666, 2010.

[13] L. Portnoy, E. Eskin, S. Stolfo, "Intrusion Detection with Unlabeled Data Using Clustering", in *Proc. ACM DMSA Workshop*, 2001.

[14] 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.

[15] K. Leung, C. Leckie, "Unsupervised Anomaly Detection in Network Intrusion Detection Using Clustering", in *Proc. ACSC05*, 2005.

[16] G. Fernandes and P. Owezarski, "Automated Classification of Network Traffic Anomalies", in *Proc. SecureComm'09*, 2009.

[17] G. Dewaele et al., "Extracting Hidden Anomalies using Sketch and non Gaussian Multi-resolution Statistical Detection Procedures", in *Proc. LSAD*, 2007.

[18] K. Cho, K. Mitsuya, A. Kato, "Traffic Data Repository at the WIDE Project", in *USENIX Annual Technical Conference*, 2000.