Anomaly Extraction in Backbone Networks using Association Rules

Daniela Brauckhoff, Arno Wagner, Xenofontas Dimitropoulos
Department of Information Technology
ETH Zurich
Zurich, Switzerland
{brauckhoff,wagner,fontas}@tik.ee.ethz.ch

Kavè Salamatian
Computing Department
Lancaster University
Lancaster, England
kave@lancaster.ac.uk

ABSTRACT

Anomaly extraction is an important problem essential to several applications ranging from root cause analysis, to attack mitigation, and testing anomaly detectors. In this work, we study the problem of extracting anomalous flows from a large set of traffic flows suspected to contain an anomaly. We divide the anomaly extraction problem in two steps: in a first step meta-data from multiple anomaly detectors is used to filter suspect flows, the second step consists in mining sets of anomalous flows with the help of association rules. Using rich traffic data from a backbone network, we evaluate the decrease in classification cost achieved with rule mining and show that our techniques effectively isolate anomalous events and relevant flow data about the events.

Keywords

Anomaly detection, Anomaly extraction, Association rules

1. INTRODUCTION

Anomaly detection techniques are the last line of defense when other approaches fail to detect security threats or other problems. They have been extensively studied since they pose a number of interesting research problems, involving statistics, modeling, and efficient data structures. Nevertheless, they have not yet gained widespread adaptation, as a number of challenges, like reducing the number of false positives or simplifying training and calibration, remain to be solved.

In this work we are interested in the problem of identifying the traffic flows associated with an anomaly during a time interval with an alarm. We call finding these flows the anomalous flow extraction problem or simply *anomaly extraction*. At the high-level, knowing anomalous flows reflects the goal of gaining more information about an anomaly alarm, which without additional meta-data is often meaningless for the administrator. Identified anomalous flows can be used for a number of applications, like root-cause analysis of the event causing an anomaly, improving anomaly detection accuracy, and modeling anomalies.

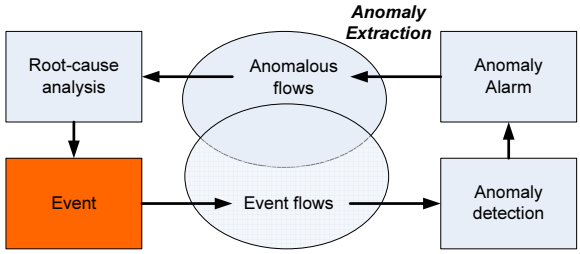

Figure 1: The high-level goal of anomaly extraction.

Our solution to the anomaly extraction problem is the following: We use several histogram-based detectors that provide fine-grained meta-data for filtering a set of suspect anomalous flows. Then we apply the concept of *association rules* in order to identify the event that triggered an anomaly alarm. We test our anomaly extraction techniques on rich network traffic data from a medium-size backbone network and identify several anomalies and associated anomalous flows. In particular, we show how the concept of association rules allows for reducing the classification cost by several orders of magnitude, and we evaluate the accuracy of the derived rulesets.

In Figure 1 we present the high-level goal of anomaly extraction. In the bottom of the figure, events with a network-level footprint, like attacks or failures, trigger *event flows*, which after analysis may raise an anomaly alarm. Ideally we would like to extract exactly all triggered event flows; however knowing or quantifying if this goal is realized is practically very hard due to inherent limitations in finding the precise ground truth of event flows in real-world traffic traces. The goal of anomaly extraction is to find a set of *anomalous flows* coinciding with the event flows. Suppose we have a large number of flows f observed during a time interval t. [1]

[1]We use the classical flow definition, where each flow has common values in the 5-tuple: source/destination IP ad-

MetaData	AD methods in literature
Interval (only)	Kalman [13], Wavelets [2], Spatial-PCA [10], Spatio-Temporal-PCA [3], Tsallis-Entropy [14]
Protocol	Maximum-Entropy [7]
IP range	Defeat [11], MR-Gaussian [5]
Port range	Maximum-Entropy [7]
TCP flags	Maximum-Entropy [7]

Table 1: Meta-data provided by existing anomaly detection systems.

The anomaly extraction process takes as input a set of n flows $F = \{f_1, \ldots, f_n\}$ observed during interval t and creates a subset F_A of anomalous flows.

An anomaly detection system may provide meta-data relevant to an alarm that help to narrow down the set of candidate anomalous flows. For example, anomaly detection systems analyzing histograms indicate the histogram bins an anomaly affected, e.g., a range of IP addresses or port numbers. Such meta-data can be used to restrict the candidate anomalous flows to these that have IP addresses or port numbers within the affected range. We refer to the step of using meta-data to restrict the candidate anomalous flows as *filtering*. In Table 1 we outline meta-data provided by various well-known anomaly detectors that can be useful for filtering.

To extract anomalous flows, one could build a model describing normal flow characteristics and use the model to identify deviating flows. However, building such a microscopic model is very challenging due to the wide variability of flow characteristics. Similarly, one could compare flows during an interval with flows from normal or past intervals and search for changes, like new flows that were not previously observed or flows with significant increase/decrease in their volume. Such approaches essentially perform anomaly detection at the level of individual flows and could be used to identify anomalous flows.

In this work, we take an alternative approach to identify anomalous flows that combines and consolidates information from multiple anomaly detectors. Compared to the other possible approaches, our method does not rely on past data for normal intervals or normal models, which substantially simplifies the problem. Intuitively, each detector provides an additional view into network traffic. Additional views or equivalently detectors explore different techniques for finding anomalies, different traffic features, or simply different calibration settings. Each detector may raise an alarm for an interval t and provide a set of candidate anomalous flows,

dress, source/destination port number, and protocol number.

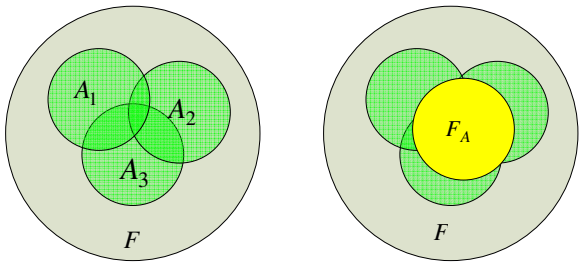

Figure 2: Anomaly extraction: (1) filtering flows identified by each detector j (left), (2) extracting the set of anomalous flows F_A from $\cup A_j$ (right).

where candidate flows are the complete set of flows observed during the interval or the set of flows that results after filtering. This is illustrated in Figure 2, where a set A_j represents the candidate flows supplied by detector j and the union $\cup A_j$ represents the candidate anomalous flows. We then use association rules to mine flows with common characteristics in the dataset A_j. The assumption we make here is that flows with common characteristics represent anomalous flows. We validate this assumption on several weeks of Netflow data captured in a backbone network that contains a large variety of anomalies.

The rest of the paper is structured as follows. How we extract anomalous traffic from Netflow traces with the help of histogram-based detectors and association rules is described in section 3. In section 4 we first describe the datasets used for this study, and then present the evaluation results. Related work is discussed in section 5. Finally, section 6 concludes the paper.

2. ANOMALY EXTRACTION

In the following section we outline our approach for generating fine-grained filters with histogram-based detectors, and for finding the set of anomalous flows with the help of association rules.

2.1 Fine-grained Metadata for Flow Filtering

Histogram-based anomaly detectors [7, 11, 5, 8] have been shown to work well for detecting anomalous behavior and changes in traffic distributions. We propose an alternative histogram-based detector that differs from its predecessors in several ways: (i) We do not train a normal model but use the previous interval as reference for comparison, thus circumventing the still unsolved problem of recalibrating a derived model. (ii) We provide a threshold-based method for identifying the histogram bins that have triggered an alert.

As detection metric we use the Kullback-Leibler (KL) that has been successfully applied for anomaly detection in [7, 8]. The KL distance measures the similarity of a given discrete distribution q to a reference distribution

p and is defined as

$$D(p||q) = \sum_{i=0}^{n} p(i) log_2(\frac{p(i)}{q(i)}). \quad (1)$$

Coinciding distributions have a KL distance of zero, while deviations in the distribution cause larger KL distance values. In general, the KL distance is an asymmetric measure, $D(p||q) \neq D(q||p)$.

It is worthwhile discussing the difference between a trained reference model and a constantly adapting reference model as in our case. Suppose the measured distribution changes at time t for n intervals due to an event and returns back to normal afterwards. The KL distance obtained from comparison with a trained reference model will show a rectangular pulse of length n, whereas comparison with an adapting reference model will result in two KL distance pulses of length 1 at time t (additive change) and $t+n$ (subtractive change). For the purpose of anomaly extraction we are particularly interested in intervals with additive changes in the traffic distribution. We have observed that the backward KL distance (taking the *current* interval as reference for computing the distribution p) reacts stronger to additive changes than the forward KL distance (taking the *previous* interval as reference for deriving p). We relate this behavior to the weight $p(i)$ in Equation 1. As a consequence of this observation, we use the backward KL distance as detection signal.

The five feature distributions we track over intervals of 5-minute duration are source/destination IP address, source/destination port number, and flow size (in number of packets). Additionally, we separate the traffic by flow direction (flows originating vs. terminating in the observed AS) and protocol (TCP vs. UDP). To obtain additional views of the traffic we maintain multiple randomized versions of each feature distribution that are generated by applying multiple independent hash functions. [11, 5] follow a similar approach. [2].

For each flow that is received we call an *update* function that processes the five flow features. In particular, for each feature and hash function it computes the hash values and increases the count in each identified histogram bin. The *detect* function is called when an interval is completed (*i*.e., , when a flow with start time larger than the interval end is received). The function is applied to each feature and hash function separately. It computes the backward KL distance and checks if it has exceeded a preset threshold value. If not, the function returns. If we have detected a deviation in the distribution, our goal is to extract the meta-data (*e*.g., the IP addresses) that have caused the change. In order to extract this meta-data we first identify the responsible hash bins and then use reverse hashing obtain the actual feature values. To find the responsible hash bins we use an iterative algorithm that simulates the removal of suspect flows until the KL distance falls below the threshold. In each round we select the bin with the largest absolute distance in the previous and current distribution. Removal of flows falling into this bin is simulated by setting the bin count in the current distribution equal to its value in the previous distribution. Having identified the set of hash bins we obtain the feature values behind via reverse hashing. For each feature we use the intersection of values identified by reverse hashing as as meta-data. The flow set A_j for feature/detector j consequently contains all flows that match at least one of the feature values.

[2]Following their findings we use five independent 10-bit hash functions.

2.2 Association Rules for Anomaly Extraction

Association rules describe items that occur frequently together in a given dataset. A typical and widely-used example of association rules is market basket analysis. A record in such basket data sets typically consists of the transaction date and the items that have been bought. An example of a rule in this scenario might be that 98% of customers that purchase tires also get automotive services done [1]. Such rules are interesting for companies to adjust store layouts, promotions, or catalog design. Formally, we can define association rules as follows: Suppose that each transaction T is a set of items $I = \{i_1, \ldots, i_m\}$, and $X \subset I$, $Y \subset I$ are disjoint item sets. An association rule $X \Longrightarrow Y$ is said to have support s in the dataset if s transactions contain $X \cup Y$.

The basic motivation for applying association rules to the anomaly extraction problem is the following: Anomalous flows typically have similar multivariate characteristics (*e*.g., source/destination IP addresses, port numbers, or flow length) since they have a common root-cause be it a network failure, a bot engine, or a Denial of Service script. We use association rules to identify these characteristics in an automated manner. The generated rules can then be directly applied for extracting the set of anomalous flows F_A.

Ideally, the rule mining step would only return rules that match anomalous flows. In reality, however, the ratio of rules capturing anomalous flows and those capturing non-anomalous flows depends largely on the quality of the available detectors that are used in the filtering step. If we use for example a simple interval-based detector (*i*.e., the meta-data provided is only the interval), we have to apply the rule mining to all flows in a given interval. Naturally, this dataset contains many non-anomalous flows that cause additional rules to be generated. On the other hand, if we use histogram-based detectors (*i*.e., we obtain fine-grained meta-data that identifies suspicious IP addresses or port number) for filtering the dataset, fewer rules will capture non-anomalous flows. At this point, a human is still required

in the loop to classify the generated association rules. We argue, however, that association rules are a great help for this task since they reduce the information to be processed from hundred thousands of flows to possibly several tenths of rules. Moreover, rules that are frequently observed but non-anomalous (*e.*g., web servers with exceptionally high load) might even be white-listed to reduce the number of rules for manual processing. A learning-based rule classification algorithm is a direction for future work.

As said previously, we take the union set of flows filtered by all available detectors $\cup A_j$. Each flow in our dataset has several features associated with it. An item in our case is a combination of the feature f and its value v, *e.*g., the item $i_1 = (f_1, v_1) = (spo, 80)$ refers to a source port number equal 80 while $i_2 = (dpo, 80)$ refers to a destination port number equal 80. A rule

$$r : X \Longrightarrow Y \tag{2}$$

expresses an association between two item sets $X = \{(f_1, v_1)\}$ and $Y = \{(f_2, v_2, (f_n, v_n))\}$. The *support* of an association rule $sup(r)$ is defined as the number of flows that match a given rule r.

2.3 Rule Mining in Detail

The problem of discovering all association rules in a dataset can be decomposed into two subproblems [1]:

- *Discover large item sets*: Find all item sets that have a support above the user-specified minimum support, *i.*e., the number of flows containing this item set is larger than the minimum support.
- *Discover association rules*: Use the large item sets found in the last step to generate the desired rules. Basically, for every non-empty subset a of every large item set l, we generate a rule $a \Longrightarrow (l - a)$ if the ratio of $sup(l)$ to $sup(a)$ is larger than a user-specified minimum confidence.

For discovering the large item sets in our data we implemented the Apriori algorithm proposed by Agrawal and Srikant [1]. The Apriori algorithm makes several passes over the data, and counts in each pass the support for candidate item sets. After each round, the large item sets are selected and used as candidate item sets for the next round. The process continues until no new large item sets are found. This algorithm is computationally more efficient than previous ones since it reduces the number of candidate item sets to be considered in each round. We omit a detailed discussion here and refer the interested reader to the original paper.

Inputs for the rule mining algorithm are the dataset ($\cup A_j$) and two parameters: the minimum support ($minsup$) and the minimum confidence ($minconf$). These two parameters determine the sensitivity of the rule mining method. Smaller $minsup$ and $minconf$ values will result in fewer rules to be generated, while larger values will result in a larger ruleset. Obviously, if the $minsup$ parameter is set to a value that is larger than the size of the common anomalous flow set $|F_A|$ it will not be detected by the algorithm. An estimation for the maximum $minsup$ parameter that is required to detect an anomalous flow set can be obtained from the detectors, *i.*e., based on the cumulative change in flows over all flagged hash bins.

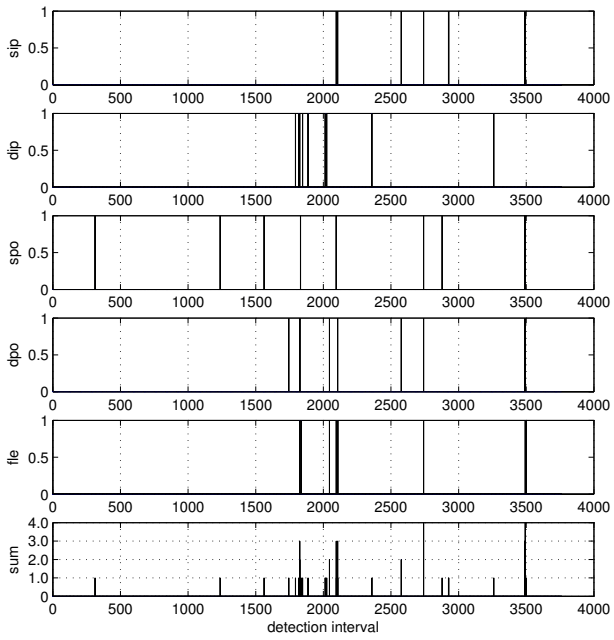

Figure 3: For each histogram-based detector (source IP address, destination IP address, source port, destination port, flow length) we plot the 10 largest anomalies it has detected. In the last row the sum of alerts per interval is shown.

3. EVALUATION

In this section we present the datasets that are used in this study. Further we evaluate the reduction in classification cost that is achieved with association rules, and the accuracy of the generated rule sets.

3.1 Data Set

For validation we use three weeks of non-sampled and non-anonymized Netflow data coming from one of the peering links of a medium-sized ISP (SWITCH, AS559). SWITCH is the backbone operator connecting all Swiss universities and various research labs (e.g.,CERN, IBM) to the Internet. The SWITCH IP address range contains about 2.2 million IP addresses. In the trace we see on average 92 million flows and 220 million packets per hour. The dataset used for this study was recorded in August 2007 and comprises a variety of traffic anoma-

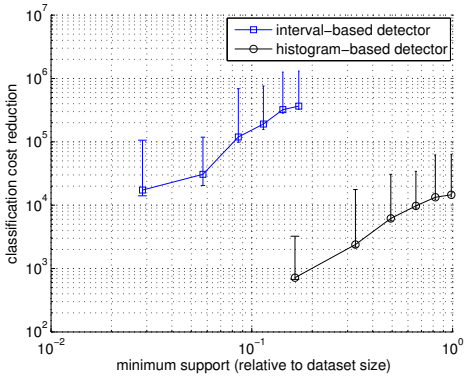

Figure 4: Reduction in classification cost vs. minimum support (relative to the dataset size $|\cup A_j|$) when compared with interval-based and histogram-based detectors (mean and min/max errorbars).

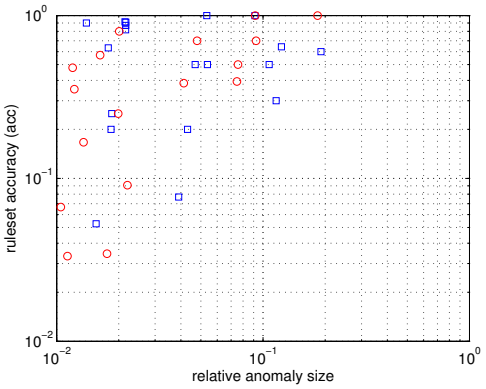

Figure 5: Ruleset accuracy for the set of selected anomalies when the input dataset is only restricted by the interval and transport protocol (worst-case). TCP anomalies are marked with squares, UDP anomalies with circles.

lies happening in daily operation such as network scans, Denial of Service attacks, alpha flows, etc. We focus in this study on flows that originate within AS559 and that do not terminate within AS559.

For our evaluation we manually verified selected intervals that have been flagged by one of the detectors as anomalous. Specifically, we selected the top10 intervals with the largest KL distance values for each detector. The distribution of these top10 intervals over time and over the different detectors is shown for TCP traffic in Fig. 3. Since there is some overlap between the top10 sets of the detectors, the final set contains 31 intervals for TCP traffic and 39 intervals for UDP traffic that have been verified manually. In each of the selected intervals at least one anomalous event was identified. Among them we have several scans for different vulnerabilities (*e.g.*, port 135,34689,6000) as well as the corresponding replies, distributed and single-source Denial of Service attacks, and spam campaigns.

3.2 Decrease in Classification Cost

Using association rules we obtain a higher level view that is based on rules instead of flows. As a consequence, the flow classification problem can be reduced to a rule classification problem. If we find the rule that matches the anomalous flows, the flow extraction problem is considered as solved.

To quantify this decrease in classification cost, we make the assumption that the classification cost is a linear function of the number of items that need to be classified. We define the reduction in classification cost $red(R)$ for a given ruleset R generated by rule mining as follows:

$$red(R) = \frac{|\cup A_j|}{|R|} \tag{3}$$

where $|\cup A_j|$ denotes the number of flows in the filtered flow set, and $|R|$ the size of the ruleset.

We consider two different cases: the histogram-based detectors described in section 2 that provide fine-grained meta-data for filtering, and an interval-based detector (considering the KL distance only) that provides only the interval and the transport protocol as input. The size of of the rule set R depends (1) on the *minsup* parameter, which gives the minimum support that a rule/anomaly needs for being considered by the algorithm, and (2) the number of features that are considered. For this study, we consider a constant set of eight features, including the source/destination IP address, source/destination port number, number of packets and bytes per flow, and flow length in msec. We generated the filtered datasets for interval-based and histogram-based detectors and counted the number of flows in each dataset. Then we applied association rule mining to these datasets varying the *minsup* parameter between 10'000 and 60'000.

In Fig. 4 we plot the average cost reduction vs. the minimum support relative to the dataset size, which is obviously much larger for interval-based detectors. For the interval-based detector, the average reduction in classification cost ranges between 10^4 and 10^5. For histogram-based detectors, the average reduction ranges between 10^3 and 10^4 for the selected range of *minsup* parameters. This proves our assumption that association rule mining can greatly simplify the anomaly extraction problem.

3.3 Ruleset Accuracy

Still, the reduced rule set is a mix of rules matching anomalous flows (true positives) and non-anomalous flows (false positives). Next, we evaluate the accuracy (or true positive rate) of the obtained rule sets. For each interval we have a set of ground truth rules R_{truth}

that have been verified to match anomalous events by a human expert. The output of the rule mining algorithm is a *ruleset* R. We define the accuracy of a rule set $acc(R)$ as

$$acc(R) = \frac{|\{r|r \in R_{truth}\}|}{|R|}. \quad (4)$$

The false positive rate becomes thus $1 - acc(R)$.

In Fig. 5 we plot the ruleset accuracy for several anomalies against the relative size of each anomaly (number of anomalous flows divided by total flows in the dataset). The *minsup* parameter has been estimated in each case based on information provided by the detectors, *i*.e., the number of flows that had to be removed for getting rid of the alarm. As input we used all flows in the given interval that match the transport protocol. Thus the results presented in Fig. 5 are a lower-bound estimation of the ruleset accuracy. By restricting the input data set to flows that match the meta-data provided by the histogram-based detectors we can further improve the ruleset accuracy.

As one would expect, the ruleset accuracy increases with the relative anomaly size. Nevertheless, even for anomalies that contribute only 1% to the total flows result in acceptable accuracy rates. This nicely validates our assumption that a large variety of anomalous events have flows with similar multivariate characteristics and can thus be captured by association rules. Another advantage of our approach is that multiple anomalies are correctly split by the algorithm if they differed in at least one feature.

4. RELATED WORK

Many different approaches have been proposed for anomaly detection in backbone networks, *e*.g., [2, 13, 10, 7]. Most of these approaches, however, focus on the detection part and then use ad-hoc methods for extracting the anomalous flows.

Histogram-based detectors similar to ours have been proposed by Dewaele *et al.* [5], Li *et al.* [11], and Gu *et al.* [7]. Nevertheless, our detector differs from previous approaches in several ways. Dewaele *et al.* [5] first use sketches to reduce the dimensionality of data, and then derive non Gaussian models for the normal behavior at different time scales. In contrast, we use the distribution from the previous interval as reference model and provide a threshold-based method for identifying anomalous distribution bins. Li *et al.* [11] also use sketches and apply the PCA-subspace method [10] for anomaly detection. We rely on the KL distance for anomaly detection and concentrate on identifying anomalous flows. Gu *et al.* [7] use the Maximum Entropy technique to estimate the distribution of baseline data. Anomalies in observed traffic are detected by computing the relative entropy between the trained baseline density model and the actual traffic distribution. In contrast to their work, we defer the model building to a later point, after the rule mining, when more precise information is available.

Association rules have been successfully applied to different problems in networking. Chandola and Kumar [4] study the problem of finding an optimal summarization for a set of anomalous flows. They propose and evaluate techniques based on clustering and rule mining. In contrast, we provide a solution for separating normal and anomalous flows. Mahoney and Chan [12] use association rule mining to find rare events that are suspected to represent anomalies in packet payload data. They evaluate their method on the 1999 DARPA/Lincoln Laboratory traces. Their approach is targeted at edge networks where mining rare events makes sense. In massive backbone data, however, this approach is less promising. Another application of association rule mining in edge networks is eXpose [9]. The system learns fine-grained communication rules by exploiting the temporal correlation between flows within 1s time windows.

Duffield *et al.* [6]. derive flow-level signatures based on alerts of packet-level sensors (SNORT) that are deployed in edge networks via machine learning techniques. Their approach is orthogonal to ours since it relies on signature-based detectors.

5. CONCLUSION

In this paper, we have posed and formalized the problem of anomaly extraction that is of uttermost importance to several applications such as root-cause analysis and detection system testing. We have presented a histogram-based detector that provides fine-grained meta-data for filtering suspect flows. Further, we have introduced a method for extracting anomalous flows that uses association rules to describe flows that have similar characteristics across several features.

We have used a rich Netflow dataset captured in a backbone network to validate the proposed techniques. Evaluation results show that the classification cost can be reduced by several orders of magnitude using association rules. Further, we evaluated the ruleset accuracy for several verified anomalies. We find that an acceptable accuracy rate of 10% can be achieved for most anomalies found in our dataset. In future work, we will concentrate on automating the rule classification and building anomaly models for testing purposes.

6. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September*

12-15, 1994, Santiago de Chile, Chile, pages 487–499. Morgan Kaufmann, 1994.

[2] P. Barford, J. Kline, D. Plonka, and A. Ron. A signal analysis of network traffic anomalies. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 71–82, New York, NY, USA, 2002. ACM Press.

[3] D. Brauckhoff, M. May, and K. Salamatian. Applying PCA for Traffic Anomaly Detection: Problems and Solutions. In *IEEE Infocom Miniconference*, 2009.

[4] V. Chandola and V. Kumar. Summarization - compressing data into an informative representation. *Knowl. Inf. Syst.*, 12(3):355–378, 2007.

[5] G. Dewaele, K. Fukuda, P. Borgnat, P. Abry, and K. Cho. Extracting hidden anomalies using sketch and non gaussian multiresolution statistical detection procedures. In *LSAD '07: Proceedings of the 2007 workshop on Large scale attack defense*, pages 145–152, New York, NY, USA, 2007. ACM.

[6] N. Duffield, P. Haffner, B. Krishnamurthy, and H. Ringberg. Rule-based anomaly detection on ip flows. In *Infocom 2009*, 2009.

[7] Y. Gu, A. McCallum, and D. Towsley. Detecting anomalies in network traffic using maximum entropy estimation. In *IMC '05: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 32–32, Berkeley, CA, USA, 2005. USENIX Association.

[8] K. R. Houerbi, K. Salamatian, and F. Kamoun. Scan surveillance in internet networks. In *Networking 2009*, 2009.

[9] S. Kandula, R. Chandra, and D. Katabi. What's going on?: learning communication rules in edge networks. In *SIGCOMM*, pages 87–98, 2008.

[10] A. Lakhina, M. Crovella, and C. Diot. Mining Anomalies Using Traffic Feature Distributions. In *ACM SIGCOMM 2005*, 2005.

[11] X. Li, F. Bian, M. Crovella, C. Diot, R. Govindan, G. Iannaccone, and A. Lakhina. Detection and identification of network anomalies using sketch subspaces. In *IMC 2006*, New York, NY, USA, 2006.

[12] M. V. Mahoney and P. K. Chan. Learning rules for anomaly detection of hostile network traffic. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 601, Washington, DC, USA, 2003. IEEE Computer Society.

[13] A. Soule, K. Salamatian, and N. Taft. Combining filtering and statistical methods for anomaly detection. In *IMC '05*, 2005.

[14] B. Tellenbach, D. Brauckhoff, and M. May. Impact of traffic mix and packet sampling on anomaly visibility. In *ICIMP 2008*, Bucharest, Romania, July 2008.