Analysis of Three Intrusion Detection System Benchmark Datasets Using Machine Learning Algorithms

H. Güneş Kayacık [1], Nur Zincir-Heywood [1]

[1] Dalhousie University, Faculty of Computer Science,
6050 University Avenue, Halifax, Nova Scotia. B3H 1W5
{kayacik,zincir}@cs.dal.ca

Abstract. In this paper, we employed two machine learning algorithms – namely, a clustering and a neural network algorithm – to analyze the network traffic recorded from three sources. Of the three sources, two of the traffic sources were synthetic, which means the traffic was generated in a controlled environment for intrusion detection benchmarking. The main objective of the analysis is to determine the differences between synthetic and real-world traffic, however the analysis methodology detailed in this paper can be employed for general network analysis purposes. Moreover the framework, which we employed to generate one of the two synthetic traffic sources, is briefly discussed.

1 Introduction

Along with benefits, the Internet also created numerous ways to compromise the stability and security of the systems connected to it. Although static defense mechanisms such as firewalls and software updates can provide a reasonable level of security, more dynamic mechanisms should also be utilized. Examples of such dynamic mechanisms are intrusion detection systems and network analyzers. The main difference between intrusion detection and network analysis is the former aims to achieve the specific goal of detecting attacks whereas the latter aims to determine the changing trends in computer networks and connected systems. Therefore network analysis is a generic tool, which helps system administrators to discover what is happening on their networks.

In this paper, we employed two machine learning algorithms for network analysis. The first method is a clustering algorithm, which aims to find the natural groupings in the dataset. The second method is based on a neural network algorithm called Self-Organizing Maps (SOMs) where topological models are built for the given dataset. Both methods act like network analyzers to discover similarities between the datasets. The objective of the analysis is to determine the robustness of our synthetic dataset in terms of its similarity to real-world datasets. However, the analysis methods could be employed for wide variety of purposes such as anomaly detection, network trend analysis and fault detection.

2 Framework for Generating Synthetic Traffic

Because of the privacy concerns and the extensive storage requirements, employing the data captured from a live network is not practical. Moreover, the captured data itself reveals very little information without further analysis. Having recognized the need for synthesizing data for training and testing intrusion detection systems, we developed a framework [1], which can: (1) Develop models of normal behavior from its observations. (2) Generate synthetic activity based on the developed analytic models. Our synthetic traffic generation framework focuses on modeling hypertext transfer protocol (HTTP). HTTP is selected because considerable amount of the Internet traffic is made up of HTTP traffic [2]. Our framework is implemented for HTTP however it could easily be tailored for many other protocols, which involve file transfers with variable user think times.

The framework [1] is divided into four tasks. (1) HTTP is not a session-based protocol; that is to say web server logs only contain the records of requested documents without any information about where a session ends. Session concept is imperative because it acts as a placeholder to group the activities of a user. The first task is to construct sessions from web server logs. (2) The second task is to develop session models by employing first order discrete Markov models [3]. Each web page is considered as a state and Markov models are developed by maintaining the frequencies of web page transitions. Models are developed for page transitions as well as transition delays (user think times). (3) Developed models are employed to generate synthetic sessions. A synthetic session starts from a special purpose start state and transitions (i.e. web page requests) are appended to the session stochastically until the special purpose end state is reached. This way, the model can simulate sessions with arbitrary lengths. (4) Synthetic sessions are processed and web page requests are passed to the web browser installed on the machine to generate the network traffic.

3 Analysis and Results

In order to determine the robustness of the generated synthetic dataset, we employed a SOM based intrusion detection system [4] and a clustering algorithm [5] as network traffic analyzers to compare the characteristics of synthetic and real-world datasets. The objective of the analysis is to determine whether our synthetic dataset shows improvements over the benchmark dataset, however similar analysis can be performed to discover the changing traffic trends in a network.

Data Description

In case of all three data sources, network traffic is recorded in *tcpdump* format. We compared our synthetic data with the standard intrusion detection system benchmark dataset, namely KDD 99 dataset, which is based on Lincoln Lab. DARPA 98 dataset.

DARPA datasets, to the best of authors' knowledge, are the most comprehensive undertaking in intrusion detection system benchmarking. Similar to the preprocessing stage of KDD 99 competition, in order to summarize packet level data into high-level events such as connections, we employed Bro network analyzer. For KDD 99 competition, Bro was customized to derive 41 features per connection. This customized version is not publicly available furthermore the publicly available version can only derive 6 features per connection. Therefore, in our analysis, we employed 6 basic features from KDD datasets and employed publicly available Bro to summarize *tcpdump* data into connection records with these 6 features. The features are duration of the connection; protocol; service; connection status; total bytes sent to destination host; total bytes sent to source host. Among three datasets provided in KDD 99 competition, 10% KDD dataset, which was the original training set, was employed. Since our synthetic dataset only contains HTTP connections, other datasets were filtered to contain only HTTP connections. Outliers in each dataset were determined and removed by box plot with fences method [6].

In addition to the synthetic traffic that we generated from Ege University Vocational School web server logs and synthetic dataset from KDD 99 competition, we employed a real-world traffic, which is the captured traffic from Dalhousie University Faculty of Computer Science server Locutus, which has hundreds of users. The traffic was recorded in one day on December 2003. The approximate size of the recorded traffic is 2 gigabytes.

Analysis with Clustering

The objective of the k-means algorithm [5] is to partition the dataset into k groups or clusters where the number of desired clusters k is fixed *a priori*. Each cluster is assigned a cluster center (centroid), which defines the geometric center of the cluster. K-means clustering utilizes an iterative algorithm to minimize the distances between the dataset instances and the centroids. Training is carried out until the position of the centroids stop changing. Resulting clusters depend on the initial placement of the centroids.

By utilizing k-means clustering, centroids are calculated for each dataset. Since the initial placement and the number of centroids influence the resulting clusters, k-means clustering was employed with 6, 18 and 36 centroids. Different number of centroids produced similar results therefore results on 36 centroids are shown in Table 1. Each value in Table 1 is a coefficient of similarity and expressed as *C (D1, D2),* where *D1* is the dataset specified in the column header and *D2* is the dataset specified in the row header. The coefficient determines the similarity between the training dataset *D1* and test dataset *D2*. Coefficient of similarity is calculated by utilizing the centroids of the training dataset. For each instance in the test dataset, Euclidean distance to the closest training centroid is calculated. Calculated distances are averaged over all test dataset instances to form the coefficient of similarity. Given there are 6 normalized features, the coefficient ranges between 0 and 2.45, where 0 indicates high similarity.

Table 1. Coefficient of similarity matrix

	KDD	Ege	Locutus
KDD	0.001	0.355	0.341
Ege	0.826	0.007	0.257
Locutus	0.789	0.127	0.227

The coefficient *C (D, D)* shows the degree of dispersion in dataset *D*. *C (KDD, KDD)* and *C (Ege, Ege)* indicate that the synthetic dataset instances are close to the centroids, which results in low dispersion, whereas *C (Locutus, Locutus)* indicates higher dispersion in real-world datasets. Additionally, results detailed in the KDD column, i.e. *C (KDD, Ege)* and *C (KDD, Locutus)*, demonstrate that KDD is dissimilar to other two datasets. Compared with KDD dataset, i.e. *C (Locutus, KDD)*, our synthetic data shows more similarities to real-world data, *C (Locutus, Ege)*.

Analysis with SOM Based Intrusion Detection System

In Kayacik et al. [4], a two level SOM hierarchy was developed for intrusion detection. The system utilizes the 6 basic features of a connection, which can be derived from packet headers without inspecting the packet payload. SOM [5] is a neural network algorithm, which employs unsupervised learning for training. At the first level, six SOMs are trained, one for each feature where the general objective is to encode temporal relationships within the features. The second level combines the information from the first level SOMs. Other than the representation of temporal features and labeling of second level neurons, no further *a priori* information is employed.

Since real-world datasets are unlabeled, hit histograms are employed to analyze datasets. Hit histograms summarize how often the neurons are excited (i.e. selected as the best matching neuron). As a neuron is excited for more patterns in a dataset, the area of the hexagon being colored becomes larger. If the training and the test datasets have similar statistical properties such as similar range, probability distribution and dispersion, then the test dataset would populate a considerable portion of the SOM.

In Figure 1, the hierarchy trained on KDD demonstrates that, out of 36 neurons, mainly one or two are excited for any given dataset. This suggests that the organization of the SOM trained on KDD is unable to distinguish the characteristics of any dataset other than its training set (to a minimal degree). The system trained on our synthetic dataset (Figure 2) showed improvements compared to the system trained on KDD. In Ege SOM hierarchy, patterns in real-world dataset Locutus start to excite multiple neurons, specifically the lower right region. This indicates that the organization of the SOM hierarchy can comparatively distinguish different patterns in a real-world dataset.

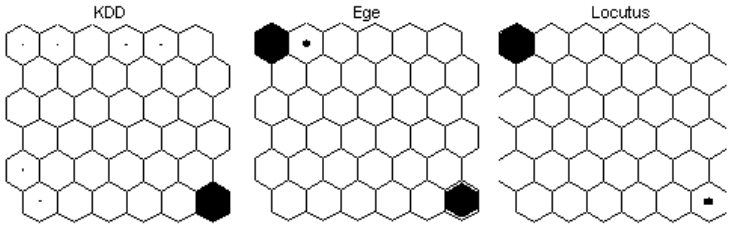

Figure 1. Hit histograms from SOM hierarchy, which is trained on KDD dataset

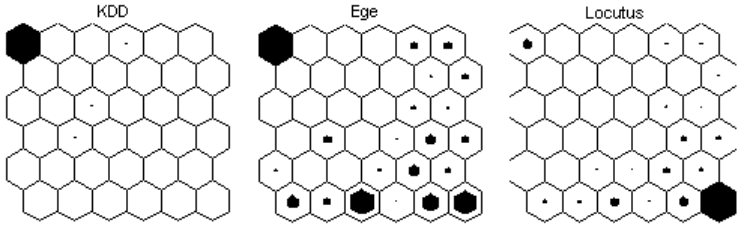

Figure 2. Hit histograms from SOM hierarchy, which is trained on our synthetic dataset Ege

4 Conclusions and Future Work

In this paper, a framework for generating synthetic network traffic is discussed along with two techniques to analyze network traffic. The framework generates synthetic network traffic based on the models developed from web server logs. The set of components that we developed within the framework comprise a comprehensive toolkit, which can develop usage models and generate traffic. Previous benchmarking approaches (particularly, KDD dataset) were criticized because: (1) Simulated normal usage is said to be similar to what one would observe in a real-world environment. However this claim was not sufficiently validated. This raises many questions about how the normal usage is modeled [7]. (2) Internet traffic is not well behaved. Moreover, real-world data is more diverse than the synthetic traffic. Mahoney et al. [8] established that the KDD dataset is clean and known to have idiosyncrasies that can lead to detection.

In order to develop anomaly based detectors that can withstand in real-world environments; it is crucial to synthesize training data that has the similar characteristics to real-world data. Although our framework currently supports one protocol, by providing analytic modeling and facilitating repeatable simulations, it fills a gap in intrusion detection system benchmarking by generating synthetic traffic similar to real-world traffic.

In addition to generating synthetic network traffic, we employed two machine learning techniques to analyze the datasets. In k-means clustering, the objective is to find the centers of mass in the dataset. If two datasets are similar, their centers of mass

will be close therefore the coefficient of similarity will be small. SOM intrusion detection system aims to achieve topological arrangement with respect to the training set, if the training and the test datasets have similar characteristics, the test dataset would excite a substantial portion of the SOM. Analysis results showed that the synthetic dataset generated by the framework shows improvements over KDD dataset in terms of being more similar to the real-world dataset.

The aforementioned techniques can also be employed for forensic analysis on network traffic. For instance given the traffic recorded at different periods from the same source, analysis can reveal the changing trends in the network traffic. Visualization methods of SOM can be employed to see the topological relationships of the datasets. Moreover analysis of the outliers can provide information about the anomalous activities. The future work will investigate the use of other datasets from commercial and governmental organizations to develop models of usage and will employ other machine learning techniques to analyze the datasets.

Acknowledgements

This work was supported in part by NSERC Discovery and CFI New Opportunity Grants from the Canadian Government. Further information regarding the work described in this publication is available through the project homepage of the NIMS research group http://www.cs.dal.ca/projectx/.

References

1. Kayacik, G. H., Zincir-Heywood, A. N., "Generating Representative Traffic for Intrusion Detection System Benchmarking", Proceedings of the IEEE CNSR 2005 Halifax, Canada, May 2005.
2. Odlyzko A., “Internet traffic growth: Sources and implications”, 2003, http://www.dtc.umn.edu/~odlyzko/doc/itcom.internet.growth.pdf Last accessed Nov. 2004.
3. Norris J.R., Markov Chains, Cambridge University Press, 1997, ISBN 0-521-48181-3
4. Kayacik, G. H., Zincir-Heywood, A. N., Heywood, M. I., "On the capability of SOM based intrusion detection systems," Proceedings of the 2003 IEEE IJCNN, Portland, USA, July 2003.
5. MacQueen, J.B., "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297, 1967.
6. Chambers J., Cleveland W., Kleiner B., and Tukey P., (1983), “Graphical Methods for Data Analysis”, Wadsworth.
7. McHugh J., “Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory”, ACM Transactions on Information and System Security, Vol. 3 No.4 November 2000.
8. Mahoney M.V., Chan P.K. “An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection”, In RAID 2003 Symposium, Pittsburgh, USA, September 8-10, 2003, Springer 2003, ISBN 3-540-40878-9