Poster: Automated Anomaly Detection Within The Toa Network Flow Data Monitoring System

José Alfredo Valles Salas
Department of Computer Science
University of Puerto Rico,
Río Piedras Campus
San Juan, PR 00936-8377
Email: alfredo.valles390@gmail.com

Ian Dávila Morales
Department of Computer Science
University of Puerto Rico,
Río Piedras Campus
San Juan, PR 00936-8377
Email: ian.davila@upr.edu

José R. Ortiz-Ubarri
Department of Computer Science
University of Puerto Rico,
Río Piedras Campus
San Juan, PR 00936-8377
Email: jose.ortiz@hpcf.upr.edu

***Abstract*—Detecting network anomalies can help protect sensitive data, prevent attacks, and strengthen network security. Toa is an open source, pseudo-real time network monitoring system (NMS) that provides an easy to deploy web interface for system and network administrators to monitor high volumes of network traffic. Through visualization, this NMS allows the detection of network anomalies regarding availability, network and protocol performance, and network attacks, although not automatically.**

The current work presents an algorithm to optimize and automate the process of anomaly detection with adaptability to the context. For the purpose of our project, an anomaly is defined as anything that deviates from the normal behavior of the network, such as network malfunctions and network attacks. To this end, we implemented statistical algorithms that use distinct time ranges, and an exponential smoothing algorithm to facilitate anomaly detection.

1. Introduction

Due to the high level of complexity of current network systems, and an ever increasing frequency of network attacks, the automatic identification of anomalies in the network behaviour is of great importance. Network traffic flows represent complex data, and collecting and analyzing this data is a difficult endeavor [1][2][3].

System administrators have the difficult task of monitoring the network for anomalous traffic behavior such as outages, configuration changes, attacks, among others [2]. Using the Toa monitoring system, the input and output octet, as well as the input and output packet network flow data can be visualized to determine any kind of anomaly.Since the Toa system stores the network flow information within a database, the data is extracted using a series of SQL queries. Each data entry is an aggregate of the input and output octets, and input and output packets of the networks that are being monitored. This allows for easy access, making it simple and time-efficient to examine large portions of the data to continuously define network behavior models [2].

By comparing the multivariate octets, one can extrapolate relations between the data to determine what may be abnormal behavior. For example, you could have a very large amount of packets incoming, but each packet has a very tiny size. This would be considered an odd event, and would need to be checked out. If one determines an anomaly exists using the Toa Network, then the rest of the information pertaining to the event, including port, origin and destination, can be used to further assess the anomaly.

The goal of this research project is to implement an automated anomaly detection back-end feature into the Toa Networking Flow Monitoring System. This feature will eventually consist of various algorithms used simultaneously to try and detect anomalies. Since the anomaly detection algorithm relies on statistical methods and algorithms, a goal of this project was to examine results from different time ranges and different amounts of data sets.

2. Methodology

Algorithms were implemented using a series of queries that extract data from the Toa database and calculate the standard deviation and average for the time interval specified by the query. The queries took data from: (1) the past day, (2) the same hour every day for one week, (3) the same hour for one month, (4) the same hour for three months, (5) every five minutes for three months, (6) and intervals of thirty minutes from a specific hour every day. The idea of using different times ranges for the same data is to examine the network's behavior in different contexts, and to see how the smoothing algorithm reacts with more data. This data is processed to determine anomalies and normal behavior within each time range [2].

$$s_0 = x_0$$
$$s_t = \alpha x_t + (1 - \alpha) s_{t-1}$$

Figure 1. Exponential Smoothing Algorithm

First, multiple tests were run to determine that the queries were giving the correct time, standard deviation, average, and data set. After these tests were run, the exponential smoothing algorithm presented in Figure 1 was implemented. The implementation was used to create graphs in order to visualize the regular and smoothed data. With the smoothed data ready to be used, the standard deviation, along with a particular factor, were used to determine if a point is an anomaly. If the point examined minus the average is greater than the standard deviation times the factor, then that particular point is an anomaly. See Figure 2 for a graphical representation of how the anomaly detection process is carried out and alerts are generated.

After analyzing the data gathered by comparing the different queries and their anomaly detection counts, as well as their efficiency, a few of the queries were eliminated. We ended up with only the (1) every 5 minutes for 24 hours, (2) same time every day for 2 weeks, (3) same time, same day, every week for 2 months; each of these with regular and smoothed data. Currently, these queries are continuously running every 5 minutes using a cron job, comparing the last point only with the standard deviation, average and factor so as to simulate the algorithm at work with real data. The information generated by each query is being recorded onto text files, with each entry consisting of the current ID being examined, the query, the data point being examined, as well as that points regular and smoothed standard deviation, regular and smoothed average, input and output octets, and input and output packets.

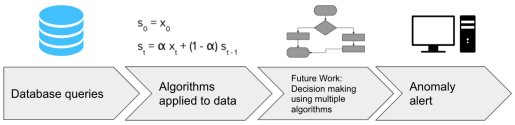

Figure 2. Anomaly detection and alert generation

3. Conclusion

In its current state, the anomaly detection system can be deployed to use with the Toa network flow monitoring system, but we want to run further tests with real and dummy data to further optimize the implementation. For the same time every day for two weeks and the same time, same day, every week for two months queries, it's necessary to test each one with a different number of data sets (15, 25, 35 and 5, 10, 15 respectively). Additionally, further tests must be run with varying alphas and factors to use with the standard deviation to determine if there is an anomaly present.

We want to implement a linear regression model to test, as well as look into clustering the data into segments to assess how efficient anomaly detection would be with an algorithm such as k Nearest Neighbor. This would require labeling the clusters of data in terms or normal or anomalous, therefore further work is necessary in order to find a way to define what an anomaly is within a contextual and time sensitive model for the network [3][4]. These algorithms must be tested the same way as the current smoothing algorithm to determine their efficiency. If they are determined to be of use to our cause, then we will consider implementing a majority vote algorithm where if the majority of the algorithms detect an anomaly in the current data point, then an alert is triggered. In the long run, we wish to implement an email system for alerts once we have reduced the amount of false positives.

Since the Toa system has a visualization platform already integrated, implementing visualization features to view the data for any of the algorithms to compare with the regular data would be useful. Creating a feature where one can select the specific time, amount of data sets, query and algorithm is left for future work; this feature would greatly facilitate examining the data. These visualization features would have to be able to extract the data from the database using time ranges selected by the user and then generate the graphs.

Acknowledgments

The authors would like to thank the members of the CSLab and the Computer Science Department. This work is partially supported by the National Science Foundation under Grant No. DUE-1438838. Presentation of this work has been supported in part by NSF Grant CNS-1042341.

References

[1] A. Patcha, J. Park *An overview of anomaly detection techniques: Existing solutions and latest technological trends*, Computer Networks Vol 51, 2007.

[2] J. Estevez-Tapiador, P. Garcia-Teodoro, J. Diaz-Verdejo, *Anomaly detection methods in wired networks: a survey and taxonomy*, Computer Communications Vol 27, 2004.

[3] P. García, J. Díaz, G. Maciá, E. Vázquez *Anomaly-based network intrusion detection: Techniques, systems and challenges* , Computers and Security Vol 28, 2009.

[4] N. Ye, S. Masum, Q. Chen, S. Vilbert *Multivariate Statistical Analysis of Audit Trails for Host-Based Intrusion Detection* , IEEE Transactions On Computers Vol 51 (7), 2009.