Detecting the Onset of a Network Layer DoS Attack with a Graph-Based Approach

Dr. William Eberle   
*Department of Computer Science  
Tennessee Tech University*Cookeville, TN  
weberle@tntech.edu

Ramesh Paudel  
*Department of Computer Science  
Tennessee Tech University*Cookeville, TN  
rpaudel42@students.tntech.edu

Peter Harlan  
*Department of Computer Science  
Western Kentucky University*Bowling Green, KY  
peterjharlan@gmail.com

*Abstract*— A denial-of-service (DoS) attack is a malicious act with the goal of interrupting the access to a computer network. The result of this type of attack can cause the computers on the network to squander their resources to serve illegitimate requests that result in a disruption of the network’s services to legitimate users. With a sophisticated DoS attack, it becomes difficult to distinguish malicious requests from legitimate requests. Since a network layer DoS attack can cause interruptions to a network while causing collateral damage, it is vital to understand the measures to mitigate against such attacks. Generally, approaches that implement distribution charts based on statistical analysis or honeypots have been applied to detect a DoS attack. However, this is usually too late, as the damage is already done. We hypothesize in this work that a graph-based approach can provide the capability to identify a DoS attack *at its inception*. A graph-based approach will also allow us to not only focus on anomalies within an entity (like a computer), but also allow us to analyze the anomalies that exist in an entity’s relationship with other entities, thus providing a rich source of contextual analysis. We demonstrate our proposed approach using a publicly-available data set.

Keywords—DoS/DDoS attack, graph-based anomaly detection, intrusion detection, network layer denial of service attack

# Introduction

A network-level DoS attack oversaturates a computer network with illegitimate traffic to prevent actual users from accessing the computer network’s services. The motivation behind such attacks can include but are not limited to revenge, prestige, politics, or money [1].An example of a network layer DoS attack is flooding a computer network with bogus packets which causes congestion on the network. To elaborate, the goal of a network layer DoS attack is to overflow a server/network with messages that have invalid return addresses, causing the targeted computer network to expend resources trying to direct packets to the fabricated address [2]. Since a DoS attack can cause serious repercussions, it is important to find the inception of the attack before actual damage has occurred. This can be done by analyzing the potential anomalies that exist on a computer network during the onset of the DoS attack.

Since data from a computer network can inherently be represented with a graph structure, where G = (V, E) and G is a graph that is composed of V as vertices and E as edges, it is possible to use a graph-based approach to help identify patterns in a computer network. The graph topology of a computer network is typically composed of nodes (or vertices) representing each device on the network, and the data that flows between two nodes as a directed edge (e.g., source de*vice 🡪*destination device). Thus, any changes to a graph’s structure can be viewed of as a potential anomaly. Another way to look at this is that a normative pattern would represent the expected traffic flow in a computer network, while *deviations* from the expected traffic flow would constitute an *anomaly*.

For the work presented in this paper, we will use a publicly-available graph-based anomaly detection tool (GBAD) and a publicly-available data set that represents a computer network with a known denial-of-service attack. In the following sections, we will present related work on denial-of-service attacks, and a brief introduction to the tool we used. We then discuss the data set, and how we created a graph from the data. We then conclude with our experimental results and how they compare to the ground-truth, analysis of this work, and where we plan to go in the future.

# Related Work

Numerous techniques have been developed to identify a network layer DoS attack, however, very little research has been conducted to identify the *inception* of a network layer DoS attack, especially when the network data is represented as a graph. Since multitudinous DoS attack detection techniques rely on a statistical distribution or honeypot approach, they do not have the ability to analyze the dataset in context. However, graph-based approaches rely on the *structure* of the interactions and relationships between the nodes in a network.

## DoS Attack Detection

There are various techniques used to identify DoS attacks. The traditional technique is to implement a statistical approach to discover a DoS attack, such as using an activity profiling [1], a machine learning classifier [2] or an autoregressive integrated moving average (ARIMA) time series model [3]. In addition, another common approach is to set up decoy machines called *honeypots* to discover a DoS attack [4].

Activity profiling analyzes the contents of a message packet (e.g., duration of communication, source, destination, time lapse between requests, etc.) and clusters them into their appropriate categories [1]. Once this is complete, using a chi-square goodness of fit test, each cluster’s activity level is compared to the expected activity level. Any activity levels detected beyond a reasonable threshold from the chi-square result will be flagged as an anomaly [1]. By using various machine learning classifiers like Naïve Bayes, Multilayer Perceptron, RBF network and Voted Perceptron, the incoming packets are classified as either attack or normal [2]. An ARIMA time series model is also a statistical based approach used to discover DoS attacks. This approach analyzes the different packets associated with the network and creates a time-based prediction using ARIMA. Traffic that falls outside the prediction is flagged as anomalous [3].

Another approach used to detect DoS attacks is honeypots. Honeypots, a proactive approach, are machines that are placed on the computer network with the intention of not receiving any legitimate traffic [4]. Any traffic that is associated with the honeypot is flagged as an anomalous instance [5]. However, the problem with honeypots is that they are deployed at fixed, detectable locations, thereby making it easier for sophisticated attacks to avoid the honeypots [6].

The deficiency of statistical approach is that they need the labeled data to train their model and the choice of the attribute they select also impact the performance of the system [7]. Moreover, that they do not have the ability to take relationships between multiple entities into consideration; thus, they are unable to discover anomalies that exist within the relationships, providing some context to the anomalies. Also, with a statistics-based approach to anomaly detection, changes and anomalies within a network may require more data for understanding normative patterns and the dynamics of the network.

## Graph-Based Approach

GraphPrints [14] divides network traffic into time slices. GraphPrints then mines small, induced subgraphs called graphlets where the building blocks of the graph and describe the local topography. It then performs outlier detection to find traffic times windows that exhibit an uncharacteristic graphlet count. GraphPrints uses a Minimum Covariance Determinant (MCD) method and normal distribution (Gaussian) to identify the anomalies in graph data [14]. In another approach, Miller et al. propose three goodness-of-fit statistics for Chung–Lu random graphs [9], and analyze their efficacy in discriminating graphs generated by the Chung–Lu model from those with anomalous topologies [13]. In addition, Noble and Cook [10] defined methods for detecting unusual patterns within graph-based data and introduced a measure for calculating the regularity of a graph, using the concept of conditional entropy.

Overall, traditional research conducted on DoS attacks is based on statistical methods using distribution charts or honeypot machines. Furthermore, the majority of the research on graph-based anomaly detection is rooted in statistics [13, 14]; as seen in the GraphPrints and the Chung-Lu model anomaly detection tool. In short, the approach used in this research does not rely on statistical methods to discover anomalies in graph data like other graph-based anomaly detection tools, and instead analyzes the structure of the network.

# GBAD

The idea behind the approach used in our work is to find anomalies in graph-based data where the anomalous substructure in a graph is part of (or attached to or missing from) a *normative pattern,* which in our implementation is a substructure that minimizesthe description length (MDL) of a graph.

The advantage of graph-based anomaly detection is that the relationships between entities can be analyzed for structural oddities in what could be a rich set of information, as opposed to just the entities’ attributes. However, graph-based approaches have been prohibitive due to computational constraints, because graph-based approaches typically perform subgraph isomorphism, a known NP-complete problem. Yet, in order to use graph-based anomaly detection techniques in a real-world environment, we need to take advantage of the structural/relational aspects found in dynamic, streaming data.

**Definition**: *A graph substructure S’ is anomalous if it is not isomorphic to the graph’s normative substructure S, but is isomorphic to S within X%.*

*X* signifies the percentage of vertices and edges that would need to be changed in order for *S’* to be isomorphic to *S*. The importance of this definition lies in its relationship to any deceptive practices that are intended to illegally obtain or hide information. The United Nations Office on Drugs and Crime states the first fundamental law of money laundering as “The more successful money-laundering apparatus is in imitating the patterns and behavior of legitimate transactions, the less the likelihood of it being exposed” [11].

GBAD (Graph-based Anomaly Detection) is an *unsupervised* approach, based upon the SUBDUE graph-based knowledge discovery method [12]. Using a greedy beam search and MDL heuristic, each of the three anomaly detection algorithms in GBAD uses SUBDUE to find the best substructure, or normative pattern, in an input graph. In our implementation, the MDL approach is used to determine the best substructure(s) as the one that minimizes the following:

**

where *G* is the entire graph, *S* is the substructure, *DL(G|S)* is the description length of *G* after compressing it using *S*, and *DL(S)* is the description length of the substructure.

There are three general *categories of anomalies*: insertions, modifications, and deletions. Insertions would constitute the presence of an unexpected vertex or edge. Modifications would consist of an unexpected label on a vertex or edge. Deletions would constitute the unexpected absence of a vertex or edge. We have developed three separate algorithms: GBAD-MDL, GBAD-P, and GBAD-MPS. Each of these approaches is intended to discover one of the corresponding possible graph-based anomaly categories as set forth earlier. The reader should refer to Eberle and Holder’s work for a more detailed description of the actual algorithms [8]. We will use the GBAD tool throughout our experiments.

# Dataset

The data set used throughout this research is gathered from Visual Analytics Science and Technology (VAST) 2011 mini challenge 2. The dataset consists of firewall logs, IDS logs, syslogs for all hosts on the network, and the network vulnerability scan report from All Freight Corporation’s computer network. The focus of this research will be on the firewall log because it keeps record of all traffic events in the network (internal as well as external network events). Although the VAST dataset captures three days of traffic on the All Freight Corporation’s computer network, we choose data from day one because the ground truth of the data indicates that the DoS attack started at 11:39 am and ends at 12:51 pm on day one. Since our main focus is to detect the *onset* of DoS attack, we decided to use the firewall log from 08:52:52 am (beginning of the day) to 11:50:59 am (11 minute after initiation of DoS attack). Our choice was driven by the fact that we wanted to include enough data that will capture the nature of traffic flow during the initialization of the DoS attack (but not the complete DoS attack traffic) so that we will be able to analyze the effect of the attack (from a graph perspective) on the network at its infancy. After several trials, we decided to use the initial 11 minutes of the DoS attack traffic. It should be noted that the choice of 11 minutes was somewhat arbitrary and not specific to the approach chosen. Table Ishows a description of the data in the firewall log.

The reason we chose to use the VAST 2011 challenge dataset is that it contains ground truth, which will enable us to evaluate the effectiveness of our approach on identifying the DoS attack on All Freight Corporation’s computer network *at its inception*. Another reason behind the selection of this data set was that it contained information about All Freight Corporation’s network topology (i.e., what was on the network and how it was connected – something a network analyst would have access to). Moreover, this dataset also includes a description of the normative functionalities of the different devices (i.e., computers, server, switches, subnets, etc.) that exist on the computer network which will aide in the design of an effective graph topology for anomaly (intrusion) detection.



Fig. 3. Sample graph topology showing source and destination node

1. Firewall Log Data Description

|  |  |
| --- | --- |
| Attributes | Description |
| ***Date/time*** | Date and time when activity was performed |
| ***Syslog priority*** | Priority of the log message |
| ***Operation*** | Type of activity being performed |
| ***Message code*** | Code associated with the message |
| ***Protocol*** | Connection protocol type |
| ***Source IP*** | Source IP associated with the activity (this field may be left empty for some log messages) |
| ***Destination IP*** | The destination IP associated with the activity (this field may be left empty for some log messages) |
| ***Source hostname*** | Name of the host machine associated with the activity (this field is empty) |
| ***Destination hostname*** | Name of the host machine associated with the activity (this field is empty) |
| ***Source port*** | Port associated with the source IP in this activity (this field may be left empty for some log messages) |
| ***Destination port*** | Port associated with the destination IP for this activity (this field may be left empty for some log messages) |
| ***Destination service*** | Name of the service associated with the destination port |
| ***Direction*** | This field is empty in this dataset |
| ***Connections built*** | Number of connections built in this operation |
| ***Connections torn down*** | Number of connections torn down in this operation |

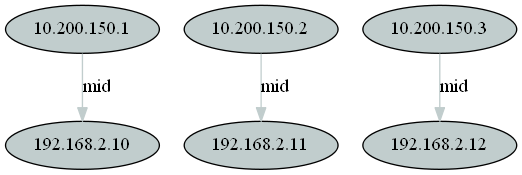


Fig 1. Ungrouped device

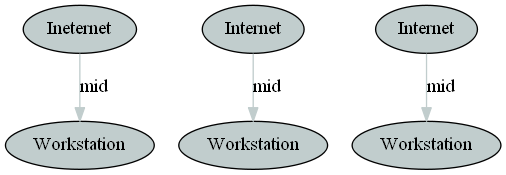


Fig. 2. Grouped device

# Data Preparation

Before the anomalies in the firewall log can be analyzed by GBAD, the dataset must be converted into a graph input file. This process will be conducted by a parser script which converts the firewall log into a graph file. In order for the research community to be able to replicate our results, this section describes the algorithm behind the parser script. (Upon publication, the dataset and tools will be made publicly available [15].)

The first step is to convert IP addresses into device descriptions such as “DNS server”, “web server”, “workstation”, etc. This process also groups devices by their type. For example, IPs in 192.168.2.10–250 were labeled as *workstations.* Likewise, all external devices communicating with All Freight’s network with the IP address of 10.200.150.1-255 were marked as *internet*. The idea is that this logical grouping of devices will make discovering patterns and behavior of similar devices easier. If this grouping was not applied to the data set, then there would be too many unique devices as well as too many connections between each unique device, which would make it harder to discover any common (normative) behavior in the network. For instance, *Fig. 1* shows an example of a graph that can be formed using a unique node without the logical groupings, and *Fig. 2* shows an example where similar devices are grouped (i.e., Internet 🡪 Workstation with an edge labeled “mid”). Once this was completed, the next process was to convert each tuple in the firewall log to nodes and edges in a graph. At first, each structurally independent graph will be divided by a specified time interval. Connections between the same group of vertices in the data will be treated as a single edge (e.g., if there are 50 different communications between *internet* [source device] and *web server* [destination device], a single edge between vertex *internet* and *web server* will be generated) with a label “*mid*” or “*high*”. If the connection count is two standard deviations above the mean of *all* similar traffic counts, the edge will be given a label of “*high*”; otherwise, the edge will be labeled as “*mid*”. *Fig. 3* demonstrates the blueprint to the described topology. It should also be noted that none of the traffic counts were “low” (not surprising given that we are dealing with a denial of service attack), but there is nothing that we implemented that would have prevented us from having such a label.

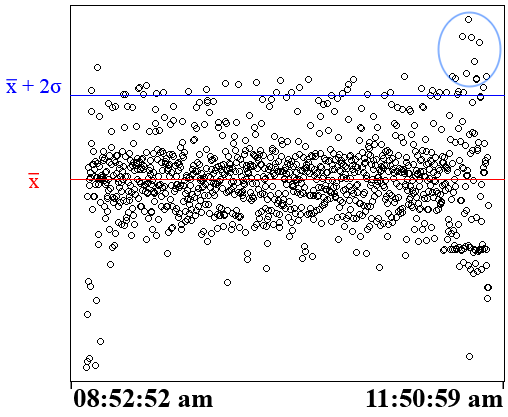


Fig 4. Number of connection from internet to web server

# Ground Truth

The dataset that was provided by the VAST competition contains ground truth; this allows the results of the research to be compared for accuracy. In detail, the ground truth describes that a DoS occurred on All Freight Corporation’s computer network on day one of the three days’ worth of captured network traffic at 11:39:51 am. VAST describes that there were five individual systems on the internet that participated in the DoS attack on the corporate’s web server (which hosts All Freight’s external web site). Finally, it was described by the ground truth that the IDS log was able to pick up on the DoS attack that occurred on the network at 11:43:29 am - 3 minutes and 39 seconds *after* the IDS reported the denial of service attack.

# Experiments

The first step in the experimental process was to understand the scope of the VAST data set. This process entailed the study of All Freight’s computer network composition. This process was important because it contributed to the design of the graph topology. In addition, by having the ground truth, the results that are derived from this research can be compared for accuracy.

After the data set was parsed (as described in the data preparation section) the main experimental process began. Several types of topologies for the graph were created based on different time intervals. First, individual graphs were created that had the same timestamp. Then, other topologies were created using varying time intervals: 1.25 seconds, 2.5 seconds, 5 seconds, and 8 seconds. The number of vertices and edges for each topology is shown in Table II. When the graphs are grouped by the same timestamp (i.e., 0 sec intervals), it results in almost twice as many graphs as the next time interval (i.e., 1.25 seconds). Similarly, we discovered that creating individual graphs using 8 seconds intervals generalizes the data too much, and results in smaller normative patterns. In short, too many subgraphs were created when segmented by matching timestamps (0 sec intervals) causing the DoS attack to be considered a normative pattern; while segmenting based on 8 second intervals generalized the data too much, resulting in uninteresting normative patterns. After many different experiments, it was concluded that 2.5 seconds was the right equilibrium on the spectrum. 1.5 second and 5 second time intervals had similar issues that plagued the grouping of the graph by the same timestamp (0 sec interval) or 8 seconds. Overall, the 2.5 second time intervals were able to segregate the traffic into appropriate proportions where the DoS attack does not become the normative pattern and the network traffic does not become too generalized.

1. Graph Topoloy based on Time Intervals and Graph counts

|  |  |  |  |
| --- | --- | --- | --- |
| Single Graph Interval | # of Vertices | # of Edges | # of graphs |
| **0 Second** | 68,267 | 59,588 | 8,478 |
| **1.25 Second** | 49,197 | 45,007 | 4,629 |
| **2.5 Second** | 42,032 | 39,957 | 3,201 |
| **5 Second** | 33,544 | 34,543 | 1,691 |
| **8 Second** | 29,607 | 32,282 | 1,140 |

To understand the amount of traffic arriving on the external web server from the internet, we first calculated the number of connections during each 2.5 second interval. The scatter plot of the traffic count over time is shown in Fig. 4, where the mean number of connections between internet and external web server are marked by , and the connections that are higher than the mean by two standard deviations are marked by . Any values that are above correspond to the edge with the label “*high*”, and below corresponds to the edge with lable “*mid*”. Thus, the scatter plot indicates that “*high*” edges are more prominet around and before 11:50am (during the active DoS attack, shown in the blue circle). However, the scatter plot also has many “*high*” edges that appear before DoS attack. Using this statistical approach, false positive rates will be high. So, instead of just using the count or similar attributes as is deployed in many statistical approaches, using a graph based approach will allow us to explore the relationship between various devices in the network to discover the anomaly (i.e., beginning of the DoS attack).

The input graph generated by the parser was fed into to the graph-based anomaly detection algorithm GBAD tool. GBAD then uses a compression technique to discover the normative patterns in the data set; the normative patterns are then used to identify the anomalous structures. In other words, GBAD analyzes the complete dataset through the lens of the selected normative pattern in order to label the anomaly. The normative pattern discovered on the 2.5 second interval graph and the associated anomalous instances are shown in Fig 5 and 6. The normative pattern shown in Fig 5(a) indicates that the number of connections from several internet devices (three in this case) to the external web server is “*mid*”. We use this normative pattern (Fig 5 (a)) to look for anomalous insertions using the GBAD-P algorithm, and the reported anomalous substructure (as shown in Fig 5(c)) has an extra node “*internet*” with the label “*high*” (indicated by white node and an edge with black color). This indicates that a certain device from internet is sending a “*high*” number of traffic to the external web server while other devices are sending an expected amount of traffic. GBAD-P reported 48 anomalous instances of this type. Upon further inspection, we found that all 48 anomalous instances reported occurred during the DoS attack with the first occurrence at 11:40:22 am - i.e., 31 seconds after the inception of the DoS attack. And all 48 instances were from one of three IPs, *10.200.150.<201, 206 and 209>*, indicating the DoS attack is being carried out using these three machines – which are three of the five machines that according to the ground truth proliferated the attack. We further used the same normative pattern (Fig 5 (a)) to look for anomalous modification using the GBAD-MDL algorithm and discovered the anomaly as shown in Fig 5(b) where the edge label (from “*internet*” to “*external web*”) was modified to “*high*” instead of “*mid*” as suggested by the normative pattern. In this case, there was just one instance of this anomaly. But, the noticeable fact was that the “*internet”* node was associated with IP *10.200.150.206* which is one of the three machine used in DoS attack. And, this anomaly occurs on the subgraph representing data at 11:40:07 am - i.e., after only 16 seconds of the inception of the DoS attack.



Fig. 5. (a) Normative Pattern I

b) Anomalous modification (edge “*high*”) discovered using GBAD-MDL

c) Anomalous insertion (extra node and edge) discovered using GBAD-P

Fig. 6. (a) Normative Pattern II

b) Anomalous insertion (extra node and edge) discovered using GBAD-P

In order to see if we can further reduce the DoS attack detection delay, we used another subgraph shown in Fig 6(a) as a normative pattern. This subgraph has two internet devices (instead of the three in the earlier case) communicating with the web server. As a result, the GBAD-P algorithm reports an anomalous insertion on this normative pattern. The anomaly as shown in Fig 5(b) where extra node “*DNS*” is hanging off the node “*external web*” (indicated by the white node and an edge with black color) was reported. There was just one instance of this anomaly but it was associated with the graph representing the data at 11:39:55 am - i.e. only 4 seconds after the inception of the DoS attack. No other anomalies are reported using the other GBAD algorithms.

# Analysis

Meaningful results from the firewall log that is concerned with the inception of the DoS attack was discovered using a graph-based anomaly detection technique. This discovery is interesting because the first instance of an anomaly reported by GBAD was at 11:39:55 am while according to the Ground truth, the inception of the DoS attack initialized at 11:39:51 am (i.e., after 4 seconds of DoS attack initialization). Also, the early detection of the anomaly was possible because the graph-based approach was able to represent the direct repercussions of the attack (e.g., calls to the DNS servers by the external web server). This resulted in the change in the relationship between the entities in the network. In this particular scenario, the new relationship between the external web and the DNS was created which was represented as new node “*DNS*” hanging off “*external web*”. Furthermore, a graph-based approach considers context and relationship between various entities potentially making it more comprehensive than a statistical approach. For example, Fig. 4 (scatter plot) obtained using a simple statistical approach has lots of “*high*” edges that do not occur during the DoS attack time (“*high*” edges outside the blue circle). This is because the statistical approach takes each data individually and doesn’t consider them in the context of the others. However, using the graph-based approach we are able to get rid of all the “*high*” edges that were occurring beyond the DoS attack time by examining the context. For example, the anomaly discovered by the graph-based approach in Fig. 5(c) marked an “*internet*” node with an edge “*high*” as an anomaly only if there are three other “*internet*” nodes with an edge of “*mid*”. Also, it should be noted that every anomaly reported is related to the DoS attack. Thus, there are not any false positives. While a nice feature of what was performed here, we know that it shouldn’t be taken as a standard for applying a graph-based approach, and potentially another data set might have produced false positives (something we plan to investigate in the future).

One may suggest that the results returned by our experiment can be considered a coincidence. However, we argue that the results can be justified with the principles associated with a DoS attack and DNS queries. As we know, the goal of a network DoS attack is to create bogus return addresses, causing the network to squander its resources, thus preventing access to legitimate users. Since the local web server does not know the bogus return address associated with the packets sent by the DoS attack, the web servers must perform a DNS query; this logical explanation supports our understanding of why the DNS servers were connected to the web server during the inception of the DoS attack. We might argue that the anomalous pattern suggested by Fig 6(b) was because of some benign scenario like the employees at All Freight Corporation accessing websites that either do not exist or are not cached in the web server. But the other anomaly shown in Fig 5(b) that appears after 16 seconds with a “*high*” amount of traffic from IP 10.200.150.206 (one of the attacking devices) supports our explanation. Moreover, the 48 instances of the anomalous pattern that start occurring after 31 seconds, as shown in *Fig. 5(b),* also further supports this assertion. We also discovered 3 out of the 5 devices (with IP *10.200.150. <201, 206 and 209>*) that are involved in the DoS attack through the unusually high traffic volume.

Our experiments demonstrate that instead of following the usual approach of trying to identify a DoS attack by measuring it directly (measuring an intense spike in the traffic), it is possible to analyze the direct repercussions (e.g., unique calls to the DNS servers by the external web server) of the attack to find its inception. In the end, a graph-based approach has the capability to identify anomalies, deviations from the normative traffic patterns on a computer network, which can be associated with the inception of a DoS attack.

# Conclusion and Future Work

Graphs are a logical choice for representing computer networks and data. The graph topology of a computer network is typically composed of nodes (or vertices) representing each device on the network, and the data that flows between two nodes as a directed edge. In this research, we claimed that a graph based approach can represent the direct repercussions of the DoS attack and discover a potential DoS attack in its early stage. The first known anomaly was reported within 4 seconds of the DoS attack inception. Also, we were able to identify three IPs from which the DoS attack was instigated by sending large amounts of traffic to the external web server.

Although the GBAD tool identified the anomalous instances related to the DoS attack after only 4 seconds, data was not processed in real-time (i.e., as data traversed the network), and a static view of the data was processed in about 1512.05 seconds (25.2008 minutes). Note that the experiments were performed on an Intel(R) Xeon(R) 2.27GHz machine with 8 CPUs. The time constraints associated with detecting a DoS attack in real time needs to be factored. To address this issue, we suggest looking into the implementation of a component that allows GBAD to analyze *streaming* data. We believe this can be accomplished by analyzing the network traffic in smaller partitions using a sliding window protocol. Another possible area to investigate is using a different definition of an anomaly than what is used in tools like GBAD. In the future, we plan on not only investigating both of these ideas but experimenting on new, possibly streaming, data sets that represent known network attacks.

##### Acknowledgments

This work is supported by US National Science Foundation under the grant number 1560434. The statements made herein are solely the responsibility of the authors.

##### References

|  |  |
| --- | --- |
| [1] | G. Carl, G. Kesidis, R. Brooks and S. Rai, "Denial-of-Service Attack Detection Techniques," *IEEE Internet Computing,* vol. 10, no. 1, pp. 82-89, 2006. |
| [2] | K. Singh and T. De, "Analysis of Application Layer DDoS Attack Detection Parameters Using Statistical Classifiers," *Internetworking Indonesia Journal,* vol. 9, no. 2, pp. 23-32, 2017. |
| [3] | S. M. T. Nezhad, M. Nazari and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks," *IEEE Communications Letters,* vol. 20, no. 4, pp. 700-703, 2016. |
| [4] | N. Weiler, "Honeypot for Distributed Denial of Service Attacks," in *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2002. WET ICE 2002. Proceedings. Eleventh IEEE International Workshops on*, 2002. |
| [5] | A. Mairh, D. Barik, K. Verma and D. Jena, "Honeypot in Network Security: A Survey," in *Proceedings of the 2011 international conference on communication, computing \& security*, 2011. |
| [6] | C. Navenna and R. Sasikala, "Analyse Honey Pot Traffics to Detect DoS Attacks Using Support Vector Machine," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology,* vol. 2, no. 6, pp. 326-329, 2017. |
| [7] | M. Alenezi and M. J. Reed, "Methodologies for detecting DoS/DDoS attacks against network servers," in *Proceedings of the Seventh International Conference on Systems and Networks Communications—ICSNC*, 2012. |
| [8] | W. Eberle and L. Holder, "Anomaly Detection in Data Represented as Graphs," *Intelligent Data Analysis,* vol. 11, no. 6, pp. 663-689, 2007. |
| [9] | F. Chung, L. Lu and V. Vu, "The spectra of random graphs with given expected degrees," *Internet Mathematics,* vol. 1, no. 3, pp. 257-275, 2004. |
| [10] | C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003. |
| [11] | M. P. Hampton and M. Levi, "Fast spinning into oblivion? Recent developments in money-laundering policies and offshore finance centres," *Third World Quarterly,* vol. 20, no. 3, pp. 645--656, 1999. |
| [12] | L. B. Holder and D. J. Cook, "Graph-Based Data Mining," *IEEE Intelligent System,* vol. 15, no. 2, pp. 32-41, 2000. |
| [13] | B. Miller, L. Stephens and N. Bliss, "Goodness-of-fit statistics for anomaly detection in Chung-Lu random graphs," *2012 IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 25-30, 2012. |
| [14] | C. Harshaw, R. Bridges, M. Lannacone, J. Reed and J. Goodall, "GraphPrints: Towards a Graph Analytic Method for Network Anomaly Detection," in *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, 2016. |
| [15] | [TBD] Dataset & Experimentation Tool Publication URL |