The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

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

Ramesh Paudel,[1] Peter Harlan,[2] William Eberle[3]
[1,3]Tennessee Technological University, Cookeville, TN
[2] Western Kentucky University, Bowling Green, KY
rpaudel42@students.tntech.edu, peterjharlan@gmail.com, weberle@tntech.edu

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 DoS 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 dataset.

Introduction

A network-level DoS attack over-saturates 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 (Carl et al. 2006). 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 (Singh and De 2017). 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 network can inherently be represented with a graph structure, it is possible to use a graph-based approach to help identify patterns in the 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 device $\rightarrow$ destination device). Thus, any changes to a graph's structure can be viewed 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) (Eberle and Holder 2007) and a publicly available dataset 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 dataset, and how we created a graph from the data. We then conclude with our experimental results and analysis, 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 most 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

The traditional technique to identify DoS attacks is to implement a statistical approach to discover a DoS attack, such as using activity profiling (Carl et al. 2006), a machine learning classifier (Singh and De 2017) or an autoregressive integrated moving average (ARIMA) time series model (Nezhad, Nazari, and Gharavol 2016). In addition, another common approach is to set up decoy machines called honeypots to discover a DoS attack (Weiler 2002).

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 (Carl et al. 2006). 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. 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 (Singh and De 2017). An ARIMA time series model is also a statistically 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 (Nezhad, Nazari, and Gharavol 2016).

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 (Weiler 2002). Any traffic that is associated with the honeypot is flagged as an anomalous instance (Mairh et al. 2011). 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 (Navenna and Sasikala 2017).

The deficiency of most statistical approaches is that they need the labeled data to train their model and the choice of the attribute they select also impacts the performance of the system (Alenezi and Reed 2012). 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 connections, 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

Graph-based approaches have been successfully applied for anomaly detection in a wide array of applications (Akoglu, Tong, and Koutra 2015). In a computer network, a graph-based approach is used with considerable success for anomaly detection. (Iliofotou et al. 2007) used traffic dispersion graphs, to analyze, monitor, visualize, and classify network traffic. (Sun et al. 2008) employed Compact Matrix Decomposition (CMD) to decompose the adjacency matrix of the network graph and use relative sum-square-error of reconstruction as a measure of change to track new snapshots of the network graph over time. (Ding et al. 2012) monitored cross-community communication behavior to spot network intrusions. GraphPrints (Harshaw et al. 2016) divides network traffic into time slices and mines small, induced subgraphs called graphlets (the building blocks of the graph describe the local topography). It then performs outlier detection to find traffic time windows that exhibit an uncharacteristic graphlet count. (Miller, Stephens, and Bliss 2012) proposed three goodness-of-fit statistics for Chung-Lu random graphs (Chung, Lu, and Vu 2004), and analyzed their efficacy in discriminating graphs generated by the Chung-Lu model from those with anomalous topologies. In addition, (Noble and Cook 2003) 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; 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, but instead, analyzes the structure of the network.

GBAD

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. The idea behind the GBAD approach used in this work is to discover anomalies in graph-based data where the anomalous substructure in a graph is part of (or attached to or missing from) a normative pattern that minimizes the description length (MDL) of a graph.

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" (Hampton and Levi 1999). GBAD (Graph-based Anomaly Detection) is an unsupervised approach, based upon the SUBDUE graph-based knowledge discovery method (Holder and Cook 2005). 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:

$$M(S, G) = DL(G|S) + DL(S)$$

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: additions, 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. Each of these approaches is intended to discover one of the corresponding possible graph-based anomaly categories. The reader should refer to (Eberle and

Holder 2007) for a more detailed description of the actual algorithms.

Dataset

The dataset used in this work 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 of a fictional organization called All Freight Corporation. The focus of this research will be on the firewall log because it keeps a record of all traffic events in the network (internal as well as external network events). Although the VAST dataset captures three days of traffic, 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. Also, the ground truth indicated that five individual systems from the internet participated in the DoS attack on the external web server, and it took the IDS 3 minutes and 39 seconds to log the initial attack. Since our main focus is to detect the onset of the DoS attack, we decided to use the firewall log from 08:52:52 am (beginning of the day) to 11:50:59 am (11 minutes after initiation of the 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. It should be noted that the choice of 11 minutes was somewhat arbitrary and not specific to the approach chosen.

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 the computer network at its inception. Another reason behind the selection of this dataset was that it contained information about 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 aid in the design of an effective graph topology for anomaly (intrusion) detection.

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 step is achieved using a parsing parser script (written in python) which converts the firewall log into a graph file. The following describes in detail the steps taken to convert the input data to a graph. For replication by the research community, the dataset and a parser tool are publicly available at https://rpaudel42.github.io/pages/dataset.html.

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 communicat-

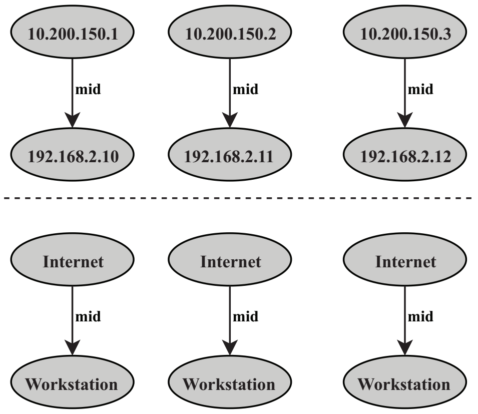

Figure 1: a) Ungrouped (top) and b) Grouped Devices (bottom)

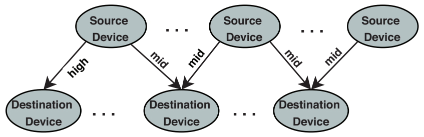

Figure 2: Sample graph topology

ing with the device in the 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 dataset, 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 (a) shows an example of a graph that can be formed using a unique node without the logical groupings, and Fig. 1 (b) 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. 2 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. It should be noted that the actual parsing of the data into its corresponding graph input file only takes ≈ 90 seconds.

Graph Interval	**# of Vertices**	**# of Edges**	**# of Graphs**		
			Normal	**DoS**	**Total**
0 Sec	68,267	59,588	7,801	677	8,478
1.25 Sec	49,197	45,007	4,295	344	4,629
2.5 Sec	42,032	39,957	2,962	239	3,210
5 Sec	33,544	33,543	1,580	111	1,691
8 Sec	29,607	32,282	1,066	74	1,140

Table 1: Graph topology based on time intervals and graph counts

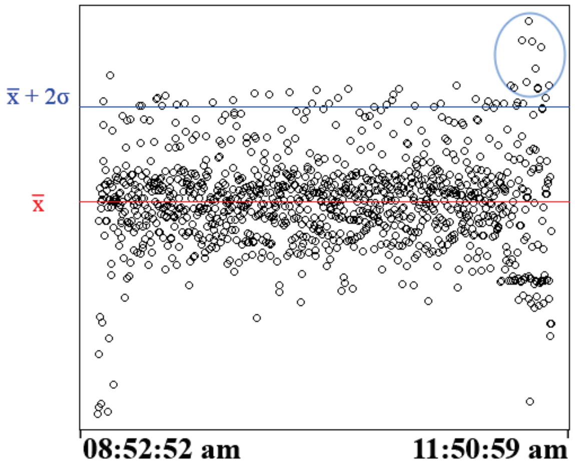

Figure 3: Number of connection from internet to web server

Experiments

The first step in the experimental process was to understand the scope of the VAST dataset. This process entailed the study of an organization'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.

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. 3, where the mean number of connections between the internet and the external web server are marked by $\overline{x}$, and the connections that are higher than the mean by two standard deviations are marked by $\overline{x} + 2\sigma$. 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 11:50 am (during the active DoS attack, shown in the blue circle). However, the scatter plot also has many *"high"* edges that appear before the DoS attack. Using this simple 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., the beginning of the DoS attack).

Graph interval	**Anom. graph reported**	**Attack source reported**	**Detection delay (sec)**	**Runtime (sec)**
0 Sec	6.35%	**5**	31	482
1.25 Sec	4.2%	4	612	289
2.5 Sec	18.4%	3	31	257
5 Sec	**96.4%**	**5**	23	118
8 Sec	1.35%	0	**4**	**102**

Table 2: Performance of GBAD on different graph topology (using normative pattern shown in Fig 4(a))

Several types of topologies for the graph were created based on different time intervals (0 sec intervals, 1.25 sec, 2.5 sec, 5 sec, and 8 sec). The number of vertices, edges, and graphs (normal, DoS attack and total) for each topology is shown in Table 1. Normal graphs are the individual graphs that does not contain a node representing known DoS attacks IP while DoS attack graphs are the individual graphs (usually constructed after the inception of the DoS attack) that contain a node representing at least one of the five known DoS attacks IP on the internet. The input graph generated by the parser was fed into the graph-based anomaly detection algorithm (GBAD) tool. GBAD then uses a compression technique to discover the normative patterns in the dataset, which 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. When the graphs are grouped by the same timestamp, 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 second intervals generalizes the data too much, and results in bigger normative patterns. In short, too many graphs with few vertices (graph to vertex ratio $\approx$ 1:8) 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 (graph to vertex ratio 1:26), resulting in uninteresting and larger normative patterns.

The normative pattern shown in Fig 4(a) was used for anomaly detection in all five graphs. The results are summarized in Table 2. Runtime was calculated by taking the average of 5 experiments for each topology. DoS attack detection delay represents the time between the inception of the DoS attack and the earliest time represented by the anomalous instance reported. Only 0-second and 5-second interval graphs were able to detect all 5 DoS attack sources. However, only 6.35% of the anomalous substructures were detected using a 0-second interval graph and the total runtime was higher than using other graph topologies. The 8-second interval graph had the lowest runtime but due to the generalized graph, it was unable to detect any DoS attack sources. Overall, the 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. Now we will explain the results of anomaly detection on the 5-second graph.

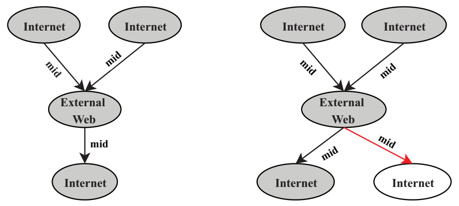

Figure 4: a) Normative Pattern I b) Anomalous addition (extra node and edge)

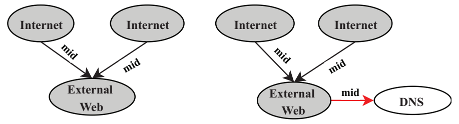

Figure 5: a) Normative Pattern II b) Anomalous addition (extra node and edge)

The normative pattern and the associated anomalous instances are shown in Fig 4 and 5. The normative pattern shown in Fig 4(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 4 (a)) to look for anomalous additions, and the reported anomalous substructure (as shown in Fig 4(b)) has an extra node *"internet"* with the label *"mid"* (indicated by a white node and a red edge). This indicates that a certain set of device (4 or 5 specific devices) from the internet are continuously sending traffic to the external web server. GBAD reports 215 anomalous instances of this type of anomaly associated with 107 (out of 111) DoS attack graphs. Upon further inspection, we found that all 215 anomalous instances reported occurred during the DoS attack with the first occurrence at 11:40:13 am - i.e., 23 seconds after the inception of the DoS attack. And all 215 instances of anomalous nodes were associated with one of the five IPs, *10.200.150.<201, 206, 207, 208 and 209>*, indicating the DoS attack is being carried out using these five machines which according to the ground truth are the five machines that proliferated the attack. In order to see if we can further reduce the DoS attack detection delay, we used another subgraph shown in Fig 5(a) as a normative pattern. This subgraph has two internet devices (instead of the three in the earlier case) communicating with the web server. GBAD reports an anomalous addition to this normative pattern as shown in Fig 5(b) where an extra *"DNS"* node is hanging off the node *"external web"* (indicated by the white node and a red edge). There were just two instances of this anomaly but it was associated with the graph representing the data at 11:39:56 am - i.e. only 5 seconds after the inception of the DoS attack. No other anomalies are reported using the other GBAD algorithms. The confusion matrix showing the result of anomaly detection using GBAD's anomalous addition's algorithm on the 5-second interval graphs using normative pattern from Fig 4(a) is shown in Table 3. Only 4 graphs out of 111 graphs associated with DoS attack were missed by GBAD.

	Predicted (DoS)	**Predicted (Normal)**
Actual (DoS)	107 (TP)	4 (FN)
Actual (Normal)	0 (FP)	1580 (TN)

Table 3: Confusion matrix for GBAD for 5 second graph using normative pattern shown in Fig 4(a)

Analysis

The result given by GBAD is interesting because the first instance of an anomaly reported was at 11:39:56 AM while according to the ground truth, the inception of the DoS attack initialized at 11:39:51 AM (i.e., only 5 seconds after the inception of the DoS attack). 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). 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 resulted in the change in the graph structure 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 a 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. 3 (scatter plot) obtained using simple statistics (traffic count) has many *"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 point individually and does not consider them in the context of the others. However, the graph-based approach was able to discover anomalous nodes/edges associated with DoS attack graphs by analyzing the relationship between several entities. For example, the substructure in Fig. 4(b) is anomalous because 4 sets of devices are sending continuous traffic to the web server (instead of 1 device sending high traffic in case of statistical approach). Similarly, in Fig 5(b) the web server had to perform a DNS query to the DNS servers during the inception of the DoS attack to identify the bogus return address sent by the attack machine. This unusual behavior of the web server was marked as an anomaly by GBAD which helped to flag the DoS attack at its inception. Also, it should be noted that every anomaly reported by GBAD is related to the DoS attack. Thus, there are not any false positives (see confusion matrix in Table 3). While a nice feature of what was performed here, we know that it should not be taken as a standard for applying a graph-based approach, and potentially another dataset might have produced false positives (something we plan to investigate in the future).

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 effects (e.g., unique calls to the DNS servers by the external web server) of the attack to discover its inception. In the end, a graph-based approach has the capability to identify anomalies, or 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 claim that a graph based approach can represent the direct repercussions of the DoS attack and discover a potential DoS attack in its early stages. The first known anomaly was reported within 5 seconds of the DoS attack inception. Also, we were able to identify all five IPs from which the DoS attack was instigated to the external web server.

Although the GBAD tool identified the anomalous instances related to the DoS attack after only 5 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 118 seconds. Note that the experiments were performed on Intel Core i5 2.6 GHz machine with 8 GB 1600 MHz DDR3. 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 GBAD. In the future, we plan on not only investigating both of these ideas but experimenting on real-world, possibly streaming, datasets that represent known network attacks.

Acknowledgement

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

Akoglu, L.; Tong, H.; and Koutra, D. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29(3):626–688.

Alenezi, M., and Reed, M. J. 2012. Methodologies for detecting dos/ddos attacks against network servers. In *Proceedings of the Seventh International Conference on Systems and Networks Communications—ICSNC*.

Carl, G.; Kesidis, G.; Brooks, R. R.; and Rai, S. 2006. Denial-of-service attack-detection techniques. *IEEE Internet computing* 10(1):82–89.

Chung, F.; Lu, L.; and Vu, V. 2004. The spectra of random graphs with given expected degrees. *Internet Mathematics* 1(3):257–275.

Ding, Q.; Katenka, N.; Barford, P.; Kolaczyk, E.; and Crovella, M. 2012. Intrusion as (anti) social communication: characterization and detection. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 886–894. ACM.

Eberle, W., and Holder, L. 2007. Anomaly detection in data represented as graphs. *Intelligent Data Analysis* 11(6):663–689.

Hampton, M. P., and Levi, M. 1999. Fast spinning into oblivion? recent developments in money-laundering policies and offshore finance centres. *Third World Quarterly* 20(3):645–656.

Harshaw, C. R.; Bridges, R. A.; Iannacone, M. D.; Reed, J. W.; and Goodall, J. R. 2016. Graphprints: Towards a graph analytic method for network anomaly detection. In *Proceedings of the 11th Annual Cyber and Information Security Research Conference*, 15. ACM.

Holder, L. B., and Cook, D. J. 2005. Graph-based data mining. In *Encyclopedia of data warehousing and mining*. IGI Global. 540–545.

Iliofotou, M.; Pappu, P.; Faloutsos, M.; Mitzenmacher, M.; Singh, S.; and Varghese, G. 2007. Network monitoring using traffic dispersion graphs (tdgs). In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 315–320. ACM.

Mairh, A.; Barik, D.; Verma, K.; and Jena, D. 2011. Honeypot in network security: a survey. In *Proceedings of the 2011 international conference on communication, computing & security*, 600–605. ACM.

Miller, B. A.; Stephens, L. H.; and Bliss, N. T. 2012. Goodness-of-fit statistics for anomaly detection in chung-lu random graphs. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 3265–3268. IEEE.

Navenna, C., and Sasikala, R. 2017. Analyse honey pot traffics to detect dos attacks using support vector machine. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology* 2(6):326–329.

Nezhad, S. M. T.; Nazari, M.; and Gharavol, E. A. 2016. A novel dos and ddos attacks detection algorithm using arima time series model and chaotic system in computer networks. *IEEE Communications Letters* 20(4):700–703.

Noble, C. C., and Cook, D. J. 2003. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 631–636. ACM.

Singh, K. J., and De, T. 2017. Analysis of application layer ddos attack detection parameters using statistical classifiers. *INTERNETWORKING INDONESIA* 9(2):23–31.

Sun, J.; Xie, Y.; Zhang, H.; and Faloutsos, C. 2008. Less is more: Sparse graph mining with compact matrix decomposition. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 1(1):6–22.

Weiler, N. 2002. Honeypots for distributed denial-of-service attacks. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2002. WET ICE 2002. Proceedings. Eleventh IEEE International Workshops on*, 109–114. IEEE.