Visualization of Anomalies using Graph-Based Anomaly Detection

Ramesh Paudel[1], Lauren Tharp[2], Dulce Kaiser[2], William Eberle[2], and Gerald C. Gannod[2]

[1]Dept. of Electrical and Computer Engineering, George Washington University, Washington D.C. 20052
[2]Dept. of Computer Science, Tennessee Technological University, Cookeville, TN 38505

Abstract

Network protocol analyzers such as Wireshark are valuable for analyzing network traffic but pose a challenge in that it can be difficult to determine which behaviors are out of the ordinary due to the volume of data that must be analyzed. Network anomaly detection systems can provide vital insights to security analysts to supplement protocol analyzers, but this feedback can be difficult to interpret due to the complexity of the algorithms used and the lack of context to determine the reasoning for which an event was labeled as anomalous. We present an approach for visualizing anomalies using a graph-based anomaly detection methodology that aims to provide visual context to network traffic. We demonstrate the approach using network traffic flows as an approach for aiding in the investigation and triage of anomalous network events. The simplicity of a visual representation supports fast analysis of anomalous traffic to identify true positives from false positives and prevent further potential damage.

Introduction

Analysis of network traffic can be daunting due to the quantity of data and its high-dimensionality. It can be especially overwhelming when the data is simply represented as lines of text, particularly as streaming data. For example, a tool like Wireshark (Combs 1998), while an indispensable asset for tasks like troubleshooting, is not useful for fast detection of suspicious activity as the analyst must know exactly what to look for. Suspicious patterns may not be evident because relevant information can be separated by hundreds of lines. As such, it is difficult to get a picture of the network as a whole by simply viewing individual packets.

One solution to this problem has come in the form of anomaly detection systems. These tools monitor network activity to determine which traffic is anomalous in reference to normal behavior, significantly narrowing down the activity that is worth investigating. However, one primary drawback of these systems is that their output is sometimes difficult to interpret, as they often give no feedback on *why* something is labeled as anomalous. Specifically, the lack of context can be lost in the conversion of the input data when compared to what is being used by the detection algorithm.

In this work, we present a system for visualizing network traffic alongside a graph-based anomaly detection system (Paudel, Harlan, and Eberle 2019). Detection is performed using the graph-based anomaly detection tool GBAD (Eberle and Holder 2007). We provide a full view of the network integrated with clear labeling of anomalous graph elements based on output of the detection algorithm. We impart context by retaining past events while styling them differently so as to provide a distinction between past and current elements. This improves analysis by allowing tracking of events over time to compensate for the dynamic nature of the network.

The remainder of this paper is organized as follows. The *Background and Related Work* section presents background information and related work in network visualization and anomaly detection. The *Anomaly Detection Approach* section highlights the methodology that we are using to detect anomalies in networks. We then discuss the design of our system in the *Visualization* section. The *Implementation* section provides details of the implementation of the current system, while the *Use cases* section that provides a visual representation of detected anomalies. Finally, we conclude and discuss future investigations.

Related Work

Graph-based approaches have been applied with considerable success for anomaly detection in network traffic. SpotLight (Eswaran et al. 2018), SnapSketch (Paudel and Eberle 2020), and GODIT (Paudel, Muncy, and Eberle 2019) use a sketching-based approach to map an anomalous graph 'far' away from 'normal' instances in the sketch space. MIDAS (Bhatia et al. 2020) uses a hypothesis testing-based framework to detect microcluster anomalies, or suddenly arriving groups of suspiciously similar edges including denial of service attacks in network traffic data. (Paudel, Harlan, and Eberle 2019) uses a graph-based anomaly detection system for detecting the onset of the DoS attack. However, the output of these approaches is difficult to interpret, as they provide no feedback on why something is labeled as anomalous. Therefore, there is a need for a visualization technique that explains and provides context on why something is marked anomalous.

Singh and Subramanian (Singh and and 2009) offer a combined approach by performing simple k-means clustering paired with visualization for anomaly detection. Liao and Striegel (Liao and Striegel 2012) propose a differential anomaly visualization by reducing the granularity of network components to communities which increases network shift tolerance and scalability. Situ (Goodall et al. 2019) is a visualization platform that takes in a stream of network data and performs probability-based anomaly detection before presenting information on suspicious IPs through multiple views.

Anomaly Detection Approach

The idea behind the graph-based anomaly detection (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 1 ***(Anomalous Substructure):*** *We define a graph substructure S' to be anomalous if it is not isomorphic to the graph G'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. GBAD 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, the anomaly detection algorithms in GBAD use 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 S.

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.

Visualization System

We present a dynamic visualization tool for viewing network activity over time while actively labeling anomalies as they are detected by the integrated anomaly detection system. An example graph is provided in Figure 1. (In this example, IP addresses are arbitrarily chosen and do not correlate to a real network representation.) Input data is divided into time windows, which we call generations (labeled G_i in Figure 1), to provide context over time. As seen in Figure 1 in G_2 and G_3, graph elements belonging to previous generations are styled to be transparent in order to make them visually distinct from the current generation. This provides temporal context that allows for comprehension of the network's transformation over time.

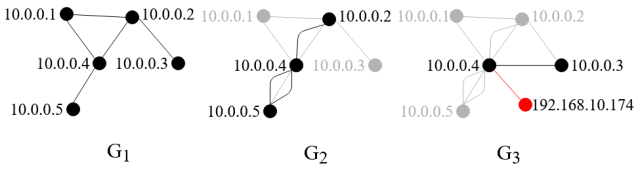


Figure 1: Example of graph design. Devices are represented as nodes labeled by IP address and connections as edges. Previous graph elements are made transparent and anomalous elements are marked red. In G_3, the node labeled 192.168.10.174 and its edge to 10.0.0.4 are anomalous.

Nodes represent devices on the network and are labeled by their respective IP address. Connections between devices are represented by edges and are identified by IP and port number pairs. Multiple edges may exist between two nodes, indicating that multiple connections exist on different ports. This can be seen in Figure 1 in the second generation (G_2) between nodes 10.0.0.4 and 10.0.0.5, as well as nodes 10.0.0.4 and 10.0.0.2.

We clearly mark anomalous nodes boldly by increasing their size and styling them a bright red color. This color was chosen as it is distinctly different from the normal node and edge color (black) and is often associated with alerts. Anomalous edges are also highlighted in red for the same reason. The node 192.168.10.174 in Figure 1 in G_3 is marked as anomalous, as well as its edge to 10.0.0.4. The transparency of previous generations also applies to anomalous elements, so anomalies from previous generations still appear red, but are visually de-emphasized in order to indicate aging.

Anomaly detection, as reported in (Paudel, Harlan, and Eberle 2019), is performed using the GBAD (Eberle and Holder 2007) Graph-Based Anomaly Detection system, a publicly available tool, which uses a graph representation of data to detect anomalies in graph structures, capturing complex relationships, rather than simply analyzing attributes. Common substructures of the graph, and therefore the network, are recognized as normative patterns and differences from these patterns are considered anomalies. That work uses network data from the VAST Challenge 2011 Mini-Challenge 2 (Grinstein et al.) dataset to present GBAD as a solution for detection of the early stages of a denial of service attack. We refer the reader to (Paudel, Harlan, and Eberle 2019) for a more thorough description of the approach.

Implementation

The current implementation of the visualization system is written in Java using the GraphStream library (Dutot et al.

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

Table 1: Confusion matrix for anomaly detection (Paudel, Harlan, and Eberle 2019)

2007). This library provides the basis for reading GML files and displaying the graphs, as well as modifying graph elements according to our design constraints.

Our example data sets are sourced from the VAST Challenge 2011 Mini-Challenge 2 dataset (Grinstein et al.), from the firewall logs on 4/13/2011. To properly render multiple edges, each edge is identified by three nodes and two edges: two nodes representing each end of the connection, one node as an intermediate, and an edge between the intermediate node and each end. These *intermediate* nodes are made transparent in the visualization to create the illusion of a seamless edge.

We identify six styles of elements including *regular* (called "regular"), *normative* (abbreviated "norm"), *anomalous* ("anom"), *removed regular* (abbreviated "rreg"), *removed normative* ("rnorm"), and *removed anomalous* ("ranom"). Regular elements are solid black, normative elements are solid green, anomalous elements are solid red, removed regular elements are transparent black, removed normative elements are transparent green, and removed anomalous elements are transparent red.

Use Cases

The tool is used for a visual demonstration of the onset of a DoS attack in a corporate network. The dataset was provided by the VAST Challenge 2011 Mini-Challenge 2 (Grinstein et al.). While this dataset consists of real, anonymized output from firewall logs, IDS logs, and syslogs, we use only the firewall log data (already converted to CSV format) as this provides all the information necessary for detection and visualization. The dataset contains three full days of traffic; however, we use a portion of the data from day one (4/13/2011) as this is when the attack occurs. A distributed denial of service attack is performed by several attack nodes with IP addresses 10.200.150.<201, 206, 207, 208, and 209> against the external web server (172.20.1.5). We use a time window of 5 seconds per generation.

This demonstration uses the results from Paudel et al. (Paudel, Harlan, and Eberle 2019) to visualize the onset of a distributed denial-of-service attack. The confusion matrix showing the results of anomaly detection in their work is shown in Table 1 (Paudel, Harlan, and Eberle 2019). Normal graphs are the 5-second generation that does not contain a node representing known DoS attack IPs, while the DoS attack graphs are the generations that contain a node representing at least one of the five known DoS attack IPs on the internet. Only 4 generations out of 111 generations associated with DoS attack were missed (Paudel, Harlan, and Eberle 2019). However, the attack was identified by the anomaly detection system within approximately five seconds, demonstrating that a near real-time detection is feasible (Paudel, Harlan, and Eberle 2019).

In Figure 3, we present three consecutive generations of graphs G_{i-1}, G_i, and G_{i+1} where G_{i-1} represents the network snapshot just before the detection of the DoS attack, G_i represents the moment the anomaly detection algorithm detected the DoS attack, and G_{i+1} represent the network snapshot with multiple attack nodes. As seen on Figure 3(a), until G_{i-1}, three machines, 10.200.150.201, 10.200.150.206, and 10.200.150.207, had been sending more than usual traffic (as noticed by the high edge width) to an external web server ((172.20.1.5). As soon as we move to G_i, another machine (10.200.150.208) joins the attack, and the anomaly detection approach flags the edge between the external web server and 10.200.150.201 as anomalous (shown as a red edge in Figure 3(b)). In the subsequent generation G_{i+1}, another attack machine 10.200.150.209 joins the attack and two events are flagged anomalous (Figure 3(c)). The visualization tool provides a clear visual context of the normal flow of traffic in the network and how the attackers are trying to disrupt the normal flow. Furthermore, it also provides the historical context to the anomaly detection that helps analysts understand how the attack is evolving.

Another potential use case involves mapping the lateral movements of an attacker across the system. This is made possible by the persistence of previous generations in the visualization. Since it is possible to see the anomalous nodes from previous generations, a comparison may be made to the current generation to view an attacker's movement over time across the network.

Conclusions and Future Work

Visual analysis supports and expedites analysis of anomalous events to allow for quick identification of false positives versus real threats. Users will likely be familiar with the network they are monitoring and may more easily identify false positives visually. Anomalies may potentially be detected even before detection by the anomaly detection system, such as in the denial of service use case where there is a dramatic increase in edge width. Our tool simplifies the understanding of anomalous events which can facilitate communication of such events to non-technical management. It can be used to display many types of anomalies or attacks, but is most useful for those that can be recognized clearly from a network graph such as with denial of service or rogue devices. It may also be used to visually track lateral movement of an attacker across devices within the network. Our current efforts are focused on using live streamed data as sourced by Wireshark (Combs 1998) and includes use of several capture sources to provide a more global view of network traffic and the anomalies detected within them.

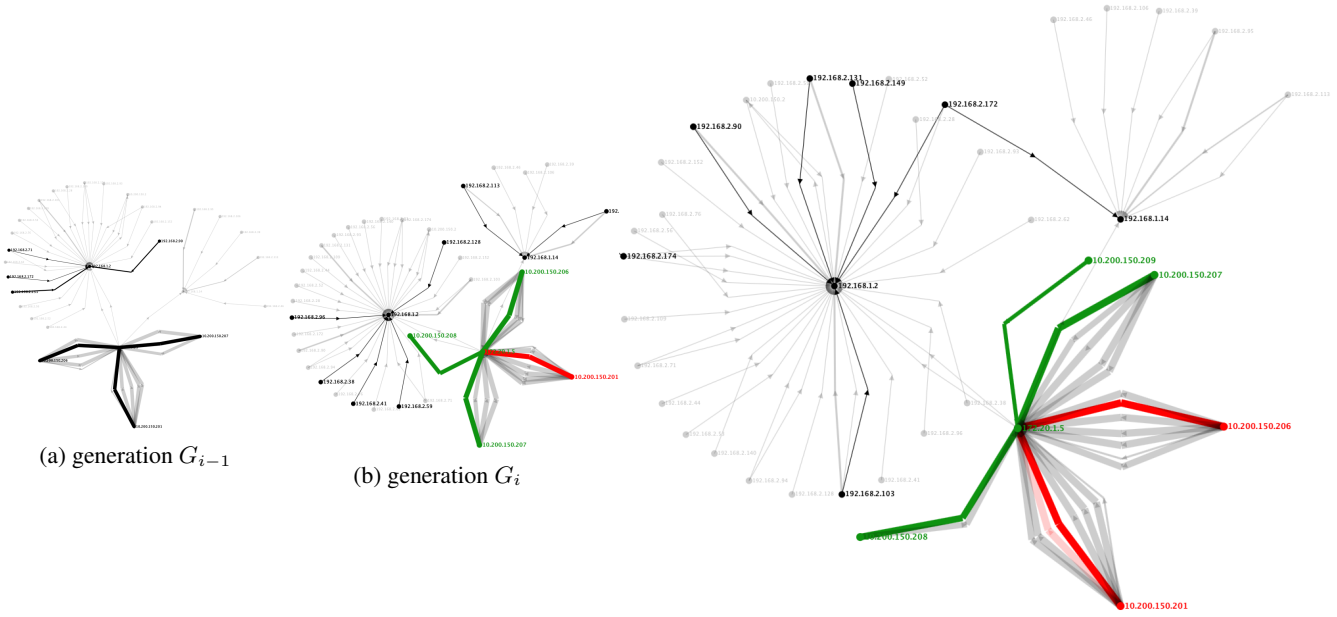
References

Bhatia, S.; Hooi, B.; Yoon, M.; Shin, K.; and Faloutsos, C. 2020. Midas: Microcluster-based detector of anomalies in edge streams. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*. Association for the Advancement of Artificial Intelligence.

Combs, G. 1998. Wireshark. https://www.wireshark.org/. Retrieved November 23, 2020,.

```
Date/time,Syslog priority,Operation,Message code,Protocol,Source IP,Destination IP,Source hostname,Destination hostname,Source port,...
13/Apr/2011 08:52:52,Info,Built,ASA-session-6-302013,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4873,135,epmap,inbound,1,0
13/Apr/2011 08:52:52,Info,Built,ASA-session-6-302013,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4874,43025,43025_tcp,inbound,1,0
13/Apr/2011 08:52:52,Info,Built,ASA-session-6-302013,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4875,43032,43032_tcp,inbound,1,0
13/Apr/2011 08:52:52,Info,Teardown,ASA-session-6-302014,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4875,43032,43032_tcp,inbound,0,1
13/Apr/2011 08:52:52,Info,Built,ASA-session-6-302013,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4876,135,epmap,inbound,1,0
13/Apr/2011 08:52:52,Info,Built,ASA-session-6-302013,TCP,192.168.2.133,192.168.1.6,(empty),(empty),4877,43025,43025_tcp,inbound,1,0
13/Apr/2011 08:52:53,Info,Teardown,ASA-session-6-302014,TCP,192.168.2.133,192.168.1.14,(empty),(empty),4699,49155,49155_tcp,(empty),0,1
13/Apr/2011 08:52:53,Info,Teardown,ASA-session-6-302014,TCP,192.168.2.133,192.168.1.2,(empty),(empty),4700,49158,49158_tcp,(empty),0,1
13/Apr/2011 08:52:53,Info,Teardown,ASA-session-6-302014,TCP,192.168.2.126,192.168.1.2,(empty),(empty),3337,49155,49155_tcp,(empty),0,1
```

Figure 2: An excerpt from a CSV file. Taken from VAST 2011 Mini-Challenge 2 Dataset Firewall Log 1 4/13/2011. (Grinstein et al.)



(a) generation G_{i-1}

(b) generation G_i

(c) generation G_{i+1}

Figure 3: Screenshot of the visualization showing three consecutive generations of the graph during the onset of DoS attack. The anomaly detection algorithm first detects the DoS attack at generation G_i.

Dutot, A.; Guinand, F.; Olivier, D.; and Pigné, Y. 2007. GraphStream: A Tool for bridging the gap between Complex Systems and Dynamic Graphs. In *Emergent Properties in Natural and Artificial Complex Systems. Satellite Conference within the 4th European Conference on Complex Systems (ECCS'2007)*.

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

Eswaran, D.; Faloutsos, C.; Guha, S.; and Mishra, N. 2018. Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1378–1386.

Goodall, J. R.; Ragan, E. D.; Steed, C. A.; Reed, J. W.; Richardson, G. D.; Huffer, K. M. T.; Bridges, R. A.; and Laska, J. A. 2019. Situ: Identifying and explaining suspicious behavior in networks. *IEEE Transactions on Visualization and Computer Graphics* 25(1):204–214.

Grinstein, G.; Cook, K.; Havig, P.; Liggett, K.; Nebesh, B.; Whiting, M.; Whitley, K.; and Konecni, S. VAST Challenge 2011 Mini Challenge 2. https://www.cs.umd.edu/hcil/varepository/benchmarks.php/.

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

Liao, Q., and Striegel, A. 2012. Intelligent network management using graph differential anomaly visualization. In *2012 IEEE Network Operations and Management Symposium*, 1008–1014.

Paudel, R., and Eberle, W. 2020. Snapsketch: Graph representation approach for intrusion detection in a streaming graph. In *Proceedings of the 16th International Workshop on Mining and Learning with Graphs (MLG)*.

Paudel, R.; Harlan, P.; and Eberle, W. 2019. Detecting the onset of a network layer DoS attack with a graph-based approach. In *FLAIRS-32: The 32nd International Conference of the Florida Artificial Intelligence Research Society*.

Paudel, R.; Muncy, T.; and Eberle, W. 2019. Detecting dos attack in smart home iot devices using a graph-based approach. In *2019 IEEE International Conference on Big Data (Big Data)*, 5249–5258. IEEE.

Singh, M. P., and and, N. S. 2009. Visualization of flow data based on clustering technique for identifying network anomalies. In *2009 IEEE Symposium on Industrial Electronics Applications*, volume 2, 973–978.