Application of the PageRank Algorithm to Alarm Graphs
(Extended Abstract)

James J. Treinen[1,2] and Ramakrishna Thurimella[2]

[1] IBM Corporation, Boulder, CO 80301, USA,
jamestr@us.ibm.com
[2] University of Denver, Denver, CO 80208, USA
ramki@cs.du.edu

Abstract. The task of separating genuine attacks from false alarms in large intrusion detection infrastructures is extremely difficult. The number of alarms received in such environments can easily enter into the millions of alerts per day. The overwhelming noise created by these alarms can cause genuine attacks to go unnoticed. As means of highlighting these attacks, we introduce a host ranking technique utilizing *Alarm Graphs*. Rather than enumerate all potential attack paths as in *Attack Graphs*, we build and analyze graphs based on the alarms generated by the intrusion detection sensors installed on a network. Given that the alarms are predominantly false positives, the challenge is to identify, separate, and ideally predict future attacks. In this paper, we propose a novel approach to tackle this problem based on the PageRank algorithm. By elevating the rank of known attackers and victims we are able to observe the effect that these hosts have on the other nodes in the Alarm Graph. Using this information we are able to discover previously overlooked attacks, as well as defend against future intrusions.

Key words: Intrusion Detection, Security Visualization, Watch Lists, Alarm Graphs, PageRank

1 Introduction

Managing the high volume of alarms generated by large intrusion detection environments can be very challenging. A major problem faced by those who deploy current intrusion detection technology is the large number of false alarms generated by Intrusion Detection Sensors (IDSs), which can be well over 90 percent [13, 14].

Since their introduction, Attack Graphs have received considerable attention as a way to model the vulnerabilities of a network. These graphs model the paths that an attacker could take in order to successfully compromise a target. Naïve representations typically result in models that grow exponentially in the number of possible states. Because the resulting graphs are unwieldy even for small networks, recent research has focused on reducing their visual complexity and making them tractable for computational purposes [11, 25].

In this paper, we propose *Alarm Graphs*, an alternative to Attack Graphs, that are built from the alarms produced by the monitoring infrastructure. We establish that several useful insights about intrusions can be gained when these graphs are augmented with knowledge of known attacks, and are analyzed using the PageRank algorithm.
Specifically, our contributions support the following goals:

- **Risk Assessment.** When faced with the task of monitoring large networks, it is easy for human analysts to develop tunnel vision, narrowing their attention to a subset of hosts such as web servers which are commonly known to be involved in attacks. In comparison, our technique allows analysts to algorithmically assess the risk of all nodes and not lose sight of the "big picture" by considering how known attacks affect their neighbors.
- **Systematic Identification of Missed Attacks.** Our technique provides a methodical analysis of the network, and reports the full extent of damage due to an attack. This data is invaluable for forensics and intrusion prevention. When our algorithm was run against historic intrusion data, it identified compromised nodes that were missed by security personnel using manual evaluation techniques.
- **Automated Watch List Generation.** The output generated by our algorithm is a list of those hosts which have higher probability of being involved in future attacks. During our experiments, our algorithm predicted a surprisingly high number of attacks when run against historic intrusion data. For exact numbers, see Section 5.4.
- **Sensor Tuning.** During the course of our analysis, we found that hosts which generated high volumes of false alarms often repeatedly earned a high rank, despite not being involved in a genuine attack. This information provides a means to create filters to remove the false alarms, thus decreasing the overall cost associated with running the monitoring infrastructure, while increasing the overall fidelity of the alarm stream.
- **Visualization.** Alarm Graphs can be visualized using tools such as GraphViz [9]. Because the alarms are reduced to a single link between distinct hosts, as opposed to full enumeration of the alarm log, visualizations produced are compact and easily digestible by a human analyst.

The remainder of this paper is organized as follows. Related work is reviewed in Section 2. An overview of the experimental environment and a discussion of representing alarms as directed graphs is provided in Section 3. The PageRank algorithm is discussed in Section 4. We present our results in Section 5, and provide examples of attacks which were discovered using our technique. Section 6 presents concluding remarks.

2 Related Work

Prior research in the area of analyzing intrusion detection alarms has focused mainly on the classification of alarms as either false or true attacks. Julisch proposes a classification system using cluster analysis to identify the root causes of

alarms in order to remove false positives from the system in [13, 14]. A technique employing machine learning in conjunction with cluster analysis to identify genuine attacks based on previously labeled attacks is described by Pietraszek in [28].

Our research draws inspiration from the field of Attack Graph generation. Attack Graphs are used to model the set of possible actions that could result in a compromised network. As described by Lippmann and Ingols in [17], research on Attack Graphs has focused on three areas. The first is the modeling of network connectivity and known vulnerability findings as a means of enumerating the options available to an attacker to successfully compromise a target host [1, 2, 11, 12, 20–22, 29]. The second is the definition of formal languages used to describe these graphs, as well as the conditions under which state transitions within them are allowed [5, 30]. The third thrust of research has focused on grouping large numbers of intrusion detection alerts by compiling end-to-end attack scenarios or strategies based on Attack Graph analysis as discussed by Ning, et al. in [21–23].

Although various works [25, 29] have discussed methods for the use of probabilistic processes to analyze Attack Graphs, they generally make the assumption that the values which describe the probability of a state transition are predefined. This is addressed by Mehta, et al. in [19], who provide a method for ranking Attack Graphs using link analysis techniques to find the values algorithmically. After the ranking values are computed for an Attack Graph, the nodes with the highest ranks are highlighted as those which have the greatest probability of being involved in an attack. Starting with these marked nodes, an analyst can then focus their attention on the most important portions of the Attack Graph, and use the information contained therein to develop mitigation strategies. It is this concept that we extend in our work by applying a similar analysis technique. Our approach differs from previous work in that rather than use Attack Graphs, we construct an Alarm Graph using the set of intrusion detection alarms triggered for a specified time period. A second key difference between our approach and the previous work is that we augment this graph with data on known attacks, and use link analysis techniques to gain deeper understanding as to how the known attacks influence other nodes in the graph.

3 Preliminaries

3.1 Data Collection

The alarms used in our analysis are generated by a set of intrusion detection sensors (IDSs) representing all major vendors. As such, our technique is technology neutral. The alarms are collected at a central Enterprise Security Manager (ESM) which consolidates them for display in a Security Operations Center (SOC). The ESM has the ability to maintain hot lists of suspicious IP addresses. If an alert is received for an address on this list, the alert is marked for higher priority review by SOC personnel. The ESM performs other automated analysis that is out of the scope of this paper.

Alarms are stored temporarily in a database on the ESM, and are periodically extracted and stored in a permanent data warehouse. The data warehouse was custom built to facilitate off-line analysis. We automatically retrieve the set of alarms used during our analysis via a query to the data warehouse, eliminating any need for manual intervention.

3.2 Modeling Alarms as Directed Graphs

Sensor Type	Source IP	Target IP	Signature	Count
Network	10.1.1.1	10.1.1.3	Share Enumeration	500
Network	10.1.1.1	10.1.1.3	Buffer Overflow	300
Network	10.1.1.2	10.1.1.3	Buffer Overflow	300
Network	10.1.1.3	10.1.1.4	Share Enumeration	100
Host	10.1.1.4	10.1.1.4	Brute Force Login	10

Table 1. Typical intrusion detection alarms

Definition 1 *The set of all intrusion detection alarms A is a set of 5-tuples $a =< t, s, d, g, n >$ which capture the information contained in an IDS alarm.*

Each $a \in A$ is comprised of the sensor type t, either host based or network based; the source IP address of the attack s; the destination, or target IP of the attack d; the alarm signature g which describes the perceived malicious activity; and a count n describing the number of times this combination repeats. This information is stored as a table in a data warehouse, and is easily retrievable.

Definition 2 *An* Alarm Graph *models the set of alarms A as a directed graph $G = (V, E)$. The set of vertices represents the IP space of A, and the set of edges models the set of detected alarms between the various IP addresses.*

Using the set of alarms A, we generate a directed graph $G = (V, E)$. We define S as the set of distinct source IP addresses, and D as the set of distinct destination IP addresses. The set of vertices $V = S \cup D$, such that each $v \in V$ represents an IP address from the set of alarms A. It is important to note that S and D are not disjoint, and in fact $S \cap D$ can make up a large percentage of the overall IP space. A directed edge $e \in E = (s, d)$ is drawn corresponding with the direction of the perceived attack. We deduce the direction of each alarm from the source IP to the destination IP address. The directed graph $G = (V, E)$ is then generated such that each IP address in the alarm set is represented as

a vertex in the graph, and each edge represents the detection of one or more detected alarms between the two vertices. Alarms which are triggered by Host Intrusion Detection Sensors (HIDS), where the sensor resides on the machine being attacked, are denoted as self-loops, as the source IP address is not captured by this type of sensor.

For the purposes of our analysis, the raw alarm data shown in Table 1 is summarized by the adjacency function $f_G : S \times D \rightarrow \{0, 1\}$. We define the adjacency function f_G such that if for any $s \in S, d \in D$ an alarm is triggered by the IDS, a corresponding entry exists $f_G(s, d) = 1$, representing the directed edge $e = s \rightarrow d \in E$. Or,

$$f_G(s,d) = \begin{cases} 1 \text{ if an alarm is triggered from s to d;} \\ 0 \text{ otherwise.} \end{cases}$$

The alarms are summarized such that independent of how many alarms are triggered between distinct pairs of hosts, only one edge is drawn. The rationale behind this approach is that given the high volume of false alarms, the structure that describes the alarm flow is more important than the actual volume. This sentiment echoes Chakrabarti, et al. [4] who note during their analysis of web graphs that the link structure of the web implies underlying social networks. We extend this concept to the social structures implied by the connections present in Alarm Graphs. Understanding this link structure provides an effective means of discovering attacks that would have otherwise gone unnoticed. The results are such that the IDS alarms which are shown in Table 1 are modeled as the directed graph shown in Figure 1.

Fig. 1. Intrusion detection alarms from Table 1 as a directed graph

4 The Ranking Algorithm

We employ Page and Brin's PageRank algorithm [3, 27] to analyze our Alarm Graphs. The page rank algorithm was originally designed to rank the relative importance of a web page among the set of all pages in the World Wide Web. PageRank utilizes the link structure provided via hyperlinks between web pages to gauge this importance. Each hyperlink from a page to a target page is considered a vote, or endorsement of a page's value by the page which links to it. PageRank is computed recursively, and as such, any page that is linked to from

a page that has high rank will itself receive a higher rank due to the fact that an important page has linked to it. A random surfer model is assumed, in which a user selects a random starting point and navigates the web via random clicks to other pages. If a surfer lands on a page with no outbound links, known as a dangling state, they are assumed to start the process again from a new random location. It is also assumed that at any point, a surfer can randomly jump to a new starting point. This random re-entry is captured via a damping factor γ, which is divided by the number of nodes in the graph, and added to all other nodes equally. This model yields Eq. 1.

$$PR(v_i) = \frac{(1-\gamma)}{N} + \gamma \sum_{v_j \in IN(v_i)} \frac{PR(v_j)}{|OUT(v_j)|} \tag{1}$$

The first term of this equation represents the probability of a node being reached via a random entry into the graph, either through a bookmark or the surfer typing a known URL into the browser. The second term is the summation of the probabilities given to a state from all nodes that link into the node. As such, $\{v_1, v_2, v_3 ... v_n\} \in V$ are the vertices in the web graph, $IN(v_i)$ is the set of pages that link in to v_i, $|OUT(v_j)|$ is the number of links out of v_j, and N represents $|V|$ [3, 27].

The output of the PageRank function is given by the vector $PR = (pr_1, pr_2, ...pr_n)$ where pr_i represents the rank of vertex v_i. The values of PR correspond to the entries of the dominant eigenvector of the normalized adjacency matrix of G. This eigenvector is defined as:

$$PR = \begin{pmatrix} pr_1 \\ pr_2 \\ \vdots \\ pr_n \end{pmatrix}$$

where PR is the solution to:

$$PR = \begin{pmatrix} \frac{1-\gamma}{N} \\ \frac{1-\gamma}{N} \\ \vdots \\ \frac{1-\gamma}{N} \end{pmatrix} + \gamma \begin{pmatrix} \alpha(v_1, v_1) & \cdots & \alpha(v_1, v_N) \\ \alpha(v_2, v_1) & \ddots & \vdots \\ \alpha(v_N, v_1) & \cdots & \alpha(v_N, v_N) \end{pmatrix} PR$$

using the adjacency function:

$$\alpha(v_i, v_j) = \begin{cases} \frac{1}{|OUT(v_j)|} & \text{if an edge exists from } v_i \text{ to } v_j; \\ 0 & \text{otherwise.} \end{cases}$$

This algorithm models the probability that a user who is randomly surfing the Internet will land on a given page [3, 19, 27].

4.1 Extending PageRank to Alarm Graphs

We extend the concept of ranking web graphs to ranking Alarm Graphs in the following manner. Each alarm in the alarm set has the potential to represent a genuine attack. For the purposes of our analysis, we think of an attack as a state transition from the node representing the attacker to a successful compromise of the target IP of the alarm. Following this logic, each path in the Alarm Graph represents a potential path of compromise by an attacker through the monitored network.

Using Alarm Graphs, we model the potential paths that an attacker could take through the network, as detected by the intrusion detection sensors, in lieu of the web graph which is proposed in the original PageRank discussion. Using this model, we can then analyze which nodes in the graph have the highest probability of being visited by an attacker, given random entry into the Alarm Graph.

The use of the PageRank algorithm requires that we model the IDS alarms as an ergodic Markov model. Simply put, ergodicity of a Markov model means that every state in the graph is reachable from every other state, given sufficient time. Ergodicity also guarantees that the model will converge to a stable state given sufficient time [8]. The model generated using IDS alarms is not ergodic without some modification. We remedy this in the same manner as is proposed in the original PageRank paper [27], by creating a link from all dangling states to all other nodes in the graph, where a dangling state is defined as a state in the graph from which no outbound links originate. The intuition here is that if an attacker reaches a dangling state, or the end of a potential attack path as detected by the IDS, that they can begin a new attack by jumping randomly to another portion of the graph. The PageRank algorithm captures the effect of this random re-entry into the graph via the damping factor, as described in Equation 1.

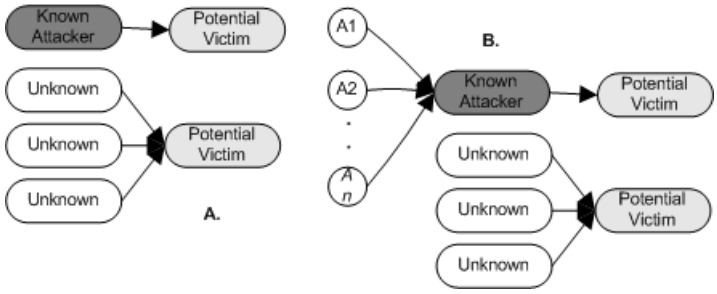

Fig. 2. Ideal coloring of an Alarm Graph

Ideally, when using this approach we would produce rankings in which nodes undergoing genuine attacks receive the highest ranks, and as the level of risk for

a host decreases, so does its corresponding rank. Using these ranks, we would like to produce visualizations that highlight nodes of highest risk as shown in Figure 2a. However, in order to accomplish this consistently, we must incorporate additional information into the graph prior to executing the ranking algorithm.

4.2 Incorporation of Known Attacks

The results of data analysis are known to improve if the analysts (or algorithm) are able to include additional up front knowledge of the data set [7]. The data warehouse that stores our intrusion detection alarms also contains a labeled data set of known attacks that have been identified by the SOC during the course of monitoring the network. We will refer to this data as the set of known security incidents. Prior to ranking the Alarm Graph G, we augment the graph with this data in a manner that improves the quality of the ranking output.

The graph augmentation occurs as follows. In the same manner that a link from one web page to another can be considered a vote or endorsement for the target page, the existence of an edge to a given node in the Alarm Graph can be considered a vote that the targeted node is involved in an attack. Extending this notion, if we know for certain that a given node is involved in an attack, we would like to observe how this fact influences the other vertices in the graph. We accomplish this by annotating the graph with a set of n auxiliary nodes, each of which casts a vote for a single known attacker or victim. The size of n is variable based on the size of the Alarm Graph as a whole. For the purposes of our experiments we uniformly set $n = 50$. Our primary goal is to evaluate the risk that other nodes are extensions of known attacks. Our analysis does not evaluate physical network connectivity, rather we examine the existence of traffic between pairs of hosts that has been perceived as malicious by the IDS. It is important to note that no edges are drawn toward auxiliary nodes, which ensures that no auxiliary vertex will appear as a highly ranked host. We illustrate this technique in Figure 2b.

Given this annotated Alarm Graph we can now calculate the influence of known attackers and victims on the remaining vertices in the graph using the PageRank algorithm. PageRank is computed recursively, and once the model converges, we are able to observe the influence of these high ranking nodes on the network. The results provide us with a realistic representation of those nodes that have the highest risk of being extensions of known attacks.

5 Results

To test the efficacy of our approach, we conducted a series of experiments using intrusion detection data from a production network. The results show that our technique can be used to conduct a more complete analysis of the data produced by the intrusion detection infrastructure. The data consisted of all alarms produced within a 24-hour period. Our experiments were conducted over a 30-day period using data produced by 125 intrusion detection sensors. On average we

observed 1,800 distinct source IP addresses and 1,000 target IP addresses per day. Note that for all examples, the true IP addresses have been obfuscated to protect the confidentiality of the subject network. The total number of alarms received at the SOC averaged 10,000 network IDS (NID) alarms, and 40,000 host IDS (HID) alarms per day. On average, computation of the ranks took between 2 to 5 minutes on a 1 CPU machine with 1Ghz processor and 2 Gbyte RAM, depending on the alarm volume for that day.

5.1 Emergence of Unseen Hosts and Forensic Analysis

During the course of our experiments we discovered that the vast majority of incidents were attributed to a small subset of the overall IP space. This has the adverse effect of causing the analysts to subconsciously focus on this familiar subset of IP addresses, and potentially overlook attacks occurring on other hosts. By using our algorithm, we were able to highlight newly emerging hosts for analysis. As the structure of the underlying Alarm Graph changed over time, new IP addresses moved to the top of the IP ranking automatically. Newly appearing hosts increased in rank and importance if they had a direct connection from an IP address that had been identified as a known attacker or target. This happened as a result of the new host inheriting a portion of the high rank associated with the known attacker or victim. A new host's rank also rose if it was the victim of a coordinated attack wherein it was targeted by multiple attackers. In either of these scenarios, our algorithm consistently marked these hosts as high risk.

5.2 Anomalous Alarm Pattern Recognition

By algorithmically identifying anomalous link patterns in the Alarm Graphs, we are able to highlight sets of alarms which have a higher probability of being genuine attacks. For example, the cluster of alerts shown in Figure 3 is an uncommon structure in the graph and represents the emergence of a Denial of Service (DoS) Attack.

5.3 Identification of Missed Attacks

Figure 4 demonstrates the ability of our algorithm to discover attacks which were missed by the SOC. The darker nodes in the graph are those hosts for which a known incident had occurred. The ranks of these vertices were artificially inflated using the previously described technique. The lighter color nodes represent hosts which inherited these high ranks, and were marked for inspection by our algorithm, but had not been discovered by the SOC. This example shows a brute force dictionary attack against an FTP service running on multiple servers. The SOC detected a portion of this attack, and opened an incident record. However, the analyst only identified half of the victims of the attack. The upper half of Figure 4 illustrates those hosts which were marked as targets, while the lower left portion shows those which were missed. By elevating the

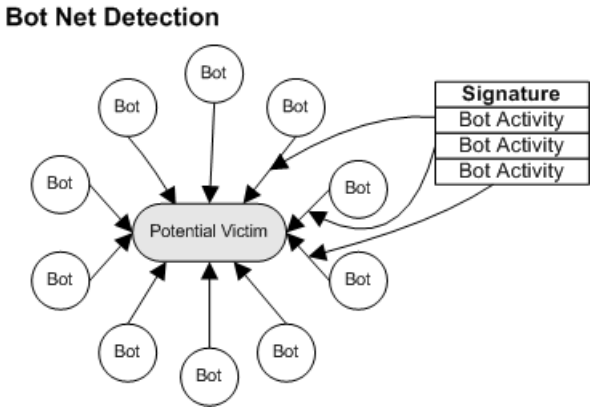

Fig. 3. Probable denial of service attack

rank of the attacking node, our algorithm highlighted the additional three hosts. Upon inspection, these were found to be victims of the same attack. We have included packet capture data from the alarms to further illustrate the attack.

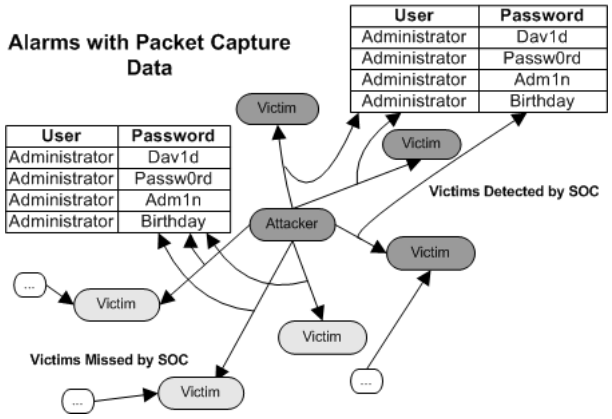

Fig. 4. Detection of partially identified dictionary attack

5.4 Automated Watch List Generation

Watch lists of suspicious IP addresses are maintained by the ESM and are used to monitor the alarm stream for any alerts generated by these hosts. Currently, these watch lists are populated manually. By using the results generated by our

algorithm, it is now possible to build these watch lists automatically. By using the ranked output, we can successfully predict those IP addresses which have the highest probability of being involved in an attack during the subsequent day. Evaluation of our watch lists showed that on average we were able to successfully predict 83% of the security incidents that were manually flagged in a 30-day sample of historic alarm data. This evaluation was conducted using a watch list comprised of the 100 highest ranked IP addresses, or 3% of the roughly 3,000 unique IP addresses that triggered an alarm in the SOC on a given day.

We define successful prediction of an incident as the inclusion of either the source or destination IP address of the alarms comprising that incident on a watch list produced by our algorithm. Using our algorithm, we were able to produce a list of those IP addresses which were suspicious based on the number distinct attackers, or because they were close to hosts which held high rank in the Alarm Graph and inherited a portion of this high ranking based on the recursive calculation of the PageRank algorithm. Figure 5 illustrates the performance of the watch lists generated via the ranking algorithm over a 30-day period. For purposes of completeness, it should be noted that a diminishing return on investment is observed in watch list size. On average, when the size of the watch list was reduced to 50, the success rate fell only by 5%. If the size was increased above 100, the results improved only slightly.

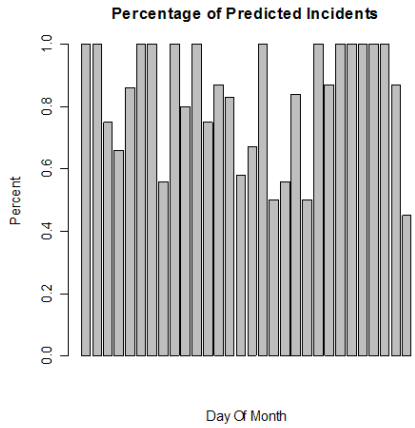

Fig. 5. 30-day trend of incident prediction using a watch list size of 100

5.5 Facilitation of Sensor Tuning

The ranking algorithm sometimes repeatedly identified hosts that received a high rank, but were not involved in genuine attacks. When this behavior was

observed over a period of time, we were able to use the patterns identified by the algorithm to filter the alarms that were causing the fictitious spikes. This type of filtering improves the overall effectiveness of the IDS infrastructure as it reduces the load on the ESM and the analysts, and improves the overall quality of the incoming alarms, resulting in a higher number of genuine attacks being detected.

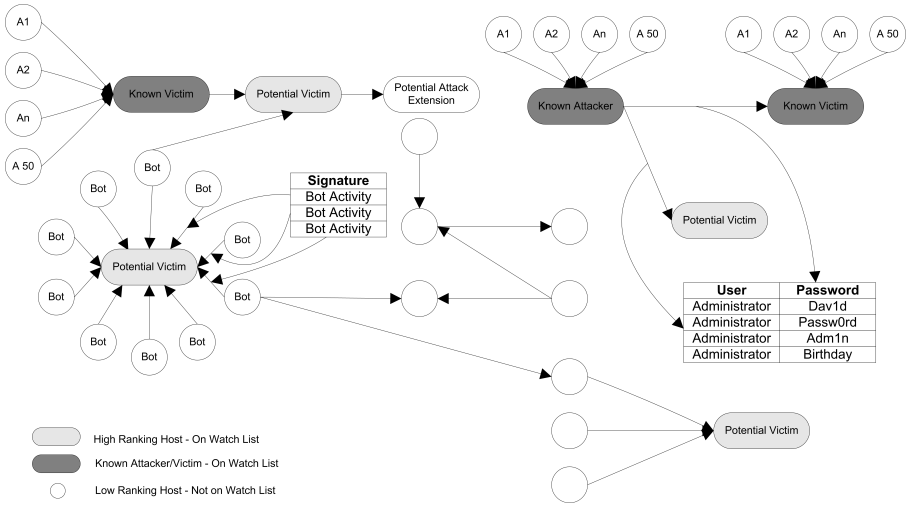

Fig. 6. Colored Alarm Graph from production network, including auxiliary nodes and attack signatures

5.6 Visualization

Figure 6 shows a subgraph of an Alarm Graph generated from production IDS data. The full Alarm Graph is too large to display in a readable manner in print. This figure illustrates two known attacks. The nodes are colored so that the darker the color of the vertex, the higher its rank. The darkest vertices in the graph are those hosts which are known to be involved in attacks, and are shown with the corresponding auxiliary nodes added. Those vertices which are a lighter shade of gray have inherited high rankings, and will appear on the watch list generated at the end of the ranking routine. Additional gray nodes exist in the form of hosts which have received IDS alarms from multiple sources. These atypical patterns are caught by our ranking algorithm, and these hosts will appear on the watch list as well.

The visual representation of the colored Alarm Graphs provides a compact model that can be used by a human analyst to quickly triage the monitored network, providing visual cues as to which systems require immediate attention.

Because the alarms are summarized into a single edge per pair of hosts for which an alarm was raised, the graphs grow slowly as compared to the overall alarm volume, and are easily understood for realistic networks.

5.7 Limitations

Certain type of attacks cannot be detected using our technique. These can be classified into the following categories.

1. **Atomic Attacks.** Attacks which are comprised of a single action are very difficult to detect using our technique. However, rules generally exist in the ESM to automatically detect this type of attack. Once they are labeled in the data warehouse the ranking algorithm will detect any propagation of these attacks to other nodes.
2. **New Hosts.** In this situation, a new IP address appears in the alarm logs that has not been previously observed. Because the host was not previously in the alarm logs, it will not be included in any watch lists. This type of host can be detected using our technique for off-line analysis if one of two conditions is true. First, if the host is a descendant of a node in the Alarm Graph which is known to be involved in an incident it will inherent a portion of the high rank and appear in the watch list. Secondly, the host will be flagged if it is linked to by a sufficient number of distinct attackers.

6 Conclusion

The PageRank algorithm, when applied to annotated Alarm Graphs, is a useful tool for efficiently and methodically analyzing large sets of intrusion detection alarms. Our technique provides an effective means of performing forensic analysis to uncover attacks which were overlooked during real-time monitoring. Additionally, we are able to generate watch lists of IP addresses which are known to have high risk of being involved in an attack. The watch lists are comprised of hosts that are in close proximity to a known attacker or victim, or that are a member of an anomalous structure in the Alarm Graph.

The incorporation of known attacks into our analysis allows us to drastically improve the quality of our results. Prior to annotating the Alarm Graphs with the incident data, the rankings produced were of minimal value, as the distributions reflected the random nature of the underlying graph. However, by including the attack data we are now able to highlight those hosts that deserve a higher rank. By forcing these high ranks, we are able to observe the ripple effect of malicious hosts throughout the network. This provides an effective means of decreasing the likelihood that an attack will be lost in the noise of the false alarms.

The algorithm is being improved in the following ways:

1. Removal of Auxiliary Nodes: The main drawback of the addition of auxiliary nodes is that the size of the graph increases with each incident. By adjusting the probabilities of the incoming edges of a victim, auxiliary nodes will no longer be required.

2. Parallel Edges: Parallel edges will be drawn for distinct alarm signatures and severities which will allow us to assign more weight to nodes which trigger multiple discrete alarm signatures, or for those hosts which trigger high severity alerts.

7 Acknowledgment

The authors would like to thank the anonymous referees for their valuable feedback.

References

1. Ammann, P., Wijesekera, D., Kaushik, S.: Scalable, Graph-Based Network Vulnerability Analysis. In: Proceedings of the 9th ACM Conference On Computer and Communications Security. New York: ACM Press. (2002) pages 217-224.
2. Artz, M.: NETSpa: A Network Security Planning Architecture. Master's Thesis. Massachusetts Institute of Technology. (2002).
3. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: Computer Networks, Volume 30, Numbers 1-7. (1998) pages 107-117.
4. Chakrabarti, S., Dom, B., Gibsony, D., Kleinberg, J., Kumar, R., Raghavan, P., Rajagopalan, S., Tomkins, A.: Mining the Link Structure of the World Wide Web. In: IEEE Computer, Volume 32, Number 8. (1999).
5. Cuppens, F., Ortalo, R.: LAMBDA: A Language to Model a Database for Detection of Attacks. In: Proceedings of the 3rd Annual International Symposium On Recent Advances in Intrusion Detection. Berlin, Germany. (2000).
6. Cuppens, F., Miege, A.: Alert Correlation in a Cooperative Intrusion Detection Framework. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy. Washington, DC. (2002).
7. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: The KDD Process for Extracting Useful Knowledge From Volumes of Data. In: Communications of the ACM. (1996) pages 27-34.
8. Grimmet, G., Stirzaker, D.: Probability and Random Processes. Oxford, Clarendon Press. (1992).
9. GraphViz. http://www.graphviz.org.
10. Honig, A., Howard, A., Eskin, E., Stolfo, S.: Adaptive Model Generation : An Architecture for the Deployment of Data Mining-based Intrusion Detection Systems. In: Applications of Data Mining in Computer Security. Barbara, D., Sushil, J., eds. Boston : Kluwer Academic Publishers. (2002) pages 153-194.
11. Ingols, K., Lippmann, R., Piwowarski, K.: Practical Attack Generation for Network Defense. In: Proceedings of the 22nd Annual Computer Security Applications Conference. Miami Beach, FL. (2006).
12. Jajodia, S., Noel, S., O'Berry, B.: Topological Analysis of Network Attack Vulnerability. In: Managing Cyber Threats: Issues, Approaches and Challenges, V. Kumar, J. Srivastava, and A. Lazarevic, Eds. Dodrecht, Netherlands: Kluwer Academic Publisher. (2003).
13. Julisch, K., Dacier, M.: Mining Intrusion Detection Alarms for Actionable Knowledge. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. (2002) pages 366-375.

14. Julisch, K.: Clustering Intrusion Detection Alarms to Support Root Cause Analysis. In: ACM Transactions on Information and System Security. Volume 6, Number 4. (2003) pages 443-471.
15. Lee, W., Stolfo, S.: Data Mining Approaches for Intrusion Detection. In: Proceedings of the 7th USENIX Security Symposium. (1998) pages 79-94.
16. Lee, W., Stolfo, S., Chan, P., Eskin, E., Fan, W., Miller, M., Hershkop, S., Zhang, J.: Real Time Data Mining-based Intrusion Detection. In: Proceedings of the 2nd DARPA Information Survivability Conference and Exposition (2001).
17. Lippmann, R., Ingols, K.: An Annotated Review of Past papers on Attack Graphs. MIT Lincoln Laboratory Technical Report (ESC-TR-2005-054). (2005).
18. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: The 8th Annual Conference on Information Security and Cryptology. Seoul, Korea. (2005) pages 186-198.
19. Mehta,V., Bartzis, C., Zhu, H., Clarke, E., Wing, J.: Ranking Attack Graphs. In: Proceedings of the 9th Annual International Symposium On Recent Advances in Intrusion Detection. Hamburg, Germany. (2006) pages 127-144.
20. Moore, A., Ellison, R., Linger, R.: Attack Modeling for Information Security and Survivability. In: Software Engineering Institute, Technical Note CMU/SEI-2001-TN-01. (2001).
21. Ning, P., Cui, Y., Reeves, D.: Constructing Attack Scenarios Through Correlation of Intrusion Alerts. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. Washington, DC. (2002).
22. Ning, P., Xu, D.: Learning Attack Strategies From Intrusion Alerts. In: Proceedings of the 10th ACM Conference on Computer and Communications Security, New York, NY. (2003) pages 200-209.
23. Ning, P., Cui, Y., Reeves, D., Xu, D.: Techniques and Tools for Analyzing Intrusion Alerts. In: ACM Transaction on Information and System Security. Volume 7. No. 2 (2004) pages 274-318.
24. Noel, S., Wijesekera, D., Youman, C.: Modern Intrusion Detection, Data Mining, and Degrees of Attack Guilt. In: Applications of Data Mining in Computer Security, Barbara, D., Sushil, J., eds. Boston : Kluwer Academic Publishers. (2002) pages 1-31.
25. Noel, S., Jajodia, S.: Managing Attack Graph Complexity Through Visual Hierarchical Aggregation. In: IEEE Workshop on Visualization for Computer Security. (2004).
26. Noel, S., Robertson, E., Jajodia, S.: Correlating Intrusion Events and Building Attack Scenarios Through Attack Graph Distances. In: Proceedings of the 20th Annual Computer Security Applications Conference. (2004).
27. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. In: http://dbpubs.stanford.edu/pub/1999-66. (1999).
28. Pietraszek, T.: Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection. In: Proceedings of the 7th Annual International Symposium On Recent Advances in Intrusion Detection. Sophia Antipolis, France. (2004) pages 102-124.
29. Sheyner, O., Haines, J., Jha, S., Lippmann, R., and Wing, J.: Automated Generation and Analysis of Attack Graphs. In: IEEE Symposium on Security and Privacy. (2002).
30. Templeton, S., Levitt, K.: A Requires/Provides Model for Computer Attacks. In: Proceedings of New Security Paradigms Workshop. (2000) pages 30-38.