POSTER: Security Logs Graph Analytics for Industry Network System

Qiaoran Meng
National University of Singapore
Singapore
e0492494@u.nus.edu

Nay Oo
NCS Cyber Special Ops-R&D
Singapore
nay.oo@ncs.com.sg

Hoon Wei Lim
NCS Cyber Special Ops-R&D
Singapore
hoonwei.lim@ncs.com.sg

Biplab Sikdar
National University of Singapore
Singapore
bsikdar@nus.edu.sg

Abstract

As Information Technology (IT) infrastructures have become increasingly complex to secure against accelerating cyber threats, current threat detection approaches have been largely silos in nature; security analysts in the environment are typically bombarded with large volume of security alerts that often cause severe fatigues and the possibility of judgement errors. This problem is further exacerbated by the number of false-positives that analysts may waste valuable time and resources pursuing. In this paper, we present how intuitive graph-based machine learning can be used to address the problem of alert fatigue and prioritize risky alerts to assist security analysts. The rationale and workflow of the proposed Graph Analysis (GA) algorithm is discussed in detail, with its effectiveness demonstrated by simulated experiments.

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Networks** → **Network algorithms**; • **Security and privacy** → *Intrusion detection systems.*

Keywords

Security Logs Analytics, Graph Analytics, Network Security

ACM Reference Format:
Qiaoran Meng, Nay Oo, Hoon Wei Lim, and Biplab Sikdar. 2023. POSTER: Security Logs Graph Analytics for Industry Network System. In *ACM ASIA Conference on Computer and Communications Security (ASIA CCS '23), July 10–14, 2023, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3579856.3592830

1 INTRODUCTION

With the size of IT infrastructures rapidly growing and more digital assets inter-connecting in industry network, the impact of cybercrimes has also increased, with a rising expected damage cost of $8 trillion globally in 2023 [1]. Advanced Persistent Threats (APT) are constantly evolving in sophistication, and large enterprises are often targeted by cyber attackers. To combat these threats, threat detection softwares such as Intrusion Detection System (IDS), are widely deployed by enterprises to timely detect cyberattacks against network systems. However, the IDSs generate a large number of alerts that require further manual analysis by security analysts. The redundancy and false positives in numerous security alerts frequently results in severe lagging and weariness in daily operations of Security Operation Centers (SOCs). Hence, approaches to partially automate security alert processing are strongly needed for SOCs to resolve the alert fatigue problem and improve working efficiency. In this paper, we propose a graph-based alert prioritizing method, which is able to handle large amount of alert data, perform effective correlation, and prioritize most severe alert groups for security analysts.

This poster is structured as follows. The relevant existing graphical methods used in cyber defense research are described in Section 2. The proposed GA method for analysing network security logs to tackle the alert fatigue problem is described in Section 3. Section 4 presents our experiments with simulated alert data and Section 5 concludes the poster.

2 RELATED WORK

The use of graphical methods for analyzing security data has been extensively researched [3]. The visualized relationship between network events revealed by attack graphs provide insights into vulnerabilities, increasing the operational decision making efficiency.

CyGraph, a tool that constructs attack graphs to outline attack paths, was introduced by the MITRE corporation. It is intended to alleviate the burden on SOC analysts who must rely on multiple tools and platforms for analysis. CyGraph works by combining and correlating data from various sources to create a unified model. The model is then transformed to a graph which highlights the relationships between network attacks [4]. However, the discovery of network asset clusters related to high-risk activities is not fully automated, user defined query is required for various analysis tasks.

Similarly, researchers have also proposed a model that builds attack graphs to highlight rare severe alerts and the alert paths [2]. The model is capable of condensing hundred-thousands of alerts into less than hundred attack graphs that effectively illustrate how the attacks happened. SOC analysts and red teams are potential users for this tool to primarily obtain insights on attacker's plan

ASIA CCS '23, July 10–14, 2023, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0098-9/23/07.
https://doi.org/10.1145/3579856.3592830

and monitor during training, respectively. However, the metrics for attack graph interpretability are not applied and the threat level of generated attack graphs from the model is not quantified.

While previous works have made valuable contributions to graph-based security data analysis, limitations still exist. Our approach builds upon their strengths and addresses these limitations by incorporating automatic graph analysis using machine learning and security risk computation to derive more comprehensive results.

3 GRAPH-BASED ALERT PRIORITIZATION

The proposed GA algorithm is useful in prioritizing alerts in enterprise network systems. Based on the nature of network data, the input network alerts are transformed into a network communication graph for preliminary analysis and visualization. Considering the high volume and false positive rate of network alerts being processed in SOCs, the algorithm collates network alerts with high similarity to reduce the noise of alert data. Clustering is then performed to group and prioritize alerts according to their threat level and related cyberattack categories. The algorithm outputs subgraphs containing network alerts of similar types, each with a severity score attached. In the following sections, basic concepts and definitions used in the algorithm are discussed, followed by detailed explanation of the algorithm.

3.1 Definitions and Concepts

Network alerts. Network alerts are security alerts triggered by suspicious network communication from a source network host to a destination network host. It is typically raised by Network Traffic Analysis (NTA) softwares or Network-based IDS (NIDS). The NIDSs extract information from network traffic packets and compare them with the attack signature database to flag suspicious network activities as alerts. Network alert monitoring and analysis gives security analysts better visibility of network traffic and effective detection of potential network attacks. The GA algorithm allows alerts from different signature-based NIDS sources as input. The input alert data is denoted as D and each alert is denoted as $d \in D$. The common properties of the IDS alerts are source and destination network hosts, timestamp, and alert-related metrics, including alert messages that describe the detected suspicious network activity, signature id that uniquely identifies a signature, and severity, quantifying the risk level of alert, denoted by $d.src$, $d.dest$, $d.time$, $d.message$, $d.sig\_id$, $d.severity$, respectively in the algorithm.

Network communication graph. Since network alerts are directional network communications between two network hosts, graphs are considered a suitable structure for visualizing and classifying alerts. We model a network communication graph as a directional graph with parallel edges $G = (V, E)$, in which $V = \{v \in V | v = d.src \vee v = d.dest, d \in D\}$ and $E = \{e \in E | e.property = d.property, d \in D\}$ ($property$ can be replaced by the properties mentioned in network alert section). Hence, input alert data is transformed to a graph such that each node represents a network host and each edge represents a network alert between two hosts.

3.2 Graph Analytics Algorithm

Here, we present the GA algorithm with each algorithm component explained in detail. The pseudo code of the algorithm is shown in `Algorithm 1`.

Graph Construction. At the beginning of the algorithm, network alerts from various NIDS sources are fused together and processed by a normalization function to remove redundant or invalid information, and converted into the desired alert format with above mentioned properties. Next, the normalized alerts are transformed into a network communication graph, as described in Section 3.1.

Alert Collation. It's commonly observed in SOCs that security alerts are overloaded with large incoming volume and frequency. There exist a high false positive level and high information overlap among multiple alerts of same source and destination. These irrelevant or repeated alerts can distract SOC operators and lower their efficiency in alert analysis. Our algorithm takes this into consideration and develops the signature-based pairwise alert collation (`line 5` to `line 15`) to reduce alert data size and noises. For each ($source$, $sink$) node pair, the messages, $e.message$, of all alerts between them are first extracted from the graph. These messages are then processed into *n-dimensional numerical vectors* by cybersecurity-specific Natural Language Processing model *word2vec* developed by [5]. These vectors are clustered by the Density-based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, which is robust to outliers and useful for clustering spatial dataset. The *epsilon*, which is the cluster radius, is set to 0.1 to ensure only alerts with high signature similarity are grouped together. Next, each alert group is collated into 1 collated alert, with updated time metrics capturing the intensity and consistency of similar alerts, and updated signature related metrics (`line 12` to `line 14`). Subsequently, the network communication graph G' is reconstructed, with the same network hosts as nodes, but fewer alerts with updated properties between each pair of nodes due to collation.

Subgraph Generation. From the reconstructed network communication graph, we aim to extract subgraphs that contain network alerts related to similar network activities. By this approach, the analysis of a certain suspicious network activity category can be carried out on a small batch of network subgraphs instead of large-volume uncategorized alert data, which improves alert analysis efficiency. To extract subgraphs, DBSCAN algorithm is performed again, but on all collated alerts (edges E' in reconstructed network graph) using alert properties listed on `line 17`. The clustering output are collated alert groups with similar signature metrics, time metrics and severity, which implies alerts related to similar network activities. Finally, each subgraph $G_i = (V_i, E_i)$ is constructed from a cluster of collated alerts (as edges E_i) with the alert endpoints as nodes. To prioritize different types of network activity, each subgraph is assigned a severity score, quantifying the overall threat level of all alerts contained in the subgraph.

4 PRELIMINARY RESULTS

For preliminary experiment and analysis, trial runs using mock network alert data were conducted and the efficiency of the proposed GA algorithm was successfully verified. To simulate the network alert volume, approximately 120,000 IT network alerts were generated by the website logs.io containing DDoS attack, VPN attack and other suspicious network activities. It is observed that the alert dataset has a relatively high false positive rate and repetitions similar to the alerts processed by SOCs. After the alert collation step, the 120,000 input alerts were condensed into 56,982 collated alerts with a reduction rate of 51.7%. The final output of the algorithm

Algorithm 1: Graph Analysis

Input: Alert Dataset D

Initialize Graph $G = (V, E)$, $V \leftarrow \emptyset$, $E \leftarrow \emptyset$, define $P \leftarrow (src, dest, time, message, sig\_id, severity)$; // Graph Construction

for $d \in D$ **do**

  normalize d, $d \leftarrow (d.p | p \in P)$;

  $V \leftarrow V \cup \{d.src, d.dest\}$, $E \leftarrow E \cup \{e\}$, where $e \leftarrow (e.p | e.p \leftarrow d.p, p \in P)$;

Initialize source nodes $X \leftarrow \{x | x \in V \wedge deg^{+}(x) > 0\}$, Sink nodes $Y \leftarrow \{y | y \in V \wedge deg^{-}(y) > 0\}$; // Alert Collation

Initialize graph $G' = (V', E')$, $V' \leftarrow \emptyset$, $E' \leftarrow \emptyset$;

for $(x, y) \in \{(x, y) | x \in X \wedge y \in Y\}$ **do**

  Edges between (x, y): $PE \leftarrow \{e | e \in E \wedge e.src = x \wedge e.dest = y\}$;

  Signature message list $M \leftarrow [word2vec(e.message) | e \in PE]$, Perform DBSCAN Algorithm on M;

  for *cluster i, messages in cluster* M_i *in DBSCAN clusters* **do**

    $PE_i \leftarrow$ edges in cluster i, $T_i \leftarrow [e.time | e \in PE_i]$ (sorted in ascending order), $num\_alerts \leftarrow |T_i|$;

    $min\_time \leftarrow min(T_i)$, $q1 \leftarrow T_i[\frac{|T_i|+1}{4}]$, $q3 \leftarrow T_i[\frac{3(|T_i|+1)}{4}]$, $dispersion \leftarrow \frac{q3-q1}{q1+q3}$, $span \leftarrow q3 - min\_time$;

    initialize collated edge e_i, $e_i.src \leftarrow x$, $e_i.dest \leftarrow y$, $e_i.time\_metrics \leftarrow (dispersion, span, min\_time, num\_alerts)$;

    $e_i.message \leftarrow set(M_i)$, $e_i.sig\_id \leftarrow \{e.sig\_id | e \in PE_i\}$, $e_i.severity \leftarrow mean([e.severity | e \in PE_i])$;

    $V' \leftarrow V' \cup \{e_i.src, e_i.dest\}$, $E' \leftarrow E' \cup \{e_i\}$;

Initialize Subgraph list $Subgraphs \leftarrow []$; // Subgraph Generation

Perform DBSCAN Algorithm on Collated Edge Data $ED \leftarrow [(e.time\_metrics, e.message, e.sig\_id, e.severity) | e \in E']$;

for *cluster i, data in cluster* ED_i *in DBSCAN clusters* **do**

  $E'_i \leftarrow$ alerts/edges in cluster i, initialize i^{th} subgraph $G_i = (V_i, E_i)$, $V_i \leftarrow \emptyset$, $E_i \leftarrow \emptyset$, $G_i.severity = mean([e.severity | e \in E'_i])$;

  for $e \in E'_i$ **do**

    $V_i \leftarrow V_i \cup \{e.src, e.dest\}$, $E_i \leftarrow E_i \cup \{e\}$;

  Add G_i to $Subgraphs$

Output: $Subgraphs$

Table 1: Top 3 risky subgraphs generated by GA algorithm

subgraph index	summarized alert message	severity	related attack	number of alerts
5	Weird inbound traffic, large DNS connection	7.5	DNS flood, DDoS attack	6
9	A real IP packets is rejected by ACL	5.2	excessive firewall denies (potential credential access attempt)	15
2	VPN-related session is started/ completed	5	VPN attack	2

were 175 subgraphs containing network alerts of comparable types, which simplifies the operations of security analysts since the scale of alert analysis can be reduced to a hundred alert batches.

For a closer look, a demo trail run of 100 network alerts randomly sampled from the above alert dataset was conducted. 12 subgraphs were generated by the GA algorithm, and the subgraphs with highest severity score are shown in Table 1. The summarized alert messages are the most frequent alert messages in alerts of each subgraph. These messages successfully identified various kinds of potential network attacks including DDoS and VPN attack, which aligned with the attack scenarios of our data.

5 CONCLUSION

With evolving digital world and growing IT systems, the network alert fatigue problem has become more severe for SOC analysts, resulting in potential risks of ignored cyberattacks. To reduce alert fatigue and automate alert prioritization, a GA algorithm is proposed in this poster, with the ability to analyse network alerts from different NIDS sources. The alerts are transformed to a network communication graph which depicts the nature of network communication and allows easy visualization. The algorithm reduces the noise of alerts by collation technique and prioritizes alert groups via a clustering algorithm. The output of the GA algorithm is subgraphs containing alerts related to similar network activities and corresponding graph severity scores. These subgraphs can effectively assist security analyst to identify high-risk network activities and their related alert groups. The simulated experiments conducted have shown high alert noise reduction and effective alert cluster prioritization of the proposed GA algorithm.

References

[1] eSentire. 2022. 2022 Cybercrime Report. https://www.esentire.com/resources/library/2022-official-cybercrime-report

[2] Azqa Nadeem, Sicco Verwer, Stephen Moskal, and Shanchieh Jay Yang. 2022. Alert-Driven Attack Graph Generation Using S-PDFA. *IEEE Transactions on Dependable and Secure Computing* 19, 2 (2022), 731–746. https://doi.org/10.1109/TDSC.2021.3117348

[3] Steven Noel. 2018. *A Review of Graph Approaches to Network Security Analytics.* Springer International Publishing, Cham, 300–323. https://doi.org/10.1007/978-3-030-04834-1_16

[4] S. Noel, E. Harley, K.H. Tam, M. Limiero, and M. Share. 2016. Chapter 4 - CyGraph: Graph-Based Analytics and Visualization for Cybersecurity. In *Cognitive Computing: Theory and Applications*, Venkat N. Gudivada, Vijay V. Raghavan, Venu Govindaraju, and C.R. Rao (Eds.). Handbook of Statistics, Vol. 35. Elsevier, 117–167. https://doi.org/10.1016/bs.host.2016.07.001

[5] Ankur Padia, Arpita Roy, Taneeya Satyapanich, Francis Ferraro, Shimei Pan, Youngja Park, Anupam Joshi, and Tim Finin. 2018. UMBC at SemEval-2018 Task 8: Understanding Text about Malware. In *Proceedings of the 12th International Workshop on Semantic Evaluation.* Association for Computational Linguistics, New Orleans, Louisiana, 878–884. https://doi.org/10.18653/v1/S18-1142