April 2022

Suffix-based Finite Automata for Learning Explainable Attacker Strategies

Azqa Nadeem [a,1], Sicco Verwer [a] and Shanchieh Jay Yang [b]
[a] *Delft University of Technology, The Netherlands*
[b] *Rochester Institute of Technology, United States*

Abstract

Learning about attacker behavior, such as their tactics, techniques and procedures (TTPs) is largely a manual and expert knowledge-driven task in defensive cybersecurity. An attack graph is a graphical representation of attacker strategies that shows all the pathways an attacker can use to penetrate a network. Existing techniques correlate system vulnerabilities and expert input regarding network topology to construct attack graphs, thus providing a static and hypothetical view of the threat landscape. These traditional attack graphs cannot directly be used to monitor ongoing attacks in Security Operations Centers (SOCs) since they do not show the dynamic strategies being employed by the attackers. Meanwhile, SOC analysts defend against cyber attacks by monitoring millions of intrusion alerts on a daily basis[2], often leading to 'alert fatigue' and reduced productivity.

In the accepted paper [1], we propose a novel paradigm of attack graphs, known as '*alert-driven attack graphs*', that are learned directly from intrusion alerts without any expert input. Thus, instead of investigating large volumes of alerts, SOC analysts can visualize a few alert-driven attack graphs to understand the corresponding attacker strategies. Utilizing machine learning to extract alert-driven attack graphs, the following constraints must be addressed: 1) *Alert-type imbalance:* Severe alerts are infrequent, while non-severe alerts (e.g., related to network scans) are common. Frequency-analysis methods discard infrequent events as noise, making most machine learning methods unsuitable for this application. 2) *Modeling context:* The same alert signature may be involved in different attacker strategies. This is indicated by the neighboring alerts, which can be used to model an alert's context so as to distinguish between similar attacker strategies. 3) *Interpretable model:* SOC analysts are often contractually obligated to investigate all alerts, making black-box models inherently unsuitable since they do not let them reverse engineer the alerts behind a classifier decision.

The accepted paper proposes SAGE[3] — an interpretable sequence learning pipeline that constructs attack graphs from the actions observed through intrusion alerts, without

[1] Corresponding Author: Azqa Nadeem, Department of Intelligent Systems, Delft University of Technology, 2628 XE Delft, Netherlands; E-mail: azqa.nadeem@tudelft.nl.

[2] https://www.imperva.com/blog/27-percent-of-it-professionals-receive-more-than-1-million-security-alerts-daily/

[3] SAGE is open-source: https://github.com/tudelft-cda-lab/SAGE

a priori expert knowledge. SAGE models the temporal and probabilistic relationships between alerts in a suffix-based probabilistic deterministic finite automaton (S-PDFA) using the FlexFringe [2] automaton learning framework. Individual attack graphs are then extracted from the S-PDFA for each objective obtained on the victim host(s). The S-PDFA at the heart of SAGE addresses the aforementioned constraints. i) *Alert-type imbalance:* A suffix-based model is specifically chosen to highlight infrequent alerts, without discarding any low-severity alerts. In our implementation, the severe alerts always appear at the end of the sequences, making a suffix-based model a natural choice. ii) *Modeling context:* The S-PDFA state identifiers capture the alert context. Using the Alergia heuristic [3] for state merging, states having similar futures and pasts are merged, while those leading to significantly different outcomes are not. iii) *Interpretable model:* The Markovian properties of the S-PDFA, together with Sink states[4] make the model components interpretable. The Markovian property ensures that the input transition symbols of a state are unique, making it easier to interpret the meaning of a state. Here, the states correspond to the milestones achieved by an attacker. Moreover, the deterministic nature of the S-PDFA makes it algorithmically transparent. The parameters of the model are selected through trial-and-error of visualizing the S-PDFA until it matches our intuition about the data, making it design transparent. The attack graphs extracted from the S-PDFA are also transparent, interpretable, and scientifically explainable.

Tested with intrusion alerts collected through three different security testing competitions, we evaluate SAGE's efficacy on distributed, multi-stage attack scenarios. The competitions host several multi-member teams that exploit a common fictitious network. For each competition, a minimal set of network-agnostic features is derived from the resulting intrusion alerts, which is given as input to SAGE. SAGE uses constant parameters for all datasets, as outlined in [1]. Collectively for the three datasets, SAGE compresses over 1425k alerts into 401 attack graphs that show how specific attacks transpired. The attack graphs capture the strategies used by the participating teams, producing directly relevant insights for SOC analysts, e.g., they reveal that attackers follow shorter paths to re-exploit objectives in 84.5% of the cases. This finding is backed by common-sense intuition that if an attacker already knows how to exploit an objective, they would skip the unnecessary hit-and-trial steps for re-exploitation. The attack graphs also provide an intuitive layout to compare attacker strategies, which allow to reason about easily exploitable objectives (depicted by shared strategies), and fingerprintable paths (depicted by strategies only employed by a single attacker).

As follow-up, we are investigating how to use alert-driven attack graphs for proactive defense. Specifically, how the S-PDFA can be used to predict next attack steps.

References

[1] Nadeem A, Verwer S, Moskal S, Yang SJ. Alert-driven Attack Graph Generation using S-PDFA. IEEE Transactions on Dependable and Secure Computing. 2022;19(2):731-46.

[2] Verwer S, Hammerschmidt CA. Flexfringe: a passive automaton learning package. In: 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME). IEEE; 2017. p. 638-42.

[3] Carrasco RC, Oncina J. Learning stochastic regular grammars by means of a state merging method. In: International Colloquium on Grammatical Inference. Springer; 1994. p. 139-52.

[4] Sinks are states that occur too infrequently to learn from. We remove low-severity sinks from the model.