Contribution Title

Pedro Pereira[1211131], Rafael Branco[1200799], Ricardo Gonçalves[1240613], Ricardo Lemos[1222708], and Svetlana Zamyatina[1240156]

School of Engineering, Polytechnic of Porto (ISEP/IPP), 4249-015 Porto, Portugal
{1211131,1200799,1240613,1222708,1240156}@isep.ipp.pt

Abstract. In this paper, we introduce a rule based expert system developed to automate the classification and response to cybersecurity alerts, addressing the challenge of managing large volumes of security incidents. Brute force attacks, unauthorized access attempts and other security breaches are common in modern organizations and a quick and reliable response is required. Our approach leverages expert-defined rules to handle alerts efficiently with minimal human intervention, reducing response time.

A Prolog based backend is used to process alerts according to a structured set of rules given by an experienced Security Operations Center analyst. This knowledge based structure gives us a way to classify alerts correctly, and with clear guidelines for handling different types of security incidents (failed logins, suspicious users, etc). Users are given an interactive interface for data entry and alert review by integration with a React based frontend.

Evaluation of the system demonstrates promising results in reducing false positives and increasing accuracy in identifying and responding to genuine threats. The rule based expert system presented in this paper provides an adaptable and reliable tool for cybersecurity that can improve response efficiency without the use of complex machine learning models.

Keywords: Cybersecurity · Expert System · Security Operations Center.

1 Introduction

Modern organizations receive a large number of security alerts due to attempts such as brute-force attacks and unauthorized access [15, 16], which make it mandatory to develop effective methods for automated classification and response [9]. Existing solutions are rather based on human discretion, which causes certain delays and raises the likelihood of overlooked or misidentified potential threats.

This paper proposes an automated expert system designed to handle such alerts intelligently. The system help users to evaluate and classify the alerts, aiming to reduce the response delay, increasing the company efficiency and helping juniors users solving the most difficult case.

The objectives of this paper are:

- To present an automated system capable of handling alert classification and resolution.
- To evaluate the system's performance in minimizing response time and reducing false positives.
- To compare the proposed system with existing alert-handling mechanisms in cybersecurity.

2 Related Work

In recent years, various approaches to security alert classification and resolution have been developed, each offering unique benefits and limitations in managing security alerts effectively. Automated alert systems, machine learning (ML) algorithms, and expert rule based systems are key methods in this domain and each has a distinct role in the cybersecurity domain facing distinct challenges. In this section, we review existing methods for security alert classification and resolution, focusing on automated systems, ML applications, and rule-based expert systems in cybersecurity.

Automated alert systems help to reduce human involvement in security incident management by automatically responding to some events (unauthorized access attempts, network intrusions, and anomalous activity detection) using predefined rules [14]. However, traditional automated alert systems are based on static rule based frameworks which are prone to false positives and are not easily adaptable to changes in the threat and user behavior [10, 17]. We have seen in studies that while these systems are efficient in structured scenarios, they can generate large amounts of noise by flagging legitimate actions as suspicious, resulting in security analysts being overwhelmed by unnecessary alerts [2]. One such example is when an unusual but benign increase in login attempts generates alerts when the system doesn't account for context specific factors. This limitation also suggests that automated systems should be integrated to receive and validate user feedback and should continue to be validated to increase alert accuracy and reduce response fatigue. That need has spurred the recent development of adaptive systems capable of re-calibrating rules in response to real time feedback, which our proposed system seeks to meet [10, 6, 17].

ML algorithms have been applied for alert classification in cybersecurity since the effectiveness of such tools is proven [1, 8, 11, 12, 19]. This approach suggests that the use of supervised models whose parameters are derived from historical alert data is useful to decide the likelihood of an alert being genuine or a false positive [8]. Other activities that show abnormal log-in patterns are other methods like clustering and outlier detection [1, 11, 19]. Hence these models are purely statistical and need huge data sets and frequent upgrading of the model for efficiency. Moreover, they still depend on manual intervention for resolution after detection.

For example, Isolation Forest, Histogram-Based Outlier Score (HBOS) on CAMLPAD platform are used to detect network anomalies and flag potential threats with high accuracy (95% adjusted rand score) using real time streaming

data [11]. The integration of ML in cybersecurity techniques has been broadly studied by researchers who claim that ML can greatly increase detection capabilities when used in conjunction with adaptive model updating to cope with new attack vectors [13, 3]. While such advancements exist, the training of ML models can still be a barrier because of the need for extensive labeled data, especially in fast changing threat landscapes where new types of alerts emerge frequently [5, 7]. Additionally, continuous retraining and recalibration of the model's relevance is required, which makes the model's operational complexity and potentially an accuracy trade off. In addition, ML models are generally not transparent and analysts cannot understand what ML models did, which reduces their trustworthiness in highly critical security contexts [4].

Expert systems employ knowledge-based approaches, to guide response to security scenarios and provide consistency and transparency in decision making [20]. Expert systems are more deterministic than ML models, and they can produce clear results which makes them best suited for the scenarios where there is a need for a clear action [21]. For instance, account lockout protocols triggered automatically by an expert system after multiple failed login attempts is a straightforward and easy to explain rule. Experts systems are capable of instant response without the need of large datasets for training, which makes them a good addition to cybersecurity frameworks, especially for small and medium sized enterprises that may not have the resources for ML model development. Nevertheless, while expert systems provide explainability, they are limited by static rules, and must be manually updated by cybersecurity professionals in order to adapt to new threats. In addition, rule based systems are inherently limited by the knowledge that is embedded in their decision trees, and are less effective in complex, dynamic threat environments than adaptive ML based approaches.

3 Implementation

To explain how the expert system was built, the architectural model adopted will be described in this section. We will elaborate the architecture of the system and provide information about the technologies applied during the creation of the system, choices made and issues encountered. Besides, we will describe the methodologies of the sub-systems that make up the system we are designing like the front-end and back-end sub-systems.

3.1 Methodology

The development of our expert system was based on the Knowledge Acquisition and Documentation Structuring (KADS) methodology. The KADS methodology, which is widely known for building knowledge based systems, emphasizes systematic acquisition, modeling and structuring of expert knowledge to guarantee accuracy and completeness [22, 18]. With this methodology, we were able to carefully define the knowledge requirements and decision making rules that are necessary for effective cybersecurity alert management.

3.2 Knowledge acquired

The success of our expert system relies heavily on the quantity and quality of the knowledge provided. In this section, the various source of knowledge that was employed in developing this work are described. As for the specialist for our project, our group chose Paulo Valdeira – an experienced SOC analyst from Redshift Global. Additionally, we gained access to company documents outlining the necessary processes when responding to specific types of alerts. Paulo Valdeira is currently a SOC Manager at Redshift Global. He has been actively involved in the IT industry for 21 years, of which 10 years was conducted in SOC. His experience includes five years in Tier 1 SOC, three years in Tier 2/3, and the last two years as SOC Manager. Valdeira's vast expertise was instrumental in guiding the project, with regular input during our weekly meetings. Company documents are official papers that the company prepares to regulate employee conduct and avoid confusion or scanty practices that may compromise organizational-functioning, especially where matters of security are involved. These are written complied texts that contain elaborate procedures of tackling certain functions. For Redshift Global, they offer clear steps for SOC analysts on how to respond to security alerts. As a part of each use case, diagrams and the action points that were in image format are encompassed for each type of alert. These were crucial for mimicking knowledge, and it was through these and discussion with the expert that the group mapped the various use cases for the system.

3.3 Architecture Overview

The implemented expert system adopts a client-server architecture, where the frontend is developed in React and the backend leverages Prolog to handle the inference engine. The frontend includes interfaces for data entry and for reviewing the conclusions, whereas the Prolog component actualizes specific entry data against certain inputs while making conclusions itself. The architecture is composed of the following key components:

- Frontend (React): A dynamic user interface for data input and result visualization. It communicates with the backend through a RESTful API.
- Backend (Prolog): A logic-based inference engine responsible for processing inputs and applying Prolog rules to generate conclusions. The Prolog backend is accessed through HTTP requests from the frontend.

The choice of React for the frontend was based on its high popularity, robustness, ability to create dynamic and reactive interfaces with ease and because the team already had prior knowledge of this technology. Integration with the backend via REST APIs is facilitated by the simplicity of HTTP request libraries such as Axios. The Prolog backend was a requirement of the PPROGIA course. Other tools such as Git were used for version control.

3.4 Frontend implementation (React)

The frontend was created using React, which is basically a JavaScript library used for creating a user interface (app/web) primarily driven by user interaction. The user interface was designed with simplicity in mind, providing input forms by which the user enters data for processing. In the provided interface, the results are displayed after the submission process is completed by the backend. In the frontend of the application the business logic controls the flow of the requests to the backend and presents the results to the user. Once the users enter the data, it is structured and forwarded to the REST API of the backend where the rules for inference are performed. After the response is received the components then update in accordance to the received result. On the communication between the frontend and the backend, REST APIs are used. The Axios library (or the Fetch API) are employed to make HTTP (POST, GET) requests as the requests are initiated asynchronously.

3.5 Backend implementation (Prolog)

The backend was implemented Prolog, a logic programming language particularly well-suited for knowledge-based systems and rule-based inference. It deals with the most important functioning of the expert system and is composed of several elements.

First, the inference engine is implemented using Prolog to define the expert system's rules as logical clauses. When the backend receives new data from the frontend, it processes this data by querying the Prolog engine, applying the defined rules to infer the appropriate results.

In order to establish connection between the created React frontend and the Prolog engine, a basic RESTful API was developed. This API was created using a Prolog framework named SWI-Prolog HTTP library, it receives data through post requests from the front-end and after processing through Prolog returns back the inference results.

In Prolog the knowledge base is essentially a collection of facts and rules that define knowledge area of the expert system. For example, the mentioned system can prescribe specific decisions based on some conditions or evidences provided by the user, using the flag made in the knowledge base in Prolog for data analysis to come up with a conclusion.

4 Results and discussion

After implementing and testing the system internally, WE presented the project to Redshift, a security firm specializing in SOC management.

Redshift representatives were very interested in the system's potential as a training tool for new analysts. They highlighted that the system's rule based approach is a good match to real world SOC process, and that it is a good place to introduce junior analysts to standard alert handling protocols. The rule sets

used in the system are structured and allow trainees to observe and understand the logic behind the decision making they would need to make in live scenarios.

From a performance standpoint, our tests demonstrated that the system was able to cut response times, with alerts being almost instantaneous based on set rules. Faster incident response is key for SOC operations, and this feature helps to achieve that. The project was a success and the feedback from Redshift is promising for deployment in SOC environments as an operational tool and as a training resource.

5 Conclusion

In this paper, a rule based expert system was presented for improving the efficiency and accuracy of cybersecurity alert management. The system securely, automatically classifies and responds to a wide range of security incidents, relieving human analysts of this burden and decreasing response times, by leveraging expert defined rules in a Prolog based backend.

ollowing a successful demonstration to Redshift, the system became not only an effective tool for incident management but also a great training resource for training new analysts. The structure of the rule based logic is very similar to real world SOC procedures, and makes it perfect for training students on how to treat alerts precisely and consistently. The fact that Redshift is interested in adopting this system for training only reiterates the practical value of a knowledge based approach in cybersecurity.

In future work the system could be expanded with more rule sets to cover a broader range of alerts or adapted to meet the needs of another SOC. Furthermore, machine learning could be integrated into the system to make it more adaptable by making the system learn from new types of alerts to increase accuracy over time. For example, clustering techniques or semi supervised learning can be used to discover emerging threat patterns and reduce reliance on manual rule updates. Such an approach would be to use the reliability of rule based methods and the adaptability of machine learning to create a more robust solution to dynamic cybersecurity threats.

References

1. Aamir, M., Zaidi, M.: Clustering based semi-supervised machine learning for ddos attack classification. J. King Saud Univ. Comput. Inf. Sci. **33**, 436–446 (2019). https://doi.org/10.1016/J.JKSUCI.2019.02.003
2. Alturkistani, H., El-Affendi, M.A.: Optimizing cybersecurity incident response decisions using deep reinforcement learning. International Journal of Electrical and Computer Engineering **12**(6), 6768 (2022)
3. Banik, S.: Future directions for ml in cybersecurity. Revista de Inteligencia Artificial en Medicina **14**(1), 401–430 (2023)
4. Booth, J., Metz, D.W., Tarkhanyan, D.A., Cheruvu, S.: Machine Learning Security and Trustworthiness, pp. 137–222. Apress, Berkeley, CA (2023).

https://doi.org/10.1007/978-1-4842-8297-7_5, https://doi.org/10.1007/978-1-4842-8297-7$_5$

5. Braun, T., Pekaric, I., Apruzzese, G.: Understanding the process of data labeling in cybersecurity. In: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing. p. 1596–1605. SAC '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3605098.3636046, https://doi.org/10.1145/3605098.3636046
6. Chen, Y.J., Chu, H.C., Chen, Y.M., Chao, C.Y.: Adapting domain ontology for personalized knowledge search and recommendation. Information Management **50**(6), 285–303 (2013). https://doi.org/https://doi.org/10.1016/j.im.2013.05.001, https://www.sciencedirect.com/science/article/pii/S0378720613000414
7. Dasgupta, D., Akhtar, Z., Sen, S.: Machine learning in cybersecurity: a comprehensive survey. The Journal of Defense Modeling and Simulation **19**(1), 57–106 (2022). https://doi.org/10.1177/1548512920951275
8. D'hooge, L., Wauters, T., Volckaert, B., De Turck, F.: Inter-dataset generalization strength of supervised machine learning methods for intrusion detection. Journal of Information Security and Applications **54**, 102564 (2020). https://doi.org/https://doi.org/10.1016/j.jisa.2020.102564, https://www.sciencedirect.com/science/article/pii/S2214212619310415
9. Gelman, B.U., Taoufiq, S., Vörös, T., Berlin, K.: That escalated quickly: An ml framework for alert prioritization. ArXiv **abs/2302.06648** (2023). https://doi.org/10.48550/arXiv.2302.06648
10. González-Granadillo, G., González-Zarzosa, S., Diaz, R.: Security information and event management (siem): Analysis, trends, and usage in critical infrastructures. Sensors **21**(14) (2021). https://doi.org/10.3390/s21144759, https://www.mdpi.com/1424-8220/21/14/4759
11. Hariharan, A., Gupta, A., Pal, T.: Camlpad: Cybersecurity autonomous machine learning platform for anomaly detection. ArXiv **abs/1907.10442** (2019). https://doi.org/10.1007/978-3-030-39442-4_52
12. Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., He, X.: Generative adversarial active learning for unsupervised outlier detection. IEEE Transactions on Knowledge and Data Engineering **32**, 1517–1528 (2018). https://doi.org/10.1109/TKDE.2019.2905606
13. Maddireddy, B.R., Maddireddy, B.R.: Evolutionary algorithms in ai-driven cybersecurity solutions for adaptive threat mitigation. International Journal of Advanced Engineering Technologies and Innovations **1**(2), 17–43 (Aug 2021), https://ijaeti.com/index.php/Journal/article/view/319
14. Maddireddy, B.R., Maddireddy, B.R.: Enhancing network security through ai-powered automated incident response systems. International Journal of Advanced Engineering Technologies and Innovations **1**(02), 282–304 (May 2023), https://ijaeti.com/index.php/Journal/article/view/316
15. Pawar, M.V., Anuradha, J.: Network security and types of attacks in network. Procedia Computer Science **48**, 503–506 (2015)
16. Pietraszek, T.: Classification of intrusion detection alerts using abstaining classifiers. Intell. Data Anal. **11**, 293–316 (2007). https://doi.org/10.3233/IDA-2007-11306
17. Pietraszek, T.: Using adaptive alert classification to reduce false positives in intrusion detection. In: Jonsson, E., Valdes, A., Almgren, M. (eds.) Recent Advances in Intrusion Detection. pp. 102–124. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
18. Schreiber, G., Wielinga, B.J., Breuker, J.: Kads : a principled approach to knowledge-based system development (1993), https://api.semanticscholar.org/CorpusID:57045801

19. Shao, M., Gu, N.: Anomaly detection algorithm based on semi-supervised collaborative strategy. Journal of Physics: Conference Series **1944** (2021). https://doi.org/10.1088/1742-6596/1944/1/012017
20. Shu-Hsien Liao: Expert system methodologies and applications—a decade review from 1995 to 2004. Expert Systems with Applications **28**(1), 93–103 (2005). https://doi.org/https://doi.org/10.1016/j.eswa.2004.08.003, https://www.sciencedirect.com/science/article/pii/S0957417404000934
21. Straub, J.: Machine learning performance validation and training using a 'perfect' expert system. MethodsX **8**, 101477 (2021). https://doi.org/https://doi.org/10.1016/j.mex.2021.101477, https://www.sciencedirect.com/science/article/pii/S2215016121002703
22. Wielinga, B., Schreiber, A., Breuker, J.: Kads: a modelling approach to knowledge engineering. Knowledge Acquisition **4**(1), 5–53 (1992). https://doi.org/https://doi.org/10.1016/1042-8143(92)90013-Q, https://www.sciencedirect.com/science/article/pii/104281439290013Q, the KADS approach to knowledge engineering