

Detecting Zero-Day Vulnerabilities in CMS Platforms: An In-depth Analysis Using DeepLog

Alberto Schiaffino^{1,*}, Matteo Reina^{1,*}, Ricardo Anibal Matamoros Aragon^{2,3,*},
Alessandro Solinas^{2,4,*} and Francesco Epifania²

¹Engitel S.p.A., Milan, Italy

²Social Things S.r.l., Milan, Italy

³Department of Computer Science, University of Milano Bicocca, Milan, Italy

⁴Politecnico di Milano, Milan, Italy

Abstract

In cybersecurity, protecting IT infrastructures from potential threats is one of the most pressing and intricate challenges. Over recent years, we have witnessed an exponential surge in techniques aimed at enhancing defense processes against attacks orchestrated by malicious users. Zero-day attacks are among the most insidious and damaging types of attacks. A Zero-Day attack exploits software vulnerabilities that are unknown to manufacturers or the general public. Due to their concealed nature, these vulnerabilities pose a particularly severe threat: anyone who becomes aware of them before the manufacturers can exploit them, launching attacks with a high likelihood of success since there are no existing solutions or patches to counteract them. Given the importance of preventing and detecting such attacks, we propose an innovative approach based on Deep Learning techniques in this paper. The goal is to conduct anomaly detection analysis on system logs of a Content Management System (CMS) platform. To achieve this, we utilized an algorithm known as DeepLog, which leverages Long Short-Term Memory (LSTM) neural networks to identify patterns in the sequential nature of the logs.

Keywords

Anomaly detection, Content Management Systems, Cybersecurity

1. Introduction

Cybersecurity has become one of the primary concerns in the contemporary technological landscape [1]. Cyberattacks have become increasingly sophisticated and harmful with the evolution of technology and network expansion [2]. In particular, Zero Day attacks stand out as one of the most insidious threats in the field of cybersecurity [3].

A Zero Day attack exploits vulnerabilities not yet known to the public or the software manufacturer, meaning that, at the time of the attack, there is no solution or patch to address the vulnerability, leaving the software highly vulnerable. The concealed nature of these attacks

AIABI 2023: 3rd Italian Workshop on Artificial Intelligence and Applications for Business and Industries, November 9, 2023, Milano, Italy

*Corresponding author.


†These authors contributed equally.

✉ alberto.schiaffino@engitel.com (A. Schiaffino); matteo.reina@engitel.com (M. Reina);

ricardo.matamoros@socialthingum.com (R. A. M. Aragon); alessandro.solinas@socialthingum.com (A. Solinas);

francesco.epifania@socialthingum.com (F. Epifania)

ORCID 0000-0002-1957-2530 (R. A. M. Aragon); 0000-0002-5428-3187 (A. Solinas)

 © 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

makes them especially dangerous, as attackers can exploit them knowing there are no ready defenses to stop them.

In this context, the analysis of system logs becomes crucial. Logs record all activities occurring within a system and can provide valuable insights into anomalous or suspicious behaviors. However, given the vastness and complexity of the recorded data, manual analysis becomes unfeasible. Adopting automated techniques, such as those based on Deep Learning, is essential to understand the logs' nature and detecting potentially distrustful activities [4, 5]. Furthermore, log data are unstructured and can vary in the format or semantics from system to system [6]. This diversity poses an additional challenge for effective log analysis. In the existing literature, various approaches employ deep learning (DL) models, as discussed in [7]. These approaches range from simple architectures such as Multi Layer Perceptron (MLP) which are often outperformed by other structures, to Convolutional Neural Networks (CNN) that extend the MLP framework with the addition of convolutional and max pooling layers, enabling the model to capture more complex patterns, while reducing the model's dimension. The most common approach is the use of Recurrent Neural Networks (RNN) because they employ a feedback mechanism that retains their states over time, enabling the learning of sequential patterns in the input data. Our approach suggests using an algorithm known as DeepLog [6], which has garnered recognition as one of the best-performing models in the domain of log analysis. It harnesses the power of Long Short-Term Memory (LSTM) neural networks, which possess a recurrent architecture specifically designed to analyze time series data, such as system logs. LSTMs excel in their ability to remember and recognize long-term patterns within data. The DeepLog algorithm follows a proactive strategy by training the neural network on "normal" logs, allowing it to learn the standard behavior of a system. Once trained, the model can continuously monitor real-time logs, promptly detecting any deviations from the established normal behavior. This capability makes DeepLog well-suited for the early detection of potential Zero Day attacks, which is paramount in today's ever-evolving cybersecurity landscape. In light of these considerations, the choice to leverage DeepLog for log analysis is not only logical but also strategically sound. Its ability to tackle the challenges posed by unstructured and diverse log data, coupled with the power of LSTM networks, positions DeepLog as a potent tool in the ongoing battle against sophisticated cyber threats, including Zero Day attacks. Implementing a system based on DeepLog within a Content Management System (CMS) platform offers several advantages [8, 9]:

- **Proactive Detection:** DeepLog can identify anomalies in real-time, allowing for a swift response to potential threats.
- **Adaptability:** The system can adapt to new norms thanks to machine learning, reducing the risk of false positives.
- **Scalability:** The Deep Learning-based approach can handle large volumes of data, making it ideal for large-scale CMS platforms.

In formal terms, the research question is posed as follows: Is it feasible, through the use of advanced Artificial Intelligence (AI) algorithms, to devise a mechanism capable of conducting anomaly analysis on the log records of a CMS [10] and, upon detecting suspicious activities, promptly notify the CMS administrators concerning potential vulnerabilities related to Zero Day attacks?.

In the subsequent sections of this article, we will illustrate the use case related to the CMS named "Spin&Go Brainy" developed by Engitel Srl. Following that, we will present the log analysis methodology based on DeepLog, describing the structure of the model and its implementation. Subsequently, the obtained results will be showcased and discussed. In conclusion, we will introduce a dashboard representation method to provide visual insights into the logs examined within the CMS.

2. Spin&Go Brainy: An Innovative CMS

Within the context of this article, the application case under examination pertains to the platform "Spin&Go Brainy", conceived, produced, and marketed by Engitel S.p.A [11]. This platform has been designed to assist clients and stakeholders in the development and continuous optimization of their online portals and related activities. It is an integrated and well-structured system, enriched by a series of modules operating synergistically, founded on Artificial Intelligence. These modules aim to guide both decision-makers and content creators through the phases of design, development, and consistent updating of web services, including those highly specialized. Furthermore, the platform promotes technological transfer and digital transformation, aligning with the vision of Industry 4.0 [12].

A CMS, or Content Management System, is software designed to manage the contents of a website in an intuitive and rapid manner, eliminating the need for specific programming expertise [13].

Spin&Go Brainy, a brainchild of Engitel, is recognized as one of the most efficient CMSs available on the market, thanks to its unparalleled stability and versatility. It has been developed using Microsoft's .NET technology but is designed to interface with any other computer system.

Its applicability is vast, attributable to its modular nature and the numerous functionalities that have been integrated over the years. Spin&Go Brainy is employed in a variety of contexts, such as institutional websites, editorial platforms and blogs, e-commerce solutions, showcase sites, applications, document archives, online communities, and smart networks. Engitel's Spin&Go Brainy CMS platform offers a range of advanced features tailored to modern digital needs:

- **Editorial Tools:** Advanced functions like cut & paste from Word, time-deferred content publication, and integration of various data types ensure streamlined content management.
- **Image Handling:** Automatic resizing eliminates the need for multiple image formats, simplifying media management.
- **SEO Optimization:** Features like automatic metadata association and intelligent URL management enhance site visibility on search engines.
- **Data Protection:** The platform ensures user data ownership, aligning with data protection best practices.
- **Document Management:** An intelligent repository system allows for efficient content storage and retrieval.
- **Notifications:** Automated email alerts notify users about new content, ensuring timely updates.

- Event Management: An interactive calendar feature allows for detailed event planning and geolocation.
- Newsletters: Direct back-office newsletter creation with customizable templates facilitates effective communication.
- User Management: Comprehensive user registration tools, including customizable forms and reserved areas, enhance user experience and data security.

In essence, Spin&Go Brainy offers a comprehensive, user-friendly, and SEO-optimized CMS solution, catering to diverse digital content needs.

3. Anomaly detection with DeepLog

3.1. Log extraction

To extract logs successfully, we employed Osquery [14]. This instrument allows its users to obtain low-level information directly from the operating system (OS), such as data from the system registry, information on executing processes, and active network connections. These gathered data can be subsequently used to detect anomaly behaviors and possible security vulnerabilities. Osquery follows a client-server architecture, where the Osquery client executes queries directly on the operating system and sends the results to the Osquery server for further processing and analysis. After acquiring the information about Osquery, the development demo project within the Google Cloud Platform (GCP) Compute Engine service created virtual machine instances [15]. The assigned VM was named "crdhost". Once the logs have been extracted, the work aims to perform an Anomaly Detection Analysis. Specifically, the purpose is to create a multidimensional time series that represents sequences of actions a user performs. An automated Python script executes the Osquery query every ten minutes, and the data is saved in a BigQuery [16] table within the GCP.

3.2. Preprocessing

To embark on the process of modeling our log data using the sequential structure of LSTM (Long Short-Term Memory), a preliminary and crucial step involves the parsing of this data into a more structured and coherent format. This transformation is imperative to ensure that the data is conducive for the intricate operations of LSTM, which thrives on structured sequences. To elucidate with a tangible example, consider the log messages such as "*idchannel=1&idcontent=143547&idmaster=143547&*" in "*idchannel=*,idcontent=*,idmaster=*,...*". This transformation involves the removal or anonymization of specific log values, retaining only the log keys. Such a procedure ensures that the data retains its structural integrity while obfuscating specific values, making it more amenable for sequential modeling.

Once this transformation is achieved, the next phase involves the grouping of these logs. We aggregate them into sequences comprising 20 units each. These sequences serve a dual purpose. Firstly, they act as the foundational input for our DeepLog model. Secondly, they set the stage for the prediction task where the model endeavors to predict the 21st log entry based on the preceding sequence.

The prediction mechanism of DeepLog is designed to offer k possible options for the 21st log entry. This is where the anomaly detection comes into play. If the actual 21st log entry, derived from the real-world data, does not align with any of the 'k' options provided by DeepLog, it is flagged as an anomaly. Such a mechanism is invaluable in scenarios where monitoring and early detection of irregularities are paramount. By leveraging the capabilities of LSTM and the structured approach of DeepLog, we aim to create a robust system that can not only predict but also identify potential anomalies in vast streams of log data.

3.3. Model architecture

The DeepLog architecture is divided in three main components: the log key anomaly detection model, the parameter value anomaly detection model, and the workflow model to diagnose detected anomalies.

During training, the model learns from log entries from the standard system execution path. Each log entry is then parsed to a log key and a parameter value vector. The log key sequence parsed from a training log file is used by DeepLog to train a log key anomaly detection model and to construct system execution workflow models for diagnosis purposes. For each distinct key, DeepLog also trains and maintains a model for detecting system performance anomalies as reflected by these metric values, trained by the parameter value vector sequence of k . During detection, a newly arrived log entry is parsed into a log key and a parameter value vector. DeepLog first uses the log key sequence to detect anomalies and then uses the parameter value sequence to diagnose the detected anomalies further. The model utilizes Long Short-Term Memory (LSTM) to capture the temporal dependencies between log entries. The LSTM block remembers a state for its input as a fixed-dimension vector. The state of an LSTM block from the previous time step is also fed into its following input, together with its (external) data input, to compute a new state and output.

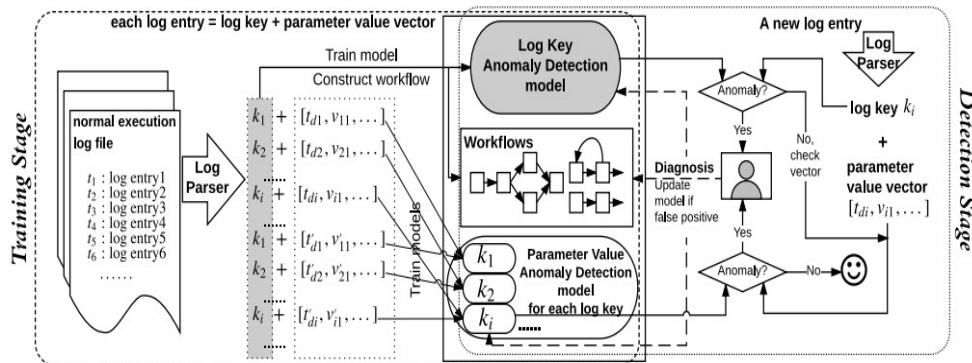


Figure 1: Deeplog architecture [6]

The LSTM block is effective because it can encode more complex patterns with respect to other approaches such as N-grams. In the model's application, given a sequence of log keys, a LSTM network is trained to maximize the probability of having $k_i \in K$ as the next log key value as reflected by the training data sequence. In other words, it learns a probability distribution

$\mathcal{P}(m_t = k_i | m_{t-h}, \dots, m_{t-2}, m_{t-1})$ that maximizes the probability of the training log key sequence [6].

3.4. Dashboard implementation

Prior to initiating the deployment phase of our project within the infrastructure of the Google Cloud Platform (GCP), our team meticulously designed and established an intricate dashboard. This was conceived with the primary objective of facilitating the monitoring process and presenting intricate data sets in a manner that is both intuitive and user-friendly. Such an approach ensures that users, regardless of their technical expertise, can seamlessly analyze the information at their disposal in a timely and efficient manner.

After a comprehensive evaluation of available tools, we decisively selected Looker Studio for this pivotal phase of our project. Our choice was influenced by Looker Studio's reputation as a premier tool in the realm of data visualization. It not only offers the capability to create dashboards and reports that are lucid and straightforward to interpret but also provides extensive customization options. This adaptability ensures that the dashboard aligns perfectly with the nuanced requirements of our project and the diverse needs of its end-users.

To elucidate further on the features we integrated:

- We incorporated two distinct drag-and-drop functionalities. These are designed to filter data based on general metrics and anomalies. Such a feature empowers users by granting them the autonomy to tailor their data viewing experience, aligning it with their specific areas of interest or concern.
- Our dashboard boasts a time series visualization. On the x-axis, it delineates the data column that signifies the log registration instance within our platform. Concurrently, the y-axis portrays the number of daily logs. This strategic representation is indispensable for stakeholders to discern patterns, trends, and potential anomalies over a specified duration.
- We have also embedded a specialized table dedicated to highlighting the most recent anomaly logs. This proactive feature ensures that users are promptly alerted to potential discrepancies, facilitating swift corrective actions.
- Furthermore, we conducted an exhaustive analysis of the status codes inherent in our dataset. This analysis is instrumental in offering stakeholders a granular view of the system's operational health and its overall performance metrics.

For stakeholders and interested parties seeking a tangible representation of our dashboard's capabilities, we direct them to [Figure 2](#). Herein, a detailed and illustrative depiction of our dashboard is presented, encapsulating its sophistication and utility.

4. Analysis of results

Deeplog has been deeply tested with literature datasets in works such as [17]. The model has the advantage of not requiring abnormal logs to build a detection model using sequential vectors, thus reducing the effort for model construction. It has also been highlighted that, unlike other

SI Engitel 4.0 Dashboard - DeepLog

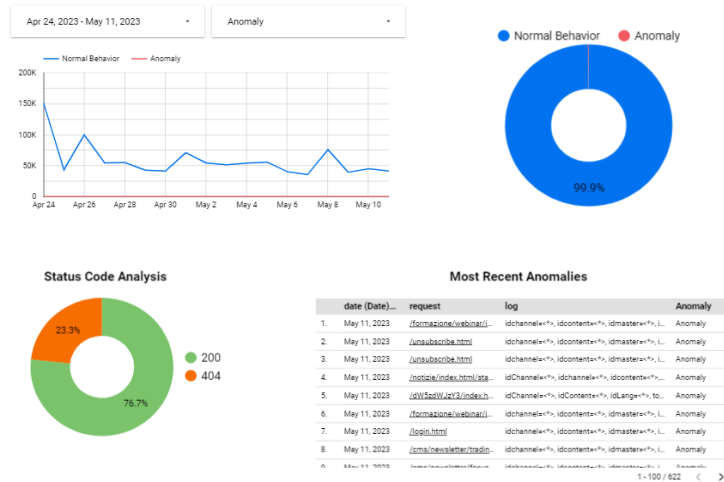


Figure 2: Dashboard visualization example

DeepLearning (DL) models, DeepLog can detect anomalies faster. Looking at the performances on baseline log datasets, Deeplog obtains Recall well above 90% and, in some cases, even at 100% while not penalizing too much the Precision, making it a reliable tool to correctly genuine anomalies but with a low False Positive rate. Unfortunately, we do not have labeled logs on our use case data, meaning we must evaluate the model through unsupervised techniques. In order to accomplish this, we set up an experiment where we remove one user at a time from the training data and evaluate if the new user logs are detected as anomalies by the model. In this way, the model works in Leave One Out Cross-Validation on the users [18], aiming at identifying anomalous user behavior on the one left out.

The underlying philosophy of the methodology we are inclined to employ hinges on a distinctive premise: to gauge the efficacy of an algorithm in assimilating a user's typical behavior, it is imperative to juxtapose its understanding against activities manifested by other users. This approach is predicated on the assumption that individuals, when observed across specific dimensions, exhibit a marked variance in their behaviors. In essence, the uniqueness of individual behavior patterns becomes the yardstick against which the algorithm's learning is measured. This hypothesis, while theoretically sound, required empirical validation to ascertain its practical applicability. To this end, we embarked on a rigorous experiment. Our analysis encompassed a substantial dataset comprising 47,604 unique log sequences. The results were illuminating. The model, trained on the aforementioned principle, flagged all but 44 of these sequences as anomalies. This outcome not only underscores the model's proficiency in internalizing and recognizing standard user behaviors but also its acumen in pinpointing deviations. Such deviations, given their divergence from the established norm, are inherently treated as suspicious, warranting further scrutiny.

Furthermore, this experiment's findings bolster the argument that individuals, even when operating within similar environments or frameworks, tend to manifest discernible behavioral differences. Our model's ability to discern these nuances and flag anomalies with such precision is a testament to its robustness and the viability of the underlying concept.

5. Conclusions

Our endeavor is a comprehensive project that strategically integrates the capabilities of DeepLog, capitalizes on the intricate LSTM architectures, and optimally utilizes the resources provided by the GCP to innovate log management processes within CMSs. The overarching objective of this initiative is to strengthen the fortifications of IT security, ensuring a more resilient and robust digital infrastructure.

By incorporating an advanced Artificial Intelligence system into our framework, we have managed to streamline the traditionally complex tasks of classification and anomaly detection within pivotal logs. The LSTM architectures, renowned for their ability to handle sequential data, have proven invaluable in our quest. They facilitate the meticulous analysis of vast log datasets, enabling us to discern and highlight even the most subtle suspicious patterns and anomalous behaviors that might otherwise go unnoticed. Nevertheless, the unsupervised nature of our approach could potentially be exploited by malicious users to camouflage their nefarious activities within the log data.

The GCP, with its renowned scalability and automation process, has been instrumental in our project. Its capabilities have allowed us to manage and process substantial volumes of real-time log data with unparalleled efficiency. The automation of log management processes, a cornerstone of our project, has led to a significant reduction in manual interventions. This not only minimizes the potential for human error but also liberates our team, allowing them to channel their expertise toward more sophisticated security challenges.

Furthermore, by harnessing Machine Learning models that are meticulously trained within the GCP environment, we have been able to craft an intelligent solution. This solution stands out for its ability to perform real-time anomaly detection, ensuring that security incidents are not just identified but also addressed with alacrity and precision. However, we recognize the imperative to implement measures to thwart potential exploitation by malicious actors, ensuring that our system remains secure and reliable.

Looking ahead, we envision several future developments:

- **Continuous Learning:** As cybersecurity threats evolve, so should our defenses. We plan to implement a continuous learning mechanism where our system will constantly update its knowledge based on new logs and emerging threat patterns [19].
- **Integration with Other Systems:** We aim to expand the compatibility of our solution, allowing it to integrate seamlessly with various other CMSs and IT infrastructures.
- **User Feedback Loop:** To further refine our anomaly detection, we intend to incorporate a feedback loop, allowing users to validate or dispute detected anomalies, thereby enhancing the system's accuracy over time.
- **Advanced Visualization Tools:** To aid in the interpretation of results and insights, we are exploring the integration of advanced visualization tools that can represent complex

patterns in an easily digestible format.

- Enhanced Supervision: In light of the potential risks associated with unsupervised approaches, we are considering the integration of supervised or semi-supervised mechanisms to provide additional layers of security and mitigate the risks of malicious exploitation.

In summation, our project epitomizes the potential and efficacy of amalgamating DeepLog and LSTM architectures within the GCP framework to redefine log management and elevate cybersecurity standards. While recognizing the potential vulnerabilities associated with unsupervised learning, we remain sanguine that a myriad of organizations, regardless of their scale or domain, can integrate this solution to optimize their log management processes and fortify their defenses against ever-evolving cybersecurity threats.

References

- [1] Dandurand, Luc, and Oscar Serrano Serrano. "Towards improved cyber security information sharing." 2013 5th International Conference on Cyber Conflict (CYCON 2013). IEEE, 2013.
- [2] Ustundag, Alp, et al. "Overview of cyber security in the industry 4.0 era." *Industry 4.0: managing the digital transformation* (2018): 267-284.
- [3] Hindy, H., Atkinson, R., Tachtatzis, C., Colin, J. N., Bayne, E., & Bellekens, X. (2020). Utilising deep learning techniques for effective zero-day attack detection. *Electronics*, 9(10), 1684.
- [4] Lorenzen, Casey, Rajeev Agrawal, and Jason King. "Determining viability of deep learning on cybersecurity log analytics." 2018 IEEE International Conference on Big Data (Big Data). IEEE, 2018.
- [5] Li, Z., & Oprea, A. (2016, November). Operational security log analytics for enterprise breach detection. In 2016 IEEE Cybersecurity Development (SecDev) (pp. 15-22). IEEE.
- [6] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. "DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning." In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*, pages 1285–1298, Association for Computing Machinery, Dallas, Texas, USA, 2017. ISBN: 978-1450349468. DOI: 10.1145/3133956.3134015. URL: <https://doi.org/10.1145/3133956.3134015>.
- [7] Max Landauer, Sebastian Onder, Florian Skopik, and Markus Wurzenberger. *Deep learning for anomaly detection in log data: A survey. Machine Learning with Applications*, Volume 12, 2023, Pages 100470. ISSN: 2666-8270. DOI: <https://doi.org/10.1016/j.mlwa.2023.100470>. URL: <https://www.sciencedirect.com/science/article/pii/S2666827023000233>.
- [8] Short, C. (2010). Web content management: CMS for competitive advantage. *Journal of Direct, Data and Digital Marketing Practice*, 11, 198-206.
- [9] de Sousa Saraiva, Joao, and Alberto Rodrigues Da Silva. "CMS-based web-application development using model-driven languages." 2009 Fourth International Conference on Software Engineering Advances. IEEE, 2009.
- [10] Kumar, Gulshan, Krishan Kumar, and Monika Sachdeva. "The use of artificial intelligence based techniques for intrusion detection: a review." *Artificial Intelligence Review* 34 (2010): 369-387.
- [11] Schiaffino, A., Reina, M., Aragon, R. A. M., Epifania, F., Ruggeri, F., Castrignano, I. M., & Marconi, L. (2021). CMS Optimisation with Deep Learning Techniques. In *AIABI@ AI* IA*.

- [12] Ghobakhloo, Morteza, et al. "Industry 4.0, innovation, and sustainable development: A systematic review and a roadmap to sustainable innovation." *Business Strategy and the Environment* 30.8 (2021): 4237-4257.
- [13] Huba, Mikuláš, and Štefan Kozák. "From E-learning to Industry 4.0." 2016 International Conference on Emerging eLearning Technologies and Applications (ICETA). IEEE, 2016.
- [14] Park, So-Hyun, et al. "Performance evaluation of open-source endpoint detection and response combining google rapid response and osquery for threat detection." *IEEE Access* 10 (2022): 20259-20269.
- [15] Krishnan, S. P. T., Gonzalez, J. L. U., Krishnan, S. P. T., & Gonzalez, J. L. U. (2015). *Google compute engine. Building Your Next Big Thing with Google Cloud Platform: A Guide for Developers and Enterprise Architects*, 53-81.
- [16] Fernandes, Sérgio, and Jorge Bernardino. "What is bigquery?." *Proceedings of the 19th International Database Engineering & Applications Symposium*. 2015.
- [17] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. *DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning*. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17)*, pages 1285–1298, Dallas, Texas, USA, 2017. Association for Computing Machinery. ISBN: 9781450349468. <https://doi.org/10.1145/3133956.3134015>.
- [18] Wong, Tzu-Tsung. "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation." *Pattern recognition* 48.9 (2015): 2839-2846.
- [19] Yamin, Muhammad Mudassar, Basel Katt, and Mariusz Nowostawski. "Serious games as a tool to model attack and defense scenarios for cyber-security exercises." *Computers & Security* 110 (2021): 102450.