



Analysis of Backdoor Shells in Web Servers Using Splunk and SPL-Based Machine Learning

Anjik Sukmaaji¹, Slamet², Sophie Anastasya Putri BR.³

^{1,2,3}Information System Department, Faculty of Technology and Informatics, Universitas Dinamika, Surabaya, Indonesia

¹anjik@dinamika.ac.id, ²slamet@dinamika.ac.id, ³21410100012@dinamika.ac.id

Abstract

Backdoor shell attacks pose a critical threat to web server security, allowing attackers to bypass authentication and gain persistent, unauthorized control. Conventional signature-based detection methods often fail against these attacks due to their polymorphic and obfuscation techniques. To address this, we propose an integrated detection approach leveraging Splunk as a log management platform combined with Search Processing Language (SPL)-based machine learning (ML) models. This study collected and preprocessed web server log data using SPL queries, transforming it into structured features for classification. We evaluated two supervised learning algorithms, Logistic Regression and Random Forest, on a labeled dataset comprising both normal traffic and simulated backdoor shell attacks. The evaluation showed that while Logistic Regression achieved a solid performance with 93.5% accuracy and 87.8% recall, the Random Forest model significantly outperformed it. Random Forest reached an accuracy of 97.2%, with a precision of 95.8%, recall of 94.1%, and an F1-score of 94.9%. Crucially, it also reduced the false negative rate (FNR) to 2.3% and the false positive rate (FPR) to 3.5%, making it more reliable for real-time applications. Our findings demonstrate that Random Forest, when integrated with Splunk's SPL, provides a highly robust and practical detection mechanism that effectively distinguishes malicious activities. The primary contribution of this research is an end-to-end architecture that combines scalable log management, effective feature engineering, and advanced ML detection, offering a scalable and practical solution for enterprise-level security monitoring.

Keywords: backdoor shell; logistic regression; random forest; search processing language (SPL); splunk; web server security

How to Cite: A. Sukmaaji, Slamet, and S. Anastasya Putri BR, "Analysis of Backdoor Shells in Web Servers Using Splunk and SPL-Based Machine Learning", *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 10, no. 1, pp. 95 - 101, Feb. 2026.

Permalink/DOI: <https://doi.org/10.29207/resti.v10i1.7215>

Received: September 28, 2025

Accepted: February 2, 2026

Available Online: February 21, 2026

This is an open-access article under the [CC BY 4.0 License](#)

Published by [Ikatan Ahli Informatika Indonesia](#)

1. Introduction

Web servers serve as a cornerstone of modern digital infrastructure, facilitating a wide array of services ranging from global e-commerce to essential government and educational applications[1], [2]. However, their pervasive role and constant exposure to the internet make them a primary target for sophisticated cyberattacks, among which backdoor shells present an especially critical threat[3], [4]. A backdoor shell is a malicious script, typically written in languages such as PHP or ASP, that is injected into a server to establish persistent and unauthorized remote access[5]. This malicious mechanism allows attackers to bypass standard security controls, execute arbitrary commands, and gain complete control over the compromised system, often without immediate detection[6], [7].

The rising prevalence of polymorphic and obfuscation techniques employed by modern attackers has rendered conventional signature-based and static analysis methods largely ineffective[5], [7]. While these traditional approaches which rely on matching known patterns or digital signatures have historically shown a high detection rate of approximately 98%, they are fundamentally vulnerable to novel or slightly modified variants of web shells[3]. For instance, empirical research has demonstrated that static analysis and string matching often achieve an accuracy of only around 75% when confronted with advanced obfuscation techniques[4], [6].

More recently, machine learning (ML) and deep learning (DL), approaches have emerged as promising alternatives for detecting these intricate attack patterns[8], [9]. Research has explored various models

that show high accuracy in controlled environments; for example, CNN-based deep learning approaches have achieved up to 98.50% accuracy[7], [10]. However, such methods often suffer from significant computational overhead, which severely limits their scalability in live environments[8], [11]. Furthermore, models like CNN and RNN-LSTM perform optimally only when applied to clean, well-structured datasets requiring labor-intensive labeling[12], [13]. These findings underscore a critical gap: while advanced methods show promise, their practical application in real-world, high-traffic production environments remains a significant challenge due to high computational costs and rigorous data quality requirements[14], [15].

To improve resilience against evasive attacks, recent research has attempted to integrate adaptive approaches such as hybrid learning and feature fusion[4], [7]. Notable examples include combining visual and textual modalities to handle obfuscation, as well as proposing CNN-BiLSTM architectures to capture complex features in both normal and anomalous traffic. Despite these advancements, several research gaps remain in the practical implementation of detection methods at the enterprise level. Existing studies rarely validate their models using large-scale log data from high-traffic production servers, raising questions about scalability[16], [17]. Moreover, limited attention has been given to the direct integration of ML models into enterprise platforms like Splunk, despite its native Search Processing Language (SPL) capabilities that enable practical, real-time monitoring. Finally, there is an insufficient analysis of advanced evasion strategies where backdoor shells disguise traffic using legitimate user-agents to bypass conventional filters[6], [18], [19].

A comparative analysis of detection methodologies reveals a critical trade-off between peak predictive performance and operational scalability for enterprise environments. Deep learning (DL) approaches, particularly those utilizing CNN-based architectures, have demonstrated exceptional detection capabilities, achieving accuracy rates as high as 98.50% within controlled settings[10], [15].

However, these advanced models are frequently associated with significant computational overhead[8] [12], which poses a challenge for real-time scalability in high-traffic production networks. Furthermore, the efficacy of DL models like CNN and RNN-LSTM is heavily dependent on the availability of clean, well-structured datasets and often requires labor-intensive labeling processes to maintain optimal performance.

In contrast, ensemble learning methods such as Random Forest (RF) provide a more pragmatic solution for large-scale security monitoring[20], [21]. While the RF model in this study achieved a slightly lower accuracy of 97.2% compared to theoretical DL maximums, it offers superior computational efficiency and a robust ability to capture complex, non-linear relationships without the same resource intensity[16], [22]. Most

importantly, the RF approach demonstrated high operational reliability by achieving a significantly low False Negative Rate (FNR) of 2.3% and a False Positive Rate (FPR) of 3.5%. This balanced performance profile makes Random Forest more suitable for real-time deployment, as it effectively minimizes both the risk of undetected attacks and the operational burden of alert fatigue[23].

To address these gaps, this study proposes an end-to-end architecture for backdoor shell detection on web servers. Our research leverages Splunk as a scalable log management and analytics platform, combined with SPL-based feature engineering and optimized machine learning models. The primary objectives of this study are threefold: First, it aims to design and implement a robust detection system within Splunk that is capable of capturing subtle evasion patterns within high-volume web server traffic. Second, the proposed system is validated using a comprehensive dataset and systematically compared with conventional methods to highlight its relative advantages in a production-like environment. Finally, the research evaluates system performance through established metrics including accuracy, precision, recall, and F1-score with a specific focus on minimizing false positives and false negatives to ensure reliability in enterprise-level deployment.

2. Methods

The methodology of this research follows a structured experimental framework designed to evaluate the effectiveness of machine learning models in a practical security environment[24]. This approach encompasses the establishment of a controlled data collection environment, the design of specialized system architecture for real-time monitoring, and a rigorous application of feature engineering and classification algorithms. The following sections detail the technical environment and the procedural pipeline used to develop the detection system.

2.1 Data and Research Environment

This study was conducted in a controlled laboratory environment designed to mimic a real-world web server deployment. The testbed comprised a web server running on Linux Ubuntu 22.04 LTS and Apache HTTP Server 2.4.x[16]. To ensure comprehensive data collection, the default Apache logging configuration was supplemented to capture detailed request attributes, including timestamp, src_ip, request_method, uri_path, status_code, and user_agent.

Data were continuously collected over a period of 30 days, amassing over 50,000 log entries. The dataset was meticulously constructed to include two distinct classes: normal web traffic and simulated backdoor shell attack activities. The simulated attacks were carefully orchestrated to include common backdoor shells (e.g., b374k.php, r57.php) as well as obfuscated and polymorphic variants to challenge the detection models[4], [7]. All logs were ingested in real time into

Splunk Enterprise 9.x, which served as the central platform for log indexing, management, and subsequent analysis[6], [18], as shown in Table 1.

Table 1. Example of a Web Server Access Log Snippet

Timestamp	IP Address	Request method	URL Accessed	Status Code	User Agent
2025-07-01 12:32:10	192.168.1.45	GET	/index.php?id=1	200	Mozilla/5.0
2025-07-01 12:33:20	192.168.1.78	POST	/upload.php	500	curl/7.64.1
2025-07-01 12:35:02	192.168.1.99	GET	/b374k.php	200	Python-urllib/3.8

2.2 System Architecture

The proposed system architecture is designed as a scalable solution for real-time backdoor shell detection, consisting of five main operational stages. Initially, log data is ingested into the Splunk Indexer, followed by preprocessing through Search Processing Language (SPL) queries to filter noise and extract key features such as unusual file access patterns[18]. The refined data is then utilized for machine learning modeling within the Splunk Machine Learning Toolkit, enabling real-time detection and automated alerting via the Splunk Dashboard. Finally, the system undergoes rigorous evaluation against standard performance metrics and benchmarking against traditional methods to validate its effectiveness in production environments.

The end-to-end detection pipeline, as illustrated in Figure 1, consists of five main stages designed to ensure effective and practical deployment. First, log data from the web server is continuously ingested into the Splunk Indexer, serving as the central repository for raw information. Next, initial preprocessing is conducted through SPL queries to filter noise and extract key features, including unusual file access patterns and abnormal user-agent strings[5], [6], [19]. The refined data is then utilized for machine learning modelling within the Splunk Machine Learning Toolkit, where supervised classifiers are trained to distinguish malicious from benign traffic. Once deployed, the model enables real-time detection and alerting, automatically notifying administrators via the Splunk Dashboard when suspicious activity occurs. Finally, the system undergoes pattern analysis and rigorous evaluation against standard performance metrics, with results benchmarked against traditional methods to demonstrate the enhanced effectiveness of the proposed approach.

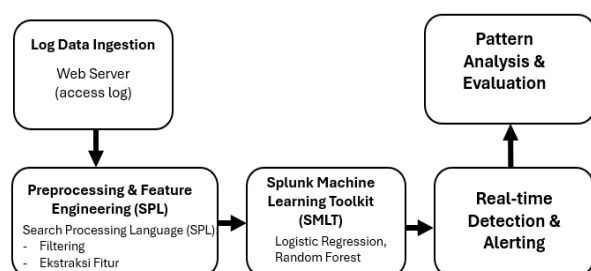


Figure 1. Backdoor Shell Detection System Architecture

The log in Table 1 shows indications of anomalies, such as access to b374k.php, a popular backdoor shell frequently used by hackers.

2.3 Application of SPL and Machine Learning

The integration of SPL and machine learning forms the core of the detection pipeline, enabling the transformation of raw log data into actionable security intelligence. This process transitions from raw data filtering to complex algorithmic classification to ensure high detection accuracy. SPL serves as the primary tool for feature engineering, identifying preliminary indicators of web shell activity such as successful PHP requests through filters like:

```

index=webserver_logs | stats count by src_ip,
uri_path, status | where like (uri_path, "%.php")
AND status=200.

```

SPL serves as the primary tool for feature engineering, identifying preliminary indicators of web shell activity such as successful PHP requests through specialized filters. Beyond basic filtering, advanced features including URL length and parameter count, were engineered to help models recognize obfuscated attacks.

The machine learning component was implemented using the Splunk Machine Learning Toolkit (MLTK), which allows for the training and deployment of models directly on Splunk data without requiring external data export[6], [19]. Two supervised learning algorithms were selected for this binary classification task: Logistic Regression and Random Forest. Logistic Regression was adopted as the baseline model due to its computational efficiency and interpretability[22], establishing a benchmark for distinguishing normal traffic. Random Forest was utilized as an ensemble method to capture complex, non-linear relationships and mitigate overfitting, which is particularly vital for identifying sophisticated backdoor shell requests[8], [10], [16].

Two supervised learning algorithms were employed in this study to address the binary classification task, each selected for its distinct advantages. Logistic Regression was adopted as the baseline model owing to its computational efficiency, interpretability, and minimal overhead, making it suitable for establishing a clear benchmark in distinguishing between normal and anomalous traffic. In contrast, Random Forest was applied to capture complex, non-linear relationships within the dataset. As an ensemble method, it aggregates multiple decision trees to mitigate overfitting and improve detection accuracy,

particularly in identifying sophisticated or obfuscated backdoor shell requests.

The training process utilized a dataset of over 50,000 entries, partitioned using a stratified sampling approach to ensure representativeness. Specifically, 80% of the entries were allocated for training, while the remaining 20% were reserved for an unbiased testing set to avoid data leakage. The final pipeline involves fitting the models on labeled data where normal traffic is assigned a class of 0 and anomalous activity a class of 1, followed by continuous deployment on live log streams for real-time anomaly detection[24].

2.4 Experimental Design

The effectiveness of the proposed system was evaluated through quantitative experiments using a testing set partitioned via a stratified sampling approach. Performance was assessed using standard classification metrics including Accuracy, Precision, Recall, and F1-Score[20]. Given the critical nature of security applications, a detailed analysis of False Positive (FP) and False Negative (FN) rates was conducted, as minimizing the FN rate is the highest priority to ensure no threats go undetected[25]. To ensure robustness and prevent overfitting, a k-fold cross-validation procedure was performed on the training data before final testing on the unseen dataset[26]. The mathematical formulation for these metrics is provided in Equations 1-4.

Accuracy measures the overall correctness of the model's predictions.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

TP is the True Positive, TN is the True Negative, FP is the False Positive, and FN is the False Negative.

Precision assesses the model's ability to avoid false alarms.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Recall (Sensitivity) measures the model's ability to correctly identify all instances of attacks.

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

F1-Score: Provides a balanced measure of both precision and recall, which is particularly useful for imbalanced datasets.

$$F1 = 2X \frac{Precision \times Recall}{Precision+Recall} \quad (4)$$

Given the critical nature of security applications, a detailed analysis of False Positive (FP) and False Negative (FN) rates was conducted. This analysis is crucial for evaluating the model's real-world practicality.

False Positive (FP) occurs when a legitimate request is mistakenly flagged as an attack. A high FP rate can

lead to alert fatigue and wasted resources for security teams.

False Negative (FN) occurs when a malicious attack is incorrectly classified as normal traffic, allowing the threat to go undetected. Minimizing the FN rate is the highest priority in a security system to ensure no attacks are missed.

To ensure the robustness and generalization of the models, a k-fold cross-validation procedure was performed on the training data. This process helps to mitigate the risk of overfitting. Finally, the performance metrics were computed on the unseen testing set to provide a final, unbiased assessment. The results of the Logistic Regression and Random Forest models were then compared to identifying the optimal algorithm for deployment.

3. Results and Discussions

The evaluation of the proposed framework provides evidence of the synergy between rule-based filtering and predictive modeling. The following analysis details the results of the initial SPL filtering, the comparative performance of the classification algorithms, and the practical security implications of the findings.

3.1 SPL Implementation and Initial Analysis

Initial analysis of the 50,000 log entries using SPL successfully identified suspicious access patterns, revealing that 4.4% of total requests were anomalous. The data highlights that malicious activity leaves a clear footprint even in high-volume environments. SPL proved effective as a first-stage filter, identifying 1,240 anomalous PHP file accesses, 530 suspicious uploads, and 430 direct web shell interactions. This preprocessing step is vital as it reduces the dataset size and enhances the quality of data fed to the machine learning models. The breakdown of anomalous requests is summarized in Table 2.

Table 2. Summary of SPL Implementation Results

Access Category	Number of Requests	Percentage
Normal (HTML, API, JS)	47,800	95.6%
File access .php anomali	1,240	2.5%
Upload.php	530	1.1%
Webshell (.php backdoor)	430	0.8%

The data highlights a critical finding: even in a high-volume environment, malicious activity, although a small fraction of the total traffic, leaves a clear footprint. The ability of SPL to act as a powerful first-stage filter is demonstrated by its success in identifying requests to known webshells (b374k.php), suspicious file uploads (upload.php), and other unusual PHP file accesses. This preprocessing step is vital as it reduces the dataset size and enhances the quality of data fed to the machine learning models, thereby improving their efficiency and accuracy. Although the number of anomalies is relatively small, this pattern is crucial because it is an early indicator of backdoor shell

insertion. SPL has proven effective as a first-stage filter before proceeding with machine learning.

3.2 Machine Learning Model Performance and Error Analysis

The performance evaluation on the reserved testing set demonstrated a clear distinction between the two algorithms. Random Forest significantly outperformed Logistic Regression across all metrics, reaching an accuracy of 97.2% and a recall of 94.1%, compared to the baseline's 93.5% accuracy and 87.8% recall. The high recall in the Random Forest model is particularly critical for security applications, as it demonstrates a superior ability to identify actual attacks and minimize the risk of threats going undetected. Table 3 presents a comprehensive comparison of the two algorithms across key performance metrics.

Table 3. Comparison of Algorithm Performance

Algorithm	Accuracy	Precision	Recall	F1-score
Logistic Regression	93.5%	90.2%	87.8%	89.0%
Random Forest	97.2%	95.8%	94.1%	94.9%

As shown in Table 3, the Random Forest model consistently and significantly outperformed Logistic Regression across all metrics. The most notable improvement was in Recall, where Random Forest achieved 94.1% compared to Logistic Regression's 87.8%. This is a particularly critical result for security applications, as a high recall value indicates the model's superior ability to correctly identify a larger proportion of actual attacks, thus minimizing the risk of a threat going undetected.

This superiority is further evidenced by the analysis of error rates, where Random Forest achieved a False Negative Rate (FNR) of only 2.3% and a False Positive Rate (FPR) of 3.5%. In contrast, Logistic Regression suffered from a 5.0% FNR and a 7.2% FPR. The reduction in FPR is essential for real-world deployment, as it decreases alert fatigue for security administrators, while the lower FNR ensures that the system is more reliable at preventing attacks from bypassing detection.

The real-world applicability of a security system is determined by its ability to minimize both false positives and false negatives. Our analysis shows a stark contrast between the two models in Table 4.

Table 4. FP and FN of the two models

Algorithm	False Positive Rate (FPR)	False Negative Rate (FNR)
Logistic Regression	7.20%	5.00%
Random Forest	3.50%	2.30%

These results underscore why Random Forest is the more suitable algorithm for real-time deployment. It achieved a significantly lower False Negative Rate (FNR) of only 2.3%, meaning it missed fewer actual attacks compared to Logistic Regression's 5%. This is crucial in preventing attacks from bypassing the

detection system. Additionally, the lower False Positive Rate (FPR) of 3.5% for Random Forest reduces alert fatigue for security administrators, ensuring that fewer legitimate requests are mistakenly flagged as threats.

3.3 Security Implications and Future Directions

The findings highlight a robust synergy where SPL serves as a reliable rule-based filter for known patterns, while machine learning addresses zero-day or obfuscated attacks. This hybrid approach offers a more adaptive mechanism than conventional techniques and enhances operational efficiency through automated real-time alerting. Future research should focus on the application of unsupervised learning techniques, such as Isolation Forest, to strengthen detection of novel threats and expanding the dataset to include broader web-based threats like SQL injection. Additionally, large-scale evaluation in real production environments remains essential to optimize system performance under high-traffic conditions.

The results of this research present significant implications for strengthening web server security. First, the proposed system demonstrates improved detection capability, offering a more adaptive mechanism for identifying backdoor shells compared to conventional signature-based techniques. Second, by leveraging automated real-time alerting through the Splunk dashboard, operational efficiency is enhanced as IT teams can minimize manual log inspection and respond more promptly to threats. Finally, the framework supports proactive threat mitigation, with the potential to integrate automated response mechanisms such as firewall rules to block malicious IP addresses upon anomaly detection. Collectively, these contributions underscore the system's practical value for enterprise-level security operations.

Based on these findings, several promising directions for future research are identified. First, the application of unsupervised learning techniques, such as Isolation Forest or Autoencoders, could strengthen the system's ability to detect novel or zero-day attacks beyond predefined patterns. Second, extending integration with broader SIEM platforms would enable correlation of diverse log sources, offering a more holistic perspective of attack chains. Third, expanding the dataset to encompass additional web-based threats, including SQL injection, cross-site scripting (XSS), and brute-force login attempts, would enhance the model's generalizability and robustness. Finally, large-scale evaluation in real production environments remains essential to optimize system performance and assess resource efficiency under high-traffic conditions.

A key limitation of this study lies in the reliance on a lab-generated dataset, which, although controlled and well-structured, may not fully capture the complexity, variability, and noise present in real-world web traffic. As a result, the model's performance might differ when

deployed in dynamic enterprise environments where attack patterns are more diverse and often obfuscated. To address this, future research should focus on validating the system against real-world traffic collected from production servers or large-scale network environments. Such validation would provide a more accurate assessment of the model's robustness, scalability, and applicability in operational security contexts.

4. Conclusions

This research successfully implemented a practical and scalable solution for detecting backdoor shell attacks by integrating Splunk, SPL-based feature engineering, and machine learning models. The results demonstrate that while traditional methods have limitations, the synergy of log management and Random Forest classification achieves a high accuracy of 97.2% and a low false negative rate of 2.3%. This integrated architecture provides a robust framework for enterprise-level security monitoring. Future work will focus on incorporating unsupervised learning and automated response mechanisms to further transform the framework into a proactive defense system.

This research presented a practical and scalable solution for detecting backdoor shell attacks on web servers by integrating Splunk, its native Search Processing Language (SPL), and machine learning (ML) models. The findings demonstrate that this combined approach offers a more robust and effective detection mechanism than traditional methods alone. The initial analysis using SPL proved highly effective as a preprocessing and filtering stage, successfully identifying 2,200 suspicious requests from a total of 50,000 log entries. This step highlights SPL's capability to efficiently pinpoint anomalous patterns in high-volume web traffic, setting a clean and well-structured foundation for the subsequent ML analysis.

The core of our contribution lies in the performance of the ML models. While Logistic Regression achieved a respectable 93.5% accuracy, the Random Forest model significantly outperformed it across all metrics, reaching an accuracy of 97.2%, with a notable recall of 94.1%. The superior recall of the Random Forest model is particularly crucial for security applications, as it demonstrates a higher ability to correctly identify genuine attacks and minimize false negatives. A detailed analysis confirmed this, with Random Forest achieving a significantly lower false negative rate (2.3%) compared to Logistic Regression (5.0%), proving its reliability for real-time threat detection.

Ultimately, the synergy of Splunk's powerful log management capabilities with ML-based pattern recognition creates a robust, adaptive, and practical system. This approach not only provides a solution for detecting sophisticated, evasive attacks but also offers a scalable and efficient framework for enterprise-level security monitoring.

Building upon the findings and limitations of this study, several directions for future research are recommended to further enhance the system's effectiveness. First, the integration of unsupervised learning methods, such as Isolation Forest or Autoencoders, could strengthen the system's capacity to detect zero-day attacks without predefined patterns. Second, expanding its scope through integration with additional SIEM tools would enable holistic threat correlation across diverse data sources, including firewalls, intrusion detection systems (IDS), and web application firewalls (WAFs), thereby offering a more comprehensive view of attack campaigns. Third, enlarging the training dataset to cover a broader spectrum of web-based attacks such as SQL injection, cross-site scripting (XSS), and brute-force attempts would improve the model's generalization and resilience. Fourth, performance optimization through large-scale deployment and evaluation in production environments is essential to ensure scalability and resource efficiency under heavy traffic conditions. Finally, the incorporation of automated response mechanisms, such as firewall or WAF integration to block malicious IPs in real time, would transform the framework into a more proactive and adaptive defense system for enterprise-level security.

References

- [1] Z. Ai, N. Luktarhan, A. J. Zhou, and D. Lv, "WebShell attack detection based on a deep super learner," *Symmetry (Basel)*, vol. 12, no. 9, Sep. 2020, doi: [10.3390/SYM12091406](https://doi.org/10.3390/SYM12091406).
- [2] T. Dinh Tu, C. Guang, G. Xiaojun, and P. Wubin, "Webshell Detection Techniques in Web Applications." [Online]. Available: <http://cctea-m.ru/update/c999shell>,
- [3] Y. Guo, H. Marco-Gisbert, and P. Keir, "Mitigating webshell attacks through machine learning techniques," *Future Internet*, vol. 12, no. 1, Jan. 2020, doi: [10.3390/fi12010012](https://doi.org/10.3390/fi12010012).
- [4] A. Zhou, N. Luktarhan, and Z. Ai, "Research on webshell detection method based on regularized neighborhood component analysis (Rnca)," *Symmetry (Basel)*, vol. 13, no. 7, Jul. 2021, doi: [10.3390/sym13071202](https://doi.org/10.3390/sym13071202).
- [5] R. Project, T. Un, and A. Duong Trinh, "Analyzing Obfuscation Techniques for Evasion: A Case Study on Machine Learning-based Malware Detection," <https://norma.ncirl.ie/id/eprint/7718>
- [6] P. Shelke and T. Frantti, "Exploring the Possibilities of Splunk Enterprise Security in Advanced Cyber Threat Detection." *International Conference on Cyber Warfare and Security*, vol. 20, no. 1, pp. 605-613, 2025, doi: [10.34190/iccws.20.1.3326](https://doi.org/10.34190/iccws.20.1.3326)
- [7] Y. Zhang, D. Li, and Y. Xie, "GAShellBreaker: A Novel Method for Java Fileless Webshell Detection Based on Grayscale Images and Deep Learning," *Electronics (Switzerland)*, vol. 14, no. 8, Apr. 2025, doi: [10.3390/electronics14081678](https://doi.org/10.3390/electronics14081678).
- [8] B. Yong et al., "Ensemble machine learning approaches for webshell detection in Internet of things environments," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 6, Jun. 2022, doi: [10.1002/ett.4085](https://doi.org/10.1002/ett.4085).
- [9] A. Alshuaibi, M. Almaayah, and A. Ali, "Machine Learning for Cybersecurity Issues: A systematic Review," *Journal of Cyber Security and Risk Auditing*, vol. 2025, no. 1, pp. 36-46, 2025, doi: [10.63180/jcsra.thestap.2025.1.4](https://doi.org/10.63180/jcsra.thestap.2025.1.4).

- [10] A. Waqas, S. D. Khan, Z. Ullah, M. Ullah, and H. Ullah, "Comparative Analysis of Deep Learning Models for Intrusion Detection in IoT Networks," *Computers*, vol. 14, no. 7, Jul. 2025, doi: [10.3390/computers14070283](https://doi.org/10.3390/computers14070283).
- [11] S. Elsayed, K. Mohamed, and M. A. Madkour, "A Comparative Study of Using Deep Learning Algorithms in Network Intrusion Detection," *IEEE Access*, vol. 12, pp. 58851–58870, 2024, doi: [10.1109/ACCESS.2024.3389096](https://doi.org/10.1109/ACCESS.2024.3389096).
- [12] C. E. Ben Ncir, M. A. Ben HajKacem, and M. Alattas, "Enhancing intrusion detection performance using explainable ensemble deep learning," *PeerJ Comput. Sci.*, vol. 10, 2024, doi: [10.7717/PEERJ-CS.2289](https://doi.org/10.7717/PEERJ-CS.2289).
- [13] M. C. Wibowo, I. Setyawan, A. Setiawan, and I. Sembiring, "Deep Learning-Based Visualization of Network Threat Patterns Using GAN-Generated Infographic," *Jurnal RESTI*, vol. 9, no. 4, pp. 765–770, 2025, doi: [10.29207/resti.v9i4.6717](https://doi.org/10.29207/resti.v9i4.6717).
- [14] H. Y. Kwon, T. Kim, and M. K. Lee, "Advanced Intrusion Detection Combining Signature-Based and Behavior-Based Detection Methods," *Electronics (Switzerland)*, vol. 11, no. 6, Mar. 2022, doi: [10.3390/electronics11060867](https://doi.org/10.3390/electronics11060867).
- [15] K. Lamon, "Intrusion detection for industrial control systems," Eastern Washington University, Washington, 2021. [Online]. Available: <https://dc.ewu.edu/theses>
- [16] M. Ridwan, I. Sembiring, A. Setiawan, and I. Setyawan, "Analysis of Attack Detection on Log Access Servers Using Machine Learning Classification: Integrating Expert Labeling and Optimal Model Selection," *Scientific Journal of Informatics*, vol. 11, no. 1, pp. 119–126, Feb. 2024, doi: [10.15294/sji.v11i1.49424](https://doi.org/10.15294/sji.v11i1.49424).
- [17] R. Jain, S. Tschisgale, and J. Fröhlich, "A collision model for DNS with ellipsoidal particles in viscous fluid," Mar. 2022, doi: [10.1016/j.ijmultiphaseflow.2019.103087](https://doi.org/10.1016/j.ijmultiphaseflow.2019.103087).
- [18] X. Pan, "Independent Study of Splunk," *OALib*, vol. 11, no. 04, pp. 1–16, 2024, doi: [10.4236/oalib.1111496](https://doi.org/10.4236/oalib.1111496).
- [19] R. Sahay, M. Sumasadan, B. Eapen, W. Meng, and M. R. Al Mamu, "Enhancing Threat Hunting with Splunk and Generative AI for Automated Security Operations," Oct. 14, 2025, doi: [10.21203/rs.3.rs-7515771/v1](https://doi.org/10.21203/rs.3.rs-7515771/v1).
- [20] K. Shaukat et al., "Performance comparison and current challenges of using machine learning techniques in cybersecurity," May 01, 2020, MDPI AG, doi: [10.3390/en13102509](https://doi.org/10.3390/en13102509).
- [21] A. S. Dina, A. B. Siddique, and D. Manivannan, "Effect of Balancing Data Using Synthetic Data on the Performance of Machine Learning Classifiers for Intrusion Detection in Computer Networks," *IEEE Access*, vol. 10, pp. 96731–96747, 2022, doi: [10.1109/ACCESS.2022.3205337](https://doi.org/10.1109/ACCESS.2022.3205337).
- [22] M. F. Prayogi, A. F. Setiawan, and X. Ariwibisono, "Perancangan Sistem Keamanan Web Menggunakan Metode Random Forest (Studi Kasus : Institusi X)," *JATI (Jurnal Mahasiswa Teknik Informatika)*, 2025, DOI: [10.36040/jati.v9i6.15457](https://doi.org/10.36040/jati.v9i6.15457).
- [23] S. Tariq, M. B. Chhetri, S. Nepal, and C. Paris, "Alert Fatigue in Security Operations Centres: Research Challenges and Opportunities," *ACM Comput. Surv.*, vol. 57, no. 9, Apr. 2025, doi: [10.1145/3723158](https://doi.org/10.1145/3723158).
- [24] A. K. Jakkani, "Real-Time Network Traffic Analysis and Anomaly," *Journal of Electronics, Computer Networking and Applied Mathematics*, vol. 04, pp. 32–44, 2024, doi: [10.55529/jecnam.44.32.44](https://doi.org/10.55529/jecnam.44.32.44).
- [25] M. Xue, C. Yuan, H. Wu, Y. Zhang, and W. Liu, "Machine Learning Security: Threats, Countermeasures, and Evaluations," *IEEE Access*, vol. 8, pp. 74720–74742, 2020, doi: [10.1109/ACCESS.2020.2987435](https://doi.org/10.1109/ACCESS.2020.2987435).
- [26] S. A. Mandowen, A. M. Vulfin, V. I. Vasilyev, E. R. Khairullin, and J. K. Wororomi, "A Robust Multi-Validation Approach for Evaluating Machine Learning-Based Intrusion Detection Models," *BIO Web Conf.*, vol. 213, p. 03001, Jan. 2026, doi: [10.1051/bioconf/202621303001](https://doi.org/10.1051/bioconf/202621303001).