AI-Driven Malware Detection Using Static Code Inspection and Network Flow Monitoring

Roheen Qamar[1], Baqar Ali Zardari[1], Zahid Hussain[1], Aijaz Ahmed Arain[2], Areeba Qamar[2]
[1]Department of Information Technology, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah, Pakistan
[2]Department of Computer Science, Quaid-e-Awam University of Engineering, Science and Technology, Nawabshah, Pakistan
*__Correspondence__: roheen.qamar04@yahoo.com

Citation | Qamar. R, Zardari. B. A, Hussain. Z, Arain. A. A, Qamar A "AI-Driven Malware Detection Using Static Code Inspection and Network Flow Monitoring", IJIST, Vol. 7 Issue. 10 pp 176-183, December 2025
Received| November 13, 2025 **Revised|** December 05, 2025 **Accepted|** December 10, 2025 **Published|** December 13 2025.

Malware refers to software designed to cause harm, steal data, disrupt services, or gain unauthorized access, often employed by attackers for financial gain or sabotage. Traditional detection methods struggle against novel threats that evade signature-based systems. This research introduces an AI-driven hybrid malware detection framework combining static code inspection with real-time network flow analysis. Developed in Python using YARA and Scapy, the system uses machine learning to enhance detection accuracy by correlating code-level and behavioral indicators, effectively identifying both dormant and active threats while reducing false positives. It also features an interactive interface for educational purposes, helping users understand malware behavior and cybersecurity concepts. Experimental results show improved detection precision and faster analysis compared to traditional methods, making this framework a scalable and instructive solution for malware detection and cybersecurity awareness.

Keywords: Malware Detection, Artificial Intelligence, Machine Learning, Static Code Analysis, Network Flow Monitoring, Hybrid Malware Detection, YARA, Scapy, Cybersecurity.

Introduction:

In recent years, cyber threats, particularly malware, have immensely disrupted individuals and businesses, leading to financial losses, business interruptions, and compromised data security. Malware is designed to infiltrate and exploit systems, and its growing complexity has made traditional detection methods less effective. Conventional approaches, such as signature-based techniques, fail against zero-day and unknown threats, while behavior-based methods require substantial resources. Significant processing capacity is essential for continuously monitoring and analyzing system activities to identify new and evolving threats. A more reliable and effective detection system is needed due to the ever-changing threat landscape, as shown in Figure 1." revised to "as shown in Figure. 1.

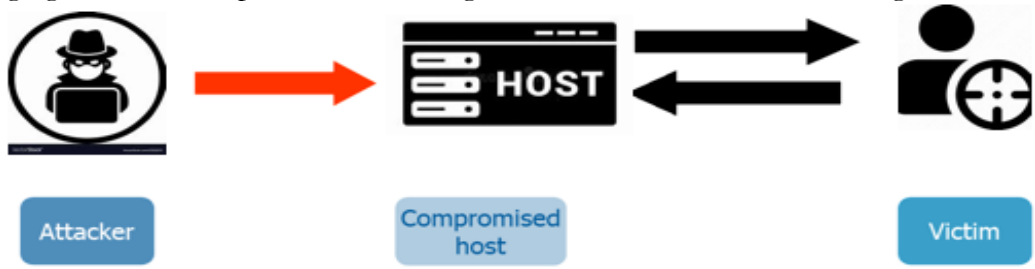

Figure 1. Malware Attack

In figure 1 Malware is classified by its spreading mechanisms: Viruses attach to legitimate files and require user action to execute, causing damage during spreading. Worms exploit network vulnerabilities automatically, using bandwidth and potentially delivering additional malicious payloads. Trojans pretend to be legitimate software, tricking users into installation and allowing attackers unauthorized system control [1].

Spyware, ransomware, and adware are malware types that compromise user privacy and security. Spyware steals sensitive data, leading to privacy violations and identity theft. Ransomware encrypts files, demanding payment for access, resulting in operational and financial harm. Adware tracks user behavior, displaying ads and risking further infections. Together, these threats undermine data integrity for both individuals and organizations [2].

Malware threatens system integrity and privacy, necessitating effective detection methods. This research presents a hybrid malware detection system combining static analysis with network monitoring. Utilizing Python's YARA for code detection and Scapy for packet analysis, the system enhances the identification of file-based and network threats, improving accuracy and efficiency. Additionally, it serves as an educational resource for cybersecurity principles, offering a robust solution to evolving cyber threats [3].

Literature Review:

Research discusses the evolution of malware detection methodologies in response to complex cyber threats, classifying them into static analysis, dynamic analysis, and hybrid approaches [4]. Hybrid methods, integrating various detection mechanisms, enhance accuracy and efficiency, particularly important for resource-constrained environments like the Internet of Things (IoT), where efficient security solutions are essential due to computational limitations and network vulnerabilities.

Static analysis examines binary code without execution, efficiently detecting known malware using signature-based detection, heuristics, and rule-based mechanisms [5]. Tools like YARA enhance this process by enabling precise pattern matching for malware families. However, it struggles with polymorphic and obfuscated malware, which employ advanced techniques like code packing and encryption to evade detection. Despite these challenges, static analysis is a critical first line of defense in malware detection due to its speed, scalability, and effectiveness when combined with more advanced methods.

Network monitoring is crucial for detecting malware by analyzing real-time network traffic, capturing packets to identify suspicious connections and abnormal communication patterns [6]. It offers behavioral insights into malware interactions, unlike static analysis, which focuses on file structure. Key advantages include the detection of stealthy malware and Advanced Persistent Threats (APTs) that evade traditional methods. By monitoring traffic anomalies, such as unusual DNS requests and unauthorized access attempts, it uncovers threats that static analysis may miss. Techniques like deep packet inspection provide deeper visibility into network behavior, though network monitoring also presents challenges.

Intelligent filtering mechanisms are essential to minimize false positives due to encrypted malware traffic, evasion techniques, and high network data volumes [7]. Advanced solutions utilize machine learning and artificial intelligence to improve detection accuracy by distinguishing between legitimate and malicious traffic.

Hybrid malware detection systems have emerged as a robust solution, addressing the limitations of static and network-based approaches by combining various techniques for improved security [8]. These systems integrate static analysis with network monitoring, enabling comprehensive threat detection, including known and unknown threats like zero-day malware. Recent advancements include machine learning-enhanced models that analyze extensive malware datasets to uncover hidden correlations and emerging patterns. These models utilize supervised learning to classify malicious and benign traffic, unsupervised anomaly detection for suspicious behaviors, graph-based methods for mapping malware relationships, and natural language processing for analyzing malware logs and documentation.

Hybrid approaches enhance malware detection in IoT by merging network-based anomaly detection with lightweight static analysis, thus ensuring resource efficiency [9]. The integration of AI and deep learning, particularly models like CNNs, RNNs, and Transformers, has improved threat detection, allowing for effective analysis of binary code and network traffic to identify evasive malware.

Edge computing is gaining recognition for malware detection in IoT ecosystems by facilitating local threat analysis on edge devices, thus reducing latency and bandwidth use while lessening dependence on centralized cloud security [10]. To address the limited resources of edge devices, there is a need for optimized, lightweight detection models that ensure a balance between performance and energy efficiency advancements in behavioral analytics, federated learning, and threat intelligence sharing, facilitating proactive detection and response to cyber threats, thereby enhancing the resilience of networks against sophisticated cyber-attacks.

The evolution of malware detection has resulted in sophisticated methodologies, particularly hybrid approaches combining static analysis and network monitoring, which effectively address modern cyber threats [11]. This framework is especially relevant for detecting malware in IoT and cloud environments. The integration of AI, ML, and deep learning enhances the adaptability and proactivity of these systems against emerging threats. Continuous research and innovation are essential as cybercriminals develop advanced evasion techniques, emphasizing the need to protect digital infrastructures.

Project Simulation and Results:

The proposed malware detection system integrates static analysis and network monitoring to identify malware and suspicious activities through a three-phase approach: static analysis, network monitoring, and system integration, employing specialized tools and techniques from cybersecurity and network analysis [12].

Tools and Technologies:

Python: The system was implemented in Python due to its extensive cybersecurity libraries, including YARA and Scapy, and its strong ecosystem for machine learning integration, wide libraries (YARA, Scapy), and strong community support.

YARA: Detects malware using custom pattern-matching rules based on suspicious functions, file behavior, and network patterns.
Scapy: Captures and analyzes network packets in real time to detect unusual or malicious traffic [13].
VS Code: IDE used for coding, debugging, and version control with Git.
Static Analysis Approach:
Inspects files **without executing them**.
Uses **YARA rules** to identify suspicious patterns like eval, file modifications, or network calls.
Files are scanned, matched with rules, and flagged if threats are found.
Network Monitoring Approach:
Uses **Scapy** to monitor live network traffic.
Compares packet IPs against a blacklist of known malicious addresses.
Detects abnormal connections or data exfiltration attempts.
Real-Time Alert System:
Generates instant alerts when suspicious files or traffic are detected.
Helps users take immediate action against potential malware.
System Integration & Execution Flow
Combines static analysis and network monitoring in a single CLI tool.
Users can scan files (--file <path>) or monitor networks (--network).
Both modules can run together for comprehensive protection.
Results and alerts are displayed in real time [14].

This integrated approach ensures that both local and network-based threats are detected and addressed efficiently. Figure 2 illustrates a dual Research Methodology.

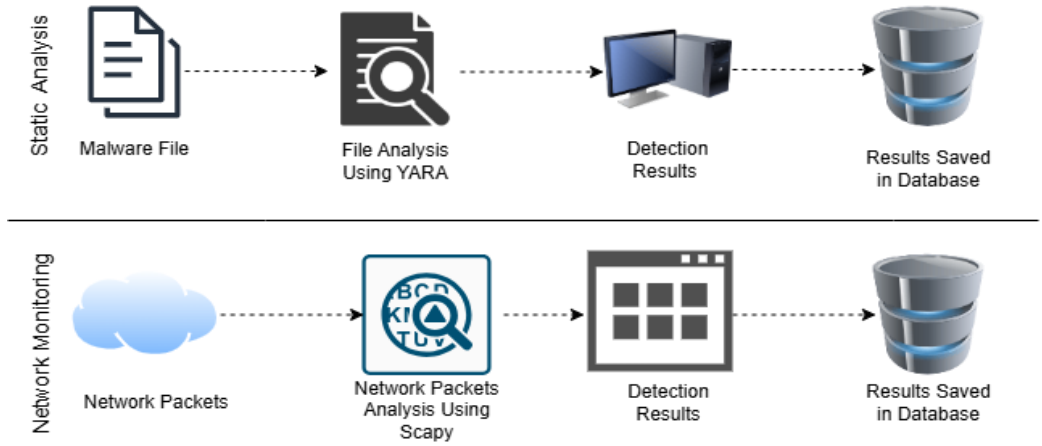

Figure 2. Dual Research Methodology for Hybrid Malware Detection

Figure 3 illustrates a dual Research Methodology for malware detection: Static Analysis and Network Monitoring. Static Analysis employs YARA for examining malware files, with results saved in a database. Network Monitoring utilizes Scapy to analyze network packets for suspicious activity, also storing outcomes in a database. This integrated approach boosts detection accuracy through both signature-based file scanning and real-time network traffic analysis.

Results:

The system was tested in a controlled environment using a variety of malware samples and simulated network traffic to validate its ability to detect malicious files and suspicious activity.

Static Analysis Results: The static analysis component scanned a dataset of 100 executable files, 60 of which were verified malware samples, and 40 were benign files. The YARA rules

were designed to detect specific code patterns, strings, and behaviors commonly found in malware [15].

Table 1. Results of malware detection

File Type	Detected as Malicious	Not Detected	Detection Rate
Malware (60)	56	4	93.3%
Benign (40)	4 (False Positives)	36	-

True Positives: 56
False Negatives: 4
False Positives: 4
True Negatives: 36

The static analysis component achieved a detection accuracy of 92%, demonstrating strong capability in identifying known malware signatures. The false positive rate of 10% indicates that a small proportion of benign files were incorrectly flagged, primarily due to heuristic pattern overlaps. False negatives were mainly associated with obfuscated or polymorphic malware variants, highlighting areas for future improvement the few false negatives occurred primarily with obfuscated or polymorphic malware [16].

Figure 3 shows the overall, the document seems to be data-oriented, combining numerical tables with explanatory or descriptive text.

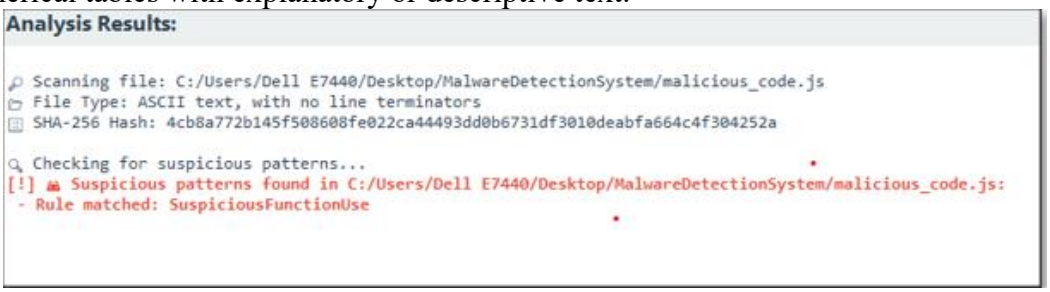

Figure 3. Static Analysis Results for Malicious File

Figure 4 illustrates the interface that displays the results of a static analysis performed on a suspicious JavaScript file (malicious_code.js). The system detects obfuscated code, suspicious function use, and PE file characteristics, suggesting potential malicious behavior. The static analysis results are presented in Figure 4**.**

Figure 4. Static Analysis Results for Benign File

Network Monitoring Results:

For dynamic testing, benign and malicious network traffic was captured and analyzed in real-time. Malicious traffic included communication with blacklisted IPs and abnormal data flows.

Table 2. Traffic Types Detection

Traffic Type	Detected Anomalies	Missed Threats	Detection Rate
Malicious Sessions	23 out of 25	2	92%
Benign Sessions	2 flagged (FP)	48	-

Total packets analyzed: 2000+
Alerts generated: 25
Real-time response time: <1 second per alert

Figure 5 this interface demonstrates that the malware detection system can successfully start and complete network monitoring operations via command-line execution.

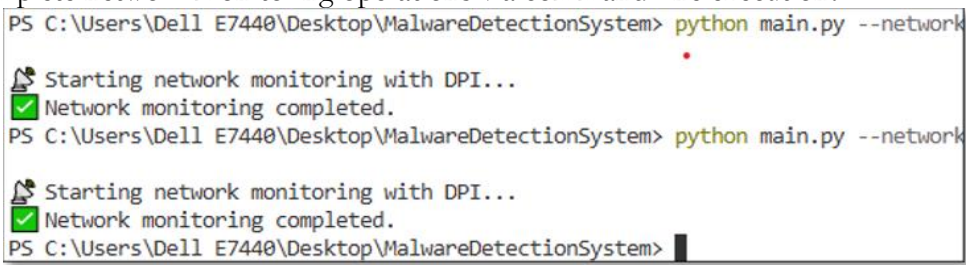

Figure 5. Network Monitoring Results with Deep Packet Inspection

Figure 5**:** The figure indicates that a "Malware Detection System" is being used to start and complete "network monitoring with DPI" (Deep Packet Inspection), suggesting a successful scan or analysis of network traffic. The command is run twice, and both times the output confirms the completion of the monitoring process. The network monitoring system effectively identified suspicious communication patterns, especially outbound traffic to known malicious IPs.

Discussion Section:

The results show that the proposed hybrid malware detection system effectively improves threat detection by combining static code analysis and network monitoring. The static analysis module using YARA achieved about 92% detection accuracy, successfully identifying most malware samples. However, a few false positives and false negatives occurred due to similarities between benign files and malicious patterns, as well as the presence of obfuscated malware.

The network monitoring module using Scapy also demonstrated strong performance, detecting 23 out of 25 malicious network sessions with a 92% detection rate and generating alerts in less than one second. This highlights the effectiveness of analyzing real-time network traffic to identify suspicious communication patterns that static analysis alone might miss.

By integrating both approaches, the system provides a multi-layered malware detection strategy, improving overall accuracy and reliability. Static analysis detects malicious code in files, while network monitoring captures abnormal runtime behaviors. Despite its effectiveness, the system still faces challenges in detecting encrypted traffic and highly sophisticated polymorphic malware. Future work can improve the framework by integrating advanced machine learning models and behavioral analysis techniques to enhance detection capabilities against evolving cyber threats.

Conclusion:

This study presented a hybrid malware detection framework that integrates static code analysis with real-time network monitoring to enhance the identification of modern cyber threats. By leveraging YARA for signature-based malware detection and Scapy for live network traffic inspection, the proposed system effectively detects both file-based malicious code and suspicious communication patterns. The experimental results demonstrate strong detection performance, achieving high accuracy while maintaining a rapid response time. In addition to its technical contribution, the system serves as an educational platform for understanding malware behavior and core cybersecurity concepts, making it valuable for academic and training environments. Despite its effectiveness, certain limitations remain. The framework does not currently analyze encrypted traffic, which may conceal malicious communication, and it faces challenges in detecting highly sophisticated polymorphic or heavily obfuscated malware variants. Addressing these limitations through advanced behavioral analysis and machine learning enhancements represents an important direction for future research. Overall, the findings reinforce the importance of a multi-layered defense strategy that combines static

and dynamic analysis techniques to combat the continuously evolving landscape of cyber threats.

References:

[1] Zahid Akhtar, "Malware Detection and Analysis: Challenges and Research Opportunities," *arXiv:2101.08429*, 2021, [Online]. Available: https://arxiv.org/abs/2101.08429

[2] Azar Abid Salih, Maiwan Bahjat Abdulrazzaq, "Cyber security: performance analysis and challenges for cyber attacks detection," *Indones. J. Electr. Eng. Comput. Sci.*, 2023, [Online]. Available: https://ijeecs.iaescore.com/index.php/IJEECS/article/view/30893

[3] S. Rana and R. Chicone, "Fortifying the Future: Harnessing AI for Transformative Cybersecurity Training," *Fortifying Futur. Harnessing AI Transform. Cybersecurity Train.*, pp. 1–131, Jan. 2024, doi: 10.1007/978-3-031-81780-9.

[4] R. H. Mahdi and H. Trabelsi, "Detection of Malware by Using YARA Rules," *2024 21st Int. Multi-Conference Syst. Signals Devices, SSD 2024*, pp. 568–575, 2024, doi: 10.1109/SSD61670.2024.10549308.

[5] Haocong Li, Jie Li, "Automatically Generate Malware Detection Rules By Extracting Risk Information," *ACM Int. Conf. Proceeding Ser.*, 2024, [Online]. Available: https://dl.acm.org/doi/10.1145/3670105.3670209

[6] M. A. Khalifa, I. Almomani, and W. El-Shafai, "SAMA: A Comprehensive Smart Automated Malware Analyzer Empowered by ChatGPT Integration," *2024 IEEE 30th Int. Conf. Telecommun. ICT 2024*, 2024, doi: 10.1109/ICT62760.2024.10606026.

[7] H. T. Zaw, T. Aung, A. H. Maw, and M. Thida Mon, "Comparative Analysis of YARA and VirusTotal Integrations with Wazuh for Enhanced Malware Detection," *2024 5th Int. Conf. Adv. Inf. Technol. ICAIT 2024*, 2024, doi: 10.1109/ICAIT65209.2024.10754931.

[8] "Malware Analysis: Digital Forensics, Cybersecurity, And Incident Response: Botwright, Rob: 9781839385315: Amazon.com: Books." Accessed: Feb. 07, 2026. [Online]. Available: https://www.amazon.com/Malware-Analysis-Forensics-Cybersecurity-Incident/dp/1839385316

[9] Pooja Kumari, Ankit Kumar Jain, "A comprehensive study of DDoS attacks over IoT network and their countermeasures," *Comput. Secur.*, vol. 127, p. 103096, 2023, [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0167404823000068

[10] Anshuman Singh, Brij B. Gupta, "Distributed Denial-of-Service (DDoS) Attacks and Defense Mechanisms in Various Web-Enabled Computing Platforms: Issues, Challenges, and Future Research Directions," *Int. J. Semant. Web Inf. Syst.*, vol. 18, no. 1, p. 43, 2022, [Online]. Available: https://www.igi-global.com/article/distributed-denial-of-service-ddos-attacks-and-defense-mechanisms-in-various-web-enabled-computing-platforms/297143

[11] B. A. Dahri, K. A., Vighio, M. S., & Zardari, "Detection and prevention of malware in the Android operating system," *Mehran Univ. Res. J. Eng. Technol.*, vol. 40, no. 4, pp. 847–859, 2021, [Online]. Available: https://www.researchgate.net/publication/355269909_Detection_and_Prevention_of_Malware_in_Android_Operating_System

[12] Ö. A. Aslan, R. Samet, "A Comprehensive Review on Malware Detection Approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020, [Online]. Available: https://ieeexplore.ieee.org/document/8949524

[13] Pedro LocoSebastian AlonsoGeorge HartmannJames WhitmoreEdward McLaughlin, "Adaptive Behavior-Based Ransomware Detection via Dynamic Flow Signatures,"

Sciety, 2024, [Online]. Available: https://sciety.org/articles/activity/10.21203/rs.3.rs-5317374/v1

[14] I. Mršulja, J. Slivka, and G. Sladić, "Obfuscated Malware Classification Using Hierarchy-Based Pipeline," *Lect. Notes Networks Syst.*, vol. 860 LNNS, pp. 401–409, 2024, doi: 10.1007/978-3-031-71419-1_34.

[15] M. N. Ahmed, Z. Iqbal, H. Mahmood, and M. Abdullah, "CAP - A Context-Aware Framework for Detection and Analysis of Packed Malware," *2024 Int. Conf. Front. Inf. Technol. FIT 2024*, 2024, doi: 10.1109/FIT63703.2024.10838408.

[16] Kenan Begovic, Abdulaziz Al-Ali, Qutaibah Malluhi, "Cryptographic ransomware encryption detection: Survey," *Comput. Secur.*, vol. 1321, p. 103349, 2023, [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404823002596