Enhancing AI Malware Detection Using Neural Network with Binary Data Analysis

Muhammad Sufi Afifi Abdul Sakti[1], Mohsen Mohamad Hata[2*], Zulaikha Hanan Bolhan[3] and MohamadYusof Darus[4]

[1, 2*, 3, 4] College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia
mohsen@uitm.edu.my

Abstract: Recent advancements in the realm of software have brought forth evolving risks that impact both individuals and organizations. The emergence of AI powered malware with its growing stealth, adaptability and harmful capabilities presents obstacles to cybersecurity defenses. This type of malware is adept at changing its behavior imitating actions and slipping past antivirus tools unnoticed. Ongoing efforts in malware research are centered on harnessing AI technologies to bolster the detection and categorization of malware especially when dealing with novel or unfamiliar threats. Alongside these strategies are being developed to fortify defenses against cyber-attacks driven by AI emphasizing the importance of enhancing the security and resilience of AI systems against threats. Tackling the challenges posed by AI driven malware necessitates exploration and innovative approaches to enhance the detection, prevention and overall security of AI systems. Our research introduces a technique for improving the detection of AI driven malware using networks coupled with binary data analysis. By utilizing AI technology, this technique aims to enhance detection accuracy and efficiency. Unlike dynamic analyses, binary analysis enables a comprehensive examination by leveraging all information from binary data while maintaining vital details through compression. The technique involves gathering data, developing models and conducting testing to assess the performance of the malware system. This technique tackles the drawbacks of antivirus programs by combining cutting edge networks with binary data examination. The results showcase the efficiency of this method in identifying and categorizing software, underscoring its ability to bolster cybersecurity measures, against cyber threats.

Keywords: Neural – Network, Anti Malware, Malware Detection, Threats

1 Introduction

The advancement of artificial intelligence (AI) has largely affected cybersecurity, technically, with the emergence of AI-driven malware. These threats imitate normal activities using machine learning, rendering them impossible to detect. Advanced techniques were developed as countermeasures to the threats such as deep learning, which also uses artificial neural networks and genetic algorithms to allow continuous adaptation to new alterations, and more complicated detection techniques. (Wolsey, 2022; Savenko, 2017). Such advanced malware poses significant challenges for cybersecurity professionals as they face increasingly sophisticated AI-driven cyberattacks. (Guembea & Azeta, 2022).

Static analysis normally works by hashing the infected files and comparing its unique hash output with other hash results. It is faster detecting malware than dynamic analysis or hybrid analysis or binary analysis, but the hashing mechanism itself can partially disrupt or even fully dismantle the significant data that is important to understand the underlying processes which is crucial if it had a built-in AI capability. (Damodaran & Di Troia, 2022). All malware, whether AI-driven or not, must have their processes written into their binary or source code, this can be used to deduce its nature. With AI-powered malware, the stealth trait can allow no detection by antiviruses, however the threat actor can execute a simple instruction such as an application programming interface (API) request call to an AI provider which generates the malicious payload encoded in the source code, which then executed during runtime. The static analysis would not be able to detect such pattern if the significant data were destroyed in the process

Binary analysis is a type of analysis on the binary itself, it requires the binary to be decompiled and disassembled to get the important trace on how it will be executed. The result of the binary analysis is a nearly perfect replica of the source code in low level language. A neural network can be trained to understand the binary analysis result in detecting malware since natural language processing is already currently used to translating and understand languages. The neural network that could be used is the transformer-based model such as the popular BeRT and GPT architecture, this can only further improve threat identification and classification (Rahali & Akhloufi, 2021). Deep learning neural networks have significant advantages with binary analysis since it enhances effectiveness by detecting the pattern and forming relational context even in fragmented files. (Wen & Chow, 2021).

Our technique significantly improves detection accuracy against AI-powered malware, minimizing false negatives where static analysis can not even detects it due the malware stealth feature. To demonstrate its efficacy, outputs from comprehensive binary analysis were trained using a bidirectional transformer (BeRT) neural network model. Results compared to those from BeRT, GPT, and CNN models with segmented binary analysis and result from the alternative analysis with the similar models. The dataset includes diverse AI-powered malware variants from the same family. The study achieved up to 90% accuracy in classifying AI-capable malware, underscoring the proposed method's potential to enhance cybersecurity defenses against sophisticated threats.

2 Literature Review

2.1 Malware

2.1.1 Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search

State-of-the-art AI malware is now being developed using machine learning, which trains models to identify patterns within data, making malware more challenging to identify. Deep learning, a subset of machine learning using artificial neural networks, is also can be used to create even more elusive malware. Additionally, malware authors can also employ genetic algorithms, a form of AI that constantly enables malware to evolve stated in the paper "Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search" (G. Savenko S. Lysenko A. Nicheporuk B. Savenko, 2017).

The paper introduces a new technique for detecting metamorphic viruses by searching for equivalent functional blocks. This method addresses the challenges posed by obfuscation techniques like block reordering. It involves two stages which is the first, identifying equivalent functional blocks based on statistical evaluation of instruction appearances, and second, refining the choice of these blocks to form a feature vector of similarity. The technique uses fuzzy logic for classification, reducing false positives compared to previous methods. The approach is validated through experiments, showing high detection efficiency and low false positive rates, even with obfuscated metamorphic viruses.

2.1.2 A Review of The Emerging Threat of AI-driven Cyber-attacks

The paper "A Review of The Emerging Threat of AI-driven Cyber-attacks" (Guembe B. Azeta,A. Misra S. Osamor V. C. Fernandez-Sanz L. Pospelova V., 2022), shows Offensive AI strategies enable rapid, large-scale assaults that evade detection. Cyberattacks pose a significant global threat, causing financial losses and disruption. In the US, data breaches cost an average of $8.19 million each, contributing to a $400 billion annual impact worldwide. These attacks exploit computer systems, challenging traditional cybersecurity measures. As attacks become more sophisticated, cybersecurity specialists face unprecedented challenges. Existing tools struggle to keep up. To combat these threats, proactive and innovative cybersecurity strategies are essential. Continual adaptation is crucial to safeguard digital environments against AI-driven attacks.

The paper discusses the rising threat of AI-driven cyberattacks. It highlights how cybercriminals are increasingly using AI-based techniques to evade detection and cause significant damage. Commonly used are the API based code payload.

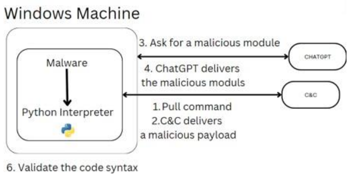

Windows Machine
Malware
Python Interpreter
3. Ask for a malicious module
4. ChatGPT delivers the malicious moduls
CHATGPT
C&C
1.Pull command
2.C&C delivers a malicious payload
6. Validate the code syntax

Fig 1. API Based Code Payload Process Flow

The figure above shown that the malware executed an API call instruction during runtime, sending a request to an API server provider such as ChatGPT which hosts prompt generation service, the server response in form of executable codes, often malicious syntax, which when received, the malware executes the payload. This allows for the malware to go undetectable, and the scope of the potential damages are immeasurable.

2.2 Binary Analysis

2.2.1 A Comparison of Static, Dynamic, Hybrid, Binary Analysis

The paper "Methods for Detecting Malware Using Static, Dynamic and Hybrid Analysis" (Alexandru-Radu BELEA, 2023) reveals that fully dynamic analysis, which monitors API calls during malware execution, is the most effective method for malware detection, offering comprehensive real-time insights and higher accuracy. In contrast, hybrid analysis, which combines static and dynamic methods, did not consistently enhance detection rates. This indicates that merely integrating multiple techniques isn't automatically beneficial. Effective hybrid strategies require strategic planning to leverage each method's strengths. Additionally, binary analysis, which involves examining the binary code of software, provides valuable insights but may lack the real-time behavioral context offered by dynamic analysis.

The paper compares static, dynamic, and hybrid analysis techniques for malware detection. It uses Hidden Markov Models (HMMs) to train on both static and dynamic feature sets, evaluating detection rates across various malware families.

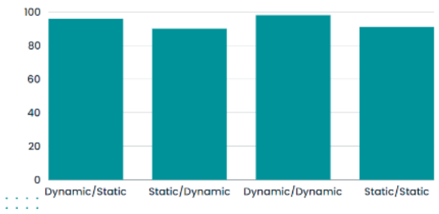


Fig 2. Static-Dynamic-Hybrid Analysis Comparison

The figure above shows that a fully dynamic approach generally yields the best detection rates at 98 percent compared to other approaches. The research aims to understand the advantages and disadvantages of each technique, particularly focusing on whether hybrid approaches offer any inherent benefits. The paper is organized into sections discussing background information, experiments, results, and futurework.

2.2.2 A Review of Deep Learning-Based Binary Code Similarity Analysis

The paper "A Review of Deep Learning-Based Binary Code Similarity Analysis" (Du, J. Wei, Q. Wang, Y. Sun, X, 2022) proposes a method to detect fragments of Windows Portable Executable (PE) files, crucial for handling diverse fragment lengths using deep learning which mimicking human brain mechanisms, emerges as a promising approach for analyzing raw binary inputs, offering potential solutions to the challenges posed by small malware fragments lacking distinct feature extraction points. Binary analysis is a type of code review that examines files composed of binary code. It assesses the content and structure of files directly in their binary form which helps understand and evaluate binary files structure without relying on the original source code.

The paper reviews deep learning-based binary code similarity analysis (BCSA) research papers, highlighting its importance in software engineering for tasks like bug search, code clone detection, and patch analysis. Traditional antivirus tools predominantly rely on signature-based detection, employing pattern matching against a database of malware signatures. While effective against known threats, this approach fails with new or unknown viruses. To address this, AI-based methods aim to uncover underlying patterns in malicious code using machine learning, though modern malware employs advanced evasion tactics like file fragmentation and self-destruction post-attack.

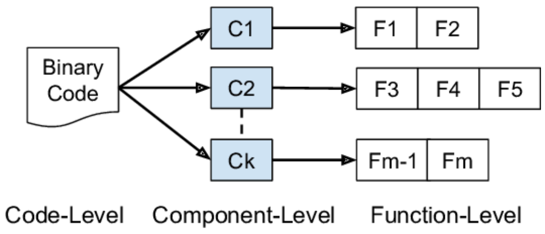


Fig 3. Binary Code Analysis Overview

The figure above shows how binary code analysis processed binary data. The binary data were separated into components where the entities further define into functions. For every function such as API request or method invocation are coded into the binary, it is possible to retrace the function by analyzing the binary.

2.3 Neural Network

2.3.1 MALBERT: Using Transformers for Cybersecurity and Malicious Software Detection

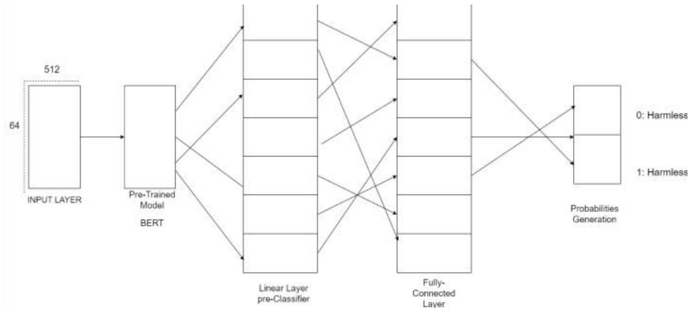


Fig 4. Bert Architecture

In the paper “MALBERT: Using Transformers for Cybersecurity and Malicious Software Detection” (Rahali A. Akhloufi M. A, 2021) proposed using a Transformer-based algorithm for malware detection by experimenting with different model architectures on a dataset containing 11 malware categories. The technique focuses on static analysis of Android app source code, considering the entire code as feature representation. It can perform binary classification (malicious vs. benign) and cross-category classification at the malware level. Recent research leverages Transformer-based models like BERT and XLNet. These models excel in natural language processing and computer vision tasks due to attention mechanisms.

The paper discusses using Transformer-based models, specifically BERT, for detecting malicious software in Android applications. It highlights the increasing cyber threats and the need for automated malware detection. The proposed method involves static analysis of Android app source code, using features like permissions and activities to classify malware. The study shows that Transformer models, particularly BERT, outperform traditional methods like RNNs and CNNs in both binary and multi-class malware detection tasks. The results indicate high accuracy and effectiveness, suggesting that Transformer-based approaches are promising for cybersecurity applications.

3 Methodology

To handle this challenge, we introduce our technique to improve malware detection with the help of transformer-based neural networks and binary analysis. The process flows through the gathering and dissembling of binary data, capturing its behavior in a controlled environment to extract a vital pattern. These patterns are converted into strings and analyzed through pre-trained language models that are fine-tuned for malware detection. It deals with imbalanced data, continuous learning to adapt to new threats, explains transparency in decision-making, and provides real-time monitoring for ongoing improvement. Our approach contains 8 detailed steps.

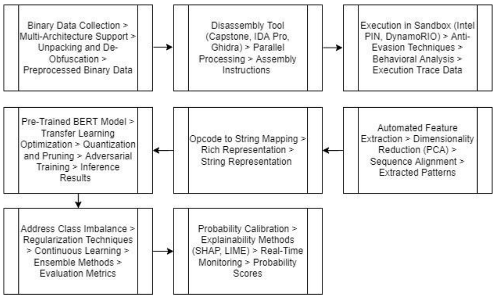

Binary Data Collection > Multi-Architecture Support > Unpacking and De-Obfuscation > Preprocessed Binary Data
Disassembly Tool (Capstone, IDA Pro, Ghidra) > Parallel Processing > Assembly Instructions
Execution in Sandbox (Intel PIN, DynamoRIO) > Anti-Evasion Techniques > Behavioral Analysis > Execution Trace Data
Automated Feature Extraction > Dimensionality Reduction (PCA) > Sequence Alignment > Extracted Patterns
Opcode to String Mapping > Rich Representation > String Representation
Pre-Trained BERT Model > Transfer Learning Optimization > Quantization and Pruning > Adversarial Training > Inference Results
Address Class Imbalance > Regularization Techniques > Continuous Learning > Ensemble Methods > Evaluation Metrics
Probability Calibration > Explainability Methods (SHAP, LIME) > Real-Time Monitoring > Probability Scores

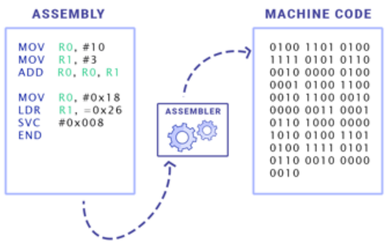
Fig 5. Methodology of the Proposed Technique

3.1 Binary Data Analysis Preprocess

The initial stage involves collecting binary executable files from various sources. The technique should cover support to myriads of architectures (x86, x64, ARM, etc.) and in later stages requires developers to employ automated data collection systems to update the dataset continuously. Preprocessing steps include unpacking and de-obfuscation to handle packed and obfuscated binaries, ensuring accurate analysis in subsequent stages. Disassembly converts binary data into human-readable assembly instructions.

Fig 6. Myriads of the Architecture

The low-level language such as Assembly were translated to the corresponding machine code or binary. Therefore, the technique uses powerful tools such as Python ObjDump library for reverse disassembling and decompiling the binary. To enhance performance, especially for large binaries, parallel processing can be implemented. The disassembly phase produces a sequence of assembly instructions that form the basis for further analysis. The result of the reverse engineered binary should look like a segment of Assembly language.



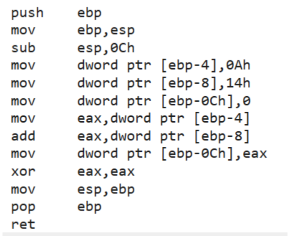
```
MOV   R0, #10
MOV   R1, #3
ADD   R0, R0, R1

MOV   R0, #0x18
LDR   R1, =0x26
SVC   #0x008
END
```

```
0100 1101 0100
1111 0101 0110
0010 0000 0100
0001 0100 1100
0010 1100 0010
0000 0011 0001
0110 1000 0000
1010 0100 1101
0100 1111 0101
0110 0010 0000
0010
```

Fig 7. WebAssembly Programming Language

3.2 Tracing Execution Flow

Capturing an execution trace from assembly instructions derived from a binary requires building a detailed Control Flow Graph (CFG), conducting static path analysis, and simulating the program's execution flow using these instructions. This approach facilitates a deeper understanding of program behavior, enabling insights into its operation. Despite inherent limitations, static techniques are instrumental in revealing critical insights about program behavior and detecting potential security vulnerabilities within binary code. These steps should help lessen the burden on the neural network by providing more context.



```
push    ebp
mov     ebp,esp
sub     esp,0Ch
mov     dword ptr [ebp-4],0Ah
mov     dword ptr [ebp-8],14h
mov     dword ptr [ebp-0Ch],0
mov     eax,dword ptr [ebp-4]
add     eax,dword ptr [ebp-8]
mov     dword ptr [ebp-0Ch],eax
xor     eax,eax
mov     esp,ebp
pop     ebp
ret
```

Fig 8. Assembly Language Data

3.3 Pattern Extraction and Text Presentation

Feature extraction is a process that involves converting raw data into a more manageable and informative format to analysts by either using methods like PCA, which can retain the original variance for dimensionality reduction, or through other ways. Herein, key features may be extracted from rather complex data to allow ease in carrying out machine learning and data analysis.

These patterns are then mapped to their string representations through low-level programming language mapping, which retains contextual information from the binary instructions. These strings have captured the flow in binary data using advanced embedding techniques, hence making them suitable for neural network processing.

Neural networks can be trained to identify AI-related patterns in the assembly representation of binaries, irrespective of instruction set architecture, to detect a wide range of malware threats rapidly and accurately for their AI capabilities.

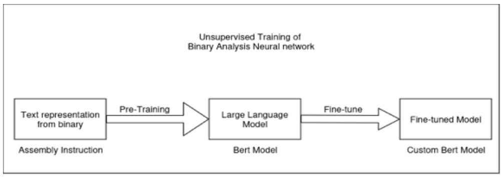


Fig 9. Unsupervised Training of Binary Analysis Neural Network

3.4 Vector Embedding and Neural Network

A neural-network model, from natural language processing, such as Bidirectional Encoder Representations from Transformers (BERT), Text-to-Text-Transfer-Transformer (T5) which pre-trained with common languages. It needs to go through a process called fine-tuning, where the model undergoes the same process of pre-training but with different datasets which a large corpus of assembly instructions and execution traces, is employed for inference.
The model benefits from transfer learning, reducing the need for extensive retraining on new datasets. Optimization techniques, including quantization and pruning, ensure the model operates efficiently in real-time environments. Adversarial training is used to improve the model's robustness against evasion tactics.

3.5 Neural Network Inference and Identification Result

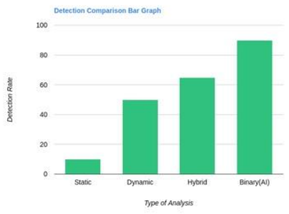


Fig 10. Type of Analysis Result

Finally, the results of the model's inferences would label binaries as malicious or benign. In the case of a class imbalance, data augmentation and regularization techniques have been used to avoid overfitting, enhancing generalization. The test results using modified BlackMamba keylogger malware provide higher detection rates upon performing AI-powered binary analysis as compared to other methods, although this was limited to only a single malware type in this case study.

4 Result and Findings

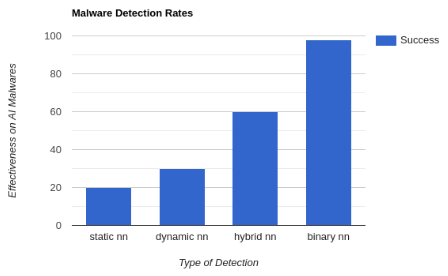


Fig 11. Type of Detection Result

This research article discussed how binary data analysis can enhance detection rates while reducing false positives using 100 variants of a modified BlackMamba malware family. The dataset is tested against a neural network model with traditional methods of detection. The following figure compares conventional and AI-driven detection with our proposed approach with respect to improvements in malware identification.

The results showed the neural network with binary analysis had the best detection rate of 90, outperforming the other methods. There was a variety of malware samples that allowed achieving broad detections, especially for broad-spectrum analysis. Accuracy was changing over time, and it was established then that while the method enhances detection, ensuring the performance under different conditions remains a challenge.

5 Conclusion

Deep learning malware and cyberattacks push cybersecurity beyond the breaking point. Malware powered by deep learning and genetic algorithms outpaces traditional security controls, creating serious bottom-line effects on finances and operation. Breaches of valuable data in extremely costly ways force cybersecurity experts to keep innovating against rapidly changing threats.

In our experiment, we infected 100 modified BlackMamba malware variants with a neural network model. The results showed that the neural network with binary data analysis outperforms static, dynamic, and hybrid techniques in terms of detection rate. However, variability in the accuracy of the models suggests further refinement is needed to establish consistency in performances across malware samples.

It is particularly effective with deep learning for the detection of malicious code fragments in AI-powered binary analysis. For malware classification, neural networks such as BERT and XLNet will definitely add much richness to fast and effective malware classification. We also propose a novel approach combining transformer-based neural networks with binary data analysis that offers better accuracy and effectiveness, hence holding much potential to reinforce cybersecurity defenses against complex threats.

References

1. N. M. Alamri, "A Review on Neural Network and Deep Learning," International Jour- nal of Current Engineering and Technology, vol. 10, no. 05, pp. 734–739, 2020. doi: 10.14741/ijcet/v.10.5.7.
2. J. Devlin, "BERT: Pre-training of Deep Bidirectional Transformers for Language Un- derstanding," 2018. doi: 10.48550/arXiv.1810.04805.
3. G. Savenko, S. Lysenko, A. Nicheporuk, and B. Savenko, "Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search," ICTERI, pp. 555–568, 2017. Available: http://ceur-ws.org/Vol-1844/10000555.pdf.
4. Z. Zhang, P. Qi, and W. Wang, "Dynamic Malware Analysis with Feature Engineer- ing and Feature Learning," Proceedings of the AAAI Conference on Artificial Intel- ligence, vol. 34, no. 1, pp. 1210–1217, 2020. doi: 10.1609/aaai.v34i01.5474.
5. BELEA, Alexandru-Radu. "Methods for Detecting Malware Using Static, Dynamic and Hybrid Analysis." Proceedings of the International Conference on Cybersecurity and Cybercrime-2023. Asociatia Romana pentru Asigurarea Securitatii Informatiei, 2023.
6. Rahali, A., & Akhloufi, M. A. (2021). MalBERT: Malware Detection using Bidirectional Encoder Representations from Transformers. 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). https://doi.org/10.1109/smc52423.2021.9659287.
7. Wen, Q., & Chow, K. (2021). CNN based zero-day malware detection using small binary segments. Forensic Science International. Digital Investigation, 38, 301128. https://doi.org/10.1016/j.fsidi.2021.301128.
8. Du, J., Wei, Q., Wang, Y., & Sun, X. (2023). A review of Deep Learning-Based Binary Code Similarity Analysis. Electronics, 12(22), 4671. https://doi.org/10.3390/electronics12224671.
9. Guembe, B., Azeta, A., Misra, S., Osamor, V. C., Fernandez-Sanz, L., & Pospelova, V. (2022). The emerging threat of AI-driven cyber-attacks: a review. Applied Artificial Intelligence, 36(1). https://doi.org/10.1080/08839514.2022.2037254.
10. Akhtar, M. S., & Feng, T. (2022). Malware analysis and detection using machine learning algorithms. Symmetry, 14(11), 2304. https://doi.org/10.3390/sym14112304.

Acknowledgments: The authors acknowledge the College of Computing, Informatics and Mathematics, Universiti Teknologi MARA for supporting this study.