A ML FRAMEWORK FOR ANALYZING BINARY CODE TO IDENTIFY POTENTIAL HACKERS

[1]HANI ABU SHARKH, [2]ASEM OMARI, [3]AZMI ALAZZAM

[1,2,3]Computer Information Systems Higher Colleges of Technology Al Ain, UAE
E-mail: [1]habusharkh@hct.ac.ae, [2]aomari@hct.ac.ae, [3]aalazzam@hct.ac.ae

Abstract - This paper aims to develop a comprehensive frame work for analyzing binary code to identify potential hackers. This framework will significantly enhance cybercrime investigations by tracing the origins of malicious software and pinpointing the cybercriminals responsible. The process involves converting binary code into readable formats, clustering similar codes, clustering different codes to filter the malicious codes, then analyzing them to uncover unique coding patterns that match known hacker profiles database. This is done by leveraging advanced techniques of sentiment analysis, and coding style examination. This research will provide law enforcement agencies with a powerful tool for tracking and prosecuting cybercriminals. Furthermore, the framework can serve as a preventive mechanism by revealing recurring vulnerabilities that attackers exploit. In doing so, it bridges the gap between traditional forensic approaches and modern machine learning-driven techniques, contributing to more resilient cyber security practices.

Keywords - Cybercrime, Malicious Codes, Clustering, Coding Patterns, Sentiment Analysis

I. INTRODUCTION

The exponential rise of cybercrime has presented significant challenges for investigators, governments, and organizations worldwide. Traditional cyber-forensic methods rely heavily on digital footprints, IP tracking, and signature-based detection systems. While these methods are effective in some contexts, they often fail when attackers employ advanced obfuscation, anonymization, and constantly evolving strategies to conceal their identities [1], [2]. As a result, attribution of cyber-attacks remains one of the most complex and unresolved issues in cyber security research.

In recent years, scholars and practitioners have begun exploring new dimensions of cybercrime investigation, including behavioral analysis and code profiling, as promising alternatives to conventional methods. The underlying idea is that, much like linguistic writing styles or personal hand writing, every programmer exhibits unique coding habits and stylistic traits that persist even when code is compiled into binary form. These subtle characteristics such as algorithmic choices, structural organization, and coding fingerprints offer untapped potential for tracing the origins of malicious software. Leveraging such attributes could significantly enhance the capacity of investigators to identify and prosecute cyber criminals.

Building on this foundation, the present research introduces an innovative framework that focuses on the stylistic analysis of binary code. The study seeks to apply sentiment analysis to group similar source codes into categories [3], identify those that are more likely to be associated with hacking activities, and examine them against a model that integrates unique coding patterns, algorithms, and coding fingerprints. By doing so, the research not only aims to improve the precision of cybercrime attribution but also to propose preventive measures derived from observed hacker coding patterns.

II. LITERATURE REVIEW

Recent research highlights the growing role of artificial intelligence (AI) and machine learning (ML) in enhancing cyber-attack detection and code analysis. Studies such as [1], [3] have shown the potential of combining kernel methods, sentiment analysis, and hybrid fuzzy-logic models to improve the identification of malicious code. In parallel, deep learning frameworks have demonstrated strong performance for intrusion detection in IoT and networked environments [2], [5].

Federated and distributed learning approaches have also emerged as effective techniques to enable malware detection across edge devices without compromising privacy [6]. To address the issue of interpretability, explainable AI (XAI) techniques such as SHAP and LIME have been applied to cyber-attack detection, enhancing transparency in decision making [7]. Block chain-enabled secure forensic data sharing [8] and adversarial training [9] have been further proposed to increase robustness and resilience against evolving cyber threats. Binary similarity detection using graph neural networks (GNNs) [10] illustrates the rise of deep representation learning in reverse engineering and code attribution.

Clustering and representation-based approaches are also central to recent developments. Improved DBSCAN clustering methods for malware families [11], deep embedding approaches for cross-architecture binary analysis [12], and survey-based reviews of insider threats and attack attribution methods [13], [16] reflect an emphasis on scalability and reliability. Additional works [14], [15] explore

semi supervised models, NLP-based script detection, auto encoder based anomaly detection, hybrid CNN-LSTM models, and transfer learning for constrained systems. Collectively, these contributions demonstrate a convergence of ML, cyber forensics, and binary analysis, establishing the foundation for this study's proposed framework.

III. METHODOLOGY

The methodology for this research is divided into six main steps, each crucial for the accurate identification of potential hackers through their coding styles in binary code. Figure 1 shows our proposed steps starting from data collection and binary code input to knowledge extraction and investing it to propose prevention measures from hackers.

A. Disassembling Binary Code and Decompiling Assembly Code

- Utilize industry-standard disassembly tools (such as IDA Pro,Ghidra,orRadare2)[17],[19] to convert binary code into assembly code. These tools break down the binary code into its fundamental machine instructions, making it more readable.
- Ensure accuracy and completeness in the disassembly process to retain all functional aspects of the original binary code.
- Apply sophisticated decompilers (such as Hex-Rays De-compiler [20]) to convert the assembly code into high-level source code.
- Maintain a high level of accuracy in the de-compilation process to ensure the source code is as close as possible to the original code written by the hacker, preserving coding styles and structural elements.

B. Clustering Source Codes

- Compare different NLP sentiment analysis models [3] (such as K-means, DBSCAN, or hierarchical clustering) to find the best in identifying potential malicious code.
- Potentially modify one of these algorithms or develop a new one.
- Use these algorithms to categorize source codes into groups, identifying those that exhibit unusual or suspicious patterns indicative of malicious intent.

C. Analyzing Suspicious Code

- Develop a matrix for analyzing suspicious code incorporating various factors such as:
  - Coding Styles and Fingerprints: Identify uniquecod-ingpatterns,indentationstyles,variablenamingcon-ventions, and other stylistic elements that can act as a fingerprint for the hacker.
  - Algorithms and Structures: Examine the algorithms used, data structures implemented, and the overall architecture of the code to identify distinctive approaches.
  - Procedural Steps and Logic Flow: Analyze the logical flow and procedural steps within the code to detect unique operational sequences.
- Implement machine learning to process the matrix and determine the likelihood of a match with known hacker profiles.

D. Identifying Potential Hackers

- Build and maintain a comprehensive database of known hacker coding styles compiled from past cyber crime cases and publicly available repositories.
- Use the specific matrix developed in the previous step to compare suspicious codes against the database.
- Apply similarity metrics and classification [4], [21], [22] techniques to match suspicious coding patterns with pro-files in the database, identifying potential hackers with a high degree of confidence.

E. Proposing Prevention Measures

- Conduct in depth analysis of identified hacker patterns to understand common vulnerabilities and exploits used in the code.
- Develop and propose targeted prevention measures such as:
  - Security Patches and Updates: Recommend specific patches and updates to address vulnerabilities exploited by identified hackers.
  - Code Hardening Techniques: Suggest best practices for writing secure code, including the use of secure coding standards and code obfuscation techniques.
  - Intrusion Detection Systems (IDS): Implement IDS with enhanced capabilities to detect coding patterns and anomalies identified during the research.
- Collaborate with cyber security professionals and organizations to implement these measures, providing training and resources to ensure widespread adoption.

IV. PROJECT CHALLENGES

A. Data Collection and Privacy Concerns

- **Challenge:** Gathering a comprehensive and representative dataset of hacker coding styles can be difficult due to privacy and ethical concerns.
- **Solution:** For research, there are several sources like Git Hub, Google Big Query Public Database, Kaggle, Open-source projects, and others.

B. Clustering Source Codes Effectively

- **Challenge:** Identifying the most effective algorithm for clustering source codes and potentially modifying it to better suit the task.
- **Solution:** Evaluate different clustering algorithms (e.g., K-means, DBSCAN, hierarchical

clustering) through rigorous testing. If necessary, develop a custom algorithm tailored to the specific requirements of the project.

C. Developing a Comprehensive Analysis Matrix

- **Challenge:** Creating a detailed matrix that accurately captures unique coding styles, algorithms, structures, and logic flows.
- **Solution:** Leverage existing research in code and extend it with additional features relevant to hacker identification. Regularly test and refine the matrix using real-world samples.

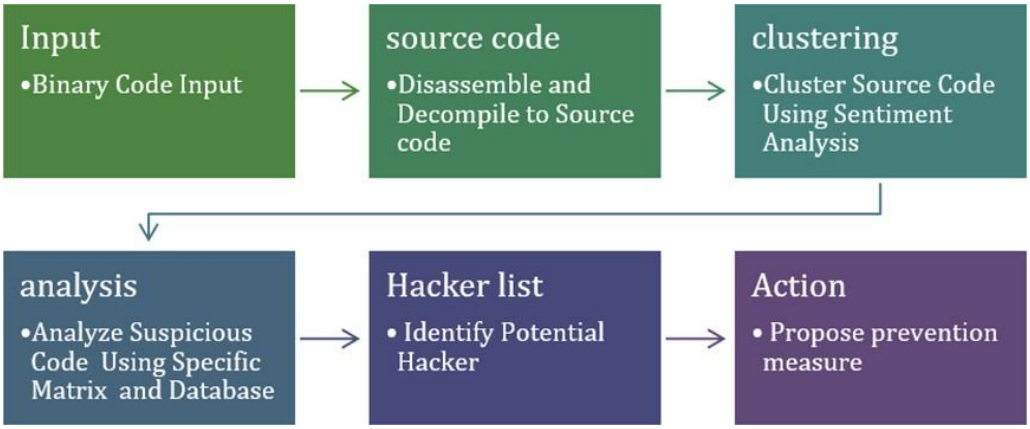


Fig.1: The frame work methodology

D. Ensuring Machine Learning Model Accuracy

- **Challenge:** Training machine learning models to accurately match suspicious codes with known hacker profiles without high false positives.
- **Solution:** Use large and diverse datasets for training. Implement cross-validation and continuous learning techniques to improve model accuracy and adaptability to new hacking techniques.

E. Resource and Time Management

- **Challenge:** Managing the extensive resources and time required for such a comprehensive project.
- **Solution:** Develop a detailed project plan with milestones and deadlines. Regularly review progress and adjust the plan as needed to stay on track.

V. CONCLUSION AND FUTURE WORK

This research project will establish a novel method for identifying hackers through the analysis of their coding styles in binary code. The proposed framework will enhance cybercrime investigations and improve overall cyber security practices by providing new tools and insights for preventing future attacks. This approach will significantly contribute to the field of cyber security and offer a robust solution to the growing problem of cybercrime.

ACKNOWLEDGMENT

The authors gratefully acknowledge the support of the Higher Colleges of Technology (HCT), United Arab Emirates, in facilitating this research. The resources, research environment, and institutional encouragement provided by HCT were instrumental in the successful completion of this work.

REFERENCES

[1] J. Gao, J. Li, H. Jiang, and Y. Li, "A novel cyber-attack detection approach based on kernel extreme learning machine using FR-conjugate gradient,"in*Proc.39thChineseControlConf.(CCC)*,Shenyang, China,2020, pp. 7637–7642.

[2] S. N. A, B. K, G. E, and M. I. C, "Deep learning for enhanced cyber-attack detection," in *Proc. Int. Conf. Cognitive Robotics and IntelligentSystems (ICC-ROBINS)*, Coimbatore, India, 2024, pp. 874–878.

[3] H.Qin, P.Xue, X.Ma, and B. Gao, "Code quality evaluation in IT communities based on sentiment analysis of user comments," in*Proc. IEEE 23rd Int. Conf. Software Quality, Reliability, and Security Companion (QRS-C)*, Chiang Mai, Thailand, 2023, pp. 423–429.

[4] K. Ananthi, M. Balakrishnan, M. Pandi, and N. S. Madasamy, "Bug recognition using hybrid fuzzy logic algorithm and SVM classification," in Proc. 4th Int. Conf. Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2022, pp. 84–88.

[5] Y.Wang,H.Yang,andJ.Chen,"Deeplearning-basedintrusiondetectionsystem for IoT," *IEEE Access*, vol. 8, pp. 219650–219662, 2020.

[6] T. Nguyen and M. Z. A. Bhuiyan, "Federated learning for malware detection in edge devices," *Future Generation Computer Systems*, vol.115, pp. 496–505, 2021.

[7] M. Rahman and R. Gupta, "Explainable AI for cyber-attack detection using SHAP and LIME," *Computers & Security*, vol. 127, p. 103084,2023.

[8] X. Zhou, J. Li, and H. Wang, "Blockchain-enabled secure data sharing for cyber forensics," *IEEE Internet of Things Journal*, vol. 9, no. 6, pp.4453–4465, 2022.

[9] Y. Sun, H. Liu, and P. Yu, "Adversarial learning for robust malware detection," in *Proc. 30th IJCAI*, 2021, pp. 3271–3277.

[10] A. Ali and K. P. Singh, "Binary code similarity detection using graphneural networks," *Neuro computing*, vol. 537, pp. 20–33, 2023.

[11] K. Patel and S. Gupta, "Clustering malware families using improved DBSCAN," *Future Generation Computer Systems*, vol. 128, pp. 39–51,2022.

[12] X. Zhang and Y. Zhou, "Cross-architecture binary analysis using deep learning embeddings," *IEEE Trans. Dependable Secure Comput.*, vol.20, no. 3, pp. 1745–1759, 2023.

[13] I. Ahmed and M. Alsmadi, "Machine learning approaches for

insider threat detection: A review," *Computers & Security*, vol. 96, p. 101935,2020.
[14] A. Baba and F. Kara, "Reverse engineering obfuscated binaries with ML-based deob fuscation," in *Proc. IEEE Symp. Security and Privacy Workshops*, 2021, pp. 122–128.
[15] R. Chaudhary and N. Sharma, "Deep autoencoders for anomaly-based intrusion detection," *Applied Soft Computing*, vol.135,p.110016,2023.
[16] P. Vermaand A. Joshi, "Survey of cyber-attack attribution methods," *ACM Comput. Surveys*, vol.54,no.7,pp.1–35,2021.
[17] C. Eagle, *The IDA Pro Book: The Unofficial Guide to the World's Most Popular Disassembler*. [Book].
[18] S.VanscheeuwijcK,'EvaluationofGhidraasareverseengineering tool,"Ph.D. dissertation, Univ. of Twente, 2020.
[19] L. Kovacs and P. Anderson, 'Evaluating the capabilities of Radare2 as a reverse engineering tool," *J. Cyber Security Technology*, 2018.
[20] Hex-Rays Decompiler project. [Online]. Available: https://hexrays.com/products/decompiler/manual/sdk/
[21] J. Mac Queen, 'Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Mathematical StatisticsandProbability,vol.1:Statistics*,1967,pp.281–297.Univ.ofCaliforniaPress.
[22] M.Ester,H.-P.Kriegel,J.Sander,andX.Xu,'Adensity-basedalgorithmfor discovering clusters in large spatial databases with noise," in *Proc.2nd Int. Conf. Knowledge Discovery and Data Mining (KDD-96)*, 1996, pp.226–231.AAAIPress.

★★★