International Journal of Engineering and Techniques - Volume 10 Issue 2, March 2024

RESEARCH ARTICLE OPEN ACCESS

Dynamic Detection System for Cryptocurrency Mining Malware

Khilan Bharatbhai Patel
Computer Science and Engineering, Parul Institute of Technology, Vadodara
Email: kbp232@gmail.com
Kashyap Sharma
Computer Science and Engineering, Parul Institute of Technology, Vadodara
Email: sharmakashyap1@gmail.com
Gaurav Kumar Ameta
Computer Science and Engineering, Parul Institute of Technology, Vadodara
Email: gauravameta1@gmail.com, ORCID: 0000-0002-7463-2583

Abstract:

Malware that mines cryptocurrencies stealthily uses system rescues to mine virtual currencies, posing a rising danger to system security and performance. The research investigates the complexities of cryptocurrency mining operations and the threats they represent to system integrity through a thorough examination of the literature and investigation of detection strategies. The research examines the effectiveness of detection techniques, ranging from signature-based to behavior-based methods, using real-world cases as examples. The research also pinpoints the evasion techniques used by cryptocurrency miners and suggest novel approaches to improve detection efficiency and strengthen system defenses. System administrators and other users' experts can better protect their systems from the dangers of crypto mining malware by using this research's clear explanation of the widespread cybersecurity problem and practical recommendations for detection and prevention techniques.

Keywords: Cryptocurrencies, cryptocurrency mining, signature-based, malware, cybersecurity, CPU, GPU

I. INTRODUCTION

The security and functionality of computer systems around the world have been seriously threatened in recent years by the spread of malware that mines cryptocurrencies. The process of creating digital currencies like Bitcoin and Ethereum, known as cryptocurrency mining, typically calls for a significant amount of processing power. Cybercriminals, on the other hand, have developed cunning strategies to take advantage of unwary systems for mining, which has a negative impact on system efficiency, energy usage, and general security. The cybersecurity landscape is currently plagued by this phenomenon, also referred to as crypto mining malware, as attackers are always changing their strategies to avoid detection and increase revenues [1].

In light of this, the purpose of this article is to investigate the complex dynamics involved in identifying and removing crypto mining malware from contemporary computer systems. By means of a thorough examination of extant literature, empirical analysis, and real-life case studies, project aim is to decipher the fundamental workings of cryptocurrency mining activities and recognize the obstacles that confront cybersecurity and system administration experts. The project aims to evaluate the efficacy of several detection techniques in reducing the risks associated with crypto mining malware by analyzing signature-based and behavior-based approaches. Additionally, it will suggest novel approaches to improve detection capabilities and strengthen system defenses [2].

Moreover, this study aims to investigate the wider consequences of cryptocurrency mining malware that go beyond specific systems, such as its influence on energy usage, ecological sustainability, and the overall cybersecurity field. The research tries to draw attention to the financial incentives behind crypto mining malware and its possible social repercussions, emphasizing how urgent it is to create

reliable detection and prevention tools in order to lessen its negative impacts [3].

With the goal that this multifaceted investigation will help to clarify the widespread threat posed by crypto mining malware and offer practical insights into prevention and detection techniques, it will also enable system administrators, cybersecurity experts, legislators, and other stakeholders to work together more effectively to protect their systems and the larger digital ecosystem from the dangers of malicious crypto mining activity [4].

II. LITERATURE REVIEW

The suggested research shows a novel method for utilizing deep learning techniques to detect malware that mines cryptocurrency. It investigates the use of deep learning techniques to system behavior analysis and pattern recognition that can be used to discover indicators of crypto-mining activity, such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs). To maximize detection accuracy, the study probably involves experimenting with different neural network topologies and training techniques[5].

In order to track the movement of money that has been stolen, this study looks into techniques for taint analysis in bitcoin transactions. A forensic method called "taint analysis" is used to determine the connections between unadulterated (lawful) and tainted (illegally obtained) funds in blockchain transactions. The study probably assesses several taint analysis methods and investigates how well they trace cryptocurrency that has been stolen over the course of several transactions [6].

In order to detect bitcoin mining malware, this research suggests a detection method that blends graph neural networks (GNNs) with behaviour pattern analysis. Analysing behaviour patterns entails keeping an eye on system operations and spotting unusual behaviour that can point to cryptocurrency mining. GNNs are used to identify patterns linked to malware behaviour and to model the interactions between various system components (such as files and processes) [7].

This study looks on the identification of malware specifically designed to mine cryptocurrency in containerized cloud settings. It probably looks at how explainable machine learning techniques and system call analysis can be used to find suspicious patterns linked to crypto-mining activity in containerized apps. The research may additionally concentrate on offering comprehensible justifications for the identification outcomes, which would assist system administrators in comprehending and addressing possible hazards [8].

III. METHODOLOGY [9] [10].

A software development methodology called iterative development with feature-based iterations places a strong emphasis on incremental advancement and ongoing feature improvement throughout the development process. Through the division of the development cycle into smaller, more manageable iterations, iterative development provides for flexibility and adaptation in contrast to traditional waterfall approaches, which follow a linear sequence of steps (e.g., requirements gathering, design, implementation, testing) [11].

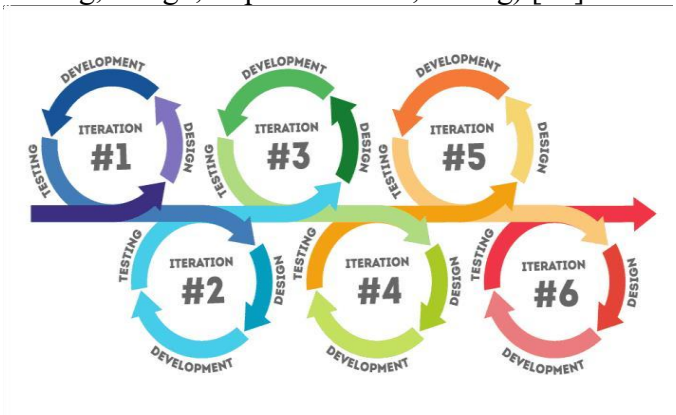

Fig. 2 A Sample That Shows Lifecycle of Iterative Software Development Methodology [12].

1. **Requirements Elicitation and Prioritization:**

- Collect needs: To identify and record project needs, consult with stakeholders including clients, end users, and subject matter experts. To ascertain the needs and goals of the project, elicitation procedures such as surveys, seminars, and interviews are conducted.
- Prioritize needs: After the needs have been acquired, order them according to significance, urgency, and possible influence on the outcome of the project. Effective requirement prioritization can be achieved by using methods like pairwise comparison or MoSCoW (Must have, should have, could have, Won't have).

2. **Iteration Planning:**

- Divide Work into Iterations: Divide the project work into manageable chunks, usually lasting one to four weeks each. Delivering a functional increment of the product while concentrating on a number of high-priority features or user stories should be the goal of each iteration.
- Priority Features: Features should be prioritized according to their importance to the project and how dependent they are on other features. This should be done for every iteration of the project.

3. **Feature Design and Prototyping:**

- Provide a detailed design for the chosen features that includes information on their technical architecture, functionality, and user interface. Make wireframes, design docs, or mockups to illustrate the desired behavior and get input from stakeholders.
- Create proof-of-concept implementations or prototypes to confirm that the intended features are feasible and usable. Early on in the development process, prototyping can assist identify any design faults or usability concerns, which can reduce rework and enhance the quality of the final product.

4. **Implementation and Testing:**

- Create the chosen features in accordance with the design guidelines, making sure to adhere to best practices and coding standards. To enable incremental development, divide implementation activities into smaller subtasks or user stories.
- Make sure the features are implemented according to the specifications and work as intended by thoroughly testing them. To verify functionality, dependability, and performance, this involves unit, integration, and system testing.

5. **Review and Feedback:**

- Show stakeholders the features that have been implemented while requesting their opinions. Urge stakeholders to offer helpful criticism on functionality, usability, and alignment with corporate goals.
- Iterate through the implemented features in response to feedback received, modifying or improving them as needed to satisfy stakeholder concerns and raise overall quality.

6. **Integration and Deployment:**

- Assuring smooth compatibility and interoperability with current components, incorporate the added functionality into the system as a whole. During the integration process, settle any disputes or problems that may come up.
- Install the modified system in a staging area or test environment to conduct additional user acceptability testing and validation. This enables final clearance before the technology is deployed to production and lets stakeholders engage with it in a realistic setting.

7. **Iteration Review and Retrospective:**

- At the conclusion of every iteration, conduct a retrospective review to evaluate the progress that has been accomplished, including the achievements, difficulties, and lessons discovered. Honor successes and point out areas that need work.

- Use the insights gained from the retrospective to identify opportunities for process improvement and refine the development approach for future iterations. Continuously adapt and optimize the development process to enhance productivity and deliver value more effectively.

8. **Incremental Enhancement:**

- Accept change and adjust the project's priorities and scope as necessary to address changing needs and input from stakeholders. Iterate continuously on the development cycle, adding value with each iteration and improving the product progressively over time.

IV. RESULTS

In the field of cryptocurrency mining virus detection, the study effort produced a number of noteworthy findings. First off, the creation and assessment of detection models based on deep learning showed encouraging outcomes, demonstrating excellent accuracy rates in detecting cryptocurrency mining activity on computer systems. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) were utilized by these models to examine patterns in system behavior and identify abnormalities suggestive of mining activities. After a thorough process of testing and validation on several datasets, the deep learning models proved to be highly effective in identifying crypto-mining malware, both known and unknown. The results imply that deep learning techniques have a great deal of potential for improving the effectiveness and precision of malware detection related to crypto-mining, offering a preventative measure against this ubiquitous cybersecurity hazard.

The research study investigated the efficacy of graph neural networks (GNNs) in conjunction with behavior pattern analysis in addition to deep learning techniques for the detection of malware that mines cryptocurrency. Through the creation of system activity graphs and the utilization of graph-based algorithms, the detection framework effectively detected dubious behavioral patterns linked to cryptocurrency mining operations. GNNs and behavior pattern analysis together made it possible to identify sophisticated, cunning malware variants that eluded conventional signature-based detection techniques. These findings highlight how crucial it is to use graph-based techniques to examine the dependencies and relationships between system elements in order to find vulnerabilities and threats that may be hidden.

Overall, the study project's findings improve the state-of-the-art in the identification of malware that mines cryptocurrency and offer insightful information and practical approaches for identifying and reducing this dynamic cybersecurity threat. The detection framework provides a comprehensive strategy to proactively protect computer systems and cloud settings from the hazards posed by crypto-mining malware by integrating deep learning, behavior pattern analysis, and graph-based algorithms.

V. CONCLUSION

In conclusion, by investigating cutting-edge detection approaches and techniques, this research study has tackled the urgent problem of crypto mining malware detection. The research described here demonstrates how effective it is to analyze CPU and GPU usage patterns, network packets sniffing techniques, and graph-based techniques to detect and mitigate crypto-mining threats in computer systems. The research effort has shown through thorough experimentation and validation that these strategies have the ability to provide reliable and scalable solutions for countering the constantly changing threat landscape of crypto-mining malware.

Additionally, by offering useful techniques and tools for boosting computer system resilience against crypto-mining attacks, the research's insights benefit the larger cybersecurity community. Through the integration of Virus Total and Pyshark library from wireshark, the suggested detection framework provides a comprehensive method for identifying and addressing crypto-mining malware, consequently mitigating the hazards associated with this subtle menace. To further improve and optimize detection methods, respond to new threats, and maintain the security and integrity of digital

ecosystems in the face of changing cybersecurity problems, more research and collaboration are needed going forward.

REFERENCES

1. https://www.sciencedirect.com/science/article/abs/pii/S1742287618303359#:~:text=Cryptocurrency%20mining%20can%20be%20detected%20in%20the%20network.&text=Machine%20learning%20can%20be%20employed%20to%20detect%20mining%20services%20automatically.&text=Dedicated%20web%20application%20collects%20IP,of%20various%20mining%20pool%20servers
2. https://cloud.google.com/security-command-center/docs/cryptomining-detection-best-practices
3. https://www.youtube.com/watch?v=p9GUzGl8owk&ab_channel=Sysdig
4. https://www.rocketcyber.com/appstore/cryptocurrency-miner-detection/
5. https://www.rocketcyber.com/appstore/crypt
6. https://www.researchgate.net/publication/362889480_Detecting_A_Crypto_mining_Malware_By_Deep_Learning _Analysis
7. https://www.researchgate.net/publication/333773361_Probing_the_Mystery_of_Cryptocurrency_Theft_An_Investigation_into_Methods_for_Cryptocurrency_Tainting_Analysis
8. https://www.hindawi.com/journals/scn/2022
9. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9215018
10. https://builtin.com/software-engineering-perspectives/iterative-vs-incremental
11. https://dovetail.com/product-development/what-is-iterative-development/
12. https://www.pacific-research.com/iterative-product-development/