International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT2390376

A Comprehensive Analysis of Software Obfuscation Techniques

Priyanka Kadam*, Hrishikesh Devgude, Srivaramangai R
Department of Information Technology, University of Mumbai, Mumbai, Maharashtra, India

ARTICLEINFO

Article History:

Accepted: 10 May 2023
Published: 30 May 2023

Publication Issue
Volume 9, Issue 3
May-June-2023

Page Number
314-320

ABSTRACT

In today's environment, a huge and increasing range of unauthorised practises allows hackers to get unauthorised access to devices and private information by inserting harmful code. The purpose of this research is to assess the efficiency of obfuscation strategies in safeguarding software against reverse engineering and tampering. This study tries to evaluate various obfuscation approaches, such as code obfuscation, control flow obfuscation, and data obfuscation. The findings of this study will contribute to the establishment of best practises for software developers to protect their code from unauthorised access and alteration by providing useful insights into the efficiency of obfuscation approaches in software protection.

Keywords: Unauthorized, Obfuscation, Reverse-Engineering, Tampering, Software Protection.

I. INTRODUCTION

The internet is a vast space for us to explore, create, and build. It is a virtual world that is different from the real world, yet very similar to it. As the real world involves people imitating each other, the digital world also includes people making pirated copies of online digital content. Piracy is copying work of a creator without permission and selling at marginal rates to the public through online or offline. Piracy of anything cause loss to owner, create adverse consequences like unemployment, lack of innovation, less creativity and many more. The pirated product is used without permission; therefore, it cuts revenue of company as well as tax of government. Using a good licencing system and obfuscation to safeguard your code is much needed. Obfuscation of software and data is one of the subcategories of software security. Software is frequently safeguarded against harmful reverse engineering using obfuscation techniques. Obfuscators alter the source code to make it more challenging for an attacker to understand and analyse. This means that it is done by changing the appearance of the source code and maintaining the functional nature of software. Obfuscation as an invasive technique can also be used as a defence solution in the field of software and vital information protection against security threats. It's harder for unauthorized third parties to gain insights into the internal workings of an application. Obfuscation changes the code and its data without

modifying the behaviour of the application or the user experience. It ranges from renaming classes or methods to transforming arithmetic or modifying the control flow of the app or encrypting app data. Following section describes the different obfuscation techniques.

A. Code Obfuscation

This technique modifies the structure and logic of the source code without affecting its functioning. It attempts to hide the code by renaming variables, methods, and classes with meaningless names, inserting dummy code, adding unnecessary comments, or changing the code flow.

B. String Encryption

This approach encrypts or encodes sensitive strings such as API keys, URLs, or configuration settings. When necessary, the encrypted strings are decoded at runtime. This makes it more difficult for attackers to extract significant information by statically analysing the code.

C. Control Flow Obfuscation

This approach alters the program's control flow by introducing additional conditional statements, loops, or jumps that make it more difficult to understand the execution path.

D. Key hiding obfuscation

This technique is used to protect sensitive information by hiding critical data within the code. The primary goal of this strategy is to make it harder for attackers to discover and extract keys or essential information from the code.

These are just four obfuscation techniques that can help protect your software/application against malicious actors. To maximize your protection against decompilers or disassemblers, it is best to implement multiple, advanced techniques.

II. CURRENT OBFUSCATION TECHNIQUES

All the research in this domain speaks about the techniques and algorithms for software protection and how it can be enhanced by using best practices. The paper discusses about COOPS, a control flow transformation-based code obfuscation solution that successfully protects the program control flow, hybrid obfuscation techniques and analysed the effects of various obfuscation techniques. Some of the techniques have limitation which are advised with a finding to resolved it.

A. Code Obfuscation Method Based on Obscuring Program Semantics (COOPS) [1]

This technique is used for protecting software based on program semantics information. The original semantic level of the program is destroyed by establishing a switch relationship between intra-function control flow and inter-function calling. Control flow transformation method used are inlining and outlining which can effectively change the flow structure of program. This technique is developed to achieve automation obfuscation program and uses source code written in C/C++ as the input and output the obfuscated binary file. The COOPs Obfuscation Technique runs on following three phases explained through Figure 1.

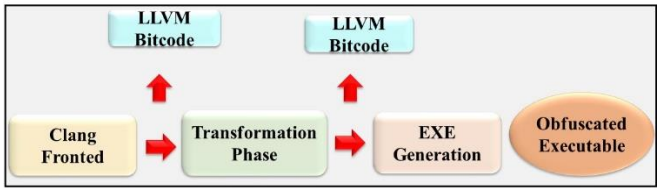

Fig.1 Working of COOPs Obfuscation Technique

The relative effectiveness was evaluated on OpenSSL and SpecInt-2000 test sets shows that difference in the calling function before and after obfuscation differ more than 90%. This suggest that COOPS effectively

alter the control flow of the program, making it harder to understand and analyse.

B. Hybrid Obfuscation of Encryption [2]

New Hybrid Obfuscation technique is used for protecting the code by following two phases as illustrated in figure 2.

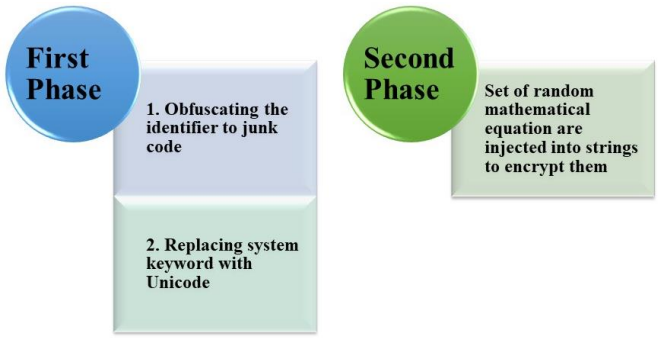

Fig. 2 Two Phases of Hybrid Obfuscation Technique

Four reversing tools were used for this experiment; the tools are CAVAJ, JAD, DJ, and JD to check the effectiveness of technique. The errors occurred for the four tested cases.

Table. 1 Effectiveness of Reversing techniques

Reversing tool	Testing Component	Before hybrid technique	After hybrid technique
CAVAJ	Compiled reversed code error test	ZERO	6
	De-Crypt String test		1
JAD	Output correctness		7
	Compiled reversed code error test		100
	Methods and classes correctness test		22
DJ	Output correctness		0
JD	Identifiers names test		0

Thus the limitations of current obfuscation techniques are describes in Table 2.

Table. 2 Limitations of current Obfuscation Techniques

List	Limitations
• **Logistic map** • **Cipher block chaining** • **Symmetric cipher**	Key is randomly generated at the time of encryption. Reverser can easily guess the key can use to decrypt the entire code.
• **Renaming** • **Hiding**	Hide code can be made available by doing reverse engineering.
Encryption	Longer keys used for encryption but itself leads to slower encryption speed.
Junk code obfuscation	Reverser can use the refactor function to create meaningful names.
Classes combination obfuscation	Reverser can identify the classes by using analysis function.

C. Dynamic Multi Levels Java Code Obfuscation Technique (DMLJCOT) [11]

This technique combines three phases of obfuscation: source code, lexical transformation, and data transformation, in which we hide the source code and byte-code transformation data structures.

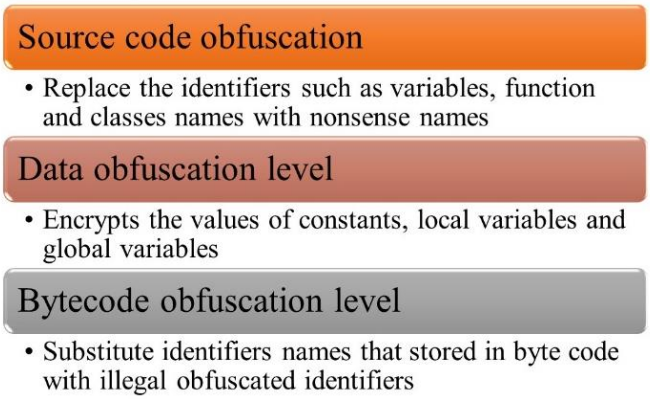

Fig. 3 Three Phases of DMLJCOT

III.RELATED WORK

Mariano Ceccato et al **[3],** This paper presents the results of a large-scale study to analysed the effects of various obfuscation techniques, as measured by several metrics on Java code. Paper discusses study of 44 algorithms and findings show that code obfuscation impacts all the considered metrics, however different algorithms have different effects. This paper recommend which obfuscation technique should be applied as per security requirements. Himanshu Patel et. al **[4].** This paper evaluated the effectiveness of static and dynamic analyses against code obfuscation and the survival ratio of malware after varying levels of obfuscation. It was observed that Polymorphic obfuscation strategies were found to have a lower detection ratio than monomorphic obfuscation techniques. This paper includes data and features obtained by static and dynamic analysis methods that can be used for further research into how these features can be used to improve the effectiveness of machine learning algorithms for malware detection.

Savio Antony Sebastian et al **[5],** This paper study the review of code obfuscation. Discusses the importance of obfuscation in meeting security needs. Explained effectiveness criteria based on complexity metrics, Resistance to attacks and Program State. Different techniques were explained, along with their advantages and disadvantages. Sana Khan et al **[6],** This paper is a report about code obfuscation that reviews about different techniques and methods. Few of the methods that are discussed in this paper are Insert dead code, Add Redundant Operands, Extend Loop and even Loop Transformation. Kumar R et al **[7],** This paper analyses system to detect lexical and string obfuscation in Java malware. Identified a set of eleven features that characterizes obfuscated code, and use it to train a machine learning classifier to distinguish between obfuscated and non-obfuscated malware. Sebastian, et al **[8],** This paper presents a review of code obfuscation and analyse the different techniques which are used to thwart reverse engineers and to protect against malicious code injection and attacks. In this paper we surveyed the need for code obfuscation to make the resulting code unintelligible for human and hard to reverse engineer or being tampered by automated tools. Briefly reviewed code obfuscation methods and techniques and quantitatively evaluate their benefits in accordance to a set of reasonable criteria. Continuous study in this field would help to find more defences that would finally lead to the total endorsement of this new paradigm in our everyday life without any concerns. Popa Romana, et al **[9],** This paper investigates the most common obfuscation techniques for software program source code. Engineering elements of the compiling and interpreting processes are presented form the most widely used programming language based on Java Development Kit and .NET Framework. The reverse engineering of software is implemented on taking into account the architecture of the software development platforms used by the most part of software developers. In order to prevent unauthorized disclosure of software engineering techniques, techniques of the source code obfuscation are used. On the other hand, the reverse engineering of software is used in critical software fields like antivirus program development. Dan Si **[10]**,

This paper is representing how reverse engineering is affecting Software. And in this paper 5 comparative experiments have been given on the proposed random opcode obfuscation algorithm. Random opcode obfuscation anti-coincidence execution is really more effective than false control flow and control flow flattening in OLLVM, which is not simply realized by code stacking. Also, explain How the Code obfuscation technique can secure our source code. Collberg et al **[12],** This paper represents "Sandmark" tool that measures the effectiveness of software-based approaches for preventing software piracy, tampering, and reverse engineering. The Sandmark team's goal is to provide methodologies that will allow users to experimentally decide which algorithms have the lowest performance overhead and the most resilience to attacks. Krishan Kumar et al **[13],** This paper survey the literature on code obfuscation. Many obfuscation approaches have been studied in the literature, each with its own set of constraints. Some provide security against static reverse engineering, while others provide protection against dynamic reverse engineering. This work thoroughly investigates the characteristics of existing algorithms and would aid in the development of effective software protection strategies. Hui Xu et al **[14],** This paper represents the layered obfuscation which applies the idea of layered security to software obfuscation. Taxonomy divides present obfuscation approaches into four categories based on their obfuscation aims. Each layer is further divided into sub-categories or obfuscation techniques. The obfuscation techniques employed by various taxonomic branches are orthogonal to one another. In this approach, it can help users choose obfuscation techniques for constructing layered obfuscation solutions. Kumar, Bhaskari, et al **[15]**, Due to the increasing piracy of the software, a novel attempt is made to discuss and implement some of the obfuscation methods in this paper. Normally after obfuscation, the complexity of the code increases according to logically as well as structurally because of the insertion, removal or rearrangement of the code. The techniques presented have been found to be effective. Here the initial step is taken to obfuscate the code without much increasing the complexity. These mentioned obfuscation techniques have been implemented and analysed. The future work is aimed at the development of a framework for automation of the presented techniques and to provide as a plug-in to support other obfuscation techniques. Also, the aim has been set to implement the proposed idea for large scale software protection and improvement. Ishwar Khadka **[16],** This research explains about software piracy and how it affects to the software companies. Some discussion has done on the software prices weak individuals can't afford it but pirated copies are freely available on internet such pirated copies are developed with malwares which affects to users. Moise, et al **[17],** This article review about the most common forms of piracy are as follows: software piracy and online piracy The article presents and analyzes aspects of the criminal investigation of software piracy and online piracy. Analyzes both some of the criminal investigation acts commonly used in software piracy such as technical-scientific findings and forensic expertise of copyrighted software or related rights, and some methodological issues related to forensic investigation of software piracy and online piracy. Bhattathiripad, et al **[18],** This paper shows how such post-piracy modifications are of special interest to a cybercrime expert investigating software piracy and suggests that the present software piracy forensic approaches require amendments to take in such modifications. Some of elements discussed in this paper, like the positioning of the post piracy fields and dates of post-piracy modification, are often incorrectly discounted as not too reliable; but under clever and careful handling, they can provide valuable supporting evidence. Hosseinzadeh et al **[19],** This paper review systematic literature studies that discuss diversification/obfuscation techniques for improving software security. Paper also suggested the future work of using diversification on container interfaces as well as inside enclaves in trusted execution environments.

Caspar et al **[20],** This paper explains discrete choice model, by combining the well=known concepts of Bayesian interface and information entropy. Finding of this paper provide stepping stones towards the understanding of obfuscation-based decision-making.

IV.CONCLUSION

Obfuscation is a type of software security technology used to protect systems from malicious bugs. The goal of this strategy is not to eliminate security flaws, but to make it difficult for an attacker to exploit them. In this study we reviewed different obfuscation techniques for improving software security. By studying the existing works, we concluded that there should be diversification of obfuscation techniques, should have an option to insert different verbal languages, for the sake of encryption to increase the level of security. We can improve the obfuscation by applying obfuscation technique at runtime this can be done by automation by developing machine learning algorithms that can automatically identify code patterns that are vulnerable and apply appropriate obfuscation techniques. We can expand the software security by converting high level programming language to low-level programming language in addition we can add other hybrid obfuscation techniques can be used to make code more difficult to understand and analyse.

V. REFERENCES

[1]. Li, Yang, Fei Kang, Hui Shu, Xiaobing Xiong, Zihan Sha, and Zhonghang Sui. "COOPS: A Code Obfuscation Method Based on Obscuring Program Semantics." Security and Communication Networks 2022 (2022).

[2]. Al-Hakimi, Asma'A., and Abu Bakar Md Sultan. "Hybrid Obfuscation of Encryption." Coding Theory Essentials.

[3]. Ceccato, Mariano, Andrea Capiluppi, Paolo Falcarin, and Cornelia Boldyreff. "A large study on the effect of code obfuscation on the quality of java code." Empirical Software Engineering 20 (2015): 1486-1524.

[4]. Patel, Himanshu, Deep Patel, Jaspreet Ahluwalia, Vaishali Kapoor, Karthik Narasimhan, Harmanpreet Singh, Harmanjot Kaur, Gadi Harshitha Reddy, Sai Sushma Peruboina, and Sergey Butakov. "Evaluation of Survivability of the Automatically Obfuscated Android Malware." Applied Sciences 12, no. 10 (2022): 4969.

[5]. Sebastian, Savio Antony, Saurabh Malgaonkar, Paulami Shah, Mudit Kapoor, and Tanay Parekhji. "A study & review on code obfuscation." In 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), pp. 1-6. IEEE, 2016.

[6]. Khan, Muhammad Salman, Sana Siddiqui, and Ken Ferens. "Cognitive modeling of polymorphic malware using fractal based semantic characterization." In 2017 IEEE International Symposium on Technologies for Homeland Security (HST), pp. 1-7. IEEE, 2017.

[7]. Kumar, Renuka, and Anand Raj Essar Vaishakh. "Detection of obfuscation in java malware." Procedia Computer Science 78 (2016): 521-529.

[8]. Sebastian, Savio Antony, Saurabh Malgaonkar, Paulami Shah, Mudit Kapoor, and Tanay Parekhji. "A study & review on code obfuscation." In 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), pp. 1-6. IEEE, 2016.

[9]. Popa, Marius. "Techniques of program code obfuscation for secure software." Journal of Mobile, Embedded and Distributed Systems 3, no. 4 (2011): 205-219.

[10]. Si, Dan. "Design and implementation of code obfuscator based on random opcode." In Journal of Physics: Conference Series, vol. 2005, no. 1, p. 012096. IOP Publishing, 2021.

[11]. Yasin, Adwan, and Ihab Nasra. "Dynamic Multi Levels Java Code Obfuscation Technique (DMLJCOT)." International Journal of Computer Science and Security (IJCSS) 10, no. 4 (2016): 140.

[12]. Collberg, Christian, G. R. Myles, and Andrew Huntwork. "Sandmark-a tool for software protection research." IEEE security & privacy 1, no. 4 (2003): 40-49.

[13]. Kumar, Krishan, and Prabhpreet Kaur. "A thorough investigation of code obfuscation techniques for software protection." Int. J. Comput. Sci. Eng 3, no. 5 (2015): 158-164.

[14]. Xu, Hui, Yangfan Zhou, Jiang Ming, and Michael Lyu. "Layered obfuscation: a taxonomy of software obfuscation techniques for layered security." Cybersecurity 3, no. 1 (2020): 1-18.

[15]. Behera, Chandan Kumar, and D. Lalitha Bhaskari. "Different obfuscation techniques for code protection." Procedia Computer Science 70 (2015): 757-763.

[16]. Khadka, Ishwor. "Software piracy: A study of causes, effects and preventive measures." (2015).

[17]. Moise, Adrian Cristian. "Particularities of the Forensic Investigation of Software Piracy and Online Piracy." In Proceedings of the 14th International RAIS Conference on Social Sciences and Humanities, pp. 71-76. Scientia Moralitas Research Institute, 2019.

[18]. Bhattathiripad, Vinod, and S. Santhosh Baboo. "Software Piracy Forensics: Impact and Implications of Post-Piracy Modifications." (2011)

[19]. Hosseinzadeh, Shohreh, Sampsa Rauti, Samuel Laurén, Jari-Matti Mäkelä, Johannes Holvitie, Sami Hyrynsalmi, and Ville Leppänen. "Diversification and obfuscation techniques for software security: A systematic literature review." Information and Software Technology 104 (2018): 72-93.

[20]. Chorus, Caspar, Sander Van Cranenburgh, Aemiro Melkamu Daniel, Erlend Dancke Sandorf, Anae Sobhani, and Teodóra Szép. "Obfuscation maximization-based decision-making: Theory, methodology and first empirical evidence." Mathematical Social Sciences 109 (2021): 28-44.

Cite this article as :

Priyanka Kadam, Hrishikesh Devgude, Srivaramangai R, "A Comprehensive Analysis of Software Obfuscation Techniques", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 9, Issue 3, pp.314-320, May-June-2023. Available at doi : https://doi.org/10.32628/CSEIT2390376
Journal URL : https://ijsrcseit.com/CSEIT2390376