ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 12, Issue, 11, pp.14966-14969, November, 2020

DOI: https://doi.org/10.24941/ijcr.40257.11.2020

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

ENHANCE SOFTWARE SECURITY UTILIZING CODE OBFUSCATION

***Noha M. Altalhi, Emad Alsuwat**

Department of Computer Science, College of Computers and Information Technology

ARTICLE INFO

Article History:
Received 20th August, 2020
Received in revised form
17th September, 2020
Accepted 25th October, 2020
Published online 30th November, 2020

Key Words:
Obfuscationcode,
Software Security,
Reverse Engineering,
Obfuscation Tools.

ABSTRACT

The software-based applications have increased dramatically over the last couple of years. The main focus of software developers is the functionality of the developed application keeping security behind. Therefore, security of these software applications became a major issue. A key threat that is facing software security is reverse engineering attack, which is also known as software analysis. Reverse engineers' main objective is to discover the concealed code. To prevent and detect this kind of threat, we can use code obfuscation. Code obfuscation can not only provide security to the developed software application, but also hide software flaws via techniques such as masking of the fingerprint of the developed software. In this paper, we survey recent research related to code obfuscation and reverse engineering attacks. In addition, we review some techniques that a reused in code obfuscation. Moreover, we present the common tools that are used to enhance software security against reverse engineering by code obfuscation.

Citation: ***Noha M. Altalhi, Emad Alsuwat.*** **2020.** "Enhance Software Security Utilizing Code Obfuscation", *International Journal of Current Research*, 12, (11), 14966-14969.

INTRODUCTION

Nowadays, with the development of technology, software applications have become a daily event. These software applications depend on the performance and the system of the program. One of the goals to keep in mind is to increase software security and develop plans to protect programs against threats, another reason to increase software security is the illegal copying of software due to the high price of operating systems and other applications. Most security problems, according to previous studies (McGraw, 2015) were due to a 50% software vulnerability, Such as knowing software vulnerabilities and exploiting them for penetration. It is not necessary to build a strong program, but more importantly, to maintain the structure of software while developing it in order to ensure software security.

Therefore, it is necessary to protect this software and prevent illegal copying and misuse of it by using code Obfuscation Code obfuscation prevents internal inspection of the program, defense against reverse engineering, and masking of the fingerprint of software and is also used to hide software flaws.

***Corresponding author: Noha M. Altalhi,**
Department of Computer Science, College of Computers and information technology.

This paper surveys recent research related to code obfuscation and the general attacks of reverse engineering and finally some technique and tools used to enhance software security against reverse engineering by code obfuscation.

Overview of Code obfuscation: Code obfuscation is a technique to make programming codes incomprehensible (Xu, 2017). It is commonly used to protect intellectual property by obfuscating the code of the program and to prevent the reverse engineering that the attackers perform on the programs. The code obfuscation is based to encrypt entire or some of code, or adding some unused code to the original code or by renaming the variables to hide and protect the code. The obfuscation code should be to protect the original code but not change the functionality of the code itself. The code obfuscation can divide into three type based on the aim of protection: preventive obfuscation, transformation obfuscation, and polymorphic obfuscation (Gautam, 2019). preventive obfuscation: the main goal of preventive obfuscation to protect the original codes by hindering attackers from obtaining it.

Transformation obfuscation: reduce the ability to read the original code by using some of these type (Balachandran, 2013):

- Layout transformation, make human unable to understand the program.

- Control transformation, make the flow of execution more difficult for reverse engineering.
- Data transformation, transforming data and the structures of these data to obfuscation form.
- Polymorphic obfuscation: the goal of polymorphic obfuscation to obstruct attackers from specify the advantage of target.

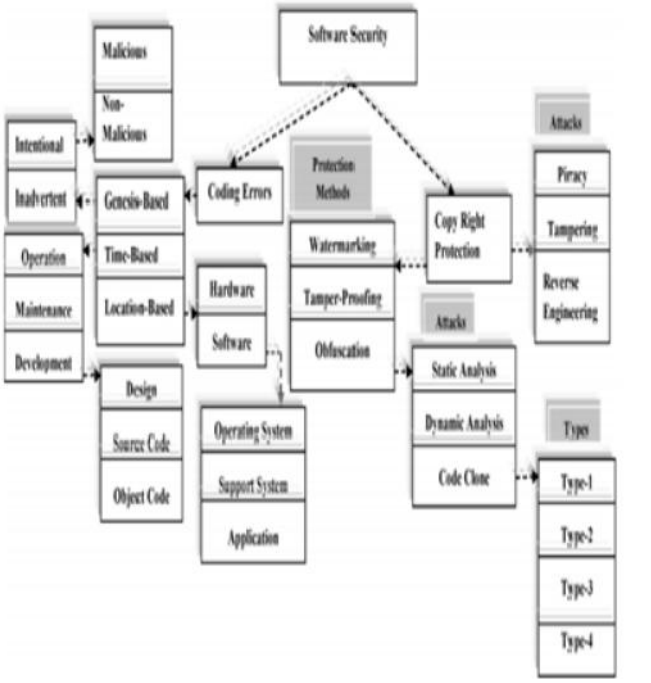


Fig. 1. Software security problem[Gautam, 2019].

The advantages and assets of obfuscation: The obfuscation has many feature and the main one is protection to make code difficult to understand, the flexibility of obfuscation that mean ability to prevent automated deobfuscation by transmogrification, the Diversity that known as generate clear illustration that have same functionality of original code but different in structure and among the features of the obfuscation that recently added is cost and performance (Kulkarni, 2014).

The assets of obfuscation are evaluated according to four categories:

- **Potency:** The amount of ambiguity in the program.
- **Resilience:** The amount of difficulty to break the deobfuscation.
- **Cost:** the amount of overhead that needed to add obfuscation to program.
- **Stealth:** the way to add obfuscation code to original program.

The difference between code obfuscation and model obfuscation: The obfuscation approach has two program categories: code-oriented obfuscation and model-oriented obfuscation. The code obfuscation is important for software security engineers to provide protection for software, it clarifies the gaps with model obfuscation (Nicolson, 2017). What is important for scientists who are pursuing the studies about circuits or Turing Machines is Model-oriented obfuscation, also important for cryptographers (Barak, 2012). There is some different problem in code obfuscation and model obfuscation.

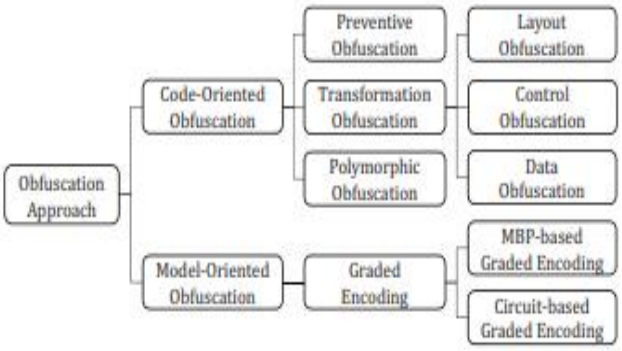


Fig 2. Obfuscation program approach [3].

First, the code is very intricate than the general model. Second, these two categories have different purpose of attacks. In the code obfuscation the purpose of adversarial is related to understanding the software program, and the methods that uses to attacked could be de-obfuscation. And about model obfuscation the purpose of adversarial is related to ability to find the function of computation model.

Reverse engineering attacks: There are many of threats on software such as reverse engineering, tampering, and piracy. the attacker can find and catch the functionality of software by using one of these attacks. This paper focus on reverse engineering and the type of reverse engineering. Reverse engineering also known as software analysis (Cappaert, 2012), that refer to examining the inside code of software and the attacker trying to elicit the covert algorithms or some other information from software.

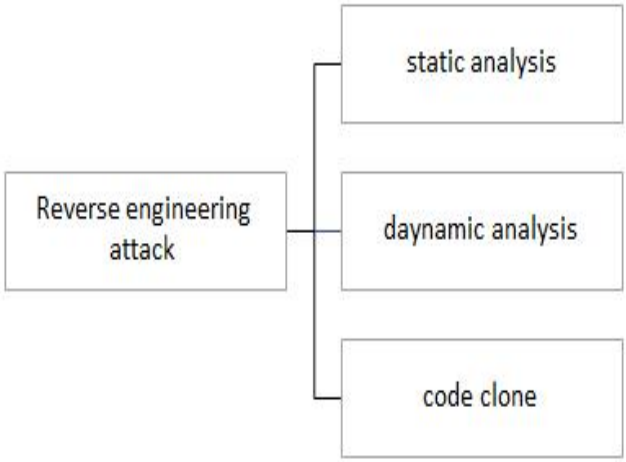


Fig 3. Type of reverse engineering attacks

As shown in Fig3, there are three type of reverse engineering attacks: static analysis, dynamic analysis and code clone. Static analysis : or static code analysis that refer to the ability of adversary to analyze information of code or binary without need to executing the code (Cappaert, 2012). This process helps to understanding the structure of hidden code. Dynamic analysis: that means the ability of adversary to analyze the result of code by run the code on the different input set. This helps to studying the code at run-time (Afianian, 2018). The third type of reverse engineering attacks is software clone, that refer to trying to remove the obfuscation code from the original code to understanding the structure of original code. Sometimes in software system need to use software code clone for some reasons in maintenance and the software developer

Obfuscation tools	Allowance	Characteristics
DashO Pro	Commercial tools	- It has GUI to simplify the interactions. - Able to control the flow. - Contain obfuscation and watermarking algorithm. - The renaming is cross JAR.
Allatori	Commercial tools	- Ituses the Name, Flow and Debug obfuscation. - It provides a strong protection against popular De-compiler. - Utility for stack tracking. - Has full watermarking functionality. - Boosts speed.
Pro-Guard	Thesource is open	- It provides improvement for byte-code. - Flow obfuscation technique. - Stack trace translate tool. - It uses the reserved for obfuscation.
Y-Guard	Free tool	- Name obfuscation technique. - Shrinking code.
ZelixKlass Master	Commercial tools	- Easy to use. - Powerful obfuscation. - Incremental obfuscation - Flow and Name obfuscation technique.
DexGuard	Open source	- Advance obfuscation and run time protection. - Fully compatible with all build tools. - It is very easy to upgrade.

using the code clone also, but the copy, cut and paste are the bad practices according to the study of uses software.

Enhance software security: This section about the way to enhance the security of software by using code obfuscation techniques and some tools when attacker want to use the code of software, he must understand the source code before anything, so the code obfuscation used as a one of important technique to protect the original code from the misused and to prevent reverse engineering malware.

Code obfuscation techniques: The code obfuscation has many techniques that must use at the design level obfuscation, where developers change the source code according to these techniques to make the code difficult to read and understand for attacker. These techniques are classified into:

- **Name obfuscation:** This technique is concerned about renaming the variable and change the class to unmeaning name that help to reduce the ability to read the code (Buzatu, 2012).
- **Byte code obfuscation:** this technique also called mirror of the code real (Mahfoud, 2018), it contain encrypted name and class from the source code. This technique is the effective one to provide protection for the original code.
- **Control flow obfuscation:** In this technique the code will modified by assemblage and re-organizing the code. The assemblage operation will convert the computations, while the re-organizing operation will organizing the code in random way and keeping the integration of basic blocks (Sebastian, 2016).
- **Debug info obfuscation:** the task of this technique is to concealing the debug information, these information may be helps to obtain significative data like source name and number of line (Yasin, 2016).
- **Lexical Obfuscation:** this type of technique used to alter and clear the information of lexical from compiler and for correcting the information and comment. this technique shouldn't use alone, it must be combined with other techniques to increase the protection (Mahfoud, 2018).
- **Code Shield:** this technique is concerned with class file, the size of executable file will be reduced by this technique and obfuscate the classes name and it will and also the flow of code will controlled.

```
1  public static void initSetting(Editor editor){
2    editor.remove("deviceSalt");
3    editor.remove("encryptedKeyPass");
4    editor.commit();
5  }
6  private void goNextStage(){
7    gotoStage(this.stage + 1);
8  }
```

(a) Original classes.dex.

```
1  public static void a(Editor arg0){
2    arg0.remove(R.a("c\u0018}\u0014h\u0018X\u001cg\t"));
3    arg0.remove(android.support.v7.appcompat.R.a(
                                        "u,sDi2d't\tu;@4c"));
4    arg0.commit();
5  }
6  private void a(){
7    a(this._ + 1);
8  }
```

(b) Obfuscated classes.dex.

Fig. 4. Obfuscation example by Allatoritools [15]

Code obfuscation tools: There are many different tools used to obfuscate code and provide protection against the Reverse Engineering, these are the most common tools outlined below:

Y-Guard: The aim of this tool to protect the original file by change the method, class and the name of field with a new random characters to make the code complex to understand (Rajadurai, 2011). Pro-Guard: this tool similar to the other tools in minimize the size of byte-code of the file. It provide both obfuscating and optimization for the code, by reduce the all instructions that not used in the original file and also change the name of class and methods to provide protection for original code (Wang, 2017).

Allatori: This tool protects the original code against the reverse engineering by renaming and minimize the class, methods and variable of byte-code in addition to correctly used methods of obfuscation.

Sand Mark: This tool is the academic tool among the other, it provides protection from piracy, manipulating, and reverse

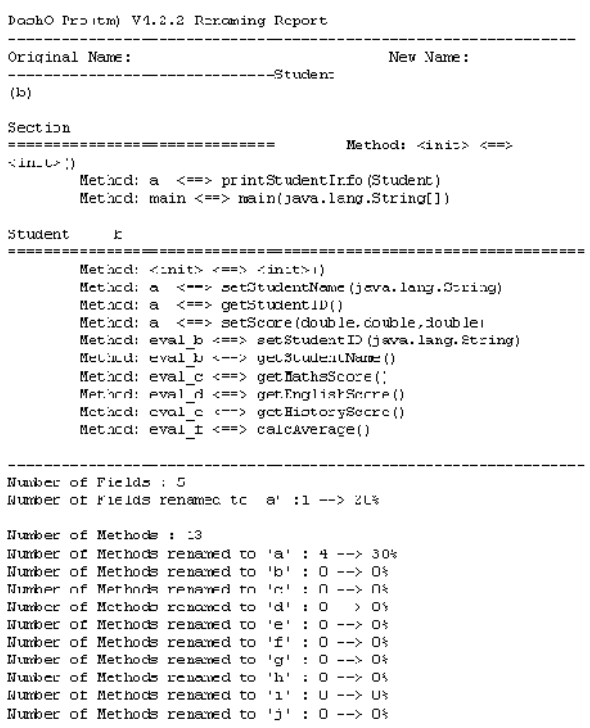
engineering. this tool contain two type of algorithm: code obfuscation and watermarking (Collberg, 2003).

- **Zelix Klass Master:** This type of obfuscation tool used to minimize the size of byte-code, it change the code flow and logs in addition to rename the original name to increase the protection (Pizzolotto, 2019).

Fig. 5. Obfuscation example by ZKM tools

- **Dash-O-Pro:** Its advantage is obfuscation, improvement and watermark. This tool has three mode: Advanced Mode, Quick Jar Mode and Command Line Mode (Patki, 2008).

Fig. 6. Obfuscation example by DashO Pro tools.



Conclusion

The software vulnerabilities may cause security problems, so should maintain the security of software structure. The major problem that are facing application developers is reverse engineering attacks. In this paper, we presented the three main types of reverse engineering attacks. We also presented code obfuscation which is used to prevent such attacks and to enhance the software security. This paper also showed the benefits of code obfuscation and the common techniques that are used in code obfuscation, such as tools for obfuscate the code and their features.

REFERENCES

Afianian, A. *et al*. 2018. *Malware dynamic analysis evasion techniques: A survey*. arXiv preprint arXiv:1811.01190.

Balachandran, V. and S. Emmanuel, 2013. *Potent and stealthy control flow obfuscation by stack based self-modifying code.* IEEE Transactions on Information Forensics and Security. 8(4): p. 669-681.

Barak, B., *et al*. 2012. *On the (im) possibility of obfuscating programs.* Journal of the ACM (JACM), 59(2): p. 1-48.

Buzatu, F. 2012. *Methods for obfuscating Java programs. Journal of Mobile, Embedded and Distributed Systems*, 4(1): p. 25-30.

Cappaert, J. 2012. *Code obfuscation techniques for software protection.* Katholieke Universiteit Leuven: p. 1-112.

Collberg, C., G.R. Myles, and A. Huntwork, 2003. *Sandmark-A tool for software protection research.* IEEE Security & Privacy. 1(4): p. 40-49.

Gautam, P. and H. Saini, 2019. *Intensification of Software Security using Secure Obfuscation with Injecting and Detecting Code Clones*.

Kulkarni, A. and R. Metta. 2014. *A new code obfuscation scheme for software protection*. in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*. IEEE.

Mahfoud, A., *et al* 2018. *Code Obfuscation. Where is it Heading?* International Journal of Engineering & Technology. 7(4.1): p. 22-27.

McGraw, G., 2015. *Software security and the building security in maturity model (BSIMM).* Journal of Computing Sciences in Colleges, 30(3): p. 7-8.

Nicolson, K.A., *et al*. 2017. *Obfuscation assisting apparatus.* Google Patents.

Patki, T. 2008. *Dasho java obfuscator*.

Pizzolotto, D., R. Fellin, and M. Ceccato. 2019. *OBLIVE: Seamless Code Obfuscation for Java Programs and Android Apps*. in *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*.

Rajadurai, K.P. and M.I. Ravi. 2011. *Formulating a defensive technique to prevent the threat of prohibited reverse engineering*. in *The 2011 International Conference and Workshop on Current Trends in Information Technology (CTIT 11)*. IEEE.

Sebastian, S.A. *et al*. 2016. *A study & review on code obfuscation*. in *2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave)*. IEEE.

Wang, Y. and A. Rountev. 2017. *Who Changed You? Obfuscator Identification for Android*. in *2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*..

Xu, H., *et al*. 2017. *On secure and usable program obfuscation: A survey*. arXiv preprint arXiv:1710.01139.

Yasin, A. and I. Nasra, 2016. *Dynamic Multi Levels Java Code Obfuscation Technique (DMLJCOT).* International Journal of Computer Science and Security (IJCSS), 10(4): p. 140.

****