Scientific Journal Impact Factor (SJIF) : 3.518
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745 ISSN (PRINT) : 2393 - 8161

A CODE LEVEL OBFUSCATOR FOR CORE JAVA

N.Dhakshyani[1],K.Ganga devi[2],G.S.Siva shankari[3] and M.Ashok[4]
[1,2,3] Information Technology, Rajalakshmi Institute of Technology,
[4].Sr AP/Information Technology, Rajalakshmi Institute of Technology

Abstract—The incidents of software tampering and piracy have become common in this era. It is possible for software pirates to extract piece of code and incorporate it into their own programs with considerable case. In this paper, target java applications are available here as java class files. Java byte code is platform independent and makes Java executables highly capable to being reverse engineered. Obfuscation is a technique to protect against security threats. Reading, writing and reverse engineering can be made into a program by obfuscation. This paper demonstrates the working of code obfuscator that operates on java file and produce obfuscated versions with different level of security.

Keywords—Piracy, software pirates, java class files, Obfuscation, Reverse Engineering.

I. INTRODUCTION

Many companies are expanding their service offer through World Wide Web (www) to accommodate their customer. Now a day's software industries mainly focus on online services. In fact e-Governance is seen for improving the economic conditions of the country. The online computing and querying becomes common, it concerning about data protection and security has taken on new urgency. The remaining of the paper is structured as section II describes the background process. Section III describes the existing system of code obfuscator. Section IV illustrates the proposed system of code obfuscator with different levels of security for achieving effective software protection. Section V illustrates the experimental analysis of existing and proposed system and finally conclusion is demonstrated in section VI.

II. BACKGROUND

A. Piracy

Software piracy is the illegal copying, distribution or software usage [6]. An organization, which works on software piracy will get more profit and it get the attention of crime group in many countries.

B. Software Pirates

A person who involved in illegal copying, distribution or use of software is called as software pirates. Reverse Engineering is common in software world , the meaning of reverse engineering is taking an application for which source code and documentation is not available and trying to recover details regarding to its design and implementation [4].Fig1 shows the reverse engineering is related to the several features of software security. It is understandable that, how some actions are performed
in a program. Hence we need to protect the software against software threats and code obfuscation is a secure mechanism for this threat [7].

Figure 1. Process of reverse engineering

C. Code Obfuscation

Obfuscation is a means of "obscuring" the real meaning of our java code. Code obfuscation is one of the best method for protecting Java code from reverse engineering. Fig 2 shows that the Obfuscation renders software unintelligible but still functionality equivalent to the original code. It is difficult to understand the program and it is resistant to reverse engineering.

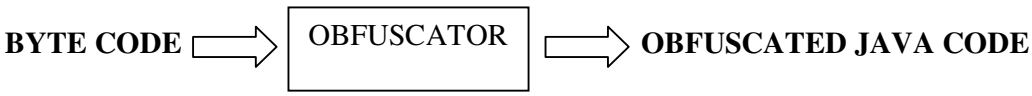

Figure 2. Process of code obfuscation

The code obfuscation [8], [3] problem is formally stated as follows:

Given a set of obfuscation transformations T={T1,T2…TN} and a P consisting of source code objects (classes, methods, statements, etc.) {S1, S2,...Sk}find a new program i)P'={..,S' j=Tj(Sj),…} such that:

Has the same observable behavior as P, i.e. the transformations logic are maintained in its original state.

i) The obscurity of P' is maximized, i.e. understanding and reserve engineering P' will be strictly more time consuming than understanding the reverse engineering P.

ii) The cost time, space and other factor are high for transformation.

D. Machine intelligence

Intelligence exhibit as artificial (non-natural, man-made) entity , the essential quality of a machine thinking level similar to the same general level of human being is said to be machine intelligence[2],[1]. A machine performance are similar to human intelligence [5] , such as learning , self-correcting and adapting .

III. EXISTING SYSTEM

Obfuscation is the obscuring of intended meaning (Changing the code into non-understandable format).Fig 3 shows that in the Existing obfuscators works in two stages:

Stage 1: The obfuscator accepts the candidate file as input and it removes comments and indentations from the program and it display the obfuscated version as output.

Stage 2: The obfuscator accepts the stage1 obfuscated file and obfuscated the following fields of input.

- **a.** Declared methods
- **b.** Declared fields
- **c.** Interfaces implemented by the class[9]
- **d.** Nested classes

The above fields are changed to random integers and the obfuscated version is printed as output. In this paper we describe the need of code protection and propose code obfuscation with different level of security i.e. instead of substituting the random integers, Random underscores are substituted. We are creating this obfuscation code as a plugin for Net Beans.

Figure 3. Function of Existing obfuscator

The drawback of the existing system is when the random numbers are replaced with some string literals the original code may be acquired.

IV. PROPOSED SYSTEM

In order to overcome the disadvantage of existing system, Fig 5 shows that we provide a new idea with different level of security for code obfuscation.
Fig 5 illustrates the architecture of the proposed system for code obfuscation. The aim is to incorporate and enhanced code obfuscating security features with three level of security.
i) First, the Java code is given as input to the compiler, the compiler will generate the class file and this class file will undergo first level of obfuscation giving the obfuscated code as output.
ii) Similarly it undergoes second and third level obfuscation giving more obfuscated code. Then, if the class file is decompiled to get the original Java file in obfuscated version.
The Working function of the proposed architecture as level by level is explained as follows:

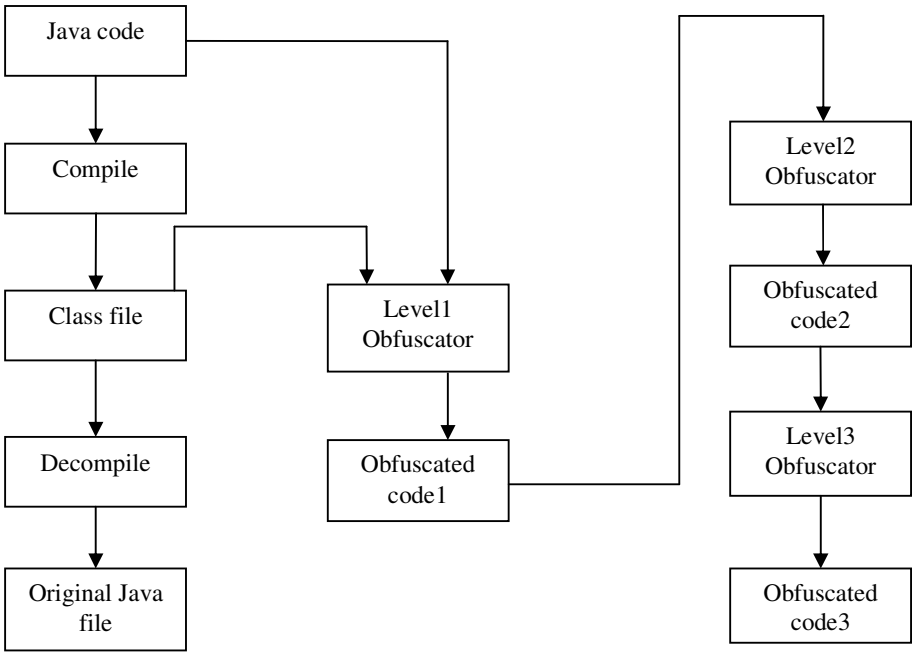

Figure 4: Proposed system architecture

LEVEL 1:

In this level, the obfuscator finds user defined variable functions and classes then it obscure all the user defined variables with random underscores. Fig 4.1 shows the working function of level1 obfuscator. If the hacker performs reverse engineering process, they will see only the symbols and keywords and not the logic of the code. Though it is difficult to the hackers to find the logic, the security in this level is moderate.

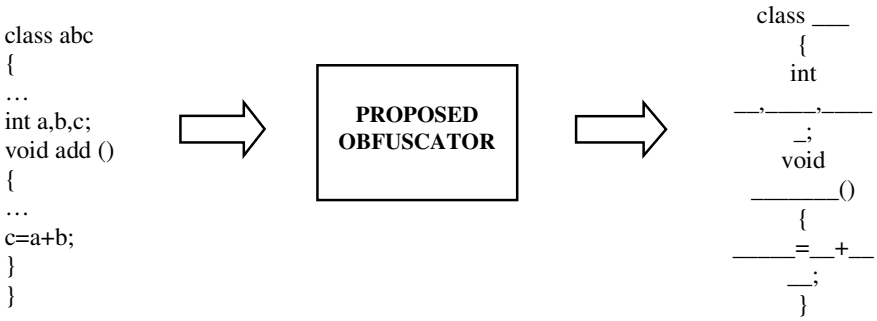

Figure 4.1. Level1 obfuscator function

LEVEL 2:

This level is similar to level1 but instead of random underscores, the user defined variables are obscured with the very long randomly operated string with the combination of underscores, numbers and characters. While comparing to level 1 the security level is high.

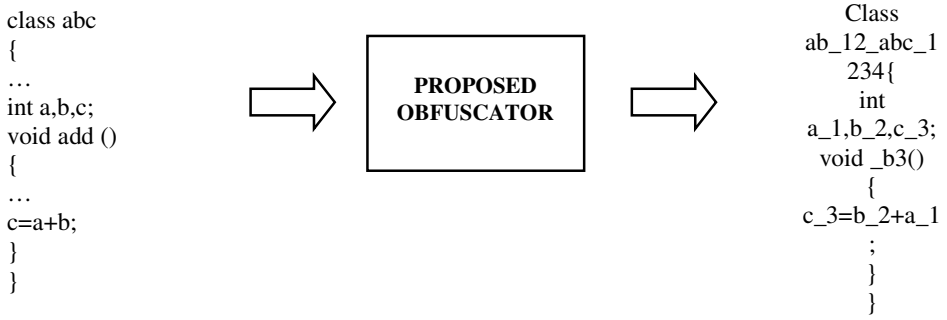

Figure 4.2. Level2 obfuscator function

LEVEL 3:

In this the security level is very high, by the obscuring the entire program into unreadable form, after this the functionality of obscured java file is not same as original file.

Figure 4.3. Level3 obfuscator function

V. EXPERIMENTAL ANALYSIS

Fig 5 shows the difficulties in finding the source code from the obfuscated version of the code. In the existing system, the probability of finding the logic of the original code little bit easy when the user defined variables, classes, functions are replaced by the string literals. This is shown by the curve named as existing in fig. In the proposed system, we are going to replace the user defined variables, classes, functions with the special characters with different level of security. So, the probability of difficulty to find the logic of the original source code is high when comparing with the existing system. This is shown by the curve named as proposed in fig 6.

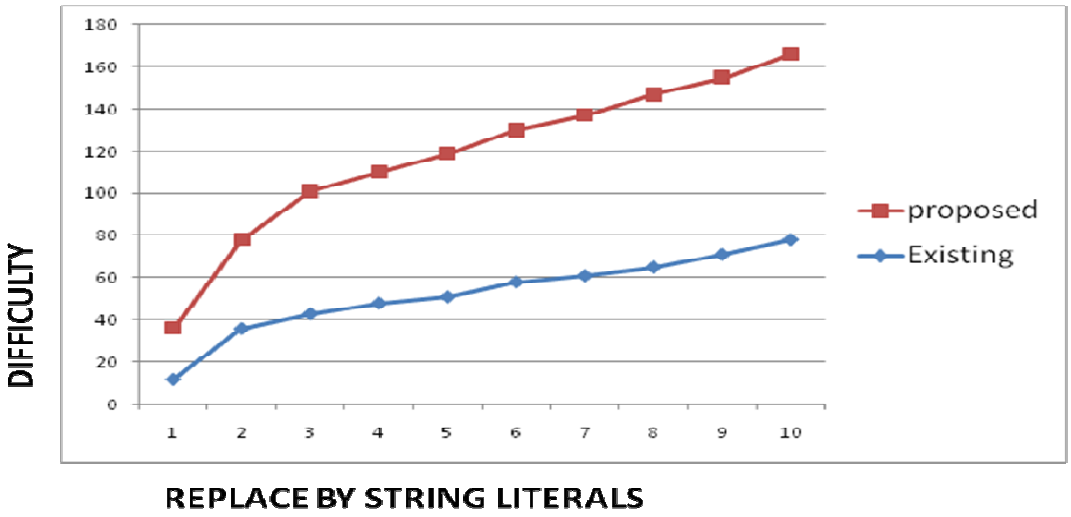

Figure 5: Comparison of Difficulties

VI. CONCLUSION AND FUTUREWORK

Obfuscation is proposed as a PLUGIN for software products. Thus, we protect the class files against the security threats. The paper also illustrates the working of a code obfuscator that operates the java files and produces obfuscated versions as output in two stages. The existing obfuscation techniques are applied to programs in a static manner. The obfuscation technique is not implemented dynamically. So, as a future work we strive to retrieve to realize our proposed level-based approach by implementing dynamic decision making ability to the software. Darwinian evolution shows the behavior of naturally occurring intelligent entities and can be used to guide the creation of entities that are capable of intelligent behavior. Our efforts are directed towards encouraging the development process in software programs.

VII. REFERENCE

[1] P. Singh, V. Sidana, K. P. Aggarwal, N. Verma, S. Verma, and A.B.Patki, "Introducing Machine Intelligence quotient as a new COTS Evaluation measure," in *proc. 4th National conference*, INDIAcom 2010.

[2] H.J.Park, B.K.Kim, "Measuring the machine intelligence quotient (MIQ) of human machine cooperative systems".IPSJ Journal, vol .67, no.68, Aug. 2008, pp. 4334-4569.

[3] K. Fukushima, S. Kiyomoto and T.Tanaka, "An Obfuscation Scheme Using Affine Transformation and its Implementation", IPSJ Journal, vol .47, no. 8, Aug. 2006, pp. 2556-2569.

[4] E. Eilam, Revering: Secrets of Reverse Engineering.1st ed. USA .Wiley, 2005

[5] D. B. Fogel and L. J. Fogel, "Evolution and Computational Intelligence," in *Proc. IEEE, Neural Networks*, vol. 4, pp. 1938-1941, Dec. 2005.

[6] G.Naumovich and N.Memom, "Preventing piracy, reverse engineering and tampering."IEEE Computer, vol, 36, no.7.pp.64-71.July 2003.

[7] P. C. Van Oorschot, "Revisiting Software Protection," in *proc. 6th International Conf. Information Security (ISC 03), LNCS 2851,* Springer - Verlag, 2003, pp. 1–13.

[8] C. Collberg and C. Thomborson, "Watermarking, tamper-proofing and obfuscation-tools for software protection," IEEE Trans on Software Engineering, vol.28, n0. 8, pp. 735-746, Aug. 2002.

[9] H. Schildt, *The Complete Reference Java 2*, 5th edition, New Delhi, India: Tata McGraw-Hill, 2002. ch. 17, pp. 546-577.

