AN ENHANCED TOOL FOR TESTING THE SOFTWARE QUALITY

J. Keerthana [1], K.S. Jeen Marseline[2]

ABSTRACT

This paper proposes a new software quality assessment tool named as SQRA (Software Quality Risk Assessment), SQRA tool permits user to check the software quality and risk based on different features.

The benefits of SQRA comprise early imperfection recognition in code, malicious code finding, duplicate code detection and defect root localization etc. It constructed a model that combines metrics to provide and measure of test code quality and risk assessment. The proposed deals with the issues of reliability and optimal quality verification using traditional and new object oriented metrics. This also helps to find the risk and quality of software using opcode sequence detection algorithm. In this paper the SQRA tool helps to find the malicious files using OPCODE sequences and frequency patterns which are extracted from the program files after disassembly. In this paper the result and analysis of SQRA tool is proposed with a fast and accurate report on the selected program or dll which is to detect quality and risk of program files which will later be used to improve the whole software.

[1]Research Scholar, Sri Krishna Arts & Science College, Coimbatore - 641 008, Keerthu0306@gmail.com

[2]Head, Department of Information Technology, Sri Krishna Arts & Science College, Coimbatore-641 008, jeenmarselineks@skasc.ac.in.

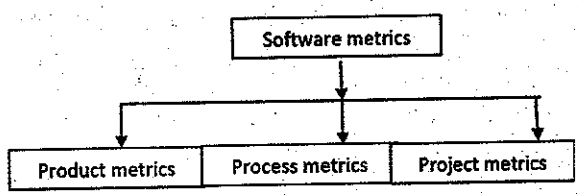
Key words: OPCODE, SQRA, Disassembly

I. INTRODUCTION

In general, testing is the process of evaluating software against bugs and errors. Software testing and software development life cycle constitutes an important part in it [2]. Many software-related problems in the past due to the lack of testing of the software, and in fact many of them are due to social problems [2].

Software Metrics

Software metrics is used for measuring software, or some of the property of a segment of software which is defined by its specifications [1]. Software metrics using those techniques together are meaningful and timely management information (MI) and measurement-based technologies which are used to deliver the software development process and to improve software quality which is the process of assessing the size of its river systems that provide continuous use [1].



Dig 1.Software Metrics

A number of software metrics widely used in the software industry are still not well understood [2]. Software metrics can be classified into three Software metrics Product metrics Process metrics Project metrics categories which are product metrics, process metrics and project metrics.

Product metrics describe the characteristics of the products such as size, complexity, design features, performance, and quality of the level of the product

[4]. Process metrics are used to improve the software development and maintenance. Examples of the process includes , that lack of testing during the development of the arrival time, the capability of removing the defect, and the standard process includes the response time [4]. Project metrics describe the project characteristics and execution. Examples are of the number of software developers involved in the project software, cost, schedule, and product life cycle times on the employee [5].

Product Metrics

SQRA tool has its development (analysis, design, coding, and testing) at any stage of the production and quality of the software solution which gives the software quality research to understand the defining characteristic of this product to handle measurements [6]. It also measures the software design, source code, practical designs of complex measurements, maintenance, performance, ease of preparation on the scale, the size of the project, experimental techniques [6].

II. Existing Research

There are several tools available to detect a variety of software quality [7]. Although each result is a measurement of permutations, depending on the amount of variation in the system, they can be difficult to find the meaning is different [1]. A program such as heuristic analysis in which, some of the quality or availability of diagnostic methods used to detect malicious code; however, these methods are rarely used for quality care with a personal system, they generally are prone to false positives.

Object - Oriented Metrics :

- Number of scenario scripts (NSS)
- Number of key classes (NKC)
- Number of support classes (e.g. UI classes, database access classes, computations classes, etc.)
- Average number of support classes per key class
- Number of subsystems (NSUB)

Risk identification metrics:

Code Emulation :

Code emulation is an effective tool for detecting malicious codes, since it involves in emulate the software on a virtual machine rather than a real processor [8]. Because when the virus is in a prohibited environment, the system that emulates the virus will not run any risk of dangerous side-effects of the virus [9].

Pattern based scanning:

One of the most common approaches that are used is string or signature scanning for detecting viruses,

[10]. At a higher level, a variety of signature scanning techniques are available , but in nonmalicious code viruses are not found [11], which are not found in the wild cards contain bytes, involves a string. While scanning the signature works well for most viruses, the possibility of a mutation of the virus, which it is almost impossible to create enough differences in the use made of a reliable signatures.

III. Proposed System

This paper proposes a new tool named as SQRA (Software Quality Risk Assessment) for determining the quality of the software.

Contributions:

The contributions of the work in this paper are summarized as follows:

- It constructed a model that combines metrics to provide a measure of test code quality and risk assessment.
- Software Quality and Risk Assessment (SQRA) tool has been proposed.
- This follows the object oriented metrics such as coupling and coherence and safety related metrics.
- Tool customization option.
- Using the Opcode sequence the tool can perform safety metrics.
- User policy based quality verification.

This paper describes the construction of a multi parameter quality testing tool, called **SQRA (Software Quality and Risk Assessment)**, using Opcode sequence detection algorithm along with object oriented metrics, which is used to verify assembly code compiled from a C # code from the compiled assembly language as the paper source object-oriented metrics, which are used to verify, using combined Opcode sequence detection algorithm, SQRA (software quality and risk assessment) of the multi-parameter quality checking tool. It reduces the constraints on the implementation of the testing process for this type of a namespace, comprehensive code of points required in the program Data values. The SQRA tool performs the following.

- SQRA, the program code is used to check the quality and risk assessment, and integrates metrics.
- The connection parameters are the as objectoriented measure of coherence and security related metrics.
- Using the Opcode sequence, the SQRA can perform safety related metrics.
- Opcode sequence also helps to find the code reuse or misuse attack in the software.
- This customizes user policy, which can be changed according their different quality requirements.

The paper has investigated opcode frequency, which is used to identify and differentiates the malicious code from the other codes. The main contribution of this research includes safety metrics with opcode frequency for a compiled executable files, which can't be analyzed in the previous tools. The testing procedures went beyond standard Chi-Square tests

in an attempt to isolate the opcode that are most strongly associated with certain malware classes.

The SQRA overcomes the challenges in the Existing system. The challenging tasks with type-checking assembly code versus Source code:

1. Program sequence detection mechanism is required with unrelated type at different level of points. Some programs may differ and the end users requirements may differ.

2. In certain cases, there are critical dependency is not easily identified, so it's done using this tool. The SQRA tool covers the above issues along with overall performance analysis.

The following tables list outs the metric used in the proposed framework:

Source	Metric	OO Construct
Traditional	Cyclomatic Complexity (CC) Lines of Code (LOC) Comment Percentage (CP)	Method
	Number of attributes defined in the class	
	Number of methods defined in the class	
New Object Oriented	Weighted Method Per Class (WMC)	Class/Method
	Response for Class (RFC)	Class/Message
	Lack of Cohesion of Methods (LCOM)	Class/Cohesion
	Coupling between Objects (CBO)	Coupling
Risk Metrics	Malicious Objects	
	Sequence metrics	

Figure 2. Software Quality and Risk Metrics

IV. METHODOLOGY

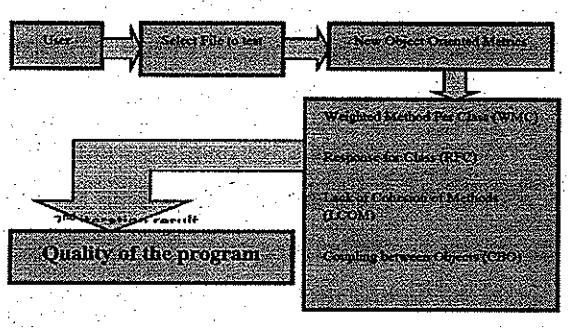


Figure 3. Methodology

The idea of applying Machine Learning (ML) techniques for the detection of different malwares based on their respective binary codes. The 3 different feature extraction (FE) approaches were employed: features that are extracted from the Portable Executable (PE) section, and the meaningful plain-text strings that are encoded in the programs files, and the byte sequence features. Additionally all the code will be tested with the opcode considerations. So in this paper the fastest tool among other existing approaches are been developed.

OPCODE:

An opcode is short for 'Operation Code. An opcode is a single instruction that can be executed by the CPU.

Example

MOV, AL, 34h

The opcode is the MOV instruction. The other parts are called the 'operands'.

Operands are manipulated by the opcode. In this example, the operands are the register named AL and the value 34 hex

OPCODE EXTRACTIION:

The process of identifying and extracting opcode from the assembly is the major process, where the system stores its assembly data in the form of .dll (dynamic link library)

The first step consisted of gathering random samples of malicious and non-malicious binaries.

Figure 4. Opcode Extraction Process

This paper will be able to find the malware and mismatch files with the help of sequence of opcode. The frequency measurement has been used here. . Some of the opcode (i.e. mov or push), however, have a high frequency of appearance within malware and benign executables, therefore the resultant similarity degree (if based on opcode frequency) between two less can be somehow distorted. Hence, this proposes a way to avoid this phenomenon and to give each opcode the relevance that it really has.

OPCODE CHECKING ALGORITHM:

SAMPLE CODE:

Input: Assembly file

Output: opcode count

Steps:

1. Get the file name (f) of the assembly f(As)
2. load the assembly
   a. Load(As)=readall(f)
3. get all the methods(M) within the loaded assembly Ls
   a. for each M in Ls.,
      M=Ls1,Ls2…Lsn.
4. check if the method has a body.
5. get the operand of the current operation.

Sequence testing:

The algorithm performs the sequence test by applying the sequence feature identification.

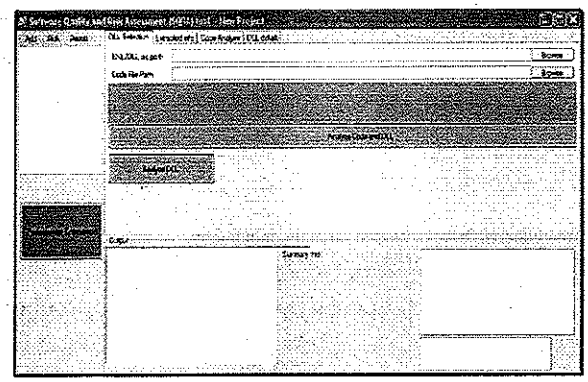
1. Read every opcode and finds sequence number
2. Check opcode name with the following opcodes
3. If(opcode equal to oplist(i)) then add to list
   a. plist="MOV", "JMP", "NOP", "SUB", "EBP", "ADD", "PTR", "CMP", "DWORD", "4F", "JNZ", "SHORT", "LOCK", "PUSH", "CALL" , "PULL", "AND"
4. Store the opcode details into count.dll
5. Get count and opcode details and check
6. If the details are mismatched then return code as malicious
7. Else return valid(true)

The proposed system performs the following steps to identify the malicious or abnormal code.

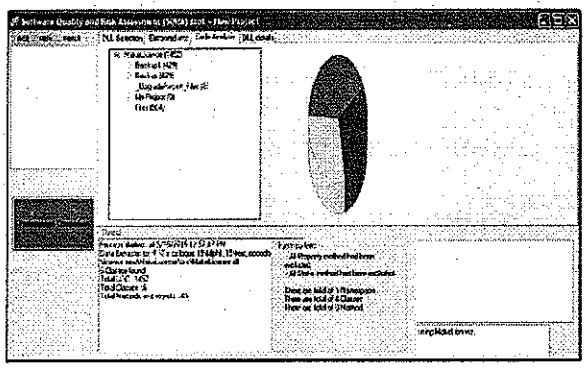
The SQRA tool technique has been introduced with the help of opcode.

IV Experimental Results

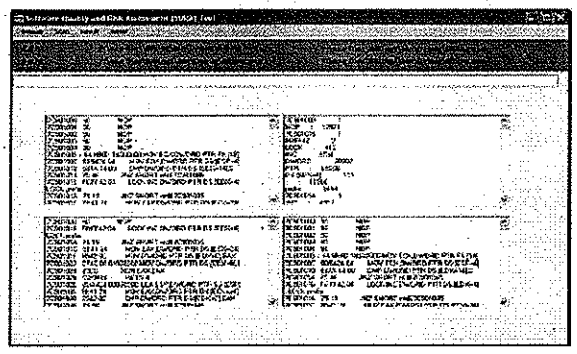
The experimental results show that the tool is been designed to check out the quality of the program by testing the .dll or. exe files.



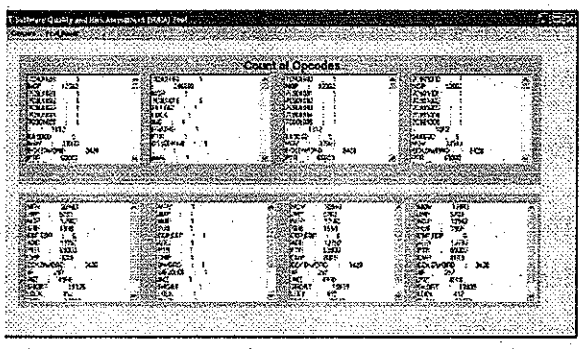
This tool is designed to check the step by step metrics that is been listed out.



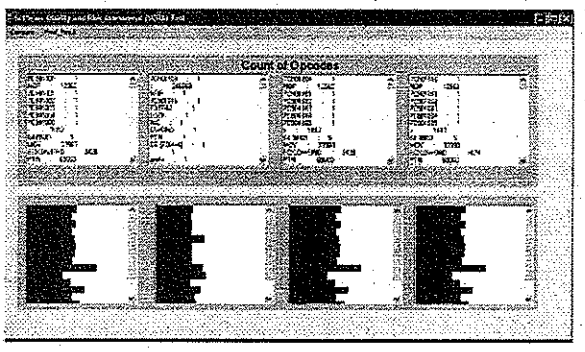
The number of opcode are been matched and been listed as per the training opcode.



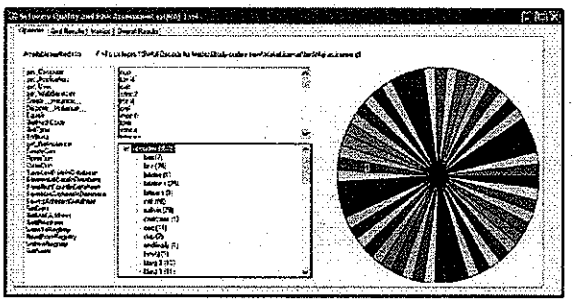
The count of the opcode for the exe or dll files is been listed out



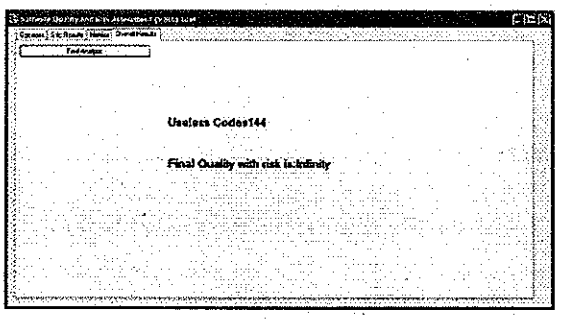
The risky files are been shown in red color to determine that the files have the malicious code and green color determines that the codes are out of risk.



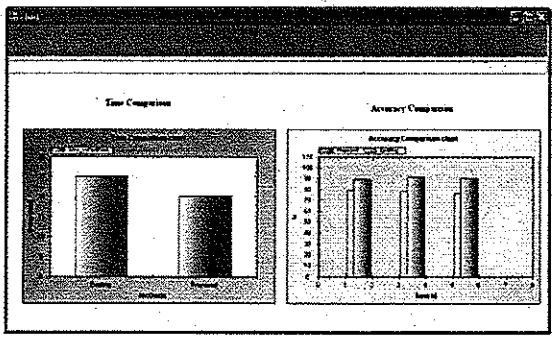
The chart shows that the no of trained opcode in the list.



The useless codes are been calculated and the final quality risk is identified.



The chart shows the comparison for existing process and proposed process and the accuracy for the process is been calculated by using chart.



V. Conclusion and Future Work

This paper has presented a new method for the tool that performed an extensive evaluation using a test collection comprising more than 30 methods with 4 assembly files. The evaluation consisted of three experiments. In the first experiment, this found hat the frequency and sequence representation then it will count the opcode and provides the results. For the future work the codes from the beginning of the process can be checked out by modernizing the tool so that the accuracy and time can be determined earlier for faster approach.

VI. References

[1] Abreu F B and Carapuca R, *"Object-Oriented Software Engineering: Measuring and Controlling the Development Process"*, Proceedings of the 4th International Conference on Software Quality, ASQC, McLean, VA, USA, 1994.

[2] Amit Sharma, Sanjay Kumar Dubey, *"Comparison of Software Quality Metrics for Object- Oriented System and Analysis"*, ISSTA'04, 86–96, 2004.

[3] Bansiya J. and C. G. Davis (2002) *"A Hierarchical Model for Object - Oriented Design Quality Assessment"*, IEEE Transactions onSoftware Engineering, pp. 4-17, 2002

[4] Briand, L.C., J.W. Daly, and J.K. Wust, *"A Unified Framework for Coupling in Object-Oriented Systems"*, IEEE Transactions on Software Engineering, v.25(1): p. 91-121, 1999.

[5] Chen, J-Y., Lum, J-F.: *"A New Metrics for Object-Oriented Design."*, Information of Software Technology 35,4(April 1993) :232-240.

[6] R. Sekar, M. Bendre, D. Bollineni, and Bollineni, R. Needham and M. Abadi, Eds., *"A fast automatonbased method for detecting anomalous program behaviors,"* in Proc. 2001 IEEE Symp. Security and Privacy, IEEE Comput. *Soc.*, Los Alamitos, CA, USA, 2001, pp. 144–155.

[7] A. Lakhotia, E. U. Kumar, and M. Venable, *"A method for detecting obfuscated calls in malicious binaries," IEEE Trans. Software Eng.*, vol. 31, no. 11, pp. 955–968, Nov. 2005

[8] Christodorescu, M. and Jha, S. (2003) 'Static analysis of executables to detect malicious patterns'.

[9] D. Bilar, *"Opcodes as predictor for malware,"* Int. J. Electron. Security Digital Forensics, vol. 1, no. 2, pp. 156–168, 2007.

[10] Clementi, A. (2007) Anti-Virus Comparative No. 13, Innsbruck, Germany, February, p.7.

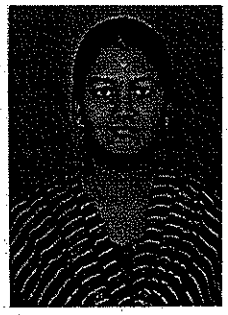
[11] Cai DM, Gokhale M, Theiler J: Comparison of feature selection and classification algorithms in identifying malicious executables. Computational Statistics and Data Analysis 2007, 51:3156-3172.

[12] Kim, S., Tsui, K. and Borodovsky, M. (2006) 'Multiple hypothesis testing in large-scale.

[13] Commtouch Inc. (2007) Malware Report: Server-side Polymorphic Viruses Surge Past AV Defenses, Netanya, Israel.

[14] Schultz M, Eskin E, Zadok E, Stolfo S: Data mining methods for detection of new malicious executables. Proc of the IEEE Symposium on Security and Privacy, IEEE Computer Society 2001, 38.

Authors' Biography



Keerthana. J was born in Coimbatore, Tamil Nadu (TN), India, in 1991. He has received the Bachelor of Science in Computer Technology degree from Dr. Mahalingam college of Engineering college and technology, pollachi which is affiliated to the Anna University, coimbatore, India, in year of 2011 and the Master of computer applications degree from Nallamuthu gounder mahalingam college, pollachi, affiliated to the Bharathiar University, Coimbatore, TN, India, in year of 2014. He is currently pursuing the Master of Philosophy in Sri Krishna Arts & Science College, Coimbatore with the Department of Computer Science, which is affiliated to the Bharathiar University, Coimbatore, TN, India. His research interests include software testing and quality testing.

Jeen marseline K.S received her B.Sc, MCA and M.Phil degree from Bharathiar University Coimbatore, TN, India. in 1995 respectively. She received her PhD from Avinashilingam university for women, Coimbatore, TN, India. in 2016. She is the Prof & Head of the Department of Information Technology, in Sri Krishna Arts And Science College , Coimbatore, TN, India. She has 18 years of teaching experience.She has more than 25 publications at national and International level. in Image Processing and Networking.