Prof. D.M.Thakore, Torana N.Kamble / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 5, September- October 2012, pp.1574-1577

Validating UML Diagram for Security

Prof. D.M.Thakore *, Torana N.Kamble.**
*(Department of Computer Engineering, Bharti Vidyapeeth Deemed University, College of Engineering, Pune)
**(M.Tech. Student, Department of Computer Engineering, Bharti Vidyapeeth Deemed University College of Engineering, Pune)

ABSTRACT

Now days there are different quality attributes of software artifacts but security has got less responsiveness because of different reasons. Like lack of knowledge about which properties must be considered when it comes to evaluate security because definition of security is different for different organization and accordingly they implement the security metrics. Secure software development is still research topic in many organizations, because of the failure in designing security at early stage. Traditional approaches focus primarily on antivirus, firewall, intrusion detection, but all are at the level of individual program statements and so on. This approach makes it difficult and costly to determine and repair weakness caused by design errors. It has been seen that flaws are left in the software design during development process are responsible for successful attack. The proper care should be taken at the design level only.

Proposed approach describes identification of secure design at early stage with the help of design level security metrics and Genetic algorithm (GA).

The reason of using this algorithm is to exploit previous and alternate solution and provides multiple solutions to speedup. Then it determines the security aspects of program during execution using set of metrics. It can be achieve by finding some method to log all occurrences of object instantiations, deletions, method invocations, and direct reference to attributes while the system is executing.

Keywords- Design, Measurement, Software Security, Security, Quality Metrics, Software Metrics.

I. INTRODUCTION

Due to the modularity and reusability many software projects are shifted from traditional structured development to object oriented design. In this Object Oriented approach metrics are useful tool to measure different quality attributes. Metrics are a means for attaining more accurate estimations of project milestones, and developing a software system that contains minimal faults. There is project base metrics which keep track of project maintenance, budgeting etc. whereas Design based metrics describe the complexity, size and robustness of object oriented and keep track of design performance. Mainly two kinds of metrics are used for object oriented design namely-Static metrics and Dynamic metrics. Static metrics are obtained from static analysis and dynamic metrics are computed on the basis of data collected during the execution of code. Traditional metrics for measuring software such as Lines of Code (LoC) have been found to be insufficient for analysis of object-Oriented software. The suite of metrics proposed by **Chidamber** and **Kemerer** are useful to measure static feature of code. These code metrics computes different aspects of complexity of the source code, but not able to accurately predict the dynamic behaviour of an application is as yet unproven. [8]

Evaluating the dynamic behaviour of an application at run time with static metrics is difficult because its behaviour will be influenced by the operational environment as well as complexity of object-oriented software.Different metrics have been developed for software quality attributes of object-oriented designs such as performance, reusability, and reliability. However, metrics which measure the quality attribute of information security have received little attention as security is non-functional quality attribute.

Moreover, existing security metrics measures the system at high level i.e. the whole system's level or at a low level i.e. the program code's level. These approaches are tough and costly to determine and fix weaknesses caused by software design errors. [3]

To overcome these difficulties design metrics have been developed which measures security at design level. Proposed system applies these design level metrics and gives secure design .The advantage of this technique is the cost and efforts needed to solve the problems after implementation get reduced as it discovers errors at early stage. And after implementing that secure design it can be tested by different dynamic metrics.

II. LITERATURE SURVEY

There are different ways of reducing security risks and vulnerabilities. But a common approach is to enforce security at the implementation stage only [5].

A survey has shown that most of the security metrics calculate the security at system level it considers system as whole. These are referred as a high level metrics. These metrics check

out not only software but also many other aspects of the system.[2]

Several projects have inspected information flow through computer program code, by type analysis, data/control flow analysis, and different other ways of identifying and eliminating program code vulnerabilities.[10] There are number of security metrics that assess the security of a given program based on code inspections. However, their metrics require full system implementations to assess security which makes it impossible to fix problems at design time. [4] Instead, the most efficient approach is to enforce security at early phases of the software development lifecycle such as during the design phase. The **National Institute of Standards and Technology** stated that eliminating vulnerabilities in the design stage can cost 30 times less than fixing them at a later stage. One of the earliest studies in this area was the development of software security design principles, these principles are intended as guidance to help develop secure systems, mainly operating systems, and are not capable of quantifying the security levels of designs. Thus, there is a need for security metrics which objectively measure the security of a given program directly from its design artifacts. [6] Study conducted by B. Alshammari et al. [3] defined different security metrics to identify secure design among different design which are obtained after refactoring.

III. EXISTING TOOLS AND COMPARATIVE STUDY

Following are the existing tools which measures quality of software based on defined metrics. Some of them can be applied at design level and some are at code level.[13]

Classycle: It analyses static class and package dependencies in java. It overcome limitation of JDepend i.e. it finds cyclic dependencies between classes and packages

JDepend: JDepend automatically measures quality of design by managing package dependency. It examines design and checks weather design shows specified quality or not during refactoring. But it has some limitations like Cyclic dependency detection, does not collect source code complexity metrics and it can't differs Java interfaces and Java abstract classes.

JHAWK: It is a stand-alone, Eclipse Plugin, command line version It useful to measure parameters like Cyclometric Complexity,NOP by computing loops and iterations existing in a program. Its disadvantage is Halstead metrics are not used for measuring the program. Limitation of this tool is it accepts only java code.

ES2: It is atool for collecting object oriented design metrics from C++ and Java Code.It is helpful for making quality management decisions in practices.

Chidamber & Kemmerer Java Metrics: It is an open source tool to access CK Object Oriented Design quality metric by processing the byte-code of compiled Java files. It is command line version and generates output in text format.

QJ-Pro: It is java review tool used to find the errors related with the language standards. It normally used during testing phase.

VizzAnalyzer: It is a quality analysis tool, reads software code and other design specifications as well as documentation and performs a number of quality analyses.

Semmle: Semmle is basically java testing tool used to enforce coding conventions by finding programming bug patterns, to compute software metrics. All these tasks can be formulated as queries in an object-oriented query language named QL.

OOMeter: This tool is useful for measuring each artifacts produced during software development life cycle, like requirements, specification, design model, source code, test specification**.**

Table I shows different existing tools with the different coupling metrics .There are design level tools as well as code level tools.

Table I – Supported metrics in Existing Available Tools

Tools	Metrics						
	CBO	WMC	NOC	RFC	COF	DIT	DAC
Classycle	---	---	---	---	---	Y	---
JDepend	---	---	Y	---	---	Y	---
ES2	Y	Y	Y	---	---	---	---
Semmle	---	---	Y	Y	---	Y	---
OOMeter	Y	---	Y	---	---	Y	---
VizzAnalyzer	Y	Y	Y	Y	---	Y	---
CKJ	Y	Y	Y	Y	---	Y	---

IV. COUPLING AND SOFTWARE SECURITY

Coupling is the interaction between different software components. It is an internal software property whereas security is external. But many studies have been shown that coupling is closely associated with the Security of software. The reason behind this is when information passed among different components there is more probabilities that it is exposed. It makes sense to assume that coupling is important factor that affects security of software. Thus, proposed system aims to

develop tool which device security metrics by taking into consideration this association.

Supported Coupling Metrics:-

Here are some of the coupling metrics that can be devised in proposed system[6]-

1. RFC (Response for a Class) is the number of the methods that can potentially be invoked in response to a public message received by an object of a particular class. If the number of methods that can be invoked from a class is high, it is more difficult and highly coupled to some methods.
2. WMC (Weighted Methods per Class) is defined as the weighted sum of all class's methods. It is a measure for the complexity of classes. More complex classes are more error-prone, harder to analyze and test. It is expected that complex classes have higher change rates because of bug fixing and refactoring activities.
3. CBO (Coupling Between Object Classes) is the number of classes that a class is coupled to. It is calculated by counting other classes whose attributes or methods are used by the given classes plus those that use the attributes or methods of the given class.If a class is highly coupled with other classes, changes in other classes can also cause changes in that class.
4. (DIT) Depth of Inheritance Tree: the maximum depth of the class in the inheritance tree. It measures the number of potential ancestor classes that can affect a class, *i.e.*, it measures inter-class coupling due to inheritance.
5. (NOC) Number Of Children: the number of immediate sub-classes of a class or the count of derived classes. If class *class A* inherits class *classB*, then c*lass B* is the base class and class *A* is the derived class. In other words, class *A* is the children of class *class B*, and *class B* is the parent of class *class B*. NOC measures inheritance complexity.
6. COF(Coupling factor):Coupling factor measures the actual coupling among classes .Maximum coupling accursed when all classes are coupled with each other. But maximum coupling leads to complexity which is in tern leads to need of security.
7. DAC(Data Abstraction Coupling): This metric measures the number of instantiations of other classes within the given class. It is not caused due to inheritance or the object oriented paradigm. The higher the DAC, the more complex the data structure (classes) of the system

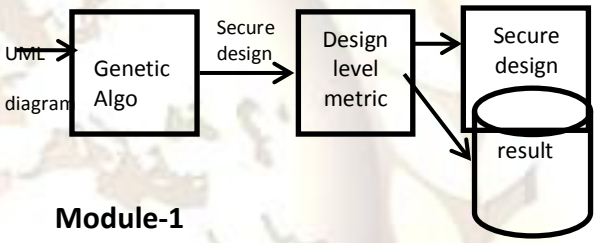
V. SYSTEM DESCRIPTION

From above study it is clear that there are different tools to measure the quality of software at design level as well as code level. Each tool includes different set of metrics to measure different quality attributes. When user uses any tool at design level to validate the design with the help of included set of metrics then there is need to assess the implemented code with same set of metrics.

Proposed tool facilitate user to measure security by applying tool at design level as well as code with same set of metrics. In this it concerns only one feature of object oriented design which is highly associated with the security i.e. coupling.

Proposed system is involves three modules, which works in sequential manner.

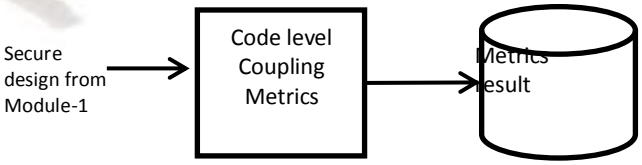
First module accepts UML diagram as a input and applies Genetic algorithm on it. Here GA is used to obtain more than one design which fulfills different levels of fitness function. These alternate designs are of three levels of security i.e. High secure, medium secure, low secure.



Module-1

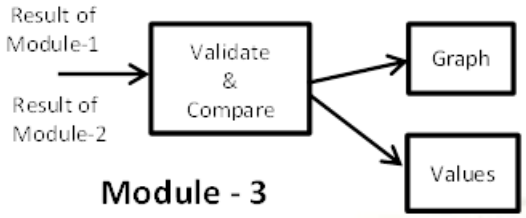
GA is a search in which successor states are generated from two parent states. It starts with randomly generated population. It evolves through three phases selection, crossover and mutation. Finally best solution is chosen based on the optimization criteria. Each state is assessed by fitness function. Then different Security related coupling metrics are applied on all levels of secure design and stores result.

In second module code is implemented for highly secure or medium or low secure design .Then this implemented code get evaluated based on the security related same coupling metrics. Means it checks coupling aspect at code level to provide security.



Module - 2

Third module is nothing but the validation of all designs. It compares metrics results computed at design level and code level to generate the graph.



VI. CONCLUSION

Existing software metric tools interpret and implement the definitions of object-oriented software metrics differently. This provides results based on tool-dependent metrics and has even allegations on the results of analyses based on these metrics results. In short, the metrics based assessment of a software system and measures taken to improve its design differ considerably from tool to tool. So the limitation is all tools are platform dependent. There are separate tools for dynamic metrics, static metrics and for reverse engineering.
But the proposed system develops a tool which applies dynamic as well as static metrics on design by considering coupling of classes and objects. And also applies reverse engineering to validate the design. Again in proposed tool uses genetic algorithm for alternate designs. Benefit to use this algorithm is it is easy to understand and multi object optimizer.

There is lot of scope in future work as the dynamic metrics are related with the behaviour of the program, they have advantage of more precise, but difficult to implement compare to static one .So there is clear opportunity for researcher to work in hybrid approach where dynamic results can be augmented by static information for collection of metrics data.

REFERENCES

[1] J. Bansiya and C. G. Davis, *"A hierarchical model for object-oriented design quality assessment,"* *IEEE Transactions on Software Engineering, vol. 28*, pp. 4–17, 2002 ..

[2] P K. Manadhata, K. M. C. Tan, R. A. Maxion, and J. M. Wing, *"An approach to measuring a system's attack surface,"* Tech. Rep. CMU-CS- 07-146, Carnegie Mellon University, Pittsburgh, PA, August 2007.

[3] B. Alshammari, C. J. Fidge, and D. Corney, *"Security metrics for object-oriented class designs,"* in Proceedings of the Ninth International Conference on Quality Software (QSIC 2009), (Jeju, Korea), pp. 11–20, IEEE, 2009

[4] I. Chowdhury, B. Chan, and M. Zulkernine, *"Security metrics for sourcecode structures,"* in Proceedings of the Fourth International Workshop onSoftware Engineering for Secure Systems, (Leipzig, Germany), ACM, 2008..

[5] Smriti Jain, *"A Review of Security Metrics in Software Development Process*" et al / (IJCSIT) International Journal of Computer Science and Information Technologies, 2011.

[6] IstehadChowdhury, Mohammad Zulkernine "*Can Complexity, Coupling, and Cohesion Metrics be Used as Early Indicators of Vulnerabilities?"* ACM 2010.

[7] S. Chidamber and C. Kemerer, *"A metrics suite for object oriented design,"* IEEE Transactions on Software Engineering, vol. 20, pp. 476–493, 1994.,

[8] M. Fowler, *Refactoring: Improving The Design of Existing Code*. Reading, MA: Addison-Wesley, 1999

[9] Payal Khurana&Puneet Jai Kaur *DYNAMIC METRICS AT DESIGN LEVEL ,International Journal of Information Technology and Knowledge Management* July-December 2009, Volume 2, No. 2, pp. 449-454

[10] AmjanShaik,C. R. K. Reddy, BalaManda, Prakashini. C, Deepthi. K, *"An Empirical Validation of Object Oriented Design Metrics in Object Oriented Systems"* Journal of Emerging Trends in Engineering and Applied Sciences (JETEAS) ,(ISSN: 2141-7016).

[11] John Lloyd1 and Jan Jürjens2, *'Security Analysis of a Biometric Authentication System 'Using UMLsec and JML*,* A. Schürr and B. Selic (Eds.): MODELS 2009, LNCS 5795, pp. 77–91, 2009.,© Springer-Verlag Berlin Heidelberg 2009

[12] M. Y. Liu and I. Traore, *"Empirical relation between coupling and attackability in software systems: a case study on DOS,"* in Proceedings of the 2006 Workshop on Programming Languages and Analysis for Security Ottawa. Ontario, Canada: ACM, 2006, pp. 57–64

[13] Rüdiger Lincke, Jonas Lundberg and Welf Löwe,"Comparing Software Metric Tools", 2008 ACM 978-1-59593-904-3/08/07.