A System for Identification and Assessment of Secure Design using Dynamic Security Metrics

[1] DEVENDRASINGH THAKORE, [2] DR. AKHILESH R. UPADHYAY

[1, 2] Research Scholar, JJTU, India,
Dept.of EC Engg., Sagar Institute of Research and Technology ,
Bhopal, Madhya Pradesh, India.

deventhakur@yahoo.com, akhileshupadhyay@yahoo.com

ABSTRACT : *Secure software development is still an underexplored topic in many organizations, who failed to design security in at the early stages of the development process .Traditional approaches to security focus primarily on antivirus, firewall, intrusion detection, and so on. Software security is a critical issue for today's computing departments, and yet in many cases, it has received little or no attention. Previous studies have discovered that security when taken into account at early stage system development helps in reducing many software vulnerabilities. It has been discovered 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.*
Measuring quality attributes of object oriented design (e.g. maintainability, performance) has been covered by a number of studies. However, these studies have not considered security as much as other quality attributes. Also, most security studies focus at the level of individual program statements. This approach makes it hard and expensive to discover and fix vulnerability caused by design error .Proposed system focus on identification of secure design at early stage and discover the vulnerabilities. It helps to determine the security aspects of program during execution using dynamic metrics. It also performs reverse engineering to validate the design.

KEYWORDS: Design Security, Software Vulnerabilities, Quality Attributes, Object Oriented Design, Reverse Engineering.

1. INTRODUCTION

Object oriented development requires a different way of thinking than traditional structured development. The main advantage of object oriented design is its modularity and reusability. Because of which software projects are shifting to object oriented design.

Object oriented metrics are used to measure properties of object oriented designs. Metrics are a means for attaining more accurate estimations of project milestones, and developing a software system that contains minimal faults. Project based metrics keep track of project maintenance, budgeting etc. 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 whereas 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 inadequate for analysis of object-Oriented software. In recent years many researchers and practitioners have proposed a number of static code metrics for object-oriented software, e.g. the suite of metrics proposed by Chidamber and Kemerer. These code metrics quantify different aspects of complexity of the source code. However the ability of such static metrics to accurately predict the dynamic behaviour of an application is as yet unproven.[8]

Static metrics alone may be insufficient in evaluating the dynamic behaviour of an application at run time, as its behaviour will be influenced by the operational environment as well as complexity of object-oriented software.

Several 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.

Moreover, existing security metrics measure either the system from a high level (i.e. the whole system's level) or from a low level (i.e. the program code's level). These approaches make it hard and expensive to discover and fix vulnerabilities caused by software design errors.[3]

2 SYSTEM DESCRIPTION

Here the input to the system will be UMLsec Diagram and the output will be the source code

generated. The system accepts UMLsec Diagram and applies different refactoring methods to generate alternate designs. To identify most secure design it will apply the design level static and dynamic metrics .Then generates the code for that secure design and compare with the design documents with the help of reverse engineering concept. [3]

Refactoring the class diagram to get multiple designs is the normalization of that class diagram. Refactoring is the process in which refactoring of the behaviour of classes is done to increase their cohesion and/or to reduce the coupling between them.

Proposed system identifies secure design based on two design principles.

1. Least Privilege:- The programs and users should run with the least Privilege to complete their job.
2. Reduce Attack Surface: - This Principles aims to limit access to secret data. [3]

Coupling is one of the most important software design properties. It is defined as the interaction degree an object has with other objects, measured by the number of links it has with these objects. The impact of coupling on security shows a strong correlation between coupling and a system's attack ability.

Following are the metrics Used in Proposed System-

1) CCC (Critical Classes Coupling) - which finds degree of coupling between classes and classified attribute in given class diagram. Here classified attribute is a attribute tagged with secrecy i.e. confidential data. It uses possible links with classified attributes in a given design based on directed weighted link theory.

2) CBO (coupling between object classes) -Use links between classes define the detailed architecture of the application, just as use links between packages define the highest level architecture. These use links play a determining role in design quality, notably in development and maintenance facilities.

3) RFC (response for class)-The metric called the response for a class (RFC) measures the number of different methods that can be executed when an object of that class receives a message (when a method is invoked for that object). Ideally, we would want to find for each method of the class, the methods that class will call, and repeat this for each called method, calculating what is called the transitive closure of the method's call graph.

Steps Involved:-

1. Input for the system will be UML diagram (Design D).
2. Apply the refactoring algorithm to get multiple design of same diagram D i.e.D1, D2... Dn.
3. Apply design level static and dynamic metrics on all designs (D1,D2...Dn) to get secure design based on result of design level metrics.
4. Then implement that secure design (Map that design into source code)
5. Apply reverse engineering concept for the validation of secure design.

3. SYSTEM SCOPE AND REQURIEMENTS

- **Use of UMLsec**

UMLsec is a UML extension for secure systems development. It uses the standard UML extension mechanisms, and can be employed to evaluate UML specifications for vulnerabilities using a formal semantics of a simplified fragment of UML. Established rules of security engineering can be encapsulated and hence made available even to developers who are not specialists in security. [3]

- **Extraction of different UML Diagrams**

Static metrics alone may be insufficient in evaluating the dynamic behavior of an application at run time, as its behavior will be influenced by the operational environment as well as complexity of object-oriented software.[10] In order to develop static and dynamic metrics two different UML diagram are necessary .

For static metrics class diagram is sufficient to evaluate static features of the code. But for dynamic metrics dynamic behaviour is need to be identifying with the help of sequence diagram.

A sequence diagram in a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. [3]

- **Generation of alternate designs using refactoring.**

A refactoring is a parameterized behavior-preserving program transformation that updates an application's design and underlying source code. A refactoring is typically a simple transformation that has a straightforward (but not necessarily trivial) impact on application source code. Refactoring improves design, makes software easy to understand. Poorly designed code usually takes more code to do the same things, often because the code quite literally does the same thing in several places. Thus an important aspect of improving design is to eliminate duplicate code. The importance of this lies in future modifications to the code. [9] These feature of refactoring can be used to relate it with requirements of a certain security design principle. It provides alternate design that can be used for the same design using different refactoring rules like Encapsulate Field, Inline Method, Extract Method, Inline field.

4. CONCLUSION AND FUTURE WORK

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.

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.

5. 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 source code structures," in Proceedings of the Fourth International Workshop on Software Engineering for Secure Systems, (Leipzig, Germany), ACM, 2008.

[5] Amjan Shaik,C. R. K. Reddy, Bala Manda, 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)

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

[7]Istehad Chowdhury, Mohammad Zulkernine "Can Complexity, Coupling, and Cohesion Metrics be Used as Early Indicators of Vulnerabilities?" ACM 2010.

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

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

[10]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