A Roadmap for Security

Dr. Raees Ahmad Khan[1], Suhel Ahmad Khan[2]

Department of Information Technology

Babasaheb Bhimrao Ambedkar University, Lucknow

{khanraees@yahoo.com, ahmadsuhel28@gmail.com}

Abstract

The matter of security is important. It is a best approach to provide security of software at early stage of development. Design works as a foundation. Object oriented approach is used to represent the problem domain to clear to develop and understand design. This paper defines complexity as a key factor of security. By proper adjustment of design characteristics and relation among them, complexity can be controlled for acquiring secure software.

Keywords: *Security, complexity, inheritance, cohesion, coupling, polymorphism, association, aggregation, dependency*

1. Introduction

This is modern age where technology is rapidly changing. Human's lives are influenced with the chronological changes. Peoples accepting these changes and hoping that they are reliable, but the failure figure tells the story that there is much to do conserve the sanctity of security and to avoid the repercussions of it. Security is a complex quality that means it needs to be defined by a set of attributes not a single dimension. Security measures are different for different organization. In military it is confidentiality, in media it is integrity measured in terms of security is., in business it is availability. Security as a state or condition is resistance to harm. Security is a system property [1]. Security measures are the concept of protection from intruders and resist system from attack surface.

To develop a secure system is the contributory process of different steps and reflections of each phase is the matter of study to quantify the accurate impacts of security. Providing security is a continuous process for every phase in development life cycle. Microsoft reports that more than 50% of the security related problem for any firm has found at design and architectural level. [2] The designing phase of software engineering is the most beneficial for security practices. The identification of the common designing problems of security in early stage of development helps to analyze the risk evaluation and threat analysis. It also provides better prevention techniques for security measures. Design review validates the requirements and implements the design description. The identification of design flaws by reviewing design phase may reduce the same in further phase of development cycle. The flaw which arises at design like *curse* for the other phase of development life cycle, which is apparent matter of security. As a matter of well established facts, design affects security. Object oriented technology is the emerging trend for software design and development. Object oriented technology provide product with high quality and lower cost. [3].To increase the productivity of software it is mandatory to design and develop secures software. The design hierarchy and relation among them is the key factor of object oriented design.

Design complexity is not only the factor that makes things hard to understand but with enough complexity anything can become harder to understand.[4] It means enhanced complexity increases the chance of fault and attack surface for intruders. Complexity should not be exceeding beyond a certain limit. To increase the security of software it appears mandatory to decrease design complexity of software to identify faults and attack surface. Design complexity of software can be optimized by a proper alteration of design attributes and their relations.

This paper establishes a relation between design complexity and software security. Impacts of design complexity on software security are discussed in section 2. Section 3 discusses the issues with the concept of object oriented deigning approach. Section 4 summaries the contribution. Section 5 is the conclusion part.

2. Is There any Relation Between Complexity and Security?

McGraw also points out as one of three major factors of software security problem the other causes together are Complexity, connectivity and extensibility [5].Design

complexity has a negative impact on security beyond certain acceptable level. At the level of simplicity where software is easy to design and understand, a low rate of faults but maximum possibilities of attack. As design complexity increases the possibilities of faults increases and decreases the security of software. The security of software increases when design complexity decreases but the quality of product should be very competitive. Fig 1 shows a relation between complexity and security. The design complexity impacts on software security. It is a mandatory step to manage design complexity for secure software. Increasing complexity reduces the probability that product development will be successful at fixed level of investment.

Security incidents reported to the computer emergency readiness team coordination center (CERT/CC) that reported vulnerabilities are rapidly in increasing trends. These faults can impacts on design issues, cost and security. [2]. Due to unprecedented growth in code of different versions of window operating system it is very difficult to avoid bugs at 40 million lines of code and complex design. These issues are highly remarkable for security.

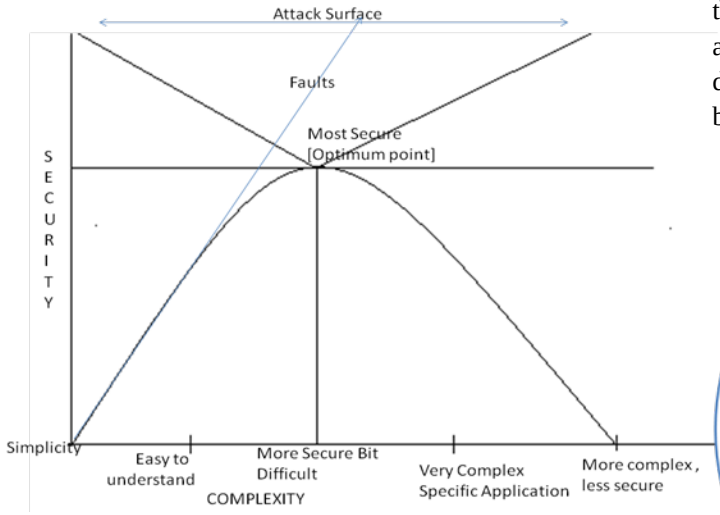

Fig. No.-1

$$S \propto \frac{1}{C} \qquad \text{¥} \qquad C > 0$$

Where, S = Security,
C = Complexity

The design complexity of components lengthens the product development cycle. The profit of the product becomes half due to required no of functions and design iterations at product launch time. A case of HP desk jet 500 series printer loses its profit half due to such extension of development cycle. [2,6]

Another issue related to security that software must be reliable. An unnoticeable error converts into disaster. In 1991 in gulf war, the chopping error that missed 0.000000095 second in every 10^{th} of second, accumulating for 100 hr. made the patriot missile fail to intercept a scud missile.[7]

3. Complexity: In the Mirror of Object Oriented Design Concept:

Object oriented technology has become increasingly more popular language for software development environment. This technology supports to provide software product more secure and more reliable at lower maintenance cost. The object oriented paradigm starts by defining classes (types) of objects that contains related attributes and operations. These classes maintain a hierarchical tree for objects which inherits all the attributes and operations of its parent class. An object can communicate with each other through message passing when certain messages are passed between two objects the objects are coupled.[8] Object oriented design supports the design characteristics abstraction, information hiding, inheritance, polymorphism, cohesion, and coupling and relation among them through classes like association, dependency, qualified association, aggregation, generalization.[9] Fig 2 reference that design complexity is influenced by these attributes and it can be controlled by proper adjustment of these qualities. [10]

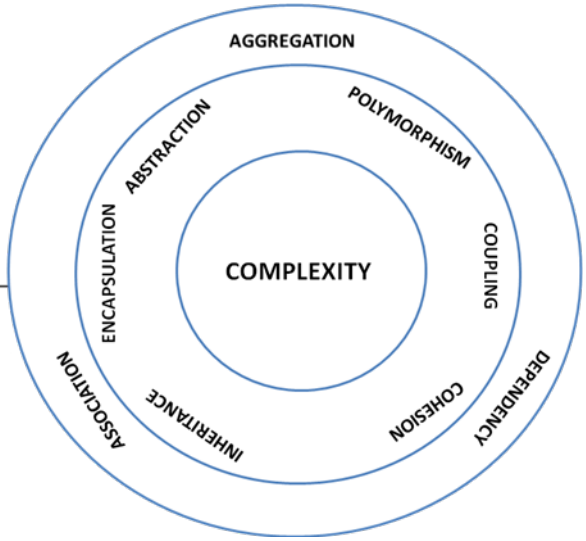

Fig.No.-2

A) Abstraction:

Abstraction is the process of arranging the details of a system so that it can focus on the important details. Abstracting a system can reduce its complexity for each parts of design because it can hide and remove the irrelevant details from each

part of design consideration. When it abstracts something, it hides some details to gain complexity and development productivity advantages.. Abstraction reduces complexity of designing process which is helpful to enhance the security of software at design.[9,11]

B) Encapsulation:

A basic property of an object which wraps the data structure and behavior in a single unit to support a well defined abstraction. Encapsulation provides the freedom to implement the abstraction with interfaces. A class is encapsulated if its internal structure is hidden from other classes. Other objects can use an encapsulated object without knowing the inner workings. The implementation of an object can be changed without affecting any object as long as the interface is preserved. It is a best way to insure the integrity of data is preserved. [9,12]. Limiting the use of class data or related methods, this minimizes the design complexity. This prevents unauthorized access there by giving an additional security to existing application.

C) Inheritance:

In inheritance super class accumulates all the common feathers of subclass. Inheritance forms 'as –a relationship' implying that an object of sub class is also an instance of super class. It also promotes reusability to inherit the feathers of existing class in new one. It forms inheritance hierarchy to find the level of nesting between classes. The more class in hierarchy, the greater no of methods is likely to inherit. Deeper trees constitute greater design complexity and maximum possibilities of having faults, attack surface. At the root class when depth of inheritance is zero fault-proneness will be negligible because of less complexity but due to increased inherited methods and classes design complexity will increased with more faults and attack surface.[8]

D) Polymorphism:

The concept of polymorphism plays an important role to built flexible system. Each objects having the ability about what and how the task will execute. Many forms of a single object are called Polymorphism. It is an ability to take several forms with same method. It allows that an operation on object can be implemented in different ways in different classes that operation depends on that object. Polymorphism is used to reduce complexity of design to preserve the semantics of operations within the object and provide common interfaces to types of object that are similar. [13]

E) Coupling:

The basic concept of coupling is that object interacts to each other that mean the strength of interconnection between objects. If there are two objects and methods of first object uses method or instance variables of another object then these two objects are coupled.[8] strong interconnections shows high coupling between modules which increases the complexity of design, efforts and faults.It is easy to modifiable if the less coupling of the class with other classes. Coupling between object should be minimum for a better understandable design to enhance security.

F) Cohesion:

Cohesion refers to the internal consistency within the parts of design. Cohesion is centered on data that is encapsulated within an object and how methods interact to provide a well defined behavior. Degree of similarity of methods is a major aspect of object class cohesiveness. The objective is to achieve maximum cohesion. It is a property that specifies how much elements are tightly bound to one another. To support a well defined abstraction it is mandatory to have all the elements of each class should highly cohesive. [14] Low cohesion increase complexity. Classes with low cohesion could probably be subdivided into two or more sub classes with increased cohesion.

G) Relationship Between Classes:

A complex system is composed with many objects of different classes. In object oriented design the objects interact to each other to achieve the goal. They use the services of others through message passing. The goal of design is to define the relationship between objects so that it can be implemented properly. The relationship is generally association, aggregation, and dependency. If an object uses the services of other object then there is a link between these objects. Such type of link is called association between two objects. For designing concerned an issue of visibility impacts on association that which object should be visible to whom. Higher order association is more difficult to draw and implement. [15] At design phase association may be evolved as dependency, aggregation or generalization.

Aggregation is a special type of association depicting the whole /part –of relationship between objects. The objects can be representing like a component or entire assembly. Like if an object A is an aggregation of object B and C, then object B and C will generally be within object A which implies containment.

Dependency is the concept to define the dependency relationship between objects. It shows the affects of changes from one class to another. [16]

A class is having a limit to facilitate at once. Smaller the no of classes can more secure, reliable and reusable than large complicated classes, but what is the exact number to break a class is very difficult to predict. Thumb rule says that break the class if attributes, associations are divided into two or more different groups that are not related to each other. [17] These relations impact on design complexity. To reduce design complexity and provide a better security mechanism it is must to adjustment of these relations with classes and better implementation approach for design characteristics.

4. Purpose:

The security of software can be achieved by controlling the complexity. The study present in this paper strengthen that Complexity is a security factor. There has to be always a balance between the complexities of design with respect to security. Security should be so important but design complexity never goes beyond normal acceptable level which looses the model effectiveness in terms of financial viability, user friendliness and in other terms the overall market value of the product. The determination of work as follows:

- Complexity is the key factor of security has been identified.
- Complexity, a key factor for security is major cause of software security issues.
- Complexity can be controlled by proper adjustment of design characteristics and relation among classes at design level.

5. Conclusion:

The design of a software system is the most critical factor affecting the security of software. It can't ignore those factors which impacts on security at design level. Complexity is a factor which affect on security most. By controlling the design characteristics and their relation it is possible to develop a product which is more reliable and secure.

ACKNOELEDGEMENT:

This work is sponsored by University Grant Commission (UGC), New Delhi, India, under F.No.34-107\2008 (SR).

References:

1. Gary Mcgraw: Software security: building security in Addition Wesley software security series.
2. Davis, N.; Humphrey, W.; Redwine, S.T., Jr.; Zibulski, G.; McGraw, G. Processes for producing secure software Security & Privacy ,IEEE Volume 2, Issue 3, Page(s):18 – 25, May-June 2004
3. R.A.Khan, K. Mustafa, Fault proneness model for object oriented software: design phase perspective, Information Technology Journal, 2008.
4. http://www.codesimplicity.com
5. Y.Shin, is complexity really the enemy of software security? Conference on Computer and Communications Security ,Proceedings of the 4th ACM workshop on quality of protection, Pages: 47-50,2008
6. Adekunle Fagade,Deepak kapoor, David Kazmer, Deptt. Of Mech & Ind. Eng., Univ. Mass. Amhert. A discussion of design and manufacturing complexity,
7. Jiantao Pan, Software reliability, Carnegie Mellon University, 18-848b, , Dependable Embedded Systems, 1999.
8. Chidamber, S.R.; Kemerer, C.F, A metrics suite for object oriented design Software Engineering, IEEE Transactions on Volume 20, Issue 6, Page(s):476 – 493, Jun 1994
9. Pankaj Jalote: An integrated approach to software engg., Second Edition,Narosa Publication House.
10. R.A.Khan, K.Mustafa, S.I.Ahson,"An empirical validation of object oriented design quality metrics" j. king saud University, Vol19, com & info Sci, , p1-16, Reyadh,2007
11. http://www.edn.com/article/CA426084.html.
12. Alan Snyder "Encapsulation and inheritance in object oriented programming language" OOPSLA 86 Proceedings, ACM, , p38-45, 1986
13. Rogenberg Linda ,"Software quality Metrics for Object Oriented system Environments" A report of SATC's research on OO metrics.
14. Jehad Al Dallal," A design –Based Cohesion Metric for Object Oriented Classes", International Journel of computer science and Engg. Volume1 nov.3, p.193-200.
15. James Rumbaugh, Object – Oriented Modeling and Design , Pearson Education,Inc.
16. Dazhou Kang Baowen Xu Jianjiang Lu Chu, W.C.**,** A complexity measure for ontology based on UML, Distributed Computing Systems, Proceedings. 10th IEEE International Workshop on Future Trends Publication Date:26-28 On page(s): 222- 228, May 2004
17. Rules of Thumb for Developing Secure Software: Analyzing and Consolidating Two Proposed Sets of Rules, Proceedings of the 2008 Third International Conference on Availability, Reliability and Security,Pages:1204-1209,2008