# Journal of Software Engineering

ISSN 1819-4311

Academic Journals Inc.

www.academicjournals.com

Journal of Software Engineering 2 (1): 39-46, 2008
ISSN 1819-4311


# Object Oriented Software Security Estimation Life Cycle-Design Phase Perspective

S. Chandra and R.A. Khan
Department of IT, BBA University, Lucknow, India

**Abstract:** Security upgrading is possible with the help of security attributes, security metrics and models. Unifying attributes, metrics models and tools a security estimation life cycle has been proposed in this study. Security estimation is needed to identify and mitigate security threats, holes and attacks. In absence of any standard framework or model to estimate software security, it appears worthwhile proposing a methodology to predict software security early in the development life cycle. It has been observed that security estimation at early stage of development life cycle assist developer to mitigate vulnerability and to produce highly secured software. In addition, early detection of vulnerabilities, threats, worms and attacks reduces cost, time and rework. The proposed lifecycle is yet to be implemented in order to analyze the tryout data and to verify the effectiveness.

**Key words:** Software security, software metric, security estimation, software design, security attributes, security metric

## INTRODUCTION

Computer and internet are now playing an important role in every aspect of human lives including transactions, information storage and its retrieval. Software systems are used in every sector including government, military, aerospace etc. In such a scenario, it is desirable to keep all sensitive data, information, transaction safe from security breaches. All users of computing and information system expect these systems to perform their task timely and accurately. No doubt, there is great demand of having such a system in order to perform secured action to prevent any type of data loss, theft, of misuse. In order to achieve the objective, it is required to design and incorporate security policy and measures well in advance.

Developing secured software is not an advantage but has become a necessity for software organization. Measuring security plays an important role in order to mitigate vulnerabilities while producing security end software. Security estimation is also needed to identify and mitigate security threats, holes and attacks. No doubt, security estimation helps in reducing the development cost and building confidence among end users (Wang and Wolf, 1997). It is inevitable fact that security must be estimated to come up with the non-vulnerable software. But, the major problem of introducing estimates is where to measure security vulnerability during development. As a matter of fact, early estimates will assist in producing cost effectively secured software within the time and with the use of optimal resources (Khan and Mustafa, 2008).

Software design is a backbone of any software. Software design serves well as a communication medium between the designer and the user on the one end and act as a basis for implementation on the other end. Design is an important stage spanning the whole software lifecycle, not only for software development but also for re-developing legacy systems (Peterson, 2004). It is concerned with accurately mapping the requirements from the analysis stage to logical models for implementation. The

**Corresponding Author:** S. Chandra, Department of IT, BBA University, Lucknow, India

security estimates at software design heavily affects the security of the final products. Controlling and improving software design security has been one of the important issues in software security engineering.

Recently object-oriented technology is becoming increasingly popular in industrial software development environments. This technology offers support to provide software product with higher quality and lower maintenance costs. Object oriented software is different from traditional in many respects. Therefore, the available methodology for security estimation of traditional software may not be appropriate for object oriented software. Hence, there appears a need to develop a mechanism to quantify object oriented software security early in the development life cycle.

## SOFTWARE SECURITY

Software security is to protect software from unauthorized access and malevolent approach. It is an idea of engineering software so that it continues to function correctly under malicious attack. Software security implies the protection of software assets such as application programs, the operating system and stored information. Software security takes care of both issues including security mechanisms and design for security (McGraw, 2004). Obfuscation, watermarking and code signing technologies are the defensive technologies used by software developers to improve software security (Stytz and Whitaker, 2003).

Software security is the process of building, designing, testing and implementing the software (McGraw, 2004). It provokes the developers to build secure software which performs well under malicious attacks, bugs, threats, viruses etc and also tackles with unauthorized access. Software security feature can not be added through the addition of sets of features, it must be design and integrated with the every phase of the software development life cycle (McGraw and Mead, 2005). Only penetration testing or penetrate and patch are not sufficient enough for the purpose of security. The existence of vulnerabilities in the software reflects that the software can be compromised at any point of time. The more vulnerability a software has the more attack prone it is. It is required to implement it from the ground of the software development. Security concerns should be based on product security goals and attention must be given to security of sensitive information (Sahinogly, 2005).

## SOFTWARE SECURITY ESTIMATION

Software security estimation is required to assess performance and the degree of protection (Stytz and Whitaker, 2003). Undesirable threats, takes advantage of hardware and software weaknesses or vulnerabilities can impact the violation and breakdown of availability, integrity, confidentiality and nonrepudiation as well as other aspects of software security such as authentication, privacy and encryption (Sahinogly, 2005). An IT industry suffers grievous damages due to the lack of security estimation before software deployment. Recently slammer worm affecting SQL servers because of software vulnerability (Scandariato *et al*., 2006).

Security team can collaborate during design phase to make software secure. During design phase, software is in extremely malleable phase. At the end of the software development to implement security increases the complexity and cost of making changes (Peterson, 2004). Security should be integrated and treated on a par with other system properties. The only way to develop systems with required functionality and performance that can also withstand malicious attacks is to design and implement them to be secure. By treating software security risk explicitly throughout the software life cycle, one can properly identify and mitigate the consequences of security failure and identify security attacks successfully.

Using the concept of software security estimation during development of software, security can be measured by analyzing the design activities. Measurement of security attributes and its impact on software, security team may improve and control software security. This will affect the performance and quality of software.

## RECENT DEVELOPMENTS

It is evident from literature survey that plenty of work has been done in the area of software security. In the following section, some pertinent recent works have been discussed with their findings in order to strengthen the need and importance of having security measures early in the development life cycle (Chandra and Khan, 2008).

In 2008, Beznosov and Chess presents a report on Security for the Rest of Us: An Industry Perspective on the Secure-Software Challenge. This report is based on the state of the practice and recent advances in engineering secure software for the wide range of industrial application domains. Author discussed the needs and importance of security requirements analysis and threat analysis and suggested that developer needs to properly collect and analyze security requirements (Beznosov and Chess, 2008).

Reijo Savola, introduce a high abstraction level taxonomy in 2007, supports the development of feasible security metrics, along with a survey of the emerging security metrics from the academic, governmental and industrial perspectives. Using this classification the author employing efforts to bridge the gap between information security management and ICT products and services security engineering. It would be easier to make business and engineering decisions concerning information security. Security measurement within RandD organizations should make a move from ad hoc practices to a more systematic process, because business demands change faster. The taxonomy can be used as a basis for developing composite or hierarchical security, trust and dependability metrics that are aligned to the common business objectives and it also offer realistic security evidence for different user groups-business management, information security management, product, system and service security management and technical system developers (Savola, 2007).

Nichols and Peterson in 2007 (Nichols and Peterson, 2007), introduces a metric framework which upgrade security of software applications. Author suggested that enterprises must focus on the security of the web application itself. In this study they discuss the importance of design-time metrics. Design time metrics have ability to identify and categorize weaknesses at early stage of software development life cycle. Authors suggested that the developer also need to concern what vulnerabilities may exist in software.

Scandariato *et al.* (2006) tried to give shape to the idea of security properties of software that are quantitative in nature with regard to assessment, allow proactive estimation of software security, especially during the architecture/design phases and can be measured at different levels of abstraction. In a paper entitled Towards a Measuring Framework for Security Properties of Software, published in the Proceedings of the 2nd ACM workshop on Quality of protection they analyzed security principles that are relevant to the purpose of unearthing security properties and proposed suitable metrics to measure them.

McGraw and Taylor in 2005 worked on security improvement program and concluded that generally standard software process approaches focused on sequentially building a level of sufficiency in four areas in a particular order: process, controls, metrics and improvement. Unfortunately, following these basic steps in the prescribed order implies that we don't address metrics until late in the program. By then, it might be too late because processes and controls put in place early on might not be properly designed to provide the kinds metrics needed later. In such cases, some significant rework might be required to achieve business alignment. Consequently, it's now generally agreed that measurement and analysis must be included much earlier in the process development model (McGraw and Taylor, 2005).

A team of researchers from Pennsylvania State University, Polytechnic University and SAP, in 2004 worked on Security Scoring Vector (S-vector) for Web Applications. The proposed S-vector metric will be used rate a web application's implementation against its requirements for technical capabilities (i.e., security functions), structural protection (i.e., security properties) and procedural methods (i.e., processes used in developing, validating and deploying/configuring the application) in order to produce an overall security score (Barton *et al*., 2004).

In 2003 Stytz and Whitaker presents three form of protection: watermarking, obfuscation and application performance degradations. These techniques perform three main functions: detection of attempts to pirate, misuse, or tamper with software against those attempts and alteration of software to ensure that its functionality degrades in an undetectable manner if protection fails. Four primary research areas: algorithms, environments, benchmarks and metrics and integration in the application security fields have been discussed. Author suggests that application security techniques and damage mitigation practices must be part of every application developer's toolkit (Stytz and Whitaker, 2003).

An exhaustive review of the literature concludes that there is no software security estimation tool or model available to quantify security of software. Therefore, there is a high demand to develop such a tool to estimate security in early stage of software development life cycle. Norman (Scheneidweind, 1992) establishes a framework which puts together the concepts and definition of quality factors, quality metric, validated metric, quality function, validity criteria and a metric validation process. It has been inferred from the Norman's work that there is a possibility to develop a framework for the development of secured software.

## THEORETICAL BASIS

There are different standards available including Trusted Computer System Evaluation Criteria (TCSEC), Information Technology Security Evaluation Criteria (ITSEC), Canadian Trusted Computer Product Evaluation Criteria (CTCPEC), Federal Criteria for Information Technology Security (FC), which specify several criteria against which software security evaluation can be made (Savola, 2007; DoD, 1985). One more interesting guideline series is VAHTI introduced by Finnish Government Information Security Management Board addresses security management and security auditing (Kajava and Savola, 2005).

Security of the software can not be measured directly. It can be measured with the combined use of models, metrics and attributes (Web Reference 1). In security certification process, it is required to measure the specific security non-functional attributes of the software (Web Reference 2; Copigneaux and Martin, 1988). The analysis of multiple security attribute and their tradeoffs will yield insights into system's strengths and weaknesses and provide basis for carrying out cost and benefit analysis (Madan *et al*., 2002). Wang and Wulf have given a framework for security measurement. They strongly felt that absence of security vulnerability does not guarantee the risk free system (Wang and Wolf, 1997).

The foregoing discussion provides a strong theoretical basis in order to develop a mechanism to estimate security early in the development life cycle. Security practitioners and researchers strongly believe in estimating security with the development life cycle in a quantitative manner.

## THE LIFE CYCLE

Security of the target system is guaranteed if formal and mathematical methods are used, but it makes the system engineering so complex process that the formal methods are seldom used to improve system security. Formal Methods should be recognized during system security engineering to provide tools for deeper security analysis if required.

It is long-familiar paradigm that an activity can not be controlled if it can not be measured. Software security also comes in this rubric. Until now, security assessment process is carried out after

development of software using qualitative criteria by security experts, which is an expensive affair. The problem lies in, knowing how and when it should be measured (Khan and Mustafa, 2008). These situations make way for a more rigorous approach, able to estimate security in design phase of software development life cycle. In order to estimate software security quantitatively a Security Estimation Life Cycle is proposed and has been depicted in Fig. 1. Three Stages of the life cycle are:

**Stage 1:** Input process
- HLD/LLD

**Stage 2:** Security Estimation Process
- Identify Security Factors
- Identify/Design Metric Suite
- Validate Metric Suite
- Quantify Security Factors
- Estimate Security

**Stage 3:** Output Process
- Qualitative Analysis
- Overall Security analysis

The first step of the life cycle sets out for input profiling. High level Diagram (HLD)/Low Level diagram (LLD) may be given as an input to the process. The second step consist security estimation

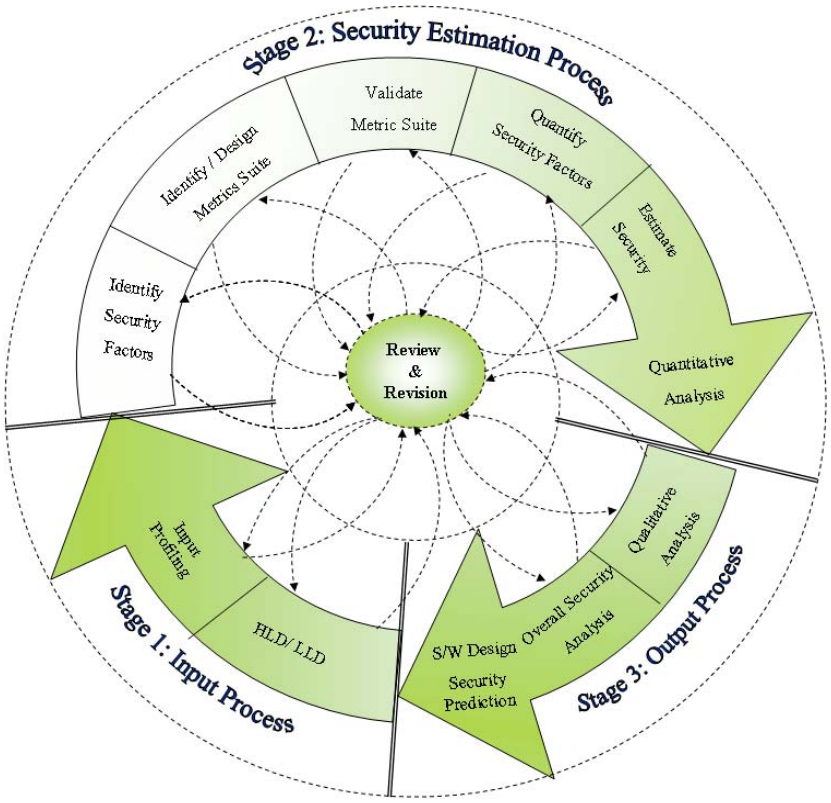


Fig. 1: Security estimation life cycle

process and are performed in five phases. The first phase is the identification of security factors. Identify/Design metric suite is the second phase, followed by the phases validate metric suite, quantify security factors and estimate security, as a third, fourth and fifth phases, respectively. Quantitative assessment of security will provide the base for qualitative analysis and overall security analysis. Security Estimation Life Cycle provides roadmap to developer or industry personal to estimate software security in design phase. Goal of software Security Estimation Life Cycle is to provide high level protection to the software and contribute to the mitigation of security failures.

## MOTIVATIONS

Development of security architecture for software is not a one time built process; it is based on reuse of existing security specifications. There is a common agreement between researchers and security practitioners to integrate security at the preliminary stage of software development life cycle in order to develop secured software (Nichols and Peterson, 2007). In 2007, Shirley C. Payne at SANS Institute remarks that metrics can be an effective tool for security managers to discern the effectiveness of various components of their security programs. Further, it was stated that metrics can also help identifying the level of risk. Seven key steps to guide the process of establishing a security metric program has been proposed (Payne, 2007).

Security upgrading is possible with the help of security attributes, security metrics and models. Unifying attributes, metrics models and tools a security estimation life cycle has been proposed in this study. Security estimation life cycle has been carried out in three phases. The life cycle provides guideline to estimate security of the software at the early stage of life cycle. Security estimation life cycle may provide roadmap to practitioners and researchers to develop security estimation model and tool to quantitatively estimate software security.

## DISCUSSION

This study presents a security estimation life cycle for quantitative estimation of software security. The study undertaken is a step towards the security estimation of software in early phase of the development life cycle. Heterogeneous architecture of software does not allow estimating security at a glance. To solve the purpose a structured approach is proposed to model security attributes and metrics in such a way that security of the software at early stage is measured and captured. The work deals with identification of security factors and security metrics to identify vulnerability, to mitigate risks and to facilitate the achievement of the secured software systems. The proposed framework is different from the models and frameworks available in literature in various ways, including the followings:

- The proposed life cycle is prescriptive in nature and may be used for securing the software system in general rather than any specific system
- The proposed framework produces a list of available security factors, security metrics in order quantify security
- Literature survey strengthens that none of the security experts and industry professionals have made an effort in quantifying security early in the development life cycle. The proposed model provides a generic guideline to estimate security in early stage
- There is no security estimation life cycle available to quantify security. It seems to be worthwhile proposing a life cycle to accomplish the task
- There is no mechanism available in literature to assign ranks to software at security level. The proposed framework will help to find out the contributions of each identified factors to overall security

Future goal of the work is to identify security factors and design security metrics for measurement of security attributes in order to quantify security of the software. All stages of proposed life cycle are quasi-experimental in nature. Much cannot be said about the life cycle before empirical validation and tryout. The next step of the work will be to experimentally validate the proposed life cycle and assess the efficiency. These experiments should provide us with a better understanding of the behavior of the software from security point of view. It may help us to derive decisions that need to change in the architecture in order to improve security.

## CONCLUSION

A fundamental goal of software development is to deliver highly secure products that are correct, consistent and complete. This goal has driven researchers to investigate and develop software security estimation methodology that can support effective design and analysis of non-functional properties. One of the major advantages of introducing security estimation life cycle is that, the life cycle may detect and mitigate vulnerabilities earlier and in turn reduce development time and cost (Dai and Cooperb, 2006; Orlandi, 1990). The quantitative estimates of software security will assist in comparing, contrasting and making quantifiable statements. Quantitative results may provide the basis for improvement and control on software security at the early stage of life cycle. Much cannot be said until the life cycle is implemented.

## REFERENCES

Barton, R.R., W.J. Hery and P. Liu, 2004. An S-vector for Web application security management. Work paper, Pennsylvania State University, University Park, PA. http://www.smeal.psu.edu/cdt/ebrcpubs/res_papers/2004_01.pdf.

Beznosov, K. and B. Chess, 2008. Security for the rest of Us: An industry perspective on the secure-software challenge. IEEE Software, 25: 10-12.

Chandra, S. and R.A. Khan, 2008. Software security estimation in early stage of development life cycle. Proceedings of National Conference on Emerging Technologies (NCET). March 29-30, pp: 1-3.

Copigneaux, F. and S. Martin, 1988. Software security evaluation based on a top-down McCall-Like approach. Aerospace Computer Security Applications Conference, Fourth. Dec. 12-16, IEEE, pp: 414-418.

Dai, L. and K. Cooperb, 2006a. Modeling and performance analysis for security aspects. Sci. Comput. Program., 61: 58-71.

Department of Defense, 1985. Trusted Computer System Evaluation Criteria. CSC-STD-001-83. URL:http://csrc.nist.gov/publications/history/dod85.pdf.

Kajava, J. and R. Savola, 2005. Towards better information security management by understanding security metrics and measuring processes. Proceedings of the European University Information Systems, EUNIS 2005. June 20-21, The University of Manchester UK., pp: 1-6.

Khan, R.A. and K. Mustafa, 2008. Software vulnerability life cycle. Developer IQ, 8: 27-30.

Madan, B.B., G. Katerina, K. Vaidyanathan and K.S. Trivedi, 2002. Modeling and quantification of security attributes of software systems. Proceedings of the International Conference on Dependable Systems and Networks (DSN), June 23-26, IEEE, pp: 505-514.

McGraw, G., 2004. Software Security. IEEE Security Privacy, 2: 80-83.

McGraw, G. and N. Mead, 2005. A portal for software security. IEEE Securit. Privacy, 3: 75-79.

Nichols, E.A. and G. Peterson, 2007. A metrics framework to drive application security improvement. IEEE Securit. Privacy, 5: 88-91.

Orlandi, E., 1990. Computer security: A consequence of information technology quality. Proceedings of IEEE International Carnahan Conference on Security Technology Crime Countermeasures. Oct. 10-12, pp: 109-112.

Payne, S.C., 2007. A guide to security metrics. SANS Institute,

Peterson, G., 2004. Collaboration in a secure development process, part 1. Inform. Securit. Bull., 9: 165-172.

Sahinogly, M., 2005. Security meter: A practical decision-tree model to quantify risk. Infrastruct., IEEE Securit. Privacy, 3: 18-24.

Savola, R., 2007. Towards a security metrics taxonomy for the information and communication technology. International Conference on Software Engineering Advances (ICSEA). Aug. 25-31, IEEE, pp: 60-60.

Scandariato, R., B.D. Win and W.J. Distrinet, 2006. Towards a measuring framework for security properties of software. Proceedings of the 2nd ACM Workshop on Quality of Protection. Oct. 30, pp: 27-30.

Scheneidweind, N.F., 1992. Methodology for validating software metrics. IEEE. Trans. Software Eng., 18: 410-422.

Stytz, M.R. and J.A. Whitaker, 2003. Software protection: Security's last stand. IEEE Securit. Privacy, 1: 95-98.

Taylor, D. and G. McGraw, 2005. Adopting a software security improvement program. IEEE Security Privacy, 3: 88-91.

Wang, C. and W.A., Wulf, 1997. A framework for security measurement. Proceedings of National Information Systems Security Conference, Oct. 7-10, Baltimore, pp: 522-533.