Volume 2, No. 2, February 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

Towards the Application of Security Metrics at Different Stages of Information Systems

Irshad Ahmad Mir[*1], Mehraj-U-Din Dar[2], S.M.K Quadri[3]

[*1] Research Scholar Department of Computer Science, University of Kashmir, Srinagar, Jammu and Kashmir, India
irshad.mir@hotmail.com[1]
[2] Director IT&SS, University of Kashmir, Srinagar, Jammu and Kashmir, India
md_dar@yahoo.com.com[2]
[3] Head Department of Computer Science, University of Kashmir, Srinagar, Jammu and Kashmir, India
quadrismk@hotmail.com

Abstract: **A formal approach to the measurement of security in Information Systems is essential. However little thought has been given to this aspect of Information system life cycle. The attention towards the security aspect of the system has got least attention during the development process and much focus has been given to the functionality provided by the system. As the threats in the operational environment increased the attention towards incorporating security got the attention. With such incorporation of security mechanisms, the question now is how much we secure we are and what is the level of security in the system. The answer to this question can be possible by the application of security metrics and to analyse the results. Security metrics play a vital role at every stage of Information Systems development and in operational environment. This paper focus on the applicability of security metrics at the different stages of Information Systems life cycle and identifying some metrics framework present for each level of the system.**

Keywords: Information Security, Security Metrics, Security Patterns, Measurement, Information Systems, Security Engineering.

INTRODUCTION

Lord Kelvin said "If you can't measure it you can't improve it". This fact also applies to information security issue. Traditionally security has been treated as an afterthought leading to penetrates and patch cycle [33] which means after the exploitation of vulnerability the patches are added to provide resistance against attacks. The problem with such approach is as soon as the adversary came to know about the patches then it becomes more vulnerable for future attacks. It is easier to make informed engineering and management decision concerning security if sufficient and credible security evidence is available [34]. At the higher level Security policy describes both what is allowed as well as not allowed in the system. Security metrics are measurement to assess security related imperfections introduced during System development and under operational environment of information systems. Many information System quality attributes have been studied and measured extensively including maintainability, performance, reusability and reliability. The metrics which measure the security attributes of information security have received attention lately. It is inevitable fact that security must be estimated to come up with the non vulnerable system. But, the major problem of introducing estimate is where to measure the security during system development. Information system security cannot be added through the addition of set of features it must be designed and integrated with the every phase of software development life cycle [6] . Mostly the measurement approach was either from the higher level (i.e. Whole system level) or from a low level (i.e. code level). There is a need to incorporate the appropriate security metric framework at different levels of Information system life cycle.

In this paper we look at the various quality attributes of the security metrics and the various stages of the security architecture life cycle. Further we identify the various levels of information systems life cycle and the metrics suite applicable at each level. We also look at the various security metrics frameworks already proposed and there applicability at different level in the Information system life cycle, from the development to the operational phase

The rest of the paper is organized in seven Steps.(2) Related work (3) Need for security metrics (4) Information systems security architecture life cycle (5) Security metrics framework for each level of Information system life cycle (6) Security patterns for security metrics .(7) Security evaluation life cycle (8) Conclusion and future work .

2. RELATED WORK

Security engineering has been carried out in isolation of other system focus areas [10]; consequently, security has been considered as "add-on" property Therefore, software intensive system developers in general have not been involved [11]. Part of this "security in side role syndrome" is also that in some metrics approaches, security has been treated only as just another aspect of software quality [12], Organizations and companies are now recognizing the importance of security in the life cycle from network security, to system security and application security as an integrated end-to-end process.[1,2,3] Microsoft, among others, has instituted a security initiative that is corporate-wide. It is hoped that these efforts will produce software that is inherently more secure.[4] The highly volatile computing environment requires that security be viewed as a continuing process to meet the changing needs of the environment. Even with good requirements, security design flaws are still prevalent. Currently, companies like Citigal and @Stake are offering assistance and tools for verifying the security of software.[5,6,7] They include security fault injection tools and attack trees. Beside the efforts made towards incorporating security less efforts has been made towards measuring the security throughout the life cycle of Information Systems.

Based on the number of times a software system is mentioned in different security bulletins, Alhazmi *et al.* [8] propose a new metric called vulnerability density, (VD). VD is defined as the number of vulnerabilities in the unit size of code. Using this as the parent metric, a suite of security metrics are coined such as known vulnerability density (VKD), vulnerability discovery rate, residual vulnerability density (VRD), and ratio of vulnerable density to defect density

(VD/DD). VKD is the number of the already identified vulnerabilities. Vulnerability discovery rate is the number of vulnerabilities reported/identified per unit time. Residual vulnerability density is defined as, VRD = VD -VKD. There have been some more attempts in quantifying the security aspect of software systems .Manadatha *et al.* [9] uses an intuitive approach to measure a system's attack surface, the set of ways in which an adversary can attack the system. There are different standards available including Trust Security Evaluation Criteria (TCSEC), Information Technology Security Evaluation Criteria (ITSEC), Canadian Trusted Computer Product Evaluation Criteria (CTCPEC) which specify several criteria against which Information System security can be made. Security of the Information Systems cannot be measured directly. It can be made with the combined use of models metrics and attributes.

3. NEED AND CHARACTERISTICS OF SECURITY METRICS.

It is helpful to notice the difference between the metrics and measurements. Measurements provide single-point-in time views of specific, discrete factors while metrics are derived from comparing two or more measurements taken over time with a predetermined baseline [13]. Furthermore according to Alger [14] measurements are generated by counting, whereas metrics are generated from analysis. The WISSSR (Workshop on Information security Systems , Scoring and Ranking) of 2001[15] suggested that the expression IS* (Information Security*) be used as synonym for metric, measure, score, rating, rank or assessment results . According to Jelen [1**3**], a good metrics is specific, Measureable, Attainable, Repeatable and Time-dependent ("SMART").

It is widely accepted management principle that an activity, cannot be managed well if it cannot be measured. Overall, metrics provide four fundamental benefits- to characterize, to evaluate, to predict and to improve. Examples of using security metrics for assessment include **[30]**:

- Risk management activities in order to mitigate security risks,
- Comparison of different security controls or solutions,
- Obtaining information about the security posture of an organization, a process or a product
- Security testing (functional, red team and penetration testing) of a system
- Certification and evaluation (e.g. based on Common Criteria) of product or an organization, and
- Intrusion detection in a system.

In security engineering, security correctness, security effectiveness and security efficiency can be seen as the main fundamental measurement objectives [9]. These can be defined in the following way:

Security correctness denotes assurance that security enforcing mechanisms have been correctly implemented in system under investigation (SuI), and the system its components, interfaces and the processed data meet the security requirements

Security effectiveness denotes assurance that the stated security requirements are met in the SuI, and the expectations for resiliency in the use environment are satisfied, while the SuI does not behave in any way other than what is intended : and

Security efficiency denotes assurance that the adequate security quality has been achieved in the SuI, meeting the resources, time and cost constraint

The ideal metrics should be:

- Simple, precisely definable so that it is clear how the metric can be evaluated.
- Objective to the greatest extent possible
- Easy obtainable
- Valid – the metric should measure what it is intended to measure.
- Robust - relatively insensitive

4. INFORMATION SYSTEM SECURITY ARCHITECTURE LIFE CYCLE

The security architecture process is an iterative process that unifies the evolving business, technical and security domains. The four main phases (fig. 1) in the process are (1) Architecture risk assessment, (2) Security architecture and design (3) Implementation and (4) operations and monitoring [31].

Architecture Risk Assessment: assesses the business impact to critical business assets, the probability and impact of security threats and vulnerabilities. Since security is a system property, the architectural level is the proper level of abstraction to identify many of the most critical security flaws.

Security Architecture and Design: architecture and design of security services that enable business risk exposure targets to be met. The policies and standards, and risk management decisions drive the security architecture and the design of the security processes and defense in depth stack.

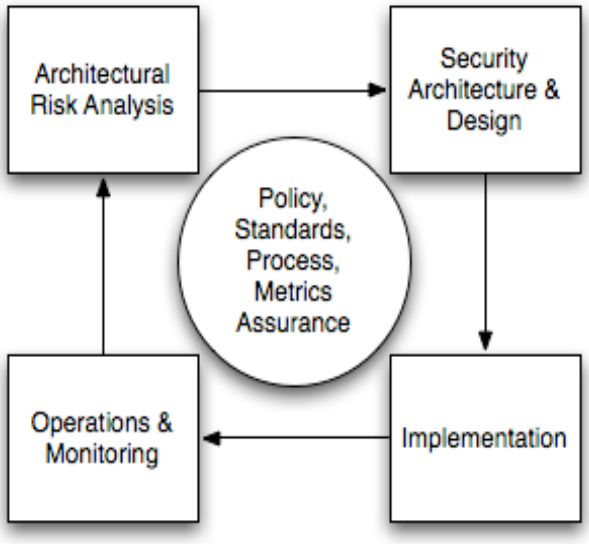

Fig.1. Security Architecture Life Cycle

Implementation: security processes and services implemented, operational, and managed. Assurance services are targeted at verifying that the Risk Management, Security Policy and Standards, Security Architecture decisions are reflected in the actual runtime implementation.

Operations and Monitoring: Ongoing processes, such as vulnerability management and threat management that monitor and manage the operational state as well as the breadth and depth of systems security. Operational and monitoring processes should be instrumented with security metrics to better measure the runtime environment.

As one can see from *fig.1* security metrics are at the core of security architecture life cycle.

5. SECURITY METRICS FRAMEWORK FOR DIFFERENT LEVEL OF INFORMATION SYSTEM

If there is no malicious intent, there will be no need for security solutions. In Practice such situations are very rare. Instead of looking from the higher level of abstraction (i.e. from system level perspective) security metrics can be found at different levels of Information system (fig.2). With each level of security metrics is associated the framework from which these metrics can be conceived. An example of framework for system level security metrics is using attack surface by [16]. Rather than counting the bugs at code level, it measures the system's attackability at higher level. Such type of metrics comes under the System (high) level. It measures the system attacakbility across the three abstract dimensions (method, data, and channels). Inutility the larger the attack surface, the more likely the system will be attacked.

Orthodox security assessment process is carried out at the System level using qualitative criteria by security experts. The problem lies in, knowing how and when it should be measured [21]. These situations make way for more rigorous approach, able to estimate security at design phase of Information System lifecycle. Nichols and Peterson [18] introduce a metric framework which upgrades the security of software application. It discusses the importance of design time metric. Design time metrics have the ability to identify and categorize weaknesses at early stage of System development. Scandariato [19] tried to give up a shape to the idea of security properties of software that are quantities in nature with regard to assessment m, allow proactive estimation of software security , especially during the design phase.

The next level is the code level security metrics: it can be useful to analyze defects at this lower level. Code level security metrics specifically look into code structure and implementation language issues. An example of such a frame work for code level metrics is by Yonghee and Laurie Williams [23]. They identified code complexity metrics that differentiate vulnerable functions from non-vulnerable functions and faulty-functions, and to investigate whether code metrics can be useful for vulnerability prediction. As we go down to the lower level of abstraction, there seems to be fewer and fewer metrics fig.2 highlights this scenario. As an example the metrics proposed by [20] such as coupling corruption propagation, stall ratio, and critical element ratio. These metrics measure source code quality properties that can enhance program security. The primary objective of such metrics is to quantify whether an attacker can produce any undesirable effect in the program by exploiting the imperfections in source code structure.

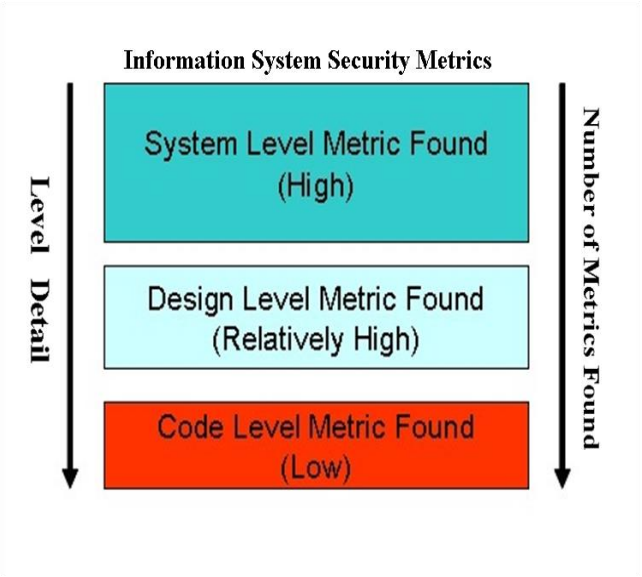

Fig.2. Current state of Metrics Types Found[20].

Beside the work done in devising security metric framework for various stages of system development there is not a general acceptable approach followed by the system developers, because of the increasing complexities for security related issues. In the next section we look at the different security patterns for different levels system development.

6. SECURITY PATTERNS FOR SECURITY METRICS FRAMEWORK

In the design phase of software development we should design security functions to satisfy the security properties of assets in requirement phase, specifically we can design such functions using access controls, authentication, cryptography , electric signatures; what security functions are needed depends not only the security properties but also on the security strategy. Youder and Barcalon first introduced conceptual security architecture as patterns [25]. They provided the natural language description of seven security patterns. Fernandez and Pan illustrated security patterns using UML [26].

In implementation/coding phase we write the software which satisfies a given design. To develop secure software we must correctly implement various mechanisms that support security. An implementation flaw is a mistake made by a programmer while writing a software program. Unfortunately not all programmers have sufficient knowledge to create a secure system. Many researchers have developed guidelines that support the sharing of knowledge about how to write a secure code. A number of security flaws at implementation level have been identified and documented. In [2] it presents 18 implementation rules that programmers should note in order to eliminate common security problems. These rules are all informally described in natural language. Similar rules are shown in [27] which provide more than 20 practices in six categories. In addition in [28] it provides well known guidelines for writing secure program for Linux and UNIX systems. In [28] not only the description of security vulnerabilities but also language specific issues for C/C++, Perl, Python, Shell Scripting, Ada. Java, TCL and PHP have been presented. In [29] it introduces 12 rules for writing security criteria for java code.

These guidelines and security patterns only provides what should be considered during secure system development in advance but not the level of security assurance achieved, still the secure system development is not achieved at its best . The reason behind this is the lack of formal framework against which the design, implementation/coding of system should be validated for security. These security patterns should play a useful part in the development of security metric for different stages of information systems. If we map and quantify these security patterns into security metrics against which the security is to be evaluated then it reduces the efforts and risk at subsequent higher level.

7. SECURITY EVALUATION LIFE CYCLE

Literature survey shows that many efforts have been made in devising the security metrics framework for each level of Information Systems. During the System development the security related issues especially the security measurement is usually underestimated. With the advent of today's networked environment and automated tools used by the adversaries, Information systems are vulnerable to various security attacks. To mitigate and reduce risk of such attacks, it is not sufficient to measure the security posture of information system at single level of abstraction. Security measurement must start from the very beginning (i.e. from design and coding) stages and subsequently follow to the actual operation phase. By following such strategy maximum level of security can be achieved and it becomes easier to manage the security related

issues at overall system level in the operational environment. For instance if there are vulnerabilities at the code level, latter an attacker may exploit these vulnerabilities. One solution is to add patches latter to overcome it but adding patches can become more vulnerable once an attacker knows about it. So we need a systematic approach to measure and mitigate the security at all the phases and abstraction levels of information system. Fig .3 shows different stages of Information systems and the security metrics applicable for each stage. At each level the evaluation process can be repeatedly employed till the certain assurance level is achieved.

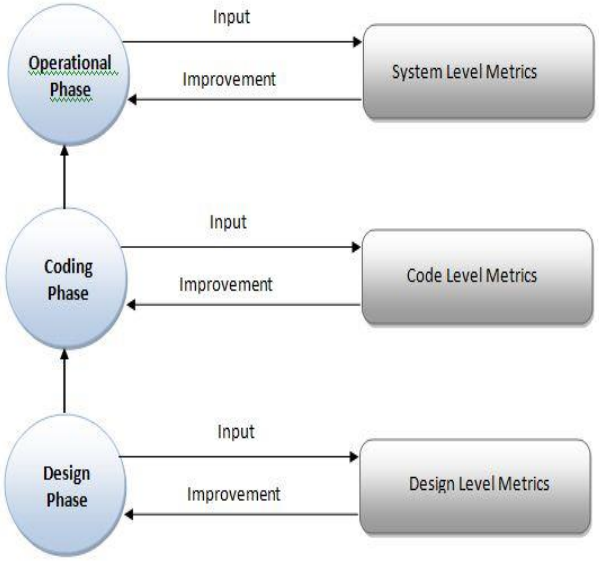

Fig.3. Security Evaluation process for Information Systems

For each level depicted above (fig.3) security estimation can be performed with the security estimation life cycle [32]. The stages of estimation cycle are.

1. Input.
2. Security estimation process
    a. Identify security factors
    b. Identify metric suit
    c. Quantify security factors
    d. Estimate security
3. Output process
    a. Qualitative analysis
    b. Overall security analysis

8. CONCLUSION & FUTURE WORK

We have proposed taxonomy for measuring the security of at various stages of Information system life cycle. For each level we have analyzed the various security metrics frameworks against which security can be measured. The fundamental goal of system development is to deliver highly secure products. One of the major advantages of introducing security evaluation life cycle is that it may detect and mitigate vulnerabilities from the lower level of abstraction and reduces the efforts and cost in future operational stages. The future work will be to explore and propose a security metrics frame work for software development process which can be applicable at early development phases.

REFERENCES

[1] G. McGraw, “Building Secure Software: A Difficult but Critical Step in Protecting Your Business,” *Cigital, White Paper*, available at: http://www.cigital.com/whitepapers/.

[2] M. Bishop, Computer Security: Art and Science, Addison-Wesley-Longman, Nov. 2002.

[3] K. Hoo, A. Saudbury and A. Jaquith, “Tangible ROI through Secure Software Engineering,” Secure Business Quarterly, Q4, 2001

[4] M., Howard and D. LeBlanc, Writing Secure Code, Microsoft Press, Redmond, WA, 2002.

[5] G. McGraw, “Building Secure Software: A Difficult But Critical Step in Protecting Your Business,” Cigital, White Paper, available at: http://www.cigital.com/whitepapers/.

[6] G. McGraw, “Software Risk Management for Security,” IEEE Computer, 32(4), April 1999 pp. 103-105.

[7] A. Jaquith, “The Security of Applications: Not All Are Created Equal,” Research Report, @Stake, February 2002,

[8] Alhazmi, O.H., Malaiya, Y.K., and Ray, I. 2007. Measuring, analyzing and predicting security vulnerabilities in software systems. Computers and Security Journal 26, 3 (May 2007), 219-228.

[9] P. K Manadhata, and J. M. Wing 2005. An Attack Surface Metric. Technical Report. School of Computer Science,

[10] C. Meadows, “The feasibility of quantitative assessment of security,” Conf. on Distributed Computing for Critical Applications (DCCA-4), San Diego, Jan. 1994.

[11] R. Savola, H. Abie, “On-line and off-line security measurement framework for mobile ad hoc networks,” Journal of Networks, Vol. 4, No. 7, Sep. 2009, Academy Publisher, pp. 565-679.

[12] B. Chess, “Metrics that matter: quantifying software security risk,” Workshop on Software Security Assurance Tools, Techniques and Metrics, Long Beach, California, Nov. 7-8, 2005.

[13] G. Jelen, SSE-CMM Security Metrics. NIST and CSSPAB Workshop, Washington, D.C., June 2000.

[14] J. I. Alger, On Assurance, Measures, and Metrics: Definitions and Approaches. Proc. of Workshop on Information Security System Scoring and Ranking (WISSSR), ACSA and MITRE, Williamsburg, Virginia, May, 2001, proceedings published 2002.

[15] R. Henning et al., Proceedings of Workshop on Information Security System, Scoring and Ranking – Information System Security Attribute Quantification or Ordering (Commonly but improperly known as “Security Metrics”), ACSA and MITRE, Williamsburg, Virginia, May, 2001, proceedings published 2002.

[16] P. Manadhata J.M. Wing “An Attack Surface Metric” In Technical Report CMU-CS-05-155, July 2005

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

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

[19] Scandariato, R., B.D. Win and W.J. Distrinet, “Towards a measuring framework for security properties of software. Proceedings of the 2nd ACM Workshop on Quality of

Protection, Oct. 30, ACM New York, NY, USA. 2006. pp: 27-30.

[20] I. Chowdhury , B. Chan , and M. Zulkernine, "Security Metrics for Source Code Structures," in Proceedings of the 4th International Workshop on Software Engineering for Secure Systems, Leipzig, Germany, May 2008, pp.57-64.

[21] Khan, R.A. and K. Mustafa,. Software vulnerability life cycle. Developer IQ, 2008, 8:27-3

[22] R. Savola, "A security metrics Taxonomization model for software-intensive systems," *Journal of Information Processing Systems, Vol. 5, No. 4, 2009, 10 p.*

[23] Y.Shin and L.Williams, "An Empirical Model to Predict Security Vulnerabilities using Code Complexity Metrics" Department *of Computer Science North Carolina State University Raleigh, NC 27695, U.S.A.*

[24] P. Giorgini, F Massacci, J Mylopoulos, and N. Zanone, "Modelling security requirements through ownership, permission and delegation," *In 13th IEEE International Conference on Requirements Engineering 2005*, pp 167-176, 2005.

[25] J. Yoder and J.Barcalow, " Architectural pattern for enabling application security", In *proceedings of PLOP, 97 conference* 1997

[26] E. B Fernandez, R. Pan "A pattern language for security model" *In proceedings of PLOP* 2001.

[27] E. B Fernandez, J.C. Pelae and M.M. LarrondoPetrie " Security pattern for VoIP network" *in proceedings for International Multi conference on computing in Global IT* , 2007.

[28] D.A. Wheeler, "Secure programming for LINUX and UNIX", 1999.

[29] G.McGraw and E. Felten, "Twelve rules for developing more secure java code", 1998 http://www.javaworld/jw-12-1998/jw-12-securityrules.htm.

[30] R. Savola "Towards a Security Metrics Taxonomy for the Information and Communication Technology Industry" *International Conference on Software Engineering Advances (ICSEA). IEEE*, Aug.25-31 , 2007.

[31] Gunnar Peterson, "Security Architecture Blueprint", *Arctec Group LLC*, 2007.

[32] Chandra S, Khan RA "Object oriented software security estimation life cycle-design phase perspective". *Journal of Software Engineering.* 2, (2008), pp. 39-46.

[33] D. Gilliam, T. Wolfe, J. Sherif, and M. Bishop, Software Security Checklist for the Software Life Cycle", *Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, 2003.

[34] R. Savola, "On the feasibility of utilizing security metrics in software intensive systems", *International Journal of Computer Science and Network Security*, Vol. 10, No. 1, Jan. 2010, pp. 230-239.

