System Engineering Security

Esmiralda Moradian

Department of Computer and System Sciences,
Stockholm University
Forum 100, 164 40 Kista, Stockholm, Sweden
esmirald@dsv.su.se

Abstract. Organizations' integrate different systems and software applications in order to provide a complete set of services to their customers. However, different types of organisations are facing a common problem today, namely problems with security in their systems. The reason is that focus is on functionality rather than security. Besides that, security, if considered, comes too late in the system and software engineering processes; often during design or implementation phase. Moreover, majority of system engineers do not have knowledge in security. However, security experts are rarely involved in development process. Thus, systems are not developed with security in mind, which usually lead to problems and security breaches. We propose an approach of integration security throughout engineering process. To assure that necessary actions concerning security have been taken during development process, we propose semi-automated preventive controls.

Keywords: System engineering, software development, security, risk management, control, security breach.

1 Introduction

Enterprises, governmental, medical, and defence sectors are dependent on software developed by system engineers. Software weaknesses and defects can result in software application failure, but also be exploited by attackers. [16]. Software is everywhere [16]. Therefore, a poorly designed and developed as well as not properly tested software will induce security breaches that, for example, can cause economical problems, and/or affect humans' health and, in the worse case, even cause death. Thus, systems that handle sensitive, secret and/or valuable information must operate correctly and have a stable status. Organisations' systems handling sensitive data are attractive targets for attackers. Despite of a large number of different security standards and mechanisms and tools existing today, vulnerabilities in systems are still present. During the recent years we could observe criminal activities performed on so called "secure systems" as well as errors in software. One example is bank systems. During few years direct aimed attacks were performed on Nordea bank's systems. Another example is the medical sector. Due to software error in the Therac-25, caused by software bug many people received radiation overdoses that affected their lives [16]. Vulnerabilities are usually the result of defective specifications, design,

J.D. Velásquez et al. (Eds.): KES 2009, Part II, LNAI 5712, pp. 821–828, 2009.

implementation, and testing that are unknowingly injected into software by system developers. [13] To a large extent flaws and defects can be minimized. However, to provide system reliability, correct operation and precise performance - security aspects should be considered early in system development lifecycle. Systematic and structured actions such as risk management (identification, analysis, evaluation of risks as well as risk treatment, monitoring and review) and threat modelling are required. Authors in [20] presented the survey made by UK Department of Trade and Industry. According to the survey many companies do not have sufficient internal controls of systems on the Internet. Organisations are often satisfied with "secure" software they purchase for these reasons.

Unfortunately, many organisations are not considering software security during the development life cycle. Commonly, security is added sometimes after the system is developed. Bishop in [3] states that systems where security mechanisms were added after the system was built are not trustworthy. We propose an approach where security is interspersed in the system engineering process. Semi-automated control of performed security measures during different phases of software development life cycle (SDLC) can facilitate building necessary and required security. The approach involves not only computer specialists but also managers/decision makers who have responsibility for the information security, and customers who present requirements.

2 Related Work

Mouratidis, et al. [14] argues for the need to develop a methodology that considers security as an integral part of the whole system development process. Authors propose an approach that considers security concerns as an integral part of the entire system development process. The different stages of the approach are described with the help of a health and social care information system.

N. Haridas in [8] point out that application developer never consider or have a disciplined process to address security in any phases of SDLC. The author proposes incorporating security in different SDLC phases. N. Haridas [8] presents security factors followed by an example. Further the author presents a list of prioritized factors that could be considered as security guideline during the development phase

G. McGraw in [13] proposes building security in SDLC. The author presents a detailed approach for putting software security into practice. G. McGraw [13] provides guidance on how to build secure software and shows gaps in the development process and to how to improve the process. However, while there exist different approaches and guidance's on how to integrate security in system development process, decision makers and developers hardly adopt those. We propose approach of integration security throughout engineering process with support of semi-automated and preventive controls in each phase of SDLC process.

3 Security Issues in Development Life Cycle

"The essence of good security engineering is understanding the potential threats to a system, and then applying an appropriate mix of protective measures – both

technological and organisational – to control them" [1]. In many organisations managers and developers teams start, in the best case, thinking about security when system is already shipped in production, which affects quality, security and economics. Vulnerabilities embedded in software and system components affect security of the system. However, as a result of vulnerability, security breaches can be costly both in terms of effort to fix problems and damage to organisations. For example security breaches cost more money, take more time and can bankrupt a company and destroy information. (See Figure 1)

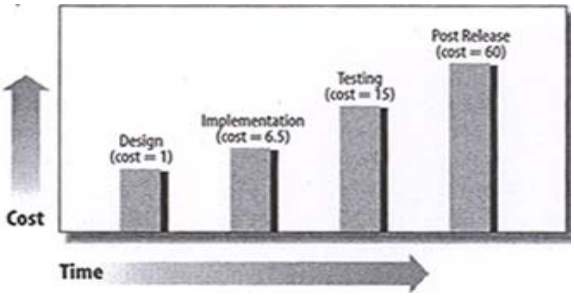

Fig.1. Cost of fixing security flaws during different development phases Source: Secure Coding – Principles&Practices Mark G. Graff, Kenneth R. van Wyk O'Reily

Moreover, adding security afterwards can impact people's everyday's life. Customers and end users are rare aware of software insecurity and will expect systems, applications, products and services to be secure. [9, 18] Development process deals with transformation of requirements into a software product or software-based system that meets the customer's defined needs. Gathering and capturing requirements is one of the critical parts of development process, which affect system development during all other phases. Requirements express behaviour of the system, the system and object states and the transition from one state to another. Requirements also describe system's activities. Usually, requirements are described as functional and non-functional [15]. The purpose of software and system requirements analysis "is to establish the requirements of the software elements of the system" and "transform the defined stakeholder requirements into a set of desired system technical requirements that will guide the design of the system" [11]. However, it is common that security requirements of software and system are not even identified. During the design developers trying to identify "which system requirements should be allocated to which elements of the system." [11] Software design is design that implements and can be verified against requirements. However, security requirements that are not identified and analysed will lead to security flaws in system architecture and design. Testing ensures that the implementation of each system requirement is tested and that the system is ready for delivery [11]. Though again, missing security requirements, security flaws in architecture and design will produce unreliable test results. Security testing should be a part of development life cycle. In this paper we propose integration of security throughout engineering process.

Several standards, procedures, methods, and tools was developed in purpose to help organisations to understand, develop and implement software systems as well as

manage and control organisations' processes. However, it seems to creates problems "in management and engineering, especially in integrating products and services" [11]. For example, C.B. Haley in [7] argues that ISO and NIST documents "provide little guidance on how to connect the functionality to the security needs. Instead of describing when and why objects are to be protected, they describe how the objects are to be protected." Existing standards for information and software provide guidance and support for organisations. For example, ISO 12207 provide guidance to software life cycle architecture [11]. ISO 27001[10] provide a "Plan-Do-Check-Act model" to structure the organizations processes. The purpose is establishing, implementing, maintaining, and improving information security management system [10]. ISO 27002 provide guidance for "initiating, implementing, maintaining, and improving information security management within an organization" [10]. ISO/IEC 15026 [22] defines a process for establishment of integrity levels. Risk analysis and system architecture analysis are the part of the process. COSO-ERM (Enterprise Risk Management - Integrated Framework) was created to help businesses and other entities assess and enhance their internal control systems [6]. Common Criteria (CC) is a framework for evaluating security products and systems. [5] However, many organisations are not working according to one or more security standards because of different reasons. One reason can be difficulty to choose the most suitable standard for organisation. As appear from C. Magnusson's [12] report there exist gaps and overlaps in standards [12]. Author studied five standards, among others ISO 27001 and COSO-ERM.

4 Development and Control

To build secure software system requires effort from all parties: managers, architectures, designers, developers, testers [21], as well as customers and consumers. Security experts are seldom involved in the process; consequently security is left aside. Moreover, people involved in software system development process often go beyond their expertise and do work in other domain areas. That causes problem but seems to be the rule rather than exception in software development. [19] Software and system engineers, usually, are experts in one or more area, such as software architecture, programming, testing. The Engineer's expertise comprises the development process and maintenance process. [19] The managers' expertise comprises organisational life cycle process that for example includes initiation and scope definition, planning, review and evaluation, and closure. [19]. Managers, developers, and customers are usually not experts in security and have different understandings and views on it. This can create misunderstanding and consequently lead to wrong decisions that impact the result. It is important to emphasise that a system probably will be attacked if it has valuable assets that attract attackers and entry points. Thus, to be able to build secure enough systems it is necessary to involve security experts early in the development process.

To provide reliable system and/or service software life cycle and security life cycle should be joined and melt together. Our approach is intersperse security in the system engineering process. Software engineering process lacks visibility and continuity. It is difficult to see progress in software construction. [19] To see and track decisions,

implemented security measures, and mechanisms, semi-automated control of performed actions, such as, checks, logs and feedbacks are required in every phase of system engineering process. To assure incorporation of security in the life process, we propose the semi-automated control regardless organisation's branch, development process model, techniques and tools used. Semi-automated control with tracking of performed actions during different phases of software engineering process can facilitate in building necessary and required security (see Figure 2). Security activities are summarised according to classic V-model. Activities in the left side are concentrated on security requirements and secure design, while activities on the right side focus on verification and validation throughout the entire software life cycle. We propose to start from managers, including security managers that make decision to start the project/process of software development. Those need to classify level of security depending on different inputs. Examples of inputs are: business goals and conditions, policies, standards, environment where system should operate, requirements, and knowledge and expertise, and limitations. (See Figure 2)

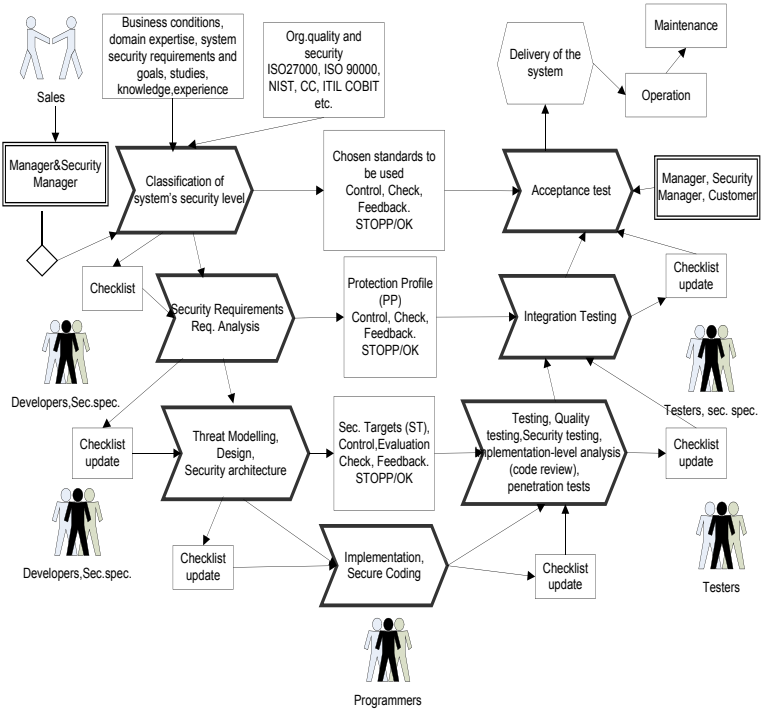

Fig. 2. Controlled Development life cycle

Depending on application, security level will vary. The security level of bank system that handles electronic payment through online and Internet banking will be much higher than security level of system that handles parking tickets. However, it is important to point out that every software system should have basic level of security, i.e. security baseline. Risk assessment should also be done. Based on classification of the security level, system generates an output – automated security guideline and control.

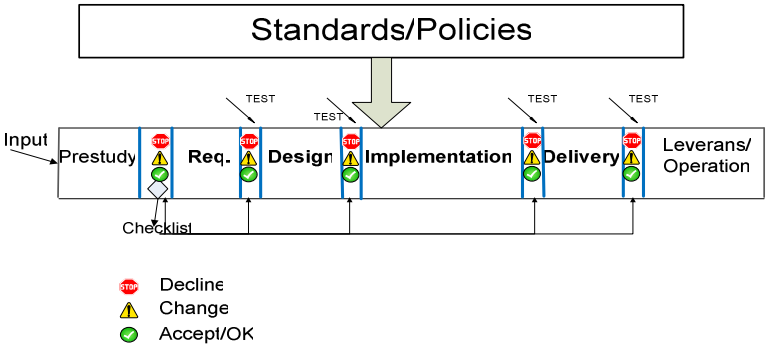

Fig. 3. Control checkpoints in system development process

It can be constructed as checklists with multiple checkpoints. We use software agents for this purpose. These checkpoints should be considered and implemented during development process. The checklist can contain points concerning information security, secure design, network security, and security testing. Some examples of checkpoints are: random password generation, access control, error messages not reveal too much information to an attacker, threat analysis, penetration tests. If some points were not implemented or changed during the process an explanation should be present in updated on each phase checklist. Checks are performed on every phase of development process and all actions are monitored and verified. (See Figure 2 and Figure 3) System will record all changes in checklist, track users and record their actions during the process. Thus, checklist can be considered on the one hand as support for managers, developers and testers in decision making and on the other hand as control of actions taken.

Moreover, ability to provide trusted system or service requires dependability. Allen et al. state that dependable software is software that will "never deviate from correct operation under anticipated conditions" [4]. Dependability requirement of the system can be different, depending on application. [2] Dependability encompasses following attributes: reliability, safety, security (with respect to confidentiality, integrity, availability) and maintainability [2] and sets requirements on the entire system [21]. Depending on the security requirements, protection profile (PP), specific actions will be required to be taken by system developers. System will track decisions made. Decision(s) should be evaluated by evaluator and feedback given about "whether a PP is complete, consistent, and technically sound and hence suitable for use in developing an ST" (Security Targets). [5] Risk management process is an essential part in implementing adequate security level [17]. It includes communication, environment (intern and extern) and risk assessment. Risk assessment consists of risk identification, analysis and evaluation. The purpose of risk assessment is to produce knowledge about relevant security characteristics of systems. Identifying valuable assets and performing risk assessment and threat modelling are the necessary actions to be completed before implementation phase. Pfleeger [15], Swiderski and Snyder [18] points out that organization must understand the vulnerabilities to which it may be exposed. Vulnerabilities can be determined by performing risk analysis. Moreover, risk analysis includes threat identification, analysis and modelling. These can describe when

and why objects need protection. The authors in [3] point out two types of risk analysis methods, namely quantitative and qualitative methods. However, to perform proper risk analysis integration of both is necessary.

Design should include designing and modelling controls that detect and/or prevent risks identified in previous phase. This stage also includes modelling rules that will be enforced in implementation phase. During design phase all stated requirements should be satisfied. Otherwise the process is stopped and cannot continue. Desired security behaviour should be in balance with functionality requirements. Control check list must be updated. All changes should be documented. Implementation stage is about to transform design into one or more programming language and assure that system is working as expected. Testing is done iteratively in order to verify specified requirements and validate that all requirements for a specific intended use of the software work product are fulfilled. Security testing is necessary in order to verify that the system design and code can stand against attack [9]. If or when security flaw is found in the design or in code it should be fixed, recorded and analyzed. Security testing is about determining if features work in different way than it was anticipated by developer(s) [9].

5 Conclusion and Further Work

Security in system engineering plays an increasingly important role in nowadays business. Security has impact core business processes in every organization. We have emphasized some important problems in development process in order to enlighten the managers and the developers about the gaps in communication as well as common lack in visibility and continuity that have impact on security of system engineering. We proposed an approach of integration of security in each phase of system engineering process and implementation of semi-automated control in order to log changes in checklist, track users and record their actions during the process. However, the outcome will not exclude earlier made bad decisions that affect the continuous decisions and actions in system developing process. The proposed approach can enable qualitative, secure and effective way of system development.

The next step is to examine every step in detail, starting from requirement analysis.

References

1. Andersson, R.: Security Engineering A guide to building Dependable Distributed Systems, 2nd edn.
2. Avizienis, A., Laprie, J.-C., Randell, B.: Fundamental Concepts of Dependability. UCLA CSD Report no. 010028 LAAS Report no. 01-145 Newcastle University Report no.CS-TR-739
3. Bishop, M.: Introduction to Computer Security. Pearson Education, Inc., London (2005)
4. Allen, J.H., Barnum, S., Ellisson, R.J., McGraw, G., Mead, N.: Software Security Engineering A Guide for Project Managers. Addison-Wesley, Reading (2008)
5. CC, 2006. Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Information. Version 3.1 Revision 1 (September 2006)

6. COSO-ERM Enterprise Risk Management —Integrated Framework. Executive Summary (September 2004)
7. Haley, C.B., Moffett, J.D., Laney, R., Nuseibeh, B.: A Framework to security requirements engineering. In: SESS 2006, Shanghai, China, May 20–21, 2006. ACM, New York (2006)
8. Haridas, N.: Software Engineering – Security as a Process in the SDLC, April 2, 2007. SANS Institute InfoSec Reading Room (2007)
9. Howard, M., LeBlanc, D.: Writing Secure Code, 2nd edn. Microsoft Press (2003) ISBN 0-7356-1722-8
10. http://www.27000.org/
11. http://www.12207.com/
12. Magnusson, C.: Corporate Governance, Internal Control and Compliance (September 2007), http://www.svensktnaringsliv.se/material/rapporter/article35898.ece
13. McGraw, G.: Software Security Building Security in. Addison-Wesley, Pearson (2006)
14. Mouratidis, H., Giorgini, P., Manson, G.: When security meets software. engineering: A case of modelling secure information systems (2005) ISSN: 0306-4379
15. Pfleeger, S.L.: Software Engineering Theory and Practice, 2nd edn. Prentice-Hall, Inc., Englewood Cliffs (2001)
16. Rice, D.: Geekonomics The Real Cost of Insecure Software. Pearson Ed. Inc., London (2008)
17. Sherwood, J., Clark, A., Lynas, D.: Enterprise Security Architecture A Business-Driven Approach. CMP Books (2005) ISBN 1-57820318-X
18. Swiderski, F., Snyder, W.: Threat Modelling. Microsoft Press (2004) ISBN 0-7356-1991-3
19. Van Vliet, H.: Software Engineering Principles and Practice, 2nd edn. John Wiley and sons, Chichester (2004)
20. Boer, T., Booijink, T., Liezenberg, C., Nienhuis(Innopay), J.J., Bryant, C., Pruneau(EBA), A.: E-invoicing 2008 European market description and analysis. V. 1.0 February 2008 Copyright © 2008 Euro Banking Association (EBA) and Innopay
21. Lindström, C., Näsström, S.: Handbook for Software in Safety-Critical Applications. Swedish Armed Forces (2005)
22. http://www.iso.org