Activities of Security Engineering in System Development Life Cycle: Security Engineer’s View

YOUNG-GAB KIM
Department of Computer and Information Security
Sejong University
209, Neungdong-ro, Gwangjin-gu, Seoul 143-747, Korea
KOREA
alwaysgabi@sejong.ac.kr http://sites.google.com/site/alwaysgabi

Abstract: - The software engineering is a fundamental component to produce information systems and related software components. However, the traditional software engineering is not adequate and effective for developing secure information systems because, in the field of software engineering, security concerns are treated as of secondary importance. Therefore, in order to treat security features in development of secure information systems, many security engineering technologies have been proposed. However, most of the researches are focused on isolated, inconsistent software engineering-based or risk analysis-based technologies. Moreover, the existing security engineering methodologies, in practice, does not integrated smoothly with the other engineering disciplines (e.g., software engineering, system engineering). In this paper, we propose holistic, consistent, and integrated security engineering methodology, especially focused on security-related activities in all phase of software development life cycle (SDLC), which provide a way to support design, coding, testing, and maintenance for consistent security management. The proposed security engineering methodology combines security risk control, enterprise security architecture (ESA), and security management as an integrated framework.

Key-Words: - security engineering, enterprise information system (EIS), enterprise security architecture (ESA), security risk analysis, security management

1 Introduction
Recently, many forms of security attacks against enterprise information systems (EISs) has emerged that attempt to compromise the EISs and organizations. Therefore, managing the security of EISs has become a critical issue. Even though the software engineering discipline provides principles, methodologies, and tools for the development of information systems, it is not adequate and effective for designing, developing, and maintaining secure software systems [1]. The main reasons are as follows: 1) In many cases, security features in the field of software engineering are treated as of secondary importance, and this neglect results in the system engineer developing buggy code with weak security measures [2]. 2) Software engineering does not support a holistic and integrated approach to control security features [3, 4]. In the traditional software engineering environment, security-related activities (or safeguards) are temporally and inconsistently conducted.

In addition, in order to develop secure EISs, the roles of the stakeholders are very important. However, most system engineers, especially software engineers, have no security-relevant knowledge about such issues as security risk analysis, and security mechanisms and services. They also do not consider security as a functional requirement because processes for managing system development life cycles prioritize functional requirements (e.g., specific behaviors of a system) over nonfunctional requirements (e.g., security, performance, reliability, and usability) [5]. Similarly, most security engineers do not have the systems-engineering background required to approach a security problem holistically [6]. Furthermore, they often do not consider the entire vulnerabilities that exist in information systems and outside threats in their entirety, focusing instead on particular defensive strategies and ignoring serious gaps in security architectures [7]. Besides, many large organizations, in many cases, adopt security solutions in the face of security attacks. However, these solutions are focused mainly on how to defend their systems against various inside and outside threats rather than on overcoming the causes of the security issues in the information systems [8].

In this paper, in order to develop secure EISs, we propose a holistic, consistent, and integrated security engineering approach especially for security engineers. The main contributions of this paper are as follows: 1) The definition of the security engineering is defined as an engineering that focuses on the security aspect, especially from security engineer’s view, in the software development life cycle (SDLC). 2) A layered security engineering model, which is composed of three important components: security management, enterprise security architecture (ESA), and security mechanism and services, is proposed. 3) The activities of security engineering in the SDLC and its tasks are proposed. The proposed security engineering methodology involves the repeatable and systematic procedures in an effort to ensure that the security activities in SDLC are complete, consistent and easy to understand and analyzable by the security engineers.

The rest of the paper is organized as follows. We review related work in Section 2 with focus on existing approaches to security engineering. In Section 3, we explain our layered security engineering model and the activities of security engineering in the SDLC. In Section 4, we compare the existing approaches and the proposed method. Finally, Section 5 concludes the paper and suggests future work.

2 Related Works

2.1 Definition of Security Engineering

“Security Engineering” terminology becomes frequently used in computer engineering, software engineering, and web engineering community. However, meaning of the terminology is so diverse that it causes confusion based on user’s context, viewpoint and background [9]. Howe [10] firstly defined the system security engineering as an empirically based methodology for composing and evaluating systems within a structure of standards and which encompasses operation as well as planning design implementation and operation. Vaughn et al. [11] and Jei et al. [9] defined the security engineering as an instance of software engineering. Vaughn et al. described that security engineering is the same approach as systems and software engineering. Jei et al. defined the security engineering as a set of methodologies and technologies for fast and cheap development and operation of high quality security systems by means of applying cryptology, information security technology and software engineering. Anderson [12] who one of the pioneers of security engineering defined that security engineering is a field of engineering that deal with building secure systems that will remain dependable in the face of malice, error, or mischance. It focuses on the tools, processes, and methods needed to design, implement, and test complete systems, and to adapt existing systems as their environment evolves. More diverse definition of security engineering can be found in [9]. In this paper, security engineering is defined as a field of engineering that focuses on the security aspect, especially from security engineer’s view, in the SDLC. It is a consistent and integrated security engineering approach that covers all phases of development processes (i.e., requirement analysis, design, coding, testing, deployment, and maintenance). The security engineering defined in this paper involves aspects of information security (e.g., cryptography, security mechanisms & services, security policies, conventional information security technologies) and software engineering (e.g., SDLC, architecture technologies).

2.2 Related Works

There have been numerous approaches to provide security engineering. Most of the approaches belong to the following categories: security-related standard, software engineering-based, security requirement engineering-based, and risk analysis-based approaches.

2.1.1 Standards Related to Security Engineering

There are several international standards [13-19] related to security engineering: SSE-CMM (Systems Security Engineering – Capability Maturity Model, ISO/IEC 21817) [13, 14] is a process model that describes the essential systems security processes and management tasks that any organization must perform. This model provides high-level and security risk analysis-based activities. Furthermore it is focused on the requirements for implementing security in software systems or related systems components.

ISO/IEC 15408 [15] is a common criterion (CC) for evaluating the security of information system or software, entitled “Information Technology – Security Techniques – Evaluation Criteria for IT Security.” It defines two security requirements: security functional requirements (SFR) [16] and security assurance requirements (SAR) [17]. Its purpose is to allow users to specify security
requirements, developers to specify the security attributes of their products, and evaluators to determine whether products actually meet their claims. ISO/IEC 13335 [ISO/IEC 13335-1:2004 Information technology -- Security techniques -- Management of information and communications technology security] is a guidance on the management of IT security, entitled “Information Technology – Guidelines for the Management of IT Security (GMITS)”. It presents the concepts and models fundamental to a basic understanding of IT security, and addresses the general management issues that are essential to the successful planning, implementation and operation of IT security. ISO/IEC 27001 [ISO/IEC 27001:2013 Information technology -- Security techniques -- Information security management systems -- Requirements] is an information security standard, entitled “Information Technology – Security Techniques – Information Security Management Systems - Requirements”. It specifies the requirements for establishing, implementing, maintaining and continually improving an information security management system within the context of the organization. It also includes requirements for the assessment and treatment of information security risks tailored to the needs of the organization. ISO/IEC 27002 [18] is an information security standard, entitled “Information Technology – Security Techniques – Code of Practice for Information Security Management.” It provides a best practice guide to information security controls. It has established guidelines and general principles for initiating, implementing, and improving information security management with an organization. ISO/IEC 15026 [19], entitled “Systems and Software Engineering -- Systems and Software Assurance -- Part 3: System Integrity Levels”, defines a process for establishment of integrity levels. It provides a risk analysis and development approaches as a part of the process. However, these standards are complex, and it is hard to understand and implement them. Furthermore, they do not explain how to implement each security control successfully within all states of the SDLC [20].

### 2.1.2 Software Engineering-based Approaches

CLASP (Comprehensive, Lightweight Application Security Process) [21, 22] is a framework whose purpose is to provide an approach for locating security concerns in the early stages of the software development lifecycle. It actually consists of a set of process activities that can be integrated into any software development process. It also provides an extensive wealth of security resources that make implementing these activities reasonable. Jurjens [23] proposed an approach to support model-based security engineering using UML (Unified Modeling Language) by providing tool-support for the analysis of UML models (i.e., UMLsec models) for security requirements. This approach utilizes the automated theorem-prover (ATP) SETHEO to verify the security properties of the UMLsec model, which make use of cryptography such as cryptographic protocols. Cheng et al. [24, 25] proposed an information security engineering environment (ISEE) based on ISO/IEC information security standards. ISEE integrates various tools and functions for supporting continuous and consistent design, development, and management of the security facilities of information systems with high security requirements. McGraw [26, 27] proposed building security in SDLC, detailed approach for putting software security into practices. This approach provides guidance on how to build secure software and show gaps in the development process and to how to improve the process.

#### 2.1.3 Security Analysis-based Approaches

Several other studies [7, 8, 28-34] have focused on security requirements engineering or risk analysis as an early activity of software development. Nunes et al. [28] proposed a security engineering approach, named PSSS (Process to Support Software Security), based on the activities derived from SSE-CMM, ISO/IEC 15408, ISO/IEC 27002, and OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) [29, 30]. PSSS includes security activities such as software-based vulnerability, threat, and impact and risk assessments, which also contribute to information security strategic goals.

Evans et al. [7] introduced the mission-oriented risk and design analysis (Morda) developed by the US National Security Agency. The Morda helps system security engineers develop reasonable design strategies to build secure systems by supporting a framework for analyzing complex ISs risk postures. Wang et al. [8] proposed a process of security requirements that consists of nine steps and deals with the security requirements in the early stages of system design. They used a systematic approach to integrate software engineering process into develop security requirements. Mead et al. [31, 32] developed the security quality requirements engineering (SQUARE) methodology. SQUARE methodology provides a means for extracting, categorizing, and prioritizing security requirements for information systems and applications. It helps system engineers integrate security considerations
into the early stages of the software development lifecycle. Charles et al. [33] proposed a framework for security requirements representation and analysis. They defined what security requirements are, and presented a structure for satisfaction arguments for validating whether the system can satisfy the security requirements. Schmidt [34] proposed a threat and risk-driven methodology to security requirement engineering. This methodology extends the security engineering process using patterns (SEPP) [35] by a threat and risk-driven procedure to select adequate security mechanisms. However, most of these approaches do not focus specifically on the combination of risk analysis and software engineering disciplines.

3 Proposed Activities in SDLC

The main goal of the security engineering proposed in this paper is to provide integrated procedures for analyzing, designing, developing, testing, and maintaining secure EISs. In the current security engineering (or software engineering) environment, security-related activities are isolated, temporal, and inconsistent. However, in the proposed environment, these activities are integrated together and provide a holistic and consistent security management. Therefore, we can obtain maximum security strength of information systems from the proposed approach.

Fig. 1 illustrates the basic concept of a layered security engineering model, which is composed of three important components: security management, ESA, and security mechanism and services.

Security management describes the specific needs for managing security risk, including the security policy, as the logical model of organization’s business requirements for security and risk. The results of a security risk assessment, the risk assessment report, are also used to manage security risk mitigations (e.g., security mechanism, security service, and potential safeguards).

ESA is a main component of security engineering for implementing secure EISs. It provides the contextual, conceptual, logical, physical, component, and operational security of security-related policies, mechanisms, and procedures to give shape to a security management. The ESA proposed in this paper is expanded from the layered security architectures derived from SABSA (Sherwood Applied Business Security Architecture) [36] methodology.

Security Mechanisms & Services describes the detailed security technology for making concise ESA to provide the confidentiality, integrity, and availability of an organization’s information. Generally, security services are implemented as the objectives of security solutions. However, the compatibility and interoperability of security solutions are not guaranteed. That is, they do not provide holistic and integrated strategies. This paper proposes a way to map these security services into an ESA.

The layered security engineering model can be mapped into the activities of security engineering in
the SDLC as illustrated in Fig. 3. Each security-related activity is connected with other activities to support consistent security management. Furthermore, these activities are repeatedly performed throughout the iterative and incremental SDLC, but with different emphasis depending on where the activity is situated with the SDLC, and each activity will generate releases of various artifacts. A more detailed description about security activities in the SDLC will be presented in the following subsections.

### 3.1 Activities in the Requirement Analysis Phase

The proposed security engineering approach consists of several security-related activities that form a process to support the development of a more secure information system. The activity begins with security risk analysis (i.e., asset, threat, and vulnerability analysis) in the requirement analysis phase of SDLC. The main goal of the security risk analysis is to identify potential security problems and their impact. From the security risk analysis, the types of assets, threats, and vulnerabilities are identified and analyzed. The analysis includes what kinds of threats and vulnerabilities exist for a specific asset and the probabilities of threats that will occur. The security risk analysis enables us to develop secure information management and establish practical security policies for organization. In addition, it provides valuable analysis data for future risk estimation. Therefore, once the security risk analysis has been conducted, the security management in the maintenance phase can use various risk mitigation techniques to complete the process [5].

Simultaneously, security requirement analysis, which defines the required security properties, which the enterprise information systems should be satisfied, is conducted. The security requirement analysis is one of the critical parts of development process, which affect secure system development during all phase in SDLC. That is, understanding security requirements is a prerequisite to designing a secure EIS. If missing security requirements, security defects in system design will make untrusted test results. In order to make easy the treatment and specification of the security risk analysis and security requirements, several concepts and tools such as UMLSec [37], misuse cases [38], threat scenario [5], threat/attack trees [39], and security resources repository can be used. The result of security risk analysis and the security requirements play a role in the design of an ESA.

### 3.2 Activities in the Design Phase

The system design should be constructed from the report of risk analysis and the verified security requirements. First, in the design phase of SDLC, the output of three activities (i.e., threat, asset, and vulnerability analysis) is used to estimate security risk though risk assessment activity, and then

![Fig. 2 Overview of the Proposed Security-Related Activities and Relationship in SDLC](image-url)
suitable risk mitigations (e.g., security mechanism, security service, and potential safeguards) are selected and implemented. In risk assessment, security risk of organization or systems is evaluated by summing up all the risks of the systems components considering the existing threats in the core assets of the organization and the degree of vulnerability per threat [40].

Next, the ESA, which is composed of the six-layered security architecture, is implemented. As mentioned earlier, the ESA proposed in this paper is based on the SABSA, and it gives a six by six matrix of cells, which represents the whole model for the enterprise security architecture. Each security architecture (SA) represents the view of a different stakeholder (e.g., architect, designer, and developer etc.) in the process of specifying, designing, and constructing.

To implement secure information systems, correct design (i.e., ESA) is required. By asking these questions, we can make each SA that meet security requirements verified in the requirement analysis phase. The basic idea of SA modeling is to start with an abstract architecture (i.e., Contextual SA) of the EIS under development while more detailed architectures are built by refining the initial architecture. Finally, in order to find vulnerabilities in the ESA design, an ESA review activity is iteratively conducted.

3.3 Activities in the Coding Phase
In the coding phase, the selected safeguards, including the security mechanisms and services in the previous phase, are implemented, and the code for security issues is thoroughly reviewed in terms of whether it contains security vulnerabilities. In order to securely implement the safeguards and security services, software developer should avoid common programming errors (e.g., buffer overflows) by following secure coding method. There are many resources available [41-44] software developers can incorporate security into their code more securely. In code review activity, the code review is thoroughly performed by another programmer from a separate team to make sure all security requirements have been satisfied.

3.4 Activities in the Testing Phase
In the testing phase, the implemented safeguards and the enterprise information system are tested by both risk-based white- and black-box testing in terms of whether there are bugs at the implementation phase, or whether the EIS meets the security requirements based on the security risk analysis report. The risk-based white-box testing (also known as transparent testing) refers to analyze the safeguards and the program modules (or the information systems) at the level of the source code. It examines the internal logic and structure of the code and finds out which codes are behaving inappropriately. Optimization of the source code by revealing hidden errors is able to remove possible vulnerabilities. On the other hand, the risk-based black-box testing refers to analyze a running the safeguards and the information systems by probing it with various test cases which are derived from the security requirements and the output of risk analysis. This kind of testing doesn’t require any knowledge of the internal structure or coding in the program.

Besides both white- and black-box testing, in order to assure the security of information systems and its services, a penetration testing is required. The penetration testing is about to verify the absence of insecure functionality (e.g., security threats and vulnerabilities), so that security weaknesses can be fixed before they arise. It evaluates computer and network security by simulating an attack on a computer system or network from internal and external threats. The process involves an active analysis of the system for any potential vulnerabilities that could result from poor or improper system configuration, both known and unknown hardware, or software flaws, or operational weaknesses in process or technical countermeasures. This analysis is carried out from the position of a potential attacker and can involve active exploitation of security vulnerabilities.

3.5 Activities in the Management Phase
The deployment phase constitutes the process that installs the system and makes it operational in the production environment. This phase can be one of the most critical in SDLC since the production environments and considerations for integration issues can sometimes be ignored. For example, sometimes the system runs differently in the production environment even though it has been validated and verified in the test phase. That is, when complex subsystems or large systems are integrated together, we need to ensure that the system is secure. To achieve this, a security deployment review based on the security requirements is conducted in the deployment phase.

3.6 Activities in the Deployment Phase
The goal of security management is to identify, evaluate, and manage key security risks that impact an organization’s stability and thus its ability to achieve its objectives and strategies. It is a continuous process of establishing risk management objectives, assessing risks within the context of established tolerances, developing strategies and implementing risk management processes, and monitoring and reporting upon those processes. In the maintenance phase, the risk assessment report, which is the summary and conclusions of the risk analysis, is utilized to manage the residential and potential security vulnerabilities in EISs. Furthermore, an ESA review activity is conducted.

4 Conclusion
Nowadays, security in the EISs plays an important role for successful business. In order to implement security features to EISs, many approaches for the security engineering have been proposed. However, most of the research studies on the development of secure information systems are focused on isolated, inconsistent software engineering-based or risk analysis-based technologies. Therefore, in order to provide a holistic and consistent security management in all phases of SDLC, the new concept of security engineering methodology is presented.

The security engineering methodology proposed in this paper covers the entire enterprise security-related activities especially from security engineer’s perspective for developing secure enterprise information systems. It combines security risk control, ESA, and security management as an integrated framework. Moreover, the security mechanisms and services are designed, implemented, and supported as an essential part of the enterprise information systems. In addition, the security-related activities provide a way to support design, coding, testing, and maintenance for consistent security management so that security engineer can understand and manage security features more easily in SDLC. In this paper, we proposed a new security engineering methodology for the development of secure EISs, especially focused on the security-related activities. In future work, we aim to propose the detailed artifacts in each activity of security engineering methodology, and tool support for automated security management in SDLC.

Acknowledgements

References:
Workshop on Secure Systems Methodologies using Patterns (SPatterns), pp.734-738, 2007


