AN INTEGRATED RISK ANALYSIS FRAMEWORK FOR IMPROVING SECURITY OF SOFTWARE DEVELOPMENT LIFE CYCLE

Tsung-Han Yang, Department of Information Management, National Chung Cheng University, Chia-Yi County, Taiwan, ROC, 886-5-2721500, h730615@hotmail.com
Cheng-Yuan Ku, Department of Information Management, National Chung Cheng University, Chia-Yi County, Taiwan, ROC, 886-5-2721500, cooperku@mis.ccu.edu.tw
Yung-Ting Chuang, Department of Information Management, National Chung Cheng University, Chia-Yi County, Taiwan, ROC, 886-5-2721500, ytchuang@mis.ccu.edu.tw

ABSTRACT

Software development life cycle (SDLC) plays an essential role in most software construction projects. Unfortunately, the system users or developers are rarely concerned with the security requirements for these processes. Even though some researchers have attempted to investigate specific methodologies in the area of secure software engineering, there are still few frameworks that can address the security issues.

The primary objective of this research is to design a unified framework of architectural risk analysis to fulfill the security requirements. The proposed framework can help system users to: 1) systematically handle breach issues that exist in software architecture, 2) fulfill security requirements in information security management, and 3) handle benchmark issues in risk assessment.

Keywords: software security engineering, secure software development life cycle, risk analysis, security requirement

I. INTRODUCTION

Software is ubiquitous and widely adopted by various organizations since many workflows and services provided by organizations are highly dependent on software systems. However, sensitive information and application systems are becoming more vulnerable to unintended and unauthorized use nowadays. Hence compliance with confidentiality, integrity, and availability requirements is rather important for implementing and protecting the primary and supporting information assets.

According to [6], the countermeasures for the current software applications are no longer adequate. In this survey, Gartner identifies application security as the major concern for many chief information officers (CIOs). As a result, business leaders and informed consumers are more aware of the scarcity of security practitioners and the requisite competencies when they deal with the software security issues [1]. This phenomenon reflects that software security have rapidly become a critical issue in the field of information security management, business continuity management, disaster recovery, incident response, and risk management.

Systematically secure software should be implemented as such it would have ability to sustain its productivity and efficiency, as well as having ability to continue operating in the presence of threatening

events. Currently, the secure software is still mostly constructed by considering the guidelines, best practices, or undocumented expert knowledge. The lack of systematic and structured disciplines in software development often lead to insecure software with exploitable weaknesses.

II. THEORETICAL BACKGROUND

We consider the following three works for designing the risk analysis framework because of their relevance to the domain of secure SDLC: 1) CLASP (Comprehensive, Lightweight Application Security Process of OWASP [4], 2) Microsoft's Security Development Life cycle (SDL) [2], and 3) McGraw' Touchpoints [3]. We chose above standards in order to elicit an integrated reference framework, which includes the needed processes and activities for security of the software development life cycle. CLASP, contributed to and reviewed by several leading security companies of the OWASP consortium, is considered as lightweight and more affordable for small organizations with less strict security demands. SDL, used by Microsoft internally for product development, is considered as more heavyweight, rigorous, and more suitable for large organizations. Touchpoints, applying experiences from industrial projects in order to first distill an approach and then to ensure the proposed activities, are both executable and feasible.

III. METHODS

We propose a unified framework by considering the methodologies (CLASP, SDL, and Touchpoints) mentioned above with the focus on architectural risk analysis and its related phases. In order to develop a reasonable framework, our unified framework filters out the useful component of the referenced framework. Our architectural risk analysis frameworks can be divided into the following components: asset analysis, ambiguity analysis, attack resistance analysis, weakness analysis, risk strategy analysis, and measure software security. For clearly presenting the architectural risk analysis framework, we use the knowledge model of the CommonKADS Methodology [5]. CommonKADS, a method for both knowledge engineering and knowledge management, is prominently used for defining the structure of the expertise models. We decompose the main tasks into detailed inferences and present the task knowledge of the architectural risk analysis framework in Figure 1.

IV. VALIDATION

Until now, we have already finished implementing this unified framework of architectural risk analysis and still continue improving it. However, the practical test is under preparation. That is, we will adopt the prototype of this framework to analyze the architectural risk in a real system implemented in some organization. Then, from these real data, we will perform the efficiency and performance analysis. Moreover, we will also compare our framework with other similar projects for further validating the efficiency and performance.

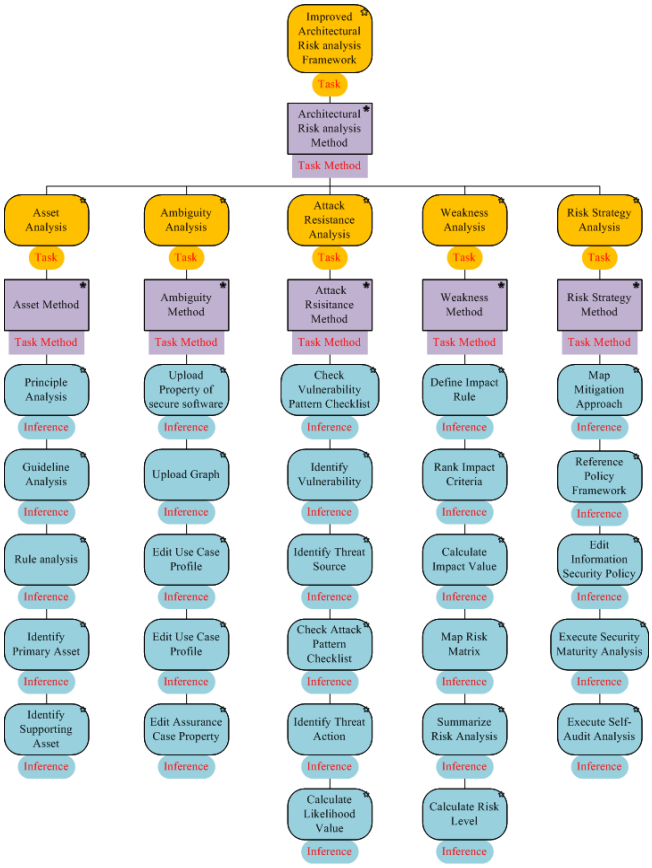

Figure 1 Task knowledge of the proposed architectural risk analysis framework

ACKNOWLEDGEMENT

This research is supported by NSC 102-2410-H-194-101 of National Science Council, Taiwan, R.O.C.

REFERENCES

[1] Carey, A., *2006 Global Information Security Workforce Study, Framingham, MA: IDC, 2006*. https://www.isc2.org/uploadedFiles/Industry_Resources/wfs_gov.pdf

[2] Howard, M. and Lipner, S., *The Security Development Lifecycle (SDL): A Process for Developing Demonstrably More Secure Software*, Microsoft Press, 2006.

[3] McGraw, G., *Software Security: Building Security In*, Addison Wesley, 2006.

[4] *OWASP, Comprehensive, lightweight application security process*, http://www.owasp.org, 2006.

[5] Schreiber, G., Akkermans, H., Anjewierden, A., Hoog, R. D., Shadbolt, N., Velde, W. V. D., and Wielinga, B., *Knowledge engineering and management: The CommonKADS Methodology*, MA: MIT Press, 2000.

[6] Tohmatsu, D. T., *2007 Global Security Survey: The Shifting Security Paradigm, 2007*. http://www.deloitte.com/