January 2010

Integrating Software Assurance into the Software Development Life Cycle (SDLC)

Full Article Available Online at: Intellectbase and EBSCOhost | JISTP is indexed with Cabell's, JournalSeek, etc.

JOURNAL OF INFORMATION SYSTEMS TECHNOLOGY & PLANNING

Journal Homepage: www.intellectbase.org/journals | ©2010 Published by Intellectbase International Consortium, USA

INTEGRATING SOFTWARE ASSURANCE INTO THE SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

Maurice Dawson[1,2], Darrell Norman Burrell[3,4], Emad Rahim[5,6] and Stephen Brewster[7]

[1]Morgan State University, USA, [2]Colorado Technical University, USA, [3]A. T. Still University, USA, [4]Virginia International University, USA, [5]Morrisville State College, USA, [6]Walden University, USA and [7]Capitol College, USA

ABSTRACT

This article examines the integration of secure coding practices into the overall Software Development Life Cycle (SDLC). Also detailed is a proposed methodology for integrating software assurance into the Department of Defense Information Assurance Certification & Accreditation Process (DIACAP). This method for integrating software assurance helps in properly securing the application layer as that is where more than half of the vulnerabilities lie in a system.

Keywords: *Secure Coding; Software Assurance; Secure Software Development Lifecycle.*

INTRODUCTION

In the past software product stakeholders did not view software security has high priority. It was believed that a secure network infrastructure would provide the level of protection needed against malicious attacks. In recent history network security alone has proved inadequate against such attacks. Users have been successful in penetrating valid channels of authentication through techniques such as cross site scripting, Structured Query Language (SQL) Injection, and Buffer Overflow exploitation. In such cases system assets were compromised and both data and organizational integrity were damaged. The Gartner Group reports that more than 70 percent of current business security vulnerabilities are found within software applications rather than the network boundaries (Aras, Barbara, & Jeffrey, 2008). A focus of application security emerged in order to reduce the risk of poor software development, integration, and deployment. Through this need software assurance quickly became an Information Assurance (IA) focus area in the financial, government, and manufacturing sectors to reduce the risk of unsecure code.

MEETING DEPARTMENT OF DEFENSE (DOD) DEMANDS

The United States Army is the primary customer may defense contractors. The Army is managed and ran by the Department of Defense (DoD). The primary objective of the DoD is to provide military forces in an effort to deter war and to protect the security of the United States of America. The Department of Defense (DoD) has addressed security through governance issued under the Office of Management and Budget (OMB) Circular A-130. The focus of Information Technology security was further derived by DoD Directive 8500.2. It specifically states that all IA and IA-enabled IT products incorporated into DoD information systems shall be configured in accordance with DoD-approved security configuration guidelines. On April 26, 2010, the DoD released the third version of the Application Security and Development Security Technical Implementation Guide (STIG) provided by the Defense Information Systems Agency (DISA). This document provides DoD guidelines and requirements for integrating security throughout the software development lifecycle. As a leader in the development and fielding of unmanned aerial vehicles, it is our responsibility to meet the needs and demands of our customer to the best of our ability. Therefore we must adhere to the integration of security throughout our SDLC in an effort to meet the requirements of our customer.

COMMON INDUSTRY STANDARDS FOR SOFTWARE DEVELOPMENT

Software engineering is the process of developing and implementing algorithms. Software Assurance is the level of confidence that software algorithms function as specified free of intentional and unintentional vulnerabilities. Generally an organization's software development life cycle is based upon the waterfall model. There are five phases to the Software Development Life Cycle (SDLC). The figure below details a process flow diagram of the waterfall SDLC.

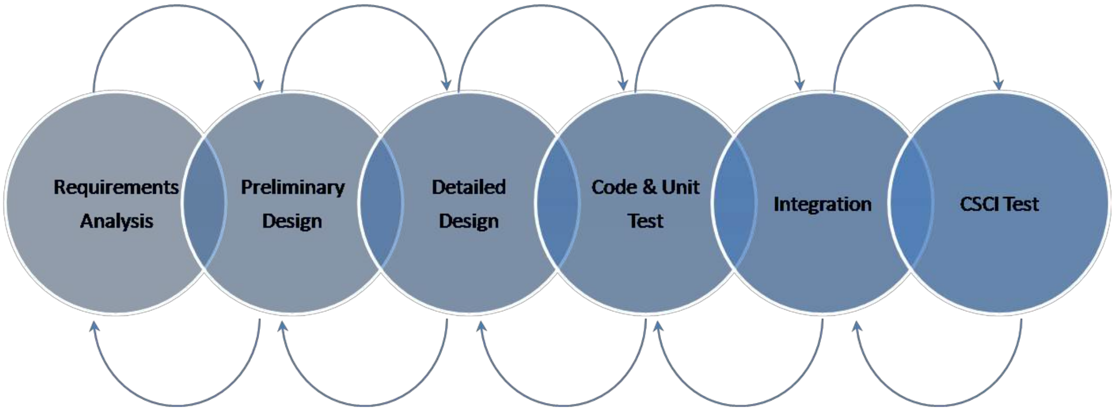

Figure 2: *Common Defense & Aerospace SDLC*

An allocated baseline is created during the Requirements and Analysis phase. This baseline contains all of the requirements for a specific system allocated across four different functional areas. Once each functional area lead identifies its allocated requirements as correct, the allocated baseline becomes a verified baseline. Software is one of the four functional areas in which system requirements are allocated. These requirements are then used to design code, integrate and test a completed software configuration item within the system. The IA security controls for the system are identified during the requirements and analysis phase. They are provided by customer and implemented through the Defense Information Assurance Certification and Accreditation Process (DIACAP) in compliance with DoD Instruction (DoDI) 8510.01. The respective Program Management Office of the DoD for provides an organization's IA department with a list of IA requirements that are to be met. These requirements serve as the DIACAP Implementation Plan (DIP), which must be executed in order to reduce the security risk of the system to an acceptable level and receive an Authority To Operate (ATO). The ATO is needed in order for our systems to be fielded within the DoD network. However, the execution of the DIP occurs during the CSCI Test phase of the SDLC. Therefore any and all vulnerabilities are being identified and mitigated after the software has been designed, developed, unit tested, and submitted for computer software configuration item testing.

PROCESS TO SECURE SOFTWARE CODE

In the event of a vulnerability finding, the software code may require redesign and implementation. This iterative cycle is costly in time and resources. To truly understand security threats to a system, security must be addressed beginning with the initiation phase of the development process. For an organization this means they must allow the IA controls and requirements to drive design and influence the software requirements. Therefore, any identified security threats found during the requirements and analysis phase will drive design requirements and implementation. Security defects discovered can then be addressed at a component level before implementation. The cost of discovery and mitigation can be absorbed within the review, analysis and quality check performed during the design, and implementation of our SDLC. The resultant product is one with security built in rather than security retrofitted. A study was performed by the IBM System Science Institute in order determine the relative cost in order to fix defects within the SDLC. Figure 2 displays their findings.

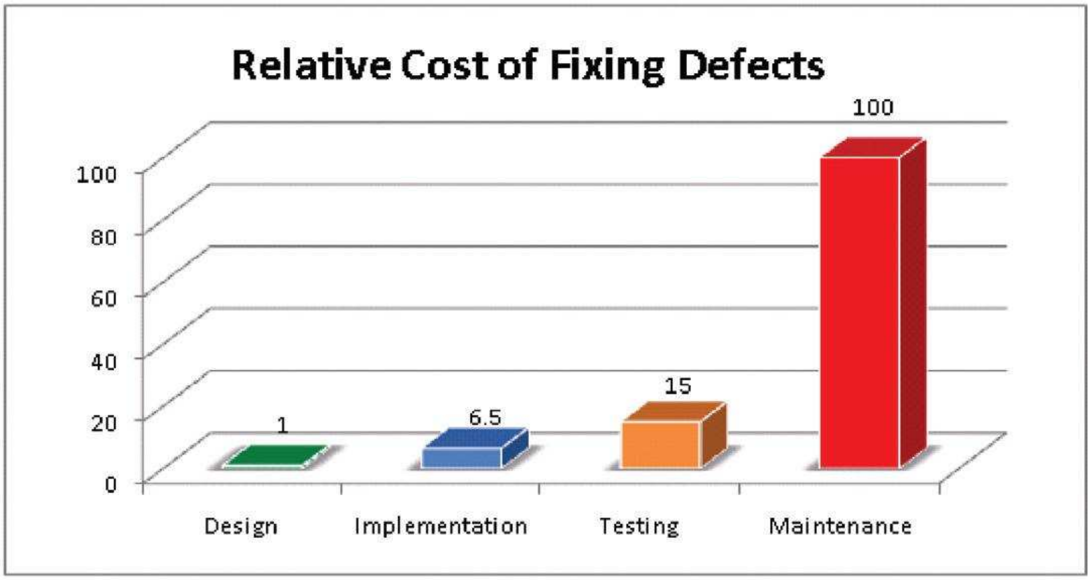

Figure 3: *IBM System Science Institute Relative Cost of Fixing Defects*

Defects found in testing were 15 times more costly than if they were found during the design phase and 2 times more than if found during implementation.

SECURE SDLC

DoDI 8500.2, IA Implementation, states that the Information Systems Security Engineer (ISSE) must work with the system architects, engineers, and developers to ensure that IA controls are designed and implemented into the system throughout the development process. Though this requirement is for government entities, it serves as a guide into how an organization could also integrate security into software development. The software development process which an organization should have should serve as the baseline process in which the integration of security controls and activities must take place. The objectives are as follows for secure development:

- Reduce cost of fixing vulnerabilities.
- Increase the integrity, availability, and confidentiality of our software.

- Conform to DoD standards of secure software development

The security activities involved should seamlessly interface with existing activities found with the organization's SDLC. In order to achieve such a unified process we must first examine the activities required within a Secure SDLC. The International Information Systems Security Certification Consortium, Inc (ISC)2, a global leader in the creation of security certification standards, has published best practices for integrating security into the system development life cycle. The security activities suggested by (ISC)2 should be further derived into the secure SDLC using existing SDLC phase definitions. The following diagram depicts the activities within the Secure SDLC with the Department:

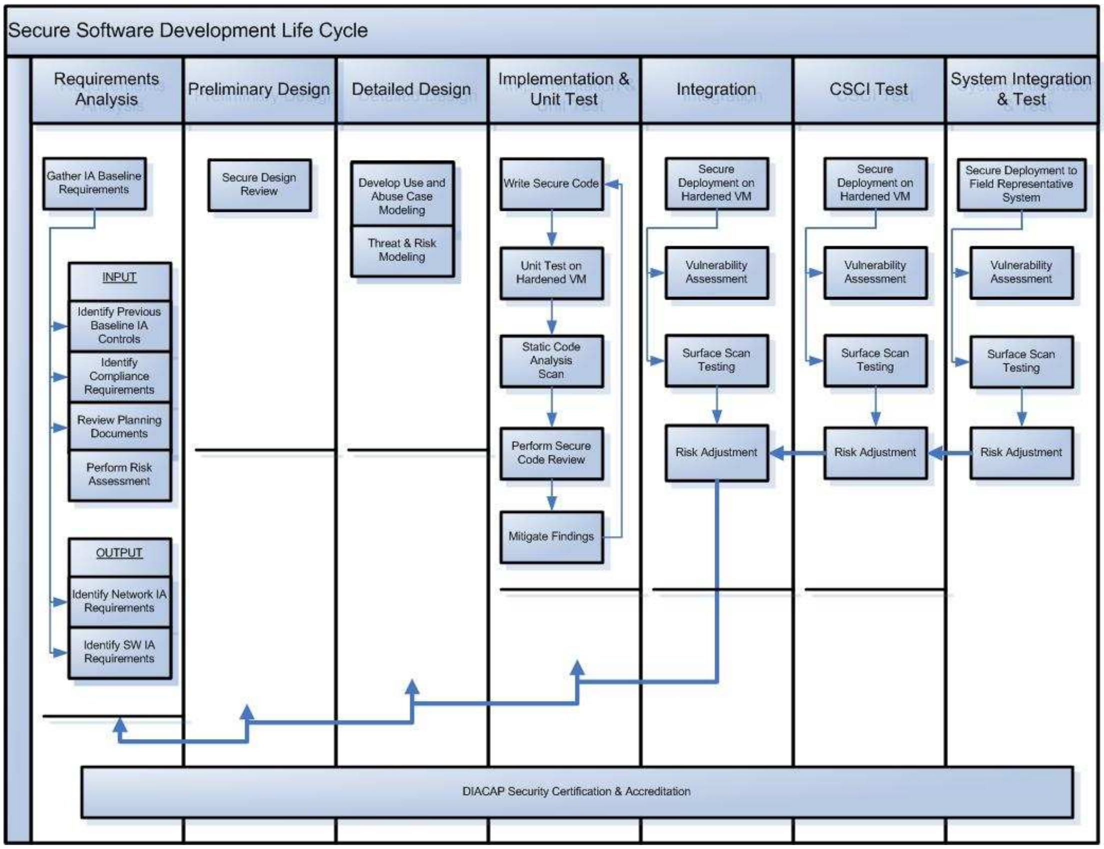
Secure Software Development Life Cycle
Requirements Analysis
Preliminary Design
Detailed Design
Implementation & Unit Test
Integration
CSCI Test
System Integration & Test
Gather IA Baseline Requirements
INPUT
Identify Previous Baseline IA Controls
Identify Compliance Requirements
Review Planning Documents
Perform Risk Assessment
OUTPUT
Identify Network IA Requirements
Identify SW IA Requirements
Secure Design Review
Develop Use and Abuse Case Modeling
Threat & Risk Modeling
Write Secure Code
Unit Test on Hardened VM
Static Code Analysis Scan
Perform Secure Code Review
Mitigate Findings
Secure Deployment on Hardened VM
Vulnerability Assessment
Surface Scan Testing
Risk Adjustment
Secure Deployment on Hardened VM
Vulnerability Assessment
Surface Scan Testing
Risk Adjustment
Secure Deployment to Field Representative System
Vulnerability Assessment
Surface Scan Testing
Risk Adjustment
DIACAP Security Certification & Accreditation

Figure 4: *Industry Standard Secure Software Development Life Cycle Activities*

Using this outlined Secure SDLC, security can be addressed over the course of the software's development life cycle. DIACAP artifacts can now be gathered during each phase and compiled in order to deliver a more complete system profile. Installation and deployment activities will incorporate security controls in order to maintain the security posture of the system from implementation through integration and test. Threat analysis, system modeling, and security design review will provide the opportunity to identify system exploitable attributes before any code is written. Vulnerability assessments using static code scan tool and surface scan tools will provide output for determining if the software was developed to specifications identified during the security design review. The end product will be both hacker resistant and security hardened.

SUMMARY

The Secure SDLC has as its base components all of the activities and security controls needed to develop DoD compliant and industry best practices hardened software. A knowledgeable staff as well as secure software policies and controls is required in order to truly prevent, identify, and mitigate exploitable vulnerabilities within developed systems. Not meeting the least of these activities found within the secure SDLC provides an opportunity for misuse of system assets from both insider and outsider threats. Security is not simply a network requirement, it is now an Information Technology requirement which includes the development of all software for the intent to distribute, store, and manipulate information. Therefore, as a developer in the defense industry contractors must implement the highest standards of development in order to insure the highest quality of products for its customers and the lives which they protect.

REFERENCES

Aras, O, Barbara, C, & Jeffrey, L. (2008). Secure software development-the role of it audit. *Information Systems Control Journal*, *4*.

Defense Information Systems Agency, DISA Field Security Operations. (2006). *Application services security technical implementation guide,* Washington, DC: Defense Information Systems Agency. Retrieved from http://iase.disa.mil/stigs/stig/application-services-stig-v1r1.pdf

Defense Information Systems Agency, DISA Field Security Operations. (2010). *Application services security technical implementation guide,* Washington, DC: Defense Information Systems Agency. Retrieved from http://iase.disa.mil/stigs/stig/

Paul, M. (2008). *The need for software security*. Retrieved from https://www.isc2.org/uploadedFiles/(ISC)2_Public_Content/Certification_Programs/CSSLP/CSSLP_WhitePaper.pdf

Dowd, M, McDonald, J, & Schuh, J. (2007). *The art of software security assessment*. Boston, MA: Pearson Education, Inc.

Maxon, R. (2008). *Software assurance best practices for air force weapon and information technology systems – are we bleeding?.* Published manuscript, Department of Systems and Engineering Management, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH. Retrieved from http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA480286&Location=U2&doc=GetTRDoc.pdf