Software Engineering Institute | Carnegie Mellon University

Software Security Engineering: A Guide for Project Managers

Gary McGraw
Julia H. Allen
Nancy Mead
Robert J. Ellison
Sean Barnum

May 2013

ABSTRACT: Software is ubiquitous. Many of the products, services, and processes organizations use and offer are highly dependent on software to handle the sensitive and high-value data on which people's privacy, livelihoods, and very lives depend. National security—and by extension citizens' personal safety—relies on increasingly complex, interconnected, software-intensive information systems—systems that in many cases use the Internet or Internet-exposed private networks as their means for communication and transporting data.

INTRODUCTION

Dependence on information technology makes software security a key element of business continuity, disaster recovery, incident response, and national security. Software vulnerabilities can jeopardize intellectual property, consumer trust, business operations and services, and a broad spectrum of critical applications and infrastructures, including everything from process control systems to commercial application products.

The integrity of critical digital assets (systems, networks, applications, and information) depends on the reliability and security of the software that enables and controls those assets. However, business leaders and informed consumers have growing concerns about the scarcity of practitioners with requisite competencies to address software security [Carey 2006]. They have concerns about suppliers' capabilities to build and deliver secure software that they can use with confidence and without fear of compromise. Application software is the primary gateway to sensitive information. According to the Deloittesurvey of 169 major global financial institutions, 2007 Global Security Survey: The Shifting Security Paradigm [Deloitte 2007], current application software countermeasures are no longer adequate. In the survey, Gartner identifies application security as the number one issue for chief information officers (CIOs).

The absence of security discipline in today's software development practices often produces software with exploitable weaknesses. Security-enhanced processes and practices—and the skilled people to manage them and perform

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

Phone: 412-268-5800
Toll-free: 1-888-201-4479

www.sei.cmu.edu

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **MAY 2013**	2. REPORT TYPE	3. DATES COVERED **00-00-2013 to 00-00-2013**

4. TITLE AND SUBTITLE **Software Security Engineering: A Guide for Project Managers**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Software Engineering Institute,Pittsburgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Software is ubiquitous. Many of the products, services, and processes organizations use and offer are highly dependent on software to handle the sensitive and high-value data on which people???s privacy, livelihoods, and very lives depend. National security???and by extension citizens??? personal safety??? relies on increasingly complex, interconnected, software-intensive information systems???systems that in many cases use the Internet or Internet-exposed private networks as their means for communication and transporting data.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a REPORT **unclassified**	b ABSTRACT **unclassified**	c THIS PAGE **unclassified**	**Same as Report (SAR)**	**11**	

them—are required to build software that can be trusted to operate more securely than software being used today.

That said, there is an economic counter-argument, or at least the perception of one. Some business leaders and project managers believe that developing secure software slows the process and adds to the cost while not offering any apparent advantage. In many cases, when the decision reduces to “ship now” or “be secure and ship later,” “ship now” is almost always the choice made by those who control the money but have no idea of the risks. Information to combat this argument, including how software security can potentially reduce cost and schedule, is becoming available based on earlier work in software quality and the benefits of detecting software defects early in the life cycle along with documented experiences such as Microsoft’s Security Development Lifecycle.

THE GOAL OF SOFTWARE SECURITY ENGINEERING

Software security engineering is using practices, processes, tools, and techniques that enable you to address security issues in every phase of the software development life cycle (SDLC). Software that is developed with security in mind is typically more resistant to both intentional attack and unintentional failures. One view of secure software is software that is engineered “so that it continues to function correctly under malicious attack” [McGraw 2006] and is able to recognize, resist, tolerate, and recover from events that intentionally threaten its dependability. Broader views that can overlap with software security (for example, software safety, reliability, and fault tolerance) include proper functioning in the face of unintentional failures or accidents and inadvertent misuse and abuse, as well as reducing software defects and weaknesses to the greatest extent possible regardless of their cause.

The goal of software security engineering is to build better, defect-free software. Software-intensive systems that are constructed using more securely developed software are better able to

- continue operating correctly in the presence of most attacks by either resisting the exploitation of weaknesses in the software by attackers or tolerating the failures that result from such exploits
- limit the damage resulting from any failures caused by attack-triggered faults that the software was unable to resist or tolerate and recover as quickly as possible from those failures

SOFTWARE SECURITY PRACTICES

No single practice offers a universal silver bullet for software security. With this in mind, Software Security Engineering: A Guide for Project Managers provides software project managers with sound practices that they can evaluate and selectively adopt to help reshape their own development practices. The objective is to increase the security and dependability of the software produced by these practices, both during its development and its operation.

The book (and material referenced on the Build Security In Web site described below) identifies and compares potential new practices that can be adapted to augment a project's current software development practices, greatly increasing the likelihood of producing more secure software and meeting specified security requirements. As one example, assurance cases can be used to assert and specify desired security properties, including the extent to which security practices have been successful in satisfying security requirements.

Software developed and assembled using software security practices should contain significantly fewer exploitable weaknesses. Such software can then be relied on to more capably recognize, resist or tolerate, and recover from attacks and thus function more securely in an operational environment. Project managers responsible for ensuring that software and systems adequately address their security requirements throughout the SDLC can review, select, and tailor guidance from the book, the Build Security In Web site, and the sources cited throughout the book as part of normal project management activities.

The five key take-aways of Software Security Engineering are as follows:

1. Software security is about more than eliminating vulnerabilities and conducting penetration tests. Project managers need to take a systematic approach to incorporate the sound software security practices into their development processes. Examples include security requirements elicitation, attack pattern and misuse/abuse case definition, architectural risk analysis, secure coding and code analysis, and risk-based security testing.
2. Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks.
3. Software security initiatives should follow a risk management approach to identify priorities and what is good enough, understanding that software security risks will change throughout the development lifecycle. Risk management reviews and actions are conducted during each phase of the SDLC.
4. Developing secure software depends on understanding the operational context in which it will be used. This context includes conducting end-to-end analysis of cross-system work processes, working to contain and recover from failures using lessons learned from business continuity, and exploring

failure analysis and mitigation to deal with system and system of system complexity.

5. Project managers and software engineers need to learn to think like an attacker in order to address the range of things that software should not do and how software can better resist, tolerate, and recover when under attack. The use of attack patterns and misuse/abuse cases throughout the SDLC encourages this perspective.

Practice Maturity and Relevance

As a community, we recognize that some software security practices are in broader use, and thus more tested and mature, than others, such as security coding practices and testing for vulnerabilities. As a practice description and selection aid, descriptive tags mark the book's sections and key practices in two practical ways:

- Identifying the content's relative "maturity of practice" as follows:
  - L1: The content provides guidance for how to think about a topic for which there is no proven or widely accepted approach. The intent of the description is to raise awareness and aid in thinking about the problem and candidate solutions. The content may also describe promising research results that may have been demonstrated in a constrained setting.
  - L2: The content describes practices that are in early pilot use and are demonstrating some successful results.
  - L3: The content describes practices that are in limited use in industry or government organizations, perhaps for a particular market sector.
  - L4: The content describes practices that have been successfully deployed and are in widespread use. These practices can be used with confidence. Experience reports and case studies are typically available.
- Identifying the designated audiences for which each chapter section or practice is most relevant:
  - E: executive and senior managers
  - M: project and mid-level managers
  - L: technical leaders, engineering managers, first line managers, and supervisors

BUILD SECURITY IN: A KEY RESOURCE

Since 2004, the U.S. Department of Homeland Security Software Assurance Program has sponsored development for the Build Security In (BSI) Web site, which is one of the significant resources used in developing Software Security Engineering. BSI content is based on the principle that software security is fundamentally a software engineering problem and must be managed in a systematic way throughout the SDLC.

BSI contains and links to a broad range of information about sound practices, tools, guidelines, rules, principles, and other knowledge to help project managers deploy software security practices and build secure and reliable software. Contributing authors to this book and the articles appearing on the BSI site include senior staff from the Carnegie Mellon Software Engineering Institute (SEI) and Cigital, Inc., as well as other experienced software and security professionals.

BSI content is referenced throughout the book. Readers can consult BSI for additional details, ongoing research results, and information about related Web sites, books, and articles.

Start the Journey

As software and security professionals, we will never be able to get ahead of the game by addressing security solely as an operational issue. Attackers are creative, ingenious, and increasingly motivated by financial gain. They have been learning how to exploit software for several decades; the same is not true for software engineers, and we need to change this. Given the extent to which our nations, our economies, our businesses, and our families rely on software to sustain and improve our quality of life, we must make significant progress in putting higher quality and more secure software into production. The practices described in Software Security Engineering serve as a useful starting point.

Each project manager needs to carefully consider the knowledge, skills, and competencies of their development team, their organizational culture’s tolerance (and attention span) for change, and the degree to which sponsoring executives have bought in (a prerequisite for sustaining any improvement initiative). In some cases, it may be best to start with secure software coding and testing practices given that these are the most mature, have a fair level of automated support, and can demonstrate some early successes, providing visible benefit to help software security efforts gain support and build momentum. On the other hand, secure software requirements engineering and architecture and design practices offer opportunities to address more substantive root cause issues early in the life cycle that if left unaddressed will show up in code and test. Practice selection and tailoring are specific to each organization and project based on objectives, constraints, and the criticality of the software under development.

Project managers and software engineers need to better understand what constitutes secure software and develop their skills to think like an attacker so this mindset can be applied throughout the SDLC. The book describes practices to get this ball rolling, such as attack patterns and assurance cases. Alternatively, if you have access to experienced security analysts, adding a few of them to your development team can get this jump started.

Two of the key project management practices are (1) defining and deploying a risk management framework to help inform practice selection and determine where best to devote scarce resources and (2) identifying how best to integrate software security practices into the organization's current software development life cycle.

John Steven states [Steven 2006]

> *"Don't demand teams to begin conducting every activity on day one. Slowly introduce the simplest activities first, then iterate.*
>
> *"[Have] patience. It will take at least three to five years to create a working, evolving software security machine. Initial organization-wide successes can be shown within a year. Use that time to obtain more buy-in and a bigger budget."*

Clearly there is no one-size-fits-all approach. Project managers and their teams need to think through the choices, define their tradeoff and decision criteria, learn as they go, and understand that this effort requires continuous refinement and improvement.

IN CLOSING

Sound software security engineering practices should be incorporated throughout the entire software development life cycle. Software Security Engineering is one resource that captures both standard and emerging software security practices and explains why they are needed to develop more security-responsive and robust systems.

REFERENCES

[Allen 2008a] Allen, Julia; Barnum, Sean; Ellison, Robert; McGraw, Gary; Mead, Nancy. Software Security Engineering: A Guide for Project Managers, Addison-Wesley, 2008.

[Allen 2008b]	Allen, Julia & Pollak, William. “Building More Secure Software.” CERT Podcast Series: Security for Business Leaders,” May 2008. http://www.cert.org/podcast/show/20080527allen.html
[Carey 2006]	Carey, Allan. “2006 Global Information Security Workforce Study.” Framingham, MA: IDC, 2006.
[Deloitte 2007]	Deloitte Touche Tohmatsu. 2007 Global Security Survey: The Shifting Security Paradigm. September 2007.
[McGraw 2006]	McGraw, Gary. Software Security: Building Security In. Boston, MA: Addison-Wesley Professional, 2006 (ISBN 0-321-35670-5).
[Steven 2006]	Steven, John. “Adopting an Enterprise Software Security Framework.” IEEE Security & Privacy 4, 2 (March/April 2006): 84–87. https://buildsecurityin.us-cert.gov/resources/building-security-in/adopting-an-enterprise-software-security-framework

For additional information about the book, including a full table of contents, please refer to:

http://www.sei.cmu.edu/publications/books/cert/software-security-engineering.html

http://www.informit.com/store/product.aspx?isbn=032150917X

AUTHORS

Julia H. Allen

Julia H. Allen is a senior member of the technical staff within the CERT Program at the Software Engineering Institute (SEI). In addition to her work in software security and assurance, Ms. Allen is engaged in developing and transitioning executive outreach programs in enterprise security and governance. She is the author of The CERT Guide to System and Network Security Practices (Addison-Wesley, June 2001), Governing for Enterprise Security (CMU/SEI-2005-TN-023, 2005), and the CERT Podcast Series: Security for Business Leaders (2006/2008).

Sean Barnum

Sean Barnum is a Principal Consultant at Cigital and is technical lead for their federal services practice. He has over 20 years of experience in the software industry in the areas of development, software quality assurance, quality management, process architecture and improvement, knowledge management, and security. He is very active in the software assurance community and is involved in numerous knowledge standards-defining efforts, including the Common Weakness Enumeration (CWE), the Common Attack Pattern Enumeration and Classification (CAPEC), and other elements of the Software Assurance Programs of the Department of Homeland Security and the Department of Defense. He is also the lead technical subject matter expert for the Air Force Application Software Assurance Center of Excellence.

Robert J. Ellison

As a member of the Survivable Systems Engineering Team within the CERT Program at the SEI, Robert Ellison has served in a number of technical and management roles. Mr. Ellison regularly participates in the evaluation of software architectures and contributes from the perspective of security and reliability measures. His research draws on that experience to integrate security issues into the overall architecture design process. His current work explores developing reasoning frameworks to help architects select and refine design tactics to mitigate the impact of a class of cyber attacks. He was a member of the Carnegie Mellon University team that wrote the proposal for the SEI; he joined the new FFRDC in 1985 as a founding member.

Gary McGraw

Gary McGraw is the CTO of Cigital, Inc., a software security and quality consulting firm with headquarters in the Washington, D.C. area. He is a globally recognized authority on software security and the author of six best selling books on this topic. The latest, Exploiting Online Games, was released in 2007. His other titles include Java Security, Building Secure Software, Exploiting Software, and Software Security; and he is editor of the Addison-Wesley Software Security series. Dr. McGraw has also written over 90 peer-reviewed scientific publications, authors a monthly security column for darkreading.com, and is frequently quoted in the press.

Nancy R. Mead

Nancy R. Meadis a senior member of the technical staff in the Survivable Systems Engineering Group within the CERT Program at the SEI. She is a faculty member in the Master of Software Engineering and Master of Information Systems Management programs at Carnegie Mellon University. Her research interests are in the areas of information security, software requirements engineering, and software architectures.Dr. Mead has more than 100 publications and invited presentations.

LINKS

Book page on Software Engineering Institute site
https://buildsecurityin.us-cert.gov/redirect?url=http%3A%2F%2Fwww.sei.cmu.edu%2Fpublications%2Fbooks%2Fcert%2Fsoftware-security-engineering.html

Book site
https://buildsecurityin.us-cert.gov/redirect?url=http%3A%2F%2Fwww.softwaresecurityengineering.com%2F