Software is ubiquitous. Many of the products, services, and processes that 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 relies on 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.

Dependence on IT 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 functions, and national security. Software vulnerability, disaster recovery, incident response, and security are key elements of business continuity, and software defect and weakness reduction to the greatest extent possible (regardless of its cause).

The goal of software security engineering is to build better, defect-free software. The book: “Software Security Engineering: A Guide for Project Managers” [1]—and its key resource, the Build Security In (BSI) Web site—provide software project managers with sound practices that they can evaluate and selectively adopt to help reshape their own development practices. Software developed and assembled using these practices should contain significantly fewer exploitable weaknesses.

No single practice offers a universal silver bullet for software security. With this in mind, “Software Security Engineering” 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 BSI Web site at <https://build securityin.us-cert.gov>—identify and compare potential new practices that can be adapted to augment a project’s current software development practices. These resources also greatly increase the likeliest...
hood 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 be relied on to more capably recognize, resist or tolerate, and recover from attacks—in turn functioning 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 and Web site as part of normal project management activities.

The five key takeaways from the book 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 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 life cycle. Risk management reviews and actions are conducted during each SDLC phase.

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-systems complexity.

5. Project managers and software engineers need 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 vulnerability testing. As a practice description and selection aid, descriptive tags mark the book’s sections and key practices in two practical ways:

1. Identifying the content’s relative maturity of practice as follows:
   - **Maturity Level 1 (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.
   - **Maturity Level 2 (L2):** The content describes practices that are in early pilot use and are demonstrating some successful results.
   - **Maturity Level 3 (L3):** The content describes practices that are in limited use in industry or government organizations, perhaps for a particular market sector.
   - **Maturity Level 4 (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.

2. 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.
   - T: Technical leaders, engineering managers, first-line managers, and supervisors.

**Build Security In: A Key Resource**

Since 2004, the DHS Assurance Program has sponsored development for the BSI Web site, a significant resource used in developing “Software Security Engineering.” BSI content, referenced throughout the book, 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 both 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 articles appearing on the BSI site include senior staff from the SEI and Cigital, Inc.

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, ingenuous, 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 nations, economies, businesses, and families rely on software to sustain and improve the 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; they are the most mature, have a fair level of automated support, and can demonstrate some early successes, providing visible benefits in helping software security efforts gain sup-
port 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 the code and test phase. 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 develop a better understanding of what constitutes secure software—honoring their skills to think like an attacker—applying this mindset 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 the best way to integrate software security practices into the organization's current SDLC.

Also keep in mind that this process, if done properly, will take time. As John Steven stated:

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. [5]

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 SDLC. “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

Note
**Systems Assurance: Preparation and Promise**

**About the Authors**

Julia H. Allen is a senior member of the technical staff within the CERT Program at the SEI. In addition to her work in software security and assurance, Allen conducts research in security governance, operational resilience, and metrics. She is the author of “The CERT Guide to System and Network Security Practices,” and “Governing for Enterprise Security,” and co-hosts discussions for the CERT Podcast Series: “Security for Business Leaders.”

**SEI**
4500 Fifth AVE
Pittsburgh, PA 15213-3890
Phone: (412) 268-7700
E-mail: jha@sei.cmu.edu

Robert J. Ellison, Ph.D., is a member of the Survivable Systems Engineering Team within the CERT Program at the SEI, and has served in a number of technical and management roles. 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. Ellison’s current work explores developing reasoning frameworks to help architects select and refine design tactics to mitigate the impact of a class of cyberattacks.

E-mail: ellison@sei.cmu.edu

Nancy R. Mead, Ph.D., is 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. Mead is a fellow for the IEEE and the IEEE Computer Society and is also a distinguished member of the Association for Computing Machinery. Mead has more than 100 publications and invited presentations.

E-mail: nrm@sei.cmu.edu

Sean Barnum is a principal consultant at Cigital and is technical lead for their federal services practice. He has more than 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 involved in numerous knowledge standards-defining efforts, including Common Weakness Enumeration, Common Attack Pattern Enumeration and Classification, and other elements of software assurance programs for the DHS and DoD. He is also the lead technical subject matter expert for the Air Force Application Software Assurance Center of Excellence.

Cigital, Inc.
21351 Ridgetop CR, STE 400
Dulles, VA 20166
Phone: (703) 473-8262
E-mail: sbarnum@cigital.com

Gary McGraw, Ph.D., is the chief technical officer of Cigital, Inc. He is a globally recognized authority on software security and the author or co-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”; he is also contributing editor for the Addison-Wesley Software Security series. McGraw has written more than 90 peer-reviewed scientific publications, and authors a monthly column for Dark Reading.

Phone: (703) 404-9293
E-mail: gem@cigital.com

**COMING EVENTS**

March 9-12
12th Semi-Annual Software Assurance Forum
McLean, VA
https://buildsecurityin.us-cert.gov/daisy/bsi/events.html

April 26-29
22nd Annual Systems and Software Technology Conference
Salt Lake City, UT
www.sstc-online.org

May 3-7
DISA Customer Partnership Conference 2010/AFCEA Technology Showcase
Nashville, TN
http://events.jspargo.com/da10

May 10-14
PSQT 2010 West
Las Vegas, NV
www.psqtconference.com/2010west

May 24-27
Siemens PLM Connection 2010
Nashville, TN
http://event.plmworld.org

June 6-10
IBM Rational Software Conference
Orlando, FL
http://www-01.ibm.com/software/rational/innovate

June 6-11
Better Software Conference
Las Vegas, NV
www.sqe.com/BetterSoftwareConf

**COMING EVENTS:** Please submit coming events that are of interest to our readers at least 90 days before registration. E-mail announcements to: <marek.steed.ctr@hill.af.mil>