



Software Assurance Curriculum Master Bibliography and Course References

Nancy R. Mead, Software Engineering Institute
Julia H. Allen, Software Engineering Institute
Mark Ardis, Stevens Institute of Technology
Thomas B. Hilburn, Embry-Riddle Aeronautical University
Andrew J. Kornecki, Embry-Riddle Aeronautical University
Richard Linger, Software Engineering Institute

September 2010; Revised June 2011

Copyright 2011 Carnegie Mellon University.

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Internal use. Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use. This document may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Table of Contents

Section 1: Software Assurance Curriculum Master Bibliography.....	2
Section 2: Master of Software Assurance Course Outlines References	32
<i>Assurance Assessment (3.1, 3.2, 3.3, 6.4)</i>	<i>32</i>
<i>Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3)</i>	<i>35</i>
<i>Assured Software Analytics (6.3, 6.4)</i>	<i>37</i>
<i>Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]).....</i>	<i>38</i>
<i>Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])</i>	<i>41</i>
<i>Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]</i>	<i>43</i>
<i>System Operational Assurance (7.1, 7.2, 7.3)</i>	<i>46</i>
<i>System Security Assurance (5.1, 5.2, 5.3)</i>	<i>48</i>

Section 1: Software Assurance Curriculum Master Bibliography

This master bibliography was compiled from *Software Assurance Curriculum Project Volume I: Master of Software Assurance Reference Curriculum* and *Software Assurance Curriculum Project Volume II: Undergraduate Course Outlines*. It also includes references from the *Master of Software Assurance Course Outlines*,¹ which are included in their entirety as Section 2 of this document. URLs are valid as of August 2010.

This bibliography was expanded in June 2011 to include references from *Software Assurance Curriculum Project Volume III: Master of Software Assurance Course Syllabi* and *Software Assurance Curriculum Project IV: Community College Education*.

ABET, Inc. *ABET: Leadership and Quality Assurance in Applied Science, Computing, Engineering, and Technology Education*. <http://www.abet.org/> (2010).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Ahern, Dennis M.; Clouse, Aaron; & Turner, Richard. *CMMI® Distilled: A Practical Introduction to Integrated Process Improvement*. 3rd ed. Addison-Wesley Professional, 2008 (ISBN-10: 0321461088).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*, *Software Assurance Curriculum Project Volume III References*]

[Abstract from publisher:

CMMI® (Capability Maturity Model® Integration) is an integrated, extensible framework for improving process capability and quality across an organization. It has become a cornerstone in the implementation of continuous improvement for both industry and governments around the world. Rich in both detail and guidance for a wide set of organizational domains, the CMMI Product Suite continues to evolve and expand.

Updated for CMMI Version 1.2, this third edition of *CMMI® Distilled* again provides a concise and readable introduction to the model, as well as straightforward, no-nonsense information on integrated, continuous process improvement. The book now also includes practical advice on how to use CMMI in tandem with other approaches, including Six Sigma and Lean, as well as new and expanded guidance on preparing for, managing, and using appraisals.

Written so that readers unfamiliar with model-based process improvement will understand how to get started with CMMI, the book offers insights for those more

¹ This document is available for download at <http://www.cert.org/mswa>.

experienced as well. It can help battle-scarred process improvement veterans, and experienced suppliers and acquirers of both systems and services, perform more effectively. *CMMI® Distilled* is especially appropriate for executives and managers who need to understand why continuous improvement is valuable, why CMMI is a tool of choice, and how to maximize the return on their efforts and investments. Engineers of all kinds (systems, hardware, software, and quality, as well as acquisition personnel and service providers) will find ideas on how to perform better.

The three authors, all involved with CMMI since its inception, bring a wealth of experience and knowledge to this book. They highlight the pitfalls and shortcuts that are all too often learned by costly experience, and they provide a context for understanding why the use of CMMI continues to grow around the world.]

Alberts, Christopher; Dorofee, Audrey J.; Higuera, Ron; Murphy, Richard L.; Walker, Julie A.; & Williams, Ray C. *Continuous Risk Management Guidebook*. Software Engineering Institute, Carnegie Mellon University, 1996.

<http://www.sei.cmu.edu/library/abstracts/books/crmguidebook.cfm>

[Source: *Master of Software Assurance Course Outline References – Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3)*]

Alberts, Christopher; Allen, Julia; & Stoddard, Robert. *Integrated Measurement and Analysis Framework for Software Security* (CMU/SEI-2010-TN-025) (forthcoming). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Software Assurance Curriculum Project Volume III References*]

Alberts, Christopher J. & Dorofee, Audrey J. *Risk Management Framework* (CMU/SEI-2010-TR-071). Carnegie Mellon University, Software Engineering Institute, 2010.

<http://www.sei.cmu.edu/library/abstracts/reports/10tr017.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

Alexander, Ian. "Misuse Cases: Use Cases with Hostile Intent." *IEEE Software* 20, 1 (January-February 2003): 58-66.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Software Assurance Curriculum Project Volume III References*]

[Misuse and abuse cases.]

Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley Professional, 2008.

[Source: *Master of Software Assurance Course Outline References – Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation], Software Assurance Curriculum Project Volume I, II, and III References*]

[Abstract from publisher:

Software that is developed from the beginning with security in mind will resist, tolerate, and recover from attacks more effectively than would otherwise be possible. While there may be no silver bullet for security, there are practices that project managers will find beneficial. With this management guide, you can select from a number of sound practices likely to increase the security and dependability of your software, both during its development and subsequently in its operation.

Software Security Engineering draws extensively on the systematic approach developed for the Build Security In (BSI) Web site. Sponsored by the Department of Homeland Security Software Assurance Program, the BSI site offers a host of tools, guidelines, rules, principles, and other resources to help project managers address security issues in every phase of the software development life cycle (SDLC). The book's expert authors, themselves frequent contributors to the BSI site, represent two well-known resources in the security world: the CERT Program at the Software Engineering Institute (SEI) and Cigital, Inc., a consulting firm specializing in software security.

This book will help you understand why

- Software security is about more than just eliminating vulnerabilities and conducting penetration tests
- Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks
- Software security initiatives should follow a risk-management approach to identify priorities and to define what is “good enough”—understanding that software security risks will change throughout the SDLC
- Project managers and software engineers need to learn to think like an attacker in order to address the range of functions that software should not do, and how software can better resist, tolerate, and recover when under attack]

Altran Group. *Correctness by Construction*. <http://www.altran-praxis.com/cbyc.aspx> (Accessed 2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume III References*]

Anderson, Ross J. *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd ed. Wiley, 2008.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3), Software Assurance Curriculum Project Volume III References*]

[Abstract from publisher:

The world has changed radically since the first edition was published in 2001. Spammers, virus writers, phishermen, money launderers, and spies now trade busily with each other in a lively online criminal economy—and as they specialize, they get better. New applications, from search to social networks to electronic voting machines, provide new targets. And terrorism has changed the world. In this indispensable, fully updated guide, Ross Anderson reveals how to build systems that stay dependable whether faced with error or malice.

Here's straight talk about

- Technical engineering basics—cryptography, protocols, access controls, and distributed systems
- Types of attack—phishing, Web exploits, card fraud, hardware hacks, and electronic warfare
- Specialized protection mechanism—what biometrics, seals, smartcards, alarms, and DRM do, and how they fail
- Security economics—why companies build insecure systems, why it’s tough to manage security projects, and how to cope
- Security psychology—the privacy dilemma, what makes security too hard to use, and why deception will keep increasing
- Policy—why governments waste money on security, why societies are vulnerable to terrorism, and what to do about it]

American Association of Community Colleges (AACC). *Students at Community Colleges*. <http://www.aacc.nche.edu/AboutCC/Trends/Pages/studentsatcommunitycolleges.aspx> (2011).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

Ardis, Mark A. & Ford, Gary. *SEI Report on Graduate Software Engineering Education (1989)* (CMU/SEI-89-TR-021). Software Engineering Institute, Carnegie Mellon University, 1989.
<http://www.sei.cmu.edu/library/abstracts/reports/89tr021.cfm>
 [Source: *Software Assurance Curriculum Project Volumes I and II References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Program Details: Computer Science*. http://www.acmccecc.org/pgm_inventory/programdetail.aspx?pID=38 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Assessment Rubric for Course Learning Outcomes: Computer Science I*. http://www.acmccecc.org/reports/report_assessmentRubric.aspx?cID=43 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Assessment Rubric for Course Learning Outcomes: Computer Science II*. http://www.acmccecc.org/reports/report_assessmentRubric.aspx?cID=47 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Assessment Rubric for Course Learning Outcomes: Computer Science III*. http://www.acmccecc.org/reports/report_assessmentRubric.aspx?cID=98 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Program Details: Software Engineering*. http://acmccecc.org/pgm_inventory/programdetail.aspx?pID=40 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

ACM Committee for Computing Education in Community Colleges (CCECC). *Course Details: Introduction to Software Engineering*. http://acmccecc.org/course_inventory/coursedetail.aspx?cID=113 (2009).
 [Source: *Software Assurance Curriculum Project Volume IV References*]

Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS) Joint Curriculum Task Force. *Computing Curricula 1991*. ACM Press and IEEE Computer Society Press (1991).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Association for Computing Machinery (ACM) Two-Year College Computing Curricula Task Force. *Computing Curricula Guidelines for Associate-Degree Programs: Computing Sciences*. ACM Press (1993).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS) Joint Task Force on Software Engineering Ethics and Professional Practices (SEEPP). [*Software Engineering Code of Ethics and Professional Practice \(Version 5.2\)*](#). ACM & IEEE-CS, 1999. <http://www.acm.org/serving/se/code.htm>

[Source: *Software Assurance Curriculum Project Volume IV References*]

The Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS). *Software Engineering Code of Ethics and Professional Practice (Version 5.2)*. ACM/IEEE-CS joint task force on Software Engineering Ethics and Professional Practices (SEEPP). <http://www.acm.org/about/se-code> (1999).

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3), Software Assurance Curriculum Project Volume I and III References*]

Association for Computing Machinery (ACM) Two-Year College Computing Curricula Task Force. *Computing Curricula 2003: Guidelines for Associate-Degree Curricula in Computer Science*. ACM Press (2003).

[Source: *Software Assurance Curriculum Project Volume IV References*]

The Association for Computing Machinery (ACM); The Association for Information Systems (AIS); & The Institute of Electrical and Electronics Engineers Computer Society (IEEE-CS). “Computing Curricula 2005.” *Computing Curriculum Series*. http://www.computer.org/portal/cms_docs_ieeeecs/ieeeecs/education/cc2001/CC2005-March06Final.pdf (2005).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

The Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS). “Computer Science Curriculum 2008: An Interim Revision of CS 2001.” *Computing Curriculum Series*. <http://www.acm.org/education/curricula/ComputerScience2008.pdf> (2008).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

The Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS). “Computing Curricula: Information Technology Volume.” *Computing Curriculum Series*. http://www.computer.org/portal/cms_docs_ieeeecs/ieeeecs/education/cc2001/IT_draft_curriculum.pdf (2008).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Association for Computing Machinery (ACM). *Computing Curricula 2009: Guidelines for Associate-Degree Transfer Curriculum in Computer Science*. ACM and IEEE Computer Society, 2009.

<http://www.acmccecc.org/committee/CommitteeFileUploads/2009ComputerScienceTransferGuidelines.pdf>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Association for Computing Machinery (ACM) Inc. *ACM Code of Ethics and Professional Conduct*. <http://www.acm.org/constitution/code.html> (2011).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Australian/New Zealand Standard (AS/NZS) & International Organization for Standardization (ISO). *AS/NZS ISO 31000: 2009 Risk Management—Principles and Guidelines*, 1st ed. AS/NZS, November 2009.

[Source: *Software Assurance Curriculum Project Volume III References*]

Bagheri, Hamid & Mirian-Hosseinabadi, Seyed-Hassan. “Injecting Security as Aspectable NFR into Software Architecture,” *Proceedings of the 14th Asia-Pacific Software Engineering Conference*. Nagoya, Japan. Dec. 2007. IEEE Computer Society Press, 2008.

[Source: *Software Assurance Curriculum Project Volume III References*]

Barbacci, Mario R.; Ellison, Robert J.; Lattanze, Anthony J.; Stafford, Judith A.; Weinstock, Charles B.; & Wood, William G. *Quality Attribute Workshops (QAWs)*, 3rd ed. (CMU/SEI-2003-TR-016). Software Engineering Institute, Carnegie Mellon University, 2003.
<http://www.sei.cmu.edu/library/abstracts/reports/03tr016.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

Berg, Clifford J. *High Assurance Design: Architecting Secure and Reliable Enterprise Applications*. Addison-Wesley Professional, 2005.

[Source: *Software Assurance Curriculum Project Volume III References*]

Bishop, Matt. *Computer Security: Art and Science*. Addison-Wesley Professional, 2002.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3), Software Assurance Curriculum Project Volume II and III References*]

Blanchette, Stephen, Jr.; Crosson, Steven; & Boehm, Barry. *Evaluating the Software Design of a Complex System of Systems* (CMU/SEI-2009-TR-023, ESC-TR-2009-023). Software Engineering Institute, Carnegie Mellon University, 2009.

<http://www.sei.cmu.edu/library/abstracts/reports/09tr023.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

Bloom, B. S., ed. *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I, Cognitive Domain*. Longmans, 1956.

[Source: *Software Assurance Curriculum Project Volumes I, II, III, and IV References*]

Bode, Stephan; Fischer, Anja; Kühnhauser, Winfried; & Riebisch, Matthias. “Software Architectural Design Meets Security Engineering,” *Proceedings of the 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*. San Francisco, CA, April 2009. IEEE Computer Society Press, 2009.

[Source: *Software Assurance Curriculum Project Volume III References*]

Boehm, Barry & Turner, Richard. *Balancing Agility and Discipline: A Guide for the Perplexed*. Addison-Wesley Professional, 2003.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Bosworth, Seymour & Kabay, M. E., eds. *Computer Security Handbook*. John Wiley, 2002.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3)*]

Carnegie Mellon University (CMU), College of Humanities and Social Sciences Information Systems (IS) degree program. *Course 67-309: Special Topics: Information Assurance and Security*. <http://www.cmu.edu/information-systems/electives/> (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume II References*]

Carnegie Mellon University (CMU), Electrical & Computer Engineering (ECE) degree program. *Course 18-730: Introduction to Computer Security*. <http://www.ece.cmu.edu/courses/18730> (2010).

[Source: *Software Assurance Curriculum Project Volume II References*]

Carnegie Mellon University (CMU), H. John Heinz III College. Course descriptions for programs in the H. John Heinz III College. <http://www.heinz.cmu.edu/admitted-students-pt/pre-arrival-information-pt/course-descriptions-and-registration/index.aspx> (2010).

[Source: *Software Assurance Curriculum Project Volume II References*]

Carnegie Mellon University (CMU), H. John Heinz III College, School of Information Systems & Management. Courses in the Master of Science in [Information Security Policy & Management \(MSISPM\)](#) program. <http://www.heinz.cmu.edu/school-of-information-systems-and-management/information-security-policy-management-msispm/curriculum/course-information-BAK/index.aspx> (2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Carnegie Mellon University (CMU), H. John Heinz III College, School of Information Systems & Management. Curriculum of the MSISPM program. <http://www.heinz.cmu.edu/school-of-information-systems-and-management/information-security-policy-management-msispm/curriculum/index.aspx> (2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Carnegie Mellon University (CMU), The Information Networking Institute (INI). Courses in the INI. http://www.ini.cmu.edu/degrees/pgh_msistm/courses.html (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Carnegie Mellon University (CMU), The Information Networking Institute. Master of Science in Information Security Technology and Management (MSISTM).

http://www.ini.cmu.edu/degrees/pgh_msistm/index.html (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Carnegie Mellon University (CMU), Master of Information Systems (MISM) degree program. *95-750 Security Architecture & Analysis* course descriptions. <http://www.andrew.cmu.edu/course/95-750> (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volumes II References*]

Carnegie Mellon University (CMU), School of Computer Science. *Course 15-392: Special Topic—Secure Programming*.
<http://www.cs.cmu.edu/afs/cs/usr/cathyf/www/ugcoursedescriptions.htm> (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume II References*]

Carnegie Mellon University (CMU), School of Computer Science (SCS). International Software Research Institute (ISRI) Masters programs. <http://mse.isri.cmu.edu/software-engineering/web1-Programs/index.html> (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Caralli, Richard; Stevens, James F.; Bradford, J. Wilke; & Wilson, William R. *The Critical Success Factor Method: Establishing a Foundation for Enterprise Security Management* (CMU/SEI- 2004-TR-010). Carnegie Mellon University, Software Engineering Institute, July 2004. <http://www.sei.cmu.edu/library/abstracts/reports/04tr010.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

CERT. *Survivability and Information Assurance Curriculum*. Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/sia/> (2007).

[Source: *Software Assurance Curriculum Project Volumes I, II, and IV References*]

CERT. *An Experience-Based Maturity Model for Software Security* (podcast). Software Engineering Institute, Carnegie Mellon University.

<http://www.cert.org/podcast/show/20090331mcgraw.html> (2008).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

CERT. *Identifying Software Security Requirements Early, Not After the Fact* (podcast). Software Engineering Institute, Carnegie Mellon University.

<http://www.cert.org/podcast/show/20080708mead.html> (2008).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

CERT. *Insider Threat and the Software Development Life Cycle* (podcast). Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/podcast/show/20080304cappelli.html> (2008).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*. *Software Assurance Curriculum Project Volume III References*]

CERT. *OCTAVE* (podcasts, methods, and publications). <http://www.cert.org/octave> (2008).

[Source: *Master of Software Assurance Course Outline References – Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3)*]

CERT. *Integrating Privacy Practices into the Software Development Life Cycle* (podcast). Software Engineering Institute, Carnegie Mellon University.

<http://www.cert.org/podcast/show/20091222hood.html> (2009).

[Source: *Software Assurance Curriculum Project Volume III References*]

CERT. *Advanced Incident Handling* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

CERT. *Advanced Information Security for Technical Staff* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

CERT. *The CERT Oracle Secure Coding Standard for Java*.

<https://www.securecoding.cert.org/confluence/display/java/The+CERT+Oracle+Secure+Coding+Standard+for+Java> (2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

CERT. *CERT[®] Resilience Management Model* (reports and presentations).

<http://www.cert.org/resilience/rmm.html> (2010).

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*, *Software Assurance Curriculum Project Volume III References*]

CERT. *Fundamentals of Incident Handling* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

CERT. *Information Security for Technical Staff* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

[CERT's *Information Security for Technical Staff* courseware (introductory, advanced)]

- **Introductory:** This five-day course is designed to provide participants with practical techniques for protecting the security of an organization's information assets and resources, beginning with concepts and proceeding on to technical implementations. The courses focus on understanding and applying the concept of survivability through the effective management of risk, threats, policy, system configuration, availability, and personnel. The course also addresses incident response and provides a technical foundation for working with TCP/IP security and cryptography. The final section of the course helps participants learn to design secure network architecture managing host systems, securing network services, and infrastructure, working with firewalls, and understanding intrusion detection and prevention.
- **Advanced:** This four-day course is designed to increase the depth of knowledge and skills of technical staff charged with administering and securing information systems and networks.]

CERT. *Malware Analysis Apprenticeship* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

CERT. *Managing Enterprise Information Security: A Practical Approach for Achieving Defense-in-Depth* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

[This three-day course begins with a brief review of the conceptual foundations of information security. Next, students will be introduced to the CERT Defense-in-Depth Framework: eight *operationally* focused and inter-dependent management components which will be synergistically applied to a fictitious organization's Information Technology (IT) enterprise. Through lectures, demonstrations, scenario-based exercises, small group activities, and open discussions, students will learn high-level best practices for effectively integrating each of these eight components into all aspects of IT operations. Further, the course scenario is used extensively to reinforce these best practices with technical information security implementations.]

CERT. *Network Situational Awareness (NetSA)* (tool suite). <http://www.cert.org/netsa/> (2010).

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

CERT. *Secure Coding Standards*. Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/secure-coding/scstandards.html> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation], Software Assurance Curriculum Project Volume III References*]

CERT. *SQUARE* (educational materials for download). Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/sse/square.html> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Software Assurance Curriculum Project Volume III References*]

CERT. *Homepage*. <http://www.cert.org/> (2011).

[Source: *Software Assurance Curriculum Project Volume III References*]

Chen, Grace. "The Reverse Transfer Process." *Community College Review*. 2008.

<http://www.communitycollegereview.com/articles/22>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Cifuentes, Cristina. "Improving Software Quality with Parfait." Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009. <https://www.vte.cert.org/vteweb/go/2697.aspx>.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*]

CloudFail.net. *Homepage*. <http://cloudfail.net/> (2011).

[Source: *Software Assurance Curriculum Project Volume III References*]

CMMI Product Team. *CMMI for Acquisition, Version 1.2* (CMU/SEI-2007-TR-017, ESC-TR-2007-017). Software Engineering Institute, Carnegie Mellon University, 2007.

<http://www.sei.cmu.edu/library/abstracts/reports/07tr017.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

CMMI Product Team. *CMMI for Services, Version 1.2* (CMU/SEI-2009-TR-001, ESC-TR-2009-001). Software Engineering Institute, Carnegie Mellon University, 2009.

<http://www.sei.cmu.edu/library/abstracts/reports/09tr001.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

CMMI Product Team. *CMMI for Development, Version 1.3* (CMU/SEI-2010-TR-033). Carnegie Mellon University, Software Engineering Institute, November 2010.

<http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm>

[Source: *Software Assurance Curriculum Project Volume III References*]

CoFI. *CASL from CoFI*. <http://www.informatik.uni-bremen.de/cofi/wiki/index.php/CASL> (2008).

[Source: *Software Assurance Curriculum Project Volume III References*]

Committee on National Security Systems (CNSS). *Instruction No. 4009, National Information Assurance Glossary*. Revised June 2009.

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Committee for Advancing Software-Intensive Systems Producibility; Computer Science and Telecommunications Board; & Division on Engineering and Physical Sciences. *Critical Code: Software Producibility for Defense*. National Academy of Sciences, 2010.

[Source: *Software Assurance Curriculum Project Volume III References*]

Community College Research Center (CCRC). *History/Mission*.

<http://ccrc.tc.columbia.edu/History.asp> (2011).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Community Z Tools. *Overview*. <http://czt.sourceforge.net/> (2007).

[Source: *Software Assurance Curriculum Project Volume III References*]

Cooper, Stephen; Nickell, Christine; Pérez, Lance C.; Oldfield, Brenda; Brynielsson, Joel; Gençer Gökce, Asım; Hawthorne, Elizabeth K.; Klee, Karl J.; Lawrence, Andrea; & Wetzel, Susanne. *Towards Information Assurance (IA) Curricular Guidelines* (ITiCSE 2010 Working Group Report). ACM, 2010. <http://delivery.acm.org/10.1145/1980000/1971686/p49-cooper.pdf?key1=1971686&key2=6295195031&coll=DL&dl=ACM&ip=128.237.28.14&CFID=22435773&CFTOKEN=11620354>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Croll, Paul & Moss, Michele. “Leveraging CMMs and Standards for Assurance.” *Paper presented at the Systems & Software Technology Conference (SSTC) 2008*. Las Vegas, NV, Track 6, April 30, 2008.

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Croll, Paul & Moss, Michele. “Leveraging the CMMI to Address Assurance—Benchmarking Assurance Practices and Managing Assurance Risks.” *Paper presented at the NDIA CMMI Technology Conference 2008*. Denver, CO, November 19, 2008.

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

CyberWatch. *College Curriculum*.

http://www.cyberwatchcenter.org/index.php?option=com_content&view=article&id=99&Itemid=64 (Accessed June 2011). Note: Downloading the curriculum requires registration.

[Source: *Software Assurance Curriculum Project Volume IV References*]

Dark, Melissa; Harter, Nathan; Morales, Linda; & Garcia, Mario A. “An Information Security Ethics Education Model.” *Journal of Computing Sciences in College* 23 6 (June 2008): 82-88.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3), Software Assurance Curriculum Project Volume III References*]

Department of Homeland Security (DHS) Software Assurance (SwA). *Build Security In*. <https://buildsecurityin.us-cert.gov/daisy/adm-bsi/home.html> (2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Department of Homeland Security (DHS). *Build Security In, Architectural Risk Analysis* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/architecture.html> (2008-2009).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])*]

Department of Homeland Security (DHS). *Build Security In, Attack Patterns* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/attack.html> (2008-2009).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])*]

Department of Homeland Security (DHS). *Build Security In, Best Practices*. <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices.html> (2008–2009).

[Source: *Software Assurance Curriculum Project Volume III References*]

Department of Homeland Security (DHS). *Build Security In, Deployment and Operations* (articles). <https://buildsecurityin.us-cert.gov/adm-bsi/articles/best-practices/deployment.html> (2009).

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

Department of Homeland Security (DHS). *Build Security In, IEEE Security & Privacy*. Edited by Gary McGraw. <https://buildsecurityin.us-cert.gov/bsi/resources/414-BSI.html> (2004-2006).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*]

[Relevant articles on testing include “Software Penetration Testing”, “Static Analysis for Security,” and “Software Security Testing.”]

Department of Homeland Security (DHS). *Build Security In, Secure Software Development Life Cycle (SDLC) Process* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/sdlc.html> (2008-2009).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development I - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Software Assurance Curriculum Project Volume III References*]

Department of Homeland Security (DHS). *Security Requirements Engineering* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements.html> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development I - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*], [Source: *Software Assurance Curriculum Project Volume III References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Workforce Education and Training Working Group. *Software Assurance: A Curriculum Guide to the Common Body of Knowledge to Produce, Acquire, and Sustain Secure Software*. Edited by Samuel T. Redwine, Jr. <https://buildsecurityin.us-cert.gov/daisy/bsi/940-BSI/version/default/part/AttachmentData/data/CurriculumGuideToTheCBK.pdf> (2007).

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA). *Processes and Practices Working Group*. <https://buildsecurityin.us-cert.gov/swa/procwg.html> (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA). *Summary of Assurance for CMMI Efforts*. https://buildsecurityin.us-cert.gov/swa/downloads/Assurance_for_CMMI_Pilot_version_March_2009.pdf (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Acquisition Working Group. *Software Assurance in Acquisition: Mitigating Risks to the Enterprise*. https://buildsecurityin.us-cert.gov/swa/downloads/SwA_in_Acquisition_102208.pdf (2008).

[Source: *Software Assurance Curriculum Project Volume III References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Acquisition & Outsourcing Working Group. “Contract Language for Integrating Software Security in the Acquisition Life Cycle.” *Software Assurance Pocket Guide Series: Acquisition & Outsourcing, Volume 1, Version 1.0*. Department of Homeland Security Software Assurance Program, June 2009.

https://buildsecurityin.us-cert.gov/swa/downloads/PGCLMWV10_04AM090604.pdf

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Acquisition & Outsourcing Working Group. “Software Supply Chain Risk Management and Due Diligence.” *Software Assurance Pocket Guide Series: Acquisition & Outsourcing, Volume 2, Version 1.2*. Department of Homeland Security Software Assurance Program, June 2009. https://buildsecurityin.us-cert.gov/swa/downloads/DueDiligenceMWV12_01AM090909.pdf

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Acquisition & Outsourcing Working Group. “Contract Language for Secure Software.” *Software Assurance Pocket Guide Series: Acquisition & Outsourcing, Volume 3, Version 1*. Department of Homeland Security Software Assurance Program, March 2009. https://buildsecurityin.us-cert.gov/swa/downloads/Contract_Language_PocketGuide_Print.pdf

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Development Working Group. “Risk-Based Software Security Testing.” *Software Assurance Pocket Guide Series: Development, Volume 3, Version 0.5*. Department of Homeland Security Software Assurance Program, September 2009. https://buildsecurityin.us-cert.gov/swa/downloads/PG_testing_091013_Cover_n_text.pdf

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Processes & Practices Working Group & Technology, Tools and Product Evaluation Working Group. “Key Practices for Mitigating the Most Egregious Exploitable Software Weaknesses,” Table 1. *Software Assurance Pocket Guide Series: Development, Volume 2, Version 1.3*. Department of Homeland Security Software Assurance Program, March 2009. https://buildsecurityin.us-cert.gov/swa/downloads/KeyPracticesMWV13_02AM091013.pdf

[Source: *Software Assurance Curriculum Project Volume I References*]

Department of Homeland Security (DHS) Software Assurance (SwA) Workforce Education and Training Working Group. *Software Assurance CBK/Principles Organization*. <https://buildsecurityin.us-cert.gov/bsi/dhs/927-BSI.html> (2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Dougherty, Chad; Sayre, Kirk; Seacord, Robert; Svoboda, David; & Togashi, Kazuya. *Secure Design Patterns* (CMU/SEI-2009-TR-101, ESC-TR-2009-010). Carnegie Mellon University, Software Engineering Institute, 2009.

<http://www.sei.cmu.edu/library/abstracts/reports/09tr010.cfm>

[Source: *Software Assurance Curriculum Project Volume I References*]

Dowd, Mark; McDonald, John; & Schuh, Justin. *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*. Addison Wesley, 2007.

[Source: *Software Assurance Curriculum Project Volume III References*]

Drew, Christopher. “Wanted: ‘Cyber Ninjas.’” *New York Times*, December 29, 2009.

<http://www.nytimes.com/2010/01/03/education/edlife/03cybersecurity.html?emc=eta1> (Accessed January 2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Eagle, Chris. *The IDA Pro Book: The Unofficial Guide to the World’s Most Popular Disassembler*. No Starch Press, 2008.

[Source: *Master of Software Assurance Course Outline References – Assured Software Analytics (6.3, 6.4)*, *Software Assurance Curriculum Project Volume III References*]

[This textbook describes a widely used tool that supports automated analysis of software executables. It could be used as a reference textbook for an example software analysis tool.]

Eilam, Eldad. *Reversing: Secrets of Reverse Engineering*. John Wiley, 2005.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3)*, *Assured Software Analytics (6.3, 6.4)*]

[This textbook covers methods for reverse engineering of software.]

Ellis, Heidi J. C.; Demurjian, Steven A.; & Naveda, Fernando J. *Software Engineering: Effective Teaching and Learning Approaches and Practices*. Information Science Reference, 2008.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Ellison, Robert J.; Goodenough, John B.; Weinstock, Charles B.; & Woody, Carol. *Evaluating and Mitigating Software Supply Chain Security Risks* (CMU/SEI-2010-TN-016). Software Engineering Institute, Carnegie Mellon University, 2010.

<http://www.sei.cmu.edu/library/abstracts/reports/10tn016.cfm>

[Source: *Software Assurance Curriculum Project Volumes I and III References*]

Ellison, Robert J. & Woody, Carol. *Considering Software Supply Chain Risks*.

<https://buildsecurityin.us-cert.gov/bsi/resources/1185-BSI/1207-BSI.html> (2010).

[Source: *Software Assurance Curriculum Project Volume III References*]

Embry-Riddle Aeronautical University (ERAU). *Master of Software Engineering Program*.

<http://www.erau.edu/db/degrees/ma-softwareeng.html> (2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Epstein, Jeremy; Matsumoto, Scott; & McGraw. “Software Security and SOA: Danger, Will Robinson!” *IEEE Security & Practice* 4, 1 (January/February 2006): 80–83.

[Source: *Software Assurance Curriculum Project Volume III References*]

ERAU. Department of Computer and Software Engineering. *Courses SE 450 and 451: Software Team Projects I and II*. <http://prescott.erau.edu/degrees/catalog/prescott-catalog-1011.pdf> (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Fitzgerald, John. *Welcome to the VDM Portal*. <http://www.vdmportal.org/twiki/bin/view> (2010).

[Source: *Software Assurance Curriculum Project Volume III References*]

Ford, Gary. *1991 SEI Report on Graduate Software Engineering Education* (CMU/SEI-91-TR-002/ESD-TR-91-002). Software Engineering Institute, Carnegie Mellon University, 1991.

<http://www.sei.cmu.edu/library/abstracts/reports/91tr002.cfm>

[Source: *Software Assurance Curriculum Project Volume I References*]

Garcia, Suzanne & Turner, Richard. *CMMI[®] Survival Guide: Just Enough Process Improvement*. Addison-Wesley Professional, 2006.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*, *Software Assurance Curriculum Project Volume III References*]

Gerhart, Susan; Hogle, Jan; & Crandall, Jedidiah. *How Do Buffer Overflow Attacks Work?*

<http://nsfsecurity.pr.erau.edu/bom/> (2002).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Software Assurance Curriculum Project Volume III References*]

[A nice introduction to buffer overflows and stack-smashing using animation and student exercises.]

Godse, M. & Mulik, S. “An Approach for Selecting Software-as-a-Service (SaaS)” 155-158. *Proceedings of the 2009 IEEE International Conference on Cloud Computing*. Bangalore, India, Sept. 2009. IEEE Computer Society, 2009.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Goertzel, Karen Mercedes; Winograd, Theodore; McKinley, Holly Lynne; Oh, Lyndon; Colon, Michael; McGibbon, Thomas; Fedchak, Elaine; & Viennuea. *Software Security Assurance: State-of-the-Art Report (SOAR)*. Information Assurance Technology Analysis Center (IATAC) & Data and Analysis Center for Software (DACs), 2007.

<http://iac.dtic.mil/iatac/download/security.pdf>

[Source: *Software Assurance Curriculum Project Volume III References*]

Gold, Nicolas; Mohan, Andrew; Knight, Claire; & Munro, Malcolm. “Understanding Service-Oriented Software,” *IEEE Software* 21, 2 (March/April 2004): 71–77.

[Source: *Software Assurance Curriculum Project Volume III References*]

Golze, Andreas; Sarbiewski, Mark; & Zahm, Alain. *Optimize Quality for Business Outcomes: A Practical Approach to Software Testing*. Wiley Publishing, 2008.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation], Software Assurance Curriculum Project Volume III References*]

Gorgone John T.; Davis, Gordon B.; Valacich, Joseph S.; Topi, Heikki; Feinstein, David L.; & Longenecker, Herbert E., Jr. “IS 2002: Model Curriculum and Guidelines for Undergraduate Programs in Information Systems.” *Computing Curriculum Series*. Association for Information Systems. http://192.245.222.212:8009/IS2002Doc/Main_Frame.htm (2002).

[Source: *Software Assurance Curriculum Project Volumes I and III References*]

Graham, Dan. *Introduction to the CLASP Process*. <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/548-BSI.html> (2006).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Grembi, Jason. “Secure Software Development - A Security Programmer’s Guide.” Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009.

<https://www.vte.cert.org/vteweb/go/2699.aspx>

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation], Software Assurance Curriculum Project Volume III References*]

Haley, Charles B; Laney, Robin; Moffett, Jonathan D.; & Nuseibeh, Bashar. Ch. 214, “Arguing Satisfaction of Security Requirements.” *Information Security and Ethics: Concepts, Methodologies, Tools, and Applications*. 6 vols. Idea Group Reference, 2008.

[Source: *Software Assurance Curriculum Project Volume III References*]

Hansson, Jörgen; Wrage, Lutz; Feiller, Peter H.; & Morley, John. “Architectural Modeling to Verify Security and Nonfunctional Behavior.” *IEEE Security & Practice* 8, 1: (Jan./Feb. 2010): 43–49.

[Source: *Software Assurance Curriculum Project Volume III References*]

Holt, Alan & Huang, Chi-Yu, *802.11 Wireless Networks: Security and Analysis*. Springer, 2010.

[Source: *Master of Software Assurance Course Outline References – Assured Software Analytics (6.3, 6.4), Software Assurance Curriculum Project Volume III References*]

[This textbook includes wireless network security analysis and methods. It could be used as a reference that provides a picture of assurance issues in the pervasive wireless networks that support software and service operations across organizations and that are themselves software enabled.]

Howard, Michael & LeBlanc, David. *Writing Secure Code*, 2nd ed. Microsoft Press, 2003.

[Source: *Software Assurance Curriculum Project Volume III References*]

Howard, Michael; LeBlanc, David; & Viega, John. *19 Deadly Sins of Software Security*. McGraw-Hill, 2005.

[Source: *Software Assurance Curriculum Project Volume III References*]

Howard, Michael & Lipner, Steve. *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4 , Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements], Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume II and III References*]

[Abstract from publisher:

Your in-depth, expert guide to the proven process that helps reduce security bugs. Your customers demand and deserve better security and privacy in their software. This book is the first to detail a rigorous, proven methodology that measurably minimizes security bugs—the Security Development Lifecycle (SDL). In this long-awaited book, security experts Michael Howard and Steve Lipner from the Microsoft Security Engineering Team guide you through each stage of the SDL—from education and design to testing and post-release. You get their first-hand insights, best practices, a practical history of the SDL, and lessons to help you implement the SDL in any development organization.

Discover how to:

- Use a streamlined risk-analysis process to find security design issues before code is committed
- Apply secure-coding best practices and a proven testing process
- Conduct a final security review before a product ships
- Arm customers with prescriptive guidance to configure and deploy your product more securely
- Establish a plan to respond to new security vulnerabilities
- Integrate security discipline into agile methods and processes, such as Extreme Programming and Scrum

Includes a CD featuring:

- A six-part security class video conducted by the authors and other Microsoft security experts

Sample SDL documents and fuzz testing tool]Hughes, Katherine. “CTE Dual Enrollment: Preparing Students for College and Careers.” Presented at the *CA Community College Association for Occupational Education Conference*. 2011.

http://ccrc.tc.columbia.edu/DefaultFiles/SendFileToPublic.asp?ft=pdf&FilePath=c:\Websites\ccrc_tc_columbia_edu_documents\332_914.pdf&fid=332_914&aid=47&RID=914&pf=Publication.asp?UID=914

[Source: *Software Assurance Curriculum Project Volume IV References*]

Huitt, W. “The Cognitive System.” *Educational Psychology Interactive*. Valdosta State University, 2006. <http://chiron.valdosta.edu/whuitt/col/cogsys/cogsys.html> (Accessed May 22, 2008).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

IBM. *IBM Point of View: Security and Cloud Computing*.

http://www.ibm.com/common/ssi/fcgin/ssi/alias?infotype=SA&subtype=WH&appname=SWGE_TI_SE_USEN&htmlfid=TIW14045USEN&attachment=TIW14045USEN_HR.PDF (2009).

[Source: *Software Assurance Curriculum Project Volume III References*]

IBM. *Requirements Management and Definition*. <http://www-01.ibm.com/software/rational/offerings/lifecycle/> (2011).

[Source: *Software Assurance Curriculum Project Volume III References*]

IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). *Computing Curricula 2001: Computer Science, Final Report*.

http://www.acm.org/education/curric_vols/cc2001.pdf (2001).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

IEEE Computer Society (IEEE-CS). *Software Engineering Body of Knowledge (SWEBOK)*.

<http://www.computer.org/portal/web/swebok> (2004).

[Source: *Software Assurance Curriculum Project Volume I References*]

IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). “Computer Engineering.” *Computing Curriculum Series*.

http://www.computer.org/portal/cms_docs_ieeeecs/ieeeecs/education/cc2001/CCCE-FinalReport-2004Dec12-Final.pdf (2004).

[Source: *Software Assurance Curriculum Project Volume I References*]

IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). “Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering.” *Computing Curriculum Series*.

http://www.acm.org/education/education/curric_vols/CE-Final-Report.pdf (2004).

[Source: *Software Assurance Curriculum Project Volume I References*]

IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). “Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software

Engineering.” *Computing Curriculum Series*. <http://sites.computer.org/ccse/SE2004Volume.pdf> (2004).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

IEEE Computer Society. *IEEE Computer Society Educational Activities Board*. <http://www.computer.org/education/> (2011).

[Source: *Software Assurance Curriculum Project Volume IV References*]

IEEE Standards Association (IEEE-SA). IEEE Std 1061–1998 *IEEE Standard for a Software Quality Metrics Methodology*. IEEE–SA, 1998.

[Source: *Software Assurance Curriculum Project Volume III References*]

IEEE Standards Association (IEEE–SA) IEEE Std 1219-1998 *IEEE Standard for Software Maintenance*. IEEE–SA, 1998.

[Source: *Software Assurance Curriculum Project Volume III References*]

IEEE Standards Association (IEEE–SA). IEEE Std 1062–1998 *IEEE Recommended Practice for Software Acquisition*. IEEE–SA, 1998.

[Source: *Software Assurance Curriculum Project Volume III References*]

IEEE Standards Association (IEEE-SA). *IEEE Std 1062-1993 IEEE Recommended Practice for Software Acquisition*. IEEE-SA, 2004.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

IEEE Standards Association (IEEE–SA). IEEE Std 15939–2007 *IEEE Systems and Software Engineering—Measurement Process*. IEEE–SA, 2007.

[Source: *Software Assurance Curriculum Project Volume III References*]

Ingalsbe, Jeffrey A.; Kunimatsu, Louis; Baeten, Tim; & Mead, Nancy R. “Threat Modeling: Diving into the Deep End.” *IEEE Software* 25, 1 (January/February 2008).

<https://buildsecurityin.us-cert.gov/bsi/resources/articles/932-BSI.html>.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Software Assurance Curriculum Project Volume III References*]

International Organization for Standardization. *ISO/IEC/IEEE 24765 - Systems and Software Engineering – Vocabulary*. http://www.iso.org/iso/catalogue_detail.htm?csnumber=50518 (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

International Organization for Standardization and International Electrotechnical Commission (ISO/IEC). *ISO/IEC 27001:2005 Information Technology – Security Techniques – Information Security Management Systems – Requirements*. ISO/IEC, 2005.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

International Organization for Standardization and International Electrotechnical Commission (ISO/IEC). *ISO/IEC 27002:2005 Information Technology – Security Techniques – Code of Practice for Information Security Management*. ISO/IEC, 2005.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*, *Software Assurance Curriculum Project Volume III References*]

International Organization for Standardization (ISO). *ISO/IEC 15939:2007 Systems and Software Engineering - Measurement Process*, 2nd ed. ISO, August 2007.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4)*, *Software Assurance Curriculum Project Volume III References*]

International Organization for Standardization (ISO). *ISO/IEC FCD 27005: 2008 Information Technology—Security Techniques—Information Security Risk Management*, 2nd ed. ISO, June 2008.

[Source: *Software Assurance Curriculum Project Volume III References*]

Integrated Software & Systems Engineering Curriculum (iSSEc) Project. *Graduate Software Engineering 2009 (GSWE2009) Curriculum Guidelines for Graduate Degree Programs in Software Engineering, Version 1.0*. Stevens Institute of Technology, 2009.

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

(ISC)². *Certified Secure Software Lifecycle Professional (CSSLP) Candidate Information Bulletin*. <https://www.isc2.org/uploadedFiles/Downloads/CSSLP-CBT-Candidate-Information-Bulletin.pdf> (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

Jacky, Jonathan. *The Way of Z: Practical Programming with Formal Methods*. Cambridge University Press, 1996.

[Source: *Software Assurance Curriculum Project Volume III References*]

James Madison University (JMU). Secure Software Engineering curriculum. Includes independent input from Sam Redwine.

http://www.cs.jmu.edu/SSS/grad_program/curriculum.html (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Joint Task Force Transformation Initiative. *Recommended Security Controls for Federal Information Systems and Organizations* (NIST Special Publication 800-53), Revision 3. National Institute of Standards and Technology, August 2009. Updated May 2010.

http://csrc.nist.gov/publications/nistpubs/800-53-Rev3/sp800-53-rev3-final_updated-errata_05-01-2010.pdf

[Source: *Software Assurance Curriculum Project Volume III References*]

Joint Task Force Transformation Initiative. *Guide for Applying the Risk Management Framework to Federal Information Systems* (NIST Special Publication 800-37), Revision 1. National Institute of Standards and Technology, February 2010. <http://csrc.nist.gov/publications/nistpubs/800-37-rev1/sp800-37-rev1-final.pdf>

[Source: *Software Assurance Curriculum Project Volume III References*]

Kan, Stephen H. *Metrics and Models in Software Quality Engineering*, 2nd ed. Addison-Wesley, 2002.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Software Assurance Curriculum Project Volume III References*]

Kazman, Rick; Klein, Mark; & Clements, Paul. *ATAM: Method for Architecture Evaluation* (CMU/SEI-2000-TR-004). Software Engineering Institute, Carnegie Mellon University, 2000. <http://www.sei.cmu.edu/library/abstracts/reports/00tr004.cfm>

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume III References*]

Killcrece, Georgia. *Incident Management*. <https://buildsecurityin.us-cert.gov/bsi/articles/bestpractices/incident/223-BSI.html> (2005-2008).

[Source: *Software Assurance Curriculum Project Volume III References*]

Leroy, Xavier. “Computer Security from a Programming Language and Static Analysis Perspective,” 1–9. *Proceedings of the 12th European Conference on Programming*. Warsaw, Poland, April 2003. Springer-Verlag, 2003.

[Source: *Software Assurance Curriculum Project Volume III References*]

Leaders in Security. “Building In ... Information Security, Privacy And Assurance.” Paper presented at the Knowledge Transfer Network Paris Information Security Workshop. Paris, France, March 30, 2009.

[Source: *Software Assurance Curriculum Project Volume I References*]

Linger, R.; Mills, H.; & Witt, B. *Structured Programming: Theory and Practice*. Addison-Wesley, 1979.

[Source: *Master of Software Assurance Course Outline References – Assured Software Analytics (6.3, 6.4), Software Assurance Curriculum Project Volume III References*]

[This textbook is out of print but contains material on rigorous methods for structuring and reverse engineering of software to verify functionality. It is intended for faculty use only.]

Manadhata, Pratyusa K.; Kaynar, Dilsun K.; & Wing, Jeannette M. *A Formal Model for A System’s Attack Surface* (CMU-CS-07-144). School of Computer Science, Carnegie Mellon University, 2007.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])*]

Manadhata, Pratyusa K. & Wing, Jeannette M. “An Attack Surface Metric.” *IEEE Transactions on Software Engineering* 36, 4 (forthcoming).

<http://doi.ieeecomputersociety.org/10.1109/TSE.2010.60>

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])*]

Mansourov, Nicolai & Campara, Djenana. *System Assurance: Beyond Detecting Vulnerabilities*. Elsevier, 2011. <http://www.elsevierdirect.com/ISBN/9780123814142/System-Assurance>

[Source: *Software Assurance Curriculum Project Volume III References*]

Massacci, Fabio; Mylopoulos, John; & Zannone, Nicola. “Computer-Aided Support for Secure Tropos.” *Automated Software Engineering* 14, 3 (September 2007): 341–364.

[Source: *Software Assurance Curriculum Project Volume III References*]

McGarry, John; Card, David; Jones, Cheryl; Layman, Beth; Clark, Elizabeth; Dean, Joseph; & Hall, Fred. *Practical Software Measurement: Objectives for Decision Makers*. Addison-Wesley Professional, 2001.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4)*]

McGraw, Gary. *Software Security: Building Security In*. Addison-Wesley Professional, 2006.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3), Software Assurance Curriculum Project Volume II References*]

[Abstract from publisher:

The software security best practices, or touchpoints, described in this book have their basis in good software engineering and involve explicitly pondering security throughout the software development lifecycle. This means knowing and understanding common risks (including implementation bugs and architectural flaws), designing for security, and subjecting all software artifacts to thorough, objective risk analyses and testing. *Software Security* is about putting the touchpoints to work for you. Because you can apply these touchpoints to the software artifacts you already produce as you develop software, you can adopt this book's methods without radically changing the way you work. Inside you'll find detailed explanations of

- Risk management frameworks and processes
- Code review using static analysis tools
- Architectural risk analysis
- Penetration testing
- Security testing
- Abuse case development

In addition to the touchpoints, *Software Security* covers knowledge management, training and awareness, and enterprise-level software security programs.]

McGraw, Gary; Chess, Brian; & Miguez, Sammy. *Building Security In Maturity Model BSIMM v1.0*. <http://www.bsimm2.com/> (2009).

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume I, II, and III References*]

Mead, N. R.; Hough, E.; & Stehney, T. *Security Quality Requirements Engineering (SQUARE) Methodology* (CMU/SEI-2005-TR-009). Software Engineering Institute, Carnegie Mellon University, 2005. <http://www.sei.cmu.edu/library/abstracts/reports/05tr009.cfm>

[Source: *Software Assurance Curriculum Project Volume I References*]

Mead, Nancy. *Requirements Engineering Annotated Bibliography*. <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/231-BSI.html> (2008).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Mead, Nancy R. *The Common Criteria*. <https://buildsecurityin.us-cert.gov/bsi/articles/bestpractices/requirements/239-BSI.html> (2008).

[Source: *Software Assurance Curriculum Project Volume III References*]

Mead, Nancy R. *SQUARE Up Your Security Requirements with SQUARE* (webinar). Software Engineering Institute, Carnegie Mellon University, 2009.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Mead, Nancy R.; Allen, Julia H.; Conklin, W. Arthur; Drommi, Antonio; Harrison, John; Ingalsbe, Jeff; Rainey, James; & Shoemaker, Dan. *Making the Business Case for Software Assurance* (CMU/SEI-2009-SR-001). Software Engineering Institute, Carnegie Mellon University, 2009. <http://www.sei.cmu.edu/library/abstracts/reports/09sr001.cfm>

[Source: *Master of Software Assurance Course Outline References – Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3), Software Assurance Curriculum Project Volume I, II, and III References*]

Mead, Nancy R.; Allen, Julia H.; Ardis, Mark; Hilburn, Thomas B.; Kornecki, Andrew J.; Linger, Rick; & McDonald, James. *Software Assurance Curriculum Project Volume I: Master of Software Assurance Reference Curriculum* (CMU/SEI-2010-TR-005, ESC-TR-2010-005). Software Engineering Institute, Carnegie Mellon University, 2010.

<http://www.sei.cmu.edu/library/abstracts/reports/10tr005.cfm>

[Source: *Software Assurance Curriculum Project Volume, II, and III References*]

Mead, Nancy R.; Hilburn, Thomas B.; & Linger, Rick. *Software Assurance Curriculum Project Volume II: Undergraduate Course Outlines* (CMU/SEI-2010-TR-019, ESC-TR-2010-019). Software Engineering Institute, Carnegie Mellon University, 2010.

<http://www.sei.cmu.edu/library/abstracts/reports/10tr019.cfm>

[Not directly cited but part of the Software Assurance Curriculum Project report series.]

Mead, Nancy R.; Allen, Julia H.; Ardis, Mark; Hilburn, Thomas B.; Kornecki, Andrew J.; & Linger, Rick. *Software Assurance Curriculum Project Volume III: Master of Software Assurance Course Syllabi* (CMU/SEI-2011-TR-013). Software Engineering Institute, Carnegie Mellon University, 2011. <http://www.sei.cmu.edu/library/abstracts/reports/11tr013.cfm>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Meier, J. D.; Mackman, Alex; Wastell, Blaine; Bansode, Prashant; Taylor, Jason; & Araujo, Rudolph. *Security Engineering Explained*. Microsoft, 2005.

[Source: *Master of Software Assurance Course Outline References – System Security Assurance (5.1, 5.2, 5.3)*]

Mell, Peter; Kent, Karen; & Nusbaum, Joseph. *Guide to Malware Incident Prevention and Handling* (NIST Special Publication 800-83). National Institute of Standards and Technology, November 2005. <http://csrc.nist.gov/publications/nistpubs/800-83/SP800-83.pdf>

[Source: *Software Assurance Curriculum Project Volume III References*]

Mellado, Daniel; Fernández-Medina, Eduardo; & Piattini, Mario. “A Comparison of Software Design Security Metrics,” 236–242. *Proceedings of the Fourth European Conference on Software Architecture*. Copenhagen, Denmark, Aug. 2010. ACM, 2010.

[Source: *Software Assurance Curriculum Project Volume III References*]

Merkow, Mark S. & Raghavan, Lakshmikanth. *Secure and Resilient Software Development*. CRC Press, 2010.

[Source: *Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Software Assurance Curriculum Project Volume III References*]

[Abstract from publisher:

Although many software books highlight open problems in secure software development, few provide easily actionable, ground-level solutions. Breaking the mold, *Secure and Resilient Software Development* teaches you how to apply best practices and standards for consistent and secure software development. It details specific quality software development strategies and practices that stress resilience requirements with precise, actionable, and ground-level inputs.

Providing comprehensive coverage, the book illustrates all phases of the secure software development life cycle. It shows developers how to master non-functional requirements including reliability, security, and resilience. The authors provide expert-level guidance through all phases of the process and supply many best practices, principles, testing practices, and design methodologies.]

Microsoft. *Security Development Life Cycle, Version 4.1*. <http://www.microsoft.com/sdl> (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

[The SDL addresses, in separate sections, practices for product software vs. practices for internal line of business (LOB) software. Only those for product software were reviewed.]

Miller, Barton P.; Cooksey, Gregory; & Moore, Fredrick. “An Empirical Study of the Robustness of MacOS Applications Using Random Testing.” *ACM SIGOPS Operating Systems Review* 41, 1 (January 2007): 78-86.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Software Assurance Curriculum Project Volume III References*]

The MITRE Corporation (MITRE). *CAPEC: Common Attack Pattern Enumeration and Classification*. <http://capec.mitre.org/> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2) [specification, design]*, *Software Assurance Curriculum Project Volume III References*]

The MITRE Corporation (MITRE). *Common Weakness Enumeration*. <http://cwe.mitre.org/> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation], Software Assurance Curriculum Project Volume III References*]

Moltz, David. “The New Reverse Transfer.” *Inside Higher Ed*. 2009.

<http://www.insidehighered.com/news/2009/02/18/reverse>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Moss, Michele & Nadworny, Margaret. “Update on the Assurance for CMMI Practices.” Department of Homeland Security (DHS) Software Assurance Working Group, December 4, 2008.

[Source: *Software Assurance Curriculum Project Volume I References*]

Mouratidis, Haralambos & Giorgini, Paolo. *Integrating Security and Software Engineering: Advances and Future Visions*. IGI Global, 2006.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume III References*]

Mouratidis, Haralambos & Jurjens, Jan. “From Goal-Driven Security Requirements Engineering to Secure Design.” *International Journal of Intelligent Systems* 25, 8 (August 2010): 813–840.

[Source: *Software Assurance Curriculum Project Volume III References*]

Myers, Andrew. *PLD’06 Tutorial T1: Enforcing and Expressing Security with Programming Languages*. <http://www.cs.cornell.edu/andru/pldi06-tutorial/06jun-pldi-tutorial.pdf> (2006).

[Source: *Software Assurance Curriculum Project Volume III References*]

National Aeronautics and Space Administration (NASA). *Software Assurance Standard (NASA-STD-8739-8-1)*. NASA, July 2004.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4)*]

National Institute of Standards and Technology (NIST). *SAMATE – Software Assurance Metrics and Tool Evaluation*. http://samate.nist.gov/Main_Page.html (2005).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume III References*]

National Institute of Standards and Technology (NIST). *SP 800-53 A Guide for Assessing the Security Controls in Federal Information Systems*. NIST, 2008.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

National Institute of Standards and Technology (NIST). *SP 800-83 Guide to Malware Incident Prevention and Handling*. NIST, 2005.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

National Institute of Standards and Technology (NIST). *SP 800-115 Technical Guide to Information Security Testing and Assessment*. NIST, 2008.

[Source: *Master of Software Assurance Course Outline References – System Operational Assurance (7.1, 7.2, 7.3)*]

The Open Group. *TOGAF*. <http://www.opengroup.org/togaf/> (2010).

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design]), Software Assurance Curriculum Project Volume III References*]

OpenSAMP Project. *Software Assurance Maturity Model (SAMP) v1.0*.

http://www.owasp.org/index.php/Category:Software_Assurance_Maturity_Model (2009).

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements]), Software Assurance Curriculum Project Volume I and III References*]

U.S. Office of Personnel Management. *Federal Cyber Service: Scholarship for Service*.

<https://www.sfs.opm.gov/> (2010).

[Source: *Software Assurance Curriculum Project Volume I References*]

Partnership for Public Service & Booz Allen Hamilton. *Cyber IN-Security: Strengthening the Federal Cybersecurity Workforce*. Partnership for Public Service.

<http://ourpublicservice.org/OPS/publications/viewcontentdetails.php?id=135> (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

Pollice, Gary. *Ethics and Software Development*.

<http://www.ibm.com/developerworks/rational/library/may06/pollice/index.html> (2006).

[Source: *Software Assurance Curriculum Project Volume III References*]

Pressman, Roger S., *Software Engineering: A Practitioner's Approach*, 6th ed. McGraw Hill, 2009.

[Source: *Software Assurance Curriculum Project Volume III References*]

Purdue University, Purdue Homeland Security Institute, Computer Science Department. *CS42600 Computer Security* (course). <http://www.purdue.edu/discoverypark/phsi/learning/homeland-security.php> (2008).

[Source: *Software Assurance Curriculum Project Volume I References*]

Quinley, John W. & Quinley, Melissa P. *Four-Year Graduates Attending Community Colleges: A New Meaning for the Term "Second Chance."* Community College Research Center, 1988.

http://ccrc.tc.columbia.edu/DefaultFiles/SendFileToPublic.asp?ft=pdf&FilePath=c:\Websites\ccrc_tc_columbia_edu_documents\332_41.pdf&fid=332_41&aid=47&RID=41&pf=Publication.asp?UID=41

[Source: *Software Assurance Curriculum Project Volume IV References*]

Ray, Arnab & Cleaveland, Rance. "A Software Architectural Approach to Security By Design," *Proceedings of the 30th International Computer Software and Applications Conference*. Chicago, Ill, Sept. 2006. IEEE Computer Society Press, 2006.

[Source: *Software Assurance Curriculum Project Volume III References*]

Redwine, Samuel T., Jr. *Secure Software Engineering Education*. https://buildsecurityin.us-cert.gov/swa/downloads/JMU_SSE.pdf (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Rehman, S. & Mustafa, K. "Research on Software Design Level Security Vulnerabilities." *ACM SIGSOFT Software Engineering Notes* 34, 6 (November 2009): 1–5.

[Source: *Software Assurance Curriculum Project Volume III References*]

Robert, P. "Quality Requirements for Software Acquisition," 136. *Proceedings of the Software Engineering Standards Symposium and Forum*. IEEE Computer Society, 1997.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])*]

Ross, Ron; Katzke, Stu; Johnson, Arnold; Swanson, Marianne; & Stoneburner, Gary. *Managing Risk from Information Systems: An Organizational Perspective* (NIST Special Publication 800-39), 2nd draft. National Institute of Standards and Technology, April 2008.

http://www.smartgridinformation.info/pdf/2283_doc_1.pdf

[Source: *Software Assurance Curriculum Project Volume III References*]

The SANS Institute. *Introduction to the SANS Security Policy Project*.

<http://www.sans.org/security-resources/policies/> (2011).

[Source: *Software Assurance Curriculum Project Volume III References*]

Scarfone, Karen; Grance, Tim; & Masone, Kelly. *Computer Security Incident Handling Guide* (NIST Special Publication 800-61), Revision 1. National Institute of Standards and Technology, March 2008. <http://csrc.nist.gov/publications/nistpubs/800-61-rev1/SP800-61rev1.pdf>

[Source: *Software Assurance Curriculum Project Volume III References*]

Schumacher, Markus; Fernandez-Buglioni, Eduardo; Hybertson, Duane; Buschmann, Frank; & Sommerlad, Peter. *Security Patterns: Integrating Security and Systems Engineering*, Wiley Series in Software Design Patterns, 2006.

[Source: *Software Assurance Curriculum Project Volume III References*]

Seacord, Robert C. *Secure Coding in C and C++*. Addison-Wesley, 2005.

<http://www.sei.cmu.edu/library/abstracts/books/0321335724.cfm>

[Source: *Software Assurance Curriculum Project Volumes I, II, and III References*]

Seacord, Robert C. *The CERT C Secure Coding Standard*. Addison-Wesley, 2008.

<http://www.sei.cmu.edu/library/abstracts/books/0321563212.cfm>

[Source: *Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Software Assurance Curriculum Project Volume II References*]

[Abstract from publisher:

Commonly exploited software vulnerabilities are usually caused by avoidable software defects. Having analyzed nearly 18,000 vulnerability reports over the past ten years, the CERT/Coordination Center (CERT/CC) has determined that a relatively small number of root causes account for most of them. This book identifies and explains these causes and shows the steps that can be taken to prevent exploitation. Moreover, this book

encourages programmers to adopt security best practices and develop a security mindset that can help protect software from tomorrow's attacks, not just today's. Drawing on the CERT/CC's reports and conclusions, Robert Seacord systematically identifies the program errors most likely to lead to security breaches, shows how they can be exploited, reviews the potential consequences, and presents secure alternatives.

Coverage includes technical detail on how to

- Improve the overall security of any C/C++ application
- Thwart buffer overflows and stack-smashing attacks that exploit insecure string manipulation logic
- Avoid vulnerabilities and security flaws resulting from the incorrect use of dynamic memory management functions
- Eliminate integer-related problems: integer overflows, sign errors, and truncation errors
- Correctly use formatted output functions without introducing format-string vulnerabilities
- Avoid I/O vulnerabilities, including race conditions

Secure Coding in C and C++ presents hundreds of examples of secure code, insecure code, and exploits, implemented for Windows and Linux. If you're responsible for creating secure C or C++ software---or for keeping it safe---no other book offers you this much detailed, expert assistance.]

Software Assurance Forum for Excellence in Code (SAFECode). *Fundamental Practices for Secure Software Development: A Guide to the Most Effective Secure Development Practices in Use Today*. Edited by Stacy Simpson.

http://www.safecode.org/publications/SAFECode_Dev_Practices1108.pdf (2008).

[Source: *Software Assurance Curriculum Project Volumes I and II References*,
Software Assurance Curriculum Project Volume I and III References]

Software Assurance Forum for Excellence in Code (SAFECode). *Software Assurance: An Overview of Current Industry Best Practices*.

http://www.safecode.org/publications/SAFECode_BestPractices0208.pdf (2008).

[Source: *Software Assurance Curriculum Project Volume III References*]

Software Assurance Forum for Excellence in Code (SAFECode). *The Software Supply Chain Integrity Framework: Defining Risks and Responsibilities for Security Software in the Global Supply Chain*. Edited by Stacy Simpson.

http://www.safecode.org/publications/SAFECode_Supply_Chain0709.pdf (2009).

[Source: *Software Assurance Curriculum Project Volume I References*]

Stallings, W. *Network Security Essentials*, 3rd ed. Prentice Hall, 2007.

[Source: *Software Assurance Curriculum Project Volumes I and II References*]

Stevens, Danielle D. & Levi, Antonia J. *Introduction to Rubrics: An Assessment Tool to Save Grading Time, Convey Effective Feedback, and Promote Student Learning*, Stylus Publishing, Virginia (2004).

[Source: *Software Assurance Curriculum Project Volume IV References*]

Stoneburner, Gary; Hayden, Clark; & Feringa, Alexis. *Engineering Principles for Information Technology Security (A Baseline for Achieving Security)*. National Institute of Standards and Technology (NIST), 2001.

[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4), Software Assurance Curriculum Project Volume III References*]

Stoneburner Gary; Hayden, Clark; & Feringa, Alexis. “Engineering Principles for Information Technology Security (A Baseline for Achieving Security), Revision A (NIST Special Publication 800-27 Rev A).” *Computer Security*. Computer Science Division, Information Technology Laboratory, National Institute of Standards and Technology, 2004.

<http://csrc.nist.gov/publications/nistpubs/800-27A/SP800-27-RevA.pdf>

[Source: *Software Assurance Curriculum Project Volume IV References*]

Svoboda, David & Seacord, Robert. “Producing Secure Programs in C and C++.” Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009. <https://www.vte.cert.org/vteweb/go/2693.aspx>.

[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*]

Swanson, Marianne; Bowen, Pauline; Phillips, Amy Wohl; Gallup, Dean; & Lynes, David. *Contingency Planning Guide for Federal Information Systems* (NIST Special Publication 800-34), Revision 1. National Institute of Standards and Technology, May 2010.

http://csrc.nist.gov/publications/nistpubs/800-34-rev1/sp800-34-rev1_errata-Nov11-2010.pdf

[Source: *Software Assurance Curriculum Project Volume III References*]

SWIFT System. *Swift: making web applications secure by construction*.

<http://www.cs.cornell.edu/jif/swift/> (2010).

[Source: *Software Assurance Curriculum Project Volume III References*]

Teumin, David J. *Industrial Network Security*. The Instrumentation, Systems, and Automation Society (ISA), 2004 (ISBN 1556178743).

[Source: *Software Assurance Curriculum Project Volume I References*]

Thiagarajan, Val. *Information Security Management: BS 7799.2:2002: Audit Check List for SANS*. http://www.sans.org/score/checklists/ISO_17799_checklist.pdf (2003).

[Source: *Software Assurance Curriculum Project Volume III References*]

University of California Davis, Department of Computer Science. Course *ECS 153 Computer Security*. http://www.cs.ucdavis.edu/courses/exp_course_desc/153.html (Accessed August 2010).

[Source: *Software Assurance Curriculum Project Volume II References*]

Viega, John & McGraw, Gary. *Building Secure Software*. Addison-Wesley, 2002.

[Source: *Software Assurance Curriculum Project Volume I References*]

Walton, G.; Linger, R.; & Longstaff, T. “Computational Evaluation of Software Security Attributes,” 1–10. *Proceedings of the 42nd Hawaii International Conference on System Sciences*. Los Alamitos, CA, Jan. 2009. IEEE Computer Society Press, 2009.

[Source: *Software Assurance Curriculum Project Volume III References*]

- Wikipedia. *Fagan Inspection*. http://en.wikipedia.org/wiki/Fagan_inspection (2011).
[Source: *Software Assurance Curriculum Project Volume III References*]
- Wright, Marie A. & Kakalik, John S. *Information Security: Contemporary Cases*. Jones and Bartlett Publishers, Inc., 2007.
[Source: *Master of Software Assurance Course Outline References – Assurance Assessment (3.1, 3.2, 3.3, 6.4)*, *Software Assurance Curriculum Project Volume II References*]
- Wysopal, Chris; Nelson, Lucas; Dai Zovi, Dino; & Dustin, Elfriede. *The Art of Software Security Testing: Identifying Software Security Flaws*. Addison-Wesley Professional, 2006.
[Source: *Master of Software Assurance Course Outline References – Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]*, *Software Assurance Curriculum Project Volume III References*]
- Yasar, Ansar-UI-Haque; Preuveneers, Davy; Berbers, Yolande; & Bhatti, Ghasan. “Best Practices for Software Security: An Overview.” *Proceedings of the 12th IEEE International Multitopic Conference*. Bahria University, Karachi, Sindh, Pakistan, December 2008.
[Source: *Software Assurance Curriculum Project Volumes I and II References*]
- Zannone, Nicola. “The Si* Modeling Framework: Metamodel and Applications.” *International Journal of Software Engineering and Knowledge Engineering* 19, 5 (August 2009): 727–746.
[Source: *Software Assurance Curriculum Project Volume III References*]
- The ZETA System. *Overview*. <http://uebb.cs.tu-berlin.de/zeta/> (2010).
[Source: *Software Assurance Curriculum Project Volume III References*]

Section 2: Master of Software Assurance Course Outlines References

This section includes the annotated references from the *Master of Software Assurance Course Outline References*. All references and annotations also appear in Section 1, Software Assurance Curriculum Master Bibliography.

Assurance Assessment (3.1, 3.2, 3.3, 6.4)

Primary sources:

McGraw, Gary. *Software Security: Building Security In*. Addison-Wesley Professional, 2006.

[Abstract from publisher:

The software security best practices, or touchpoints, described in this book have their basis in good software engineering and involve explicitly pondering security throughout the software development lifecycle. This means knowing and understanding common risks (including implementation bugs and architectural flaws), designing for security, and subjecting all software artifacts to thorough, objective risk analyses and testing. *Software Security* is about putting the touchpoints to work for you. Because you can apply these touchpoints to the software artifacts you already produce as you develop software, you can adopt this book's methods without radically changing the way you work. Inside you'll find detailed explanations of

- Risk management frameworks and processes
- Code review using static analysis tools
- Architectural risk analysis
- Penetration testing
- Security testing
- Abuse case development

In addition to the touchpoints, *Software Security* covers knowledge management, training and awareness, and enterprise-level software security programs.]

Howard, Michael & Lipner, Steve. *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.

[Abstract from publisher:

Your in-depth, expert guide to the proven process that helps reduce security bugs. Your customers demand and deserve better security and privacy in their software. This book is the first to detail a rigorous, proven methodology that measurably minimizes security bugs—the Security Development Lifecycle (SDL). In this long-awaited book, security experts Michael Howard and Steve Lipner from the Microsoft Security Engineering Team guide you through each stage of the SDL—from education and design to testing and post-release. You get their first-hand insights, best practices, a practical history of the SDL, and lessons to help you implement the SDL in any development organization.

Discover how to:

- Use a streamlined risk-analysis process to find security design issues before code is committed
- Apply secure-coding best practices and a proven testing process
- Conduct a final security review before a product ships
- Arm customers with prescriptive guidance to configure and deploy your product more securely
- Establish a plan to respond to new security vulnerabilities
- Integrate security discipline into agile methods and processes, such as Extreme Programming and Scrum

Includes a CD featuring:

- A six-part security class video conducted by the authors and other Microsoft security experts
- Sample SDL documents and fuzz testing tool]

Secondary sources:

Alberts, Christopher; Allen, Julia; & Stoddard, Robert. *Integrated Measurement and Analysis Framework for Software Security* (CMU/SEI-2010-TN-025) (forthcoming). Software Engineering Institute, Carnegie Mellon University, 2010.

Godse, M. & Mulik, S. “An Approach for Selecting Software-as-a-Service (SaaS)” 155-158. *Proceedings of the 2009 IEEE International Conference on Cloud Computing*. Bangalore, India, Sept. 2009. IEEE Computer Society, 2009.

IEEE Standards Association (IEEE-SA). *IEEE Std 1062-1993 IEEE Recommended Practice for Software Acquisition*. IEEE-SA, 2004.

International Organization for Standardization (ISO). *ISO/IEC 15939:2007 Systems and Software Engineering - Measurement Process*, 2nd ed. ISO, August 2007.

Kan, Stephen H. *Metrics and Models in Software Quality Engineering*, 2nd ed. Addison-Wesley, 2002.

McGarry, John; Card, David; Jones, Cheryl; Layman, Beth; Clark, Elizabeth; Dean, Joseph; & Hall, Fred. *Practical Software Measurement: Objectives for Decision Makers*. Addison-Wesley Professional, 2001.

McGraw, Gary; Chess, Brian; & Miguez, Sammy. *Building Security In Maturity Model BSIMM v1.0*. <http://www.bsimm2.com/> (2009).

National Aeronautics and Space Administration (NASA). *Software Assurance Standard* (NASA-STD-8739-8-1). NASA, July 2004.

OpenSAMM Project. *Software Assurance Maturity Model (SAMM) v1.0*. http://www.owasp.org/index.php/Category:Software_Assurance_Maturity_Model (2009).

Robert, P. “Quality Requirements for Software Acquisition,” 136. *Proceedings of the Software Engineering Standards Symposium and Forum*. IEEE Computer Society, 1997.

Stoneburner, Gary; Hayden, Clark; & Feringa, Alexis. *Engineering Principles for Information Technology Security (A Baseline for Achieving Security)*. National Institute of Standards and Technology (NIST), 2001.

Wright, Marie & Kakalik, John. *Information Security: Contemporary Cases*. Jones & Bartlett Publishers, 2006.

Assurance Management (2.1, 2.2, 2.3, 4.1, 4.2, 4.3)

Primary source:

Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. Chapters 1, 7, and 8 in *Software Security Engineering: A Guide for Project Managers*, Addison-Wesley Professional, 2008.

[Abstract from publisher:

Software that is developed from the beginning with security in mind will resist, tolerate, and recover from attacks more effectively than would otherwise be possible. While there may be no silver bullet for security, there are practices that project managers will find beneficial. With this management guide, you can select from a number of sound practices likely to increase the security and dependability of your software, both during its development and subsequently in its operation.

Software Security Engineering draws extensively on the systematic approach developed for the Build Security In (BSI) Web site. Sponsored by the Department of Homeland Security Software Assurance Program, the BSI site offers a host of tools, guidelines, rules, principles, and other resources to help project managers address security issues in every phase of the software development life cycle (SDLC). The book's expert authors, themselves frequent contributors to the BSI site, represent two well-known resources in the security world: the CERT Program at the Software Engineering Institute (SEI) and Cigital, Inc., a consulting firm specializing in software security.

This book will help you understand why

- Software security is about more than just eliminating vulnerabilities and conducting penetration tests
- Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks
- Software security initiatives should follow a risk-management approach to identify priorities and to define what is “good enough”—understanding that software security risks will change throughout the SDLC
- Project managers and software engineers need to learn to think like an attacker in order to address the range of functions that software should not do, and how software can better resist, tolerate, and recover when under attack]

Secondary sources:

Alberts, Christopher; Dorofee, Audrey J.; Higuera, Ron; Murphy, Richard L.; Walker, Julie A.; & Williams, Ray C. *Continuous Risk Management Guidebook*. Software Engineering Institute, Carnegie Mellon University, 1996.

<http://www.sei.cmu.edu/library/abstracts/books/crmguidebook.cfm>

CERT. *OCTAVE* (podcasts, methods, and publications). <http://www.cert.org/octave> (2008).

Howard, Michael & Lipner, Steve. *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.

McGraw, Gary. *Software Security: Building Security In*. Addison-Wesley Professional, 2006.

McGraw, Gary; Chess, Brian; & Miguez, Sammy. *Building Security In Maturity Model BSIMM v1.0*. <http://www.bsimm2.com/> (2009).

Mead, Nancy R.; Allen, Julia H.; Conklin, W. Arthur; Drommi, Antonio; Harrison, John; Ingalsbe, Jeff; Rainey, James; & Shoemaker, Dan. *Making the Business Case for Software Assurance* (CMU/SEI-2009-SR-001). Software Engineering Institute, Carnegie Mellon University, 2009. <http://www.sei.cmu.edu/library/abstracts/reports/09sr001.cfm>

Assured Software Analytics (6.3, 6.4)

Primary sources:

Eilam, Eldad. *Reversing: Secrets of Reverse Engineering*. Wiley, 2005.

[This textbook covers methods for reverse engineering of software.]

Linger, R.; Mills, H.; & Witt, B. *Structured Programming: Theory and Practice*. Addison-Wesley, 1979.

[This textbook is out of print but contains material on rigorous methods for structuring and reverse engineering of software to verify functionality. It is intended for faculty use only.]

Secondary sources:

Eagle, Chris. *The IDA Pro Book: The Unofficial Guide to the World's Most Popular Disassembler*. No Starch Press, 2008.

[This textbook describes a widely used tool that supports automated analysis of software executables. It could be used as a reference textbook for an example software analysis tool.]

Holt, Alan & Huang, Chi-Yu. *802.11 Wireless Networks: Security and Analysis*. Springer, 2010.

[This textbook includes wireless network security analysis and methods. It could be used as a reference that provides a picture of assurance issues in the pervasive wireless networks that support software and service operations across organizations and that are themselves software enabled.]

Assured Software Development 1 - Process and Requirements (1.1, 1.2, 6.1, 6.2 [requirements])

Primary sources:

Ahern, Dennis M.; Clouse, Aaron; & Turner, Richard. *CMMI® Distilled: A Practical Introduction to Integrated Process Improvement*. 3rd ed. Addison-Wesley Professional, 2008.

[Abstract from publisher:

CMMI® (Capability Maturity Model® Integration) is an integrated, extensible framework for improving process capability and quality across an organization. It has become a cornerstone in the implementation of continuous improvement for both industry and governments around the world. Rich in both detail and guidance for a wide set of organizational domains, the CMMI Product Suite continues to evolve and expand.

Updated for CMMI Version 1.2, this third edition of *CMMI® Distilled* again provides a concise and readable introduction to the model, as well as straightforward, no-nonsense information on integrated, continuous process improvement. The book now also includes practical advice on how to use CMMI in tandem with other approaches, including Six Sigma and Lean, as well as new and expanded guidance on preparing for, managing, and using appraisals.

Written so that readers unfamiliar with model-based process improvement will understand how to get started with CMMI, the book offers insights for those more experienced as well. It can help battle-scarred process improvement veterans, and experienced suppliers and acquirers of both systems and services, perform more effectively. *CMMI® Distilled* is especially appropriate for executives and managers who need to understand why continuous improvement is valuable, why CMMI is a tool of choice, and how to maximize the return on their efforts and investments. Engineers of all kinds (systems, hardware, software, and quality, as well as acquisition personnel and service providers) will find ideas on how to perform better.

The three authors, all involved with CMMI since its inception, bring a wealth of experience and knowledge to this book. They highlight the pitfalls and shortcuts that are all too often learned by costly experience, and they provide a context for understanding why the use of CMMI continues to grow around the world.]

Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. Chapters 2 and 3 in *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley Professional, 2008.

[Abstract from publisher:

Software that is developed from the beginning with security in mind will resist, tolerate, and recover from attacks more effectively than would otherwise be possible. While there may be no silver bullet for security, there are practices that project managers will find beneficial. With this management guide, you can select from a number of sound practices likely to increase the security and dependability of your software, both during its development and subsequently in its operation.

Software Security Engineering draws extensively on the systematic approach developed for the Build Security In (BSI) Web site. Sponsored by the Department of Homeland Security Software Assurance Program, the BSI site offers a host of tools, guidelines, rules, principles, and other resources to help project managers address security issues in every phase of the software development life cycle (SDLC). The book's expert authors, themselves frequent contributors to the BSI site, represent two well-known resources in the security world: the CERT Program at the Software Engineering Institute (SEI) and Cigital, Inc., a consulting firm specializing in software security.

This book will help you understand why

- Software security is about more than just eliminating vulnerabilities and conducting penetration tests
- Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks
- Software security initiatives should follow a risk-management approach to identify priorities and to define what is “good enough”—understanding that software security risks will change throughout the SDLC
- Project managers and software engineers need to learn to think like an attacker in order to address the range of functions that software should not do, and how software can better resist, tolerate, and recover when under attack]

Secondary sources:

Alexander, I. “Misuse Cases: Use Cases with Hostile Intent.” *IEEE Software* 20, 1 (January-February 2003): 58-66.

[Misuse & abuse cases]

Boehm, Barry & Turner, Richard. *Balancing Agility and Discipline: A Guide for the Perplexed*. Addison-Wesley Professional, 2003.

CERT. *An Experience-Based Maturity Model for Software Security* (podcast). Software Engineering Institute, Carnegie Mellon University.

<http://www.cert.org/podcast/show/20090331mcgraw.html> (2008).

CERT. *Identifying Software Security Requirements Early, Not After the Fact* (podcast). Software Engineering Institute, Carnegie Mellon University.

<http://www.cert.org/podcast/show/20080708mead.html> (2008).

CERT. *Insider Threat and the Software Development Life Cycle* (podcast). Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/podcast/show/20080304cappelli.html> (2008).

CERT. *SQUARE* (educational materials for download). Software Engineering Institute, Carnegie Mellon University. <http://www.cert.org/sse/square.html> (2010).

Department of Homeland Security (DHS). *Build Security In, Secure Software Development Life Cycle (SDLC) Process* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/sdlc.html> (2008-2009).

- Department of Homeland Security (DHS). *Security Requirements Engineering* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements.html> (2010).
- Ellis, Heidi J. C.; Demurjian, Steven A.; & Naveda, Fernando J. *Software Engineering: Effective Teaching and Learning Approaches and Practices*. Information Science Reference, 2008.
- Garcia, Suzanne & Turner, Richard. *CMMI® Survival Guide: Just Enough Process Improvement*. Addison-Wesley Professional, 2006.
- Godse, M. & Mulik, S. “An Approach for Selecting Software-as-a-Service (SaaS)” 155-158. *Proceedings of the 2009 IEEE International Conference on Cloud Computing*. Bangalore, India, Sept. 2009. IEEE Computer Society, 2009.
- Graham, Dan. *Introduction to the CLASP Process*. <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/548-BSI.html> (2006).
- Howard, Michael & Lipner, Steve. *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.
- IEEE Standards Association (IEEE-SA). *IEEE Std 1062-1993 IEEE Recommended Practice for Software Acquisition*. IEEE-SA, 2004.
- Ingalsbe, Jeffrey A.; Kunimatsu, Louis; Baeten, Tim; & Mead, Nancy R. “Threat Modeling: Diving into the Deep End.” *IEEE Software* 25, 1 (January/February 2008). <https://buildsecurityin.us-cert.gov/bsi/resources/articles/932-BSI.html>.
- McGraw, Gary; Chess, Brian; & Miguez, Sammy. *Building Security In Maturity Model BSIMM v1.0*. <http://www.bsimm2.com/> (2009).
- Mead, Nancy. *Requirements Engineering Annotated Bibliography*. <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/231-BSI.html> (2008).
- Mead, Nancy R. *SQUARE Up Your Security Requirements with SQUARE* (webinar). Software Engineering Institute, Carnegie Mellon University. <http://www.sei.cmu.edu/library/abstracts/webinars/14may2009.cfm> (2009).
- Mouratidis, Haralambos & Giorgini, Paolo. *Integrating Security and Software Engineering: Advances and Future Visions*. IGI Global, 2006.
- OpenSAMM Project. *Software Assurance Maturity Model (SAMM) v1.0*. http://www.owasp.org/index.php/Category:Software_Assurance_Maturity_Model (2009).
- Robert, P. “Quality Requirements for Software Acquisition,” 136. *Proceedings of the Software Engineering Standards Symposium and Forum*. IEEE Computer Society, 1997.

Assured Software Development 2 - Architecture and Design (6.1, 6.2 [specification, design])

Primary sources:

Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. Ch. 4, “Secure Software Architecture and Design,” 115-150. *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley Professional, 2008.

[Abstract from publisher:

Software that is developed from the beginning with security in mind will resist, tolerate, and recover from attacks more effectively than would otherwise be possible. While there may be no silver bullet for security, there are practices that project managers will find beneficial. With this management guide, you can select from a number of sound practices likely to increase the security and dependability of your software, both during its development and subsequently in its operation.

Software Security Engineering draws extensively on the systematic approach developed for the Build Security In (BSI) Web site. Sponsored by the Department of Homeland Security Software Assurance Program, the BSI site offers a host of tools, guidelines, rules, principles, and other resources to help project managers address security issues in every phase of the software development life cycle (SDLC). The book’s expert authors, themselves frequent contributors to the BSI site, represent two well-known resources in the security world: the CERT Program at the Software Engineering Institute (SEI) and Cigital, Inc., a consulting firm specializing in software security.

This book will help you understand why

- Software security is about more than just eliminating vulnerabilities and conducting penetration tests
- Network security mechanisms and IT infrastructure security services do not sufficiently protect application software from security risks
- Software security initiatives should follow a risk-management approach to identify priorities and to define what is “good enough”—understanding that software security risks will change throughout the SDLC
- Project managers and software engineers need to learn to think like an attacker in order to address the range of functions that software should not do, and how software can better resist, tolerate, and recover when under attack]

The Open Group. *TOGAF*. <http://www.opengroup.org/togaf/> (2010).

[A comprehensive architecture framework and methodology which enables the design, evaluation and implementation of the right architecture for an enterprise.]

Secondary sources:

Altran Group. *Correctness by Construction*. <http://www.altran-praxis.com/cbyc.aspx> (Accessed 2010).

Department of Homeland Security (DHS). *Building Security In, Architectural Risk Analysis* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/architecture.html> (2008-2009).

Department of Homeland Security (DHS). *Building Security In, Attack Patterns* (articles). <https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/attack.html> (2008-2009).

Howard, Michael & Lipner, Steve. *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.

Kazman, Rick; Klein, Mark; & Clements, Paul. *ATAM: Method for Architecture Evaluation* (CMU/SEI-2000-TR-004). Software Engineering Institute, Carnegie Mellon University, 2000. <http://www.sei.cmu.edu/library/abstracts/reports/00tr004.cfm>

Manadhata, Pratyusa K. & Wing, Jeannette M. "An Attack Surface Metric." *IEEE Transactions on Software Engineering* 36, 4 (forthcoming). <http://doi.ieeecomputersociety.org/10.1109/TSE.2010.60>

Manadhata, Pratyusa K.; Kaynar, Dilsun K.; & Wing, Jeannette M. *A Formal Model for A System's Attack Surface* (CMU-CS-07-144). School of Computer Science, Carnegie Mellon University, 2007.

McGraw, Gary; Chess, Brian; & Miguez, Sammy. *The Building Security In Maturity Model*. <http://www.bsimm2.com/> (2009).

The MITRE Corporation (MITRE). *CAPEC: Common Attack Pattern Enumeration and Classification*. <http://capec.mitre.org/> (2010).

Mouratidis, Haralambos & Giorgini, Paolo. *Integrating Security and Software Engineering: Advances and Future Vision*. IGI Global, 2006.

National Institute of Standards and Technology (NIST). *SAMATE – Software Assurance Metrics and Tool Evaluation*. http://samate.nist.gov/Main_Page.html (2005).

Assured Software Development 3 (6.2) [low-level design, code, test, verification, validation]

Primary sources:

Seacord, Robert C. *Secure Coding in C and C++*. Addison-Wesley, 2005.

<http://www.sei.cmu.edu/library/abstracts/books/0321335724.cfm>

[Abstract from publisher:

Commonly exploited software vulnerabilities are usually caused by avoidable software defects. Having analyzed nearly 18,000 vulnerability reports over the past ten years, the CERT/Coordination Center (CERT/CC) has determined that a relatively small number of root causes account for most of them. This book identifies and explains these causes and shows the steps that can be taken to prevent exploitation. Moreover, this book encourages programmers to adopt security best practices and develop a security mindset that can help protect software from tomorrow's attacks, not just today's.

Drawing on the CERT/CC's reports and conclusions, Robert Seacord systematically identifies the program errors most likely to lead to security breaches, shows how they can be exploited, reviews the potential consequences, and presents secure alternatives.

Coverage includes technical detail on how to

- Improve the overall security of any C/C++ application
- Thwart buffer overflows and stack-smashing attacks that exploit insecure string manipulation logic
- Avoid vulnerabilities and security flaws resulting from the incorrect use of dynamic memory management functions
- Eliminate integer-related problems: integer overflows, sign errors, and truncation errors
- Correctly use formatted output functions without introducing format-string vulnerabilities
- Avoid I/O vulnerabilities, including race conditions

Secure Coding in C and C++ presents hundreds of examples of secure code, insecure code, and exploits, implemented for Windows and Linux. If you're responsible for creating secure C or C++ software---or for keeping it safe---no other book offers you this much detailed, expert assistance.]

Merkow, Mark S. & Raghavan, Lakshmikanth. *Secure and Resilient Software Development*. CRC Press, 2010.

[Abstract from publisher:

Although many software books highlight open problems in secure software development, few provide easily actionable, ground-level solutions. Breaking the mold, *Secure and Resilient Software Development* teaches you how to apply best practices and standards for consistent and secure software development. It details specific quality software development strategies and practices that stress resilience requirements with precise, actionable, and ground-level inputs.

Providing comprehensive coverage, the book illustrates all phases of the secure software development life cycle. It shows developers how to master non-functional

requirements including reliability, security, and resilience. The authors provide expert-level guidance through all phases of the process and supply many best practices, principles, testing practices, and design methodologies.]

Secondary sources:

Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. *Software Security Engineering: A Guide for Project Managers*. Addison-Wesley Professional, 2008.
[Chapter 5 is about coding and testing.]

CERT. *CERT Secure Coding Standards*. <https://www.securecoding.cert.org/> (1995-2009).
[Includes advice for Java as well as C and C++.]

Cifuentes, Cristina. “Improving Software Quality with Parfait.” Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009. <https://www.vte.cert.org/vteweb/go/2697.aspx>.

Department of Homeland Security (DHS). *Build Security In, IEEE Security & Privacy*. Series edited by Gary McGraw. <https://buildsecurityin.us-cert.gov/bsi/resources/414-BSI.html> (2004-2006).

[Relevant articles on testing include “Software Penetration Testing,” “Static Analysis for Security,” and “Software Security Testing.”]

Gerhart, Susan; Hogle, Jan; & Crandall, Jedidiah. *How Do Buffer Overflow Attacks Work?* <http://nsfsecurity.pr.erau.edu/bom/> (2002).

[A nice introduction to buffer overflows and stack-smashing using animation and student exercises.]

Grembi, Jason. “Secure Software Development - A Security Programmer’s Guide.” Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009.
<https://www.vte.cert.org/vteweb/go/2699.aspx>

Golze, Andreas; Sarbiewski, Mark; & Zahm, Alain. *Optimize Quality for Business Outcomes: A Practical Approach to Software Testing*. Wiley Publishing, 2008.
[Especially Chapter 8 on Security Testing.]

Miller, Barton P.; Cooksey, Gregory; & Moore, Fredrick. “An Empirical Study of the Robustness of MacOS Applications Using Random Testing.” *ACM SIGOPS Operating Systems Review* 41, 1 (January 2007): 78-86.

[Describes results of fuzz testing many common applications, including a review of earlier testing of Unix and Windows systems.]

The MITRE Corporation (MITRE). *Common Weakness Enumeration*. <http://cwe.mitre.org/> (2010).

Svoboda, David & Seacord, Robert. “Producing Secure Programs in C and C++.” Tutorial at *11th Semi-Annual Software Assurance Forum*. Arlington, VA, November 2009. Software Engineering Institute, Carnegie Mellon University, 2009. <https://www.vte.cert.org/vteweb/go/2693.aspx>.

Wysopal, Chris; Nelson, Lucas; Dai Zovi, Dino; & Dustin, Elfriede. *The Art of Software Security Testing: Identifying Software Security Flaws*. Addison-Wesley Professional, 2006.

System Operational Assurance (7.1, 7.2, 7.3)

Primary sources:

CERT. Advanced *Information Security for Technical Staff* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

CERT. *Information Security for Technical Staff* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[CERT's *Information Security for Technical Staff* courseware (introductory, advanced)

- **Introductory:** This five-day course is designed to provide participants with practical techniques for protecting the security of an organization's information assets and resources, beginning with concepts and proceeding on to technical implementations. The courses focus on understanding and applying the concept of survivability through the effective management of risk, threats, policy, system configuration, availability, and personnel. The course also addresses incident response and provides a technical foundation for working with TCP/IP security and cryptography. The final section of the course helps participants learn to design secure network architecture managing host systems, securing network services, and infrastructure, working with firewalls, and understanding intrusion detection and prevention.
- **Advanced:** This four-day course is designed to increase the depth of knowledge and skills of technical staff charged with administering and securing information systems and networks.]

CERT. *Managing Enterprise Information Security: A Practical Approach for Achieving Defense-in-Depth* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

[This three-day course begins with a brief review of the conceptual foundations of information security. Next, students will be introduced to the CERT Defense-in-Depth Framework: eight operationally focused and inter-dependent management components which will be synergistically applied to a fictitious organization's Information Technology (IT) enterprise. Through lectures, demonstrations, scenario-based exercises, small group activities, and open discussions, students will learn high-level best practices for effectively integrating each of these eight components into all aspects of IT operations. Further, the course scenario is used extensively to reinforce these best practices with technical information security implementations.]

Secondary sources:

CERT. *Advanced Incident Handling* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

CERT. *CERT[®] Resilience Management Model* (reports and presentations).
<http://www.cert.org/resilience/rmm.html> (2010).

CERT. *Fundamentals of Incident Handling* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

CERT. *Malware Analysis Apprenticeship* (courseware). Software Engineering Institute, Carnegie Mellon University, 2010.

CERT. *Network Situational Awareness (NetSA)* (tool suite). <http://www.cert.org/netsa/> (2010).

Department of Homeland Security (DHS). *Build Security In, Deployment and Operations*. <https://buildsecurityin.us-cert.gov/adm-bis/articles/best-practices/deployment.html> (2009).

International Organization for Standardization and International Electrotechnical Commission (ISO/IEC). *ISO/IEC 27001:2005 Information Technology – Security Techniques – Information Security Management Systems – Requirements*. ISO/IEC, 2005.

International Organization for Standardization and International Electrotechnical Commission (ISO/IEC). *ISO/IEC 27002:2005 Information Technology – Security Techniques – Code of Practice for Information Security Management*. ISO/IEC, 2005.

National Institute of Standards and Technology (NIST). *SP 800-53 A Guide for Assessing the Security Controls in Federal Information Systems*. NIST, 2008.

National Institute of Standards and Technology (NIST). *SP 800-83 Guide to Malware Incident Prevention and Handling*. NIST, 2005.

National Institute of Standards and Technology (NIST). *SP 800-115 Technical Guide to Information Security Testing and Assessment*. NIST, 2008.

System Security Assurance (5.1, 5.2, 5.3)

Primary source:

Anderson, Ross J. *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd ed. Wiley, 2008.

[Abstract from publisher:

The world has changed radically since the first edition was published in 2001. Spammers, virus writers, phishermen, money launderers, and spies now trade busily with each other in a lively online criminal economy -- and as they specialize, they get better. New applications, from search to social networks to electronic voting machines, provide new targets. And terrorism has changed the world. In this indispensable, fully updated guide, Ross Anderson reveals how to build systems that stay dependable whether faced with error or malice.

Here's straight talk about

- Technical engineering basics—cryptography, protocols, access controls, and distributed systems
- Types of attack—phishing, Web exploits, card fraud, hardware hacks, and electronic warfare
- Specialized protection mechanism—what biometrics, seals, smartcards, alarms, and DRM do, and how they fail
- Security economics—why companies build insecure systems, why it's tough to manage security projects, and how to cope
- Security psychology—the privacy dilemma, what makes security too hard to use, and why deception will keep increasing
- Policy—why governments waste money on security, why societies are vulnerable to terrorism, and what to do about it]

Secondary sources:

The Association for Computing Machinery (ACM) & IEEE Computer Society (IEEE-CS) Joint Task Force on Software Engineering Ethics and Professional Practices (SEEPP). [*Software Engineering Code of Ethics and Professional Practice \(Version 5.2\)*](#). ACM & IEEE-CS, 1999.

Bishop, Matt. *Computer Security: Art and Science*. Addison-Wesley, 2003.

Bosworth, Seymour & Kabay, M. E., eds. *Computer Security Handbook*. John Wiley, 2002.

Dark, Melissa; Harter, Nathan; Morales, Linda; & Garcia, Mario A. "An Information Security Ethics Education Model." *Journal of Computing Sciences in College* 23, 6 (June 2008): 82-88.

Eilam, Eldad. *Reversing: Secrets of Reverse Engineering*. John Wiley, 2005.

Meier, J. D.; Mackman, Alex; Wastell, Blaine; Bansode, Prashant; Taylor, Jason; & Araujo, Rudolph. *Security Engineering Explained*. Microsoft, 2005.