Review of the book

"Secure and Resilient Software Development"

by Mark S. Merkow and Lakshmikanth Raghavan

CRC Press, Taylor & Francis Group, 2010

ISBN: 978-1-4398-2696-6

Dr. Emin İslam Tatlı
Daimler TSS

July 2011

1 Summary of the review

This book is a "must read" resource for security experts focusing on application security and for application designers and developers who need to integrate security into their systems. It provides various aspects of application security for each phase of software development. Besides, the book is an OWASP[1]-oriented book and covers many of OWASP's important projects.

Reading the book, you will learn about

- all non-functional requirements of a system and understand why writing secure code is vital – with real-life failed examples
- all aspects of secure design and development of applications with best practices
- secure SDLC (software development lifecycle) methodology by looking into the best solution approaches (e.g. CLASP, openSAMM, BSIMM)
- the common web application risks from OWASP Top 10 projects and their countermeasures
- fixing the common security risks by using secure coding enterprise APIs
- various aspects of application testing – pentests, code review, vulnerability scanning etc.
- worldwide security communities, their projects and activities for career development

In this review, I will summarize the chapters, provide positive and negative aspects of the book and define the target reader group from my perspective.

2 Summary of the book

The book is divided into 12 chapters. The following will summarize the content of each chapter and give an overview of the book.

- **Chapter 1 - How does software fail thee? Let us count the ways:** Security is a non-functional requirement. Hence, it is often ignored within software development processes. It is vital to understand the importance of secure applications and the risks of insecure programs. In this chapter, the authors explain the need for secure software by giving real-life examples of failed software due to security vulnerabilities.

[1] Open Web Application Security Project.

- **Chapter 2 - Characteristics of Secure and Resilient Software:** The main topic of this chapter is the non-functional requirements. The authors introduce the details of 15 non-functional requirements (e.g. efficiency, extensibility, reliability, scalability) including security which are important for requirements and design phases in development process. Each non-functional requirement is described in details and why this requirement is needed is explained as well.

- **Chapter 3 - Security and Resilience in the Software Development Life Cycle:** This chapter is an introduction to secure SDLC (software development lifecycle) methodology. It is firstly explained why security should be integrated within development process. Various security activities are assigned to each development phase (i.e. requirements, design, development, test, deployment) and each security activity (e.g. mapping security&privacy requirements, threat modeling, systems design review, static analysis, testing, security training) is explained in details.

- **Chapter 4 - Proven best practices for resilient applications:** This chapter explains 15 best design practices for secure and resilient software. Some of the best practices are "apply defence in depth", "fail securely", "run with least privilege", "avoid security by obscrutiy" and "keep security simple".

- **Chapter 5 - Designing applications for security and resilience:** This chapter focuses on secure design of applications. Misuse case modeling, threat modeling, risk analysis, design patterns at architectural-level, design-level and implementation level are the main concerns in this chapter. Additionally, the chapter is concluded with an architecture and design review checklist containing requirements regarding authentication, authorization, configuration management, sensitive data, session management, cryptography, parameter manipulation, exception management, auditing and logging.

- **Chapter 6 - Programming best practices:** OWASP Top 10 projects provide a list of 10 common web application security risks. The list is well accepted within the worldwide security community and is used as a reference list for overall security projects. The authors focus on OWASP Top 10 in this chapter. They explain each risk in details and provide very detailed countermeasures (e.g. input validation, parameterized queries, output escaping, canonicalization, etc.) to mitigate the risks. OWASP ESAPI (Enterprise Security API) which provides a broad set of security control APIs for enterprise applications is introduced in this chapter as well. The relevant interfaces of ESAPI are given within the countermeasure section of each OWASP Top 10 risk. The chapter is concluded with two useful resources. The first one is a list of "Top 10 Secure Coding Practices". The second one provides a check list of "50 Questions to Improve Software Security".

- **Chapter 7 - Special considerations for embedded systems, cloud computing and mobile computing devices:** This chapter focuses on special security requirements and limitations of embedded devices and especially mobile devices. The security risks of cloud computing and cloud applications are introduced additionally. Security features of REST (Representational State Transfer) architecture are presented. Finally, mobile computing security features of different mobile device platforms (i.e. BlackBerry, Windows Mobile, IPhone) are explained.

- **Chapter 8 - Security testing of custom software applications:** Testing is the focus of this chapter. Unit testing, manual source code review, automated vulnerability scanning and penetration testing are explained with their pros and cons. In addition, various commercial (e.g. fortify 360) and free tools (e.g. O2-OunceOpen) are presented.

- **Chapter 9 - Testing commercial off-the-shelf systems:** When you finish reading this chapter, you would learn all details of Security Product Evaluation and Certification with the Common Criteria.

- **Chapter 10 - Implementing security and resilience using CLASP:** The next two chapters are about secure SDLC methodology. In this chapter, CLASP (Comprehensive, Lightweight Application Security Process) which is an OWASP project is introduced. CLASP views, activities, roles and best practices are explained in details.

- **Chapter 11 - Metrics and Models for Security and Resilience Maturity:** OpenSAMM (Security Assurance Maturity Model) and BSIMM (Building Security in Maturity Model) are the main topics of this chapter. These secure SDLC methodologies provide very well-structured security activities and practices to integrate within development processes.
- **Chapter 12 - Taking it to the Streets:** This chapter concludes the book and focuses on further career development within application security area. The SANS's training courses are listed, CSSLP certification is explained, WASC (Web Application Security Consortium), OWASP and their projects are introduced and finally current and potential research activities of the U.S. Department of Homeland Security are explained in details.

In addition, the book contains two appendix sections:

- **Appendix A - 2010 CWE/SANS Top 25 Most Dangerous Programming Errors:** This appendix lists the top 25 most dangerous programming errors (e.g. failure to preserve web page structure, buffer copy without checking size of input, improper access control, etc.).
- **Appendix B - ESAPI (Enterprise Security API):** This appendix introduces the ESAPI architecture and its programming interfaces. ESAPI is an OWASP project as well.

3 What is the book like (style)?

The authors have practical experience in application security and wrote a practice-oriented book. The chapters are well-structured and arranged in a good logical order. This enables the readers to follow the content easily. The book is enriched with check lists which can be used as references during real-life projects. Commercial and free tools listed in the testing chapter can be exploited within real-life projects as well. Additionally, a reference section at the end of each chapter and a glossary of security terms at the end of the book are included.

Though this book is nearly perfectly organized and written, I was not very content with the section about mobile computing and embedded devices (Chapter 7). The chapter is neither well-structured nor complete. Different mobile computing issues were put together and only discussed briefly. The transition from one topic to another one can not be followed easily. Within the mobile computing platform, the new trend Android is even not mentioned.

From my point of view, putting all these together, the book can be a reference for any kind of real-life development project and can support project members while accomplishing the required security activities.

4 Would you recommend this book?

I would strongly recommend this book to security architects, security trainers, application designers and developers. They can benefit from this book by gaining information about various design and development security aspects that are part of a software development process. Security architects can exploit best practice design patterns while security trainers can take advantage of secure coding guidelines. Similarly, application designers can gain great experience with design patterns and secure SDLC methodologies while application developers can learn how to write secure code.

On the other hand, I would not recommend the book for students who might have less security and development knowledge due to missing real-life project experience. The broad content of the book could be quite complicated for students to follow.

The reviewer is working as IT-Security Architect and Researcher by Daimler TSS (Technologies, Services, Solutions) in Germany.