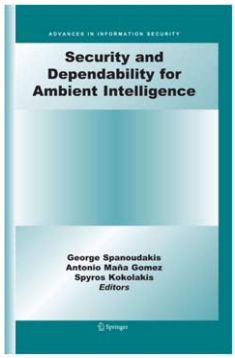
Journal of Ambient Intelligence and Smart Environments 3 (2011) 373–374
DOI 10.3233/AIS-2011-0119
IOS Press

Book review



Security and dependability for Ambient Intelligence: *Informative but busy*

Yee Wei Law [a,*] and Paul Havinga [b]
[a] *Dept. Electrical and Electronic Engineering, The University of Melbourne, Parkville, VIC 3010, Australia*
[b] *Pervasive Systems, Dept. of Computer Science, Faculty of EEMCS, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands*
E-mail: paul.havinga@utwente.nl

Abstract. The edited volume "Security and Dependability for Ambient Intelligence" is a comprehensive compilation of the research outcomes of the 3 year-long €7.8 million European Framework Programme 6 project SERENITY (FP6-IST-2006-27587). At a time when Stuxnet and large scale data breaches at PlayStation Network and RSA have taken over global news headlines, the need for a systematic approach to developing, deploying and dynamically configuring security solutions marks the timely arrival of this highly useful volume.

Keywords: Ambient intelligence, security patterns, interoperability, model-based development, adaptive security

Security and Dependability for Ambient Intelligence (Advances in Information Security, Vol. 45) by G. Spanoudakis, A. M. Gomez, and S. Kokolakis, New York, NY, USA: Springer Science+Business, LLC, 2009, ISBN: 978-0-387-88774-6.

1. Introduction

In the era of smart things (e.g., smart home, smart grid, Smart Dust), it almost seems compulsory for every researcher to re-position his/her work in the context of the latest research frontier, which in this case is Ambient Intelligence (AmI). The goal of this volume, as well as project SERENITY, is to enhance the security of AmI "ecosystems" by providing a framework supporting the development, integration, configuration, monitoring and adaptation of security mechanisms for these systems. The fact is, the volume is valid with or without AmI, because the concepts and tools provided by this project are generally applicable. The comprehensive volume is organized into six parts:

Part A General overview of security engineering
Part B Introduction to the so-called Security and Dependability (S&D) artifacts
Part C Details of the SERENITY Development Framework (SDF)
Part D Details of the SERENITY Runtime Framework (SRF)
Part E Modeling of organizational processes and legal requirements
Part F Experiences and future directions

While the amount of information provided by 392 pages is impressive, clarity suffers at places.

2. Key concepts

Project SERENITY is all about *S&D Patterns*, the purpose of which is to enable security experts of cer-

*Corresponding author. E-mail: ywlaw@unimelb.edu.au.

tain security mechanisms to encode their knowledge and experience into models that describe the mechanisms, for use by application developers with automated tools. *S&D Classes* address the key issue of interoperability, and are intended to be defined by entities interested in interoperability, such as standardization bodies. One aspect where the volume could have done better is providing a "one-stop shop" of *precise* definitions of the key S&D artifacts. The following represents our endeavor to distill a few key definitions from the volume:

- An *S&D Property* is a security objective expressed in the Formal S&D Properties Language.
- An *S&D Solution* is a security mechanism that provides one or more S&D Properties.
- An *S&D Pattern* is an abstract representation of an S&D Solution in either a formal or an informal language such as XML.
- An *S&D Implementation* is an operational representation of an S&D Solution or Solutions.
- An *Executable Component* is a concrete implementation of an S&D Pattern, that is referenced by an S&D Implementation.
- An *S&D Class* is a set of S&D Patterns characterized by supporting the same S&D Properties and having a common interface.
- An *S&D Library* is a catalogue of S&D Classes, S&D Patterns, and S&D Implementations, to be chosen by developers based on application requirements.
- The *SERENITY Development Framework* (SDF) is a set of concepts and tools (such as S&D Libraries) that assist developers with the task of developing secure applications.
- The *SERENITY Runtime Framework* (SRF) is a suite of tools that enable the dynamic configuration, binding, monitoring and replacement of S&D mechanisms in applications. The monitoring service of the SRF is called EVEREST.

When designing an AmI service, infrastructure, or application, the security requirements of the project are specified using a requirements specification language, such as the simple text-based language briefly discussed in Section 10.3.1.2. Then, the application to be developed is modeled using the SI* modeling language as part of the Secure Tropos methodology, or UMLsec. At development time, required S&D Properties are identified. To support interoperability, S&D Classes are defined or chosen from an S&D Library based on the application requirements. S&D Patterns supporting the identified S&D Properties and S&D Classes are also chosen. At run time, the SRF activates referenced Executable Components to realize S&D Solutions represented by the chosen S&D Patterns, and monitors for anomalies. In response to detection of anomalous events by EVEREST, the SRF may deactivate the Executable Components in question, and activate other S&D Patterns, thereby adapting to changing operating conditions.

3. Strengths and weaknesses

We find the reviews in Part A, the discussion of event calculus-based monitoring algorithms in Part D, the extensive bibliography and index of the volume to be particularly useful. While aspects of organizational modeling are interesting, Chapter 15 maybe too abstract to be considered useful for security practitioners. Chapter 16 on legal modeling would be more useful, had a concrete modeling example been given.

A disappointing aspect of the volume is its failure to present a high-level recipe-like use case that ties every major component together. In fact, we provided such a use case in the previous section not only to aid our own understanding but also to illustrate what could have been done. At places, the volume also fails to lay down necessary groundwork for the non-expert readers. For example, (Secure) Tropos appears a number of times without accompanying introduction. Lastly, despite its AmI focus, the volume is not generous with AmI-specific use cases.

4. Conclusion

The volume has delivered on the security part, although not so much on the dependability part. Due to its compiled nature, it is marred by repetitive content and inconsistent presentation. Ignoring the misses, security practitioners looking to build secure applications using an elegant framework backed by formal methodology will find the volume highly useful, more so if or when the SERENITY toolbox becomes publicly available (see http://www.serenity-project.org/Work-package-8-3.html). As such, the volume is a welcome addition to the security engineering literature.