Chapter 74
Evolution of Security Engineering Artifacts:
A State of the Art Survey

Michael Felderer
University of Innsbruck, Austria

Basel Katt
University of Innsbruck, Austria

Philipp Kalb
University of Innsbruck, Austria

Jan Jürjens
Technical University of Dortmund, Germany

Martín Ochoa
Technical University of Munich, Germany

Federica Paci
University of Trento, Italy

Le Minh Sang Tran
University of Trento, Italy

Thein Than Tun
The Open University, UK

Koen Yskout
iMinds-DistriNet, KU Leuven, Belgium

Riccardo Scandariato
iMinds-DistriNet, KU Leuven, Belgium

Frank Piessens
iMinds-DistriNet, KU Leuven, Belgium

Dries Vanoverberghe
iMinds-DistriNet, KU Leuven, Belgium

Elizabeta Fourneret
University of Luxembourg, Luxembourg

Matthias Gander
University of Innsbruck, Austria

Bjørnar Solhaug
SINTEF, Norway

Ruth Breu
University of Innsbruck, Austria

ABSTRACT

Security is an important quality aspect of modern open software systems. However, it is challenging to keep such systems secure because of evolution. Security evolution can only be managed adequately if it is considered for all artifacts throughout the software development lifecycle. This article provides state of the art on the evolution of security engineering artifacts. The article covers the state of the art on evolution of security requirements, security architectures, secure code, security tests, security models, and security risks as well as security monitoring. For each of these artifacts the authors give an overview of evolution and security aspects and discuss the state of the art on its security evolution in detail. Based on this comprehensive survey, they summarize key issues and discuss directions of future research.

DOI: 10.4018/978-1-4666-8473-7.ch074

1. INTRODUCTION

Due to ever changing surroundings, new business needs, new regulations and new technologies, a software system must evolve, or it becomes progressively less satisfactory (Lehman, 1980, 1998). On the one hand, the continuous system evolution makes it especially challenging to keep software systems permanently secure as changes, either in the system itself or in its environment, may cause new threats and vulnerabilities. On the other hand, security artifacts themselves like security requirements, security architectures, and secure code or security tests have to be continuously adapted in long-running software systems. Because modern open and dynamically-changing software systems like service-oriented architectures or cloud deployments determine business process implementations and deal with critical data, managing the evolution of their security artifacts in all phases of the software development lifecycle (SDLC) is of high importance.

The main phases of the SDLC are *analysis, design, implementation, testing*, as well as *deployment* and *operation* (Braude & Bernstein, 2011). In each phase, specific artifacts are created or adapted, i.e., requirements in the analysis phase, the architecture in the design phase, source code in the implementation phase, tests in the testing phase, as well as the running system in the deployment and operation phase. All these artifacts are subject to changes which is one of the main difficulties of software evolution (Mens & Demeyer, 2008) with high impact on security engineering.

Security engineering focuses on security aspects in the software development lifecycle. Security aims at protecting information and systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. The main objective of security is to guarantee confidentiality, integrity and availability of information and systems. To be most effective, security must be integrated into the software development lifecycle from the very beginning (Kissel et al., 2008).

Risk management in general is the process allowing organizations to identify what assets need to be protected, what threats prevail and with what probability and severity losses could occur. As such it is an indispensable activity in security engineering to identifying and to manage threats and vulnerabilities to information as well as systems.

Model engineering involves the systematic use of models as essential artifacts throughout the software development process (Schmidt, 2006). It has recently been applied in security engineering to provide security models for all phases of the software development lifecycle to manage the evolution of security engineering artifacts.

Figure 1 gives an overview of the security engineering activities and the assigned artifacts in the secure software development lifecycle. In each iteration, the activities analysis, design, implementation, development as well as deployment are performed consecutively. Additionally, risk management and model engineering accompany these activities. As the system and its environment evolve, all these activities are executed iteratively which is represented by the surrounding border. Each phase handles specific artifacts, i.e., requirements, architecture, code, tests, running system, models and risks.

Each of these artifacts corresponds to sections in this article with respect to its security-specific evolution aspects. Managing the evolution of security engineering artifacts is an important task that needs specific approaches to continuously guarantee security.

This article reviews the state of the art of evolution management of security engineering artifacts and draws conclusions for future research. Section *Security Model Evolution* discusses security model evolution, and Section *Security Requirements Evolution* covers evolution of security requirements. Section *Security Architecture Evolution* discusses security architecture evolution, while