Using Security Patterns to Develop Secure Systems—Ten Years Later

Eduardo B. Fernandez, Florida Atlantic University, Boca Raton, USA
Hironori Washizaki, Waseda University, Tokyo, Japan
Nobukazu Yoshioka, National Institute of Informatics, Tokyo, Japan

ABSTRACT

The authors describe continuing work on the use of patterns in the development of secure systems. This work started as collaboration among five research centers on three continents and continues with a reduced number of involved researchers. Patterns are applied to all aspects of development, from domain analysis and attack modeling to basic design, and to all aspects of the systems under development, from the database and infrastructure to policies, monitoring, and forensics. The article provides an overview of a method of development involving the full range of patterns and describes recent contributions from some of the research threads being pursued within the collaboration.

KEYWORDS

Attack Modeling, Basic Design, Domain Analysis, Infrastructure

1. INTRODUCTION

We initiated an international collaboration between our security groups about 12 years ago, centered on methodologies to build secure systems using patterns. We summarized our work at that time in (Fernandez et al., 2010). We describe here where we are now and where we are going. This article should be considered a survey of our work and not an attempt to present new work or to introduce in detail the models presented here, for that we refer the reader to our previous publications. We also provide a section comparing our work to others but again in each paper we relate our work to others in more detail. In particular, we have worked or we are working on:

1. Secure software development methodology: We have worked on a general methodology to build secure systems and have produced until now some specific aspects of it, which are described below. Of course, these aspects have value independently of this methodology and can be applied to other methodologies or used on their own;
2. Modeling and Classification of security patterns: We have tried to provide a precise characterization of security patterns that can be used as a basis for classification. A good classification makes the application of the patterns much easier along the software lifecycle. It
also helps understand the nature and value of the patterns. Another objective is to identify which patterns are missing;

3. **Misuse patterns:** A misuse pattern describes, from the point of view of the attacker, how a type of attack is performed (what units it uses and how), analyzes the ways of stopping the attack by enumerating possible security patterns that can be applied for this purpose, and describes how to trace the attack once it has happened by appropriate collection and observation of forensics data. They can be used in the lifecycle to prevent the occurrence of known types of attacks and to evaluate a completed system;

4. **Characterization and selection of access control models:** Access control is a fundamental aspect of security. There are many variations of the basic access control models and it is confusing for a software developer to select an appropriate model for her application. We have defined a way to clarify their relationships and a way to guide designers in selecting an appropriate model;

5. **Databases in secure applications:** Most applications need to include databases to store the persistent information, which constitutes most of the information assets of the institution. We have studied the effect of databases on the security of a system under development.

We have not continued working on topics 2, 4, and 5. Instead, we have started work on using our models to describe cloud and IoT systems. The work on misuse patterns has been included in the study of clouds and IoT. The following sections describe these aspects in detail.

### 2. BACKGROUND

A pattern is a solution to a recurring problem in a specific context. Software patterns are categorized as analysis, design, architecture, and security patterns. Abstract patterns describe a basic semantic aspect while Abstract Security patterns (ASP)s, realize one or more security policies able to control (stop or mitigate) a threat or comply with a security-related regulation or institutional policy (Fernandez et al., 2014). Patterns are described using a template composed of a set of sections. A problem section describes a problem and the forces that constrain and define guidelines for its solution, e.g., “the solution must be transparent to the users of the system”. Pattern solutions are usually described using modeling languages such as the Unified Modeling Language (UML), maybe combined with formal languages such as the Object Constraint Language (OCL). UML diagrams may include class, sequence, state, and activity diagrams. A set of consequences indicate how the pattern solved the specific problem and what are the advantages and disadvantages of using it; i.e., how well the forces were satisfied by the solution. An implementation section provides hints on how to use the pattern in an application maybe showing details of its use in a real system. A section on “known uses” lists real systems where this solution has been used previously, i.e., a pattern is an abstraction of a good practice. A section on related patterns indicates patterns that complement or provide alternative solutions to the one in this pattern. A pattern embodies the knowledge and experience of software developers and can be reused in new applications; carefully designed patterns implicitly apply good design principles. Patterns are also good for communication between designers and to evaluate and reengineer existing systems. While initially developed for software, patterns can describe hardware, physical entities, and combinations of these. Pattern solutions are suggestions, not plug-ins or software components. A compound pattern is composed of two or more simpler patterns.

A Reference Architecture (RA) is an abstract software architecture, based on one or more domains, with no implementation aspects (Taylor et al., 2010). An RA should define the fundamental concepts of a system expressed as ASPs and the interactions among these units. An RA should be reusable, extendable, and configurable; that is, it is a kind of compound pattern for whole architectures and it can be instantiated into a concrete software architecture by adding platform aspects. In addition to class and sequence diagrams, an RA may include a set of use cases (UC), and a set of Roles (R)
10 more pages are available in the full version of this
document, which may be purchased using the "Add to Cart"
button on the product's webpage:
www.igi-global.com/article/using-security-patterns-to-
develop-secure-systemsten-years-later/232748?camid=4v1

This title is available in InfoSci-Computer Systems and
Software Engineering eJournal Collection, InfoSci-
Surveillance, Security, and Defense eJournal Collection,
InfoSci-Knowledge Discovery, Information Management, and
Storage eJournal Collection, InfoSci-Physical Sciences,
Biological Sciences, and Engineering eJournal Collection,
InfoSci-Journals, InfoSci-Journal Disciplines Engineering,
Natural, and Physical Science, InfoSci-Journal Disciplines
Computer Science, Security, and Information Technology,
InfoSci-Select. Recommend this product to your librarian:
www.igi-global.com/e-resources/library-
recommendation/?id=163

Related Content
A Practical Framework for Policy Composition and Conflict Resolution
Ousmane Amadou Dia and Csilla Farkas (2012). International Journal of Secure
Software Engineering (pp. 1-26).
www.igi-global.com/article/practical-framework-policy-composition-
conflict/74842?camid=4v1a

Recovering Polyp Shape from an Endoscope Image Using Two Light
Sources
Hiroyasu Usami, Yuji Iwahori, Yuki Hanai, Boonserm Kijsirikul and Kunio Kasugai
www.igi-global.com/article/recovering-polyp-shape-from-an-endoscope-
image-using-two-light-sources/176666?camid=4v1a
A Survey of Web Services Provision
An Liu, Hai Liu, Baoping Lin, Liusheng Huang, Naijie Gu and Qing Li (2010).
International Journal of Systems and Service-Oriented Engineering (pp. 26-45).
www.igi-global.com/article/survey-web-services-provision/39097?camid=4v1a

Open Source Software and the Corporate World
www.igi-global.com/chapter/open-source-software-corporate-world/29509?camid=4v1a