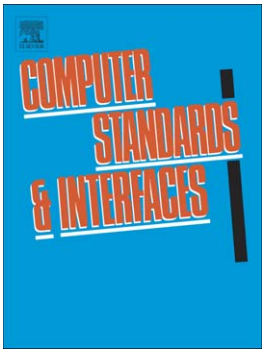

ASE: A Comprehensive Pattern-Driven Security Methodology for Distributed Systems

Anton V. Uzunov, Eduardo B. Fernandez, Katrina Falkner

PII: S0920-5489(15)00027-6
DOI: doi: 10.1016/j.csi.2015.02.011
Reference: CSI 3015

To appear in: *Computer Standards & Interfaces*

Received date: 27 July 2014
Revised date: 31 January 2015
Accepted date: 18 February 2015

Please cite this article as: Anton V. Uzunov, Eduardo B. Fernandez, Katrina Falkner, ASE: A Comprehensive Pattern-Driven Security Methodology for Distributed Systems, *Computer Standards & Interfaces* (2015), doi: 10.1016/j.csi.2015.02.011

ASE: A Comprehensive Pattern-Driven Security Methodology for Distributed Systems

Anton V. Uzunov[a,*], Eduardo B. Fernandez[b], Katrina Falkner[a]

a, School of Computer Science, The University of Adelaide, Adelaide, South Australia, Australia, 5005
b, Department of Computer and Electrical Engineering and Computer Science, Florida Atlantic University, 777 Glades Rd., Boca Raton, FL 33431

* Corresponding author

E-mail addresses: {anton.uzunov, katrina.falkner}@adelaide.edu.au (Anton V. Uzunov, Katrina Falkner), ed@cse.fau.edu (Eduardo B. Fernandez)

Abstract: Incorporating security features is one of the most important and challenging tasks in designing distributed systems. Over the last decade, researchers and practitioners have come to recognize that the incorporation of security features should proceed by means of a structured, systematic approach, combining principles from both software and security engineering. Such systematic approaches, particularly those implying some sort of process aligned with the development life-cycle, are termed *security methodologies*. There are a number of security methodologies in the literature, of which the most flexible and, according to a recent survey, most satisfactory from an industry-adoption viewpoint are methodologies that encapsulate their security solutions in some fashion, especially via the use of *security patterns*. While the literature does present several mature pattern-driven security methodologies with either a general or a highly specific system applicability, there are currently no (pattern-driven) security methodologies specifically designed for general distributed systems. Going further, there are also currently no methodologies with mixed specific applicability, e.g. for both general and peer-to-peer distributed systems. In this paper we aim to fill these gaps by presenting a comprehensive pattern-driven security methodology – arrived at by applying a previously devised approach to engineering security methodologies – specifically designed for general distributed systems, which is also capable of taking into account the specifics of peer-to-peer systems as needed. Our methodology takes the principle of encapsulation several steps further, by employing patterns not only for the incorporation of security features (via security solution frames), but also for the modeling of threats, and even as part of its process. We illustrate and evaluate the presented methodology in detail via a realistic example – the development of a distributed system for file sharing and collaborative editing. In both the presentation of the methodology and example our focus is on the early life-cycle phases (analysis and design).
Keywords: secure software engineering, security methodologies, distributed systems security, security patterns, threat patterns, security solution frames

1 Introduction

Incorporating security features is one of the most important and also one of the most challenging tasks in designing distributed systems [1, 2]. Over the last decade, researchers and practitioners have come to recognize that the incorporation of security features should proceed by means of a structured, systematic approach, combining principles from both software and security engineering [3, 4, 5, 6, 7]. Such systematic approaches, particularly those implying some sort of process aligned with the software development life-cycle, are termed *security methodologies* [8]. There are a number of security methodologies in the literature, of which the most flexible and most satisfactory from an industry-adoption viewpoint are methodologies that encapsulate their security solutions in some fashion (see [8]), especially via the use of *security patterns* [9, 10]. While the literature presents over a dozen such pattern-driven security methodologies, both young and mature [8, 11] – possessing a range of valuable and beneficial features – with respect to system applicability, these methodologies are uncomfortably positioned at two extremes of a spectrum: either they are highly specific, or highly generic. This makes such methodologies inadequate for project situations requiring the development of general distributed systems, since the methodologies will either lack provisions for the specific security concerns of general distributed systems or different types of distributed systems (too generic) [5]; or they will be incompatible with the features of the target system (too specific) [11] – whether because of the processes involved (e.g. PWSSec [12]); or because of the conceptual artefacts used (e.g. the methodology of Delessy and Fernandez [13]).

At present, there are no pattern-driven security methodologies specifically designed for general distributed systems – i.e. positioned somewhere in the middle of the specificity-generality spectrum referred to above (we are considering here exclusively methodologies using security patterns, not patterns interpreted as architectures or components as in the work of [14]). Going further, there are also currently no methodologies in the literature with mixed specific applicability [11] – for example, for both general and peer-to-peer distributed systems; or for general and web-based applications.

In this paper we aim to fill the latter gaps, by presenting a comprehensive pattern-driven security methodology specifically designed for general distributed systems, named ASE, which is also capable of taking into account the specifics of peer-to-peer systems as needed. ASE emphasizes the early life-cycle phases (analysis and design) – since