Modeling, composing, and testing of security concerns in a Model-Driven Security approach*

Phu H. Nguyen**, Jacques Klein, and Yves Le Traon

Interdisciplinary Centre for Security, Reliability and Trust (SnT),
University of Luxembourg, 4 rue Alphonse Weicker, L-2721 Luxembourg
{phuhong.nguyen,jacques.klein,yves.letraon}@uni.lu

Abstract. Model-Driven Security (MDS) has emerged as a promising sound methodology for supporting the development of secure systems nowadays. Following the advances in MDS, this research work aims at 1) developing new modeling techniques to represent multiple security concerns, 2) (automatically) composing security models with the business logic model (called target model), and 3) testing the security model composition and the resulting secure system against security requirements. These three objectives converge to an integrated MDS framework (and tool chain) which 1) allows a target system model to embed various security concerns, 2) enables the generation of implementation code including configured security infrastructures, and 3) makes these security properties testable by construction. This paper presents the main research modules, the results we have achieved so far, and the main points for future work.

Keywords: Model-Driven Security, Model-Driven Engineering, Security Modeling, Model Composition, Adaptive Security, Security Testing

1 Introduction

It is undoubted that the role of security engineering is getting more important than ever as our world is becoming more digital and networked. However, the traditional software development process has shown the inefficiency in facing with the following three main challenges for nowadays security engineering. First, (software) systems are getting more and more complex. Especially, taking into account security concerns while developing (already complex) systems makes the development process more stressful, error-prone and difficult. Second, security threats are getting more dangerous, varied, and changing quickly. These make security requirements more complex, difficult to deal with. Security features are often scattered and tangled throughout the entire system. It is hard to integrate them properly into the traditional development process. However, they are rarely taken into account at early stages of the development processes

* This work is supported by the Fonds National de la Recherche (FNR), Luxembourg, under the project C10/IS783852/ MITER.

** PhD candidate supervised by Prof., Dr. Yves Le Traon and Dr. Jacques Klein.

[2]. In other words, the overall system design often misses the cautious engineering of security. Third, even though the complexity of systems (especially including security concerns) that have to be produced and maintained is continuously increasing, economic pressure reduces development time and increases the frequency of modifications are made. All these issues constantly require more productive and flexible security engineering methods for better supporting the development and maintenance of reliable secure systems.

Model-Driven Engineering (MDE) has been considered by some researcher [1] as a solution to the handling of complex and evolving software systems. As a specialization of MDE, Model-Driven Security (MDS) aims at providing means to tackle the complexity and increase the productivity in modern secure systems development. MDS enables security models more productive, i.e., models could be manipulated automatically in every development stage. The three main challenges mentioned above could be solved by MDS [5]. However, there are weaknesses of the MDS state of the art. Our recent systematic review of MDS [5] shows that more MDS work is needed to deal with multiple security concerns at the same time. Moreover, not many MDS approaches have fully leveraged the Aspect-Oriented Modeling (AOM) techniques to specify multiple security concerns and enhance the modularity of secure systems. There is also lack of a complete tool chain (based on model transformations) to automate the derivation from MDS models to code. Following the advances in MDS, this research work aims at 1) developing new (AOM) modeling techniques to represent multiple security concerns, 2) (automatically) composing security models with the business logic model (called target model), and 3) testing the security model composition and the resulting secure system against security requirements. These three objectives converge to an integrated MDS framework (and tool chain) which 1) allows a target system model to embed various security concerns, 2) enables the generation of implementation code including configured security infrastructures, and 3) makes these security properties testable by construction.

The remainder of this paper is structured as follows. Section 2 describes the main objectives, and the main (expected) contributions of our proposed research. It is followed by Section 3 that shows our main research modules, and track record of the archived results so far and what missing. Finally, the conclusion and future work are presented in Section 4.

2 Objectives

2.1 Research Objectives

There are three main research objectives in this work. First. we aim at proposing a portfolio of well-defined security models without any consideration of a target model i.e., the model in which the security models will be inserted or composed. Consequently, each security concern will be modeled in isolation leading to a better understanding and modularization of these security concerns.

Second, we propose to automatically compose a subset of selected security models with the target system model to obtain a new model of the system

augmented of security properties. This model composition can be performed using model transformations or model composers. Once various security models specified, the automation of the composition should allow to adapt more easily the target model to different situations by allowing the automated composition of appropriate security models. Moreover, the model of the systems augmented of security properties can be used for formal analyze of security properties.

Third, we propose to exploit the model composition operators to make the final implementation testable by construction. Composing security models (viewpoints) into the target model will lead to a more detailed model, which will finally be implemented. The code production is error prone and the conformance of the implementation with the security policy must be tested. The composition operators we propose may offer an elegant way 1) to make the implemented security mechanisms testable, in the sense they can be made observable at runtime, 2) to propose a security fault model to perform mutation analysis on the final code.

2.2 Expected Contributions

From our proposed approach, the main expected contributions of our work are as follows: 1) a library of security concerns and a set of Domain Specific Languages (DSLs) for specifying these security concerns; 2) a model-driven framework for composing security models (conforming to these DSLs) with the target model; and 3) a mutation analysis approach for testing the resulting secure systems.

3 Research Modules & Track Record

This section shows the four main modules of our research, with the results that have been achieved so far, and the approach on how to complete the thesis.

3.1 Literature Review of Model-Driven Security

This module focuses on the state-of-the-art of MDS and the key aspects of our project, i.e. modeling, composing, and testing of security concerns. Because we realized that there was not any real systematic review of MDS, we conducted one whose results are presented in [5]. The results show not only a clear picture of current main approaches in MDS but also the current status of key aspects in MDS.

The results suggest that more attention should be paid for dealing with multiple security concerns at the same time. Most current approaches only deal with solely one security concern, especially authorization. Besides, there are significantly less MDS papers tackling integrity, availability, and authentication than authorization and confidentiality. An important remark is that more work should be done to have (DSLs) models with well-defined semantics of various security concerns. These models must be extensively, formally defined in order to enable the integration with automated analysis tools (based on well-established formal methods) and/or program synthesis tools. On the other hand, a tool

chain (based on model transformations) to derive from models (to models, then) to implementation code is also an important piece of future work. Regarding modeling approaches, there are very few selected papers propose a full AOM approach that security concerns are specified as aspects and eventually woven into the primary model(s). Last but not least, there is a lack of empirical studies for MDS approaches so more empirical studies should be conducted.

The details of five main MDS approaches have been discussed in our book chapter, namely Advances in MDS [4]. Moreover, we are working on extending [5] for submission to a journal.

3.2 Modeling Security Concerns

As discussed in [5], domain specific languages (DSLs) could be defined for specifying security concerns. We aim at proposing a set of DSLs which specialized for capturing well-defined semantics of various security concerns. These DSLs are used for creating well-specified security models without any consideration of a target model i.e., the model in which the security models will be inserted or composed. Currently, we are trying to introduce these DSLs in a full AOM approach, e.g. using Reusable Aspect Model [3]. Consequently, each security concern will be modeled in isolation leading to a better understanding and modularization of these security concerns.

As the first approach, we focused on dealing with the authorization problem, especially access control (role-based access control, RBAC) and delegation. In [6], it has been shown that various RBAC-based delegation features can be specified using our metamodel (DSL). Our DSL supports complex delegation characteristics like temporary, recurrence delegation, transfer delegation, multiple and multi-step delegation, etc. On the other hand, the business logic (base system) can be specified using another DSL, e.g. an architecture (component-based) metamodel [6]. In the next step, we target to deal with multiple security concerns at the same time in a full AOM approach.

3.3 Composing Security Models with the Target System

In the modeling module of our approach, security concerns are modeled independently with business logic. In this module, security models have to be composed with the target system model to obtain a new model of the system augmented of security properties. Fig. 1 presents an overview of our extensive model-driven approach for access control and delegation management [6], [7]. In our approach, delegation is considered as a "meta-level" mechanism which impacts the existing access control policies, like an aspect can impact a base program. We claim that to handle advanced delegation rules, an ideal solution is to separate the delegation rules from the access control policy, each being specified in isolation, and then compose/weave them together to obtain a new access control policy (called active security policy) reflecting the both access control and delegation.

As can be seen in Fig. 1, a complete model-driven framework has been proposed to enable dynamic enforcement of delegation and access control policies

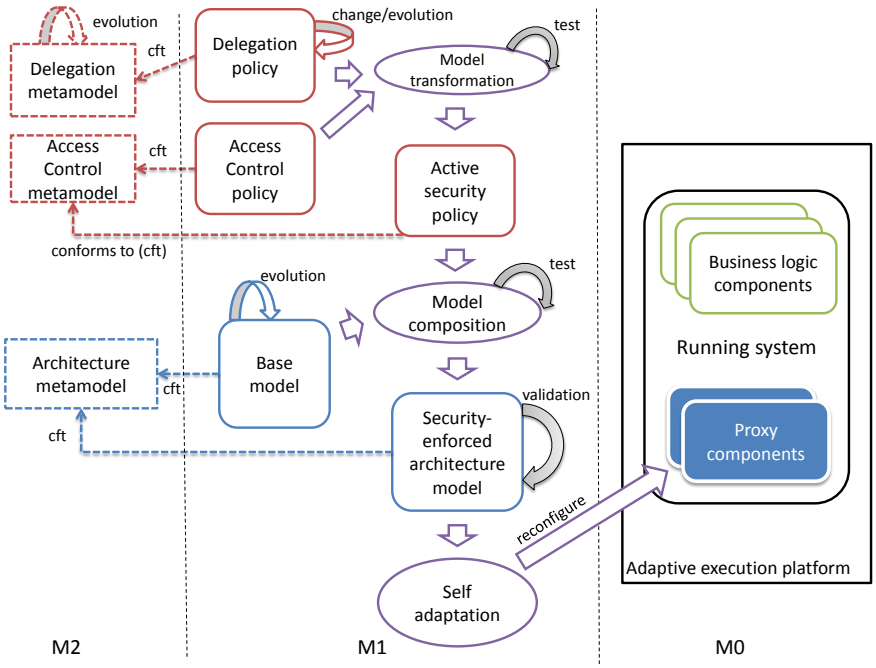

Fig. 1. Overview of our Model-Driven Adaptive Delegation approach

that allows the automatic configuration of the system according to the changes in delegation/access control rules. The enforcement of security policy to the target system is in fact the composition of security models with the target system model. The dynamic adaptation of the running system is possible thanks to the modern adaptive execution platforms like OSGi [1], Kevoree [2], which provide low-level APIs to reconfigure a system at runtime.

3.4 Model-Based Security Testing

We focus on proposing a Model-Based Security Testing and Mutation Analysis approach for the validation of the resulting secure system. From the security model(s), we plan to automate the (partial) generation of security test cases. Our goal of using mutation analysis is to derive a sufficient test set, which can detect all the security faults denoted by the mutants. In that way the correct secure system that can pass the sufficient test set will be obtained. We have adopted mutation analysis for delegation policies [8]. Mutation analysis operates by introducing artificial defects called mutants into the artifacts of the program under investigation. Our approach in [8] consists of analyzing the representation of the key components of delegation, based on which we derive the suggested set of mutant operators. These operators can then be used to introduce mutants

[1] www.osgi.org
[2] www.kevoree.org

into delegation policies and thus, enable mutation testing. There is still more work to be done to validate our idea in [8].

Here we propose to use security testing because so far formal verification methods for security still have limitations. Only some specific problem areas such as smart-cards or cryptographic protocols are applicable for formal verification methods. Formal verification is still unfeasible for larger systems due to increased complexity and dependencies.

4 Conclusion and Future Work

In this paper, we have presented our research work with its main modules and track record. Regarding the first module which is literature review, we have conducted a systematic literature review of MDS and a book chapter on advances in MDS. This module is more or less done even though we are still working on a journal version of the review. In the second module which is modeling of security concerns, we have dealt with access control and delegation. But there is still more work to be done for modeling multiple security concerns. The third module focuses on composing models. We have proposed a framework for model-driven adaptive delegation. Our model-driven framework needs to be improved for handling multiple security concerns at the same time. And last but not least, the fourth module is about security testing. Our work on testing delegation policy enforcement via mutation analysis [8] is at the beginning. We will continue working on that. In the end, we also would like to have an industrial case study for evaluating our approach.

References

1. J. Bezivin. Model driven engineering: An emerging technical space. *GTTSE*, 2006.
2. L. Cysneiros and J. Sampaio do Prado Leite. Non-functional requirements: from elicitation to modelling languages. In *ICSE 2002*, pages 699–700, 2002.
3. J. Kienzle, W. Al Abed, F. Fleurey, J.-M. Jezequel, and J. Klein. Aspect-oriented design with reusable aspect models. In *TAOSD VII*, volume 6210. 2010.
4. L. Lucio, Q. Zhang, P. H. Nguyen, M. Amrani, J. Klein, H. Vangheluwe, and Y. Le Traon. *Advances in Model-Driven Security*. Elsevier, 2014.
5. P. H. Nguyen, J. Klein, M. Kramer, and Y. Le Traon. A Systematic Review of Model Driven Security. In *Proceedings of the 20th APSEC*, 2013.
6. P. H. Nguyen, G. Nain, J. Klein, T. Mouelhi, and Y. Le Traon. Model-Driven Adaptive Delegation. In *AOSD*, 2013.
7. P. H. Nguyen, G. Nain, J. Klein, T. Mouelhi, and Y. Le Traon. Modularity and Dynamic Adaptation of Flexibly Secure Systems: A Model-Driven Approach for Delegation in Access Control Management. In *TAOSD XI*. 2014.
8. P. H. Nguyen, M. Papadakis, and I. Rubab. Testing Delegation Policy Enforcement via Mutation Analysis. In *Proceedings of the Workshop on Mutation Testing, the Sixth IEEE International Conference on Software Testing*, 2013.