Model-Driven Security with *Modularity* and *Reusability* for Secure Systems Development[⋆]

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 modern secure systems development. Following the advances in Mds, our work described in this paper has proposed a solution to better support secure systems development, and further strengthens Mds. Our Mds solution focuses on *modularity* and *reusability* in secure systems development. On one hand, we have proposed a modular approach for modularity and dynamic adaptation of flexibly secure systems. On the other hand, we have been working on Mds based on a library-like *System of generic Security design Patterns* (shortly called SoSPa) in which security design patterns are collected, specified as reusable aspect models to form a coherent system of them that guides developers in systematically selecting the right security design patterns for the job. Either way, security (design pattern) models can be automatically woven into the target system model. The woven secure system model can then be used for (partial) code generation, including (configured) security infrastructures. We have also worked on model-based security testing to validate the resulting secure systems.

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

1 Problem and Motivation

Model-Driven Engineering (Mde) has been considered by some researcher [2] 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. However, there are still weaknesses in the state of the art of Mds. 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 Modelling* (Aom) techniques for specifying multiple security concerns, and for enhancing

⋆ 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.

the *modularity* of secure systems. MDS also lacks a complete tool chain (based on model transformations) to automate the derivation from MDS models to code. Besides, security patterns which are based on domain-independent, time-proven security knowledge and expertise, can be considered as *reusable* security bricks upon which sound and secure systems can be built. But security patterns are not applied as much as they could be because developers have problems in selecting them and applying them in the right places, especially at the design phase. Following the advances in MDS, this paper describes the late-PhD work that has proposed a solution to better support secure systems development, and further strengthens MDS. Our MDS solution focuses on *modularity* and *reusability* in secure systems development. On one hand, we have proposed a modular approach for modularity and dynamic adaptation of flexibly secure systems. On the other hand, we have been working on MDS based on a library-like *System of generic aspect-oriented Security design Patterns* (SoSPa) in which security design patterns are collected, specified as reusable aspect models to form a coherent system of them that guides developers in systematically selecting the right security design patterns for the right job. Either way, security (design pattern) models can be automatically woven into the target system model. The woven secure system model can then be used for (partial) code generation, including (configured) security infrastructures. We have also worked on model-based security testing to validate the resulting secure systems. Our research work aims towards an integrated MDS framework (and tool chain) which 1) allows a target system model to embed various security solutions, 2) enables the generation of (partial) implementation code including (configured) security infrastructures, and 3) makes these security properties testable by construction.

2 Related Work and Our Systematic Review of MDS

There have been a significant number of MDS approaches proposed so far. Recently, we have conducted a systematic review of MDS and then published a paper reporting the results of our review [5]. One of the main results is that we have found out five primary MDS approaches according to our selection criteria. These approaches are listed as follows: *SecureUML*, *UMLsec*, *SECTET*, *SecureMDD*, and *Secure data warehouses*. More details about these MDS approaches can be found in [5], and in our book chapter, namely Advances in MDS [4]. Moreover, we are working on extending [5] for submission to a journal.

The results in [5] 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 *Domain Specific Languages* (DSLs) or models with well-defined semantics of various security concerns. These models must be extensively, formally defined in order to enable

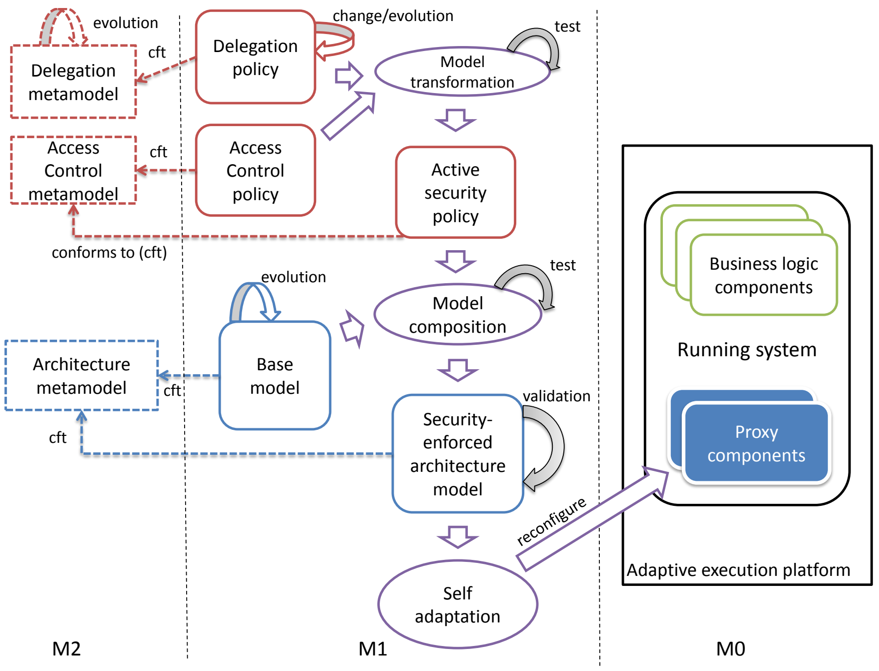

Fig. 1. Overview of our Model-Driven Adaptive Delegation approach as shown in [7]

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, and 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 of MDS should be conducted.

3 MDS with Modularity and Dynamic Adaptation

As the first approach, we focused on dealing with the authorization problem, especially role-based access control (RBAC) and delegation. In [7], 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 [7].

In the modeling phase, security concerns are modeled independently with business logic. Then, 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 [7], [8]. 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 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.

4 MDS with Modularity and Reusability

In the previous section, we has shown our MDS approach for modularity and dynamic adaptation of secure systems. In that approach, access control and delegation policies are modeled by using a tailored DSL which is limited for only authorization (access control and delegation). In this section, we present an exploratory MDS approach that can be used for developing secure systems dealing with multiple security concerns (e.g., authentication, authorization, encryption, etc.) at the same time. To be more specific, our approach is based on a system of *reusable* aspect-oriented security design patterns which is generic and extensible to address all security concerns with the interrelations among them.

Security patterns which are based on domain-independent, time-proven security knowledge and expertise, can be considered as *reusable* security bricks upon which sound and secure systems can be built. A security pattern solely can not help to secure a system against different threats. Thus, it is necessary to build a system of security patterns in which inter-pattern relations are specified [10]. To the best of our knowledge, none of existing MDS approaches has proposed a *System of generic aspect-oriented Security design Patterns* which provides not only well-defined security design patterns but also the support meta-information that can guide developers in systematically selecting the right security design patterns for the right job. We have proposed an exploratory MDS approach based on a *System of generic aspect-oriented Security design Patterns* (shortly called SoSPa[3]) [6] in which security design patterns are collected, specified as reusable aspect models (RAM) [3] to form a coherent system of them. Our approach is inspired by [1] in which the authors propose an approach based on RAM for designing software with concern as the main unit of reuse. Here, we specifically

[1] www.osgi.org
[2] www.kevoree.org
[3] If you like spa, just pronounce it so!

target security with a system of aspect-oriented security design patterns which can be specified using RAM. Roughly speaking, a developer could use SoSPa as an extensible library like a programmer would reuse generic classes in Java/.Net programming libraries. More than that, SoSPa and our MDS framework based on it provide means for a developer to systematically select the right security design patterns for the right job. Technically, the system of security design patterns consists of an extensible set of security concerns (e.g. authentication, authorization, encryption, etc.) which can fulfill the security objectives (e.g., confidentiality, integrity, availability, privacy). Each security concern is composed of a set of aspect-oriented security design patterns that realizes the security concern. The meta-info about interrelationships/relations among security design patterns are well specified within the system of them, i.e. by using an extended feature model. For example, some security design patterns could complement one another, or exclude each other. Moreover, each security design pattern also contains other meta-info describing the side effects of its adoption on other non-functional quality criterion, e.g. performance, usability, etc. All these meta-info are useful for analysis of the trade-off among alternatives which leads to a thoughtful decision on systematically selecting the right security design patterns for the job. As we target a full MDS approach, the SoSPa is proposed to be built on a meta-model which is part a full *Model-Driven Software Development* framework [6]. Our framework allows the selected security design patterns to be automatically composed with the target system model. The woven secure system model can then be used for (partial) code generation, including (configured) security infrastructures.

5 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 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 [9]. Mutation analysis operates by introducing artificial defects called mutants into the artifacts of the program under investigation. Our approach in [9] 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 into delegation policies and thus, enable mutation testing. There is still more work to be done to validate our idea in [9].

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.

6 Conclusion and Future Work

In this paper, we have presented a late-PhD work on MDS with *modularity* and *reusability* for secure systems development. The first main contribution of this work is a systematic literature review of MDS [5], and a book chapter on advances in MDS [4]. This literature review module is more or less done even though we are still working on a journal version of the review paper. Secondly, we have proposed a modular approach for modularity and dynamic adaptation of secure systems, i.e. model-driven adaptive delegation [7], [8]. But there is still more work to be done for dealing with multiple security concerns. Towards the third main contribution, we have been working on a MDS approach based on a library-like *System of generic aspect-oriented Security design Patterns* in which security design patterns are collected, specified as reusable aspect models to form a coherent system of them that guides developers in systematically selecting the right security design patterns for the right job [6]. And last but not least, the final main contribution is about model-based security testing. Our work on testing delegation policy enforcement via mutation analysis [9] is at the beginning. We will continue working on that. For validating the approach presented in [7] and [8], we have used three different case studies (LMS, VMS, and AMSM). For the exploratory approach in [6], we plan to validate our ideas using the case study CCCMS described in [3], and target three main security concerns, i.e. authentication, authorization, and cryptography, with interrelationships among them. In the end, we also would like to have an industrial case study for evaluating our exploratory approach presented in [6].

References

1. O. Alam, J. Kienzle, and G. Mussbacher. Concern-oriented software design. In *Model-Driven Engineering Languages and Systems*. 2013.
2. J. Bezivin. Model driven engineering: An emerging technical space. *GTTSE*, 2006.
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, J. Klein, and Y. Le Traon. Model Driven Security with a System of Aspect-Oriented Security Design Patterns. In *VAO workshop*, 2014, to appear.
7. P. H. Nguyen, G. Nain, J. Klein, T. Mouelhi, and Y. Le Traon. Model-Driven Adaptive Delegation. In *AOSD*, 2013.
8. 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.
9. 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.
10. K. Yskout, T. Heyman, R. Scandariato, and W. Joosen. A system of security patterns, 2006.