Model-based Security Analysis and Applications to Security Economics (Invited Talk)[1]

Jan Jürjens[1,2], Amir Shayan Ahmadian[1]
[1] *Software Engineering, Dep. of Computer Science, TU Dortmund, Dortmund, Germany*
[2] *Fraunhofer Institute for Software and Systems Engineering ISST, Dortmund, Germany*
{jan.jurjens,shayan.ahmadian}@cs.tu-dortmund.de

Keywords: Model-based Software Development, IT Security Analysis, Security Economics.

Abstract: In this invited presentation, we give an overview on a soundly based approach to Secure Software Engineering based on the UML extension UMLsec. More specifically, one main current focus is the automated, formally based analysis of software artefacts against security requirements. This is motivated by the observation that the current state of security engineering in practice is far from satisfactory.

The goal is thus to start with the actual industrial engineering methods of security-critical software-based systems, to identify problems which are practically amenable to tool-supported, formally sound analysis methods, and to try to solve these problems using these methods. An important objective is to ensure that these analysis methods can actually be used in practice by keeping the additional overhead in using them bounded: First, they take as input artefacts which are already available in current industrial software development (such as UML models and program source code) and do not have to be constructed just to perform the analysis. Second, the tools should be reasonably easy to use and have a strong emphasis on automation.

We also present results from some recent work on applying model-based security analysis to the analysis of economic aspects of securing critical infrastructures.

1 INTRODUCTION

1.1 A need for security

Modern society and modern economies rely on infrastructures for communication, finance, energy distribution, and transportation. These infrastructures depend increasingly on networked information systems. Attacks against these systems can threaten the economical or even physical well-being of people and organizations. There is widespread interconnection of information systems

[1] This work was partially supported by funding from the project “SECONOMICS: Socio economics meets security”, an Integrated Project supported by the European Commission, Seventh Framework Programme.

via the Internet, which is becoming the world's largest public electronic marketplace, while being accessible to untrusted users. Attacks can be waged anonymously and from a safe distance. If the Internet is to provide the platform for commercial transactions, it is vital that sensitive information (like credit card numbers or cryptographic keys) is stored and transmitted in a secure way.

1.2 Problems

Developing secure software systems correctly is difficult and error-prone. Many flaws and possible sources of misunderstanding have been found in protocol or system specifications, sometimes years after their publication. Many vulnerabilities in deployed security-critical systems have been exploited, sometimes leading to spectacular attacks. For example, as part of a 1997 exercise, an NSA hacker team demonstrated how to break into U.S. Department of Defense computers and the U.S. electric power grid system, among other things simulating a series of rolling power outages and 911 emergency telephone overloads in Washington, D.C., and other cities.

1.3 Causes

Firstly, security requirements are intrinsically subtle, because they have to take into account interaction of the system with motivated adversaries that act independently. Thus some security mechanisms, for example security protocols, are notoriously hard to design in a correct way, even for experts. Also, a system is only as secure as its weakest part or aspect.

Secondly, risks are very hard to calculate because of a positive reinforcement in the failure occurrence rates over repeated system executions: security-critical systems are characterized by the fact that the occurrence of a failure (that is, a successful attack) at system execution time dramatically increases the likelihood that the failure will occur during any following execution of a system using the same error-prone part of the design. For some attacks (for example against web sites), this problem is made worse by the existence of a mass communication medium that is currently largely uncontrolled and enables fast distribution of exploit information (again, the Internet).

Thirdly, many problems with security-critical systems arise from the fact that their developers, who employ security mechanisms, do not always have a strong background in computer security. This is problematic since in practice, security is compromised most often not by breaking dedicated mechanisms such as encryption or security protocols, but by exploiting weaknesses in the way they are being used. As an example, the security of Common Electronic Purse Specifications (CEPS) transactions depends on the fact that in the immediately envisaged scenario (use of the card for purchases in shops) it is not feasible for the attacker to act as a relay between an attacked card and an attacked terminal (Jürjens, 2001). However, this is not explicitly stated, and it is furthermore planned to use CEPS over the Internet, where an attacker could easily act as such a relay (Jürjens, Wimmel, 2001).

Thus it is not enough to ensure correct functioning of used security mechanisms; they cannot be "blindly" inserted into a security-critical system, but the overall system development must take security aspects into account. Building trustworthy components does not suffice, since the interconnections and interactions of components play a significant role in trustworthiness.

Lastly, while functional requirements are generally analyzed carefully in systems development, security considerations often arise after the fact. Adding security as an afterthought, however, often leads to problems. Also, security engineers get less feedback about the secure functioning of the developments in practice, since security violations are often kept secret in fear of harm for a company's reputation. Ad hoc development has lead to many deployed systems that do not satisfy relevant security requirements. Thus a sound methodology supporting secure systems development is needed (Fernández-Medina, Jürjens, Trujillo, and Jajodia, 2009).

2 MODEL-BASED SECURITY ENGINEERING

2.1 Secure Software Engineering

The objective of our research is a soundly based approach to Secure Software Engineering. More specifically, one main current focus is the automated, formally based analysis of software artefacts against security requirements. This is motivated by the observation that the current state of security engineering in practice is far from satisfactory. The goal is thus to start with the actual industrial engineering methods of security-critical software-based systems, to identify problems which are practically amenable to tool-supported, formally sound analysis methods, and to try to solve these problems using these methods. An important objective is to ensure that these analysis methods can actually be used in practice by keeping the additional overhead in using them bounded: First, they take as input artefacts which are already available in current industrial software development (such as UML models and program source code) and do not have to be constructed just to perform the analysis. Second, the tools should be reasonably easy to use and have a strong emphasis on automation.

2.2 Model-based Security

So far, part of our research has focused on the specification level. There, starting from the practical software security point-of-view and based on experiences from industrial application projects, an extension of the industrial specification notation UML (Unified Modeling Language) has been defined which is used to include static and behavioural security requirements into models of software systems (Jürjens, 2005). To facilitate the treatment of complex security concepts, an aspect-oriented approach to modeling security requirements in this context is investigated in (Houmb, Georg, France, Bieman, Jürjens, 2005) and (Houmb, Georg, Jürjens, France, 2006).

2.3 Formal Verification

Depending on the security problems to be addressed, different automated, formally-based tools are used to establish practical security requirements (Höhn, Jürjens, 2008). More concretely, this includes automated theorem provers for first-order logic (such as SPASS and Setheo), evaluation using Prolog , and model-checking using Spin and SMV. This work is based on a formal execution semantics for UML diagrams (including Statecharts and Sequence Diagrams). The verification tools are integrated into a modular, extensible UML verification tool framework available as open-source. In particular, reasoning techniques for security requirements have been examined, such as refinement (proposing a solution for the "refinement paradox" well-known in the formal methods for security community), protocol layering , and composability issues (Jürjens, 2000).

2.4 Source Code Analysis

More recently, we have applied automated verification for security requirements to the source code level. This has been supported by a joint project on Verifying Implementations of Security Protocols in C with Microsoft Research (Cambridge) (Aizatulin, Gordon, and Jürjens, 2011). To support the source code analysis, we have also performed research in the topic of program comprehension (Ratiu, Feilkas, and Jürjens, 2008).

2.3 Testing

A link between formal and industrial verification techniques is made with work on automated model-based test-sequence generation for security-critical systems against security requirements using constraint-solving techniques (Jürjens, Wimmel, 2001). An empirical study on applying this approach was published in (Jürjens, 2008).

2.5 Industrial Applications

A lot of our work has been done in cooperation with industrial partners including BMW, HypoVereinsbank, O2 (Germany), Infineon, Deutsche Telekom, Munich Re, IBMRational, Deutsche Bank, Allianz, and others. For example, a collaboration with O2 (Germany) on applying model-based security engineering techniques to mobile communication systems was performed. An application of model-based security engineering to an intranet information system at BMW was carried out. As an application of our research, in the context of the Verisoft-project funded by the German Ministry of Education and Research (BMBF), a practical, formally based security engineering approach was developed and applied at the hand of a biometric authentication system developed by Deutsche Telekom. Of particular interest are also the application domains of smart-card based systems and electronic payment systems.

2.6 Security Configurations

The second main difficulty with security-critical software in practice, besides a correct enforcement of security requirements during development, is the correct configuration of security-critical applications. For example, security permissions in the financial sector (such as the permission for an employee to grant a credit) have to obey general security policy rules, such as the "four-eye-principle" (i.e., large credits have to be confirmed by two employees). With large permission sets (60,000 in the case of the bank that motivated this particular research, the HypoVereinsbank), dynamic changes (for example, vacancy substitutions), and complicated permissions (such as delegation of rights), a continuous manual inspection is infeasible. This motivates our work on developing a logic-based analysis method for security-critical permission configuration, and the associated tool-support in Prolog.

2.7 Complexity-theoretical Soundness of Symbolic Analysis

Another difficulty with security-critical systems is that adversaries will generally try to attack the weakest link in the system, and in particular exploit any mismatches between security requirements and guarantees on different conceptual or physical levels of a system. Therefore, the approach taken has to be as seamless as possible, which is another focus of research. As one example, in joint work with Martin Abadi, it is shown that, under suitable assumptions, the usual abstract approach to security analysis using formal methods is faithful with respect to the more fine-grained complexity-theoretical security models for cryptography.

2.8 Dependability

In other strands of research, the methods and insights gained in the security engineering field are being transferred to other application domains with sophisticated non-functional requirements, such as dependable, safety-critical, real-time, or performance-sensitive systems.

3 APPLICATIONS TO SECURITY ECONOMICS

In recent work within the EU-project “SECONOMICS: Socio economics meets security”, we have been developing tools for the analysis of economic aspects of securing critical infrastructures.

The project SECONOMICS develops approaches and software tools to analyze socioeconomical aspects of information security, especially in the context of cyber-physical systems. The developed models have been validated onto three use cases: the international air transport, urban transportation (TMB in Barcelona) and the critical national infrastructure (energy and gas networks of National Grid UK and US). The developed approaches incorporate risk analysis with economical aspect to develop software tools, which aid the decision makers. The contribution of Fraunhofer ISST and TU Dortmund focusses on the model-based analysis of IT security risks. In particular, a generalized policy "toolkit" that will assist decision makers in identifying and reacting coherently (within the appropriate social context) to future and emerging threats that may arrive long after the project has been developed.

REFERENCES

Aizatulin, M., Gordon, A.D., Jürjens, J., 2011. Extracting and Verifying Cryptographic Models from C Protocol Code by Symbolic Execution. In: *18th ACM Conference on Computer and Communications Security (CCS 2011)*, pp. 331-340.

Breu, R., Burger, K., Hafner, M., Jürjens, J., Popp, G., Wimmel, G., Lotz, V., 2003: Key Issues of a Formally Based Process Model for Security Engineering. In: *Proceedings of the 16th International Conference on Software & Systems Engineering and their Applications (ICSSEA 2003)*

Fernández-Medina, E., Jürjens, J., Trujillo, J., Jajodia, S., 2009. Model-Driven Development for secure information systems, *Information and Software Technology* 51 (5), 809-814

Höhn, S., Jürjens, J., 2008: Rubacon: automated support for model-based compliance engineering. In: *30th International Conference on Software Engineering (ICSE 2008)*, ACM. pp. 875-878

Houmb, S.H., Georg, G., France, R.B., Bieman, J.M., Jürjens, J., 2005. Cost-Benefit Trade-Off Analysis Using BBN for Aspect-Oriented Risk-Driven Development. In: *10th Intern. Conference on Engineering of Complex Computer Systems (ICECCS 2005)*, IEEE, pp. 195–204.

Houmb, S.H., Georg, G., Jürjens, J., France, R.B., 2006. An Integrated Approach to Security Verification and Security Solution Design Trade-off Analysis. In: H. Mouratidis (ed.), *Integrating Security and Software Engineering: Advances and Future Vision*, Idea Group, pp. 190–219. Invited chapter.

Jürjens, J., 2000. Secure information flow for concurrent processes. In: *Concur 2000, International Conference on Concurrency Theory*, Pennsylvania, LNCS 1877, Springer-Verlag.

Jürjens, J., 2001. Modelling Audit Security for Smart-Cart Payment Schemes with UML-SEC. In: *IFIP TC11 Sixteenth Annual Working Conference on Information Security (IFIP/Sec'01)*, Kluwer, pp. 93-108

Jürjens, J., 2005. *Secure Systems Development with UML*, Springer-Verlag, Heidelberg.

Jürjens, J., Wimmel, G., 2001. Security Modelling for Electronic Commerce: The Common Electronic Purse Specifications. In: *First IFIP Conference on E-Commerce, E-Business, E-Government (I3E 2001)*, Kluwer, pp. 489-506.

Jürjens, J., Wimmel, G., 2001. Formally Testing Fail-Safety of Electronic Purse Protocols. In: *16th IEEE International Conference on Automated Software Engineering (ASE 2001)*, IEEE Computer Society, pp. 408-411.

Jürjens, J., 2008. Model-based Security Testing Using UMLsec: A Case Study. *Electr. Notes Theor. Comput. Sci. 220(1)*: 93-104

Ratiu, D., Feilkas, M., Jürjens, J., 2008: Extracting domain ontologies from domain specific APIs, In: *12th European Conference on Software Maintenance (CSMR 2008)*, IEEE, pp. 203–212.