Formal Development and Verification of Security-Critical Systems with UML (Position Paper)

Jan Jürjens*

Computing Laboratory, University of Oxford, GB

Vers. 2.IV.01 of a paper at AVoCS01. Current version and other material: www.jurjens.de/jan .

Abstract. We sketch ongoing work on using a formal fragment of the object-oriented modelling language UML (Unified Modeling Language) in development and verification of secure systems.

We sketch some current challenges for the formal (in particular automated) development and verification of (in particular) security-critical systems and note how some of our work tries to address these.

When trying to use formal (and in particular automated) techniques to develop or verify real-life security-critical systems whose specifications can easily consist of several hundred pages or more (e.g. the Common Electronic Purse Specifications (CEPS) considered in [Jür01c]) one faces a number of problems in scaling up existing approaches:

(1) Usually a formal specification of the system is not available. Constructing it requires expert knowledge and can be very time-consuming. Currently a large part of effort both in verifying and in implementing specifications is wasted since these are often formulated imprecisely and unintelligibly [Pau98].

(2) It is usually only feasible to construct the specification of a small security-critical part of the system (e.g. security protocols [Low96, LR97, RSG+01, SCW00], where formal analysis has in fact been quite successful). The boundaries of these components with the rest of the system need to be carefully examined, e.g. wrt implicit assumptions on the system context or the underlying physical layer [Gol00, Aba00]. In practice, most attacks do not address vulnerabilities in dedicated mechanisms (such as security protocols), but in the way these are employed in the system context [And01]. An example is given in [JW01a], where a vulnerability arises from the intended use of the CEPS payment scheme over the Internet.

* http://www.jurjens.de/jan – jan@comlab.ox.ac.uk - Supported by the Studienstiftung des deutschen Volkes and the Computing Laboratory.

(3) Stepwise development and modular reasoning are hindered by the fact that many security properties are not preserved under refinement and not compositional.
(4) Even when specifying small components, one often has to give a simplifying account. In particular, in symbolic modelling cryptographic operations one abstracts from the possibility that an adversary may simply guess secrect values.
(5) Often, vulnerabilities arise from flaws in the implementation rather than the design.

We try to address some of these issues in previous, current and future work as follows:

(1) As a specification language we propose to use a formal fragment of UML [OMG99], extended with security-relevant primitives using the UML extension mechanisms [Jür01e, Jür01c, Jür01f, Jür01b]. As UML is the de-facto industry-standard in object-oriented modelling, in some cases a specification of the system in it may already be available, or at least there may be knowledge available on the side of the developers how to construct the specification (additionally, there is work on deriving Java code from UML specifications [EHSW99] and vc. vs. [KG01], narrowing the gap between specification- and software-based verification). For mechanical verification one may take advantage of work giving UML diagrams a formal semantics[1] in CSP, allowing to use FDR2 [BD00, Cri01], and of other work towards tool-support for UML [Ste00].
(2) Through its different kinds of diagrams, UML allows to take different views on the system (e.g. on the physical layer). This allows expressing properties of and reasoning about boundaries of critical components.
(3) For stepwise development and modular reasoning, one may make use of work in [Jür01d, Jür01a] giving a notion of secrecy preserved under refinement and giving compositionality results in the setting of a simple specification language (the current work aims to transport these results into the setting of UML).
(4) The question to what extent the formal methods approach to modelling cryptographic primitives is sound is addressed in [AJ01], where a translation from a symbolic semantics of a specification language involving cryptographic operations to a complexity-theoretical one is given. Again these results apply (after adjusting the differences in the models) in particular to a formal core of UML.

[1] For some background on giving UML a formal semantics cf. e.g. [EFLR99].

(5) One may address vulnerabilities arising from flaws in the implementation by deriving test-sequences for implementations from formal specifications (e.g. following ideas from [JW01b], where this is done using a formal notation close to UML).

Beyond verification, one may also employ UML in requirements capture for security protocols [Low00] or to encapsulate rules of prudent engineering for secure systems [Jür01b].

In this talk we report on work towards verifying security properties in a formal core of UML in a compositional way.

References

[Aba00] M. Abadi. Security protocols and their properties. In F. Bauer and R. Steinbrueggen, editors, *Foundations of Secure Computation*. IOS Press, 2000.

[AJ01] M. Abadi and Jan Jürjens. Formal eavesdropping and its computational interpretation, 2001. Submitted.

[And01] R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, 2001.

[BD00] C. Bolton and J. Davies. Activity graphs and processes. In *Integrated Formal Methods*, LNCS. Springer, 2000.

[Cri01] C. Crichton. UML statecharts and CSP, 2001. In preparation.

[EFLR99] A. Evans, R. France, K. Lano, and B. Rumpe. The UML as a formal modeling notation. In J. Bezivin and P.-A. Muller, editors, *The Unified Modeling Language - Workshop UML'98: Beyond the Notation*, LNCS. Springer, 1999.

[EHSW99] G. Engels, R. Hücking, S. Sauer, and A. Wagner. UML collaboration diagrams and their transformation to Java. In *[FR99]*, pages 473–488, 1999.

[FR99] R. France and B. Rumpe, editors. *Second International Conference on the Unified Modeling Language - UML'99*, volume 1723 of *LNCS*. Springer, 1999.

[Gol00] Dieter Gollmann. On the verification of cryptographic protocols - a tale of two committees. In *Workshop on Security Architectures and Information Flow*, volume 32 of *Electronical Notes in Theoretical Computer Science*, 2000.

[Huß01] H. Hußmann, editor. *Fundamental Approaches to Software Engineering (FASE/ETAPS, International Conference)*, LNCS. Springer, 2001.

[Jür01a] Jan Jürjens. Composability of secrecy. In *International Workshop on Mathematical Methods, Models and Architectures for Computer Networks Security (MMM-ACNS 2001)*, LNCS, St. Petersburg, 21-23 May 2001. Springer.

[Jür01b] Jan Jürjens. Encapsulating rules of prudent security engineering. In *International Workshop on Security Protocols*, LNCS. Springer, 2001.

[Jür01c] Jan Jürjens. Modelling audit security for smart-card payment schemes with UMLsec. In P. Paradinas, editor, *IFIP/SEC 2001 – 16th International Conference on Information Security*, Paris, 11–13 June 2001. Kluwer.

[Jür01d] Jan Jürjens. Secrecy-preserving refinement. In *Formal Methods Europe (International Symposium)*, LNCS. Springer, 2001.

[Jür01e] Jan Jürjens. Towards development of secure systems using UMLsec. In *[Huß01]*, 2001. Also OUCL TR-9-00 (Nov. 2000), http://web.comlab.ox.ac.uk/oucl/publications/tr/tr-9-00.html.

[Jür01f] Jan Jürjens. Transformations for introducing patterns – a secure systems case study. In *WTUML: Workshop on Transformations in UML (ETAPS 2001 Satellite Event)*, Genova, 7 April 2001.

[JW01a] Jan Jürjens and Guido Wimmel. Security modelling for electronic commerce: The Common Electronic Purse Specifications. Submitted, 2001.

[JW01b] Jan Jürjens and Guido Wimmel. Specification-based testing of firewalls. Submitted, 2001.

[KG01] R. Kollmann and M. Gogolla. Capturing dynamic program behaviour with UML collaboration diagrams. In *5th European Conference on Software Maintenance and Reengineering*. IEEE, 2001.

[Low96] G. Lowe. Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR. *Software Concepts and Tools*, 17:93–102, 1996.

[Low00] G. Lowe. External specifications for security protocols, 2000. Draft.

[LR97] G. Lowe and B. Roscoe. Using CSP to detect errors in the TMN protocol. *IEEE Transactions on Software Engineering*, 23(10), 1997.

[OMG99] UML Revision Task Force, OMG. UML Specification 1.3. Available at http://www.omg.org/uml, 1999.

[Pau98] L. Paulson. Inductive analysis of the Internet protocol TLS (transcript of discussion). In B. Christianson, B. Crispo, W.S. Harbison, and M. Roe, editors, *Security Protocols – 6th International Workshop*, number 1550 in LNCS, page 13 ff., Cambridge, UK, April 1998.

[RSG+01] P. Ryan, S. Schneider, M. Goldsmith, G. Lowe, and B. Roscoe. *Modelling and Analysis of Security Protocols*. Addison Wesley, 2001. (to be published).

[SCW00] S. Stepney, D. Cooper, and J. Woodcock. *An Electronic Purse: Specification, Refinement, and Proof*. Oxford University Computing Laboratory, 2000. Technical Monograph PRG-126.

[Ste00] P. Stevens. Advanced tools for UML: now and in the future, 2000. Tutorial given at UML2000.