How to capture, model, and verify the knowledge of legal, security, and privacy experts: a pattern-based approach

Luca Compagna
SAP Research
luca.compagna@sap.com

Paul El Khoury
SAP Research
paul.el.khoury@sap.com

Fabio Massacci
University of Trento
massacci@dit.unitn.it

Reshma Thomas
University of Leuven
reshma.thomas@law.kuleuven.be

Nicola Zannone
University of Trento
zannone@dit.unitn.it

ABSTRACT

Laws set requirements that force organizations to assess the security and privacy of their IT systems and impose the adoption of the implementation of minimal precautionary security measures. Several frameworks have been proposed to deal with this issue. For instance, purpose-based access control is normally considered a good solution for meeting the requirements of privacy legislation. Yet, understanding why, how, and when such solutions to security and privacy problems have to be deployed is often unanswered.

In this paper, we look at the problem from a broader perspective, accounting for legal and organizational issues. Security engineers and legal experts should be able to start from the organizational model and derive from there the points where security and privacy problems may arise and determine which solutions best fit the (legal) problems that they face. In particular, we investigate the methodology needed to capture security and privacy requirements for a Health Care Centre using a smart items infrastructure.

Categories and Subject Descriptors: D.2.13 [**Software Engineering**]: Reusable Software.

General Terms: Design, Security, Legal Aspects.

Keywords: Security & Privacy Patterns, Legal Requirements, Organization, Pattern Validation, Health Care.

1. INTRODUCTION

Protecting sensitive information is critical to the success of any organization. The "security" reputation of a corporation is now becoming an important asset as more and more customers are basing their choice of suppliers also on the evidence of privacy and security practices.

Market pressure is not the only force. Privacy is a highly regulated area in Europe. There are strict regulations in place within the European Union that impose rules for the collection and processing of personal data (e.g., Data Protection Directive, 95/46/EC2). Therefore, organizations that handle personal data cannot escape the obligation of implementing these regulations in their IT infrastructure.

Unfortunately, it has always been difficult to bridge the gap between legal language and computer language, more importantly when legal obligations have to be converted into requirements to be enforced by the IT infrastructure.

An answer as to how to solve these problems can be provided by the patterns approach (e.g., [3, 4, 14, 15]) that can capture the expertise of security and legal experts and make it available for designers that might not have a solid security and legal background. Yet, such an approach is often tainted by two limitations: the exclusive focus on the software system, and the informal outlook of practitioners.

The first limitation makes it difficult to capture all legal requirements. Legal requirements are expressed on organizations as a whole and refer also to humans and human-run procedures (as opposed to computer-run procedures).

The second limitation affects the reliability of the patterns. Though a pattern is the result of a collaboration between security and legal experts, this human collaboration is subject to errors. Apparently right but actually wrong patterns might erode the acceptance of security patterns by application designers. One of the most effective countermeasures is represented by the application of formal methods approaches as a way to establish proof-of-concept evidence of the soundness of the solutions proposed.

This would not seem to be a major problem as the use of formal methods and logic to capture legal reasoning has a long tradition: from the early paper by Kanger [8] through the papers by Sergot [10], some of them on the very pages of ICAIL exactly 20 years ago [1], till more recent work [2]. Although logic is very good means to provide a precise model of a system, it is ineffective in communicating such a model [11] to legal and system experts who have no background in formal methods.

In this paper we show how we can combine logic, agent-oriented methodologies and practical security engineering to capture and model practical legal patterns in a concrete industrial smart-items domain. The starting idea is to use a *graphical* and easy-to-use modeling framework, in the sequel referred to as SI* [12], to support system designers lacking deep security and legal knowledge to design and deploy systems where security and privacy obligations are enforced in accordance with regulations.

Next, we assess the suitability of this framework for critical security and privacy issues at the organizational level

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICAIL'07, June 4-8, 2007, Palo Alto, CA USA.
Copyright 2007 ACM 978-1-59593-680-6/07/0006/ ...$5.00.

of a health-care scenario based on a smart items infrastructure. We show how to capture and generalize solutions to these issues into security and privacy patterns. These patterns have been validated through the automated reasoning technique presented in [6].

The security and privacy patterns described in this paper represent an excerpt of the library we are populating in the context of the EU SERENITY project.[1] This library of patterns should serve as a reference in the design and deployment of systems sensitive to security and privacy issues.

2. PRIMER ON PATTERNS

The pattern approach has been adopted into software engineering as a method for object-based reuse [5]. Security patterns are essentially security best practices presented in a template format. This format aids designers, who are not security experts, in identifying and understanding security concerns, and in implementing appropriate security measures.

Schumacher [14] applies the pattern approach to security problems by proposing a set of security patterns for the development process. Yoder and Barcalow [15] propose architectural patterns that can be applied when adding security to an application. Fernandez and Pan [4] describe patterns for the most common security models such as Authorization, Role-Based Access Control, and Multilevel Security. One of the main problems of these proposals is the lack of tools that validates patterns with respect to their expert knowledge. Another problem is the lack of a formal framework that supports the analysis of the security requirements and determines precisely the context in which a pattern can be applied. Finally, most proposals hardly make any reference to the legal theory on which they are based.

These issues are partially addressed by Cheng et al. [3]. They propose a template for security patterns that takes into account essential information that may not be necessary for design patterns but appears as mandatory in a privacy and security context. The research work of IBM in 2003 on patterns tackled security risks when designing the system [9]. The focus is on implementing efficient security and privacy solutions as an integral part of a business process and delivering value to customers by measuring risk. Yet, these patterns are only available at the workflow (as opposed to organization) level and are not validated.

A similar approach has been proposed by Mouratidis et al. [13]. Their security pattern language includes a number of patterns supporting the communication between agents belonging to different agencies, the authentication of agents, etc. Unfortunately this language lacks fundamental concepts needed to capture security aspects of organizations and a formal framework for pattern validation.

3. SMART ITEMS FOR HEALTH CARE

This section presents a smart items infrastructure for health care used as a running example through the paper. The scenario focuses on Bob, a 56 year old widowed man who has been recently discharged from hospital after a cardiac arrest. Bob's health conditions need to be monitored 24 hours a day and for this purpose Bob signed a legal contract with the Health Care Centre (HCC), a provider of medical services. The HCC equipped Bob with a smart T-shirt that incorporates a motion sensor providing an alert as soon as Bob becomes passive for two minutes and monitoring devices that regularly measure his heart rate, blood pressure, body temperature, etc. The smart T-shirt conveys all the measured data to an e-health terminal that allows Bob to promptly communicate medical data to his doctor via the Monitoring and Emergency Response Centre (MERC), a department of the HCC, that is responsible for receiving and handling patient requests for assistance.

A service offered by the HCC is the delivery of medicines at the patient's house. Since Bob feels weak, he decides to exploit this service and so sends a request to the MERC. In turn, the MERC appoints a social worker, Alison, for this task. Alison receives a message from the MERC on her e-health terminal to get the medicine to be delivered to Bob (this message contains the medical prescription that was included in the request sent by Bob to the MERC). She acknowledges the request and goes to the pharmacy. After a successful message exchange between Alison's e-health terminal and the pharmacist's computer, the medicine is given to Alison who proceeds in delivering it to Bob.

Bob has also subscribed to an experimental programme that aims, through a sensor network working behind the scenes, to enhance his daily life at home and to provide additional data for better monitoring the status of his health. The Sensor Network Provider (SNP) is a legal subcontractor of the HCC, which has installed a sensor network in Bob's house. The SNP is also responsible for maintaining the sensor network in Bob's house.

It is clear that the smart items infrastructure sketched above has to integrate a variety of sensors and devices, as well as humans and organizations accessing sensitive data. In this context security and privacy requirements need to comply with legal regulations.

4. A LEGAL PERSPECTIVE

As the above example indicates, the HCC needs to collect, store and process a wide amount of personal data to provide e-health services. This introduces several security and privacy issues in the ICT landscape. Many countries are aware of these issues and have promulgated legislation establishing special requirements with respect to personal data. We focus here on the common basis provided by the European Union Data Protection Directive, 95/46/EC (hereinafter referred to as the Directive), which provides rules governing collection, use, storage and distribution of personal data.

The Directive defines the actors that are involved in the processing of personal data. The Data Subject is defined as an identifiable person who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity. In our scenario, Bob plays the role of data subject. The Data Controller is the natural or legal person, public authority, agency or other body, which determines the purposes and means of the data processing. Moreover, he is the person who has to ensure that all principles as regards data quality under the Directive are observed. In our scenario, the HCC acts as data controller. The Data Processor is defined as a natural or legal person, public authority, agency or any other body which processes personal data on behalf of the controller.

[1] EU-IST-IP 6th Framework Programme – SERENITY 27587 – http://www.serenity-project.org

In our scenario, the SNP plays the role of data processor. The relationship between the controller and the processor must be governed by contract or legal agreement binding both parties to ensure that the processor shall act only on instructions from the controller and that the processor is bound by the same obligations as the controller.

Whenever any of the personal information of the Data Subject is outsourced to a third party, the notion of consent gains more relevance since this transfer was not foreseen in the initial contract and hence there is an additional responsibility on the Data Controller to explain the nature and consequences of such outsourcing to the Data Subject, in order to obtain his informed consent.

EXAMPLE 1. *The HCC needs to collect information about Bob to provide certain services and outsources the task of collecting information about Bob's position, movement and surrounding environment to the SNP. However, the HCC cannot outsource this task without the explicit consent of Bob, who should have authorized such outsourcing after being informed by the HCC about the nature and consequences of such collection. Thus, the obligations imposed on the SNP with regard to the processing of personal data will be the same as that which was initially imposed on the HCC.*

Other security and privacy issues are dealt with by national legislation. For instance, according to art. 1228 c.c. of Italian Civil Code ("Liability for acts of auxiliaries") data controllers who rely on third parties in the execution of data processing are also liable for their malicious, fraudulent, or negligent acts, unless otherwise intended by the parties. Thereby, employers need warranties that executors *do not repudiate* the commitment they take charge of.

EXAMPLE 2. *The MERC is responsible for the delivery of medicines to patients. The MERC appoints Alison to deliver the medicine to Bob. Although the MERC relies on another actor in the performance of the obligation, it is still liable for the failure of the service. Thus, the MERC wants warranties about the commitment so that it can defend its position.*

5. THE SI* MODELING FRAMEWORK

To understand why, how, and where solutions to security and privacy problems have to be deployed, system designers must model the goals, assets and trust relationships of the stakeholders of the socio-technical system as a whole.

Among various proposals, we have chosen Secure Tropos [6], an agent-oriented security requirements engineering methodology. Secure Tropos adopts the SI* modeling language [12] for the acquisition of requirements. SI* uses the concepts of actor, goal, task, and resource. An *actor* is an intentional entity that performs actions to achieve goals. A *goal* is a strategic interest of an actor. A *task* specifies a particular sequence of actions that can be executed to achieve a goal. A *resource* represents a physical or an informational entity. Every actor is defined along with a set of objectives, entitlements, and capabilities. *Objectives* are goals intended to be achieved, tasks intended to be executed or resources required by the actor. *Entitlements* are goals, tasks and resources controlled by the actor. Finally, *capabilities* are goals, tasks, and resources that an actor is able to respectively achieve, execute, and furnish. SI* also supports the notions of *permission delegation* and *execution dependency* to model the transfer of entitlements and responsibilities from an actor (called *delegator* or resp. *depender*) to another actor (called *delegatee* or resp. *dependee*). Moreover, the language adopts the notions of *trust of permission* and *trust of execution* to model the expectation of one actor (the *trustor*) about the behavior of another actor (the *trustee*).

From a methodological perspective, Secure Tropos is based on the idea of building a model of the system that is incrementally refined and extended. Specifically, goal analysis consists of refining goals and eliciting new social relationships among actors. They are conducted from the perspective of single actors using means-end analysis, AND/OR decomposition, and contribution analysis.

The above constructs allow designers to capture the requirements model of organizations together with their IT systems. In the graphical representation of this model, objectives, entitlements, and capabilities are represented using request (**R**), ownership (**O**), and provide (**P**) relations, respectively. Permission delegations are represented with edges labeled by (**Dp**) and execution dependency with edges labeled by (**De**). Figure 1 shows the requirements model of our health care scenario. Bob, who is the legitimate owner of his data, delegates the permission to manage them to the HCC. The HCC refines this goal into "provide patient data" and "collect patient data". These goals are in turn refined until the HCC can achieve them or there exist actors that can take care of them. For instance, the HCC appoints the SNP to collect environmental data.

6. PRIVACY AND SECURITY PATTERNS

Security and privacy patterns aim at capturing collective experience in the security and legal domains and make this know-how available and exploitable for application designers. Each pattern is a triple $\langle Context, Requirement, Solution \rangle$ where the *Context* defines the situation and conditions of applicability of the pattern, the *Requirement* is the expression of the need for a system to enhance its privacy or security level, and the *Solution* is given as an abstract formal model at the organizational level whose application guarantees proof-of-concept the achievement of the requirement.

In the rest of this section we describe two patterns addressing the legal requirements of Section 4 and we show how they apply into our health care scenario of Section 3.

6.1 Outsourcing Pattern

Outsourcing is the transfer of management control of business activities to an outside supplier. This business strategy is adopted to reduce costs, but has a strong impact on the security and privacy requirements of organizations. Several attempts have been made for defining outsourcing requirements [7]. From a privacy perspective, the organization to which data has been outsourced will be obliged by all data protection principles as mentioned above.

Context. The Data Controller outsources the execution of data processing to an outside Supplier for which Data Subject's personal data are needed. However, the Data Subject has signed a contract according to which only the Data Controller and assigned members of its organization are entitled to process his data. A SI* diagram representing the context is presented in Figure 2(a).

Requirement. The Supplier shall have the permission necessary to achieve outsourced data processing.

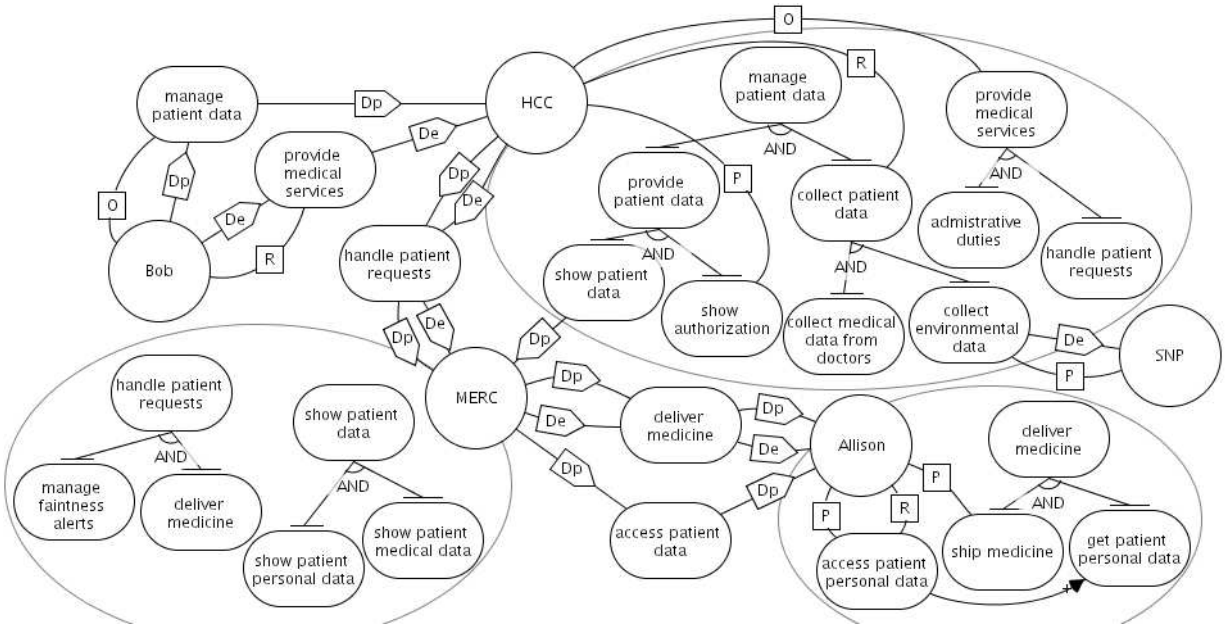
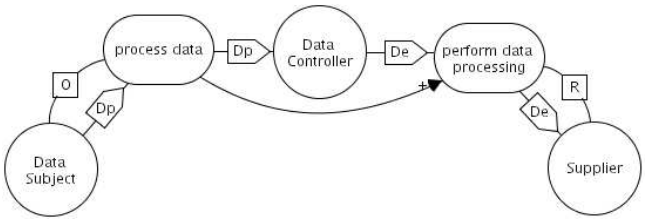
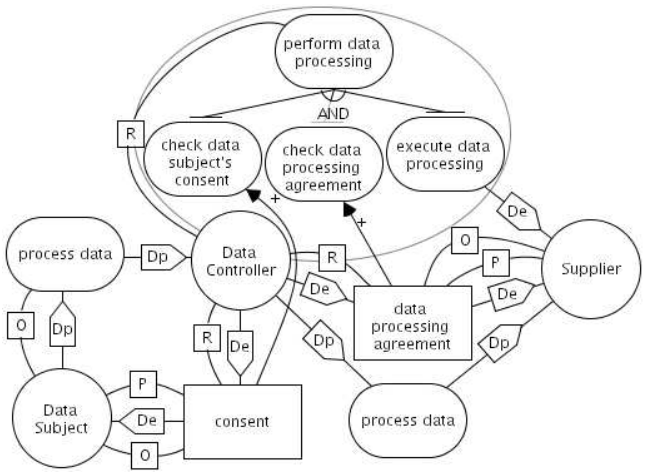


Figure 1: The SI* Model



(a) Context



(b) Solution

Figure 2: Outsourcing Pattern

Solution. Before the Data Controller can outsource the data processing to the Supplier, he has to obtain the consent of the Data Subject. The consent can be seen as a contract establishing what and how data will be processed by the Supplier. The Data Controller must also ensure, preferably by a written agreement, that the Supplier strictly follows all conditions relating to data processing that were imposed on him. A SI* diagram of the solution is shown in Figure 2(b).

Consequences. The pattern solves the problem of granting to the Supplier the permission necessary to perform outsourced data processing by assuring the Data Subject that data are processed according to the contract. At the same time, the application of the pattern introduces new issues. For instance, the Data Controller may want assurance that the Supplier does not repudiate the data processing agreement and the Data Subject does not repudiate the consent.

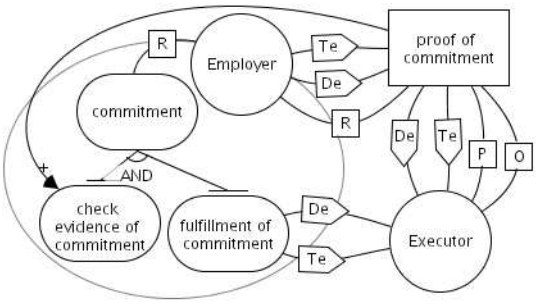
This pattern fits in our scenario. To effectively monitor Bob's health, the HCC (i.e., the Data Controller) needs the assistance of the SNP (i.e., the Supplier), which keeps track and reports all information relating to Bob's environment.

6.2 Non-Repudiation Pattern

To accomplish its daily tasks an organization exploits its social infrastructure by decomposing each task into subtasks that are then distributed/delegated among groups of actors having pre-defined relations. For those tasks and subtasks considered critical for the organization, a simple statement of delegation is not sufficient to release the responsibility of the task from one actor to another. In such cases, the employer might want a non-repudiable acknowledgment from the executor to have proof that the latter has explicitly taken the responsibility of executing the task.

Context. The Employer requests the achievement of a commitment and delegates its execution to the Executor, but the former has no warranties that the latter takes the responsibility of achieving the commitment. An SI* diagram representing the context is presented in Figure 3(a).
Requirement. The Employer shall have evidence that the Executor cannot repudiate his commitment.
Solution. The Employer refines the commitment into two sub parts. The first part is used to check the evidence about responsibilities taken by the Executor and the second represents the actual desire of fulfilling the commitment. To achieve the commitment, the Employer delegates the exe-

(a) Context



(b) Solution

Figure 3: Non-Repudiation Pattern

cution of the commitment to the Executor together with a request for a proof of commitment. Once received, this proof ensures the Employer that the Executor has taken the responsibility of the commitment. This establishes a trust relationship between them. In case the Executor fails to achieve the commitment, the Employer can rely on the proof of commitment. A SI* diagram of the solution is presented in Figure 3(b). For the sake of simplicity, we have assumed a trust relation (**Te**) between the Employer and the Executor for the proof of commitment. Actually, this relation "hides" the mechanism used to justify its presence in the model.

Consequences. The proof of commitment is very important; especially if the Executor does something wrong in the execution of the commitment, since it is the only means by which the Employer can discharge itself from any liability.

This pattern perfectly fits into our health care scenario where the medicine has to be delivered to the patient. The MERC (i.e., the Employer) delegates this task to Alison (i.e., the Executor). Applying the above pattern the MERC will dispose of a proof stating that Alison acknowledged the request and accepted the responsibility of executing it.

7. CONCLUSIONS

System designers are usually neither security nor legal experts, so they may have difficulties in deploying systems that comply with security and privacy requirements as defined in the current legislation. We have addressed this issue by presenting an approach based on patterns. The patterns proposed in this paper have been derived from a real case-study based on a Smart Items Infrastructure for health care systems and are part of a pattern library developed in the SERENITY project. We have used the formal framework presented in [6] for the validation of patterns. This framework can also assist system designers in the analysis of organizational security and privacy requirements and in the selection of the appropriate solution.

Currently, we are extending this library by considering other application domains to deploy patterns that (1) can be applied in different domains and (2) cope with others security and privacy issues than the ones discussed for the scenario presented here. We are also defining a pattern integration schema that will drive designers in the application of safe combinations of patterns where potential interferences and conflicts must be considered. Finally, we are developing a CASE tool[2] supporting the proposed framework.

Acknowledgments

This work was partially funded by the projects IST-FP6-IP-SERENITY, FIRB-TOCAI, IST-FP6-FET-IP-SENSORIA, and PAT-MOSTRO. We want to express our gratitude to all the members of the SERENITY project for all their feedback and for the very useful scientific discussions.

8. REFERENCES

[1] T. J. M. Bench-Capon, G. O. Robinson, T. W. Routen, and M. J. Sergot. Logic programming for large scale applications in law: A formalisation of supplementary benefit legislation. In *Proc. of ICAIL'87*, pages 190–198. ACM Press, 1987.

[2] T. J. M. Bench-Capon and G. Sartor. A model of legal reasoning with cases incorporating theories and values. *Artif. Intell.*, 150(1-2):97–143, 2003.

[3] K. D.M. and E. M.C. Final technical report: Security patterns for web application development. Technical report, 2002. Available at http://www.scrypt.net/

[4] E. Fernandez and R. Pan. A Pattern Language for Security Models. In *In Proc. of PLoP'01*, 2001.

[5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[6] P. Giorgini, F. Massacci, and N. Zannone. Security and Trust Requirements Engineering. In *FOSAD 2004/2005*, *LNCS 3655*, pages 237–272. Springer-Verlag, 2005.

[7] ISO. Quality Management Systems: Requirements. ISO 9001:2000, 2000.

[8] S. Kanger. Law and logic. *Theoria*, 38(3):105–132, 1972.

[9] S. Konrad, B. H. C. Cheng, L. A. Campbell, and R. Wassermann. Using security patterns to model and analyze security requirements. In *Proc. of RHAS'03*. IEEE Press, 2003.

[10] R. A. Kowalski and M. J. Sergot. Computer Representation of the Law. In *Proc. of IJCAI'05*, pages 1269–1270. Morgan Kaufmann, 1985.

[11] L. Lamport. How to write a long formula. *Formal Aspects of Comp.*, 6(5):580–584, 1994.

[12] F. Massacci, J. Mylopoulos, and N. Zannone. An Ontology for Secure Socio-Technical Systems. In *Handbook of Ontologies for Business Interaction*. The IDEA Group, 2007.

[13] H. Mouratidis, M. Weiss, and P. Giorgini. Security patterns meet agent oriented software engineering: a complementary solution for developing security information systems. In *In Proc. of ER'05*, 2005.

[14] M. Schumacher. *Security Engineering with Patterns: Origins, Theoretical Models, and New Applications*. Springer-Verlag, 2003.

[15] J. Yoder and J. Barcalow. Architectural Patterns for Enabling Application Security. In *In Proc. of PLoP'97*, 1997.

[2] A prototype is available at http://sesa.dit.unitn.it/sttool/