

SWAT: A Security Workflow Analysis Toolkit for Reliably Secure Process-aware Information Systems

Rafael Accorsi, Claus Wonnemann, Sebastian Dochow
Department of Telematics
University of Freiburg, Germany
{accorsi,wonnemann,dochow}@iig.uni-freiburg.de

Abstract—This paper reports on ongoing work on SWAT, a new toolkit for security workflow analysis. SWAT provides a platform for the realization and testing of well-founded methods to detect information leaks in workflows, both for the workflow certification and for audit based upon the execution traces. Besides presenting the SWAT’s functionality and high-level architecture, an example illustrates its operation.

Keywords—Security workflow analysis; Information flow control; Certification; Audit.

I. INTRODUCTION

Business process specifications (so-called *workflows*) are high-level pieces of software that describe automated procedures within an enterprise, e.g. open bank accounts and update patient records. A representative survey conducted in European companies shows that up to 70% of the workflows run fully automated [33]. Workflow adoption is expected to soar with the advent of cloud-computing and the deployment of distributed, configurable processes “as-a-service” [20].

Although workflows are used in mission-critical activities demanding strong security and privacy guarantees [10], there are currently *no* methods and specialized tool support for the thorough security workflow analysis. As a result, workflows deployed today may exhibit subtle control flow and data flow vulnerabilities which, if exploited, may compromise the whole enterprise information system [23]. As an example, information leaks could exist and be exploited, say, by an internal attacker to violate compliance goals, reveal confidential information, or to influence a workflow’s behavior. Overall, the workflow modeling lacks research and know-how on the design of reliably secure workflow specification.

This paper introduces and reports on work-in-progress on SWAT, the security workflow analysis toolkit. SWAT realizes the formal foundation provided by InDico [3]–[5], a Petri net-based approach to analyze workflows for compliance with security policies, e.g. data flows and separation of duty requirements. (See Section II-A for details.) Given the Petri net basis, SWAT supports both the *certification* (analysis of workflow models) and *audit* (workflow log analysis). This paper provides the following contributions:

- it presents SWAT, describing its foundation and functionality (cf. Section II-B) and high-level architecture (cf. Section III).

- it shows the analysis of a workflow and some of the properties that can be tested for (cf. Section IV).

Eventually, SWAT shall deliver an powerful toolkit that supports the thorough, well-founded analysis and testing of workflows which advances today’s open source toolbox. The design of SWAT is modular, allowing other researchers to seamlessly add their features as plugins and test them.

SWAT’s major contribution to workflow modeling, i.e. provide a deeper insight into workflow reliable security as a means to improve enterprise systems engineering, can only be achieved in the long run. We plan to classify the findings from case-studies, pilots, and projects to compile a library of “safe” workflow patterns and make them publicly available.

A. Related Work and Tool Support

Workflow analysis is a “hot topic” in business process management. However, it currently focuses on functional properties, e.g. conformance [30], structure [6], and soundness [22]. Previous research into workflow security has developed formal methods for access control [7], [8], [11], delegation [9], [14], and resilience [32], but it has *not* produced tangible tool support based on the developed methods. In particular, only single, explicit information flows (i.e. data flows) are considered, neglecting the effect of chained legal accesses and implicit information flows.

The following workflow modeling and analysis tools exist. (1) *Oryx* is a web-based tool that allows the modeling of workflows [25]. However, the free version of Oryx neither allows the storage of workflows nor their analysis; (2) Supporting process mining, i.e. the extraction of a (business) process from its execution logs, the *ProM* tool is an open source, extensible, and platform-independent framework that supports a wide variety of process mining techniques as plug-ins [26]. ProM provides no means for security analysis. (3) *SeaFlows* provides tool support for the analysis of workflows for compliance verification [28]. However, the tool is not open source and does not support security analysis. (4) *Trident* is a scientific workflow workbench developed at the Microsoft Research [29]. Trident is tailored for, e.g., the automation of scientific workflows and the data provenance; it provides, however, no support for business workflow simulation and features no security analysis.

II. SWAT FOUNDATION AND FUNCTIONALITY

A. Formal Foundation

The formal foundation used by SWAT is the “information flow net” (IFnet), a dialect of colored Petri net tailored for security analysis [5]. IFnet allows the modeling of workflows with their control flow and data flow, and the formalization of security and compliance properties. IFnet thus provides a uniform basis for security workflow analysis based upon and extending well-founded Petri net procedures, such as reachability [24] and structural net reduction [17]. Below we summarize the main characteristics of IFnet; see [4], [5] for the details.

IFnet: Information flow nets. Colored Petri nets (CPN) extend Petri nets to also allow for colored tokens. Accordingly, the flow relation – responsible for moving tokens within a net and capturing workflow dynamics – distinguishes between token colors when firing transitions. IFnet extends CPN in the following ways: Firstly, it annotates transitions with read/write operations; secondly, it assigns to each transition a subject (and alternatively a role); thirdly, it annotates transitions and colored tokens with security labels, which are necessary to capture the underlying multi-level, lattice-based security [15]: *high* for classified; *low* for public.

Besides these annotations, IFnet imposes restrictions on the structure of the net when modeling workflows. Among others, the following well-formedness criteria are given: Firstly, unique start and end points; second, connectedness, i.e. each transition lies within an execution thread of the net; thirdly, boundedness, i.e. each net has a finite set of resources modeling the workflow. Given that, each transition stands for a workflow activity and each place for a state in the workflow execution. The black token models the current execution step and the colored token the processed resources.

Security properties as IFnet stubs. SWAT builds upon a lattice-based security model [15]. Such model divides the objects, subjects, and/or their roles into different security classes according to a lattice of classes. The simplest multi-level security (MLS) model consists of two domains: *high* stands for secret and *low* for public, unclassified. Generally, a *violation*, i.e. security incident, happens whenever a piece of information meant to only appear in the high domain happens in the low domain during the execution.

These violations can be due to *direct* or *indirect* information flows. In the direct case, i.e. data flows, classified data is either written into the low domain (so-called “write down”) or read from the high domain (“read up”) [12]. (This captures the essence of the mandatory access control (MAC) model.) In the indirect case, i.e. “interferences”, classified information can be derived by subjects in the low domain by observing high’s behavior [19].

The main benefit of the MLS-model is that it allows for the *extensional* specification of security properties [27], i.e. context-independent specifications that express *what* is to

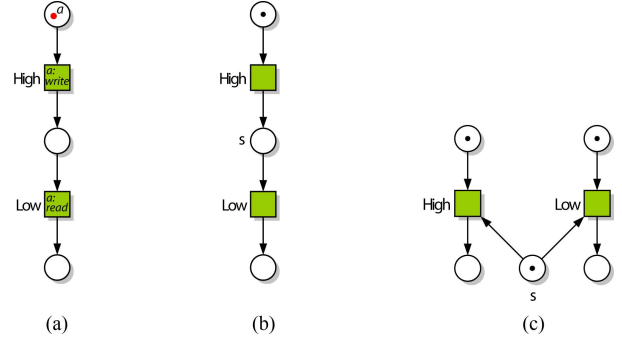


Figure 1. Stubs for the write-down (a) and BNDC properties (b,c).

be prohibited. (Their counterpart, i.e. intensional specifications, are context-dependent and captured by discretionary policies.) Hence, compiling a library of extensional specifications allows the reuse of stubs in different workflows.

SWAT captures MLS properties using IFnet stubs, which are at the moment designed manually. Fig. 1 depicts three stubs. Stub (a) captures the data flow property “write down” for the token a . Stubs (b) and (c) capture the interference property “bisimulation-based non-deducibility on composition” (BNDC) [18]. The BNDC property prohibits any relationship between the high and the low domains and is thus as expressive as the interference property of [19].

IFnet analysis with InDico. Given an IFnet model and the stubs, the analysis detects whether the stubs are *active*, i.e. reachable, during the execution of the net. Showing the opposite means that the workflow is “safe”. The analysis encompasses a static step and a dynamic step: the static step detects whether the place s of a particular stub is present in the net. If so, the dynamic step generates the marking graph of the net (i.e. its “state-space”) to determine whether it is reachable in an execution. If so, the stub is active and, hence, there might be flows from high to low.

Reachability is a decidable problem [24], but its exponential complexity poses a significant hurdle for Petri net-based analysis. However, since workflows produce small nets with straightforward structure and given that transformations can be applied to reduce the size of the net before the generation of the marking graph, SWAT can efficiently carry out security analysis on the MLS-properties stated above.

B. Range of Functions

SWAT realizes a fully-fledged, extensible toolkit for the automated, well-founded analysis of workflows to detect relevant kinds of information leaks and integrity flaws. SWAT provides for the following features:

- *Workflow authoring*: Creation and loading of workflows specified in BPEL, BPMN, and IFnet.
- *Workflow simulation and logging*: Simulate security-relevant workflow instances and generate the execution traces for analysis. To this end, we extend standard

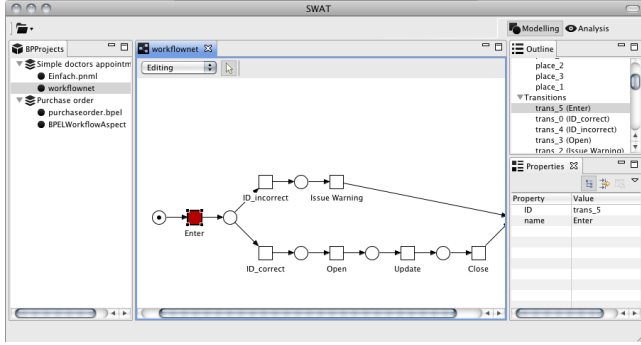


Figure 2. The SWAT application window.

simulation techniques [31] with constraints on timing of activities, number of runs, and on the subjects involved in executions. The logging-plugin allows testing log services and the generation of secure log files, e.g. [1].

- *Workflow reconstruction*: Given the execution traces of a workflow model, this feature reconstructs the original workflow model “process mining” [30]. (The realization of this feature is ongoing work.)
- *Workflow analysis*: With a (modeled or reconstructed) workflow at hand, SWAT supports two kinds of analysis: firstly, *a priori*, design-time analysis of existing workflows; secondly, *a posteriori* analysis based on reconstructed workflows.

While the focus of SWAT is the security workflow analysis, we have also been testing compliance properties, such as those prescribed by HIPAA and SOX. We employ the same procedure as for security analysis: workflows and (compliance) properties are expressed in IFnet and verified using reachability tests (see [2]). Hence, SWAT can be equally used for compliance tests.

III. SWAT ARCHITECTURE

This section overviews the overall architecture of SWAT. Firstly, the application is introduced. Secondly, the modules of the core architecture are described, followed by an outline of the plugin-structure implementing them. Finally, the realization of IFnet and InDico is described.

A. Eclipse-RCP and the User Interface

The extensibility striven for in SWAT needs mechanisms that go beyond the functionality provided by regular widget-libraries. To this end, the eclipse rich client platform (eclipse-rcp) was chosen as cornerstone for SWAT. It provides a modular plugin-based architecture and multi-platform application framework. Furthermore, eclipse provides a rich library of adjustable interactions-components, like the properties- and outline-view shown on the right side in Fig. 2. The screenshot shows the application window after opening two files. The graphical editor in the middle

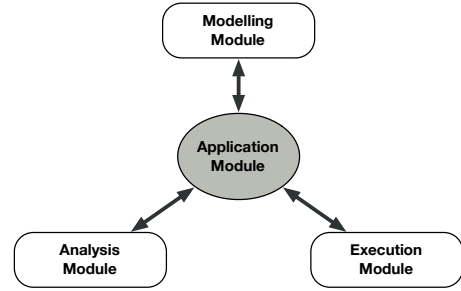


Figure 3. The core modules of SWAT

shows the currently active workflow, which corresponds to the workflow in Fig. 5. Besides interactive modification, it supports the stepwise or automated execution of workflows for testing. On the left, a tree of open files is presented along with its different aspects (source code, workflow graph). The separation between aspects facilitates the representation of particular properties while handling workflows.

SWAT distinguishes between the modeling and the analysis of workflows, by providing different *perspectives* for these use-cases. When switching from “modeling-” to the “analysis”-perspective in the top left, file list, outline and properties views are replaced by an analysis area, allowing selection and appliance of analysis algorithms.

B. Core Architecture

SWAT is built in a modular way to separate functionality and facilitate incremental development. As illustrated in Fig. 3, each working context (modeling, execution and analysis) is represented by a discrete context module (white boxes) around the application-module (grey circle). The application module is responsible for building the application and provides consistent APIs and interaction patterns for workflows. It thereby acts as a bridge between the three separate context-modules and the user. The context-modules enhance the basic application with generic elements for the user interface (UI) and helper classes for their respective tasks.

C. Plugin Structure

The implementation of the core architecture and SWAT is realized by plugins, as shown in Fig. 4. Vertical order denotes inter-plugin dependency, where higher placed plugins refine or concretize functionality by building on constructs provided by lower places plugins. Based upon eclipse-rcp, the base plugin (BPWorkbench) is the lowest plugin of SWAT and resembles the implementation of the application-module. Accordingly, it is responsible for the creation of the UI and provides a unified API for handling and presenting of files and workflows. Upon that, the three plugins for the respective context-modules are constructed. They extend the list of visual elements with specialized implementations for their respective task.

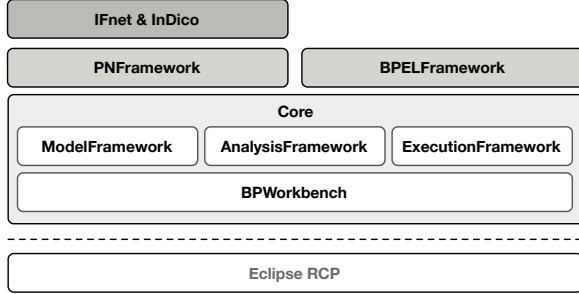


Figure 4. Plugin structure of SWAT

The whole layout allows the easy and consistent integration of different workflow formats into the main application. SWAT supports workflows in BPEL, BPMN and IFnet, where each format is implemented by a separate plugin. These plugins just provide methods for reading corresponding files, an object-model for the particular format as well as a visual representation and execution of its instances. UI elements for file handling or displaying of properties are automatically combined by the core plugins. This reduces development cost to a minimum, as only format specific functionality has to be implemented for future formats.

D. Supporting IFnet and InDico

As Fig. 4 indicates, IFnet and InDico support is implemented separately by an independent plugin on top of framework. Thus plugin simply implements functionality specific for IFnet and InDico by utilizing generic constructs of PNFramework. It creates a new IFnet description along with an adjusted visual representation. To support serialization from and to PNML-files, the description is associated with a dedicated URI for the type-attribute of PNMLs net-elements. The execution logic is implemented by an engine that is also associated with the IFnet description.

InDico was implemented as an extension of the generic analysis algorithm which is, again, associated with IFnet. It consists of a separate plugin that allows the adjustment of analysis parameters and the presentation of analysis results.

IV. EXAMPLE

This section demonstrates the use of SWAT using a simple e-health workflow model, which is to be certified for compliance with an isolation requirement. For the transformation and analysis of the workflow, the InDico framework [5] is employed, which is implemented as a component of SWAT.

Fig. 5 shows the *Open Patient Record* workflow fragment in BPMN notation, which originates from a collection of hospital processes and models the access cycle to a patient record: after entering a patient ID, access is denied if it is wrong, and a warning is issued. Otherwise, access is granted to the subject that requested it. After processing of the record has ended, it is closed and the workflow terminates.

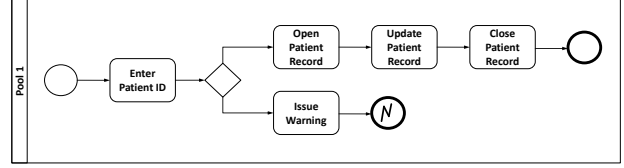


Figure 5. *Open Patient Record* workflow fragment.

Using SWAT, *Open Patient Record* is to be certified for multi-instance isolation. The goal is to find out whether multiple concurrently running instances of *Open Patient Record* are isolated or whether there are ways for them to interfere and exchange information. Isolation of process instances is a mandatory security requirement for service providers dealing with several clients, imposed by e.g. the European Network and Information Security Agency [16] and the National Institute of Standards and Technology [21].

A. IFnet and Transformation

The *Open Patient Record* workflow is read into SWAT which transforms it into an IFnet model. IFnet represents workflow activities as *transitions*, which are drawn as squares. Unfilled circles stand for *places*, which model execution paths and communication channels. Tokens, drawn as filled dots, represent both execution states and resources (e.g. documents, data items, variables). Unlike in traditional Petri nets, tokens in IFnet can be distinguished by an identifier (“color”), thereby allowing explicit modeling of different resources. In the IFnet model there are two different tokens. First, a “black” token (held by the place on the very left side) indicating that the workflow’s state is prior to the execution of the first activity “Enter”. Second, a “red” token (in the “Record”-place) denotes the patient record.

B. Transformation and Labeling

InDico includes automated strategies for the transformation of IFnet models and their annotation (labeling) with security levels in order to certify a process with regard to a specific security requirement. For the isolation case, the corresponding strategy “clones” an IFnet in order to model the behavior of two concurrently running instances and in particular their interaction and usage of shared resources. The instances are annotated with different security levels (High, Low). This annotation expresses the requirement that there must not be any information transfer from the High-labeled instance to the Low-labeled instance, i.e. that both instances are – informationally-wise – isolated from each other. Fig. 6 shows the transformed and annotated *Open Patient Record* workflow, where the High-labeled instance (indicated by an H) is at the top and the Low-labeled instance (indicated by an L) at the bottom. The only connection between the instances is the Record-place, which holds the patient record that may be accessed by both of them.

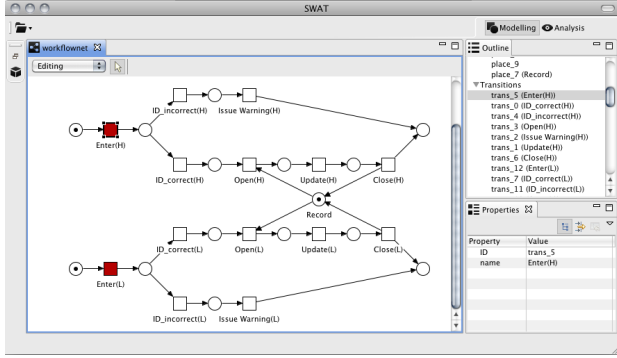


Figure 6. Cloned and labeled *Open Patient Record* workflow.

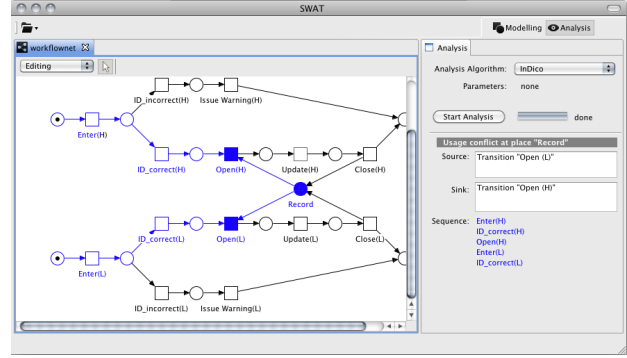


Figure 7. Detected leaks in the *Open Patient Record* workflow.

C. Certification

InDico’s analysis component verifies an annotated IFnet model to rule out information transmission from High-labeled IFnet components (data items and transitions) to Low-labeled components. The policy is a combination of two parts: First, a MAC-style access control policy, which prohibits that information from higher-labeled data objects is accessed by lower-level activities or copied to lower-labeled data objects. Second, a noninterference-style policy which prevents that High-labeled process activities can influence Low-labeled activities (and thus transfer information). Compliance with the noninterference-style policy implies fulfillment of the *Bisimulation Non-Deducibility on Composition*-property (*BNDC*), which guarantees that Low-level users cannot learn anything about the occurrence of Low-activities [13]. Put another way, the stubs (b) and (c) in Fig. 1 are violated.

The analysis of the workflow in Fig. 6 detects the following interferences, revealing the influence of the High process instance on its Low counterpart:

- **Conflict.** By firing the transition `Open(H)`, the patient record is removed from the storage place and transition `Open(L)` is blocked until the token is returned. Here, the High part of the net influences the Low part as it prevents the transition from firing. There is an information flow which allows the Low part to deduce that High is currently holding the patient record.
- **Causal.** By firing the transition `Close(H)`, the token representing the patient record is returned to its storage place and might be consumed by transition `Open(L)`. Hence, opening the patient record on the Low side requires its preceding return on the High side, which reveals to Low the fact that High has returned the record.

InDico’s certificate indicates these interferences as violations of the *BNDC* property and SWAT graphically points out the fragments of the IFnet model where they occurred. This is depicted in Fig. 7.

V. SUMMARY AND FURTHER WORK

This paper presented and reported on ongoing work on SWAT, a security workflow analysis toolkit. The goal of SWAT is to provide a modular platform for researchers to test with their analysis approaches. Further work approaches both theoretical and practical aspects. With regard to theory, we investigate and further develop the IFnet approach to allow the representation of other security properties which are not part of the current implementation. Furthermore, in this setting we develop more efficient analysis algorithms that simplify the net (representing the workflow) without missing attacks. On the practical side, we integrate further workflow formats and realize their transformations into IFnet. An important issue for SWAT regards its evaluation. In near future we will test SWAT in a set of workflows within a German bank and thereby obtain practical evidence of its adequacy and precision to cope with industrial workflows.

REFERENCES

- [1] R. Accorsi, “BBox: A distributed secure log architecture,” in *European Workshop on Public Key Services, Applications and Infrastructures*, ser. Lecture Notes in Computer Science, J. Camenish and C. Lambrinouidakis, Eds., no. 6711. Springer, 2011, pp. 109–124.
- [2] R. Accorsi, L. Lewis, and Y. Sato, “Automated certification for compliant cloud-based business processes,” *To appear in Wirtschaftsinformatik*, 2011.
- [3] R. Accorsi and C. Wonnemann, “Static information flow analysis of workflow models,” in *Conference on Business Process and Service Computing*, ser. Lecture Notes in Informatics, W. Abramowicz, K.-P. F. Rainer Alt, and L. Maciaszek, Eds., vol. 147. GI, 2010, pp. 194–205.
- [4] —, “Strong non-leak guarantees for workflow models,” in *ACM Symposium on Applied Computing*. ACM, 2011, pp. 308–314.
- [5] —, “InDico: Information flow analysis of business processes for confidentiality requirements,” in *ERCIM Workshop on Security and Trust Management*, ser. Lecture Notes in Computer Science, J. Cuellar et al., Eds., vol. 6710. Springer, 2011, pp. 194–209.

- [6] N. Adam, V. Atluri, and W.-K. Huang, "Modeling and analysis of workflows using petri nets," *Journal of Intelligent Information Systems*, vol. 10, no. 2, pp. 131–158, 1998.
- [7] A. Armando and S. E. Ponta, "Model checking of security-sensitive business processes," in *Workshop on Formal Aspects in Security and Trust*, ser. Lecture Notes in Computer Science, P. Degano and J. Guttman, Eds., vol. 5983. Springer, 2009, pp. 66–80.
- [8] V. Atluri, S. A. Chun, and P. Mazzoleni, "A Chinese Wall security model for decentralized workflow systems," in *ACM Conference on Computer and Communications Security*. ACM, 2001, pp. 48–57.
- [9] V. Atluri and J. Warner, "Supporting conditional delegation in secure workflow management systems," in *ACM Symposium on Access Control Models and Technologies*, E. Ferrari and G.-J. Ahn, Eds. ACM, 2005, pp. 49–58.
- [10] —, "Security for workflow systems," in *Handbook of Database Security*, M. Gertz and S. Jajodia, Eds. Springer, 2008, pp. 213–230.
- [11] M. Barletta, S. Ranise, and L. Viganò, "Verifying the interplay of authorization policies and workflow in service-oriented architectures," in *Conference on Computational Science (3)*, 2009, pp. 289–296.
- [12] D. Bell and L. LaPadula, *Secure Computer Systems: Mathematical Foundations*. MITRE Corporation, 1973.
- [13] N. Busi and R. Gorrieri, "Structural non-interference in elementary and trace nets," *Mathematical Structures in Computer Science*, vol. 19, no. 6, pp. 1065–1090, 2009.
- [14] J. Crampton and H. Khambhammettu, "On delegation and workflow execution models," in *ACM Symposium on Applied Computing*, R. L. Wainwright and H. Haddad, Eds. ACM, 2008, pp. 2137–2144.
- [15] D. Denning, "A lattice model of secure information flow," *Communications of the ACM*, vol. 19, no. 5, pp. 236–243, Mai 1976.
- [16] ENISA, "Security & resilience in governmental clouds," European Network and Information Security Agency, Tech. Rep., 2010.
- [17] J. Esparza and C. Schröter, "Unfolding based algorithms for the reachability problem," *Fundamenta Informaticae*, vol. 47, no. 3-4, pp. 231–245, 2001.
- [18] R. Focardi and R. Gorrieri, "Classification of security properties (Part I: Information flow)," in *Foundations of Security Analysis and Design*, ser. Lecture Notes in Computer Science, R. Focardi and R. Gorrieri, Eds. Springer, 2001, vol. 2171, pp. 331–396.
- [19] J. Goguen and J. Meseguer, "Security policies and security models," in *IEEE Symposium on Security and Privacy*, 1982, pp. 11–20.
- [20] B. Hayes, "Cloud computing," *Communications of the ACM*, vol. 51, no. 7, pp. 9–11, 2008.
- [21] W. Jansen and T. Grance, "Guidelines on security and privacy in public cloud computing," NIST: National Institute of Standards and Technology, Tech. Rep. 800-144, 2011.
- [22] N. Lohmann, P. Massuthe, C. Stahl, and D. Weinberg, "Analyzing interacting WS-BPEL processes using flexible model generation," *Data Knowledge Engineering*, vol. 64, no. 1, pp. 38–54, 2008.
- [23] L. Lowis and R. Accorsi, "Finding vulnerabilities in SOA-based business processes," *IEEE Transactions on Service Computing*, 2011, to appear.
- [24] E. Mayr, "An algorithm for the general Petri net reachability problem," *SIAM J. Comput.*, vol. 13, no. 3, pp. 441–460, 1984.
- [25] Oryx: <http://bpt.hpi.uni-potsdam.de/Oryx>, 2011.
- [26] ProM: <http://prom.win.tue.nl/tools/prom/>, 2011.
- [27] B. Roscoe, "Intensional specifications of security protocols," in *IEEE Computer Security Foundations Workshop*. IEEE Computer Society Press, 1996, pp. 28–38.
- [28] SeaFlows: <http://www.uni-ulm.de/en/in/iui-dbis/research/projects/seaflows.html>, 2011.
- [29] Trident: <http://research.microsoft.com/en-us/collaboration/tools/trident.aspx>, 2011.
- [30] W. van der Aalst, "Workflow model analysis," in *Encyclopedia of Database Systems*, L. Liu and T. Özsu, Eds. Springer, 2009, p. 3551.
- [31] —, "Business process simulation," in *Handbook on Business Process Management*, J. vom Brocke and M. Rosemann, Eds. Springer, 2010, vol. 1, pp. 313–338.
- [32] Q. Wang and N. Li, "Satisfiability and resiliency in workflow systems," in *European Symposium On Research In Computer Security*, ser. Lecture Notes in Computer Science, J. Biskup and J. Lopez, Eds., vol. 4734. Springer, 2007, pp. 90–105.
- [33] C. Wolf and P. Harmon, "The state of business process management," BPTrends Report, 2010, available at <http://www.bptrends.com/>.