Static Information Flow Analysis of Workflow Models

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

Abstract: This paper proposes a framework for the detection of information leaks in workflow descriptions based on static information flow analysis. Despite the correct deployment of access control mechanisms, certain information leaks can persist, thereby undermining the compliance of workflows to policies. The framework put forward in this paper identifies leaks induced by the structure of the workflow. It consists of an adequate meta-model for workflow representation based on Petri nets and corresponding components for the transformation and analysis. A case study illustrates the application of the framework on a concrete workflow in BPEL notation.

1 Introduction

Over 70% of all business processes deployed today rely on business process management systems for their automated execution [WH10]. The central entity here are workflow specifications that model the business process. Despite the steadily growing expenses in tool design for the secure and compliant deployment of workflows, dependability incidents are still soaring [Dif08], as well as the monetary damage they lead to [MAHS10].

A major source of dependability incidents – and in fact non-compliance – in this setting are information leaks [BKN+09]. Even if correct access control mechanisms and corresponding policies are in place, these leaks undermine the confidentiality of data items and workflow characteristics meant to remain secret. The technical issue here is that access control mechanisms only monitor a particular transmission channel between a classified and a public subject, whereas so-called "covert-channels" are neglected, i.e. channels that are not meant to transfer information but are misused to do so [Lam73].

This paper proposes a framework for the detection of information leaks in workflow descriptions based on static information flow analysis. Information flow (IF) control lays down the basis for a semantic specification of security requirements [ML97]. Compared with traditional access control policies, IF properties are stronger because they capture information propagation throughout the system (*end-to-end*) rather than mere data access. Thus, they can capture security violations that lie beyond the scope of access control mechanisms, such as the information leaks that take place over covert-channels.

The analysis of workflow specifications against IF properties is a novel, promising approach to complement access control mechanisms and, thereby, achieve strong security and compliance guarantees [AW09]. The framework put forward in this paper takes a

Petri net representation of the workflow specification (so-called "IF-net") and analyzes the corresponding net against IF properties. The analysis is mechanized and amounts to testing whether a certain state of the net is reached.

Besides outlining the approach and its elementary theoretical underpinnings, the paper illustrates the application of the framework with an example taken from a larger case study. The example shows how a workflow in BPEL notation is verified for instance isolation, i.e. whether there is any information leak between multiple instances of the workflow. For this purpose, the BPEL process is transformed into a rudimentary IFnet that models the interacting instances. The applied verification technique is tailored for the analysis of Petri nets [BG09]. It detects specific covert-channels arising from the structure of the workflow, in particular the competition for shared resources and control-flow.

The paper makes the following contributions:

- It argues for a semantic analysis of workflow descriptions and their behavior in order to provide reliable security guarantees.
- It proposes a framework for the static IF analysis of workflows and outlines its theoretical underpinning and capabilities. In particular, it presents IFnet, a meta-model based on Petri net to represent workflows for IF analysis.
- It demonstrates in a case study the application of the framework for the verification of a BPEL process for information leaks.

Overall, the approach this paper proposes merely detects the existence of a covert-channel and, thus, of a (possibly intentional) information leak. It is worth emphasizing the following two aspects. First, the detected interference does not necessarily imply a violation of a security policy, it merely indicates an IF. While distinguishing between harmful and harmless interferences is out of the scope of this paper, extensions of the framework presented in Section 2 will also address this issue. Second, once a covert-channel is considered "dangerous", countermeasures must be taken to stop the leak or at least reduce its throughput. Since covert-channel elimination is traditionally infeasible and ends up adding more covert-channels [PN92], other mechanisms, such as redundancy and uniform timing delays, must be set in place to control the leak. Future work aims at evaluating existing strategies for covert-channel control.

This paper is structured as follows: Section 2 outlines the framework for information flow control in workflows. Section 3 presents the case study demonstrating the detection of information flow between interacting instances of a BPEL process. Section 4 lists related work and Section 5 concludes the paper.

Preliminaries on information flow analysis. Information flow analysis is a formal approach to computer security to guarantee the absence of data leaks between designated subjects of a system. For this, the subjects which interact with the system (and actions they can perform) are assigned security labels that indicate their clearances. The security labels are arranged in a lattice, which in the simplest case consists of the two labels High and Low with $\mathit{High} \succ \mathit{Low}$ [Den76].

In IF analysis, the criteria for the security of a system is based on the notion of non-interference [GM82], i.e. the absence of "interferences" between the $High$ part of the system and the Low part. Put another way, the actions of $High$ users lead to *no* observable effect on the behavior of Low. Demonstrating this property ensures that the Low part cannot deduce any information on the behavior of $High$. Interferences can happen over various "channels": besides explicit data transmission through message sending or shared storage (dataflow), there is a variety of *covert-channels*. These include, for instance, control flow, timing and termination behavior, or the probability distribution of processed data [SM03]. To formally prove that a system does not allow IF, the relevant aspects of its behavior are modeled and checked for an adequate non-interference property.

2 A Framework for Information Flow Control in Workflows

This section outlines the proposed approach for IF control in automated workflows. Given a workflow specification, e.g. a BPEL or a BPMN model, the first step consists of *translating* this specification into a Petri net representation. IFnet is the target meta-model. Essentially, it is a colored Petri net enriched with annotations to express the security classes for IF analysis. The IFnet model obtained from the translation of a workflow specification is then *annotated* with the corresponding security classes. Subsequently, a static IF analysis is carried out upon the resultant IFnet. The following provides further details on the corresponding steps and indicates where ongoing work is needed. Section 3 shows an instantiation of this framework with existing tool-support.

IFnet. IFnet is based on Petri nets, as these feature formal semantics and various notions of non-interference for Petri net-based systems are available [BG03]. Further, with their flow-oriented graphical notation, being alike with most workflow management systems, Petri nets are suitable for the formal modeling of workflows [LVD09]. However, the majority of approaches in workflow modeling using Petri nets focuses on the control flow, disregarding dataflow and further covert channels, which are crucial for information flow analysis. IFnet integrates and extends existing approaches to build a workflow meta-model to express different aspects of the workflow, such as its dataflow and resource exhaustion. Further, it is planned to integrate also dynamic aspects into IFnet, such as timing behavior and the probability distribution of execution paths. This information is gained from execution log traces and enables the forensic detection of information flows.

Transformation to IFnet. Transformation of workflows specified in common industrial formats (such as BPEL and BPMN) into IFnet is carried out automatically by extending existing mapping functions [LVD09]. For the BPEL case, there are at least two tools which provide a feature-complete mapping of a workflow's control flow to standard Petri nets, namely "BPEL2oWFN" [Loh07] and "WofBPEL" [OVvdA$^+$05]. Building upon these tools, generated Petri nets will be subjected to a second transformation step that produces the correspondent IFnet models.

Annotation of IFnet models. The goal of the annotation component is to label an IFnet model for IF analysis. Different strategies can be taken to do so, depending on the kind

of covert-channel which is to be detected. As shown in Section 3, one strategy is to split the net into symmetric parts representing High and Low users. This "splitting" has been very handy while analyzing IFnet for storage channels. In the dataflow setting, a possible strategy is to derive the labeling from the corresponding access control policies for the particular workflow. Further, the framework also allows the user to label the workflow at his/her will, as in traditional IF analysis.

Focusing on the case of security policies for sake of example, the approach is to generate a lattice of security labels, which is arranged according to the data access modalities specified in the policy, and to assign these labels to the corresponding activities and data items of the IFnet model. As an example, consider a policy that denotes that a Subject 1 may read some data item File A, while Subject 2 must not read that file. An annotation function could classify Subject 1 and File A as "High", while Subject 2 would be "Low". This requirement is significantly stronger than the original access control policy, as it rules out any actions taken by Subject 1 that have an observable effect on Subject 2 (and thus might transmit information on File A).

IF analysis. The analysis tool checks an annotated IFnet model for possible interferences between its "higher"-level and "lower"-level parts. For the analysis of dataflows (i.e. the explicit exchange of data items), so-called "propagation graphs" are extracted from the IFnet model which denote the possible data exchange among the involved subjects. The propagation graphs are subsequently traversed to detect illicit dataflows [AW10]. The detection of covert channels is based on different bisimulation-based notions of noninterference which have been formalized for Petri nets [BG03]. In particular, *Bisimulation-based Non-Deducibility on Composition* (BNDC) [FG95] is a suitable property, as it can be efficiently checked through a semi-static analysis (see Section 3).

3 Case Study

To illustrate the idea behind the framework, this section presents a case study for the verification of a BPEL process. To this end, it employs preliminary variants of the described components.

To preserve confidentiality, several compliance requirements, such as HIPAA and Sarbanes-Oxley, demand that workflows with different security clearances are isolated from each other, i.e. there must be no information exchange between instances of corresponding workflows. A well-known example for such a requirement is the Chinese-Wall security model [BN89]. It aims to prevent conflicts of interest in organizations dealing with clients, which are, for instance, direct competitors in the same market.

This case study demonstrates the verification of such an isolation requirement for concurrently running instances of a BPEL workflow. The BPEL workflow is transformed into a preliminary variant of IFnet. The transformation encompasses two steps: first, the BPEL model is mapped to an equivalent Petri net model using an existing transformation tool; second, the resultant Petri net is "unfolded" into an IFnet instance which models concurrently running instances and their access to shared resources. The IFnet model is subse-

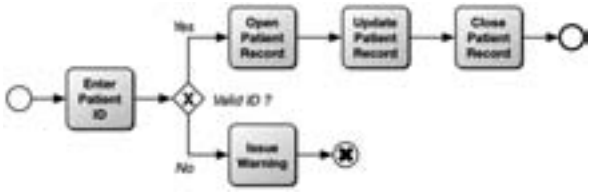

Figure 1: "Update Patient Record" workflow.

quently checked for the BNDC property, which asserts that there can be no information flow from one instance to the other. For the verification, the *Petri Net Security Checker* is used, which implements a checking algorithm for BNDC on Petri nets [FGF08].

Exemplary BPEL Process. The case study uses the "Update Patient Record" workflow depicted in Figure 1, which originates from a project on the automation of business process compliance that was carried out in cooperation with a hospital. Albeit simple, it is a central fragment that recurs in several of the hospital's information systems, e.g. for accounting, medical treatment and insurance billing. It consists of five activities (depicted as boxes) and an exclusive choice (x-diamond) that represents an if-statement. The following fragment shows the corresponding BPEL tags of the workflow.

```
<sequence name="main">
  <empty name="Enter_Patient_ID"/>
  <switch name="Switch_1">
    <case>
      <sequence>
        <empty name="Open_Patient_Record"/>
        <empty name="Update_Patient_Record"/>
        <empty name="Close_Patient_Record"/>
      </sequence>
    </case>
    <otherwise>
      <empty name="Issue_Warning"/>
    </otherwise>
  </switch>
</sequence>
```

The concurrent access to the same patient record causes information leaks. If, for instance, a doctor opens a specific record and blocks an access attempt to the same record by another subject – irrespective of the fact that access control mechanisms are in place –, this subject obtains information on the doctor's behavior: the subject knows that the doctor uses the specific record and might deduce from this fact which patient the doctor is currently treating, as well as the time the doctor needs to update the record.[1] Hence, there would be an IF among workflow instances violating the isolation requirement. The goal of the analysis is to detect these leaks. The following demonstrates how the approach proposed in the previous section achieves this goal.

[1] It is important to note that the patient record in this case holds only administrative information which may legitimately be accessed by all involved parties, i.e. access to the record alone does not constitute an illegal IF.

3.1 Transformation of Workflow Descriptions in IFnet

3.1.1 BPEL to Petri net

The transformation of the initial BPEL process employs the tool "BPEL2oWFN", which implements a feature-complete mapping of BPEL 2.0 to so-called "Open Workflow Nets", a variant of Petri nets tailored to workflow modeling. Some basic definitions of Petri nets are given to facilitate the understanding of the framework.

Petri nets. A Petri net is a tuple $N = (P, T, F)$, where P is a finite set of *places*, T is a finite set of *transitions* such that $P \cap T = \emptyset$, and $F \subseteq (P \times T) \cup (T \times P)$ is a set of directed arcs, called the *flow relation*.

At any time a place contains zero or more *tokens*. The *marking* (or *state*) is the distribution of tokens over places. A marking is a *bag* over the set of places P, i.e. a function from P to the natural numbers: $M \in P \longrightarrow \mathbb{N}$. A partial ordering is defined to compare states. For any two states M_1 and M_2, $M_1 \leq M_2$ *iff* for all $p \in P$: $M_1(p) \leq M_2(p)$. The *sum* of two bags $(M_1 + M_2)$ is defined in a straightforward way.

A *marked* Petri net is a pair (N, M), where $N = (P, T, F)$ is a Petri net and M is a bag over P denoting the marking of the net. The set of all marked Petri nets is denoted $\mathcal{N}$.

Let $N = (P, T, F)$ be a Petri net. Elements of $P \cup T$ are called *nodes*. A node x is an *input node* of another node y *iff* there is a directed arc from x to y (i.e. xFy). Node x is an *output node* of y *iff* yFx. For any $x \in P \cup T$, $\overset{N}{\bullet} x = \{y \mid yFx\}$ and $x \overset{N}{\bullet} = \{y \mid xFy\}$; the superscript N can be omitted if it is clear from the context.

The number of tokens may change during the execution of the net. Transitions are the active components in a Petri net. They change the state of the net according to the following *firing rule*:

1. A transition t is *enabled iff* each input place p of t contains at least one token.
2. An enabled transition may *fire*. If transition t fires, then t *consumes* one token from each input place p of t and *produces* one token for each output place p of t.

Given a Petri net $N = (P, T, F)$ and a state M_1, the following notation is defined:

- $M_1 \xrightarrow{t} M_2$: transition t is enabled in M_1 and firing t in M_1 results in state M_2.
- $M_1 \longrightarrow M_2$: there is a transition t such that $M_1 \xrightarrow{t} M_2$.
- $M_1 \xrightarrow{\sigma} M_n$: the firing sequence $\sigma = t_1 t_2 t_3 ... t_{n-1}$ from state M_1 leads to state M_n via a (possibly empty) set of intermediate states $M_2, ..., M_{n-1}$, i.e. $M_1 \xrightarrow{t_1} M_2 \xrightarrow{t_2} ... \xrightarrow{t_{n-1}} M_n$.

A state M_n is *reachable* from M_1 (notation $M_1 \xrightarrow{*} M_n$) *iff* there is a firing sequence σ so that $M_1 \xrightarrow{\sigma} M_n$. The empty firing sequence is also allowed, i.e. $M_1 \xrightarrow{*} M_1$.

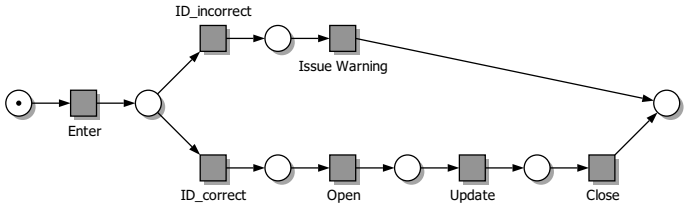

Figure 2: Petri net-model of the workflow.

Transformation to IFnet. Application of the "BPEL2oWFN"-tool to the BPEL code of the process in Figure 1 yields the Petri net depicted in Figure 2. The activities of the workflow are modeled by transitions, which are depicted as green (or gray) boxes. In addition to activities, the "BPEL2oWFN"-tool has introduced extra transitions which stand for the two cases of the exclusive choice. Places – depicted as circles – represent conditions and the state of the workflow is marked by a single token, which is depicted as a black dot which initially resides in the start place. The arcs are depicted as arrows.

3.1.2 Step 2: Petri net to IFnet

The Petri net in Figure 2 models the behavior of a single workflow instance. To check instance isolation, the net is "unfolded" to model two instances of the workflow and their explicit interaction. Interaction between instances of the "Update Patient Record" workflow occurs through access to the patient record, which is a shared, limited resource: multiple instances may attempt to access a specific patient record, but only one instance can hold the record at one point in time.

Modeling Resource Consumption. Resource consumption of a workflow is modeled through a relation $\mathcal{D}$ which denotes the pairs of transitions which obtain and release a resource, respectively. For a Petri net $(N, M) = ((P, T, F), M)$ with initial marking M, $\mathcal{D} \subseteq (T \times T)$ is a binary relation over the transitions of N with the following property:

- For each $(t_1, t_2) \in \mathcal{D}$ there are states M_1, M_2 and firing sequences σ_1, σ_2 such that $M \xrightarrow{\sigma_1 t_1} M_1$ and $M_1 \xrightarrow{\sigma_2 t_2} M_2$. That is, there must be at least one path through the net where t_1 precedes t_2.

Unfolding to IFnet. An IFnet models the interaction of two instances of a Petri net-model. The two instances are denoted "High" (H) and "Low" (L), expressing the difference in their security clearances. For a marked Petri net $(N, M) = ((P, T, F), M)$ and a resource relation $\mathcal{D}$, the corresponding IFnet is a tuple $(P_L \cup P_H \cup P_\mathcal{D}, T_L \cup T_H, F_L \cup F_H \cup F_\mathcal{D}, M_L + M_H + M_\mathcal{D})$ where:

- $((P_L, T_L, F_L), M_L)$ corresponds to the net $((P, T, F), M)$.

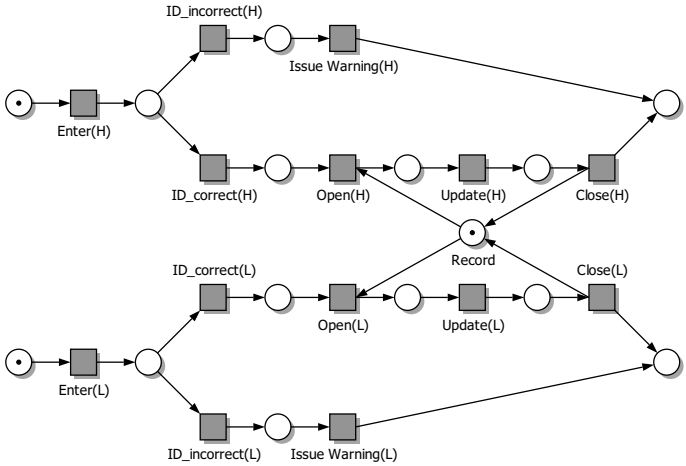

Figure 3: IFnet of the workflow.

- $((P_H, T_H, F_H), M_H)$ is an equivalent net to $((P_L, T_L, F_L), M_L)$ with its elements renamed for distinction. The function $\Upsilon_{T_L \to T_H} : T_L \longrightarrow T_H$ maps the transitions from T_L to their counterparts in T_H.
- $P_{\mathcal{D}}$ is a set of places which model the blocking of resources. There exists exactly one $p \in P_{\mathcal{D}}$ for each pair $(t_0, t_1) \in \mathcal{D}$. The function $\Upsilon_{P_{\mathcal{D}} \to \mathcal{D}} : P_{\mathcal{D}} \longrightarrow \mathcal{D}$ maps the places from $P_{\mathcal{D}}$ to the corresponding pair of transitions in T (and therewith in T_L).
- $M_{\mathcal{D}}$ denotes the initial marking of places $P_{\mathcal{D}}$. $M_{\mathcal{D}}$ marks each $p \in P_{\mathcal{D}}$ with exactly one token.
- $F_{\mathcal{D}}$ denotes the arcs which connect the places in P to the blocking and releasing transitions in T_L and T_H. For each $p \in P_{\mathcal{D}}$ with $\Upsilon_{P_{\mathcal{D}} \to \mathcal{D}}(p) = (t_0, t_1)$ it contains the following arcs:
  - (p, t_0) denotes the blocking of the resource modeled by p through transition t_0.
  - (t_1, p) denotes the release of that resource through transition t_1.
  - $(p, \Upsilon_{T_L \to T_H}(t_0))$ denotes the blocking of resource p through the transition in T_H which corresponds to t_0.
  - $(\Upsilon_{T_L \to T_H}(t_1), p)$ denotes the corresponding release of the resource.

For the Petri net from Figure 2, the resource relation is $\mathcal{D} = \{(Open, Close)\}$, denoting that the transition Open blocks a patient record and transition Close releases it. The corresponding IFnet is depicted in Figure 3. The upper part of the Figure shows the "High"-instance of the workflow and the lower part its "Low"-instance. The patient record is modeled through a token which initially resides in an extra place representing its storage. Whenever one of the Open transitions fires, the token is consumed and the storage place

remains empty until the token is returned by the firing of the corresponding Close transition. Inspection of the IFnet model indicates that the interaction of the two instances can lead to a violation of the isolation requirement, a fact that is proven by the following formal verification.

3.2 Static Information Flow Analysis

The possibility of in IF between the "High" and the "Low" instances of the workflow is assessed according to the BNDC property. Originally defined on a process algebra, it is defined for Petri nets in [FGF08]. BNDC rules out any information transmission from the "High" part of a system to its "Low" part. It formalizes information transmission in a notion of non-interference [GM82]: a Petri net fulfills BNDC if, whatever behavior the "High" part of the net exhibits, there must be no observable effect on its "Low" part.

The *Petri Net Security Checker* (PNSC) implements a checking algorithm for BNDC. Specifically, PNSC verifies the so-called PBNI+ property (*Positive Place-Based Non-Interference*) which has been proven to be equivalent to BNDC [BG09]. PBNI+ is checked using a semi-static analysis, which firstly inspects the Petri net for patterns where "High" and "Low" parts might interfere and, subsequently, checks whether these patterns are *active*, i.e. if they can be executed.

The analysis of the IFnet from Figure 3 shows that the storage place (labeled Record in Figure 3) is both an *active conflict place* and an *active causal place*. This means that two types of information flow might exist from the "High" instance to the "Low" instance:

1. Firing of 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 (through a so-called *resource exhaustion channel*) which allows the "Low" part to deduce that "High" is currently holding the patient record.

2. Firing of 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: this causality reveals to "Low" the fact that "High" has returned the record.

Although rather simple, this case study illustrates the kind of analyses the framework aims to carry out and the types of information leaks it is able to detect. Current case studies involve more complex workflow models, thereby showing and justifying extensions from the core analysis framework. Particularly interesting for the investigation of data leaks in workflow models is the concept of "swimlane" in BPMN descriptions. Since each swimlane characterizes one type of "actor", e.g. along a sensitive supply-chain, it is relevant and evident to elucidate in a formal manner what these actors can deduce from the others.

4 Related Work

The idea of using information flow control for the verification of workflow models is to the best of our knowledge new. The closest work in this direction is by Hutter and Volkamer. They present an approach for the secure composition of web services in a service-oriented architecture based on language-based information flow techniques [HV06]. The majority of approaches towards workflow security is based on access control models, e.g. [BRV09]. Namiri and Stojanovic propose a pattern-based methodology stipulating the construction of business processes from recurring patterns that are deemed secure [NS07]. While this includes several relevant organizational patterns, such as separation of duties and four-eye rule, it does not focus on information leaks. The REALM approach envisages the automated generation of rules for an access control monitor from a high-level specification language [GLMP05]. Overall, the goal of these approaches is to provide security guarantees concerning subjects' data access. They cannot however detect covert-channels as the one illustrated in the case study.

Considering merely dataflows – an overly simplified form of interference – Knorr presents an approach for the verification of multi-level security policies, such as Bell-LaPadula, in workflow models based on Petri nets [Kno01]. Sun et al. and Trčka et al. propose methodologies for the integration of the dataflow perspective into formal workflow models and the verification for sets of predefined error patterns [SZNS06, TvdAS09]. These approaches do not allow to detect data leaks happening over covert-channels.

5 Summary

This paper presented a framework for the information flow analysis of workflow specifications and demonstrated its applicability in a case study, where pointers to ongoing and related work were given. Overall, the static detection of information flows in workflow models is a novel, promising research direction to ensure formally-founded security guarantees for workflows. However, the enhanced expressive power provided by information flow analysis has a limitation: the interferences detected during the analysis may not necessarily denote an information leak, although they theoretically do so. Hence, obtaining a fine-grained distinction between leaks and necessary flows is necessary. Here, the use of security policies as declassification rules [SS05] might provide a way to distinguish between interferences.

The framework presented in this paper is part of a larger research effort, whose goal is to provide the fundamental building blocks for the forensic analysis of workflow executions. In this setting, a forensic reconstruction algorithm is being developed which allows to extract IFnet models solely from log traces of workflow executions. This allows a posteriori analysis of workflow executions if no explicit model is present. Other covert-channels arising from the dynamics of workflow execution, such as timing and the probability distribution of execution paths, can be detected.

References

[AW09] Rafael Accorsi and Claus Wonnemann. Detective Information Flow Analysis for Business Processes. In W. Abramowicz et al., eds, *Proc. of Business Processes, Services Computing and Intelligent Service Management*, vol. 147 of *LNI*, pages 223–224, 2009.

[AW10] Rafael Accorsi and Claus Wonnemann. Auditing Workflow Executions against Dataflow Policies. In W. Abramowicz and R. Tolksdorf, eds, *Proc. of Business Information Systems*, vol. 47 of *LNBIP*, pages 207–217. Springer, 2010.

[BG03] Nadia Busi and Roberto Gorrieri. A Survey on Non-interference with Petri Nets. In J. Desel et al., eds, *Lectures on Concurrency and Petri Nets*, vol. 3098 of *LNCS*, pages 91–113. Springer, 2003.

[BG09] Nadia Busi and Roberto Gorrieri. Structural Non-interference in Elementary and Trace Nets. *Mathematical Structures in Computer Science*, 19(6):1065–1090, 2009.

[BKN+09] Kai-D. Bussmann, Oliver Krieg, Claudia Nestler, Steffen Salvenmoser, Andreas Schroth, Alex Theile, and Daniela Trunk. Wirtschaftskriminalität 2009 – Sicherheitslage in deutschen Großunternehmen. Universität Halle-Wittenberg and PricewaterhouseCoopers AG, 2009. (in German)

[BN89] David Brewer and Michael Nash. The Chinese-Wall Security Policy. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 206–214, 1989.

[BRV09] Michele Barletta, Silvio Ranise, and Luca Viganò. Verifying the Interplay of Authorization Policies and Workflow in Service-Oriented Architectures. In *Proc. of the Conference on Computational Science (3)*, pages 289–296, 2009.

[Den76] Dorothy Denning. A Lattice Model of Secure Information Flow. *Communications of the ACM*, 19(5):236–243, 1976.

[Dif08] Whitfield Diffie. Information security: 50 years behind, 50 years ahead. *Communications of the ACM*, 51(1):55–57, 2008.

[FG95] Riccardo Focardi and Roberto Gorrieri. A Taxonomy of Security Properties for Process Algebras. *Journal of Computer Security*, 3(1):5–34, 1995.

[FGF08] Simone Frau, Roberto Gorrieri, and Carlo Ferigato. Petri Net Security Checker: Structural Non-interference at Work. In P. Degano et al., eds, *Proc. of the Formal Aspects in Security and Trust*, vol. 5491 of *LNCS*, pages 210–225. Springer, 2008.

[GLMP05] Christopher Giblin, Alice Liu, Samuel Müller, and Birgit Pfitzmann. Regulations Expressed as Logical Models. In *Proc. Legal Knowledge and Information Systems*, pages 37–48. IOS Press, 2005.

[GM82] Joseph Goguen and José Meseguer. Security Policies and Security Models. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 11–20, 1982.

[HV06] Dieter Hutter and Melanie Volkamer. Information Flow Control to Secure Dynamic Web Service Composition. In J. Clark et al., eds, *Proc. of Security in Pervasive Computing*, vol. 3934 of *LNCS*, pages 196–210. Springer, 2006.

[Kno01] Konstantin Knorr. Multilevel Security and Information Flow in Petri Net Workflows. In *Proc. of Telecommunication Systems*, 2001.

[Lam73] Butler Lampson. A Note on the Confinement Problem. *Communications of the ACM*, 16(10):613–615, 1973.

[Loh07] Niels Lohmann. A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. In M. Dumas and R. Heckel, eds, *Proc. of Web Services and Formal Methods*, vol. 4937 of *LNCS*, pages 77–91. Springer, 2007.

[LVD09] Niels Lohmann, Eric Verbeek, and Remco M. Dijkman. Petri Net Transformations for Business Processes - A Survey. In K. Jensen and W. van der Aalst, eds, *Transactions on Petri Nets and Other Models of Concurrency*, vol. 5460 of *LNCS*, pages 46–63. Springer, 2009.

[MAHS10] Günter Müller, Rafael Accorsi, Sebastian Höhn, and Stefan Sackmann. Sichere Nutzungskontrolle für mehr Transparenz in Finanzmärkten. *Informatik Spektrum*, 33(1):3–13, 2010.

[ML97] Andrew Myers and Barbara Liskov. A decentralized model for information flow control. In *Proc. of ACM Symposium on Operating Systems Principles*, pages 129–142, 1997.

[NS07] Kioumars Namiri and Nenad Stojanovic. Using Control Patterns in Business Processes Compliance. In M. Weske et al., eds, *Proc. of the Workshop on Web Information Systems Engineering*, vol. 4832 of *LNCS*, pages 178–190. Springer, 2007.

[OVvdA+05] Chun Ouyang, Eric Verbeek, Wil van der Aalst, Stephan Breutel, Marlon Dumas, and Arthur ter Hofstede. WofBPEL: A Tool for Automated Analysis of BPEL Processes. In B. Benatallah et al., eds, *Proc. of Service-Oriented Computing*, vol. 3826 of *LNCS*, pages 484–489. Springer, 2005.

[PN92] Norman Proctor and Peter Neumann. Architectural implications of covert channels. In *Proc. of National Computer Security Conference*, pages 28–43, 1992.

[SM03] Andrei Sabelfeld and Andrew Myers. Language-Based Information-Flow Security. *IEEE Journal on Selected Areas in Communications*, 21(1):5–19, 2003.

[SS05] Andrei Sabelfeld and David Sands. Dimensions and Principles of Declassification. In *Proc. of IEEE Computer Security Foundations Workshop*, pages 255–269. IEEE 2005.

[SZNS06] Sherry Sun, Leon Zhao, Jay Nunamaker, and Olivia Liu Sheng. Formulating the Data-Flow Perspective for Business Process Management. *Information Systems Research*, 17(4):374–391, 2006.

[TvdAS09] Nikola Trčka, Wil van der Aalst, and Natalia Sidorova. Data-Flow Anti-patterns: Discovering Data-Flow Errors in Workflows. In P. van Eck et al., eds, *Proc. of the Conference on Advanced Information Systems Engineering*, vol. 5565 of *LNCS*, pages 425–439. Springer, 2009.

[WH10] Celia Wolf and Paul Harmon. The State of Business Process Management. BPTrends Report, 2010. (Available at `http://www.bptrends.com/`.)