Information Flow Security for Business Process Models - just one click away

Andreas Lehmann[1] and Dirk Fahland[2]

[1] University of Rostock, Germany
andreas.lehmann@uni-rostock.de
[2] Eindhoven University of Technology, The Netherlands
d.fahland@tue.nl

Abstract. When outsourcing tasks of a business process to a third party, information flow security becomes a critical issue. In particular *implicit* information leaks are an intriguing problem. Given a business process one could ask whether the execution of a *confidential* task is kept secret to a third party which can observe some *public* (nonconfidential) tasks. A business process is secure in sense of implicit information flow if a third party can not deduce the execution of confidential tasks based on observations of public tasks. We will show that we can verify much faster whether a given process model is secure, support a new information flow property, and support the modeler to create a secure process using a graphical modeling tool. The demo might be interesting for all process modelers and those who are concerned with security in the BPM community.

1 Introduction

When outsourcing certain tasks of a business process to third-party organizations one could "leak" sensitive information (e. g., customer data, trade secrets, or financial details) to the involved third parties. This is undesirable, be it for legal or economic reasons. Information flow security concerns about such leaks which are called *interferences*, so the absence of such information leaks is called *noninterference* [6]. A standard approach to model information flow security is to label *all* tasks of a business process as either confidential or public, such a labeling is called a *complete assignment*. Given a complete assignment one could verify whether a given process is secure; however existing tools [3, 9] fail to verify industrial business processes [2]. Additionally, creating a complete assignment is cumbersome and any found interference requires a corrected and again complete assignment. We provide a solution for both problems: (1) the tool *Anica* is able to verify industrial business processes using a decomposition strategy, and (2) the modeling tool *Seda* permits a modeler to specify *partial assignments* which are automatically completed and verified by Anica. Altogether this provides modeling support and instant feedback for guaranteed noninterference.

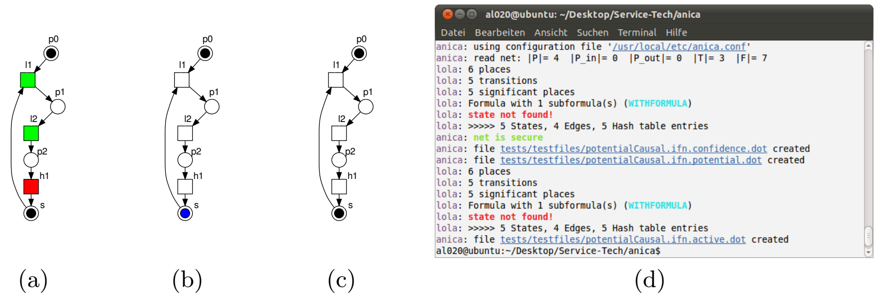
In the rest of the paper we explain the tools *Anica* and *Seda* and report on experimental evaluation with over 550 industrial business processes [8] putting secure business processes just one click away.

2 Features

Verification. *Anica* (Automated Non-Interference Check Assistant) verifies the structural Place-based Noninterference properties PBNI+ [4] and PBNID [10] for safe Petri nets with complete assignments. Basically both, PBNI+ and PBNID, characterize noninterference violations in specific places of the process model where confidential information could be leaked to the public domain. In addition, PBNID offers additional *downgraders* (of confidentiality), which permit controlled information flow from the confidential to the public domain. This way intransitive noninterference requirements can be expressed. Although noninterference properties are defined structurally they require verification on the process behavior. Potential interferences can be identified on the net structure, the decision whether a potential interference is an active one is a verification problem on the behavior of the net; see [2, 11] for details.

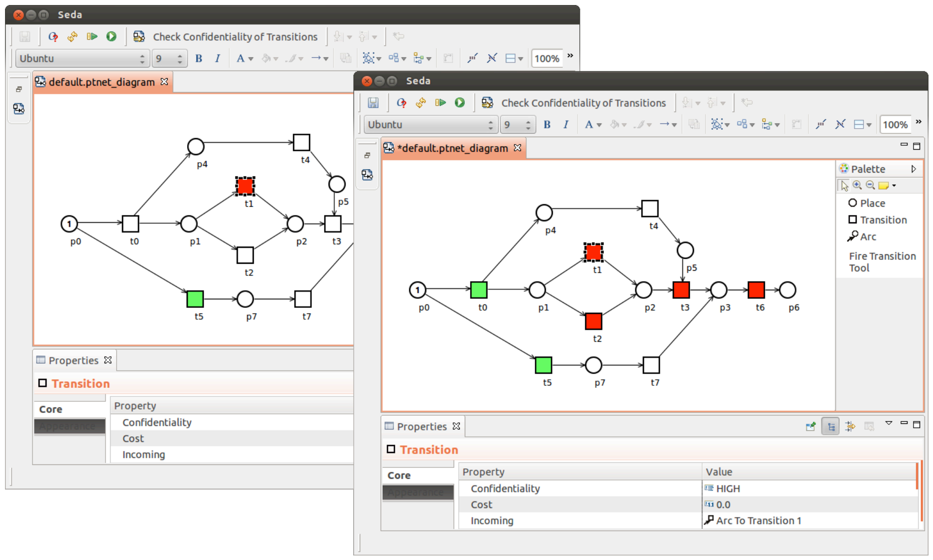
Noninterference verification is not limited to Petri net-based process models. We have shown in several evaluations [2, 11] that many modeling languages, such as WS-BPEL, BPMN, and EPC can be translated to Petri nets [12], making our technique available to high-level languages as well. Also, safe nets are no restriction under the assumption of sound [1] process models (which are bounded and hence can also be represented as safe Petri nets).

To verify PBNI+ or PBNID a completely assigned and safe Petri net model of the business process is necessary. Existing tools first compute the complete state space of the net and then search for information leaks making the approach fail on large industrial business processes. Our command line interface (CLI)-based tool *Anica* instead decomposes the noninterference verification into many, typically smaller *reachability problems* [2] which can be verified using state-of-the-art model checkers using state space reduction techniques; we use *LoLA* [13]. As each noninterference violation is expressed in a specific place, those places are used to decompose the verification problem. Besides the main result *Anica* provides the following outputs: (1) colored dot files of the original assignment, (2) the found interferences together with (3) a detailed result file (certificate) and (4) a witness path (generated by *LoLA*) for each active interference. A typical industrial business process is verified in about 24 ms [2] using *Anica* and *LoLA*.


p0
l1
p1
l2
p2
h1
s
p0
l1
p1
l2
p2
h1
s
p0
l1
p1
l2
p2
h1
s
al020@ubuntu: ~/Desktop/Service-Tech/anica
Datei Bearbeiten Ansicht Suchen Terminal Hilfe
(a)
(b)
(c)
(d)

```
anica: using configuration file '/usr/local/etc/anica.conf'
anica: read net: |P|= 4  |P_in|= 0  |P_out|= 0  |T|= 3  |F|= 7
lola: 6 places
lola: 5 transitions
lola: 5 significant places
lola: Formula with 1 subformula(s) (WITHFORMULA)
lola: state not found!
lola: >>>>> 5 States, 4 Edges, 5 Hash table entries
anica: net is secure
anica: file tests/testfiles/potentialCausal.ifn.confidence.dot created
anica: file tests/testfiles/potentialCausal.ifn.potential.dot created
lola: 6 places
lola: 5 transitions
lola: 5 significant places
lola: Formula with 1 subformula(s) (WITHFORMULA)
lola: state not found!
lola: >>>>> 5 States, 4 Edges, 5 Hash table entries
anica: file tests/testfiles/potentialCausal.ifn.active.dot created
al020@ubuntu:~/Desktop/Service-Tech/anica$
```

Fig. 1. Example output of Anica.



Property	Value
Confidentiality	HIGH
Cost	0.0
Incoming	Arc To Transition 1

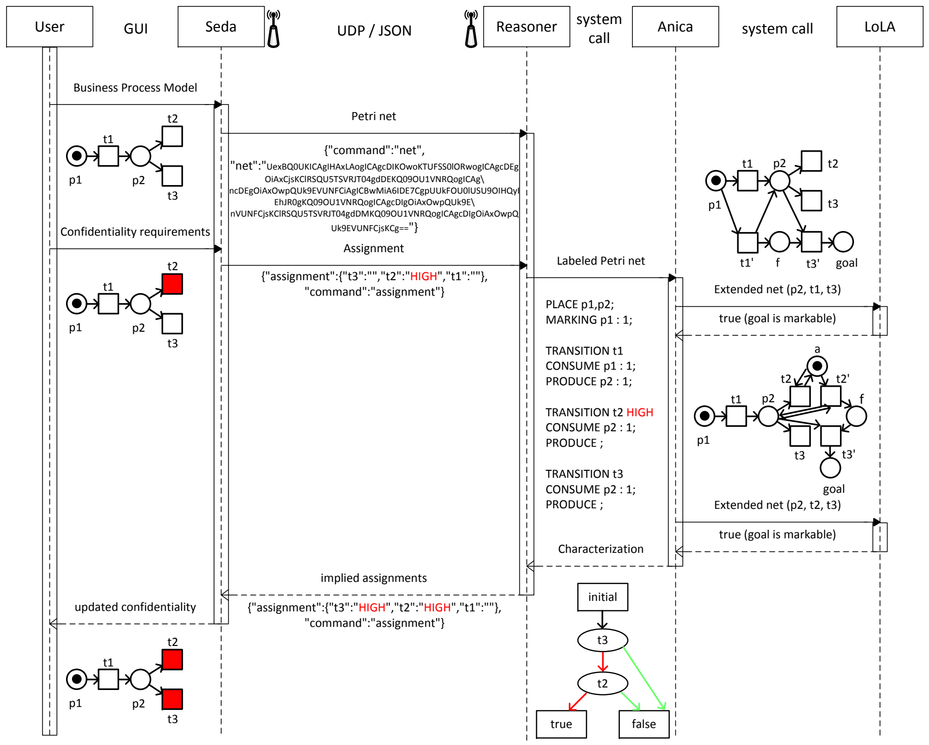
Fig. 2. Screen shots of Seda. After assigning a few transitions (left), implied assignments are calculated automatically (right).

Figure 1 shows some example dot files generated by *Anica*. The green colored tasks $l1$ and $l2$ in Fig. 1(a) are the public tasks and the red colored task $h1$ is a confidential one. As shown in Fig. 1(b) there is a potential interference in which the blue colored place s is involved (as executing $l1$ again would allow to infer that $h1$ was executed). By Fig. 1(c) this potential inferences is not active (as $l1$ can only be executed once), otherwise the place s would also be colored blue. Therefore the Petri net used in this example is secure according to the noninterference property PBNI+. A typical verification run of Anica is shown in Fig. 1(d).

Modeling support. Pure verification is often uninteresting in practical situations: a modeler would have to mark each task either as public or confidential, check interference, and if necessary reassign. This is infeasible for industrial business processes with hundreds of tasks which permit an exponential number of assignments (2^t assignments for t tasks). Rather, a modeler can create a *partial assignment* of some definitely confidential tasks and some safely public tasks, leaving other tasks *unassigned*. However, security verification requires a complete assignment.

To support the modeler in this situation, we extended our verification tool *Anica* with a so called *reasoner*, which communicates between *Anica* and the graphical editor *Seda*, an open source Eclipse-based Petri net modeling tool[3]. Seda offers the usual functionality to model and simulate Petri nets, and was extended to label each transition with a confidentiality, see Fig. 2 (left). Public tasks are labeled green, confidential tasks red, unassigned tasks are shown

[3] `http://service-technology.org/seda` (Version 1.1.3).



```
{"command":"net",
"net":"UexBQ0UKICAgIHAxLAogICAgcDIKOwoKTUFSS0lORwogICAgcDEg
OiAxCjsKClRSQU5TSVRJT04gdDEKQ09OU1VNRQogICAg\
ncDEgOiAxOwpQUk9EVUNFCiAgICBwMiA6IDE7CgpUUkFOU0lUSU9OIHQyI
EhJR0gKQ09OU1VNRQogICAgcDIgOiAxOwpQUk9E\
nVUNFCjsKClRSQU5TSVRJT04gdDMKQ09OU1VNRQogICAgcDIgOiAxOwpQ
Uk9EVUNFCjsKCg=="}
```

```
{"assignment":{"t3":"","t2":"HIGH","t1":""},
"command":"assignment"}
```

```
PLACE p1,p2;
MARKING p1 : 1;

TRANSITION t1
CONSUME p1 : 1;
PRODUCE p2 : 1;

TRANSITION t2 HIGH
CONSUME p2 : 1;
PRODUCE ;

TRANSITION t3
CONSUME p2 : 1;
PRODUCE ;
```

```
{"assignment":{"t3":"HIGH","t2":"HIGH","t1":""},
"command":"assignment"}
```

Fig. 3. Example message exchange in the current architecture.

white, i.e., in Fig. 2 (left) $t1$ is confidential, $t5$ public and the rest unassigned. The modeler may now “*Check Confidentiality of Transitions*“ using a respective button which invokes the *reasoner* and *Anica*. If the current partial assignment is insecure, the modeler gets instant feedback. If the current partial assignment is secure, it is automatically extended to all other transitions that must be set to ensure the assignment chosen by the modeler, e.g., in Fig. 2 (right) four transitions were assigned to secure confidentiality of $t1$. This way the effort for the modeler is reduced and the modeler gets feedback within a second allowing for a tight integration of modeling and security verification. Still unassigned transitions can be chosen freely by the modeler. A screen cast demonstrating the modeling support is available.[4]

Architecture and implementation details. Our integration of verification in a modeling tool also has an interesting software engineering aspect. The *reasoner* acts as a middleware between the CLI-based verification tool *Anica* written in C++ and Seda written in Java based on Eclipse. In our architecture the *reasoner*, written in C++, communicates via UDP packages based on JSON which allows to use different workstations for verification and for modeling.

[4] `http://youtu.be/L7mbIHkGb7A`

Figure 3 depicts an example message exchange for a modeling session. First, the user creates a model of the business process in Seda. The model is sent to the reasoner via UDP (using Base64 encoding and JSON). After the user has assigned the confidentiality requirements (typically a partial assignment) and pressed the button "*Check Confidentiality of Transitions*", Seda sends the assignment as JSON encoded UDP package to the reasoner as well. The user expects a completed and secure assignment for the business process model. The reasoner calculates this by the help of Anica as follows. The reasoner labels the transitions of the Petri net with the given confidentiality values and hands this labeled net to Anica via a system call. For each potential noninterference violation Anica creates an extended plain Petri net, which is given to the model checker LoLA. LoLA performs a reachability check for a specific place, called "*goal*". LoLA returns true, if and only if a reachable marking (from the initial marking) exists, in which "*goal*" is marked. Based on all of LoLA's results, Anica creates a characterization of all secure assignments (encoded as a BDD) and returns it to the reasoner. From this, the reasoner infers implied assignments of currently unassigned transitions, thus deriving a secure assignment which is sent as JSON encoded UDP package to Seda. Finally, Seda colors the previously unassigned transitions according to the received message. The modeler is free to choose confidentiality values of the remaining unassigned transitions [2, 11].

3 Evaluation

Despite being a young discipline within BPM, there exist already two other tools to check PBNI+: Frau et al. implemented PNSC [9] and Accorsi et al. developed SWAT [3]. Both tools do not scale well for large Petri net models, because they require to construct and explore the complete state space of a process model suffering state space explosion. By decomposing the problem into reachability checks and by applying state-space reduction techniques, *Anica* verifies models much faster and consumes less memory [2].

We validated our technique in an experiment on verifying noninterference of industrial process models [8]. We could reduce the number of states to check from more than 30 billion to about 62,000 states. The average time consumption of 24 ms contrasts to several hours for the approach used by both other tools [2]. Additionally, we offer - to the best of our knowledge - the only tool which can verify the property PBNID as well. When used in the modeling support scenario in combination with Seda and Anica, a partial assignment could be checked and extended in about 90 ms for an average and in about 2 seconds for the largest process of [8], demonstrating its feasibility for industrial business processes [11]. Furthermore *Anica*, the *reasoner* and *Seda* are publicly available[5] in contrast to PNSC [9] and SWAT [3].

[5] `http://service-technology.org/anica`

4 Conclusion

Lessons learnt. Our approach to decompose a verification problem into many typically smaller problems makes the verification of noninterference applicable for industrial business processes. Furthermore the achieved speed allows a model support based on partial assignment which are typically unusable for pure verification tasks and tools, but as much more interesting for practical domains.

Future work. The next step could be the integration of the noninterference checks in the context of other business process modeling tools (e.g, Oryx [5]) to support BPMN rather than Petri nets. This would require a new reasoner to communicate between *Anica* and for instance Oryx together with a background translation between BPMN and Petri nets [7]. Moreover the current error message in case of an insecure assignment can be extended to a detailed diagnostic information. Therefore the witness path (already provided by Anica and LoLA) could be visualized in Seda using its simulation feature.

Acknowledgement. The authors cordially thank Niels Lohmann for his ideas about the reasoner. This work was partially funded by the German Research Foundation in the project WS4Dsec in the priority program Reliably Secure Software Systems (SPP 1496).

References

1. Aalst, W.M.P.v.d.: The application of Petri nets to workflow management. Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
2. Accorsi, R., Lehmann, A.: Automatic information flow analysis of business process models. In: BPM 2012. pp. 172–187. LNCS 7481, Springer (2012)
3. Accorsi, R., Wonnemann, C., Dochow, S.: SWAT: A security workflow toolkit for reliably secure process-aware information systems. In: ARES 2011. pp. 692–697. IEEE (2011)
4. Busi, N., Gorrieri, R.: Structural non-interference in elementary and trace nets. Mathematical Structures in Computer Science 19(6), 1065–1090 (2009)
5. Decker, G., Overdick, H., Weske, M.: Oryx - an open modeling platform for the bpm community. In: BPM 2008. pp. 382–385. LNCS 5240, Springer (2008)
6. Denning, D.E., Denning, P.J.: Certification of programs for secure information flow. Commun. ACM 20(7), 504–513 (1977)
7. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in bpmn. Information & Software Technology 50(12), 1281–1294 (2008)
8. Fahland, D., Favre, C., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Analysis on demand: Instantaneous soundness checking of industrial business process models. Data Knowl. Eng. 70(5), 448–466 (2011)
9. Frau, S., Gorrieri, R., Ferigato, C.: Petri net security checker: Structural non-interference at work. In: FAST 2008. pp. 210–225. LNCS 5491, Springer (2008)
10. Gorrieri, R., Vernali, M.: Foundations of security analysis and design vi. chap. On intransitive non-interference in some models of concurrency, pp. 125–151. Springer (2011)
11. Lehmann, A., Lohmann, N.: Modeling wizard for confidential business processes. In: SBP 2012. Tallinn, Estonia (2012)
12. Lohmann, N., Verbeek, H., Dijkman, R.M.: Petri net transformations for business processes – a survey. LNCS ToPNoC II(5460), 46–63 (2009)
13. Wolf, K.: Generating Petri net state spaces. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS 4546, vol. 4546, pp. 29–42. Springer-Verlag (2007)