Online Compliance Monitoring of Service Landscapes

J.M.E.M. van der Werf[1] and H.M.W. Verbeek[2]

[1] Department of Information and Computing Science,
University of Utrecht, The Netherlands
J.M.E.M.vanderWerf@UU.nl
[2] Department of Mathematics and Computer Science,
Eindhoven University of Technology, Eindhoven, The Netherlands
H.M.W.Verbeek@TUE.nl

Abstract. Today, it is a challenging task to keep a service application running over the internet safe and secure. Based on a collection of security requirements, a so-called *golden configuration* can be created for such an application. When the application has been configured according to this golden configuration, it is assumed that it satisfies these requirements, that is, that it is safe and secure. This assumption is based on the best practices that were used for creating the golden configuration, and on assumptions like that nothing out-of-the-ordinary occurs. Whether the requirements are actually violated, can be checked on the traces that are left behind by the configured service application. Today's applications typically log an enormous amount of data to keep track of everything that has happened. As such, such an event log can be regarded as the ground truth for the entire application: A security requirement is violated if and only if it shows in the event log. This paper introduces the ProMSecCo tool, which has been built to check whether the security requirements that have been used to create the golden configuration are violated by the event log as generated by the configured service application.

1 Introduction

The introduction of new internet architectures like Service Oriented Architectures and Software as a Service, allows organisations to cooperate via the internet. Organisations deliver their businesses by offering services. These services may be delivered by the organisation itself, or may be composed out of services offered by other suppliers. Consequently, the service landscape of organisations become more diffuse and hence, more complex.

The service provider has to show that it fulfills its obligations to its clients. For this, many certifications and procedures exist, such as the ISO standards 27001 and 17799 [7]. Auditing is the task of checking whether all agreed obligations are adhered to. The main task within an audit is to validate whether the service landscape is configured correctly, i.e., whether all obligations have been translated into safety and security requirements, and whether these requirements have been implemented correctly. Different approaches to validate correctness exist. For example, the requirements can be embedded in the development process [11], or the process of configuring the service landscape can be automated, as proposed in [4, 5], resulting in a *golden configuration*.

The golden configuration defines an ideal set of configurations that complies with all security requirements and that implements these in a cost-efficient way. This golden configuration is used to derive the actual service landscape configuration. The actual configuration of the service landscape is then stored in a configuration management system (CMS).

In the PoSecCo project [3], the chain of defining high-level business agreements to the creation of the appropriate configuration files for a service landscape has been automated [6]. Typically, these configuration files concern controls that implement and monitor the safety and security constraints. Implementing controls is however not always feasible, nor is it guaranteed that the configuration files have been changed between two audits. To overcome this, we propose in the PoSecCo project not only to automate the creation of the golden configuration, but to complement the approach with the ProMSecCo tool, that analyzes *execution data* generated by the running service landscape.

During its execution, a service landscape produces execution data that records its usage, such as users that logged in and the actions performed by the different components on the landscape. This omnipresence of execution data, coupled with process mining techniques enable a new form of auditing: *continuous auditing* [3, 14], in which the execution data is used to monitor and detect compliance violations.

The remainder of this paper is organized as follows. Section 2 explains how semantic process mining techniques can support audits in a service landscape. In Sec. 3 we present the ProMSecCo tool, and its integration in service landscapes. Sec. 4 concludes the paper.

2 Semantic Process Mining

Process mining [1] analyzes execution data in the form of event logs. As proposed in [2, 3, 8, 9], process mining can support the auditor towards continuous auditing.

Most of the security requirements are expressed in high-level terms, on the level of the business services, whereas the execution data resides on the lowest level, at the infrastructure. In order to automatically check such high-level requirements, we need to bridge the gap between the different layers of abstraction. To do so, we need to relate elements in the event logs to concepts at higher levels of abstractions. For this, we combine techniques from the semantic web, like ontologies with process mining techniques. Semantic process mining defines *annotation rules* that relate elements from the event log with elements from other sources [14].

To use process mining techniques for checking compliance of a service landscape, its execution data has to be transformed into event logs. Then, based on the model that is used to configure the service landscape, called the PoSecCo model, the event logs can be analyzed to check compliance. Fig. 1 depicts the central idea of using process mining in service landscapes.

As an example, consider a Segregation-of-Duty (SoD) constraint that, in short, specifies that a single user should not hold both an application administrator role and a

[3] European FP7 project on POlicy and SECurity COnfiguration management, see http://www.posecco.eu/

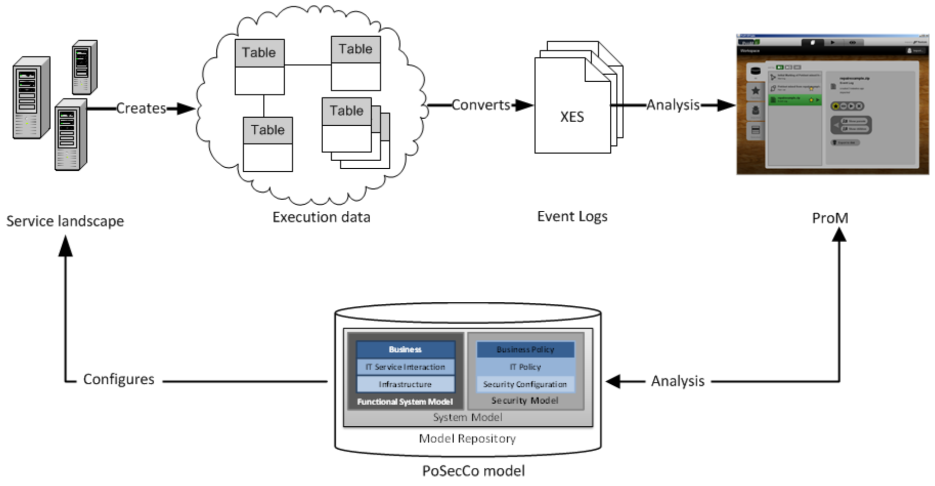


Fig. 1. Relation between configuration, execution data and process mining

database administrator role. These aspects are part of the PoSecCo model, together with the golden configuration of the service landscape. From this model we obtain information which files should be monitored, such as configuration files. During the execution of the service landscape, user access to these files is monitored and regularly exported for analysis of the constraint.

Assume that we obtained system execution data in the form of an event log [4] from the actual service landscape, and that Tbl. 1 represents this event log. The event log depicted in Tbl. 1 consists of 4 cases, with events for two activities, "Edit /etc/-tomcat6/server.xml" and "Edit /etc/my.cnf". There are three users that executed these events: "frank", "gerard" and "hank".

In a straightforward way, we can check the SoD constraint on an event log. The assumptions made on the event log are that it contains (1) a trace for every relevant file (like "/etc/tomcat6/server.xml") and (2) an event for every access to these files, where (3) the user who has accessed the file is recorded by the event.

[4] The event log and other files used can be downloaded from http://www.promtools.org/prom6/PoSecCo

Table 1. Partial event log from system landscape

Case	Activity	User
1	Edit /etc/tomcat6/server.xml	frank
1	Edit /etc/my.cnf	gerard
2	Edit /etc/tomcat6/server.xml	frank
3	Edit /etc/my.cnf	gerard
4	Edit /etc/my.cnf	gerard
4	Edit /etc/tomcat6/server.xml	hank

Based on these assumptions, we can relate the elements of the event log to elements in the PoSecCo model. As each trace contains events of a file, we map the traces to the concept “File”. A “File” has a “filename”, which is stored in the event log as the trace attribute “concept:name”. Each event indicates a file update or file read by some user. Thus, we can use the “org:resource” attribute of the event to indicate whether a user read or writes a file. This results in a set of additional annotation rules, including:

AR 1 (User accesses a file) Inverse of “File accessed by user” rule

source	`//event/string[@key='org:resource']`
relation	`http://www.posecco.eu/ontologies/landscape/file.owl#accesses`
target	`ancestor::trace`

AR 2 (File “/etc/my.cnf” configures DB) This file is a configuration file of the DB.

source	`//trace[./string[@key='concept:name' and @value='/etc/my.cnf']]`
relation	
target	`http://www.posecco.eu/ontologies/landscape/file.owl#DB`

Using all annotation rules on the event log, we can check the SoD-constraint by checking the following query on the ontology:

$$\text{User } \mathbf{and} \text{ (accesses } \mathbf{some} \text{ DB) } \mathbf{and} \text{ (accesses } \mathbf{some} \text{ APP)}$$

Clearly, any non-empty result of the defined query violates the SoD constraint, as no user should be able to do so.

3 ProMSecCo

The approach has been implemented in the tool ProMSecCo [5], which has been implemented in ProM 6 [13] as a coherent collection of packages, among which the PoSecCo package. The tool has been developed in the context of the PoSecCo project to support semantic process mining, as introduced in [12, 14]. The tool extends the functionality of ProM 6 by reading and writing context models in the form of ontologies, relating these context models with the data elements in event logs, and analyzing these enriched event logs on conformance and compliance.

In the context of PoSecCo, ProMSecCo has been integrated into a log-collecting architecture, as introduced in [3, 10]. This architecture can support the knowledge user (in this project this would be the auditor) by (partly) automating frequently occurring tasks, such as checking whether certain constraints are not violated. The log-collecting architecture assists in the following tasks:

1. regularly download system execution data from the service landscape;
2. generate event logs for the different properties; and
3. calculate the results of each of the properties.

[5] ProMSecCo can be downloaded from http://www.promtools.org/prom6/PoSecCo/.

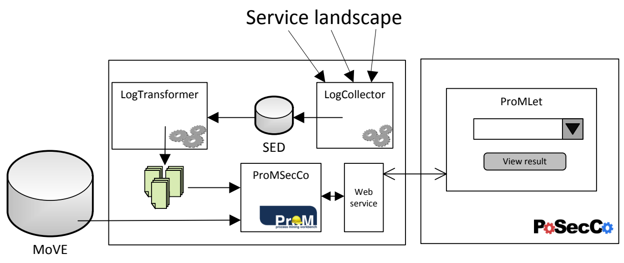
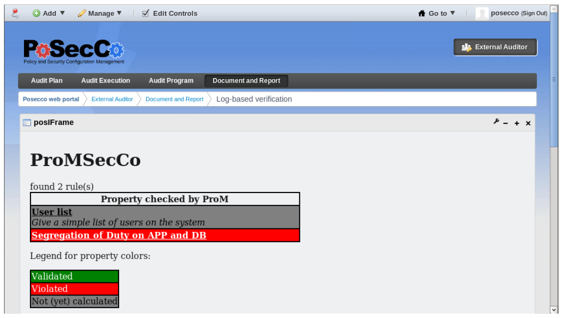


Fig. 2. ProM Deployment in PoSecCo environment



ProMSecCo

found 2 rule(s)

Property checked by ProM
User list *Give a simple list of users on the system*
Segregation of Duty on APP and DB

Legend for property colors:

Validated
Violated
Not (yet) calculated

Fig. 3. Inspection of the results of the automated checking of the SoD constraint.

The proposed architecture is depicted in Fig. 2. The log collector component is responsible for retrieving and storing the system execution data. The log transformer component transforms the available system execution data into event logs that can be used within ProMSecCo. The ProMSecCo component does the semantic process mining, as explained before. The ProMLet is a basic user interface for the web service component as it is integrated in the PoSecCo Integrated Environment. It provides a user interface to show the constraints that are checked automatically, and their results.

Figure 3 shows the results of the automated checking of the SoD constraint in this environment. In this case, the SoD constraint has been violated. By clicking on the link "Segregation of Duty on APP and DB", additional diagnostic information is provided to the end user (like who has violated this constraint, and when has it been violated). Using this diagnostic information, the end user can either inspect the corresponding event log and gain additional diagnostic information, or can inspect the raw execution data for this violation.

4 Conclusions

Auditing an entire service landscape, i.e., checking whether it complies to all regulations and agreements, is known to be a hard and time consuming task. Ensuring that the service landscape is configured correctly is an important task within auditing, but not sufficient to guarantee compliance, as configurations might have been temporarily changed in between two audits.

To overcome this risk, we present in this paper the ProMSecCo tool. ProMSecCo analyzes execution data generated by the service landscape in the form of event logs using semantic process mining techniques.

Structured log collection is essential for the use of ProMSecCo within auditing. Therefore, we propose in this paper a log-collecting architecture to integrate the tool within a service landscape. In this way, ProMSecCo allows to automate the compliance checking process, which is an essential next step towards continuous auditing.

References

1. W.M.P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, Berlin, 2011.
2. W.M.P. van der Aalst, K.M. van Hee, J.M.E.M. van der Werf, and M. Verdonk. Auditing 2.0: Using Process Mining to Support Tomorrow's Auditor. *IEEE Computer*, 43(3):102–105, 2010.
3. W.M.P. van der Aalst, K.M. van Hee, J.M.E.M. van der Werf, A. Kumar, and M.C. Verdonk. Conceptual model for on line auditing. *Decision Support Systems*, 50(3):636 – 647, 2011.
4. W. Arsac, A. Laube, and H. Plate. Policy Chain for Securing Service Oriented Architectures. In *Data Privacy Management and Autonomous Spontaneous Security*, volume 7731 of *LNCS*, pages 303 – 317. Springer, Berlin, 2013.
5. M. Bezzi, E. Damiani, S. Paraboschi, and H. Plate. Integrating Advanced Security Certification and Policy Management. In *Cyber Security and Privacy*, volume 182 of *Communications in Computer and Information Science*, pages 55 – 66. Springer, Berlin, 2013.
6. M.M. Casalino, M. Mangili, H. Plate, and S.E. Ponta. Detection of Configuration Vulnerabilities in Distributed (Web) Environments. In *Security and Privacy in Communication Networks*, volume 106 of *Lecture Notes of the Institute for Computer Science, Social Informatics and Telecommunications Engineering*, pages 131 – 148. Springer, Berlin, 2013.
7. D.A. Haworth and L. R. Pietron. Sarbanes-Oxley: Achieving compliance by starting with ISO 17799. *Information Systems Management*, 23(1):73–87, 2006.
8. M. Jans, N. Lybaert, K. Vanhoof, and J.M.E.M. van der Werf. Business process mining for internal fraud risk reduction: results of a case study. In *9th International Research Symposium on Accounting Information Systems, Paris*, 2008.
9. M. Jans, J.M.E.M. van der Werf, N. Lybaert, and K. Vanhoof. A business process mining application for internal transaction fraud mitigation. *Expert Systems With Applications*, 38(10):13351–13359, 2011.
10. J.H.W. van Loon. Design of a monitor for on-the-fly checking of business rules. Master's thesis, Technische Universiteit Eindhoven, 2011.
11. M.A. Neri, M. Guarnieri, E. Magri, S. Mutti, and S. Paraboschi. A model-driven approach for securing software architectures. In *SECRYPT 2013*, pages 595 – 602. SciTePress, 2013.
12. PoSecCo. D4.3 – Tailoring Semantic Process Mining Methods to Behavioral Landscape Models, 2011.
13. H. M. W. Verbeek, J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. XES, XESame, and ProM 6. In *Information System Evolution*, volume 72, pages 60–75. Springer, 2011.
14. J.M.E.M. van der Werf, H.M.W. Verbeek, and W.M.P. van der Aalst. Context-aware compliance checking. In *Business Process Management - 10th International Conference, BPM 2012*, volume 7481 of *LNCS*, pages 98 – 113. Springer, Berlin, 2012.