Risk-based Analysis of Business Process Executions

Mahdi Alizadeh
Eindhoven University of Technology
m.alizadeh@tue.nl

Nicola Zannone
Eindhoven University of Technology
n.zannone@tue.nl

ABSTRACT

Organizations need to monitor their business processes to ensure that what actually happens in the system is compliant with the prescribed behavior. Deviations from the prescribed behavior may correspond to violations of security requirements and expose organizations to severe risks. Thus, it is crucial for organizations to detect and address nonconforming behavior as early as possible. In this paper, we present an auditing framework that facilitates the analysis of process executions by detecting nonconforming behaviors and ranking them with respect to their criticality. Our framework employs conformance checking techniques to detect possible explanations of nonconformity. Based on such explanations, the framework assesses the criticality of nonconforming process executions based on historical logging data and context information.

CCS Concepts

•**Security and privacy** → **Security services;** *Information accountability and usage control;*

Keywords

Auditing, Risk Assessment, Alignments, Conformance Checking.

1. INTRODUCTION

Organizations are often exposed to a wide range of security incidents. These incidents might harm a company's reputation, adversely affect its clients and result in a significant financial loss. In response to a number of scandals like the Enron and HIH Insurance cases, a number of regulations and guidelines such as Sarbanes-Oxley Act, Basel III and COBIT, have been enacted. These regulations mandate organizations to have frameworks in place for auditing and managing operational risks.

Organizations often use process models to define activities and procedures to reach their business goals. However, in most organizations process models are not used to enforce a particular way of working. Thus, in practice, reality may deviate from the prescribed behavior. These deviations can correspond to infringements of security policies and have severe consequences for an organization. Therefore, organizations should be able to detect and analyze them as early as possible and take proper actions to mitigate their impact.

Auditing mechanisms currently in use are mostly human-driven and appear to be grossly ineffective. The main problem is that manual auditing of process executions is costly, error-prone and time consuming. In addition, it is not transparent (i.e., not replicable) as the decision whether a certain situation corresponds to an infringement depends on the judgment of the auditor. Moreover, organizations typically have limited resources and cannot deal with all the risks to which they are exposed. It is widely accepted that measuring the criticality of nonconforming behavior gives the opportunity to investigate the most critical infringements earlier. However, the analysis of process executions is not a trivial task because several process perspectives (e.g., control flow, data and users) should be taken into account.

In this work, we poses the basis for the definition of a risk-based auditing framework to assist security analysts in the detection of nonconformity in process executions and in assessing its criticality. To detect deviations in process executions, we rely on alignment-based conformance checking [1, 3]. Alignments provide explanations of nonconformity by pinpointing what went wrong in a process execution. However, alignments are often constructed using a predefined cost function, which can lead to incorrect diagnostics [1]. In particular, the underlying assumption is that the alignments with minimal cost (with respect to an arbitrary cost function) always provide the most probable explanations of nonconformity. We discard this assumption and compute the risks posed by nonconforming process executions by considering both the likelihood that an alignment reflects what actually happened in the system and the severity of the deviations identified by the alignment. Assessing the risk posed by nonconforming process executions makes it possible to rank such executions with respect their criticality, thus enabling security analysts to focus on the most severe incidents.

2. APPROACH

Process executions may deviate from the prescribed behavior. Organizations should be able to detect and analyze nonconforming behavior as earlier as possible to promptly react to security incidents and, thus, limit their impact. In particular, they should be able to prioritize security incidents with respect to their criticality. To assist analysts in the analysis of process executions, we propose a risk-based auditing framework. The framework allows the detection of nonconformity in process executions and the assessment of its criticality. Fig. 1 presents an overview of the framework.

The first step of the framework is to detect nonconforming behaviors (*Conformance Checking*). To this end, we rely on alignment-based conformance checking. Intuitively, an alignment relates events in an event log to activities in a process model and vice versa, thus

CODASPY'16, March 09-11, 2016, New Orleans, LA, USA.

ACM ISBN 978-1-4503-3935-3/16/03.

DOI: http://dx.doi.org/10.1145/2857705.2857742

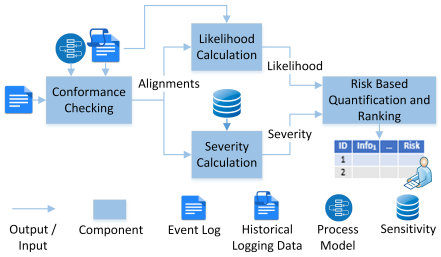

Figure 1: Risk-based Auditing Framework

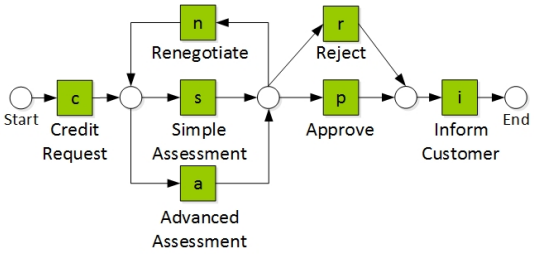

Figure 2: A process model for handling credit requests

$$\gamma_1 = \begin{array}{|c|c|c|c|}\hline c & \gg & p & i \\ \hline c & s & p & i \\ \hline \end{array} \qquad \gamma_2 = \begin{array}{|c|c|c|c|c|}\hline c & \gg & p & \gg & i \\ \hline c & s & \gg & r & i \\ \hline \end{array}$$

$$\gamma_3 = \begin{array}{|c|c|c|c|}\hline c & \gg & p & i \\ \hline c & a & p & i \\ \hline \end{array} \qquad \gamma_4 = \begin{array}{|c|c|c|c|c|}\hline c & \gg & p & \gg & i \\ \hline c & a & \gg & r & i \\ \hline \end{array}$$

Figure 3: Alignments between the net in Fig. 2 and $\sigma_1 = \langle c, p, i \rangle$

pinpointing the nonconformity between a process execution and a process model.

The criticality of nonconforming process executions is assessed by computing the operational risks posed by such process executions. In particular, we compute the likelihood that an alignment provides the actual explanation of nonconformity and the severity of the deviations pinpointed in the alignment. The likelihood of an alignment is determined with respect to historical logging data, i.e. past process executions (*Likelihood Calculation*). The severity of an alignment is computed on the basis of the information available about the process execution, e.g. the executed activity, the user who executed it and data that were accessed (*Severity Calculation*).

The risk associated to an alignment is computed by combining its likelihood and severity (*Risk Based Quantification and Ranking*). It is worth noting that more than one explanation of nonconformity may exist for a given process execution. To this end, we compute the possible alignments for a process execution and aggregate the corresponding risks. The aggregated risk represents the operational risk posed by the process execution.

Nonconforming process executions are ranked and visualized with respect to their criticality (i.e., risk level). A security analyst, thus, can evaluate them focusing on the most severe incidents. Based on this analysis, organizations can take the appropriate actions to mitigate the losses. In the remainder of the section, we describe the main components of our framework.

2.1 Conformance Checking

We use alignment-based conformance checking [1, 3] to detect the possible explanations of nonconformity between process executions and the prescribed behavior. Alignment-based conformance checking compares the observed behavior (recorded in logs) with a process model (defining the intended system behavior). We consider process models in the form of classical Petri nets. Petri nets consist of transitions, places, and directed arcs between them. Transitions are labeled with the tasks they represent. The state of a Petri net is represented by a multi-set of tokens on the places of the net, called marking. A transition is enabled if all its input places contain at least a token. When an enabled transition is fired, a token is taken from each of its input places and a token is added to all its output places. A sequence of transitions is a complete firing sequence if firing transitions in the sequence leads from the initial marking to the final marking of the net.

Fig. 2 shows a process model for handling credit requests in form of a Petri net. A process execution starts by filing a loan request (c). Depending on the amount of the requested loan either simple (s) or advanced (a) assessment is performed. If the assessment is negative, the customer can renegotiate the loan amount (n). Otherwise, the loan application is either approved (p) or rejected (r). The process terminates by informing the customer of the decision (i).

A logged instance of a task is called *event*. The sequence of events corresponding to a process instance is called *trace*. Given a trace and a Petri net, an *alignment* maps the trace to a complete firing sequence of the net (see [3] for a formal definition of alignment). Take, for example, a trace $\sigma_1 = \langle$credit request, approve, inform customer$\rangle$. Fig. 3 shows four possible alignments between σ_1 and the net in Fig. 2. The top row of alignments shows the sequence of events in the trace; the bottom row shows a complete firing sequence of the net. Deviations are explicitly shown by columns that contain $\gg$. For example, the third column in γ_2 shows that an event occurs in the trace although it is not allowed according to the net, i.e. *move on log*. The second column in γ_1 shows that a task must occur in σ_1 according to the net, but it is absent in the trace, i.e. *move on model*. Other columns for which events match the label of transitions represent *synchronous moves*.

As shown in Fig. 3, there can be more than one alignment between a trace and a Petri net. To determine the quality of alignments, a cost is assigned to each move in the alignment. The cost of an alignment is defined as the sum of the cost of the moves forming the alignment. For instance, consider a cost function that assigns 1 to moves on log/model and 0 to synchronous moves. According to this cost function, γ_1 and γ_3 have cost 1, and γ_2 and γ_4 have cost 3.

In this work, we use the approach to compute the cost function proposed in [1]. This approach uses historical logging data to determine the cost of moves based on the probability that an activity is executed in a certain state of the process instance. The cost of an alignment, thus, provides a measure of its reliability, i.e. to what extent it is close to what actually happened. We use this measure to compute the likelihood of alignments as shown in the next section.

2.2 Likelihood Calculation

An alignment provides an explanation about the possible deviations that could have occurred during the process execution. Several explanations of nonconformity can exist between a trace and a net. However, not all these explanations are equally probable. We account for the uncertainty in an explanation of nonconformity by computing the likelihood of the corresponding alignment. In particular, the likelihood of an alignment represents how likely an alignment reflects the reality, i.e. its reliability.

To measure the likelihood of an alignment, we use the cost of the alignment obtained using the approach in [1]. Intuitively, alignments that have a low cost are more likely to represent the correct relation between a trace and a model. Let π denote the function that computes the cost of an alignment according to the approach

Alignment	Cost	Likelihood	Severity	Risk
γ_1	2	0.5	0.5	0.25
γ_2	5	0.2	1.6	0.32
γ_3	5	0.2	1.0	0.20
γ_4	10	0.1	2.1	0.21

Table 1: Computation of the risk for the alignments in Fig. 3

proposed in [1]. Given an alignment γ and historical logging data $\mathcal{L}$, the likelihood of γ is:

$$\ell(\gamma) = \frac{1}{\pi(\gamma, \mathcal{L})} \tag{1}$$

Table 1 presents the cost and likelihood of the alignments in Fig. 3. The table shows that γ_1 is the alignment with minimal cost and, thus, it provides the most probable explanation of nonconformity for trace σ_1 and the net in Fig. 2. This is reflected in Table 1 by the fact that γ_1 has the greatest likelihood.

2.3 Severity Calculation

The criticality of a process execution is determined by its impact on organizational goals. We compute this impact based on the severity of the deviations that occurred in a process execution. The severity of deviations typically depends on the application domain and context information. In particular, various business process perspectives should be taken into account to determine the impact of a deviation. For instance, the impact of a deviation might depend on the tasks diverging from the prescribed behavior, the user causing the deviation and the data accessed [2].

To measure the severity of an alignment, we need a cost function ω that assigns a non-negative cost to each move based on the (negative) impact that the move has on organizational goals. Intuitively, a move with a higher severity cost has a higher impact. The severity of an alignment is computed as the sum of the moves forming the alignment. Given an alignment $\gamma = \langle m_1, ..., m_n \rangle$, where m_i ($i \in \{1, \ldots, n\}$) is an alignment move, and a severity cost function ω, the severity of γ is:

$$\lambda(\gamma) = \sum_{i=1}^{n} \omega(m_i) \tag{2}$$

The definition of a severity cost function that takes into account context information is part of our future work. Here, we use a simple cost function (Table 2) for illustrative purposes. This cost function defines the severity of skipping a given task (move on model) or executing a given task when it was not supposed to be executed (move on log). For instance, this cost function specifies that informing a client (*i*) multiple times is considered less severe than skipping the loan assessment (*s* or *a*). Moreover, the cost function assigns cost 0 to synchronous moves. The fourth column of Table 1 presents the severity of the alignments in Fig. 3 according to the severity cost function in Table 2.

2.4 Risk-based Quantification

To compute the risk level of an alignment we combine its likelihood and severity. Given an alignment γ between a trace and a net, the risk level of γ is:

$$\rho(\gamma) = \ell(\gamma) \times \lambda(\gamma) \tag{3}$$

In practice, there can be more than one alignment between a trace and a model [1, 3]. These alignments provide different explanations of nonconformity for the trace; each explanation can exhibit different operational risks. The calculation of the risk posed by the execution of a trace should take into account these risks. Thus, we

Activity	Model Move	Log Move
Credit Request (*c*)	0.1	0.1
Simple Assessment (*s*)	0.5	0.1
Advanced Assessment (*a*)	1	0.1
Inform Customer (*i*)	0.1	0.1
Renegotiate (*n*)	0.1	0.1
Approve (*p*)	0.1	1
Reject (*r*)	0.1	0.1

Table 2: Sample severity cost function

compute the risk posed by the execution of a trace by aggregating the risk level of its alignments. Let $\gamma_1, ..., \gamma_n$ the alignments constructed between a trace σ and a net, the risk posed by σ is:

$$\bar{\rho}(\sigma) = \sum_{i=1}^{n} \rho(\gamma_i) \tag{4}$$

Intuitively, the risk posed by the execution of a trace is a weighted average of the severity of the alignments constructed between the trace and a net, where the likelihood of an alignment is used as the weight for the alignment. In our example, the risk posed by the execution of trace σ_1 is equal to 0.98.

So far, we have focused on the risk assessment for a single trace. This risk assessment process can be repeated for every nonconforming process execution. This makes it possible to rank nonconforming process executions with respect to their risk level. This way, auditors can focus on the most critical incidents.

3. CONCLUSION

This work poses the basis for a novel risk-based framework for analyzing the criticality of process executions. The framework uses alignments to obtain diagnostics about process executions. The risk of nonconforming process executions is determined as the combination of the likelihood that an explanation of nonconformity corresponds to the reality and its severity. In particular, the framework computes likelihood based on historical logging data and severity based on context information. By ranking nonconforming process executions with respect to their criticality, the framework enables a security analyst to focus on the most severe incidents.

Acknowledgments This work has been funded by the NWO Cyber Security programme under the PriCE project and the Dutch national program COMMIT under the THeCS project.

4. REFERENCES

[1] M. Alizadeh, M. de Leoni, and N. Zannone. History-based construction of log-process alignments for conformance checking: Discovering what really went wrong. In *Proceedings of International Symposium on Data-driven Process Discovery and Analysis*, CEUR Workshop Proceedings 1293, pages 1–15. CEUR-WS.org, 2014.

[2] S. Banescu and N. Zannone. Measuring privacy compliance with process specifications. In *Proceedings of International Workshop on Security Measurements and Metrics*, pages 41–50. IEEE, 2011.

[3] W. M. P. van der Aalst, A. Adriansyah, and B. F. van Dongen. Replaying history on process models for conformance checking and performance analysis. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 2(2):182–192, 2012.