

Privacy-Preserving Data Publishing in Process Mining

Majid Rafiei^[0000–0001–7161–6927] and Wil M.P. van der Aalst^[0000–0002–0955–6940]

Chair of Process and Data Science, RWTH Aachen University, Aachen, Germany

Abstract. Process mining aims to provide insights into the actual processes based on event data. These data are often recorded by information systems and are widely available. However, they often contain sensitive private information that should be analyzed responsibly. Therefore, privacy issues in process mining are recently receiving more attention. Privacy preservation techniques obviously need to modify the original data, yet, at the same time, they are supposed to preserve the data utility. Privacy-preserving transformations of the data may lead to incorrect or misleading analysis results. Hence, new infrastructures need to be designed for publishing the privacy-aware event data whose aim is to provide metadata regarding the privacy-related transformations on event data without revealing details of privacy preservation techniques or the protected information. In this paper, we provide formal definitions for the main anonymization operations, used by privacy models in process mining. These are used to create an infrastructure for recording the privacy metadata. We advocate the proposed privacy metadata in practice by designing a privacy extension for the XES standard and a general data structure for event data which are not in the form of standard event logs.

Keywords: Responsible process mining · Privacy preservation · Privacy metadata · Process mining · Event logs

1 Introduction

No one doubts that data are extremely important for people and organizations and their importance is growing. Hence, the interest in *data science* is rapidly growing. Of particular interest are the so-called event data used by the *process mining* techniques to analyze end-to-end processes. Process mining bridges the gap between traditional model-based process analysis, e.g., simulation, and data-enteric analysis, e.g., data mining. It provides fact-based insights into the actual processes using event logs [1]. The three basic types of process mining are: *process discovery*, *conformance checking*, and *enhancement* [1].

An event log is a collection of events, and each event is described by its attributes. The main attributes required for process mining are *case id*, *activity*, *timestamp*, and *resource*. Table 1 shows an event log recorded by an information system in a hospital, where \perp indicates that the corresponding attribute was not

recorded. Some of the event attributes may refer to individuals. For example, in the health-care context, the *case id* may refer to the patients whose data are recorded, and the *resource* may refer to the employees performing activities for the patients, e.g., nurses or surgeons. When the individuals' data are explicitly or implicitly included, *privacy issues* arise. According to regulations such as the European General Data Protection Regulation (GDPR) [25], organizations are compelled to consider the privacy of individuals while analyzing their data.

The *privacy* and *confidentiality* issues in process mining are recently receiving more attention [22,20,9,14]. The proposed methods cover a range of solutions from privacy/confidentiality frameworks to privacy guarantees. Privacy preservation techniques often apply some anonymization operations to modify the data in order to fulfill desired *privacy requirements*, yet, at the same time, they are supposed to preserve the *data utility*. The transformed event log may only be suitable for specific analyses. For example, in [20], the privacy requirement is to *discover social networks of the individuals (resources) involved in a process without revealing their activities*, and the resulting event log is only suitable for the *social network discovery*. Moreover, the original event log may be transformed to another form of event data which does not have the structure of a standard event log. For example, in [22], the privacy requirement is to *discover processes without revealing the sequence of activities performed for the individuals (cases)*, where the transformed event data are not in the form of event logs and contain only *directly follows relations between activities*, which is merely suitable for the *process discovery*. Therefore, the modifications made by privacy preservation techniques need to be reflected in the transformed event data to inform the data analysts.

In this paper, for the first time, we formalize the main anonymization operations on the event logs and exploit them as the basis of an infrastructure for proposing privacy metadata, we also design a privacy extension for the XES standard and a general data structure to cope with the event data generated by some privacy preservation techniques which are not in the form of an event log. The proposed metadata, along with the provided tools support, supply privacy-aware event data publishing while avoiding inappropriate or incorrect analyses.

The remainder of the paper is organized as follows. In Section 2, we explain the motivation of this research. Section 3 outlines related work. In Section 4, formal models for event logs are presented. We explain the privacy-preserving data publishing in process mining in Section 5, where we formalize the main anonymization operations, privacy metadata are proposed, and the tools support is presented. Section 6 concludes the paper.

2 Motivation

Compare Table 2 with Table 1, they both look like an original event log containing all the main attributes to apply process mining techniques. However, Table 2 is derived from Table 1 by randomly substituting some activities (f was substituted with g and k), generalizing the timestamps (the timestamps got generalized to the *minutes* level), and suppressing some resources (B1 was

Table 1: An event log (each row represents an event). Table 2: An anonymized event log (each row represents an event).

Case Id	Act.	Timestamp	Res.	Age	Disease
1	a	01.01.2019-08:30:10	E1	22	Flu
1	b	01.01.2019-08:45:00	D1	22	Flu
2	a	01.01.2019-08:46:15	E1	30	Infection
3	a	01.01.2019-08:50:01	E1	32	Infection
4	a	01.01.2019-08:55:00	⊥	29	Poisoning
1	e	01.01.2019-08:58:15	E2	22	Flu
4	b	01.01.2019-09:10:00	D2	29	Poisoning
4	r	01.01.2019-09:30:00	B1	29	Poisoning
2	d	01.01.2019-09:46:00	E3	30	Infection
3	d	01.01.2019-10:00:25	E3	32	Infection
2	f	01.01.2019-10:00:05	N1	30	Infection
3	f	01.01.2019-10:15:22	N1	32	Infection
4	e	01.01.2019-10:30:35	E2	29	Poisoning
2	f	01.02.2019-08:00:45	N1	30	Infection
2	b	01.02.2019-10:00:00	D2	30	Infection
3	b	01.02.2019-10:15:30	D1	32	Infection
2	e	01.02.2019-14:00:00	E2	30	Infection
3	e	01.02.2019-14:15:00	E2	32	Infection

Case Id	Act.	Timestamp	Res.	Age	Disease
1	a	01.01.2019-08:30:00	E1	22	Flu
1	b	01.01.2019-08:45:00	D1	22	Flu
2	a	01.01.2019-08:46:00	E1	30	Infection
3	a	01.01.2019-08:50:00	E1	32	Infection
4	a	01.01.2019-08:55:00	⊥	29	Poisoning
1	e	01.01.2019-08:58:00	E2	22	Flu
4	b	01.01.2019-09:10:00	D2	29	Poisoning
4	r	01.01.2019-09:30:00	⊥	29	Poisoning
2	d	01.01.2019-09:46:00	E3	30	Infection
3	d	01.01.2019-10:00:00	E3	32	Infection
2	g	01.01.2019-10:00:00	N1	30	Infection
3	g	01.01.2019-10:15:00	N1	32	Infection
4	e	01.01.2019-10:30:00	E2	29	Poisoning
2	k	01.02.2019-08:00:00	N1	30	Infection
2	b	01.02.2019-10:00:00	D2	30	Infection
3	b	01.02.2019-10:15:00	D1	32	Infection
2	e	01.02.2019-14:00:00	E2	30	Infection
3	e	01.02.2019-14:15:00	E2	32	Infection

suppressed). Hence, a performance analysis based on Table 2 may not be as accurate as the original event log, the process model discovered from Table 2 contains some fake activities, and the social network of resources is incomplete. The main motivation of this research is to provide concrete privacy metadata for process mining without exposing details of privacy/confidentiality techniques or the protected sensitive information so that the analysts are aware of the changes and avoid inappropriate analyses.

Process mining benefits from a well-developed theoretical and practical foundation letting us perform this research. In theory, event logs, as the input data types, have a concrete structure by the definition. In practice, the IEEE Standard for eXtensible Event Stream (XES)¹ is defined as a grammar for a tag-based language whose aim is to provide a unified and extensible methodology for capturing systems behaviors by means of event logs, e.g., Figure 1 shows the first case of the event log Table 1 in the XES format. In this paper, the XES standard will be used to show the concrete relation between the theory and practice, but the concepts are general.

3 Related Work

In process mining, the research field of confidentiality and privacy received rather little attention, although the *Process Mining Manifesto* [4] already pointed out the importance of privacy. In [2], *Responsible Process Mining* (RPM) is introduced as the sub-discipline focusing on possible negative side-effects of applying process mining. In [15], the aim is to provide an overview of privacy challenges in process mining in human-centered industrial environments. In [7], a possible approach toward a solution, allowing the outsourcing of process mining while ensuring the confidentiality of dataset and processes, has been presented. In [16], the authors propose a privacy-preserving system design for process mining, where a user-centered view is considered to track personal data. In [22,23], a framework is introduced, which provides a generic scheme for confidentiality in process

¹<http://www.xes-standard.org/>

```

<?xml version="1.0" encoding="UTF-8" ?>
<log xes.version="1.0" xes.features="nested-attributes" openxes.version="1.0RC7"
  xmlns="http://www.xes-standard.org/">
  <extension name="Organizational" prefix="org" uri="http://www.xes-standard.org/org.xesext"/>
  <extension name="Time" prefix="time" uri="http://www.xes-standard.org/time.xesext"/>
  <extension name="Concept" prefix="concept" uri="http://www.xes-standard.org/concept.xesext"/>
  <global scope="trace">
    <string key="concept:name" value="UNKNOWN"/>
  </global>
  <global scope="event">
    <string key="concept:name" value="UNKNOWN"/>
  </global>
  <classifier name="Event Name" keys="concept:name"/>
  <string key="concept:name" value="sample event log"/>
  <trace>
    <string key="concept:name" value="1"/>
    <string key="age" value="22"/>
    <string key="disease" value="Flu"/>
    <event>
      <string key="concept:name" value="a"/>
      <date key="time:timestamp" value="2019-01-01T08:30:10.000+01:00"/>
      <string key="org:resource" value="E1"/>
    </event>
    <event>
      <string key="concept:name" value="b"/>
      <date key="time:timestamp" value="2019-01-01T08:45:00.000+01:00"/>
      <string key="org:resource" value="D1"/>
    </event>
    <event>
      <string key="concept:name" value="e"/>
      <date key="time:timestamp" value="2019-01-01T08:58:15.000+01:00"/>
      <string key="org:resource" value="E2"/>
    </event>
  </trace>
  <trace>
  </trace>
  <trace>
  </trace>
  <trace>
  </trace>
</log>

```

Fig.1: The XES format for the event log Table 1, showing only the first case (trace). In XES, the log contains traces and each trace contains events. The log, traces, and events have attributes, and *extensions* may define new attributes. The log declares the extensions used in it. The *global attributes* are the ones that are declared to be mandatory with a default value. The *classifiers* assign identity to each event, which makes it comparable to the other events.

mining. In [20], the aim is to provide a privacy-preserving method for discovering roles from event data, which can also be exploited for generalizing *resources* as individuals in event logs. In [13], the authors consider a cross-organizational process discovery context, where public process model fragments are shared as safe intermediates. In [9], the authors apply *k*-anonymity and *t*-closeness [12] on event data to preserve the privacy of *resources*. In [14], the authors employ the notion of differential privacy [8] to preserve the privacy of *cases* in event logs. In [21], an efficient algorithm is introduced applying *k*-anonymity and *confidence bounding* to preserve the privacy of *cases* in event logs.

Most related to our work are [19] and [18] which are focused on healthcare event data. In [19], the authors analyze data privacy and utility requirements for healthcare event data and the suitability of privacy preservation techniques is assessed. In [18], the authors extend their previous research ([19]) to demonstrate the effect of applying some anonymization operations on various process mining results, privacy metadata are advocated, and a privacy extension for the XES standard is proposed. However, formal models, possible risks, the tools support, and the event data which are not in the form of event logs are not discussed. In this paper, we provide a comprehensive infrastructure for recording privacy metadata which considers possible risks for data leakage, and the data utility. Our infrastructure is enriched by the formal models in theory and the tools support in practice.

4 Preliminaries

In this section, we provide formal definitions for event logs used in the remainder. Let A be a set. A^* is the set of all finite sequences over A , and $\mathcal{B}(A)$ is the set of all multisets over the set A . A finite sequence over A of length n is a mapping $\sigma \in \{1, \dots, n\} \rightarrow A$, represented as $\sigma = \langle a_1, a_2, \dots, a_n \rangle$ where $\sigma_i = a_i = \sigma(i)$ for any $1 \leq i \leq n$, and $|\sigma| = n$. $a \in \sigma \Leftrightarrow a = a_i$ for $1 \leq i \leq n$. For $\sigma \in A^*$, $\{a \in \sigma\}$ is the set of elements in σ , and $[a \in \sigma]$ is the multiset of elements in σ , e.g., $[a \in \langle x, y, z, x, y \rangle] = [x^2, y^2, z]$. $\sigma \downarrow_X$ is the projection of σ onto some subset $X \subseteq A$, e.g., $\langle a, b, c \rangle \downarrow_{\{a, c\}} = \langle a, c \rangle$. $\sigma \cdot \sigma'$ appends sequence σ' to σ resulting a sequence of length $|\sigma| + |\sigma'|$.

Definition 1 (Event, Attribute). Let \mathcal{E} be the event universe, i.e., the set of all possible event identifiers. Events can be characterized by various attributes, e.g., an event may have a timestamp and corresponds to an activity. Let \mathcal{N}_{event} be the set of all possible event attribute names. For any $e \in \mathcal{E}$ and name $n \in \mathcal{N}_{event}$, $\#_n(e)$ is the value of attribute n for event e . For an event e , if e does not have an attribute named n , then $\#_n(e) = \perp$ (null).

We assume three standard explicit attributes for events: *activity*, *time*, and *resource*. We denote \mathcal{U}_{act} , \mathcal{U}_{time} , and \mathcal{U}_{res} as the universe of activities, timestamps, and resources, respectively. $\#_{act}(e) \in \mathcal{U}_{act}$ is the *activity* associated to event e , $\#_{time}(e) \in \mathcal{U}_{time}$ is the *timestamp* of event e , and $\#_{res}(e) \in \mathcal{U}_{res}$ is the *resource* associated to event e .

Definition 2 (Case, Trace). Let \mathcal{C} be the case universe, i.e., the set of all possible case identifiers, and \mathcal{N}_{case} be the set of case attribute names. For any case $c \in \mathcal{C}$ and name $n \in \mathcal{N}_{case}$: $\#_n(c)$ is the value of attribute n for case c . For a case c , if c does not have an attribute named n , then $\#_n(c) = \perp$ (null). Each case has a mandatory attribute “trace” such that $\#_{trace}(c) \in \mathcal{E}^*$. A trace σ is a finite sequence of events such that each event appears at most once, i.e., for $1 \leq i < j \leq |\sigma|$: $\sigma(i) \neq \sigma(j)$.

Definition 3 (Event Log). Let \mathcal{U} be a universe of values including a designated null value (\perp), and $\mathcal{N} = \mathcal{N}_{event} \cup \mathcal{N}_{case}$ such that \mathcal{N}_{event} and \mathcal{N}_{case} are disjoint. An event log is a tuple $L = (C, E, N, \#)$, in which; $C \subseteq \mathcal{C}$ is a set of case identifiers, $E \subseteq \mathcal{E}$ is a set of event identifiers such that $E = \{e \in \mathcal{E} \mid \exists c \in C \ e \in \#_{trace}(c)\}$, $N \subseteq \mathcal{N}$ is a set of attribute names such that $N \cap \mathcal{N}_{event}$ are the event attributes, and $N \cap \mathcal{N}_{case}$ are the case attributes. For $n \in N$, $\#_n \in (C \cup E) \rightarrow \mathcal{U}$ is a function which retrieves the value of attribute n assigned to a case or an event (\perp is used if an attribute is undefined for the given case or event). In an event log, each event appears at most once in the entire event log, i.e., for any $c_1, c_2 \in C$ such that $c_1 \neq c_2$: $\{e \in \#_{trace}(c_1)\} \cap \{e \in \#_{trace}(c_2)\} = \emptyset$. If an event log contains timestamps, then the ordering in a trace should respect the timestamps, i.e., for any $c \in C$, i and j such that $1 \leq i < j \leq |\#_{trace}(c)|$: $\#_{time}(\#_{trace}(c)_i) \leq \#_{time}(\#_{trace}(c)_j)$. We denote \mathcal{U}_L as the universe of event logs.

5 Privacy-Preserving Data Publishing

Privacy-preserving data publishing is the process of changing the original data before publishing such that the published data remain practically useful while individual privacy is protected [10]. Note that the assumption is that the data analysts (process miners) are not trusted and may attempt to identify sensitive personal information. Various privacy models could be applied to the original data to provide the desired privacy requirements before data publishing. However, the transformations applied should be known in order to interpret the data. Therefore, we propose the privacy metadata in process mining based on the main anonymization operations. We consider *suppression* (*sup*), *addition* (*add*), *substitution* (*sub*), *condensation* (*con*), *swapping* (*swa*), *generalization* (*gen*), and *cryptography* (*cry*) as the main anonymization operation types in process mining. In this section, we define these operations and demonstrate how the original event log is modified by them.

5.1 Anonymization Operations

Anonymization operations are the main functions which modify the original event log to provide the desired privacy requirements and may have any number of parameters. Moreover, the operations can be applied at the case or event level, and the target of the operation could be a case, an event, or attributes of such an object. We define the *anonymizer* as a function which applies an anonymization operation to an event log.

Definition 4 (Anonymizer). *An anonymizer $anon \in \mathcal{U}_L \rightarrow \mathcal{U}_L$ is a function mapping an event log into another one by applying an anonymization operation.*

Definition 5 (Anonymizer Signature). *Let $OT = \{sup, add, sub, con, swa, gen, cry\}$ be the set of operation types. A signature $sign = (ot, level, target) \in OT \times \{case, event\} \times (\{case, event\} \cup N)$ characterizes an anonymizer by its type, the level (case or event), and the target (case, event, or case/event attribute). We denote \mathcal{U}_{sign} as the universe of signatures.*

Note that an anonymizer signature is not supposed to uniquely distinguish the anonymization operations applied to an event log, i.e., the same type of operation can be applied at the same level and to the same target multiple times, but it is designated to reflect the *type* and the *direct purpose* of the corresponding operation. This is considered as the minimum information required to make the analysts aware of the modifications w.r.t. the data utility and risks for data leakage. We say the *direct purpose* due to the interdependency of cases and events through traces, i.e., modifying cases may affect events and vice versa. This is demonstrated by some of the examples in the following. We assume that the only correlation between cases and events is the trace attribute of cases, and all the other attributes are independent. Certainly, the operations can be more accurately characterized by adding more information to the corresponding signatures, but this may lead to the data leakage in the sense of revealing

the protected information or details of the anonymization operation which is orthogonal to the ultimate goal of privacy preservation. In the following, we demonstrate the anonymization operations using some examples.

Suppression replaces some values, specified by the target of the operation and probably some conditions, with a special value, e.g., null value. The reverse operation of suppression is called *disclosure*. The group-based privacy models such as k -anonymity and its extensions [12] often utilize suppression. In the following, we provide three examples to demonstrate the effect of suppression on an event log. In [9], a group-based privacy model is proposed for discovering processes while preserving privacy.

Example 1 (Event-event suppression based on the activity attribute)

Let $L = (C, E, N, \#)$ be an event log. We want to suppress the events $e \in E$, if their activity attribute value is $a \in \mathcal{A}$. $\text{anon}_1^a(L) = L'$ such that: $L' = (C', E', N', \#')$, $E' = \{e \in E \mid \#_{act}(e) \neq a\}$, $C' = C$, and $N' = N$. $\forall e \in E' \forall n \in N' \#'_n(e) = \#_n(e)$, $\forall c \in C' \forall n \in N' \setminus \{trace\} \#'_n(c) = \#_n(c)$, and $\forall c \in C' \#'_{trace}(c) = \#_{trace}(c) \downarrow_{E'}$. The anonymizer signature of this operation is $sign = (sup, event, event)$.

Example 2 (Case-case suppression based on the trace length)

Let $L = (C, E, N, \#)$ be an event log. We want to suppress the cases $c \in C$, if the trace length of the case is $k \in \mathbb{N}_{\geq 1}$. $\text{anon}_2^k(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = \{c \in C \mid |\#_{trace}(c)| \neq k\}$, $E' = \{e \in \mathcal{E} \mid \exists c \in C' e \in \#_{trace}(c)\}$, and $N' = N$. $\forall e \in E' \forall n \in N' \#'_n(e) = \#_n(e)$ and $\forall c \in C' \forall n \in N' \#'_n(c) = \#_n(c)$. The corresponding signature for this operation is $sign = (sup, case, case)$.

Example 3 (Event-resource suppression based on the activity attribute)

Let $L = (C, E, N, \#)$ be an event log. We want to suppress the resource attribute value of the events $e \in E$, if their activity attribute value is $a \in \mathcal{A}$. $\text{anon}_3^a(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $E' = E$, and $N' = N$. $\forall e \in E' \forall n \in N' \setminus \{res\} \#'_n(e) = \#_n(e)$. For all $e \in E'$: $\#'_{res}(e) = \perp$ if $\#_{act}(e) = a$, otherwise $\#'_{res}(e) = \#_{res}(e)$. $\forall c \in C' \forall n \in N' \#'_n(c) = \#_n(c)$. The corresponding anonymizer signature is $sign = (sup, event, resource)$.

As above-mentioned, modifying cases may affect events and vice versa. In Example 1, event suppression modifies the cases through the trace attribute, which is an indirect effect not shown by the signature. Similarly, in Example 2, suppressing cases results in event suppression, i.e., all the events in the trace of the suppressed cases are removed. This is also an indirect effect which is not reflected in the corresponding signature.

Addition is often used by the *noise addition* techniques where the noise which is drawn from some distribution is randomly added to the original data to protect the sensitive values. In process mining, the noise could be added to cases, events, or the attribute values. In [14], the notion of *differential privacy* is used as a noise addition technique to perform privacy-aware process discovery.

Example 4 (Add an event to the end of traces based on the activity attribute of the last event in the trace)

Let $L = (C, E, N, \#)$ be an event log, and $N = \{trace, act, res, time\}$ such that “trace” is the case attribute and “act”, “res”, and “time” are the event attributes. We want to add an event $e' \in \mathcal{E} \setminus E$ with the activity attribute value $a_2 \in \mathcal{A}$ and the resource attribute value $r \in \mathcal{R}$ at the end of the trace of a case $c \in C$, if the activity of the last event in the trace is $a_1 \in \mathcal{A}$. $anon_4^{a_1, a_2, r}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $N' = N$, $C_{cond} = \{c \in C \mid \#_{act}(\#_{trace}(c) \mid \#_{trace}(c)) = a_1\}$, and $f \in C_{cond} \rightarrow \mathcal{E} \setminus E$ is a total injective function which randomly assigns unique event identifiers to the cases having the desired condition. We denote $E_{add} = range(f)$ as the set of added events. Hence, $E' = E \cup E_{add}$. $\forall e \in E \forall n \in N' \setminus \{trace\} \#'_n(e) = \#_n(e)$. For all $c \in C_{cond}$: $\#'_{act}(f(c)) = a_2$, $\#'_{time}(f(c)) = \#_{time}(\#_{trace}(c) \mid \#_{trace}(c)) + 1$, and $\#'_{res}(f(c)) = r$. $\forall c \in C' \setminus C_{cond} \#'_{trace}(c) = \#_{trace}(c)$, and $\forall c \in C_{cond} \#'_{trace}(c) = \#_{trace}(c) \cdot \langle f(c) \rangle$. The corresponding signature for this operation is $sign = (add, case, trace)$.

Substitution replaces some values with some substitutions specified by a set, i.e., a set of substitutions. The substitution could be done randomly or could follow some rules, e.g., a *round robin* manner. In [20], a substitution technique is used in order to mine roles from event logs while preserving privacy.

Example 5 (Event-activity substitution based on a set of sensitive activities and a set of activity substitutions)

Let $L = (C, E, N, \#)$ be an event log, $A_L = \{a \in \mathcal{A} \mid \exists e \in E \#_{act}(e) = a\}$ be the set of activities in L , $A_x \subseteq \mathcal{A} \setminus A_L$ be the set of activities used as the substitutions, $rand(X) \in X$ be a function which randomly returns an element from the set X , and $A_s \subset A_L$ be the set of sensitive activities. We want to randomly substitute the activity attribute value of the events $e \in E$, if the activity is sensitive. $anon_5^{A_s, A_x}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $E' = E$, and $N' = N$. For all $e \in E'$: $\#'_{act}(e) = rand(A_x)$ if $\#_{act}(e) \in A_s$, otherwise $\#'_{act}(e) = \#_{act}(e)$. $\forall e \in E' \forall n \in N' \setminus \{act\} \#'_n(e) = \#_n(e)$, $\forall c \in C' \forall n \in N' \#'_n(c) = \#_n(c)$. $sign = (sub, event, activity)$ is the signature corresponds to this operation.

Condensation first condenses the cases into similar clusters based on the sensitive attribute values, then in each cluster the sensitive attribute values are replaced with a collective statistical value, e.g., mean, mode, median, etc, of the cluster. In [5], the authors introduce condensation-based methods for privacy-preserving data mining. The following example shows how the condensation operates on the event logs assuming that there exists a sensitive case attribute.

Example 6 (Case-attribute condensation based on a set of case clusters, a cluster finder function, and using mode as the collective value)

Let $L = (C, E, N, \#)$ be an event log, $CL = \{cl_1, cl_2, \dots, cl_n\}$ be the set of case clusters, whose sensitive attribute value is similar, such that for $1 \leq i \leq n$: $cl_i \subseteq C$ and for $1 \leq i, j \leq n$, $i \neq j$: $cl_i \cap cl_j = \emptyset$. Also, let $f \in C \rightarrow CL$

be a function which retrieves the cluster of a case, and $n' \in N$ be the sensitive case attribute, e.g., disease. For a set X , $\text{mode}(X) \in X$ retrieves the mode of the set. We want to replace the value of n' for each case $c \in C$ with the mode of n' in the cluster of the case. $\text{anon}_6^{CL, f, n'}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $N' = N$, $E' = E$, $\forall_{c \in C'} \#_{n'}'(c) = \text{mode}(\{\#_{n'}(c') \mid c' \in f(c)\})$, $\forall_{c \in C'} \forall_{n \in N' \setminus \{n'\}} \#_n'(c) = \#_n(c)$, and $\forall_{e \in E'} \forall_{n \in N'} \#_n'(e) = \#_n(e)$. $\text{sign} = (\text{con}, \text{case}, n')$ is the signature corresponds to this operation.

Swapping aims to anonymize data by exchanging values of a sensitive attribute between individual cases. The individual cases which are chosen to exchange the sensitive attribute values are supposed to have similar sensitive attribute values. Therefore, cases need to be clustered into the clusters with the similar sensitive attribute values. The cases for swapping in the same cluster could be chosen either randomly or by some methods, e.g., the *rank swapping* method [17]. The following example shows how the swapping operates on the event logs assuming that there exists a sensitive case attribute.

Example 7 (Case-attribute swapping based on a set of case clusters and a cluster finder function)

Let $L = (C, E, N, \#)$ be an event log, $CL = \{cl_1, cl_2, \dots, cl_n\}$ be the set of case clusters, whose sensitive attribute value is similar, such that for $1 \leq i \leq n$: $cl_i \subseteq C$ and for $1 \leq i, j \leq n$, $i \neq j$: $cl_i \cap cl_j = \emptyset$. Also, let $f \in C \rightarrow CL$ be the function which retrieves the cluster of a case, and $n' \in N$ be the sensitive case attribute, e.g., disease. For a set X , $\text{rand}(X) \in X$ is a function which randomly retrieves an element from the set. We want to randomly swap the value of n' for each case $c \in C$ with the n' value of another case in the same cluster. $\text{anon}_7^{CL, f, n'}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $N' = N$, $E' = E$, $\forall_{c \in C'} \#_{n'}'(c) = \text{rand}(\{\#_{n'}(c') \mid c' \in f(c) \setminus \{c\}\})$, $\forall_{c \in C'} \forall_{n \in N' \setminus \{n'\}} \#_n'(c) = \#_n(c)$, and $\forall_{e \in E'} \forall_{n \in N'} \#_n'(e) = \#_n(e)$. The anonymizer signature of this operation is $\text{sign} = (\text{swa}, \text{case}, n')$.

Cryptography includes a wide range of techniques such as *encryption*, *hashing*, *encoding*, etc. In [22], the *connector method* is introduced as an encryption-based technique which breaks the relation between the events in the traces, while discovering the directly follows graph (DFG) [11] from an event log. In the following example, we demonstrate the effect of applying an encryption technique on the activity attribute of the events in an event log.

Example 8 (Event-activity encryption based on an encryption method)

Let $L = (C, E, N, \#)$ be an event log, ENC be the universe of encryption method names, and KEY be the universe of keys. We want to encrypt the activity attribute of all the events $e \in E$ using a method $m \in ENC$ and a key $k \in KEY$. Let \mathcal{U} be a universe of values, and \mathcal{U}_{enc} be a universe of encrypted values. $\text{enc} \in \mathcal{U} \times ENC \times KEY \rightarrow \mathcal{U}_{enc}$ is a partial function which encrypts a value $u \in \mathcal{U}$ given the name of method and a key. Given $k \in KEY$, and

$m \in ENC$, $anon_g^{k,m}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $E' = E$, and $N' = N$. $\forall_{e \in E'} \#_{act}'(e) = enc(\#_{act}(e), m, k)$, $\forall_{e \in E'} \forall_{n \in N' \setminus \{act\}} \#_n'(e) = \#_n(e)$, and $\forall_{c \in C'} \forall_{n \in N'} \#_n'(c) = \#_n(c)$. The signature of this operation is $sign = (cry, event, activity)$.

Generalization replaces some values, indicated by the target of operation and probably some conditions, with a parent value in the taxonomy tree of an attribute. The reverse operation of generalization is called *specialization*. The four main generalization schemes are *full-domain*, *subtree*, *sibling*, *cell* [10]. In the *full-domain* scheme, all values in an attribute are generalized to the same level of the taxonomy tree. In the *subtree* scheme, at a non-leaf node, either all child values or none are generalized. The *sibling* generalization is similar to the *subtree*, but some siblings may remain ungeneralized. In the *cell* generalization, some instances of a value may remain ungeneralized while in all the other schemes if a value is generalized, all its instances are generalized.

The generalization techniques are often used by group-based anonymization techniques. In the following, we demonstrate the effect of applying the generalization operation on the event logs by a simple example that uses the *full-domain* scheme to generalize the time attribute of the events.

Example 9 (Event-time generalization based on a time generalization method)

Let $L = (C, E, N, \#)$ be an event log and $TL = \{seconds, minutes, hours, days, months, years\}$ be the level of time generalization. $gtl \in \mathcal{T} \rightarrow \mathcal{T}$ is a function that generalizes timestamps to the given level of time. We want to generalize the time attribute of all the events $e \in E$ to the level $tl \in TL$. $anon_g^{tl}(L) = L'$ such that: $L' = (C', E', N', \#')$, $C' = C$, $E' = E$, and $N' = N$. $\forall_{e \in E'} \#_{time}'(e) = gtl(\#_{time}(e))$, $\forall_{e \in E'} \forall_{n \in N' \setminus \{time\}} \#_n'(e) = \#_n(e)$, $\forall_{c \in C'} \forall_{n \in N'} \#_n'(c) = \#_n(c)$. $sign = (gen, event, time)$ is the anonymizer signature of this operation.

Privacy-preserving data publishing in process mining, is done by applying a sequence of the anonymization operations to fulfill a desired privacy requirement. Note that the operations are often applied by the privacy models, where the data utility preservation is also taken into account.

Definition 6 (Privacy Preserving Data Publishing in Process Mining - ppdp). Let pr be the desired privacy requirement and \mathcal{U}_L be the universe of event logs, $ppdp^{pr} : \mathcal{U}_L \rightarrow \mathcal{U}_L$ is a privacy method that gets an event log and applies $i \in \mathbb{N}_{\geq 1}$ anonymization operations to the event log in order to provide an anonymized event log which satisfies the given privacy requirement pr . If we assume that pr is satisfied at the step (layer) n of the anonymization process, then for $1 \leq i \leq n$, $L_i = anon_i(L_{i-1})$ such that $L_0 = L$ and L_n is the anonymized event log which satisfies pr .

5.2 Data Utility and Data Leakage

We define *potential original event logs* to show how the anonymizer signature can be exploited to narrow down the set of possible original event logs.

Definition 7 (Potential Original Event Log). Let \mathcal{U}_L be the universe of event logs and \mathcal{U}_{sign} be the universe of signatures. $po \in \mathcal{U}_L \times \mathcal{U}_L \times \mathcal{U}_{sign} \rightarrow \mathbb{B}$ is a function that for a given event log, an anonymized event log, and the signature of the anonymized event log checks whether the given event log could be an original event log. $ol \in \mathcal{U}_L \times \mathcal{U}_{sign} \rightarrow 2^{\mathcal{U}_L}$ retrieves a set of potential original event logs, s.t., for $L' \in \mathcal{U}_L$ and $sign \in \mathcal{U}_{sign}$: $ol(L', sign) = \{L \in \mathcal{U}_L \mid po(L, L', sign)\}$.

To demonstrate the effect of the anonymizer signature, as a privacy metadata candidate, on *data utility* and *data leakage*, we analyze the set of potential original event logs. Here, the analysis is performed for Example 1. However, the concept is general and can be applied to all the operations. Let $L' = (C', E', N', \#')$ be an anonymized event log and $sign = (sup, event, resource)$. For an event log $L = (C, E, N, \#)$: $po(L, L', sign) = true \iff (C = C' \wedge E = E' \wedge N = N' \wedge \forall e \in E' \forall n \in N' \setminus \{res\} \#_n(e) = \#'_n(e) \wedge \forall \{e \in E' \mid \#_{res}(e) = \perp\} \#_{res}(e) \in \mathcal{U}_{res} \wedge \forall \{e \in E' \mid \#_{res}(e) \neq \perp\} \#_{res}(e) = \#'_{res}(e) \wedge \forall c \in C' \forall n \in N' \#_n(c) = \#'_n(c))$. $ol(L', sign) = \{L \in \mathcal{U}_L \mid po(L, L', sign)\}$ is the set of potential original event logs. Intuitively, $|ol(L', sign)| = |\{e \in E' \mid \#'_{res}(e) = \perp\}| \times |\mathcal{U}_{res}|$, where \mathcal{U}_{res} can be narrowed down to a few resources based on some background knowledge.

If we ignore the target information in the signature, i.e., $sign = (sup, event)$, the uncertainty regarding the original event log and the results of analyses would be much higher, since the suppressed data could be some events, or any event attribute n , s.t., $\exists e \in E \#'_n(e) = \perp$. That is, the uncertainty regarding the results of analyses is expanded from the resource perspective to all the perspectives. In contrast, if we add more information to the signature, reconstructing the original event log could be easier for an adversary. For example, if we add the condition of resource suppression to the signature, i.e., $sign = (sup, event, resource, (activity = a))$, then $|ol(L', sign)| = |\{e \in E' \mid (\#'_{res}(e) = \perp \wedge \#'_{act}(e) = a)\}| \times |\mathcal{U}_{res}|$. That is, the only information that an adversary needs to reconstruct the original event log, with high confidence, is to realize the set of resources who perform the activity a . These analyses demonstrate that privacy metadata should contain the minimum information that preserves a balance between data utility and data leakage.

5.3 Privacy Metadata

Privacy metadata in process mining should correspond to the privacy-preserving data publishing and are supposed to capture and reflect the modifications, made by the anonymization operations. For event data in the form of an event log, privacy metadata are established by the means of XES. Figure 2 shows the meta model of the proposed privacy metadata as an extension in the XES. The privacy metadata attributes are defined by an extension at the *log level*. Note that the level (log, trace, or event) that is chosen to include the privacy metadata is one of the important risk factors. Although the anonymization operations are applied at the case and event levels, adding the corresponding metadata to the same level is of high risk and may lead to the protected data leakage. Assume that we add the privacy metadata regarding the resource suppression operation

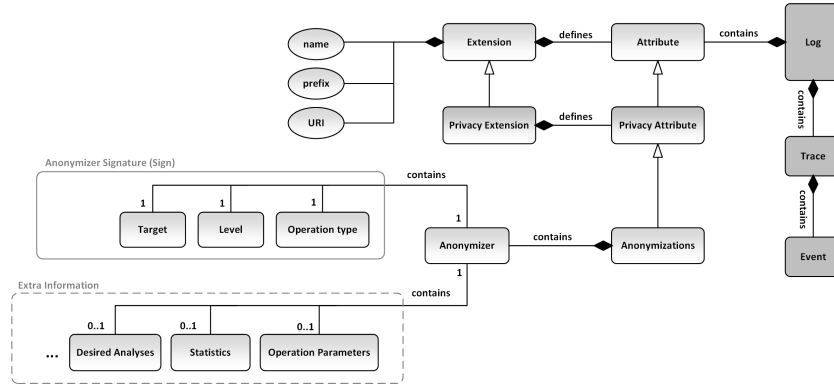


Fig. 2: The meta model of the proposed privacy metadata which is presented as an extension in XES. The privacy metadata attributes are added as the log attributes.

applied to Table 1 at the event level. By doing so, we expose the exact event that has been affected, and consequently, all the other information about the event are exposed, e.g., the activity performed by the removed resource is “r”, and background knowledge could connect this information to the protected sensitive data. A similar risk scenario can be explored for the activity substitution operation applied to Table 1. If the corresponding metadata is added at the event level, the substitution set of the activities could be partially or entirely exposed.

In Figure 2, *anonymizations* is the main privacy metadata element which contains at least one *anonymizer* element. Each *anonymizer* element corresponds to one anonymizer which is applied at a layer of the anonymization process in Definition 6. The *anonymizer* element contains two types of attributes: *mandatory* and *optional*. The mandatory attributes, residing in the solid box in Figure 2, correspond to the *anonymizer signature* (Definition 5) and should be reflected in the XES. However, the optional attributes, residing in the dash box in Figure 2, are the attributes which could be included in the XES. *Operation parameters* could contain the parameters which are passed to an anonymization operation, e.g., in Example 8, the name of encryption method is an operation parameter. *Statistics* could contain some statistics regarding the modification, e.g., the number of modified events, and *desired analyses* could contain the appropriate process mining activities which are applicable after modifying the event log. Note that the more unnecessary information included in the privacy metadata, the more risk is accepted. For example, if the *statistics* is included in an XES file where it is indicated that only 10 out of 1000 events have been modified, then an adversary makes sure that the transformed event log is almost the same as the original one (99% similarity).

Figure 3 shows the privacy metadata recorded after transforming Table 1 to Table 2. Note that *privacy* is considered as the *prefix* of the extension. At the first layer of the anonymization, a *substitution* operation has been applied which corresponds to Example 5, where $A_s = \{f\}$ and $A_x = \{g, k\}$. The metadata at

```

<list key="privacy:anonymizations">
  <container key="privacy:anonymizer">
    <string key="privacy:operation type" value="substitution" />
    <string key="privacy:level" value="event" />
    <string key="privacy:target" value="concept:name" />
  </container>
  <container key="privacy:anonymizer">
    <string key="privacy:operation type" value="generalization" />
    <string key="privacy:level" value="event" />
    <string key="privacy:target" value="time:timestamp" />
  </container>
  <container key="privacy:anonymizer">
    <string key="privacy:operation type" value="suppression" />
    <string key="privacy:level" value="event" />
    <string key="privacy:target" value="org:resource" />
  </container>
</list>

```

Fig. 3: Privacy metadata recorded after transforming Table 1 to Table 2 in order to satisfy a privacy requirement. *privacy* is considered as the *prefix* of the extension.

this layer notices the analysts that some activity substitutions have been done at the event level, but it reveals neither the set of sensitive activities nor the substitution set. At the second layer of the anonymization, a *generalization* operation has been done which generalizes the timestamps to the minutes level and corresponds to Example 9, where $tl = minutes$. At the last layer of the anonymization ($n = 3$ in Definition 6), a *suppression* operation has been done which suppresses some resources and corresponds to Example 3, where the activity attribute value is r . This lets the analysts know that some resource suppression have been done without revealing the corresponding event. Note that the concept of *layer* is implicitly established by the means of *list*, which imposes an order on the keys.

So far, we only focused on the privacy preservation techniques applying the main anonymization operations on the original event log and transform it to another event log. However, there could be the techniques that do not preserve the structure of a standard event log. For example, in [22], the original event log is transformed to another form of event data where *directly-follows relations* are extracted from the original traces for discovering *directly-follows graph*. Such intermediate results derived from event logs and intended to relate logs and models are called *abstractions* [3]. We introduce *Event Log Abstraction* (ELA) to deal with the intermediate results created by some privacy preservation techniques which are not in the form of standard event logs. ELA is an XML tag-based language composed of two main parts: *header* and *data*. The *header* part contains the privacy/confidentiality metadata, and the *data* part contains the data derived from the original event log. The privacy metadata in ELA includes: *origin*, *method*, and *desired analyses*. The *origin* tag shows name of the event log the abstraction derived from, the *method* tag contains name of the method, and the *desired analyses* contains list of the appropriate analyses w.r.t. the abstraction. The *data* part represents the data derived from the log in a tabular manner. Figure 4 shows the ELA derived from the event log “BPI Challenge 2012” [24] by applying the *connector* method [22].

5.4 Tool Support

Since most of the privacy preservation techniques in process mining have been developed in Python, in order to support the tools, we have developed a Python li-

```

<?xml version="1.0" encoding="UTF-8" ?>
<ELA>
  <header>
    <origin>BPI Challenge 2012</origin>
    <method>Connector Method</method>
    <desired_analyses>
      <field name="1">directly follows graph</field>
      <field name="2">process discovery</field>
    </desired_analyses>
  </header>
  <data>
    <item name="0">
      <field name="activity">W Completeren aanvraag</field>
      <field name="prev_activity">W Completeren aanvraag</field>
      <field name="relation_depth">1</field>
      <field name="trace_id">18</field>
      <field name="trace_length">40</field>
      <field name="connector">2be62f95e84d9bc75bc87e33aed2e64a</field>
    </item>
    <item name="1">
      <field name="activity">W Nabellen incomplete dossiers</field>
      <field name="prev_activity">W Nabellen incomplete dossiers</field>
      <field name="relation_depth">1</field>
      <field name="trace_id">80</field>
      <field name="trace_length">115</field>
      <field name="connector">f4489d7904f61899b9260e581be3b261</field>
    </item>
    <item name="2">
      <field name="activity">A_PREACCEPTED</field>
      <field name="prev_activity">A_PARTLYSUBMITTED</field>
      <field name="relation_depth">1</field>
      <field name="trace_id">75</field>
      <field name="trace_length">28</field>
      <field name="connector">764b7740c6ca99327ac62dd433dc1cb7</field>
    </item>
    <item name="3">
    <item name="4">
  </data>
</ELA>

```

Fig. 4: The event log abstraction derived from the event log “BPI Challenge 2012” after applying the *connector* method. Only the first 3 items are shown.

brary which is published as a Python package.^{2,3} The library is based on *PM4Py* [6] and includes two classes correspond to two types of event data generated by the privacy preservation techniques in process mining: *privacyExtension* and *ELA*. The general advantage of the privacy metadata library is that the privacy experts do not need to deal with the tag-based files in order to read (write) the privacy metadata. Another crucial requirement provided by the *privacyExtension* class is that it keeps the consistency of the privacy metadata by managing the order of the *anonymizers* in the *anonymizations* list. This class provides four main methods: *set_anonymizer*, *set_optional_anonymizer*, *get_anonymizations*, and *get_anonymizer*. The *set_anonymizer* method gets the mandatory attributes and appends them to the *anonymizations* list as an *anonymizer* if there already exists a privacy extension, otherwise it first creates a privacy extension and an *anonymizations* list, then adds the attributes to the list. The *set_optional_anonymizer* method is responsible to add the optional attributes and should be called after setting the mandatory attributes. This method gets the layer, which is an index in the *anonymizations* list, and optional attributes and adds the attributes to the given layer. The *get_anonymizations* method returns the whole *anonymizations* tag in the XES file as a Python *dictionary*. The *get_anonymizer* method gets a layer and returns the metadata of the layer.

²pip install p-privacy-metadata

³https://github.com/m4jidRafiei/privacy_metadata

6 Conclusions

Due to the fact that event logs could contain highly sensitive personal information, and regarding the rules imposed by the regulations, e.g., GDPR, privacy preservation in process mining is recently receiving more attention. Event logs are modified by privacy preservation techniques, and the modifications may result in the event data which are not appropriate for all the process mining algorithms. In this paper, we discussed types of event data generated by the privacy preservation techniques. We provided formal definitions for the main anonymization operations in process mining. Privacy metadata were proposed for event logs which are supported by formal definitions in order to demonstrate the completeness of the proposed infrastructure. The ELA (Event Log Abstraction) was introduced to cope with event data which are not in the form of standard event logs. We employed the IEEE XES standard in order to consistently develop our proposal for privacy metadata of event logs in practice, where a privacy extension was introduced. We also provided a Python library to support the privacy preservation tools for process mining which have been often developed in Python.

Acknowledgment

Funded under the Excellence Strategy of the Federal Government and the Länder. We also thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, Second Edition. Springer (2016). <https://doi.org/10.1007/978-3-662-49851-4>
2. van der Aalst, W.M.P.: Responsible data science: using event data in a “people friendly” manner. In: International Conference on Enterprise Information Systems. pp. 3–28. Springer (2016)
3. van der Aalst, W.M.P.: Process discovery from event data: Relating models and logs through abstractions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**(3), e1244 (2018)
4. van der Aalst, W.M.P., Adriansyah, A., De Medeiros, A.K.A., Arcieri, F., Baier, T., Blickle, T., Bose, J.C., Van Den Brand, P., Brandtjen, R., Buijs, J., et al.: Process mining manifesto. In: International Conference on Business Process Management. pp. 169–194. Springer (2011)
5. Aggarwal, C.C., Yu, P.S.: On static and dynamic methods for condensation-based privacy-preserving data mining. *ACM Trans. Database Syst.* **33**(1) (2008)
6. Berti, A., van Zelst, S.J., van der Aalst, W.M.P.: Process mining for python (pm4py): bridging the gap between process-and data science. *arXiv preprint arXiv:1905.06169* (2019)
7. Burattin, A., Conti, M., Turato, D.: Toward an anonymous process mining. In: Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference on. pp. 58–63. IEEE (2015)

8. Dwork, C.: Differential privacy: A survey of results. In: Theory and Applications of Models of Computation, 5th International Conference, TAMC 2008, Xi'an, China, April 25-29, 2008. Proceedings. pp. 1–19 (2008)
9. Fahrenkrog-Petersen, S.A., van der Aa, H., Weidlich, M.: PRETSA: event log sanitization for privacy-aware process discovery. In: International Conference on Process Mining, ICPM 2019, Aachen, Germany, June 24-26, 2019. pp. 1–8 (2019)
10. Fung, B.C., Wang, K., Fu, A.W.C., Philip, S.Y.: Introduction to privacy-preserving data publishing: Concepts and techniques. Chapman and Hall/CRC (2010)
11. Leemans, S.J., Fahland, D., van der Aalst, W.M.P.: Scalable process discovery and conformance checking. *Software & Systems Modeling* **17**(2), 599–631 (2018)
12. Li, N., Li, T., Venkatasubramanian, S.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20 (2007)
13. Liu, C., Duan, H., Qingtian, Z., Zhou, M., Lu, F., Cheng, J.: Towards comprehensive support for privacy preservation cross-organization business process mining. *IEEE Transactions on Services Computing* (1), 1–1 (2016)
14. Mannhardt, F., Koschmider, A., Baracaldo, N., Weidlich, M., Michael, J.: Privacy-preserving process mining - differential privacy for event logs. *Business & Information Systems Engineering* **61**(5), 595–614 (2019)
15. Mannhardt, F., Petersen, S.A., Oliveira, M.F.: Privacy challenges for process mining in human-centered industrial environments. In: 2018 14th International Conference on Intelligent Environments (IE). pp. 64–71. IEEE (2018)
16. Michael, J., Koschmider, A., Mannhardt, F., Baracaldo, N., Rumpe, B.: User-centered and privacy-driven process mining system design for IoT. In: Information Systems Engineering in Responsible Information Systems. pp. 194–206 (2019)
17. Nin, J., Herranz, J., Torra, V.: Rethinking rank swapping to decrease disclosure risk. *Data Knowl. Eng.* **64**(1), 346–364 (2008)
18. Pika, A., Wynn, M.T., Budiono, S., ter Hofstede, A.H., van der Aalst, W.M.P., Reijers, H.A.: Privacy-preserving process mining in healthcare. *International Journal of Environmental Research and Public Health* **17**(5), 1612 (2020)
19. Pika, A., Wynn, M.T., Budiono, S., ter Hofstede, A.H.M., van der Aalst, W.M.P., Reijers, H.A.: Towards privacy-preserving process mining in healthcare. In: Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers. pp. 483–495 (2019)
20. Rafiei, M., van der Aalst, W.M.P.: Mining roles from event logs while preserving privacy. In: Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria. pp. 676–689 (2019)
21. Rafiei, M., Wagner, M., van der Aalst, W.M.P.: TLKC-privacy model for process mining. In: 14th International Conference on Research Challenges in Information Science, RCIS 2020 (2020)
22. Rafiei, M., von Waldthausen, L., van der Aalst, W.M.P.: Ensuring confidentiality in process mining. In: Proceedings of the 8th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2018), Seville, Spain (2018)
23. Rafiei, M., von Waldthausen, L., van der Aalst, W.M.P.: Supporting confidentiality in process mining using abstraction and encryption. In: Data-Driven Process Discovery and Analysis - 8th IFIP WG 2.6 International Symposium, SIMPDA 2018, and 9th International Symposium, SIMPDA 2019, Revised Selected Papers (2019)
24. Van Dongen, B.F.: BPIC 2012. Eindhoven University of Technology (2012)
25. Voss, W.G.: European union data privacy law reform: General data protection regulation, privacy shield, and the right to delisting. *Business Lawyer* **72**(1) (2016)