Bus Inf Syst Eng
https://doi.org/10.1007/s12599-019-00613-3

RESEARCH PAPER

Privacy-Preserving Process Mining

Differential Privacy for Event Logs

Felix Mannhardt · Agnes Koschmider · Nathalie Baracaldo · Matthias Weidlich · Judith Michael

Received: 10 October 2018 / Accepted: 6 July 2019

Abstract Privacy regulations for data can be regarded as a major driver for data sovereignty measures. A specific example for this is the case of event data that is recorded by information systems during the processing of entities in domains such as e-commerce or health care. Since such data, typically available in the form of event log files, contains personalized information on the specific processed entities, it can expose sensitive information that may be traced back to individuals. In recent years, a plethora of methods have been developed to analyse event logs under the umbrella of process mining. However, the impact of privacy regulations on the technical design as well as the organizational application of process mining has been largely neglected. This paper set out to develop a protection model for event data privacy which applies the well-established notion of differential privacy. Starting from common assumptions about the event logs used in process mining, this paper presents potential privacy leakages and means to protect against them. The paper also shows at which stages of privacy leakages a protection model for event logs should be used. Relying on this understanding, the notion of differential privacy for process discovery methods is instantiated, i.e., algorithms that aim at the construction of a process model from an event log. The general feasibility of our approach is demonstrated by its application to two publicly available real-life events logs.

Keywords Differential privacy · Process mining · Event logs · Data protection · Data sovereignty

Accepted after two revisions by the editors of the special edition.

Dr. F. Mannhardt
Department of Technology Management, SINTEF Digital, P.O. Box 4760, Torgarden, 7465 Trondheim, Norway
e-mail: felix.mannhardt@sintef.no

Prof. Dr. A. Koschmider (✉)
Department of Computer Science, Kiel University, Hermann-Rodewald-Str. 3, 24118 Kiel, Germany
e-mail: ak@informatik.uni-kiel.de

N. Baracaldo
IBM Almaden Research Center, San Jose, CA, USA
e-mail: baracald@us.ibm.com

Prof. Dr. M. Weidlich
Humboldt-Universität zu Berlin, Rudower Chaussee 25, 12489 Berlin, Germany
e-mail: matthias.weidlich@hu-berlin.de

Dr. J. Michael
Software Engineering, RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany
e-mail: michael@se-rwth.de

1 Introduction

Event log files are used as input to every process mining algorithm and may originate from all kinds of systems, like enterprise information systems or hospital information systems. Often, the aim of these algorithms is to derive an as-is model of the process that created these logs which can be used to further analyze the actual process execution. To derive a process model from an event log file, the event log must at least store the order of events, often established by means of a timestamp, each event must belong to a case of the process, and events must refer to activities of the process under consideration (van der Aalst 2016). This minimal set of event log attributes already makes it possible to infer information related to individual working of entities through the analysis of the duration of activities. Thus, process mining allows a considerable insight into data, which carries the inherent risk that what is disclosed may be private. *Privacy* concerns informal self-determination, which means the ability to decide who is permitted access to information about a person (Bergeron 2000). It is considered to be a fundamental human right and, thus, is included in the legislation of different countries. Due to Europe's General Data Protection Regulation (EU GDPR), organizations are obliged to consider privacy throughout the complete development process (i.e., privacy by design) (D'Acquisto et al. 2015a), which also applies for the design of process mining systems.

Currently, process mining and privacy are considered orthogonal. Process mining algorithms aim to discover accurate process models from event logs at the expense of disclosure of information that should be protected. For instance, employee data is used in process mining for predictions of employee performance. Such a trade-off between accuracy and privacy has already been illustrated and analyzed for data-mining-based approaches (Aldeen et al. 2015). For process mining, however, such trade-offs are largely unexplored. Notably though, privacy considerations for process mining have recently been outlined by Mannhardt et al. (2018), who point to two general challenges: *technological privacy challenges* and *organizational privacy challenges*. Technology privacy challenges are related to the design of privacy-by-design or privacy-by-default approaches, while organizational privacy challenges address the understanding and audition of data use by enterprises. While Mannhardt et al. discuss various relevant privacy challenges for process mining, they do not provide any approach or solution for them.

The aim of this paper is, therefore, to fill this gap and to provide a privacy-preserving technique for process mining which considers technological challenges. More precisely, we aim to define a protection model for event log privacy with minimum loss of utility for process mining, i.e., process discovery remains useful while the disclosure of sensitive data is reduced For this, however, the following questions must be understood:

- *RQ 1* At which stage of data paths is a protection model for event log privacy required?
- *RQ 2* How can event log privacy be ensured with a minimum loss of utility for process mining?

Against this background, the remainder of this paper proceeds as follows. The next section defines the terms used as input to define a privacy-preserving technique for process mining and introduces our use case from a hospital that will be used for illustration throughout the paper. Generally, application areas of our approach are those with a demand for high privacy preservation. Section 3 investigates privacy issues of process mining for our use case with the purpose to answer *RQ 1*. Section 4 uses this use case to construct the protection model based on differential privacy which is instantiated for event logs in Sect. 5. Section 6 presents evaluation results, which are related to *RQ 2*. Related work is discussed in Sect. 7. The paper ends with a summary and an outlook on future work.

2 Foundation

Below, we discuss terms related to the context of privacy and process mining and apply them to the use case of healthcare processes in hospitals. Such processes describe activities of medical treatments as well as their organizational support. This includes the tasks that were performed, their date and the involved resources (medical staff, administrative staff and patients). Hospital information systems have a high demand for privacy and security considerations, since electronic health records need privacy protection. While we use a hospital use case to illustrate our approach, there are many similar situations in which organizations have centralized control over an event log and want to protect the privacy of individuals for whom cases are processed.

2.1 Privacy-Related Terms

As mentioned in the introduction, *privacy* concerns informal self-determination, which means the ability to decide who is permitted access to information about a person (Bergeron 2000). According to Hoepman (2014) eight privacy design strategies exist which are compliant with GDPR and can be considered as requirements for the design of privacy-preserving process mining systems:

- *minimize* The amount of personal information that is processed should be minimal.
- *hide* Any personal information that is processed should be hidden from plain view.
- *separate* The processing of personal information should be done in a distributed way whenever possible.
- *abstract* Personal information should be processed with the least possible detail in which it is (still) useful through summarizing or grouping data.
- *inform* Data subjects should be adequately informed whenever personal information is processed.
- *control* Data subjects should retain control over the processing of their personal information.
- *enforce* A privacy policy compatible with legal requirements should exist and should be enforced.
- *demonstrate* Be able to demonstrate compliance with the privacy policy and any applicable legal requirements.

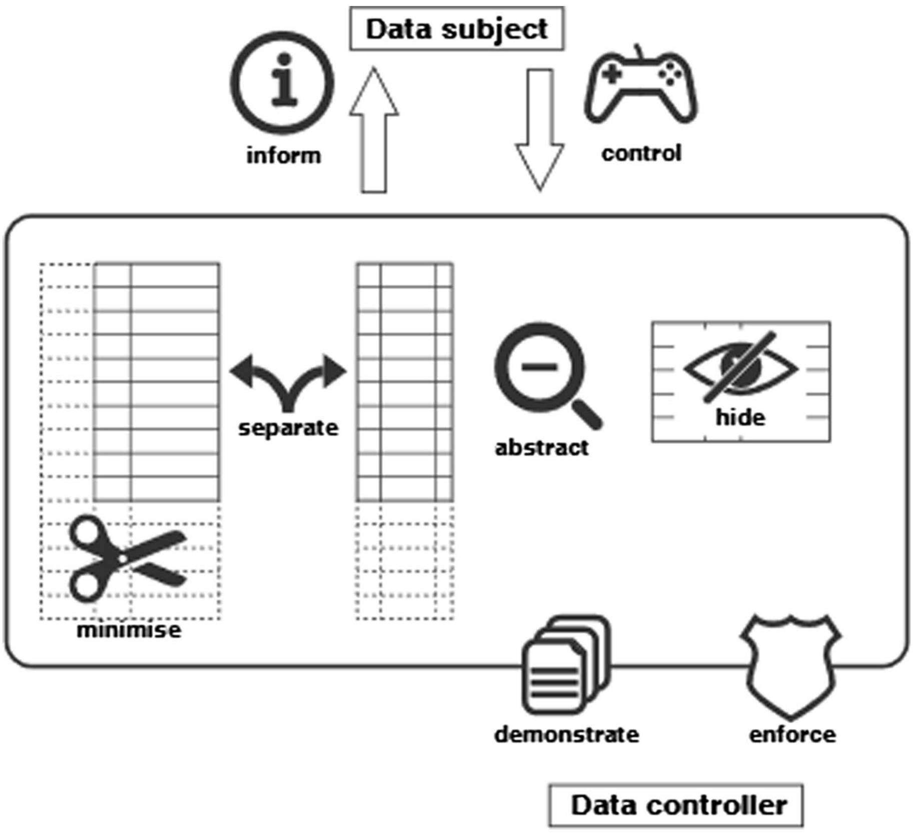


Fig. 1 An illustration of eight privacy design strategies for a database (Hoepman 2018). The strategies can be transferred to event logs and, thus, serve as requirements for the design of privacy-preserving process mining systems

Figure 1 shows the application of these privacy design strategies for a database as adopted from Hoepman (2018). These privacy design strategies can be applied analogously to event logs ensuring privacy while conducting process mining.

Now, we discuss how privacy is related to security and data sovereignty. *Security* provides the foundations required to ensure data privacy and is defined as "preservation of confidentiality, integrity and availability of information; in addition, other properties such as authenticity, accountability, non-repudiation and reliability can also be involved." ISO/IEC 27000 (2018). Additional principles which are introduced into regulations are (a) a priori consent, and explicit opt-in, (b) data sovereignty and (c) extra personal protection (Yu 2014), whereas others also discuss the notion of (d) trust (Sicari et al. 2015). While (a) and (c) are clearly defined, there exists no clear definition for the terms data *sovereignty* and trust (Sicari et al. 2015). A fairly common understanding of the term *trust* seems to be that people do not share all data with everyone, but do share certain parts of data with a person they trust because of several factors, e.g., past interactions, the type of relationships, similar personality attributes such as interests, the sensitive nature of the data we are sharing at that moment in time (Sacco et al. 2013). In the cloud domain the term data *sovereignty* is related to the geo-location of data (placing it within the borders of a particular nation) included in service level agreement (SLA) contracts (Peterson et al. 2011). Data protectionists discuss the term in connection with the personal rights of the people from whom these data originate (Mettler 2016). The main concern with data sovereignty is to maintain privacy regulations such as GDPR. This means that systems which do not comply with privacy regulations can hardly maintain data sovereignty. In this way, this paper provides an essential step towards data sovereignty.

2.2 Process Mining Concepts

Once processes conducted by an organization, e.g., the handling and treatment of patients at a hospital, are supported by modern information systems, the conduct of these processes is commonly reflected in event data. Here, an *event* denotes a recorded change of some operational state, or the execution of an activity that has led to the respective state change. In a hospital context, for instance, an event may indicate that a particular treatment step has been completed for a specific patient. An *event log* is a set of such events. However, most process mining methods do not work directly on such a set of events, but require the definition of a *case* notion. That is, the events of a log are partitioned based on which events are jointly considered as

Table 1 Excerpt of event data of an emergency department adopted from Mans et al. (2013)

Patient identifier	Day of birth	Sex	Address	Executing doctor	Requesting depart.	Execution depart.	Descr. depart.	Operation	Descr. operation
999999	7-5-1970	Man	Berlin	Van	PINT	RHMA	Lab Surgery	676700	Corpuscular radiation
999999	7-5-1970	Man	Berlin	Van	PINT	RHMA	Lab Surgery	370407D	Radia. foto
999999	7-5-1970	Man	Berlin	Van	PINT	RHMA	Lab Surgery	370712B	Radia. analysis
999999	7-5-1970	Man	Berlin	Van	PINT	RHMA	Lab Surgery	370715A	Tromb. count
999999	7-5-1970	Man	Berlin	LKC	PINT	LCHE	Lab Gastro-Enterology	370423	Hemogl. foto
999999	7-5-1970	Man	Berlin	LKC	PINT	LCHE	Lab Gastro-Enterology	370442	Leuko count

Patient identifier	Start operation	Trajectory identifier	Trajectory code	Start trajectory	Diagnosis descr.	# operations	Depart. identifier
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LHMA
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LHMA
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LHMAB
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LHMA
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LCHE
999999	13-10-2017	0000001	1345632	13-02-2018	Acute pancreas	1	LCHE

a single instance of a process. The definition of a case, therefore, depends on the analysis questions to answer by means of process mining. For instance, in a hospital, all treatment events may be grouped per patient or per medical staff member. The former then highlights how treatment is conducted from the perspective of each individual patient, whereas the latter highlights the flow of work as conducted by staff members.

To formalize the above notion of event logs, we adopt a relational model of events, which is a common model in data stream processing (Arasu et al. 2016). Events have a schema, which is modeled as a tuple of attributes $\mathcal{A} = (A_1, \ldots, A_n)$. Each attribute A_i is of a primitive type with a finite domain, the latter being denoted by $dom(A_i)$. In our setting, we assume each event schema to comprise at least two distinguished attributes: Attribute *id* captures a unique identifier per event, while *timestamp* denotes the occurrence time of the respective event. Both attributes can be assumed to have the domain $\mathbb{N}$. Given an event schema, an event is an instance of the schema, denoted by $e = (a_1, \ldots a_n)$, with a_i being the value of the respective attribute A_i.

An event log E is a set of events, as defined above. A case is induced by an attribute, or a combination thereof. That is, all events carrying the same value for the attribute(s) form a single case. For the example of a treatment process in a hospital, Table 1 illustrates an event schema and a log comprising respective events that was adopted from Mans et al. (2013). This event log describes the diagnosis, trajectory and the operation of a patient with acute pancreas. Several doctors and departments are involved in the diagnosis, trajectory and operation. Each line describes a service that has been delivered to a patient. The second line shows that the hemoglobin was determined (column "description operation") by the doctor Van (column "executing doctor") from the hematological lab (column "description department") on October 13th 2017 (column "start operation"). Note, that for this event data only the day is known on which the service has been delivered. In a hospital context, for instance, an event may indicate that a particular treatment step has been completed for a specific patient.

Here, in addition to *id* and *timestamp*, the schema comprises attributes such as *day of birth* and *# operations*, being of domains *date* and *integer*, respectively. Moreover, different notions of a case may be considered for this example. For the analysis, one may assume the perspective of a patient (the cases are induced by attribute *patient identifier*) or the work cycle of the doctor (the cases are induced by attribute *execution doctor*).

Regardless of how a case is defined, we note that the events of a case are ordered by their timestamps. In many application scenarios, this order is even total – in our example, a patient may only get a single treatment at a specific time point, or a doctor may finish a treatment step only for a single patient. We denote the sequence of events recorded for a single case as trace and the set of all traces induced by specific attributes $C \subseteq A$ over an event log E as

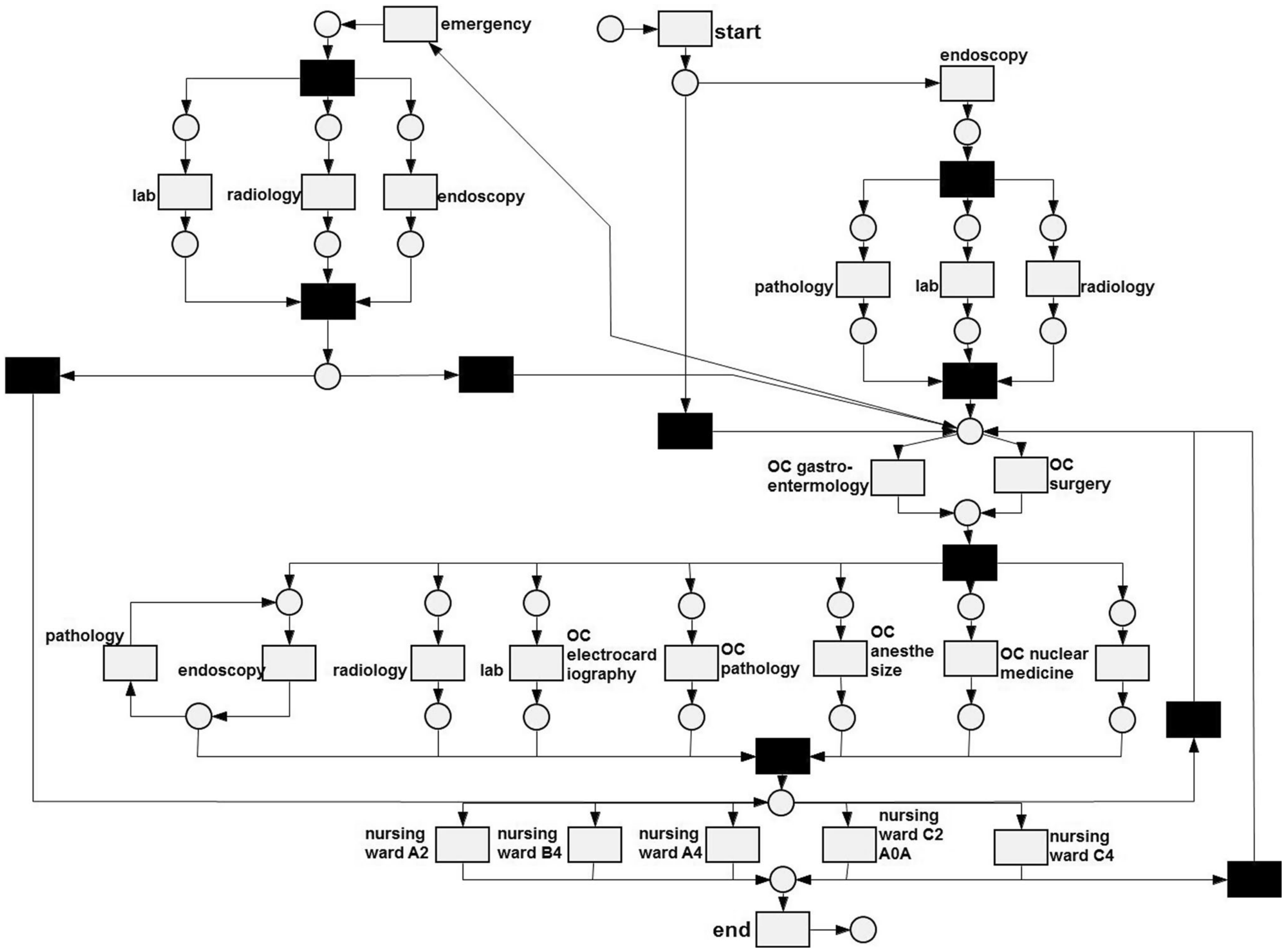


Fig. 2 The patient process before operation relying on the event schema in Table 1 and adopted from Mans et al. (2013). The black Petri net transitions are invisible transitions. They have no labels, are not recorded events and are used for routing purposes

$L_{E,C} \subseteq E^*$. The order within a trace is captured by a relation $\succ \subseteq E \times E$, such that for two events $e, e' \in \sigma$ of a single case $\sigma \subseteq L_{E,C}$, it holds that $e \succ e'$, if and only if, $e.timestamp < e'.timestamp$.

The ordering of events within a trace is important for many process mining methods. Assuming that a notion of a trace has been defined and that an attribute (or an attribute combination) has been selected to signal the activities of interest, i.e., the atomic units of a work, a process model may be discovered from the ordering of the events that represents these activities. Common process discovery algorithms, see Augusto et al. (2017) for a recent survey and comparative evaluation, generalize the observed ordering of events to extract the causal dependencies between the activities in a process. Adopting the attribute *patient identifier* as a notion of a case, Fig. 2 shows an example: a process model in Petri net notation derived for the event data of the schema illustrated in Table 1. Transitions, depicted as rectangles, represent activities and places, depicted as circles, are used to model the possible states of the process. The unnamed black transitions, also denoted as invisible transitions, are mined for routing purposes and do not represent actual activities. Together with formal execution semantics [(see e.g. in van der Aalst (2016)] the model describes all the possible process behavior.

By discovering several process models and slightly varying the filtering condition it is possible to identify patients and staff. An example would be an attempt to check for the existence of patients with rare diseases, which are likely to follow a unique sequence of activities. Together with background knowledge, it may be possible to identify the patient, for which the events were recorded and the staff who was involved in the treatment. To a certain degree process mining methods already abstract from (sensitive) details by deriving a process model that reveals only the observed sequences of activity execution. However, often occurrence frequencies, performance

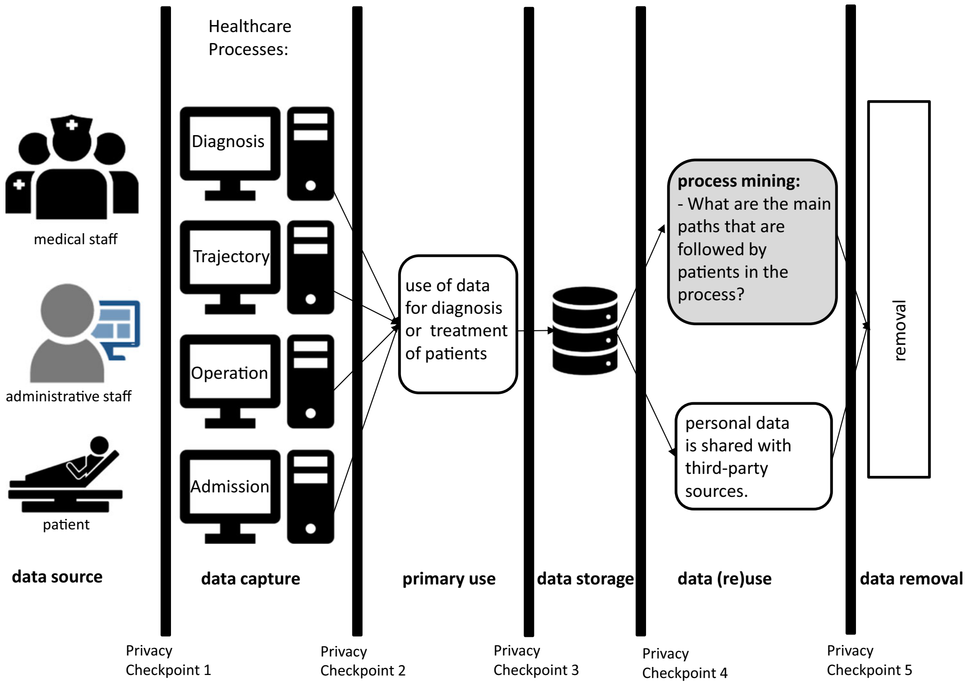


Fig. 3 Identification of data passes and privacy checkpoints for hospital health processes adapted from the privacy checkpoint model proposed by Mannhardt et al. (2018). Privacy checkpoint 4 is considered as a privacy leakage for process mining

information, and decision rules are discovered in addition to the basic control-flow of the process (Rozinat and van der Aalst 2006; van der Aalst et al. 2012), which may leak additional information from the event log. Process mining is often an iterative process in which multiple process models for different subsets of the event log, filtered according to conditions of interest, are discovered and compared (van Eck et al. 2015). Eventually, event data and particularly healthcare processes have a high demand for privacy preserving process mining. The next section studies potential privacy leakages and means to protect someone from them in the context of hospital health processes.

3 Privacy Issues for Process Mining of Healthcare Processes

Regarding the domain of healthcare processes in hospitals, we will show how the aforementioned privacy design strategies (see Sect. 2.1) become relevant in order to avoid the disclosure of personally identifiable records in event logs. To this end, we discuss privacy checkpoints for healthcare processes.

With respect to *RQ 1* (At which stage of data paths is a protection model for event log privacy required?) we apply the privacy checkpoint diagram from Mannhardt et al. (2018) to the event schema of healthcare processes shown in Table 1. According to this privacy checkpoint diagram, data passes six stages within healthcare processes, which are visualized in Fig. 3. These stages are in line with common data life-cycle models (Yu and Wen 2010), especially with those that are aligned with privacy considerations (see for a reference D'Acquisto et al. 2015b, p.26).

Specifically, the checkpoint diagram builds upon the following phases:

- *data source* Given our use case, the sources of data originate from medical staff, administrative staff and patients. We refer to this data as *personal data*.
- *data capture* Data from these data sources is captured when devices and systems log tasks of medical staff, administrative staff and patients, or when recognizing the identity or requesting actions. Since this stage tracks who does what, when and where with data, anonymization techniques should be used here protecting disclosure of sensitive events.
- *primary use* The hospital determines the purposes for which and the means by which the captured data is processed. For instance, the captured data can be used to support the work of medical and administrative staff for the diagnosis or treatment of patients.
- *data storage* personal data and events of medical staff, administrative staff and patients are stored in a database or event logs. The data might be processed by data mining approaches aiming to address performance indicators such as the number of pancreas operations, the length of waiting lists or the success rate of surgeons.
- *data (re)use* At this stage, data from event logs is used for process mining aiming to determine the main paths

that are followed by patients or medical staff in the process. Such an analysis demands privacy techniques to protect personally identifiable records in event logs. Personal data might also be retrieved from third-party sources such as public databases or other hospitals, which obviously triggers a GDPR requirement (i.e., demonstration that the data was retrieved in compliance with GDPR regulations). Compliance is a central concern in the context of hospital processes (Mans et al. 2013). At this stage, data from several sources is required, which increases the number of leakages.
- *data removal* Raw data is permanently deleted.

With regard to *RQ 1*, we consider the privacy checkpoint 4 and the stage *data (re)use* as points of privacy leakage for process mining. Although event log protection becomes relevant at the data (re)use stage, several privacy concerns must be addressed before. Data should not be captured in unauthorized ways (see stage *data capture*). Particularly, requirements for event data must be met in a way that information on cases, timestamps, and activities have been authorized to be captured. Also, data should not be processed for unapproved purposes (see stage *primary use*). To ensure GDPR-compliant process mining and, thus, to take into account all privacy checkpoints, requires organizational and technological privacy and data security measures, which we consider as future work. According to the value chain of data paths suggested in D'Acquisto et al. (2015b), the stage of data (re)use addresses the *abstract* privacy design pattern (see Sect. 2.1). Thus, the protection model as presented in the next section focuses only on the *abstract* privacy challenge, particularly for data (re)use, as explained before allowing "to release aggregate information about the data, without leaking individual information about participants". Please note that while we use a hospital use case to illustrate our protection model for event logs, there are a many similar situations in which organizations have centralised control over an event log and want to protect the privacy of individuals for which cases are processed [(e.g., public administration process as the one in de Leoni and Mannhardt (2015)]. The next section presents the privacy protection model for events logs providing differential privacy.

4 Protection Model for Event Logs Based on Differential Privacy

Several privacy frameworks have been proposed in the literature. Such frameworks have been suggested to a large extent for data mining (Aldeen et al. 2015; Mendes and Vilela 2017) and aim to find the best suitable privacy preserving technique for the data. Several notions to measure the level of privacy guaranteed by algorithms have been proposed, such as k-anonymity, l-diversity, and differential privacy. In this work, we focus on differential privacy, as it is known to provide a strong privacy model. We first summarize the underlying ideas before incorporating it into a protection model for event logs.

4.1 Introduction to Differential Privacy

The strongest privacy model available to date which provides provable privacy guarantees is *differential privacy* (Dwork 2008). Therefore, the protection model presented in this paper relies on differential privacy and it supports the *abstract* design privacy patterns (see Sect. 2.1). Differential privacy establishes a theoretical limit on the influence of a single row on a dataset (e.g., individual's data), thus limiting an attacker's ability to infer such a membership. Typically, noise is added proportionally to the *sensitivity* of the output. Sensitivity measures the maximum change of the output due to the inclusion of a single data instance.

Definition 1 [*Differential Privacy* (Dwork 2008)] A randomized mechanism $\mathcal{K}$ provides (ϵ, δ)-differential privacy if for any two neighboring database D_1 and D_2 that differ in only a single entry, $\forall S \subseteq Range(\mathcal{K})$,[1]

$$\Pr(\mathcal{K}(D_1) \in S) \leq e^{\epsilon} \Pr(\mathcal{K})(D_2) \in S) + \delta \qquad (1)$$

If $\delta = 0$, $\mathcal{K}$ is said to be ϵ-differential privacy. In Definition 1, a larger ϵ results in less privacy, while a smaller ϵ results in more privacy. However, as the noise which is typically added to fulfill Definition 1 increases, the accuracy or utility of the results diminish. Two popular mechanisms for achieving differential privacy are the Laplacian and Gaussian mechanisms (Dwork et al. 2014).

The Laplacian mechanism is used to provide differential privacy for counting the number of records in a database. Before releasing the number of records, Laplacian noise is added to the original count $\|D\|$ of records in a database D:

$$\mathcal{K}(D) = |D| + Laplace\left(0, \frac{1}{\epsilon}\right)$$

The Laplace distribution is chosen, since, due to the symmetric exponential nature of the distribution, therefore the result is likely be close to the correct one while ensuring the differential privacy property (McSherry 2010). Figure 4 illustrates this property of the Laplace distribution for example database counts. Note that restricting the type of queries to counting the number of records might seem

[1] Here, $Range(\mathcal{K})$ denotes the set of possible outputs of $\mathcal{K}$ and Pr denotes probability.

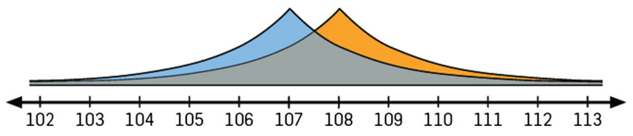


Fig. 4 Adding random noise to the result of a counting query using the Laplace distribution as illustrated in McSherry (2010). This transformation ensures differential privacy while keeping the noisy result close to the original value

limiting, but we will show later that this is sufficient for many process mining applications. Furthermore, other mechanisms exist to extend this to other aggregation queries like *averages* and *median* as well as to partition queries (McSherry 2010).

Multiple kinds of differential privacy have been proposed in the literature. In particular, a distinction can be made based on where a differential privacy mechanism can be run. In the first case, an entity can be trusted to cope with a differential private mechanism. In the second case, data owners hide their information, and hence add noise locally before sharing their data (Blum et al. 2005). In the latter case, the amount of noise injected into query results is higher to keep the privacy guarantees, which makes it more difficult to obtain high accuracy.

Additionally, event-driven differential privacy approaches have been proposed for cases where continuous observations are produced (Dwork et al. 2010). In these scenarios, data needs to be anonymized differently given that there is no concept of creating a table. As will be explained later, we model the problem in a way that a table can be constructed and standard differential privacy methods for static databases can be applied.

4.2 Privacy Protection Model for Event Logs

Given the fact that in the use case at hand hospitals already have access to the patient's and hospital records, we assume a centralized privacy approach to realize the *abstract* privacy design strategy to protect the data (re)use of event data for process mining using differential privacy. Please note that this centralized approach to handling privacy would also be possible in many other scenarios with a centralized data management, e.g. in public administration.

Figure 5 shows a schematic illustration of the envisioned protection model. The environment is divided into a trusted environment, in which data is processed to provide the primary services of the hospital (primary use) in accordance with the consent of patients and staff (data sources). Additionally, the captured sensitive data is stored as an event log in a protected data storage for later analysis with process mining methods. Up to the data storage stage

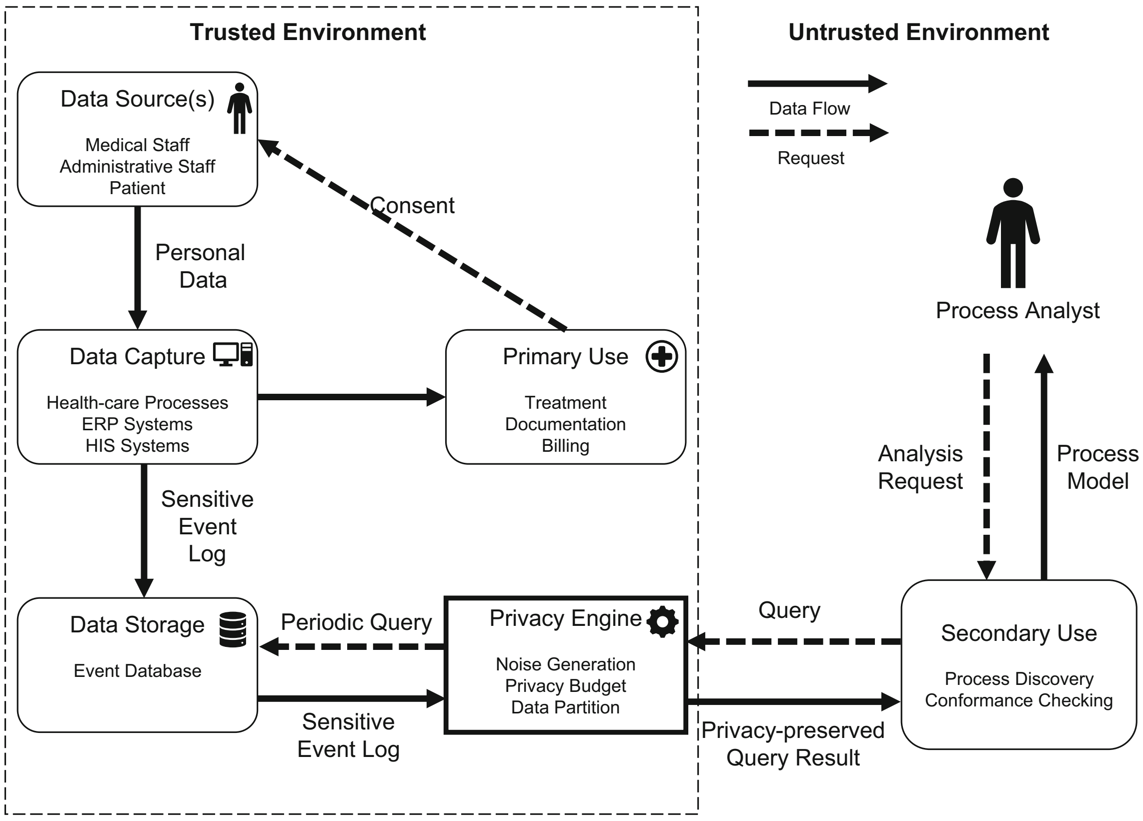


Fig. 5 Schematic illustration of the privacy protection model for event logs in reference to the stages introduced by Mannhardt et al. (2018)

we rely on organizational and technological measures (e.g., access control, encryption) to fully protect the privacy of stakeholders.

However, the (re)use of data may not be covered by the initial consent for using the data. Indeed, process mining is commonly applied to historical data in an exploratory fashion without a clear analysis question in mind. For such usage it is difficult to obtain consent and, thus, it is difficult to access the data for process mining directly inside the trusted environment. Many patients could choose to opt out of such secondary use of their data if we cannot ensure their privacy to be respected in all cases.

The main idea of the envisioned protection model is to guarantee differential privacy (cf. Definition 1) for the data providers. We introduce a *privacy engine*, which acts as the single point of access for process mining algorithms. All data required by the algorithms needs to be queried according to a set of restricted query operations. This privacy engine resides in the trusted environment and introduces noise to each query result in order to maintain differential privacy guarantees at all times. Thus, from the point of view of the data provider there is no difference (in a statistical sense) between the data used by the process mining algorithm regardless of whether data is included or not. This enables to safely reuse the collected data for process mining without jeopardizing the privacy of stakeholders.

With regard to *RQ 2* (How can event log privacy be ensured with a minimum loss of utility for process mining?) we will ensure privacy versus utility by varying the ϵ parameter.

5 Differential Privacy for Event Log Queries

First, we discuss the kind of queries required by typical process mining algorithms (Sect. 5.1) and the associated threats to the privacy of both patients and staff (Sect. 5.2). Then, in Sect. 5.3 we present strategies to provide differential privacy guarantees and discuss the implications for data providers and the process mining result.

5.1 Event Log Queries

Whereas an event log as shown in Table 1 is sufficient input for all process mining algorithms, there are varying requirements for the information absolutely necessary depending on the kind of algorithm. Generally, there are two types of input requirements for *process discovery* algorithms:

Table 2 A simplified event log to illustrate the privacy threats and protection model

Patient	Activity	Time	Staff	Further attributes
$P1$	A	0	$S1$	...
$P1$	B	5	$S2$	...
$P1$	C	5	$S3$	...
$P2$	A	15	$S2$	...
$P2$	B	7	$S3$	...
$P2$	C	10	$S2$	...
$P3$	A	15	$S2$	...
$P3$	B	7	$S3$	...
$P4$	A	10	$S1$	...
$P4$	D	10	$S3$	...
$P5$	A	0	$S4$	...

Potential sensitive information from both patients and staff may contained in the execution of activities and their timing

Table 3 *Activity sequences* or *trace variants* (left) and *directly-follows* relations (right) are possible input requirement of process discovery algorithms

Sequence	Frequency	Relation	Frequency
$\langle A,B,C\rangle$	2	$(-,A)$	4
$\langle A,B\rangle$	1	(A, B)	3
$\langle A,D\rangle$	2	(A, D)	1
$\langle A\rangle$	2	(B, C)	2
		(B, C)	2
		$(A,-)$	1
		$(B,-)$	1
		$(C,-)$	2
		$(D,-)$	1

We use $-$ to denote no predecessor or successor activity

1. those that only require successor and predecessor relations of activities and their frequency (directly-follows frequencies) and
2. those that require full sequences of activity occurrences and their frequency.

To illustrate the information required, we use the simplified event log in Table 2 and assume a fixed case notion in which the patient identifier induces cases. Table 3 illustrates the difference between the input requirements for this log. There is less information available when using only the directly-follows frequencies since the case context in which activity executions were recorded is missing. Therefore, using only directly-follows frequencies prevents certain types of analysis such as replay animation and alignment-based conformance checking (van der Aalst et al. 2012). Based on these information requirements, we can identify two queries specific to process mining that the privacy engine needs to support.

Both queries require a totally ordered set of traces $L_{E,C}$ based on the event log E and with a fixed case notion C and a single attribute $A_{act} \in A$ signaling the activity of interest as input.[2] We define $N = dom(A_{act})$ as a short-hand notation for the set of all possible activity names. Furthermore, in addition to the queries, the proposed privacy engine should support filtering the event log E based on standard relational algebra operations to provide sub logs, e.g., through the WHERE construct of SQL. Such filtering is also enabled on sensitive information contained in the event log and orthogonal to the two queries.

Definition 2 (*Query 1 – directly-follows relation frequencies*) The first query $dfr_L : N \times N \longrightarrow \mathbb{N}$ retrieves the frequency with which we observe an activity $a \in N$ to be followed by an activity $b \in N$ in the event log:

$$dfr_L(a,b) = |\{(e_a, e_b) \in E \times E \mid \langle e_1, \ldots, e_a, e_b, \ldots e_n\rangle \in L \wedge a = e_a.act \wedge b = e_b.act\}|$$

Query 1 provides the most basic information required by process discovery algorithms to construct a process model. Typically, an algorithm would query the directly-follows frequencies for any combination of activities a and b, as well as introduce artificial start- and end activities for each trace. For n activities this results in a matrix of maximum size n^2 as shown on the right side in Table 3.

Definition 3 (*Query 2 – trace variant frequencies*) The second query $seq_L : N^* \longrightarrow \mathbb{N}$ takes a sequence of activities $\langle a_1, \ldots, a_n\rangle$ as input and returns their observed frequency:

$$seq_L(\langle a_1, \ldots, a_n\rangle) = |\{(e_1, \ldots, e_n) \in L \mid \forall_{1 \leq i \leq n}(a_i = e_i.act)\}|$$

Query 2 avoids loosing information in the trace context in which an event occurred. Note that the sequence of activities ignores all other event attributes that are not relevant to discover the control-flow of the process. Different from Query 1, the set of all possible trace variants is infinite and cannot be fixed based on the finite set of activities known for a specific process at hand. In practice, the process might contain looping behavior or parallelism leading to a high number of trace variants. Therefore, the set of trace variants that should be queried is needed. We will provide a concrete method to overcome this issue later in Sect. 5.3. First, we discuss the privacy threats that we aim to counter as well as assumptions made by our method.

[2] In the case of a combination of multiple attributes signalling the activity, we can always create a single attribute by concatenation of the multiple attributes.

5.2 Privacy Threats

At first glance, it may seem that restricting the access to the event log to the two queries discussed in the previous section already protects privacy of process participants. In fact, no personal identifiers are returned. However, as illustrated in the context of the healthcare process in Sect. 3 when assuming that there are rarely visited trajectories in the process, e.g., a patient with a rare disease, it would be possible to identify the information on individuals by repeatedly querying of the event log.

In the context of our process mining use case in healthcare, we can distinguish privacy threats from a *patient perspective* and from a *staff perspective*. Typically, as in the example event logs in Tables 1 and 2, a case is associated with a single patient and each event of a case is associated to some hospital staff member. Thus, each trace (activity sequence) of the event log can be seen as personal data of the patient and the sets of events associated with staff members as their personal data. Whereas the privacy protection for hospital staff is an important issue, our primary goal in this work is to *protect the privacy of patients and analyze the privacy threats to them according to the differential privacy framework*. That is, we want to quantify the privacy risk of an individual contributing their data to the event log and have bound it to the value of the ϵ parameter, which may be chosen according to organizational or societal agreements.

Choosing epsilon for differential privacy is not-trivial. The choice of ϵ is essentially a social question and a too high value of ϵ might lead to unwanted disclosure. Usually, ϵ is a small value close to zero [(0.01, 0.1 or in some cases $ln(2)$ or $ln(3)$], which implies that e^{ϵ} is a value close to 1. If the probability that some bad event will occur is very small, it might be tolerable to increase it by factors like 2 or 3, while if the probability is already felt to be close to unacceptable, then an increase by a factor of $e^{0.01} \approx 1.01$ might be tolerable, while an increase of $e^{0.1}$ would be intolerable. However, the smaller epsilon is chosen the more noise is added when using the Laplacian mechanism. For example, when choosing an ϵ of 0.01, the added Laplacian noise $Laplace(0, \beta)$ with parameter $\beta = \frac{1}{0.01} = 100$ may cause to vanish the "real" values in many cases as the variation incurred by adding noise is larger than that the natural variation of the frequencies in the event log. The added variation decreases with an increase of ϵ and the real values become more visible.

The selection of the ϵ value might also depend on the interests of the involved parties. Hsu et al. (2014) suggest an economic method for the right choice of the ϵ value assuming two individuals with conflicting use of the data. They recommend to use a privacy budget ϵ_{max} for each individual (in our case the patient and the medical staff) that corresponds to the maximum loss of privacy that the individuals are willing to accept. The cost of each query is deducted from the budget until it is exhausted.

5.3 Privacy-Aware Queries Providing Differential Privacy

To safeguard the privacy of patients, we need to add an appropriate amount of noise to the results reported by both queries. As shown in Fig. 5, the privacy engine splits the available event data into disjoint event logs that can be partitioned by time through a periodic update. For each query received, it retrieves the answer from a pre-processed unprotected event log, adds noise to the result, and reduces the pre-configured privacy budget for the selected event log partition according to the chosen ϵ parameter. The smaller the value of ϵ, the smaller the amount is that will be removed from the privacy budget. When the privacy budget for an event log is depleted, no further access is allowed to avoid the risk of identification.

5.3.1 PINQ Framework

We employed the PINQ framework (McSherry 2010) to implement the privacy engine of our privacy protection model.[3] PINQ is a platform that provides a small number of standard declarative data queries which provide differential privacy and can be combined with each other. We show that it is possible to transform each of our queries to a composition of the queries supported by PINQ. This demonstrates that our protection model provides differential privacy guarantees. Additionally, the usage of PINQ avoids the introduction of unnecessary notation and ensures implementation correctness. In fact, we only make use of three operations: `Partition`, `NoisyCount`, and `Where`.

The `Partition` operation provides us with a privacy efficient method to apply an operation on disjoint subsets of the data based on a set of user-defined keys[4] according to which the data is split. It is important to note that if one would sequentially query information from the same data source, the privacy budget is reduced by the sum of the individual ϵ parameters. With each additional query the likelihood of a privacy breach increases. However, it can be shown that by applying the same query in each disjoint subset in parallel, only the maximum of the individual ϵ values needs to be paid (McSherry 2010). The `NoisyCount` operation uses the standard Laplacian mechanism on the original data (cf. Sect. 2) to add symmetric exponential noise to the result of a counting query. The `Where` operation fulfills the filtering requirement as it can be used to filter data with predicates similar to the SQL WHERE statement over the unprotected event log.

5.3.2 Assumptions

To simplify the discussion, we make three assumptions about the content of the event log and the purpose of the process mining analysis.

- First, we assume that there is only one case per patient in the event log with at most c events per event log. The assumption may seem problematic when considering, for example, chronic patients in the dataset. However, our protection model assumes that separate event logs are created periodically (Fig. 5), which reduces the likelihood of subsequent visits being part of the same event log. Even when including multiple visits it is possible to quantify the dilution of the privacy guarantee provided when including multiple cases per patient. The privacy bound would decrease by at most $exp(\epsilon * g)$, where g is the number of rows in which a patient participates in the dataset (Dwork 2008).
- Second, we assume that the set of possible process activities N is publicly known and that we can establish an upper bound for the length of traces of the event log. Both assumptions do not limit our approach in practice. In most cases, the activity names would be known as part of the process documentation. Process executions are bounded in practice and an estimate for the maximum trace length can often be obtained through domain knowledge. For example, in our hospital setting the length could be estimated based on the typical duration of a stay. Overestimation of the maximum trace length would affect the computation time negatively, whereas underestimation would impair the accuracy of the discovered process model as long execution may not be represented correctly.
- Third, we assume that the purpose of process mining is to discover aggregated information about large groups of patients.

Next, we propose strategies to provide differential privacy for both queries.

[3] The source code of the privacy engine based on PINQ is available as C# application at: https://github.com/fmannhardt/pddp/.

[4] The keys for the partitioning operation need to be user-defined since we do not want to leak information on which keys are present in the unprotected event log.

Table 4 Pre-processed input data for the application of PINQ to Query 1

Patient	Source	Target	Further attributes
P1	$\top$	A	...
P1	A	B	...
P1	B	C	...
P1	C	$\perp$	...
P2	$\top$	A	...
P2	A	B	...
P2	B	C	...
P2	C	$\perp$	...
P3	$\top$	A	...
P3	A	B	...
P3	B	$\perp$	...
P4	$\top$	A	...
P4	A	D	...
P4	D	$\perp$	...
P5	$\top$	A	...
P5	A	$\perp$	...

5.3.3 *Query 1: Laplacian Mechanism*

We employ a transformation method `TransformDFG` that pre-processes the set of all traces $L_{E,C}$ of an event log E with activities N to the format shown in Table 4. Note that this pre-processed table is only available to the privacy engine. Let $REL = (N \cup \{\top, \perp\}) \times (N \cup \{\top, \perp\})$ be the set of all possible binary activity relations. Instead of providing the results of Query 1 for each individual pair of activities $(a, b) \in N \times N$, we obtain the full set of directly-follows count $DFR_{public} \subseteq REL \times \mathbb{N}$ at once. This allows us to avoid repeated querying and combine the retrieval of the following single PINQ query:

$$\begin{aligned} DFR_{public} = \texttt{TransformDFG}(L).\texttt{Where}(Pred) \\ .\texttt{Partition}(REL) \\ .\texttt{NoisyCount}(\epsilon) \end{aligned}$$

In the resulting set DFR_{public} the necessary level of noise is added to the frequency for each possible directly-follows relation. We implement Query 1 by looking up the frequency for any directly-follows relation in DFR_{public}:

$$dfr_L(a, b) = n \text{ with } ((a, b), n) \in DFR_{public}$$

Note that some of the frequencies might be negative, these can be disregarded, and that non-existent directly-follows relations in the original data may be added to the query result. However, process discovery algorithms typically disregard such infrequent behavior as noise.

Table 5 Pre-processed input data for the application of PINQ to Query 2

Patient	Sequence	Further attributes
P1	$\langle A, B, C, \perp \rangle$	...
P2	$\langle A, B, C, \perp \rangle$	...
P3	$\langle A, B, \perp \rangle$	...
P4	$\langle A, D, \perp \rangle$	...
P5	$\langle A, \perp \rangle$	...

5.3.4 *Query 2: Prefix-Tree Based Counting*

Similar to our method for the first query, we define a transformation method `TransformTraces` that pre-processes the event log to the format shown in Table 5, which is suitable for the application of PINQ queries. Here, we treat each trace as a sequence of identifiers and add a $\perp$ identifier to the end of each one. However, since the set of possible activity sequences is theoretically infinite, we cannot follow the same procedures as for Query 1 and issue a `Partition` query for all possible sequences. We solve this problem by adopting a scheme similar to the one proposed by McSherry and Mahajan (2011), in which the frequency of k-length strings is counted, as well as the method proposed by Bonomi and Xiong (2013), in which sequential pattern mining with differential privacy guarantees is described.

Definition 4 (*Set of activity sequences of length n*) We define $pref(n) : \mathbb{N} \rightarrow N^*$ to return the set of all possible activity sequences of length n:

$$\begin{aligned} pref(n) = \{\sigma \in (N \cup \{\perp\})^* \mid \sigma = \langle a_1, \ldots, a_n \rangle \\ \wedge \forall_{1 \leq j \leq (n-1)} (a_j \neq \perp)\}. \end{aligned}$$

Activity sequences returned by *pref*(*n*) are either prefixes of complete sequences without the symbol $\perp$ or complete sequences, which end with $\perp$.

We iteratively query prefix sequences $SEQ_i \subseteq N^* \times \mathbb{N}$ using the following PINQ operation:

$$\begin{aligned} SEQ_i = \texttt{TransformTraces}(L).\texttt{Where}(Pred) \\ .\texttt{Partition}(pref(i)) \\ .\texttt{NoisyCount}(\epsilon) \end{aligned}$$

to build the result set $SEQ_{public} = \bigcup_{1 \leq i \leq k} SEQ_i$. Based on SEQ_{public}, we can implement Query 2 as:

$$seq_L(\langle a_1, \ldots, a_n \rangle) = n \text{ with } (\langle a_1, \ldots, a_n \rangle, n) \in SEQ_{public}$$

To avoid that an larger amount of $|N|^k$ subsets are queried than can be dealt with, we extend this method by using a second user-defined parameter p that is applied to prune low-frequency prefixes. The occurrence frequency of prefixes is equal to or higher than the frequency of complete sequences – a prefix needs to be at least as frequent by definition. Therefore, we can reduce the number of prefixes queried in each iteration by pruning the prefix tree to only contain activity sequences with prefixes that occur more often than p-times. Having obtained the frequencies for the considered prefix tree, we only retain complete activity sequences in the result SEQ_{public}, i.e., only sequences ending with $\perp$. In total, this method uses at most k queries as described above and, thus, reduces the privacy budget by $k * \epsilon$.

5.4 Limitations

We acknowledge that there are limitations to our proposed method. Since Query 1 is formulated by means of a data table in which each row corresponds to a single event instead of to a single case, the privacy guarantee for patients is diluted by at most $exp(\epsilon * g)$ as described in the assumptions. Furthermore, the prefix-tree based method for Query 2 is only computationally feasible for a relatively short maximum trace length parameter k or aggressive pruning parameter values p. Furthermore, The likelihood that traces which are not in the original event log are added to the result grows for a larger k parameter value.

6 Evaluation

We evaluated the proposed privacy protection model by testing the impact of our method on the quality of discovered process models compared to a ground truth. As ground truth we use process models discovered in the original, unprotected event log without any privacy protection. We compare both quantitatively based on the standard evaluation measures *fitness* and *precision* as well as qualitatively by discussing the differences. First, we discuss our experimental set-up.

6.1 Experimental Set-Up

As process discovery algorithm, we use Inductive Miner[5] infrequent (Leemans et al. 2013) in its variant supporting directly-follows relations as input (Leemans et al. 2018) with standard parameters and discover models for varying ϵ values. This shows how the ϵ parameter influences the trade-off between privacy and accuracy (see *RQ 2*) and gives an indication which ϵ values are feasible.

We replicate a typical scenario for process discovery and attempt to discover a model representing the main process behaviour by first removing infrequent behaviour by applying the filtering plug-in 'Filter directly follows graph' for Query 1 or the plug-in 'Filter Out Low-Frequency Traces' for Query 2, both with standard parameters. The same filters and the same discovery approach is used on the unprotected event log without added noise. Then, we measure the difference between the discovered process models based on the F1-score calculated with the projected recall and precision measure proposed by Leemans et al. (2018). Since the results returned by our privacy protection method are subject to random noise, we repeat the discovery process 10 times for each parameter setting.

As dataset, we use two publicly available event logs from the IEEE task force on process mining repository:[6] Sepsis Cases (Mannhardt 2016) and Road Traffic Fine Management (de Leoni and Mannhardt 2015). These two event logs represent two different prototypes of event logs for which we expect differences in the performance of our protection model. The sepsis log is typical for the healthcare domain and has many infrequent variants. The road fines log is more structured and only few trace variants exist. We use it as example for a simpler process that is to a large degree standardised and exhibits less infrequent variants.

Sepsis Cases This is a hospital event log with approximately 1000 cases for trajectories of patients who are suspected to have a life-threatening sepsis condition, from the emergency room of a hospital until discharge. It is a challenging dataset for our method since out of the total 1050 traces, there are 846 unique trace variants. The maximum trace length is 185 and on average traces contain 14.5 events. The main source for the large number of trace variants are three activities regarding the collection of laboratory results (Mannhardt and Blinde 2017), which occur in parallel to the remainder of the process. We use a maximum sequence length of $k = 15$ and a pruning parameter $p = 30$. We base these parameters on the average trace length and feasibility of computation. Only when using $\epsilon = 0.01$ did we need to increase the pruning parameter to $p = 350$ to keep the computation time within a few seconds.

Road Traffic Fines Road Traffic Fine Management (de Leoni and Mannhardt 2015), is an event log obtained for the process of handling road traffic fines in a local Italian police. We use a random sub-sample of 10,000 cases from

[5] We chose the Inductive Miner since it is the only process discovery algorithm available in the open-source framework ProM 6.8 that allows to use both directly-follows relations and trace variants as input.

[6] https://data.4tu.nl/repository/collection:event_logs_real.

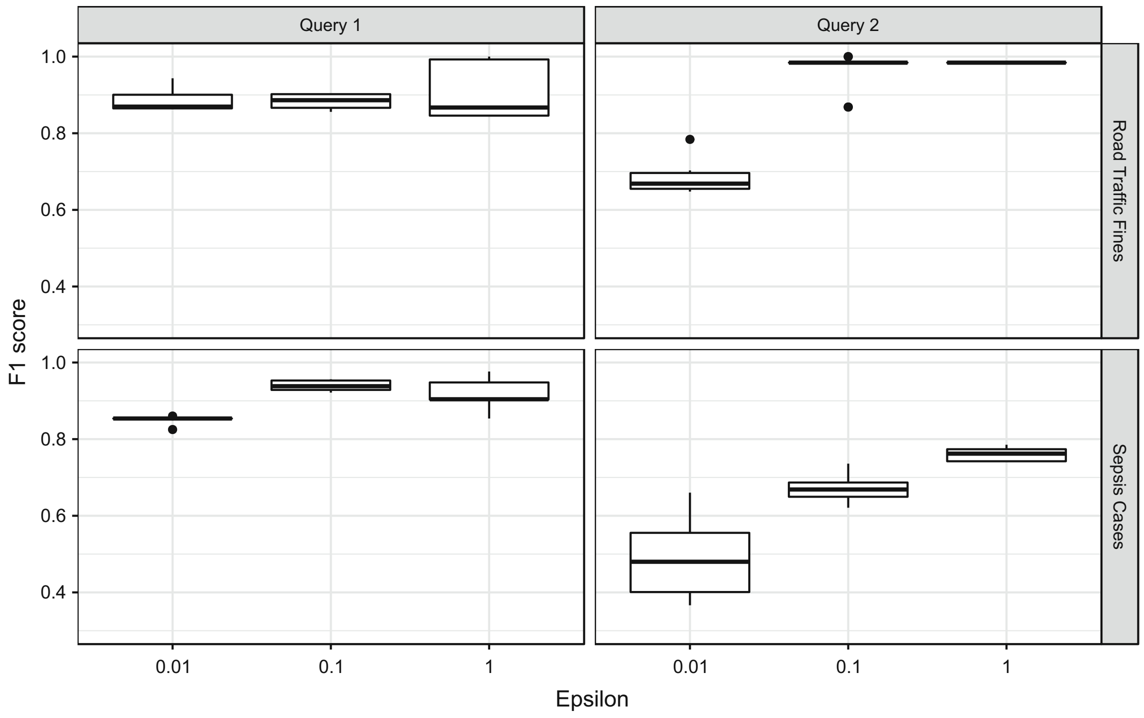


Fig. 6 F1 score based on the projected recall and precision measures when comparing the process model discovered on the unprotected data without our protection method and on the privacy-protected event log for both processes. Each box plot is based on 10 repetitions

the event log. In contrast to the Sepsis Cases log, the Road Fines event log is very structured. In the sample of 10,000 traces there are only 69 distinct trace variants. Furthermore, the maximum length of a trace is 10 and the average length is 3.7. Adding noise should affect the process discovery result from this log less. Here we use a maximum sequence length of $k = 10$ and pruning parameter $p = 200$ to keep the computation time within a few seconds.

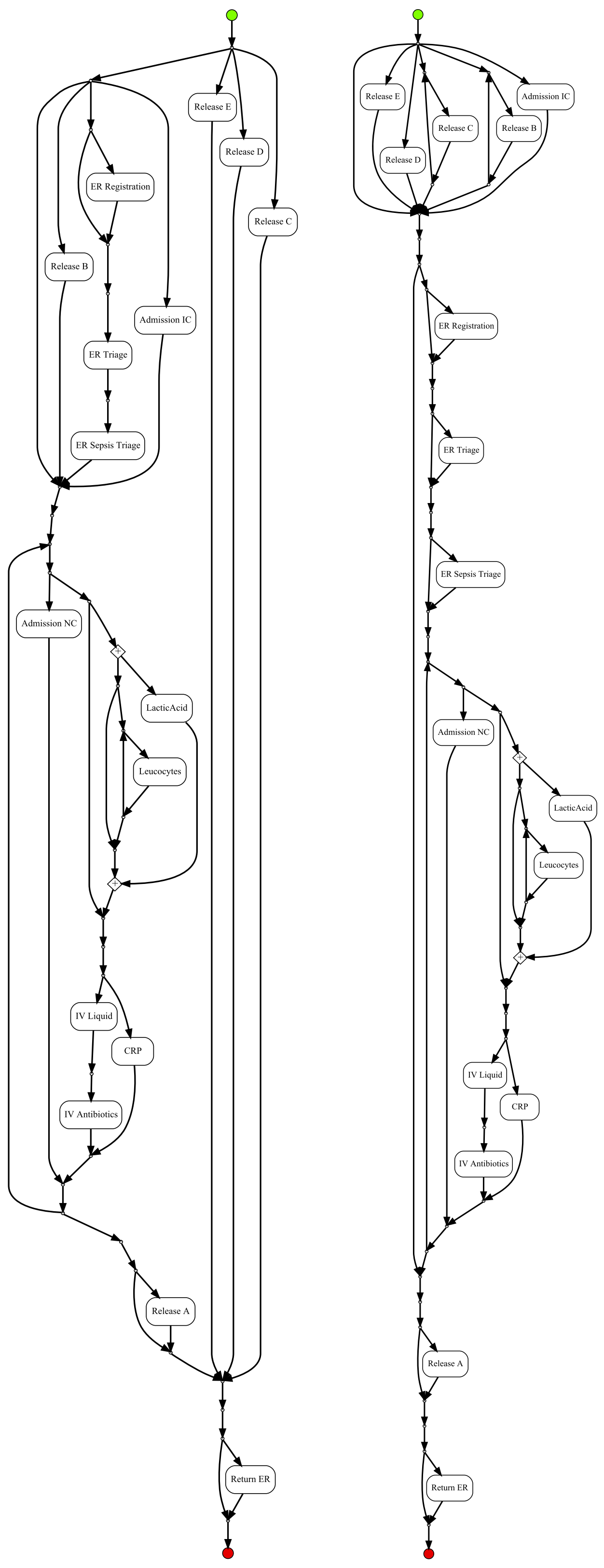
6.2 Results and Discussion

The computed F1 score indicating the difference between the process models discovered in the original unprotected event log without the usage of our protection model, i.e., without privacy guarantees, and the models discovered when using the proposed privacy protection model are shown in Fig. 6. For each combination of query and event log a box plot indicates the effect of our method and the value of ϵ on the discovery result.

The results show clear differences between both event logs and between the kind of query used. For the proposed directly-follows querying approach (Query 1), there is only little difference between the event logs used, and changes in ϵ have relatively little impact on the quality of the discovered model. This indicates that the noise added to the directly-follows relation frequencies can, largely, be filtered by the noise filtering capabilities of the Inductive Miner discovery algorithm. In fact, there are only small changes between the models[7] in Fig. 7 in which the best process models discovered for $\epsilon = 0.1$ and $\epsilon = 1.0$ using Query 1 are compared with the process model discovered for the unprotected directly-follows relations from the Sepsis Cases event log. For an ϵ value of 0.01 a difference begins to appear between Road Traffic Fines log and Sepsis Cases. Whereas the quality of the discovered Sepsis Cases process model decreases, there is still little change for the Road Traffic Fines model.

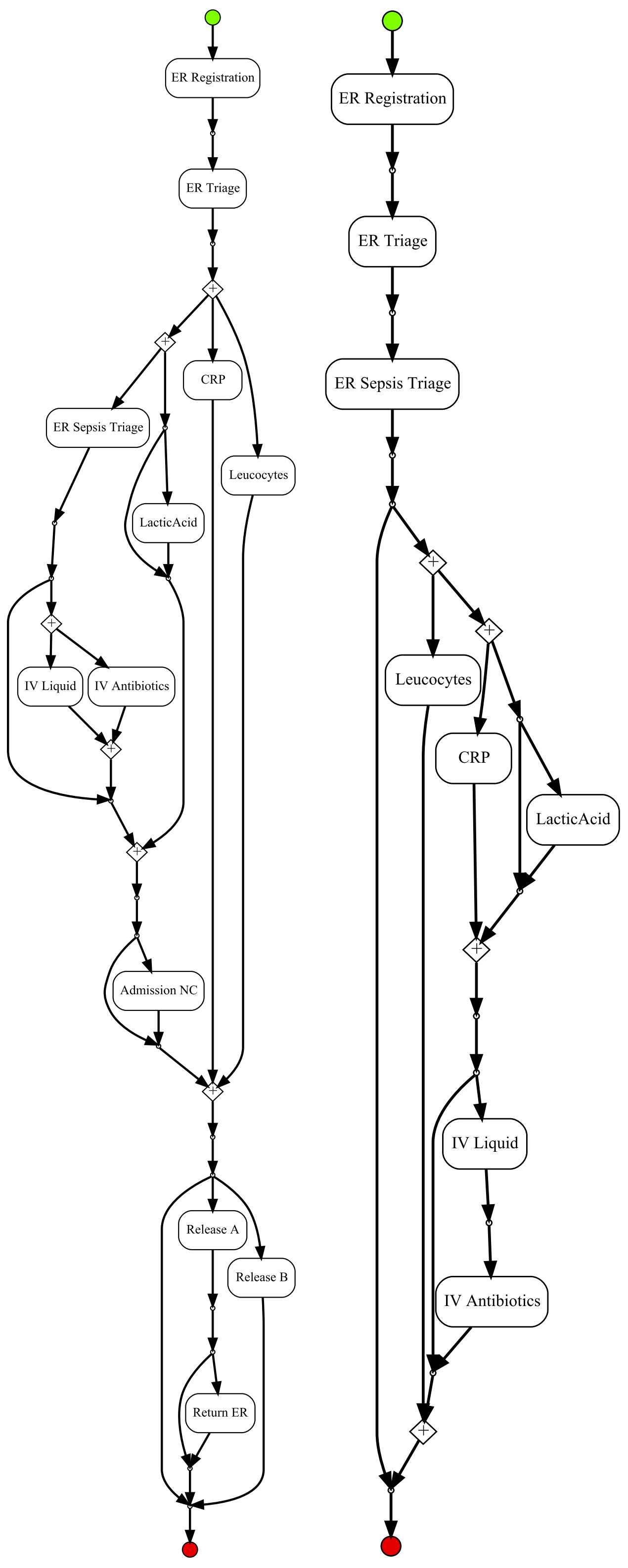
Regarding the *RQ 2* (How can event log privacy be ensured with a minimum loss of utility for process mining?) we varied the ϵ parameter. When using the proposed sequence querying approach (Query 2), there are larger differences both when reducing ϵ and between the two event logs. The result for the Road Traffic Fines event log is much better and also very stable across repetitions, which indicates that the noise added by our proposed method has only little influence on the quality of the discovered process model. In the case of the Sepsis Cases event log it can be observed that our technique produces relatively high error rates, i.e., a low F1 score when comparing to the ground truth model. This is not surprising, since the Sepsis Cases log represents a flexible process

[7] The Petri net models are visualized using the compact Inductive Visual Miner notation as described in Leemans et al. (2014).



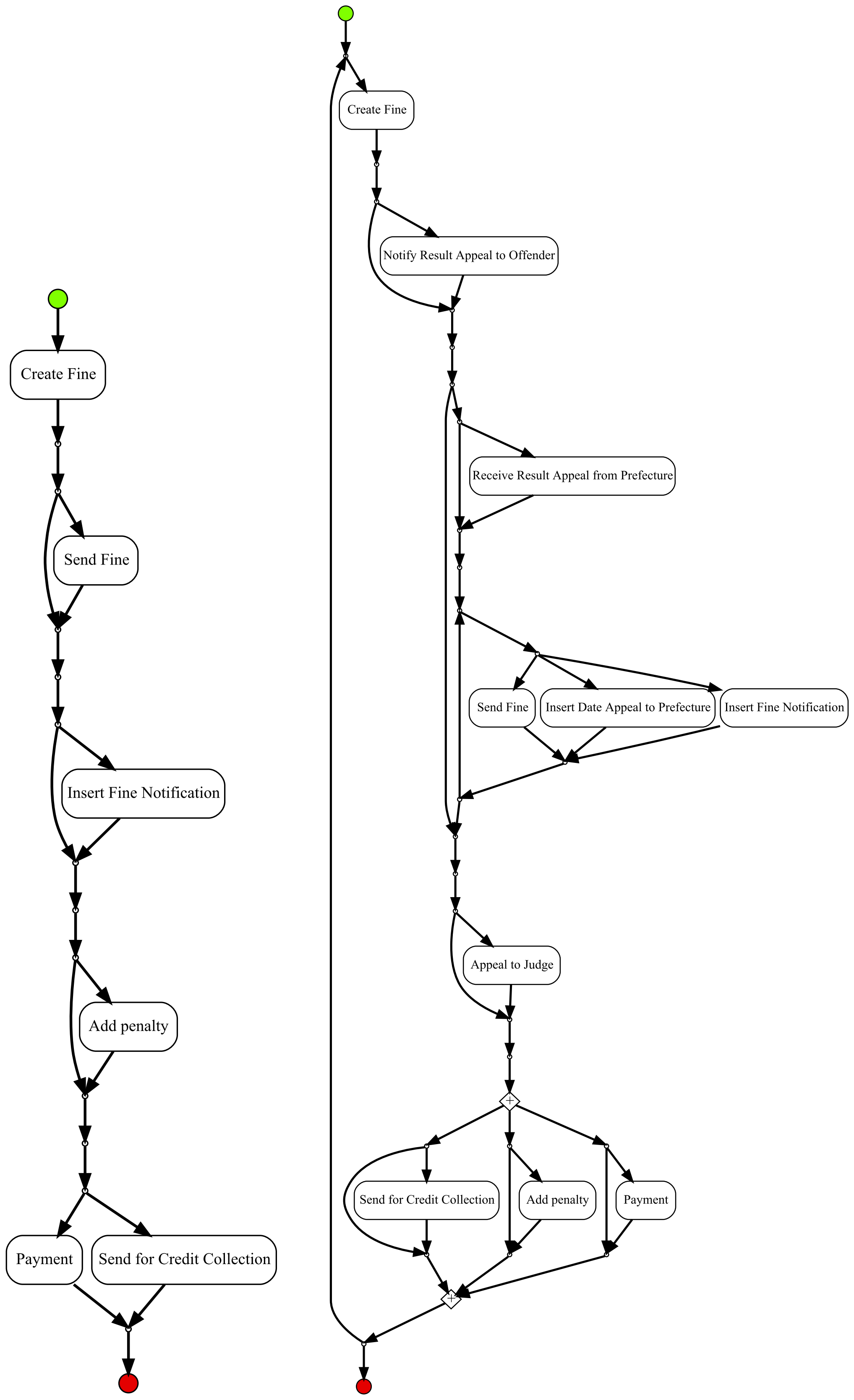
(a) Unprotected Data **(b)** $\epsilon = 0.1$

Fig. 7 Process models discovered in the unprotected sepsis event log and the privacy-protected log using Query 1 for $\epsilon = 0.1$



(a) Unprotected Data **(b)** $\epsilon = 1.0$

Fig. 8 Process models discovered in the unprotected sepsis data and the privacy-protected data using Query 2 with $\epsilon = 1.0$


Create Fine
Send Fine
Insert Fine Notification
Add penalty
Payment
Send for Credit Collection
Create Fine
Notify Result Appeal to Offender
Receive Result Appeal from Prefecture
Send Fine
Insert Date Appeal to Prefecture
Insert Fine Notification
Appeal to Judge
Send for Credit Collection
Add penalty
Payment

(a) Unprotected Data **(b)** $\epsilon = 0.01$

Fig. 9 Process models discovered in the unprotected road fines log and the privacy-protected log using Query 2 for $\epsilon = 0.01$

with many parallel branches. To mine process models from such event logs is already a challenge for existing process mining algorithms without added noise. When using $\epsilon = 0.01$ many sequences not originally in the unprotected event log are generated and we need to increase the pruning threshold p to 350 for performance reasons. This hides many of the actual traces, i.e., only trace variants occurring more than 350 times would be retained. Figure 8 shows the results obtained for the Sepsis Cases log based on Query 2 for $\epsilon = 0.1$. Infrequent trace variants and some infrequently occurring activities are hidden by the injected noise; still, parts of the main process flow remain intact. For example, in both process models the execution can start with the sequence of *ER Registration*, *ER Triage*, and *ER Sepsis Triage*. However, when using our protection model, the discharge activities *Release A* and *Release B* are no longer of the 80% most frequent trace variants.

In contrast, Fig. 9 shows that the error caused by Query 2 on the Road Traffic Fine log is small. It is noteworthy that by using Query 2 with an ϵ value of 0.1 we often obtain the exact same result as when using the unprotected event log. In this case, the F1 score is consistently 1.0 indicating that our approach can be used to protect the privacy of individual participants while still discovering the correct main process behavior for very structured processes with a small number of variants. When lowering the ϵ further to 0.01 as shown in Fig. 9, differences appear due to the added noise by our protection approach. In particular, some of the less frequent activities connected to the *appeals* part of the Road Traffic Fines process, for example *Notify Result Appeal to Offender* and *Receive Result Appeal From Prefecture*, appear in the discovered process model. Some of the noise added by our privacy protection method cannot longer be distinguished from the regular process behavior. Still, other parts of the frequent process behavior are left intact. For example, the process model starts with *Create Fine* and may end with either *Payment* or *Send for Credit Collection* as in the model discovered on the unprotected log.

7 Related Work

The paper proposes the first privacy-preserving process mining approach. Privacy-preserving *data mining* techniques (PPDM) have been considered to a large extent in the literature and have been accompanied by several experimental implementations (Zhiqiang and Longjun 2018) and platforms. Different evaluation parameters for PPDM algorithms can be found in literature. According to Verykios et al. (2004) the evaluation parameters are *performance* required to secure data, *data utility*, *uncertainty level* for the prediction of hidden data, and *resistance* in terms of tolerance against the data mining algorithms. Bertino et al. (2008) extend this list of evaluation criteria by *hiding failure*, which is “the portion of sensitive information that is not hidden by the application of a privacy preservation technique”. Algorithms for PPDM either adopt distributed frameworks or add random noise to the data (Bhowmick et al. 2006) in order to prevent the loss of user’s privacy before publishing data. To randomize data by adding noise either a known statistical distribution is used (Agrawal and Srikant 2000) or noise is multiplied with a known statistical distribution (Kim et al. 2003). There is also a vast amount of literature on privacy in databases. In this area, several efforts were made in the last decade to integrate privacy when designing databases by using multi-level security (Macedo et al. 2017) or role-based access control (Colombo and Ferrari 2015) approaches.

Since this paper applies the notion of *differential privacy for event logs*, we studied the respective domain of interest. Approaches relying on differential privacy can be found for health data (Dankar and El Emam 2013), location-based services (ElSalamouny and Gambs 2016) and smart meters (Zhang et al. 2017), which are domains with high demand for data protection. This paper uses a hospital event log for the evaluation of our approach.

Related to event log data and process mining, a large body of research exists for security-oriented analysis. The tool of Stocker and Accorsi (2014) enables the configuration of security concerns (i.e., authentication, binding of duty and separation of duties) when generating synthetic event logs. A different event log configuration according to security concerns is suggested in Fazzinga et al. (2018) who use security risk as criterion to filter related traces. To support decision making in security audits, Accorsi et al. (2013) suggest to mine the control- and the data-flow since only both perspectives make it possible to analyze security requirements. The application domain of security-oriented analysis of event logs is intrusion detection (Myers et al. 2017) or smart metering (Eibl et al. 2017). While a large body of research exists for security-oriented analysis, privacy concerns have been scarcely considered for process mining. Only the work of Mannhardt et al. (2018) discusses privacy challenges for process mining, however, without suggesting any approach for event log protection. A privacy-preserving system design for process mining has been suggested in Michael et al. (2019), which allows to specify who does what, when, why, where and how with personal data during process mining. Our approach could be integrated into the privacy-preserving system design as a privacy engineering technique to protect the event logs.

To sum up, privacy-preserving techniques for process mining have received little attention and the approach presented in this paper is the first one so far.

8 Conclusion

An increasing amount of data is continuously collected and stored by organizations and poses security and privacy challenges. While methods for knowledge extraction from data which preserve privacy have been widely considered for data mining (Mendes and Vilela 2017), privacy-preserving process mining is still in its infancy.

Contribution This paper contributes a privacy-preserving technique for process mining and an approach to protect event logs. In this way, we address technological challenges of privacy-preserving process mining. To show which privacy leakages exist while conducting process mining, we use a hospital health process use case. Clearly, this domain has a high demand for privacy protection. We have applied the concept of privacy checkpoints on an event schema of hospital health processes and identified six stages of data paths. The privacy checkpoint between data storage and data (re)use can be considered as possible privacy leakages while conducting process mining. We map the stages of data passes to the data value chain suggested in D'Acquisto et al. (2015b) and identify the *abstract* privacy design patterns as possible candidates where protection is essential. This provides an answer to *RQ 1* (At which stage of data paths is a protection model for event log privacy required?).

We present a protection model including a trusted environment for primary use purposes and an untrusted environment using a differential privacy approach, which is the strongest privacy model available to date which provides provable privacy guarantees. Here historical data may be used in an exploratory fashion without clear analysis question. Thus, it is difficult to attain consent for it afterward. We suggest to introduce a privacy engine as single access point between the two environments. This engine introduces noise to each query result according to the differential privacy framework which safeguards the privacy of patients. With this approach, it is possible to safely reuse the collected data for process mining purposes.

For evaluation purposes, we have applied our method to two publicly available real-life event logs and applied the Inductive Miner algorithm to both of them. The evaluation shows that our method can be used to discover the frequent behavior of a process while providing privacy for individual participants. For event logs from highly structured processes with few trace variants the error introduced is small, whereas for event logs with a large number of infrequent behavior leading to many trace variants the introduced noise affects the utility of the discovered process model. With regard to *RQ 2* (How can event log privacy be ensured with a minimum loss of utility for process mining?) we can conclude that the choice of the ϵ value and the structure of the event log affect the trade-off between utility and privacy.

Future work There are several avenues for future work. More accuracy may be achieved when integrating a process discovery algorithm into the differential privacy framework by placing the process mining engine in the trusted environment depicted in Fig. 5. This could help reducing the amount of noise that needs to be injected by means of exploiting properties of particular process discovery algorithms. There are several examples for such tailored approaches in the data mining domain, e.g., k-means (Blum et al. 2005) and sequential pattern mining (Bonomi and Xiong 2013) with differential privacy guarantees. Also, we plan to investigate more closely the relationship between the parameters of our method: ϵ, k, and p in the accuracy-privacy trade-off in the resulting process model. So far, we have only considered differential privacy for discovering the control-flow perspective of processes from event logs. Many useful applications of process mining rely on other perspectives such as performance information or data values. It would be possible to extend our method towards these perspectives to provide differential privacy for aggregated information, e.g., the average time between activities.

Another interesting aspect is to focus also on other privacy design strategies. Privacy models could be used to generate an information platform by means of MDA (Adam et al. 2018) that enables end users to either (a) define privacy policies, in order to determine more precisely who can do what with which data (guided consent process) and (b) monitor compliance with them, in order to allow them to see which privacy mechanisms are provided for which process stages regarding the process mining of their data. The domain specific concepts for such a privacy model can be easily extracted from event logs to be used in conceptual models, considering relevant contexts (Michael and Steinberger 2017), as event logs include data with regard to several concepts in a condensed way. This approach would support the privacy design strategies *control* and *enforce* (cf. Sect. 2.1).

To sum up, this paper presents a first technical contribution for privacy-preserving process mining using a differential privacy approach and outlines a roadmap for future research on that field.

References

Accorsi R, Stocker T, Müller G (2013) On the exploitation of process mining for security audits: the process discovery case. In: Shin Sung Y, Maldonado JC (eds) Proceedings of the 28th annual ACM symposium on applied computing, SAC '13, Coimbra, Portugal, March 18–22. ACM, pp 1462–1468

Adam K, Netz L, Varga S, Michael J, Rumpe B, Heuser P, Letmathe P (2018) Model-based generation of enterprise information systems. In: Fellmann M, Sandkuhl K (eds) Enterprise modeling and information systems architectures (EMISA'18), volume 2097 of CEUR workshop proceedings, pp 75–79. CEUR-WS.org

Agrawal R, Srikant R (2000) Privacy-preserving data mining. In: Proceedings of the 2000 ACM SIGMOD international conference on management of data, SIGMOD '00. ACM, New York, NY, pp 439–450

Aldeen YAAS, Salleh M, Razzaque MA (2015) A comprehensive review on privacy preserving data mining. SpringerPlus 4(1):694

Arasu A, Babcock B, Babu S, Cieslewicz J, Datar M, Ito K, Motwani R, Srivastava U, Widom J (2016) STREAM: the Stanford data stream management system. In: Garofalakis MN, Gehrke J, Rastogi R (eds) Data stream management: processing high-speed data streams, data-centric systems and applications. Springer, Berlin, pp 317–336

Augusto A, Conforti R, Dumas M, La Rosa M, Maggi FM, Marrella A, Mecella M, Soo A (2017) Automated discovery of process models from event logs: review and benchmark. IEEE Trans Knowl Data Eng (accepted)

Bergeron E (2000) The difference between security and privacy

Bertino E, Lin D, Jiang W (2008) A survey of quantification of privacy preserving data mining algorithms. Springer, Boston, MA, pp 183–205

Bhowmick SS, Gruenwald L, Iwaihara M, Chatvichienchai S (2006) PRIVATE-IYE: a framework for privacy preserving data integration. In: 22nd international conference on data engineering workshops (ICDEW'06), pp 91–91

Blum A, Dwork C, McSherry F, Nissim K (2005) Practical privacy: the SuLQ framework. In: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems. ACM, pp 128–138

Bonomi L, Xiong L (2013) A two-phase algorithm for mining sequential patterns with differential privacy. In: Proceedings of the 22nd ACM international conference on conference on information & knowledge management-CIKM '13. ACM Press, New York

Colombo P, Ferrari E (2015) Privacy aware access control for big data: a research roadmap. Big Data Res 2:145–154

D'Acquisto G, Domingo-Ferrer J, Kikiras P, Torra V, de Montjoye Y-A, Bourka A (2015a) Privacy by design in big data: an overview of privacy enhancing technologies in the era of big data analytics. CoRR arXiv:abs/1512.06000

D'Acquisto G, Domingo-Ferrer J, Kikiras P, Torra V, de Montjoye Y-A, Bourka A (2015b) Privacy by design in big data: an overview of privacy enhancing technologies in the era of big data analytics

Dankar FK, El Emam K (2013) Practicing differential privacy in health care: a review. Trans Data Priv 6(1):35–67

de Leoni M, Mannhardt F (2015) Road traffic fine management process. Eindhoven University of Technology, Eindhoven **(Dataset)**

Dwork C (2008) Differential privacy: a survey of results. In: International conference on theory and applications of models of computation, Springer, Berlin, pp 1–19

Dwork C, Naor M, Pitassi T, Rothblum GN (2010) Differential privacy under continual observation. In: Proceedings of the 42nd ACM symposium on theory of computing-STOC '10. ACM Press, New York

Dwork C, Roth A et al (2014) The algorithmic foundations of differential privacy. Found Trends® Theor Comput Sci 9(3–4):211–407

Eibl G, Ferner C, Hildebrandt T, Stertz F, Burkhart S, Rinderle-Ma S, Engel D (2017) Exploration of the potential of process mining for intrusion detection in smart metering. In: ICISSP

ElSalamouny E, Gambs S (2016) Differential privacy models for location-based services. Trans Data Priv 9(1):15–48

Fazzinga B, Flesca S, Furfaro F, Pontieri L (2018) Online and offline classification of traces of event logs on the basis of security risks. J Intell Inf Syst 50(1):195–230

Hoepman J-H (2014) Privacy design strategies. In: Cuppens-Boulahia N, Cuppens F, Jajodia S, Kalam AAE, Sans T (eds) ICT systems security and privacy protection. Springer, Berlin, pp 446–459

Hoepman J-H (2018) Making privacy by design concrete. In: European cyber security perspectives 2018. Radboud Repository, pp 26–28

Hsu J, Gaboardi M, Haeberlen A, Khanna S, Narayan A, Pierce BC, Roth A (2014) Differential privacy: an economic method for choosing epsilon. In: Proceedings of the 2014 IEEE 27th computer security foundations symposium, CSF '14. IEEE Computer Society, Washington, DC, pp 398–410

ISO/IEC 27000 (2018) Information technology-security techniques-information security management systems-overview and vocabulary, fifth edn. Standard, International Organization for Standardization

Kim JJ, Kim JJ, Winkler WE, Winkler WE (2003) Multiplicative noise for masking continuous data. Technical report, Statistical Research Division, US Bureau of the Census, Washington, DC

Leemans SJJ, Fahland D, vander Aalst WMP (2013) Discovering block-structured process models from event logs containing infrequent behaviour. In: BPM 2013 workshops, volume 171 of LNBIP. Springer, pp 66–78

Leemans SJJ, Fahland D, van der Aalst WMP (2014) Process and deviation exploration with inductive visual miner. In: BPM 2014 demos, volume 1295 of CEUR workshop proceedings, p 46. CEUR-WS.org

Leemans SJJ, Fahland D, van der Aalst WMP (2018) Scalable process discovery and conformance checking. Softw Syst Model 17(2):599–631

Macedo R, Paulo J, Pontes R, Portela B, Oliveira T, Matos M, Oliveira R (2017) A practical framework for privacy-preserving NoSQL databases. In: SRDS. IEEE Computer Society, pp 11–20

Mannhardt F (2016) Sepsis cases-event log. Eindhoven University of Technology, Eindhoven **(Dataset)**

Mannhardt F, Blinde D (2017) Analyzing the trajectories of patients with sepsis using process mining. In: RADAR+EMISA 2017, volume 1859 of CEUR workshop proceedings, pp 72–80. CEUR-WS.org

Mannhardt F, Petersen S, de Oliveira MFD (2018) Privacy challenges for process mining in human-centered industrial environments. In: 14th international conference on intelligent environments (IE). IEEE Xplore, pp 64–71

Mans RS, van der Aalst WMP, Vanwersch RJB, Moleman AJ (2013) Process mining in healthcare: data challenges when answering frequently posed questions. In: Lenz R, Miksch S, Peleg M, Reichert M, Riaño D, ten Teije A (eds) Process support and knowledge representation in health care. Springer, Berlin, pp 140–153

McSherry F (2010) Privacy integrated queries. Commun ACM 53(9):89

McSherry F, Mahajan R (2011) Differentially-private network trace analysis. ACM SIGCOMM Comput Commun Rev 41(4):123–134

Mendes R, Vilela JP (2017) Privacy-preserving data mining: methods, metrics, and applications. IEEE Access 5:10562–10582

Mettler M (2016) Blockchain technology in healthcare: the revolution starts here. In: 2016 IEEE 18th international conference on e-health networking, applications and services (Healthcom), pp 1–3

Michael J, Steinberger C (2017) Context modeling for active assistance. In: Cabanillas C, España S, Farshidi S (eds)

Proceedings of the ER forum 2017 and the ER 2017 demo track co-located with the 36th international conference on conceptual modelling (ER 2017), pp 221–234

Michael J, Koschmider A, Mannhardt F, Baracaldo N, Rumpe B (2019) User-centered and privacy-driven process mining system design for IoT. In: information systems engineering in responsible information systems-CAiSE forum 2019, Rome, Proceedings, pp 194–206

Myers D, Radke K, Suriadi S, Foo E (2017) Process discovery for industrial control system cyber attack detection. In: De Capitani di Vimercati S, Martinelli F (eds) ICT systems security and privacy protection. Springer, Cham, pp 61–75

Peterson ZNJ, Gondree M, Beverly R (2011) A position paper on data sovereignty: the importance of geolocating data in the cloud. In: Proceedings of the 3rd USENIX conference on hot topics in cloud computing, HotCloud'11. USENIX Association, Berkeley, CA, pp 9–9

Rozinat A, van der Aalst WMP (2006) Decision mining in ProM. In: Lecture notes in computer science. Springer, Berlin, pp 420–425

Sacco O, Breslin JG, Decker S (2013) Fine-grained trust assertions for privacy management in the social semantic web. In: 2013 12th IEEE international conference on trust, security and privacy in computing and communications, pp 218–225

Sicari S, Rizzardi A, Grieco LA, Coen-Porisini A (2015) Security, privacy and trust in Internet of Things: the road ahead. Comput Netw 76:146–164

Stocker T, Accorsi R (2014) SecSy: a security-oriented tool for synthesizing process event logs. In: Limonad L, Weber B (eds) Proceedings of the BPM demo sessions 2014 co-located with the 12th international conference on business process management (BPM 2014), Eindhoven, The Netherlands, September 10, 2014, volume 1295 of CEUR workshop proceedings, p 71. CEUR-WS.org

van der Aalst WMP (2016) Process mining: data science in action, 2nd edn. Springer, Berlin

van der Aalst W, Adriansyah A, van Dongen B (2012) Replaying history on process models for conformance checking and performance analysis. Wiley Interdiscip Rev Data Min Knowl Discov 2(2):182–192

van Eck ML, Lu X, Leemans SJJ, van der Aalst WMP (2015) PM^2: a process mining project methodology. In: Advanced information systems engineering. Springer, pp 297–313

Verykios VS, Bertino E, Fovino IN, Provenza LP, Saygin Y, Theodoridis Y (2004) State-of-the-art in privacy preserving data mining. SIGMOD Rec 33(1):50–57

Yu WE (2014) Data privacy and big data-compliance issues and considerations. ISACA J 3:27–31

Yu X, Wen Q (2010) A view about cloud data security from data life cycle. In: 2010 international conference on computational intelligence and software engineering, pp 1–4

Zhang Z, Qin Z, Zhu L, Weng J, Ren K (2017) Cost-friendly differential privacy for smart meters: exploiting the dual roles of the noise. IEEE Trans Smart Grid 8(2):619–626

Zhiqiang G, Longjun Z (2018) Privacy preserving data mining on big data computing platform: trends and future. In: Barolli L, Woungang I, Hussain OK (eds) Advances in intelligent networking and collaborative systems. Springer, Cham, pp 491–502