Privacy-Preserving Continuous Event Data Publishing

Majid Rafiei ✉ and Wil M.P. van der Aalst

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

Abstract. Process mining enables organizations to discover and analyze their actual processes using event data. Event data can be extracted from any information system supporting operational processes, e.g., SAP. Whereas the data inside such systems is protected using access control mechanisms, the extracted event data contain sensitive information that needs to be protected. This creates a new risk and a possible inhibitor for applying process mining. Therefore, privacy issues in process mining become increasingly important. Several privacy preservation techniques have been introduced to mitigate possible attacks against static event data published only once. However, to keep the process mining results up-to-date, event data need to be published continuously. For example, a new log is created at the end of each week. In this paper, we elaborate on the attacks which can be launched against continuously publishing anonymized event data by comparing different releases, so-called *correspondence attacks*. Particularly, we focus on group-based privacy preservation techniques and show that provided privacy requirements can be degraded exploiting correspondence attacks. We apply the continuous event data publishing scenario to existing real-life event logs and report the anonymity indicators before and after launching the attacks.

Keywords: Process mining · Privacy preservation · Correspondence attacks · Event data

1 Introduction

Process mining bridges the gap between *data science* and *process science* using event logs. Event logs are widely available in different types of information systems [1]. Events are the smallest units of process execution which are characterized by their attributes. Process mining requires that each event contains at least the following main attributes to enable the application of analysis techniques: *case id*, *activity*, and *timestamp*. The *case id* refers to the entity that the event(s) belongs to, and it is considered as a process instance. The *activity* refers to the activity associated with the event, and the *timestamp* is the exact time when the activity was executed for the case. Moreover, depending on the context of a process, the corresponding events may contain more attributes. Table 1 shows a part of an event log recorded by an information system in a hospital.

In Table 1, each row represents an event. A sequence of events, associated with a *case id* and ordered using the timestamps, is called a *trace*. Table 2 shows

Table 1: Sample event log (each row represents an event).

Case Id	Activity	Timestamp	Resource	Disease
1	Registration (RE)	01.01.2019-08:30:00	Employee1	Flu
1	Visit (VI)	01.01.2019-08:45:00	Doctor1	Flu
2	Registration (RE)	01.01.2019-08:46:00	Employee1	Corona
3	Registration (RE)	01.01.2019-08:50:00	Employee1	Cancer
...	...	...	...	...
1	Release (RL)	01.01.2019-08:58:00	Employee2	Flu
3	Visit (VI)	01.02.2019-10:15:00	Doctor3	Cancer
2	Release (RL)	01.02.2019-14:00:00	Employee2	Corona
3	Blood Test (BT)	01.02.2019-14:15:00	Employee5	Cancer
...	...	...	...	...

Table 2: A simple event log derived from Table 1 (each row represents a simple process instance).

Case Id	Trace	Disease
1	$\langle RE, VI, ..., RL \rangle$	Flu
2	$\langle RE, ..., RL \rangle$	Corona
3	$\langle RE, ..., VI, BT, ... \rangle$	Cancer
...	...	...

a simple trace representation of Table 1 where the *trace* attribute is a sequence of activities. Some of the event attributes may refer to individuals, e.g., the *case id* refers to the patient whose data is recorded, and the *resource* refers to the employees performing activities for the patients, e.g., surgeons. Also, some sensitive information may be included, e.g., the *disease* attribute in Table 1. When individuals' data are included in an event log, privacy issues emerge, and organizations are obliged to consider such issues according to regulations, e.g., the European General Data Protection Regulation (GDPR)[1].

The privacy/confidentiality issues in process mining are recently receiving more attention. Various techniques have been proposed covering different aspects, e.g., confidentiality frameworks [19], privacy guarantees [5,18,11], inter-organizational privacy issues [3], privacy quantification [20,16], etc. Each of these approaches considers a single event log shared at some point in time. This even log is published considering the privacy/confidentiality issues of a single log in isolation. However, event logs are recorded continuously and need to be published continuously to keep the results of process mining techniques updated.

Continuous event data publishing lets an adversary launch new types of attacks that are impossible when event data are published only once. In this paper, we analyze the so-called *correspondence attacks* [7] that an adversary can launch by comparing different releases of anonymized event logs when they are continuously published. Particularly, we focus on group-based Privacy Preservation Techniques (PPTs) and describe three main types of correspondence attacks including *forward attack*, *cross attack*, and *backward attack*. We analyze the privacy/anonymity losses imposed by these attacks and show how to detect such privacy losses efficiently. The explained anonymity analyses could be attached to different PPTs to empower them against the attacks or to change the data publishing approaches to bound such attacks. We applied different continuous event data publishing scenarios to several real-life event logs and report the anonymity indicators before and after launching the attacks for an example event log.

The remainder of the paper is organized as follows. In Section 2, we present the problem statement. In Section 3, the preliminaries are explained. Different types of correspondence attacks are analyzed in Section 4. In Section 5, we explain the attack detection techniques and privacy loss quantification. Section 6 presents the experiments. Section 7 discusses different aspects to extend the approach. Section 8 discusses related work, and Section 9 concludes the paper.

[1] http://data.europa.eu/eli/reg/2016/679/oj

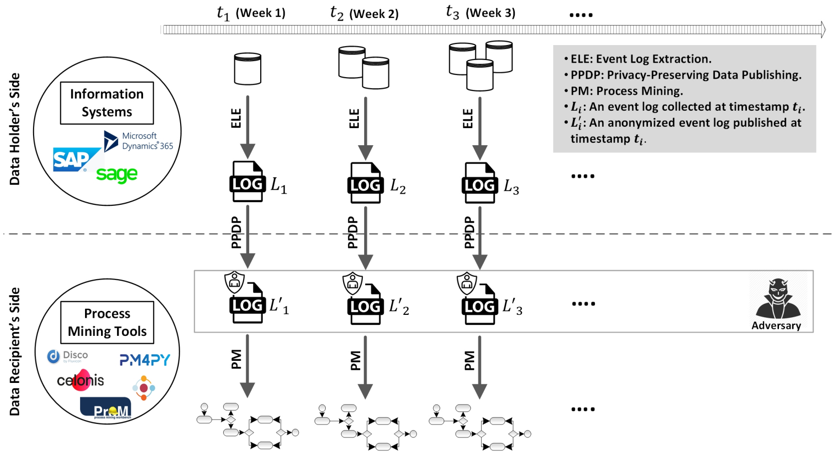


Fig. 1: The general data collection and publishing scenario.

2 Problem Statement

Figure 1 shows our general data collection and publishing scenario. Information systems, e.g., SAP, provide operational support for organizations and continuously generate a lot of valuable event data. Such data are continuously collected and published, e.g., weekly, to be used by process mining tools, e.g., ProM, Disco, etc. On the analysis side, process mining techniques are applied to event logs to discover and analyze real processes supported by operational information systems. With respect to the types of data holder's models, introduced in [9], we consider a *trusted model* where the *data holder*, i.e., the business owner, is trustworthy, but the *data recipient*, i.e., a process miner, is not trustworthy. Therefore, PPTs are applied to event logs when they are published.

Continuous data publishing is generally classified into three main categories: *incremental*, *decremental*, and *dynamic* [8]. Continuous event data publishing is considered as *incremental*, i.e., the events generated by an information system are cumulatively collected, and they are not updated or deleted after the collection. Thus, the so-called *correspondence knowledge* is gained. If we assume that in a continuous event data publishing scenario, the event logs are collected and published weekly, the correspondence knowledge is as follows: (1) Every case started in the i-th week is in the i-th event log L_i, and must be in L_j, $i<j$, and (2) Every case started in the j-th week is in the j-th event log L_j, and cannot be in L_i, $i<j$. Although each single anonymized event log L' meets the privacy guarantees specified in the corresponding PPT, the adversary, who has access to the different releases of anonymized event logs, can exploit the *correspondence knowledge* to degrade the provided privacy guarantees.

Consider Table 3 and Table 4 as two anonymized event logs, L'_1 and L'_2, published at timestamps t_1 (week 1) and t_2 (week 2), respectively. Note that the case identifiers are dummy identifiers independently assigned to the cases of

Table 3: An anonymized event log published at timestamp t_1 (e.g., week 1), meeting 2-anonymity and 2-diversity when the assumed BK is a sequence of activities with the maximum length 3.

Case Id	Trace	Disease
1	$\langle a, b, c, d\rangle$	Corona
2	$\langle a, b, c, d\rangle$	Flu
3	$\langle a, e, d\rangle$	Fever
4	$\langle a, e, d\rangle$	Corona

Table 4: An anonymized event log published at timestamp t_2 (e.g., week 2), meeting 2-anonymity and 2-diversity when the assumed BK is a sequence of activities with the maximum length 3.

Case Id	Trace	Disease
10	$\langle a, b, c, d\rangle$	Corona
20	$\langle a, b, c, d\rangle$	Flu
30	$\langle a, b, c, d\rangle$	HIV
40	$\langle a, e, d\rangle$	Fever
50	$\langle a, e, d\rangle$	Corona

each release. If we assume that an adversary's Background Knowledge (BK) is a sequence of activities with maximum length 3, both published event logs have 2-anonymity and 2-diversity. Assume the situation where the adversary knows that $\langle a, b, c\rangle$ is a subsequence of activities performed for a victim case, and that the process of the case has been started in the second week, i.e., it should be included in Table 4. Based on the correspondence knowledge, the only matching case is 30. Note that by a simple comparison of L'_1 and L'_2 based on the *disease* attribute, it is obvious that cases 10 and 20 have to be started in the first week and cannot match the adversary's BK. This is called *backward attack* (B-attack) which is a specific type of the correspondence attacks.

The provided attack scenario shows that when event logs are collected and published continuously, the corresponding PPDP approaches need to be equipped with some techniques to detect the potential attacks that can be launched by an adversary who receives various anonymized event logs. In this paper, we focus on simple event logs and group-based PPTs, i.e., k-anonymity, l-diversity, t-closeness, etc. We first describe the approach based on two releases of event logs, then we explain the possible extensions for any number of releases.

3 Preliminaries

We first introduce some basic notations. For a given set A, A^* is the set of all finite sequences over A. A finite sequence over A of length n is a mapping $\sigma{\in}\{1, ..., n\} \rightarrow A$, represented as $\sigma{=}\langle a_1, a_2, ..., a_n\rangle$ where $a_i{=}\sigma(i)$ for any $1{\leq}i{\leq}n$. $|\sigma|$ denotes the length of the sequence. For $\sigma_1, \sigma_2{\in}A^*$, $\sigma_1{\sqsubseteq}\sigma_2$ if σ_1 is a subsequence of σ_2, e.g., $\langle z, b, c, x\rangle{\sqsubseteq}\langle z, x, a, b, b, c, a, b, c, x\rangle$. For $\sigma = \langle a_1, a_2, ..., a_n\rangle$, $pref(\sigma){=}\{\langle a_1, ..., a_k\rangle \mid 1{\leq}k{\leq}n\}$, e.g., $\langle a, b, c, d\rangle \in pref(\langle a, b, c, d, e, f\rangle)$.

Definition 1 (LCS and SCS). *Let $\sigma_1 \in A^*$ and $\sigma_2 \in A^*$ be two sequences. $CSB(\sigma_1, \sigma_2){=}\{\sigma{\in}A^* \mid \sigma{\sqsubseteq}\sigma_1 \wedge \sigma{\sqsubseteq}\sigma_2\}$ is the set of common subsequences, and $LCS(\sigma_1, \sigma_2){=}\{\sigma{\in}CSB \mid \forall_{\sigma'\in CSB(\sigma_1,\sigma_2)}|\sigma'|{\leq}|\sigma|\}$ is the set of longest common subsequences. $LCS^{\sigma_1}_{\sigma_2}$ denotes the length of a longest common subsequence for σ_1 and σ_2. Also, $CSP(\sigma_1, \sigma_2){=}\{\sigma{\in}A^* \mid \sigma_1{\sqsubseteq}\sigma \wedge \sigma_2{\sqsubseteq}\sigma\}$ is the set of common super-sequences, and $SCS(\sigma_1, \sigma_2){=}\{\sigma{\in}CSB \mid \forall_{\sigma'\in CSP(\sigma_1,\sigma_2)}|\sigma'|{\geq}|\sigma|\}$ is the set of shortest common super-sequences. $SCS^{\sigma_1}_{\sigma_2}$ denotes the length of a shortest common super-sequence for σ_1 and σ_2.*

Definition 2 (Event, Event Log). *An event is a tuple $e = (c, a, t, r, d_1, ..., d_m)$, where $c{\in}\mathcal{C}$ is the case id, $a{\in}\mathcal{A}$ is the activity associated with the event, $t{\in}\mathcal{T}$*

is the event timestamp, $r \in \mathcal{R}$ is the resource, who is performing the activity, and $d_1,...,d_m$ is a list of additional attributes values, where for any $1 \leq i \leq m, d_i \in \mathcal{D}_i$. We call $\xi = \mathcal{C} \times \mathcal{A} \times \mathcal{T} \times \mathcal{R} \times \mathcal{D}_1 \times ... \times \mathcal{D}_m$ the event universe. For $e = (c, a, t, r, d_1, ..., d_m)$, $\pi_c(e)=c$, $\pi_a(e)=a$, $\pi_t(e)=t$, $\pi_r(e)=r$, and $\pi_{d_i}(e)=d_i$, $1 \leq i \leq m$, are its projections. An ***event log*** *is $L \subseteq \xi$ where events are unique.*

In continuous event data publishing, event logs are collected and published continuously at each timestamp t_i, $i \in \mathbb{N}_{\geq 1}$. L_i is the event log collected at the timestamp t_i, i.e., $L_i = \{e \in \xi \mid \pi_t(e) \leq t_i\}$. For L_i and L_j, s.t., $i<j$, L_j could contain new events for the cases already observed in L_i and new cases not observed in L_i. In the following, we define a simple version of event logs which will later be used for demonstrating the attacks and corresponding anonymity measures.

Definition 3 (Trace, Simple Trace). *A trace $\sigma = \langle e_1, e_2, ..., e_n \rangle \in \xi^*$ is a sequence of events, s.t., for each $e_i, e_j \in \sigma$: $\pi_c(e_i)=\pi_c(e_j)$, and $\pi_t(e_i) \leq \pi_t(e_j)$ if $i<j$. A simple trace is a trace where all the events are projected on the activity attribute, i.e., $\sigma \in \mathcal{A}^*$.*

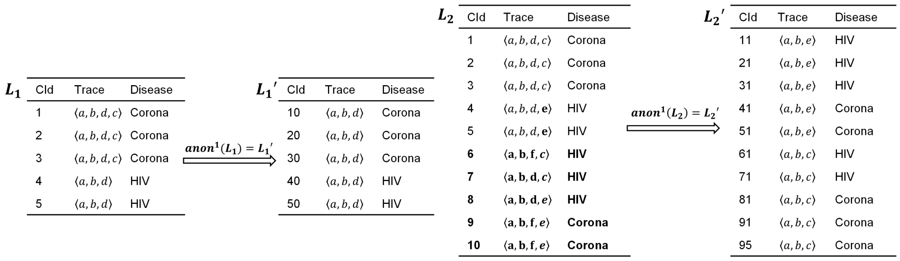
Definition 4 (Simple Process Instance). *We define $\mathcal{P} = \mathcal{C} \times \mathcal{A}^* \times \mathcal{S}$ as the universe of simple process instances, where $\mathcal{S} \subseteq \mathcal{D}_1 \cup ... \cup \mathcal{D}_m$ is the domain of the sensitive attribute. Each simple process instance $(c, \sigma, s) \in \mathcal{P}$ represents a* ***simple trace*** *$\sigma = \langle a_1, a_2, ..., a_n \rangle$, belonging to the case c with s as the sensitive attribute value. For $p=(c, \sigma, s) \in \mathcal{P}$, $\pi_c(p)=c$, $\pi_\sigma(p)=\sigma$, and $\pi_s(p)=s$ are its projections.*

Definition 5 (Simple Event Log). *Let $\mathcal{P} = \mathcal{C} \times \mathcal{A}^* \times \mathcal{S}$ be the universe of simple process instances. A simple event log is $L \subseteq \mathcal{P}$, s.t., if $(c_1, \sigma_1, s_1) \in L$, $(c_2, \sigma_2, s_2) \in L$, and $c_1=c_2$, then $\sigma_1=\sigma_2$ and $s_1=s_2$.*

4 Attack Analysis

We analyze the correspondence attacks by focusing on two anonymized releases obtained by applying group-based PPTs to simple event logs. In general, group-based PPTs provide desired privacy requirements utilizing *suppression* and/or *generalization* operations. Particularly, the group-based PPTs introduced for the event data protection are mainly based on the *suppression* operation [5,18], where some events are removed to provide the desired privacy requirements. Hence, apart from any specific privacy preservation algorithm, we define a general anonymization function that converts an event log to another one meeting desired privacy requirements assuming a bound for the maximum number of events that can be removed from each trace, so-called the *anonymization parameter*. Note that this assumption is based on the *minimality principle* in PPDP [21]. Similar attack analysis can be done for the generalization operation as well.

Definition 6 (Anonymization). *Let $\mathcal{P}$ be the universe of simple process instances and $n \in \mathbb{N}_{\geq 1}$ be the anonymization parameter. We define $anon^n \in 2^{\mathcal{P}} \rightarrow 2^{\mathcal{P}}$ as a function for anonymizing event logs. For all $L, L' \subseteq \mathcal{P}$, $anon^n(L)=L'$ if there exists a bijective function $f \in L \rightarrow L'$, s.t., for any $p=(c, \sigma, s) \in L$ and $p'=(c', \sigma', s') \in L'$ with $f(p)=p'$: $\sigma' \sqsubseteq \sigma$, $|\sigma|-n \leq |\sigma'|$, and $s'=s$.*



L_1

CId	Trace	Disease
1	$\langle a,b,d,c\rangle$	Corona
2	$\langle a,b,d,c\rangle$	Corona
3	$\langle a,b,d,c\rangle$	Corona
4	$\langle a,b,d\rangle$	HIV
5	$\langle a,b,d\rangle$	HIV

$anon^1(L_1) = L_1'$

L_1'

CId	Trace	Disease
10	$\langle a,b,d\rangle$	Corona
20	$\langle a,b,d\rangle$	Corona
30	$\langle a,b,d\rangle$	Corona
40	$\langle a,b,d\rangle$	HIV
50	$\langle a,b,d\rangle$	HIV

L_2

CId	Trace	Disease
1	$\langle a,b,d,c\rangle$	Corona
2	$\langle a,b,d,c\rangle$	Corona
3	$\langle a,b,d,c\rangle$	Corona
4	$\langle a,b,d,\mathbf{e}\rangle$	HIV
5	$\langle a,b,d,\mathbf{e}\rangle$	HIV
6	$\langle \mathbf{a},\mathbf{b},\mathbf{f},\boldsymbol{c}\rangle$	**HIV**
7	$\langle \mathbf{a},\mathbf{b},\mathbf{d},\boldsymbol{c}\rangle$	**HIV**
8	$\langle \mathbf{a},\mathbf{b},\mathbf{d},\boldsymbol{e}\rangle$	**HIV**
9	$\langle \mathbf{a},\mathbf{b},\mathbf{f},\boldsymbol{e}\rangle$	**Corona**
10	$\langle \mathbf{a},\mathbf{b},\mathbf{f},\boldsymbol{e}\rangle$	**Corona**

$anon^1(L_2) = L_2'$

L_2'

CId	Trace	Disease
11	$\langle a,b,e\rangle$	HIV
21	$\langle a,b,e\rangle$	HIV
31	$\langle a,b,e\rangle$	HIV
41	$\langle a,b,e\rangle$	Corona
51	$\langle a,b,e\rangle$	Corona
61	$\langle a,b,c\rangle$	HIV
71	$\langle a,b,c\rangle$	HIV
81	$\langle a,b,c\rangle$	Corona
91	$\langle a,b,c\rangle$	Corona
95	$\langle a,b,c\rangle$	Corona

Fig. 2: L_1 and L_2 are two simple event logs collected at timestamps t_1 and t_2. L_1' and L_2' are the corresponding anonymized releases of event logs given n=1 as the anonymization parameter. Both L_1' and L_2' have 5-anonymity and 2-diversity assuming a sequence of activities as the BK.

Note that we assume the anonymization function promises to preserve all the cases and not to produce new (fake) cases. Figure 2 shows two simple event logs that were published using the anonymization function given $n = 1$. *Specialization* is the reverse operation of the anonymization defined as follows.

Definition 7 (Specialization). *Let $\mathcal{P}$ be the universe of simple process instances and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. For $p{=}(c, \sigma, s){\in}\mathcal{P}$ and $p'{=}(c', \sigma', s'){\in}\mathcal{P}$, we say p is a specialization for p' w.r.t. n, denoted by $p'{\preceq_n}p$ iff $\sigma'{\sqsubseteq}\sigma$, $|\sigma|{\leq}|\sigma'|{+}n$, and $s = s'$.*

Consider $p'{=}(81, \langle a, b, c\rangle, Corona)$ as a process instance from the anonymized event log L_2' in Figure 2. Given n=1, the cases 1, 2, and 3 from L_2 could be a specialization for p' which are possible original process instances. We assume that the adversary's BK is a subsequence of activities performed for a victim case which can be considered as the strongest assumable knowledge w.r.t. the available information in simple event logs. Given an anonymized event log and the anonymization parameter, the adversary can distinguish a *matching set* in the anonymized release containing all the process instances having at least one specialization matching the adversary's knowledge. One of the process instances included in such a matching set belongs to the victim case.

Definition 8 (Matching Set, Group). *Let $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter and L' be an anonymized event log. $ms^{L',n}{\in}\mathcal{A}^* \rightarrow 2^{L'}$ retrieves a set of matching process instances from L'. For $bk{\in}\mathcal{A}^*$, $ms^{L',n}(bk){=}\{p'{\in}L' \mid \exists_{p\in\mathcal{P}} p'{\preceq_n}p \wedge bk{\sqsubseteq}\pi_\sigma(p)\}$. A **group** g in a matching set is a set of process instances having the same value on the sensitive attribute.*

Consider $bk{=}\langle d, e\rangle$ as the adversary's knowledge and n=1. For the anonymized event logs in Figure 2, $ms^{L_1',n}(bk){=}L_1'$, and $ms^{L_2',n}(bk){=}\{(c', \sigma', s'){\in}L_2' \mid c'{\in}\{11, 21 , 31, 41, 51\}\}$. The elements of matching sets can be identified using the following theorem without searching the space of specializations.

Theorem 1 (Elements of matching sets). *Let $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter and L' be an anonymized event log. For $bk{\in}\mathcal{A}^*$ and $p'{=}(c', \sigma', s'){\in}L'$, $p'{\in}ms^{L',n}(bk)$ iff $n \geq |bk|{-}LCS^{bk}_{\sigma'}$.*

Proof. Theorem 1 follows because one needs to add at least $|bk|-LCS^{bk}_{\sigma'}$ activities to generate a super-sequence σ of σ', s.t., $bk \sqsubseteq \sigma$. σ can be considered as the trace of a process instance p which is a specialization for p'. Note that one can always assign a value for the sensitive attribute of p', s.t., $\pi_s(p)=\pi_s(p')$.

Consider a scenario where the data holder publishes L'_1 and L'_2 as two anonymized event logs at timestamps t_1 and t_2, respectively. An adversary, who is one of the data recipients, attempts to identify a victim case vc from L'_1 or L'_2. We assume that the adversary's knowledge is a subsequence of activities performed for the vc, i.e., $bk \in \mathcal{A}^*$, and the approximate time at which the process of the vc has been started, which is enough to know the release(s) where the vc should appear. For example, if event logs are published weekly, then the adversary knows that the process of the vc has been started in the second week. Thus, its data should appear in all the event logs published after the first week. The adversary has also the *correspondence knowledge* derived from the concept of continuous event data publishing, as described in Section 2. The following correspondence attacks can be launched by the adversary.

Forward Attack (F-attack) The adversary knows that the process of the vc has been started at the approximate time t, s.t., $t{\leq}t_1$, and tries to identify the vc in L'_1 exploiting L'_2 and $bk{\in}\mathcal{A}^*$ as the BK. The vc due to its timestamp must have a process instance in L'_1 and L'_2. If there exists a $p'_1{\in}L'_1$, s.t., $p'_1{\in}ms^{L'_1,n}(bk)$ for an anonymization parameter n, there must be a $p'_2{\in}L'_2$ corresponding to p'_1. Otherwise, p'_1 does not match the BK and can be excluded from $ms^{L'_1,n}(bk)$.

Example 1 *Consider L'_1 and L'_2 in Figure 2. Assume that the adversary's knowledge is $bk{=}\langle d,e\rangle$, and the anonymization parameter is $n{=}1$. $ms^{L'_1,n}(bk){=}L'_1$ and $ms^{L'_2,n}(bk){=}\{(c',\sigma',s'){\in}L'_2 \mid c'{\in}\{11,21,31,41,51\}\}$. Both matching sets meet 5-anonymity. However, by comparing L'_1 and L'_2, the adversary learns that one of the cases 10, 20, 30 cannot have e after d. Otherwise, there must have been three cases with Corona in $ms^{L'_2,n}(bk)$. Therefore, the adversary can exclude one of 10, 20, 30. Note that the choice among 10, 20, 30 does not matter as they are equal. Consequently, k is degraded from 5 to 4.*

Cross Attack (C-attack) The adversary knows that the process of the vc has been started at the approximate time t, s.t., $t{\leq}t_1$, and attempts to identify the vc in L'_2 exploiting L'_1 and $bk{\in}\mathcal{A}^*$ as the BK. The vc because of its timestamp must have a process instance in L'_1 and L'_2. If there exists a $p'_2{\in}L'_2$, s.t., $p'_2{\in}ms^{L'_2,n}(bk)$ for an anonymization parameter n, there must be a $p'_1{\in}L'_1$ corresponding to p'_2. Otherwise, p'_2 either is started at timestamp t, $t_1{<}t{\leq}t_2$, or it does not match the BK and can be excluded from $ms^{L'_2,n}(bk)$.

Example 2 *Consider L'_1 and L'_2 in Figure 2. Assume that the adversary's knowledge is $bk{=}\langle d,e\rangle$, and the anonymization parameter is $n{=}1$. $ms^{L'_1,n}(bk) = L'_1$ and $ms^{L'_2,n}(bk){=}\{(c',\sigma',s'){\in}L'_2 \mid c'{\in}\{11,21,31,41,51\}\}$. Both matching sets meet 5-anonymity. However, by comparing L'_1 and L'_2, the adversary learns that one of the cases 11, 21, 31 must be started at timestamp t, s.t., $t_1{<}t{\leq}t_2$. Otherwise, there must have been three cases with HIV in $ms^{L'_1,n}(bk)$. Therefore, the* *adversary can exclude one of* $11, 21, 31$. *Again, the choice among* $11, 21, 31$ *does not matter as they are equal. Consequently, k is degraded from* 5 *to* 4.

Backward Attack (*B*-attack) The adversary knows that the process of the vc has been started at the approximate time t, s.t., $t_1{<}t{\leq}t_2$, and tries to identify the vc in L'_2 exploiting L'_1 and $bk{\in}\mathcal{A}^*$ as the BK. The vc has a process instance in L'_2, but not in L'_1. Hence, if there exists $p'_2 \in L'_2$, s.t., $p'_2 \in ms^{L'_2,n}(bk)$ for an anonymization parameter n, and p'_2 has to be a corresponding process instance for some process instances in L'_1, then p'_2 must be started at timestamp t, s.t., $t{\leq}t_1$ and can be excluded from the matching set $ms^{L'_2,n}(bk)$.

Example 3 *Consider* L'_1 *and* L'_2 *in Figure* 2. *Assume that the adversary's knowledge is* $bk{=}\langle d, c\rangle$, *and the anonymization parameter is* $n{=}1$. $ms^{L'_1,n}(bk){=}L'_1$ *and* $ms^{L'_2,n}(bk){=}\{(c', \sigma', s'){\in}L'_2 \mid c'{\in}\{61, 71, 81, 91, 95\}\}$. *Both matching sets meet* 5-*anonymity. However, by comparing* L'_1 *and* L'_2, *the adversary learns that at least one of the cases* $81, 91, 95$ *must be started at timestamp t,* $t{\leq}t_1$. *Otherwise, one of the cases* $10, 20, 30$ *cannot have a corresponding process instance in* L'_2. *Thus, k is degraded from* 5 *to* 4. *Note that there are only two cases with Corona which are not in* $ms^{L'_2,n}(bk)$ *and could be corresponding for cases* $10, 20, 30$. *Hence, at least one of* $81, 91, 95$ *must be started at timestamp t,* $t{\leq}t_1$.

5 Attack Detection

The correspondence attacks mentioned in Section 4 are based on making some inferences about corresponding cases (process instances). However, there are many possible assignments of corresponding cases and each of those implies possibly different event logs, which are not necessarily the actual event logs collected by the data holder. In this section, we demonstrate the *attack detection* regardless of any particular choices. To this end, we first need to define a *linker* to specify all the valid assignments. Then, we provide formal definitions for different types of correspondence attacks and corresponding anonymity indicators.

Definition 9 (Linker, Buddy). *Let* L'_1 *and* L'_2 *be the anonymized event logs at timestamps* t_1 *and* t_2, *respectively, and* $n{\in}\mathbb{N}_{\geq 1}$ *be the anonymization parameter.* $linker^n{\in}L'_1{\to}L'_2$ *is a total injective function that retrieves the corresponding process instances. For* $p'_1{\in}L'_1$ *and* $p'_2{\in}L'_2$, $linker^n(p'_1){=}p'_2$ *iff there exist* $p_1, p_2{\in}\mathcal{P}$, *s.t.,* $p'_1{\preceq_n}p_1 \wedge p'_2{\preceq_n}p_2 \wedge \pi_s(p_1){=}\pi_s(p_2) \wedge \pi_\sigma(p_1) \in pref(\pi_\sigma(p_2))$. (p'_1, p'_2) *is called a pair of* ***buddies*** *if there exists a linker, s.t.,* $linker^n(p'_1){=}p'_2$.

Definition 10 (*F*-attack). *Let* L'_1 *and* L'_2 *be two anonymized event logs released at timestamps* t_1 *and* t_2, $n{\in}\mathbb{N}_{\geq 1}$ *be the anonymization parameter,* $t{\leq}t_1$ *be the approximate time at which the process of the victim case has been started, and* $bk{\in}\mathcal{A}^*$ *be the BK. The F-attack attempts to identify x as the maximal excludable cases from* $ms^{L'_1,n}(bk)$, *s.t., for any linker, at least x cases from the matching set cannot match the BK. x is considered as crack size based on F-attack.*

Definition 11 (C-attack). *Let L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2, $n \in \mathbb{N}_{\geq 1}$ be the anonymization parameter, $t \leq t_1$ be the approximate time at which the process of the victim case has been started, and $bk \in \mathcal{A}^*$ be the BK. The C-attack tries to identify x (crack size) as the maximal excludable cases from $ms^{L'_2,n}(bk)$, s.t., for any linker, at least x cases from the matching set cannot match the BK or the timestamp of the victim case.*

Definition 12 (B-attack). *Let L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2, $n \in \mathbb{N}_{\geq 1}$ be the anonymization parameter, $t_1 < t \leq t_2$ be the approximate time at which the process of the victim case has been started, and $bk \in \mathcal{A}^*$ be the BK. The B-attack tries to identify x (crack size) as the maximal excludable cases from $ms^{L'_2,n}(bk)$, s.t., for any linker, at least x cases from the matching set cannot match the timestamp of the victim case.*

Based on the definitions for the correspondence attacks, the key for attack detection is the crack size. For calculating the crack sizes, we follow the similar approach introduced in [7] which is based on the concept of *comparability*. We define the comparability at the level of *sequences*, *process instances*, and *groups*. These definitions are later used to compute the crack sizes of attacks.

Definition 13 (Comparable Sequences). *Let $\sigma_1, \sigma_2 \in \mathcal{A}^*$ be two sequences of activities. We say σ_1 and σ_2 are comparable w.r.t. $n \in \mathbb{N}_{\geq 1}$, denoted by $\sigma_1 \overset{n}{\sim} \sigma_2$, if n is the minimum number of activities that needs to be added to σ_1 and/or σ_2 to generate a joint super-sequence, or if σ_1 can be a prefix of σ_2 by adding at least n activities to σ_2.*

Theorem 2 (Detecting comparable sequence). *Given $\sigma_1, \sigma_2 \in \mathcal{A}^*$ and $n \in \mathbb{N}_{\geq 1}$:*

$$\sigma_1 \overset{n}{\sim} \sigma_2 \iff \begin{cases} n \geq |\sigma_1| - LCS^{\sigma_1}_{\sigma_2} & \textit{if } \exists_{\sigma \in LCS(\sigma_1,\sigma_2)} \sigma \in pref(\sigma_2) \\ n \geq SCS^{\sigma_1}_{\sigma_2} - min(|\sigma_1|, |\sigma_2|) & \textit{otherwise} \end{cases}$$

Proof. If there exists a $\sigma \in LCS(\sigma_1, \sigma_2)$, s.t., $\sigma \in pref(\sigma_2)$, then $|\sigma_1| - LCS^{\sigma_1}_{\sigma_2}$ is the minimum number of activities that needs to be added to σ_2, s.t., $\sigma_1 \in pref(\sigma_2)$. Otherwise, since $SCS^{\sigma_1}_{\sigma_2}$ is the length of a shortest common super-sequence, one needs to add at least $SCS^{\sigma_1}_{\sigma_2} - min(|\sigma_1|, |\sigma_2|)$ activities to the shorter sequence to generate a joint super-sequence.

Definition 14 (Comparable Process Instances). *Let $p_1, p_2 \in \mathcal{P}$ be two process instances. We say p_1 and p_2 are comparable w.r.t. n, denoted by $p_1 \overset{n}{\sim} p_2$, iff $\pi_s(p_1) = \pi_s(p_2) \wedge \pi_\sigma(p_1) \overset{n}{\sim} \pi_\sigma(p_2)$.*

Definition 15 (Comparable Groups). *Let L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2, $bk \in \mathcal{A}^*$ be the BK, and $n \in \mathbb{N}_{\geq 1}$ be the anonymization parameter. We say two groups $g'_1 \subseteq ms^{L'_1,n}(bk)$ and $g'_2 \subseteq ms^{L'_2,n}(bk)$ are comparable w.r.t. n, denoted by $g'_1 \overset{n}{\sim} g'_2$, iff $\forall_{p'_1 \in g'_1} \forall_{p'_2 \in g'_2} p'_1 \overset{n}{\sim} p'_2$.*

Lemma 1. *Let L'_1 and L'_2 be two anonymized event logs at timestamps t_1 and t_2, $bk \in \mathcal{A}^*$ be the BK, and $n \in \mathbb{N}_{\geq 1}$ be the anonymization parameter. Consider $g'_1 \subseteq ms^{L'_1,n}(bk)$ and $g'_2 \subseteq ms^{L'_2,n}(bk)$ as two groups, s.t., $g'_1 \overset{n}{\sim} g'_2$. If $p'_1 \in ms^{L'_1,n}(bk)$ and $p'_2 \in ms^{L'_2,n}(bk)$ are buddies for a linker, then $p'_1 \in g'_1$ iff $p'_2 \in g'_2$.*

Lemma 2. *Let L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2, $bk \in \mathcal{A}^*$ be the BK, and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. Consider $g'_1{\subseteq}ms^{L'_1,n}(bk)$ and $g'_2{\subseteq}ms^{L'_2,n}(bk)$ as two groups, s.t., $g'_1\overset{n}{\sim}g'_2$. Since the buddy relationship is injective, at most $min(|g'_1|,|g'_2|)$ process instances in g'_1 have a buddy in g'_2, and there are some linkers where exactly $min(|g'_1|,|g'_2|)$ process instances in g'_1 have a buddy in g'_2.*

Theorem 3 (Crack size based on F-attack). *Let $bk{\in}\mathcal{A}^*$ be the BK, $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter, and L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2. Let $CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk)) = \{(g'_1, g'_2) \mid g'_1{\subseteq}ms^{L'_1,n}(bk) \wedge g'_2{\subseteq}ms^{L'_2,n}(bk) \wedge g'_1\overset{n}{\sim}g'_2\}$ be the set of pair of comparable groups in the matching sets. For $(g'_1, g'_2) \in CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$, g'_1 has crack size $cs = |g'_1|{-}min(|g'_1|,|g'_2|)$. $F(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk)) = \sum cs$ is the number of excludable cases from $ms^{L'_1,n}(bk)$ exploiting the F-attack, where $\sum$ is over $(g'_1, g'_2){\in}CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$.*

Proof. Consider $(g'_1, g'_2){\in}CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$. Based on Lemma 2, if $|g'_1|{>}\ |g'_2|$, at least $|g'_1|{-}min(|g'_1|,|g'_2|)$ process instances in g'_1 do not have a buddy in g'_2 for any linker. Also, according to Lemma 1, these process instances cannot match the given BK. Otherwise, they must have had buddies in g'_2.

Example 4 *Consider L'_1 and L'_2 in Figure 2, $n{=}1$, and $bk{=}\langle d, e\rangle$. $|g'_1|{=}3$ and $|g'_2|{=}2$ for the Corona groups in $ms^{L'_1,n}(bk)$ and $ms^{L'_2,n}(bk)$, respectively. $cs{=}3{-}min(3,2)$ is the crack size of $ms^{L'_1,n}(bk)$ based on F-attack.*

Definition 16 (F-Anonymity). *Let L'_1 and L'_2 be two anonymized event logs at t_1 and t_2, and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. The F-anonymity of L'_1 and L'_2 is $FA^n(L'_1{,}L'_2) = \min\limits_{bk\in\mathcal{A}^*} |ms^{L'_1,n}(bk)|{-}F(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$.*

Theorem 4 (Crack size based on C-attack). *Let $bk{\in}\mathcal{A}^*$ be the BK, $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter, and L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2. Let $CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk)) = \{(g'_1, g'_2) \mid g'_1{\subseteq}ms^{L'_1,n}(bk) \wedge g'_2{\subseteq}ms^{L'_2,n}(bk) \wedge g'_1\overset{n}{\sim}g'_2\}$ be the set of pair of comparable groups in the matching sets. For $(g'_1, g'_2){\in}CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$, g'_2 has crack size $cs = |g'_2|{-}min(|g'_1|,|g'_2|)$. $C(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk)) = \sum cs$ is the number of excludable cases from $ms^{L'_1,n}(bk)$ exploiting the C-attack, where $\sum$ is over $(g'_1, g'_2){\in}CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$.*

Proof. Consider $(g'_1, g'_2){\in}CG(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$. Based on Lemma 2, if $|g'_2|{>}\ |g'_1|$, at least $|g'_2|{-}min(|g'_1|,|g'_2|)$ process instances in g'_2 do not have a buddy in g'_1 for any linker. Such process instances either cannot match the given BK, according to Lemma 1, or they have been started at timestamp t, $t_1{<}t{\leq}t_2$.

Example 5 *Consider L'_1 and L'_2 in Figure 2, $n{=}1$, and $bk{=}\langle d, e\rangle$. $|g'_1|{=}2$ and $|g'_2|{=}3$ for the HIV groups in $ms^{L'_1,n}(bk)$ and $ms^{L'_2,n}(bk)$, respectively. $cs{=}3 - min(2,3)$ is the crack size of $ms^{L'_2,n}(bk)$ based on C-attack.*

Definition 17 (C-Anonymity). *Let L'_1 and L'_2 be two anonymized event logs at t_1 and t_2, and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. The C-anonymity of L'_1 and L'_2 is $CA^n(L'_1{,}L'_2) = \min\limits_{bk\in\mathcal{A}^*} |ms^{L'_2,n}(bk)|{-}C(ms^{L'_1,n}(bk), ms^{L'_2,n}(bk))$.*

Lemma 3. *Let L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2, $bk \in \mathcal{A}^*$ be the BK, and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. Consider $g'_2{\subseteq}ms^{L'_2,n}(bk)$, $G'_1{=}\{p'_1{\in}L'_1 \mid \exists_{p'_2\in g'_2} p'_1 \overset{n}{\sim} p'_2\}$, and $G'_2{=}\{p'_2{\in}L'_2 \mid \exists_{p'_1\in G'_1} p'_1 \overset{n}{\sim} p'_2\}$. Every process instance in G'_2 is comparable to all records in G'_1 and only those records in G'_1.*

Theorem 5 (Crack size based on B-attack). *Let $bk{\in}\mathcal{A}^*$ be the BK, $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter, and L'_1 and L'_2 be two anonymized event logs released at timestamps t_1 and t_2. Let $g'_2{\subseteq}ms^{L'_2,n}(bk)$, $G'_1{=}\{p'_1{\in}L'_1 \mid \exists_{p'_2\in g'_2} p'_1 \overset{n}{\sim} p'_2\}$, and $G'_2{=}\{p'_2{\in}L'_2 \mid \exists_{p'_1\in G'_1} p'_1 \overset{n}{\sim} p'_2\}$. g'_2 has crack size $cs = max(0, |G'_1|{-}(|G'_2|{-}|g'_2|))$. $B(ms^{L'_2,n}(bk), L'_1, L'_2) = \sum_{g'_2\in ms^{L'_2,n}(bk)} cs$ is the number of excludable cases from $ms^{L'_2,n}(bk)$ exploiting B-attack.*

Proof. According to Lemma 3, all process instances in G'_1 and only those process instances can have a buddy in G'_2. Therefore, each process instance in G'_1 has a buddy either in g'_2 or $G'_2 - g'_2$. If $|G'_1|{>}\ |G'_2|{-}|g'_2|$, then $|G'_1|{-}(|G'_2|{-}|g'_2|)$ process instances in g'_2 must be started at timestamp t, $t{\leq}t_1$.

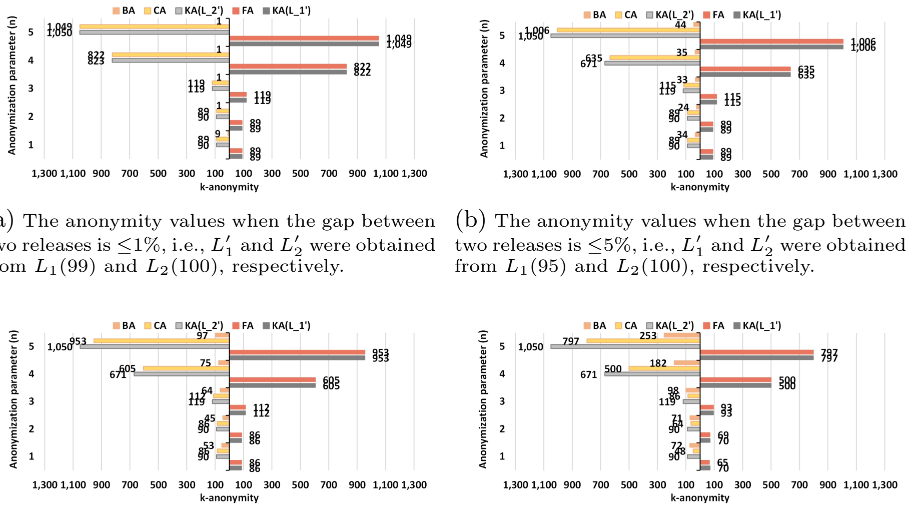
Example 6 *Consider L'_1 and L'_2 in Figure 2, $n{=}1$, and $bk{=}\langle d, c\rangle$. $|g'_2|{=}3$ for the Corona group in $ms^{L'_2,n}(bk)$, $G'_1{=}\{p'_1{\in}L'_1 \mid \pi_c(p'_1){\in}\{10, 20, 30\}\}$, and $G'_2 = \{p'_2{\in}L'_2 \mid \pi_c(p'_2){\in}\{41, 51, 81, 91, 95\}\}$. $cs{=}max(0, 3-(5-3))$ is the crack size of $ms^{L'_2,n}(bk)$ based on B-attack.*

Definition 18 (B-Anonymity). *Let L'_1 and L'_2 be two anonymized event logs at t_1 and t_2, and $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter. $BA^n(L'_1, L'_2) = \min\limits_{bk\in\mathcal{A}^*} |ms^{L'_2,n}(bk)|{-}B(ms^{L'_2,n}(bk), L'_1, L'_2)$ is the B-anonymity of L'_1 and L'_2.*

Given $n{\in}\mathbb{N}_{\geq 1}$ as the anonymization parameter, $KA^n(L'){=} \min\limits_{bk\in\mathcal{A}^*} |ms^{L',n}(bk)|$ is the k-anonymity of an anonymized event log L' w.r.t. n. Assuming L'_1 and L'_2 as two anonymized event logs at timestamps t_1 and t_2, we calculate *the proportion of the cracked cases* (PoCs) after launching the correspondence attacks as follows: $FC^n(L'_1, L'_2){=}\frac{(KA^n(L'_1){-}FA^n(L'_1,L'_2))}{KA^n(L'_1)}$, $CC^n(L'_1, L'_2){=}\frac{(KA^n(L'_2){-}CA^n(L'_1,L'_2))}{KA^n(L'_2)}$, and $BC^n(L'_1, L'_2){=}\frac{(KA^n(L'_2){-}BA^n(L'_1,L'_2))}{KA^n(L'_2)}$.

6 Experiments

In this section, we employ *Sepsis* [10] as a real-life event log and simulate different continuous event data publishing scenarios. We report privacy losses and anonymity values based on the correspondence attacks. Note that *Sepsis* is one of the most challenging



(a) The anonymity values when the gap between two releases is ≤1%, i.e., L'_1 and L'_2 were obtained from $L_1(99)$ and $L_2(100)$, respectively.

(b) The anonymity values when the gap between two releases is ≤5%, i.e., L'_1 and L'_2 were obtained from $L_1(95)$ and $L_2(100)$, respectively.

(c) The anonymity values when the gap between two releases is ≤10%, i.e., L'_1 and L'_2 were obtained from $L_1(90)$ and $L_2(100)$, respectively.

(d) The anonymity values when the gap between two releases is ≤25%, i.e., L'_1 and L'_2 were obtained from $L_1(75)$ and $L_2(100)$, respectively.

Fig. 3: The anonymity values for different variants of pairs of anonymized releases in Scenario I. $KA(L'_1)$ is k-anonymity of L'_1, $KA(L'_2)$ is k-anonymity of L'_2, FA is k-anonymity of L'_1 after launching F-attack, CA is k-anonymity of L'_2 after launching C-attack, and BA is k-anonymity of L'_2 after launching B-attack.

event logs for PPTs [5,18,11]. We consider two main scenarios to cover various situations w.r.t. *event data volume* and *velocity of event data publishing*. In both scenarios, we consider two releases to be published.

In **Scenario I**, we consider the entire event log as the second collection of events $L_2(100)$. Keeping the second collection of events as $L_2(100)$, we generate four different variants for the first collection of events named $L_1(99)$, $L_1(95)$, $L_1(90)$, and $L_1(75)$, s.t., $L_1(x)$ contains x% of cases. Note that we ignore the decimal points for percentages, e.g., 90% could be 90.01% or 90.95%. In **Scenario II**, we filter 50% of cases as the first collection of events $L_1(50)$. Keeping the first collection of events as $L_1(50)$, we generate four different variants for the second collection of events named $L_2(51)$, $L_2(55)$, $L_2(60)$, and $L_2(75)$, s.t., $L_2(x)$ contains x% of cases. To filter the event logs, we use *time-frame filtering* where the start time is always the start time of the event log and the end time is changed to pick the desired percentage of cases.

In both scenarios, the gap between two collections varies, s.t., it contains at most 1%, 5%, 10%, or 25% new cases. We focus on the percentage of cases rather than a fixed time window, e.g., daily, weekly, etc., because a fixed time window could contain different amount of data in different slots. We employ the extended version of TLKC-privacy model [17] as the group-based PPT where one can adjust power and type of BK.[2] The model removes events from traces w.r.t. *utility loss* and *privacy gain* to provide the desired privacy requirements. We consider all the possible sequences of activities in the event log with the maximal length 5 as the candidates of BK, and

[2] https://github.com/m4jidRafiei/TLKC-Privacy-Ext

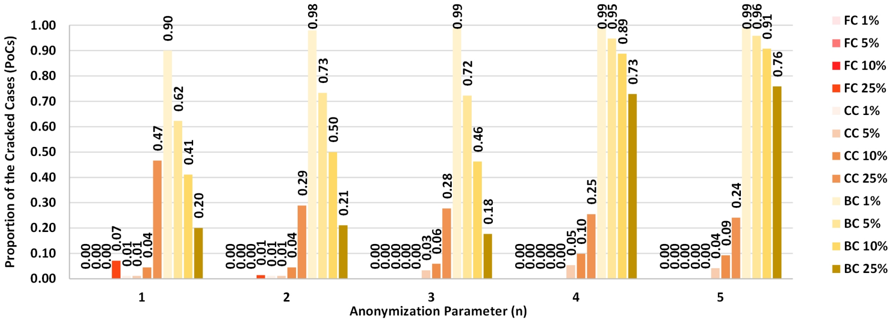


Fig. 4: Let x be the maximal gap between two anonymized releases. FC x%, CC x%, and BC x% show the PoCs exploiting F-attack, C-attack, and B-attack, respectively. For each anonymization parameter, the first, the second, and the third 4 bars show the results for F-attack, C-attack, and B-attack, respectively.

k=20 as the lower bound for k-anonymity, i.e., the privacy model guarantees that a single release of the event log meets at least 20-anonymity for all the candidates of BK. On the data recipient's side, in each scenario, four different pairs of anonymized releases are received. We developed a Python program to detect the attacks and report the anonymity values. The source code and other resources are available on GitHub.[3]

Figure 3 shows the anonymity values before and after launching the attacks in Scenario I. Note that when n is equal to the length of the BK, all cases already fall into the matching sets. Therefore, the maximal value for the anonymization parameter is 5 which is the maximal length assumed for the BK. Figure 3a shows that when the gap is at most 1% and n=1, the anonymized release L'_2 has 90-anonymity. However, after launching the B-attack, 81 cases are cracked, i.e., 90% of cases, and k-anonymity is degraded to 9, i.e., $BA^1(L'_1, L'_2)$=9. For n>1, the B-anonymity is 1, i.e., there exists a sequence of activities of the maximal length 5 that can be used to uniquely identify a case assuming that at most n>1 activities have been removed by the PPT. Note that the second release includes only 1 new case when the gap is at most 1%.

Figure 4 shows how the PoCs are changed when we vary the anonymization parameter n in Scenario I. Each pair of the anonymized releases is indicated with the percentage of the gap, e.g., 1% in Scenario I indicates two releases obtained from $L_1(99)$ and $L_2(100)$. When the gap between two releases is small, the B-attack results in much higher values for the PoCs compared to the other attacks. However, when the gap becomes larger, the PoCs of the B-attack decreases. This happens because for the smaller L'_1s, there exist fewer cases that can be excluded from the matching sets in L'_2 because of their timestamps. The C-attack shows different behavior that is due to the assumed timestamp for the victim case, i.e., for the larger gaps, there exist more cases that their timestamps comply with the second release L'_2 and cannot have a corresponding case in L'_1. The F-attack cracks fewer cases, which is expected because its target release is L'_1, and it only exploits the BK mismatching. Note that greater values for the anonymization parameter mean that the adversary assumes higher data distortion which results in greater values for the anonymity. We had similar observations for Scenario II, and the results are available in our GitHub repository.

[3] https://github.com/m4jidRafiei/PP_CEDP

7 Extensions

The two releases scenario can be extended to the general scenario where more releases are involved. In the general scenario, we consider $m{\in}\mathbb{N}_{>2}$ collections of events $L_1, L_2, \ldots, L_m$ collected at timestamps $t_1, t_2, \ldots, t_m$ and published as $L'_1, L'_2, \ldots, L'_m$. The correspondence knowledge is also extended, s.t., every case in L'_i has a corresponding case in L'_j, $i{<}j{\leq}m$. Consider the introduced attacks based on two releases as *micro attacks*. Given more than two releases, the adversary can launch two other types of attacks, so-called *optimal micro attacks* and *composition of micro attacks* [7].

Optimal micro attacks: The idea is to find the best background release which results in the largest possible crack size. For instance, consider the F-attack on L'_i. The adversary can choose any L'_j, $i{<}j{\leq}m$, as the background release. Let $bk \in \mathcal{A}^*$ be the background knowledge, $n{\in}\mathbb{N}_{\geq 1}$ be the anonymization parameter, and cs_{ij} be the crack size of a pair of comparable groups $(g'_i, g'_j) \in CG(ms^{L'_i,n}(bk), ms^{L'_j,n}(bk))$. The optimal crack size of g'_i is $\max\limits_{i<j\leq m} cs_{ij}$.

Composition of micro attacks: The idea is to compose multiple micro attacks to increase the crack size of a group. The micro attacks are launched one after the other. Note that the composition is not possible for any arbitrary choice of micro attacks. It is possible only if all the micro attacks in the composition assume the same timestamp for the victim case, and the required correspondence knowledge holds for the next attack after the previous attack [7]. Hence, considering L'_i, L'_j, and L'_l, as the anonymized releases, s.t., $i{<}j{<}l{\leq}m$, only two compositions are possible: (1) B-attack on L'_i and L'_j followed by F-attack on L'_j and L'_l, and (2) B-attack on L'_i and L'_j followed by C-attack on L'_j and L'_l.

Here, we focused on k-anonymity which is the foundation for the group-based PPTs. The proposed approach can be extended to cover all the extensions of k-anonymity introduced to deal with *attribute linkage* attacks, e.g., l-diversity, (α, k)-anonymity, confidence bounding, etc. The measures of such PPTs can be modified to consider the cracked cases. Moreover, new group-based PPTs for process mining can be designed to consider $F/C/B$-anonymity. For example, a naive algorithm is to start with the maximal possible anonymity, i.e., having only one trace variant, e.g., the longest common subsequence, and then adding events w.r.t. their effect on data utility and privacy loss.

8 Related Work

Privacy/confidentiality in process mining is growing in importance. The work having been done covers different aspects of the topic including *the challenges* [2,4,13], *confidentiality frameworks* [19], *privacy by design* [12], *privacy guarantees* [11,6,18,5], *inter-organizational privacy issues* [3], and *privacy quantification* [20,16]. Confidentiality is one of the important challenges of the bigger sub-discipline of process mining called *Responsible Process Mining* (RPM) [2]. In [13], the authors focus on data privacy and utility requirements for healthcare event data. A general framework for confidentiality in process mining is proposed in [19]. In [12], the goal is to propose a privacy-preserving system design for process mining. In [14], the authors introduce a privacy-preserving method for discovering roles from event logs. In [5], k-anonymity and t-closeness are adopted to preserve the privacy of *resources* in event logs. In [11,6], the notion of *differential privacy* is utilized to provide privacy guarantees. In [18], the TLKC-privacy is introduced to deal with high variability issues in event logs for applying group-based anonymization techniques. A secure multi-party computation solution is proposed in [3] for preserving privacy in an inter-organizational setting. In [20], the authors propose a measure to evaluate the re-identification risk of event logs. Also, in [16], a general privacy quantification framework, and some measures are introduced to evaluate the effectiveness of PPTs. In [15], the authors propose a privacy extension for the XES standard to manage privacy metadata.

9 Conclusion

In practice, event data need to be published continuously to keep the process mining results up-to-date. In this paper, for the first time, we focused on the attacks appearing when anonymized event data are published continuously. We formalized three different types of the so-called *correspondence attacks* in the context of process mining: F-attack, C-attack, and B-attack. We demonstrated the attack detection techniques to quantify the anonymity of event logs published continuously. We simulated the continuous event data publishing for real-life event logs using various scenarios. For an example event log, we showed that the provided privacy guarantees can be degraded exploiting the attacks. The attack analysis and detection techniques can be adjusted and attached to different group-based PPTs to enhance the privacy guarantees when event data are published continuously. In this paper, we mainly focused on *suppression* as the anonymization operation. In future, other anonymization operations such as *addition* or *swapping* could be analyzed. Similar attack analysis can be done for other types of PPTs, e.g., *differential privacy*, in the context of process mining to protect provided privacy guarantees. Moreover, one could evaluate the effect of continuous publishing scenarios on privatized process mining results.

Acknowledgment

Funded under the Excellence Strategy of the Federal Government and the Länder. We also thank the Alexander von Humboldt 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. Elkoumy, G., Fahrenkrog-Petersen, S.A., Dumas, M., Laud, P., Pankova, A., Weidlich, M.: Secure multi-party computation for inter-organizational process mining. In: Enterprise, Business-Process and Information Systems Modeling - 21st International Conference, BPMDS. Springer (2020)
4. Elkoumy, G., Fahrenkrog-Petersen, S.A., Sani, M.F., Koschmider, A., Mannhardt, F., von Voigt, S.N., Rafiei, M., von Waldthausen, L.: Privacy and confidentiality in process mining - threats and research challenges. CoRR **abs/2106.00388** (2021), https://arxiv.org/abs/2106.00388

5. 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 (2019)
6. Fahrenkrog-Petersen, S.A., van der Aa, H., Weidlich, M.: PRIPEL: privacy-preserving event log publishing including contextual information. In: Business Process Management - 18th International Conference, BPM. Lecture Notes in Computer Science, vol. 12168, pp. 111–128 (2020)
7. Fung, B.C.M., Wang, K., Fu, A.W., Pei, J.: Anonymity for continuous data publishing. In: 11th International Conference on Extending Database Technology. ACM International Conference Proceeding Series, vol. 261, pp. 264–275 (2008)
8. 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)
9. Gehrke, J.: Models and methods for privacy-preserving data analysis and publishing. In: Proceedings of the 22nd International Conference on Data Engineering, ICDE. p. 105. IEEE Computer Society (2006)
10. Mannhardt, F.: Sepsis cases-event log. Eindhoven University of Technology (2016)
11. 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)
12. 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)
13. 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)
14. 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)
15. Rafiei, M., van der Aalst, W.M.P.: Privacy-preserving data publishing in process mining. In: Business Process Management Forum - BPM Forum 2020. pp. 122–138. Springer (2020). https://doi.org/10.1007/978-3-030-58638-6_8
16. Rafiei, M., van der Aalst, W.M.P.: Towards quantifying privacy in process mining. In: International Conference on Process Mining - ICPM 2020 International Workshops, Padua, Italy, October 4-9, 2020 (2020)
17. Rafiei, M., van der Aalst, W.M.P.: Group-based privacy preservation techniques for process mining. Data & Knowledge Engineering **134**, 101908 (2021). https://doi.org/https://doi.org/10.1016/j.datak.2021.101908
18. Rafiei, M., Wagner, M., van der Aalst, W.M.P.: TLKC-privacy model for process mining. In: Research Challenges in Information Science - 14th International Conference, RCIS. pp. 398–416. Springer International Publishing (2020)
19. Rafiei, M., von Waldthausen, L., van der Aalst, W.M.P.: Supporting condentiality 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)
20. von Voigt, S.N., Fahrenkrog-Petersen, S.A., Janssen, D., Koschmider, A., Tschorsch, F., Mannhardt, F., Landsiedel, O., Weidlich, M.: Quantifying the re-identification risk of event logs for process mining - empiricial evaluation paper. In: Advanced Information Systems Engineering, CAiSE (2020)
21. Wong, R.C.W., Fu, A.W.C., Wang, K., Pei, J.: Minimality attack in privacy preserving data publishing. In: Proceedings of the 33rd international conference on Very large data bases. pp. 543–554 (2007)