International Journal of Combinatorial Optimization Problems and Informatics, 15(5), Dec 2024, 202-211. ISSN: 2007-1558.
https://doi.org/10.61467/2007.1558.2024.v15i5.579

www.editada.org

An Assessment of Encryption-based Approaches for Business Process Models Discovery

Héctor Alán De la Fuente Anaya[1,3], *Heidy Marisol Marin Castro*[2*], *Miguel Morales Sandoval*[1*] *and José Juan García Hernández*[3]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, 72840, Puebla.
[2] Universidad de las Américas Puebla, Departamento de Computación, Electrónica y Mecatrónica, Cholula, 72810, Puebla.
[3] Cinvestav Unidad Tamaulipas, Parque Científico y Tecnológico de Tamaulipas, Victoria, 87138, Tamaulipas.

hector.delafuente@inaoep.mx, heidy.marin@udlap.mx, mmorales@inaoep.mx, jjuan.garcia@cinvestav.mx
*Corresponding author(s).

Abstract. Process Mining as a Service (PMaaS) could enable organizations to delegate the analysis of their business processes to third parties in the cloud and then avoid costs related to storage and processing. However, this involves outsourcing event log data to the service provider, which can be honest but curious and learn from the data in clear. This potentially compromises the confidentiality and privacy of the event log data that can be sensitive, for example, in the healthcare domain. Thus, it is necessary to both safeguard the privacy of the event log data and guarantee their usefulness in the process mining tasks. This paper presents a study and experimental evaluation of two cryptographic-based methods suitable for PMaaS with security and privacy guaranties, tested under the process discovery task and evaluated through a set of experiments in terms of utility loss, performance and computational cost, using available event logs in the healthcare domain.

Keywords: Process Mining as a Service, Process Discovery, Event Log, Encryption, Privacy-preserving.

Article Info
Received May 10, 2024.
Accepted Nov 20, 2024.

1 Introduction

A business process refers to a set of interrelated activities that are executed in a consistent and coordinated manner within an organization to achieve a business objective. An example of a business process is the purchase of a product, which is initiated when a customer places an order. This is followed by shipment of the product, delivery, and invoicing. A business process can be executed multiple times, where each execution corresponds to an individual instance of the process, also known as a case or trace. An event log is the collection of events recorded during the execution of a business process. Some of the typical attributes of an event include: the identifier of the case in which the event occurs ("*Case ID*"), the timestamp in which the activity was performed ("*Timestamp*") associated with the event, the label of the activity performed ("*Activity*") and the entity responsible for executing the activity ("*Resource*"), as shown in Table 1.

Process Mining (PM) [15] is an area of data science that allows extracting knowledge from the event log to gain a better understanding of the business process, in order to improve the quality of the business process. One of the tasks in PM is process model discovery: based on the event log data, the business process model is automatically discovered. This discovered model reflects the actual behavior of the business process and not

© Editorial Académica Dragón Azteca S. de R.L. de C.V. (EDITADA.ORG), All rights reserved.

the one expected from the documented (ideal) process. It is through the discovered model that the running process can be analyzed and areas for improvement can be identified.

Table 1. Example of a medical data event log.

Case ID	Timestamp	Activity	Cost	Resource	Patient
1	06-01-2021 15:20:15	Register	100	Pedro	Brenda
1	06-01-2021 15:22:02	Triage	50	Ana	Brenda
1	06-01-2021 15:25:43	Blood Test	800	Julio	Brenda
2	06-01-2021 15:43:08	Register	100	Jorge	Isidro
2	06-01-2021 15:43:50	X-Rays	500	Pedro	Isidro
3	07-01-2021 15:46:27	Register	100	Pedro	Marta
3	07-01-2021 15:48:14	Triage	50	Ana	Marta

Under a Bigdata context, the wide availability of data from multiple executions of a business process and the considerable size of some event logs motivates the use of external (cloud) services, not only for storage but also for executing process mining algorithms, thus leading to the concept of Process Mining as a Service (PMaaS) [11]. However, some event logs such as those in the healthcare domain (like the one presented in Table 1), may contain highly sensitive data such as patient, diagnosis, and treatment information. Therefore, an event log in possession of a non-trusted third party becomes a risk to the confidentiality and privacy of the organization's data. This makes evident that, under a PMaaS context, it is mandatory to guarantee confidentiality and privacy of event logs.

In the literature, several approaches [4, 9, 11] have been proposed to preserve confidentiality and privacy in an event log, such as data sharing, generalization or noise aggregation. In this work, we focused on the study, evaluation, and experimental evaluation of two cryptographic frameworks in the context of PMaaS, particularly under the process discovery task [13]. Evaluation is done using real data event logs [5, 6], that is, models are discovered using both the event log in clear and in encrypted form. The results revealed that there is no loss of utility in the resulting process models from encrypted event logs under three different security levels.

This article is an extension of the work reported in [21]. As new content, the related work is updated with recent references, the analysis of the design of the strategies is more detailed and the evaluation is more extensive using new data and measures.

This article is organized as follows. Section 2 presents some relevant PM concepts and cryptographic frameworks. Section 3 presents a discussion of related work. Section 4 describes relevant features of selected methods for discovery of process models using encrypted logs. Section 5 presents an evaluation of the two methods under study with real data. Finally, Section 6 concludes this work.

2 Background

Usually, process discovery algorithms start by transforming the event log data into a Directed Follows Graph (DFG) scheme [14], and output a model in a visual notation language, such as a Petri Net (PN) or Business Process Modeling Notation (BPMN). The discovered model should allow replicating each of the *traces* recorded in the event log, where a *trace* is the chronological sequence of events of a given case. Formally, an *event log* L is a set of *n traces*, $L = \{t_1, t_2, t_3, \dots t_n\}$, each trace t_i containing the sequence of events associated to the executed activities in the case. For example, the event log in Table 1 is represented as the set of traces, L = {< "*Register*", "*Triage*", "*Blood Test*" >, < "*Register*", "*X-Rays*" >, < "*Register*", "*Triage*" >}.

While nodes in the DFG represent process' activities, the arcs represent the relationships between activities. An arc (a,b) in the DFG indicates that activity b is executed after activity a. $(a,b)^k$ indicates that the arc (a,b) appears k times in the DFG, i.e., that relationship appears in more than one trace in the event log. For example, the event log in Table 1 is represented as DFG = {("*Register*", "*Triage*")^2,("*Register*","*X-Rays*")^1, ("*Triage*","*Blood Test*")^1} and graphically shown as a PN process model in Figure 1.

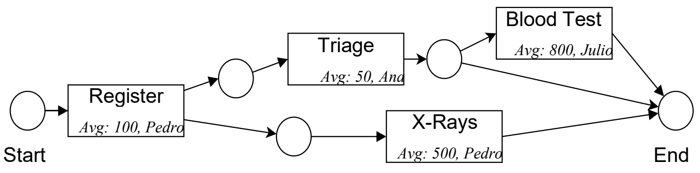


Figure 1. Process model associated to the event log in Table 1.

The Alpha algorithm [15] is one of the first algorithms for process model discovery. It takes an event log L as input and produces a process model M in PN. Using the event log $L = \{< a,b,e,f >, < a,b,e,c,d,b,f >, < a,b,c,e,d,b,f >, <a,b,c,d,e,b,f >, < a,e,b,c,d,b,f >\}$, as example, the Alpha algorithm proceeds to discover the process model through the following 8 steps:

1. Identifies the distinct set of activities (Δ) in $L \rightarrow \Delta = \{a, b, c, d, e, f\}$.
2. Discover the initial activities (Δ^I) on each $t_j \rightarrow \Delta^I = \{a\}$.
3. Discover the final activities (Δ^O) on each $t_j \rightarrow \Delta^O = \{f\}$.
4. Group activities using order relationships: causality (→), parallelism (||) and choice (#) to create the set of relations X_L. Where $X_L = \{ (\{a\},\{b\}),\ (\{a\},\{e\}),\ (\{b\},\{c\}),\ (\{b\},\{f\}),\ (\{c\},\{d\}),\ (\{d\},\{b\}),\ (\{e\},\{f\}),\ (\{a,d\},\{b\}),\ (\{b\},\{c,f\}) \}$.
5. Remove pairs of X_L to create an optimized set of relationships $Y_L = \{ (\{a\},\{e\}),\ (\{c\},\{d\}),\ (\{e\},\{f\}),\ (\{a,d\},\{b\}),\ (\{b\},\{c,f\}) \}$.
6. Determine the set of nodes (λ) and the start and end nodes (M^I, M^O) of the model. Where $\lambda = \{ p_{(\{a\},\{e\})},\ p_{(\{c\},\{d\})},\ p_{(\{e\},\{f\})},\ p_{(\{a,d\},\{b\})},\ p_{(\{b\},\{c,f\})},\ M^I,\ M^O \}$.
7. Nodes are connected λ through arches γ. Where $\gamma = \{ (M^I, a),\ (f, M^O),\ (a, p_{(\{a\},\{e\})}),\ (p_{(\{a\},\{e\})}, e),\ (c, p_{(\{c\},\{d\})}), \ldots,\ (p_{(\{b\},\{c,f\})}, c),\ (p_{(\{b\},\{c,f\})}, f) \}$.
8. Return $\alpha(L)$ as a PN, which is represented by the triad $(\lambda, \Delta, \gamma)$.

Cryptographic frameworks for event log privacy-preserving cause minimal loss of utility in process discovery algorithms [9]. Encryption is a function $\mathcal{E}$ that converts a *readable* message m into an *unreadable* ciphertext using an encryption key $k^{\mathcal{E}}$. The reverse process (decryption) can be carried out using a function $\mathcal{D}$ and decryption key $k^{\mathcal{D}}$. Encryption has already been used in process mining and several types of ciphers considered, such as *symmetric encryption* ($k^{\mathcal{E}} = k^{\mathcal{D}}$), *asymmetric encryption* ($k^{\mathcal{E}} = f(k^{\mathcal{D}})$), with f a one-way function), *deterministic encryption* ($\mathcal{E}(m_1, k_1^{\mathcal{E}}) = \mathcal{E}(m_2, k_2^{\mathcal{E}})$ if $m_1 = m_2$ and $k_1^{\mathcal{E}} = k_2^{\mathcal{E}}$), and *homomorphic encryption* (calculations on encrypted messages is possible). This last type of encryption is relevant in the sense that when the encrypted result is decrypted, it matches the result that would be obtained by performing the same operations on clear messages. Formally, in homomorphic encryption ($\mathcal{D}(\mathcal{E}(m_1, k_1^{\mathcal{E}}) \circledast \mathcal{E}(m_2, k_2^{\mathcal{E}}), k^{\mathcal{D}}) = m_1 * m_2$, being $*$ y $\circledast$ valid operations defined for clear data and encrypted data, respectively. In the context of PMaaS and for privacy of data in the clear, processing homomorphically encrypted data is essential. Some of the most relevant homomorphic encryption protocols are Secure Equality Checking (SEQ) [18] and Secure Multiplication Protocols (SMP) [19].

3 Related Work

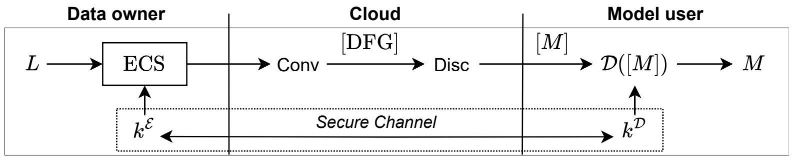
Different strategies [7, 9, 10] have been proposed to attack the privacy problem in the context of process mining. Cryptographic frameworks [2, 12, 11] manage to adopt encryption techniques for data confidentiality, so that context-related data is not leaked and can still be used in process mining algorithms without loss of utility, i.e., the results achieved by using protected event logs are the same than if using the original (unprotected) data. Burattin et al. [2] propose one of the first cryptographic frameworks, particularly using deterministic encryption for process mining. The framework takes advantage of the differentiation property of ciphertexts to still give results without loss of utility in process model discovery algorithms. Tillem et al. [12] proposed the use of homomorphic encryption to ensure semantic security of encrypted event log data and still discover a process model. Particularly, that strategy is based on the use of Alpha algorithm [13] using homomorphically encrypted data. Rafiei et al. [11] proposed a framework that uses deterministic encryption and guarantees security against frequency analysis. This type of encryption prepares the event log data for causality-based process discovery algorithms, i.e., those that make use of an initial DFG structure. More recent works such as [1, 20] focus on protecting event log data for process mining, relying on microaggregation or differential privacy, thus improving computational efficiency by balancing utility loss against data privacy.

4 Model Discovery from Encrypted Logs

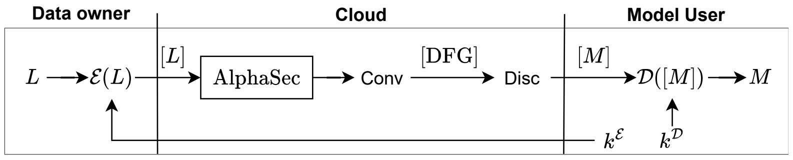
After a literature review, two strategies for process discovery from encrypted event logs were selected and studied in terms of data utility and performance: (1) *External Confidentiality Strategy* (ECS) [11] and (2) *AlphaSec Algorithm* [12]. Table 2 shows the notation used in this section to describe both strategies under three environments: (1) the data owner side, (2) the service provider (cloud) side, and (3) the discovered process models end-user side. Figure 2 shows an overview of the flow of operations in each of these strategies.

Table 2. Notation used in this work.

Notation	Description
e_i^j	Event where $e_i^j \in t_j$ and $i \in \mathbb{Z}$
t_j	Trace where $t_j \in L$ and $j \in \mathbb{Z}$
L	Multiset of traces, where $e_i^j \in t_j \in L$
Δ	Different activities in L
M	Process model
$[x]$	Encrypted x element
$[L]$	Encrypted L, where $[e_i^j] \in [t_j] \in [L]$
$\mathcal{E}$	Encryption function
$\mathcal{D}$	Decryption function
$k^{\mathcal{E}}$	Encryption key
$k^{\mathcal{D}}$	Decryption key



(a) External Confidentiality Strategy (ECS).



(b) AlphaSec algorithm.

Figure 2. ECS and AlphaSec strategy environments.

4.1 External Confidentiality Strategy (ECS)

The strategy presented by Rafiei et al. in [11] encrypts an event log through five stages: (1) Filtering, (2) Selection, (3) Encryption, (4) Conversion, and (5) Connector, all executed in a trusted environment. The encrypted log can then be outsourced to an untrusted environment for model discovery.

Stage 1. Filtering: Cases in the same trace are grouped, thus forming a multi-set of traces L. Trace filtering is performed $t_j \in L$, so that those of lower frequency are removed according to a set threshold (determined by the process context). The result is a filtered event log L^1.

Stage 2. Selection: Attributes that are irrelevant to the process mining analysis or data that are not sensitive are removed from L^1. This produces L^2 as a new event log version.

Stage 3. Encryption: Numerical values in the event log are coded with Paillier [8]. While for text data, a symmetric deterministic AES encryption is applied [3]. This step is illustrated in the "Activity", "Resource" and "Cost" attributes of the clear event log in Table 1 and the encrypted log ($[L]$) in Table 3. The result of this step is an encrypted event log L^3.

Stage 4. Conversion: A new (secret) date is selected. The timestamp of each event $e_i^j \in t_j \in L^3$ is replaced by its difference with the selected secret date. This stage is illustrated in the attribute "*Timestamp*" of Tables 1 and 3. The result is a new version of the event log, L^4. This prevents the time of the dates in the log from being identified.

Stage 5. Connector: Allows to extract a DFG from the already encrypted event log and reconstruct L in its original form. The above only occurs if $k^{\mathcal{D}}$ and relative date values are known.

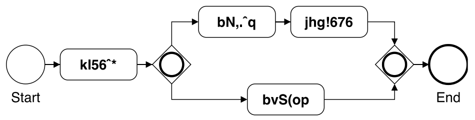
1. The attribute indicating the previous activity is added. ("*Prev. Activity*") to each event $e_i^j \in t_i \in L^4$ to identify which activities are directly connected. The first event of each trace $t_j \in L^4$ will have a previous artificial activity "Start" encrypted with the same parameters with which the "*Activity*" attribute was encrypted.
2. The information contained in "*Connector*" allows the grouping of events in traces, keeping the possibility to recreate a L in its original form. This is done by giving to each event a random event ID ("*ID*") and indicate the previous ID ("*Prev. ID*"). These new log attributes make it possible to identify the next event at the trace level. The "*Prev. ID*" of the initial event in each $t_j \in L^4$ will always be 0.
3. Next, the attributes "*ID*" and "*Prev. ID*" are hidden, as they are used only to rebuild L. These are concatenated and encrypted using the deterministic encryptor AES (as in Stage 3) and "*Connector*" is added as a new attribute.
4. Next, the "*Timestamp*" attribute is used to protect the time values in events with the same "*Case ID*". These are made relative to the preceding timestamp. Except for the first event of each trace, which is kept the same value. This allows the calculation of the duration in each of the arcs (as Activities) in a DFG, but makes difficult to identify events based on the time interval in which they occurred.
5. Finally, the "*Case ID*" attribute is removed, and the order of all rows is disorganized. The result is an event log L^5 protected by the five stages of ECS (see Table 3) to be outsourced to the cloud, converted to a DFG structure (generated based on "*Activity*" and "*Pre-activity*") and used in causality-based business process model discovery algorithms.

Table 3. Event log after implementation of the ECS strategy.

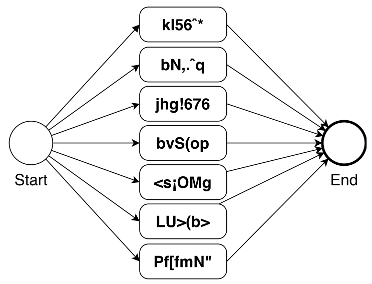
Timestamp	Activity	Prev. Activity	Resource	Cost	Connector
00-00-0000 00:00:42	y7y4PUi2	NM7Jgoum	XRCDyLgS	59301	5q8aL2at
00-00-0000 00:01:47	UGdnk8fh	y7y4PUi2	hLrq2mYD	46012	KQBindVr
00-00-0000 15:46:27	bvS(28op	UGdnk8fh	4hIDYn0q	98744	CKl07FSq
00-00-0000 00:01:47	y7y4PUi2	NM7Jgoum	tpwUTcAl	58430	N9a1qeto
00-00-0000 00:03:41	jhg!676	y7y4PUi2	XRCDyLgS	81023	XIQ7ZnqA
00-00-0000 15:20:15	y7y4PUi2	NM7Jgoum	XRCDyLgS	59015	M4qAwqqz
00-00-0000 15:43:08	UGdnk8fh	y7y4PUi2	hLrq2mYD	42110	z5Zb56jY

4.2 Strategy Based on Alpha Algorithm (AlphaSec)

The AlphaSec protocol works with homomorphically encrypted event log data, at the data owner side. Particularly, AlphaSec uses Paillier, a non-deterministic homomorphic asymmetric cryptosystem. By itself, this encrypted log is not useful for PMaaS because the activities encrypted with this type of encryption lose the differentiation property. Figure 3 shows an example of the result that conventional discovery algorithms with homomorphically encrypted or nondeterministic event logs would have. AlphaSec solves this problem by adapting the basics of a conventional process discovery algorithm with homomorphic encryption protocols in order to discover the model.



(a) Process model from deterministic encryption.



(b) Process model from non-deterministic encryption.

Figure 3. Discovery of process models from encrypted data.

AlphaSec concentrates on adapting the first 4 steps of the original Alpha algorithm [15]. AlphaSec consists of discovering Δ, Δ^I and Δ^O from the encrypted domain and find the relationships (X_L, in Alpha's description). As shown in Algorithm 1, the AlphaSec protocol is composed of 3 subprotocols: (1) *Secure activity discovery* (SP1), where encrypted activities are discovered; (2) *Secure direct succession discovery* (SP2), where the activities' orders are determined; y (3) *Secure Modelling* (SP3), where the Zero Check Matrix (ZCM) is generated to be converted to a DFG structure. The following is a description of the Alphasec subprotocols using the example event log $[L] = \{<[a],[b],[c]>^1, <[a],[d]>^1, <[a],[b]>^1\}$.

Algorithm 1. AlphaSec

```
Input: $[L]$, $\Delta$                                    ▷ Activities are identified
Output: ZCM
 1: for all $[t_i] \in [L]$ do
 2:     $([AD^{t_i}], [\Delta^I], [\Delta^O]) = $ SP1$([t_i])$
 3:     $[R] = $ SP2$([AD^{t_i}])$                        ▷ Direct successions are identified
 4: end for all                                           ▷ Relational matrix ZCM is obtained
 5: ZCM = SP3$([R], [\Delta^I], [\Delta^O])$
 6: DFG = Conv(ZCM)
```

Secure activity discovery (SP1): It is aimed at safely discovering the activities $[\Delta]$, $[\Delta^I]$ and $[\Delta^O]$. As a result, the service provider collaborates with the data owner to compare each $[e_i^j]$ with each $[\Delta_m]$ (where $m = |\Delta|$) using a SEQ protocol. If $[e_i^j] = [\Delta_m]$, $AD_{m,n}^{t_j}$ is established in "[1]" (encrypted "1"), otherwise, it is set to "[0]". (encrypted "0"). As shown in Equation 1, this is done for each $[e_i^j]$ of each $[t_j]$ in $[L]$.

$$[\mathrm{AD}^{t_j}] = \left\{ \begin{array}{cccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] \\ [\boldsymbol{a}] & [1] & [0] & [0] \\ [\boldsymbol{b}] & [0] & [1] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [1] \\ [\boldsymbol{d}] & [0] & [0] & [0] \end{array}, \begin{array}{ccc} & [\boldsymbol{a}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [1] & [0] \\ [\boldsymbol{b}] & [0] & [0] \\ [\boldsymbol{c}] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [1] \end{array}, \begin{array}{ccc} & [\boldsymbol{a}] & [\boldsymbol{b}] \\ [\boldsymbol{a}] & [1] & [0] \\ [\boldsymbol{b}] & [0] & [1] \\ [\boldsymbol{c}] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] \end{array} \right\}$$
$$[\mathrm{AD}^{t_j}] = \{[\mathrm{AD}^{t_1}], [\mathrm{AD}^{t_2}], [\mathrm{AD}^{t_3}]\} \quad \textbf{(1)}$$

Secure direct succession discovery (SP2): To detect subsequent events between two events $[e_i^j]$ and $[e_{i+1}^j]$, two subsequent columns of the matrix are merged $[\mathrm{AD}^{t_j}]$ through a SMP protocol. As shown in Equation 2, each item in the previous column, $[\mathrm{AD}_{*,n}^{t_j}]$, is safely multiplied with each element of the transpose of the subsequent column $[\mathrm{AD}_{*,n+1}^{t_j}]^T$. Then, the result is added to the corresponding index of a new $[R]$ matrix, as shown in Equation 3.

$$\begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [1] & [0] & [0] \\ [\boldsymbol{b}] & [0] & [0] & [1] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} = \begin{array}{cc} & [\boldsymbol{a}] \\ [\boldsymbol{a}] & [1] \\ [\boldsymbol{b}] & [0] \\ [\boldsymbol{c}] & [0] \\ [\boldsymbol{d}] & [0] \end{array} \times \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{b}] & [0] & [1] & [0] & [0] \end{array} + \begin{array}{cc} & [\boldsymbol{a}] \\ [\boldsymbol{a}] & [0] \\ [\boldsymbol{b}] & [1] \\ [\boldsymbol{c}] & [0] \\ [\boldsymbol{d}] & [0] \end{array} \times \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{c}] & [0] & [0] & [1] & [0] \end{array}$$
$$[R^{\mathrm{AD}^{t_1}}] = [\mathrm{AD}_{*,1}^{t_1}] \times [\mathrm{AD}_{*,2}^{t_1}]^T + [\mathrm{AD}_{*,2}^{t_1}] \times [\mathrm{AD}_{*,3}^{t_1}]^T \quad \textbf{(2.1)}$$

$$\begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [0] & [0] & [1] \\ [\boldsymbol{b}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} = \begin{array}{cc} & [\boldsymbol{a}] \\ [\boldsymbol{a}] & [1] \\ [\boldsymbol{b}] & [0] \\ [\boldsymbol{c}] & [0] \\ [\boldsymbol{d}] & [0] \end{array} \times \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{b}] & [0] & [0] & [0] & [1] \end{array} \quad \textbf{(2.2)}$$

$$[R^{\mathrm{AD}^{t_2}}] = [\mathrm{AD}^{t_2}_{*,1}] \times [\mathrm{AD}^{t_2}_{*,2}]^T$$

$$\begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [1] & [0] & [0] \\ [\boldsymbol{b}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} = \begin{array}{cc} & [\boldsymbol{a}] \\ [\boldsymbol{a}] & [1] \\ [\boldsymbol{b}] & [0] \\ [\boldsymbol{c}] & [0] \\ [\boldsymbol{d}] & [0] \end{array} \times \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{b}] & [0] & [1] & [0] & [0] \end{array} \tag{2.3}$$

$$[R^{\mathrm{AD}^{t_3}}] = [\mathrm{AD}^{t_3}_{*,1}] \times [\mathrm{AD}^{t_3}_{*,2}]^T$$

$$\begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [2] & [0] & [1] \\ [\boldsymbol{b}] & [0] & [0] & [1] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} = \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [1] & [0] & [0] \\ [\boldsymbol{b}] & [0] & [0] & [1] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} + \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [0] & [0] & [1] \\ [\boldsymbol{b}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} + \begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & [0] & [1] & [0] & [0] \\ [\boldsymbol{b}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{c}] & [0] & [0] & [0] & [0] \\ [\boldsymbol{d}] & [0] & [0] & [0] & [0] \end{array} \tag{3}$$

$$[R] = [R^{\mathrm{AD}^{t_1}}] + [R^{\mathrm{AD}^{t_2}}] + [R^{\mathrm{AD}^{t_3}}]$$

Secure Modelling (SP3): The cloud queries each index of the $[R]$ matrix to the owner to obtain a "0" (unencrypted) if the decrypted data is 0, or "1" otherwise. The ZCM matrix of the example is shown in Equation 4.

$$\begin{array}{ccccc} & [\boldsymbol{a}] & [\boldsymbol{b}] & [\boldsymbol{c}] & [\boldsymbol{d}] \\ [\boldsymbol{a}] & 0 & 1 & 0 & 1 \\ [\boldsymbol{b}] & 0 & 0 & 1 & 0 \\ [\boldsymbol{c}] & 0 & 0 & 0 & 0 \\ [\boldsymbol{d}] & 0 & 0 & 0 & 0 \end{array} \tag{4}$$

$$ZCM$$

5 Evaluation and Discussion

The two strategies previously discussed build an encrypted DFG structure capable of being used by most (DFG-based) process model discovery algorithms without losing data utility. However, their workflows differ, mainly in the amount of computations assigned to each environment, which leads to different performance. In this section, we present an evaluation and comparison in workflow, performance, and utility loss for the ECS and AlphaSec strategies.

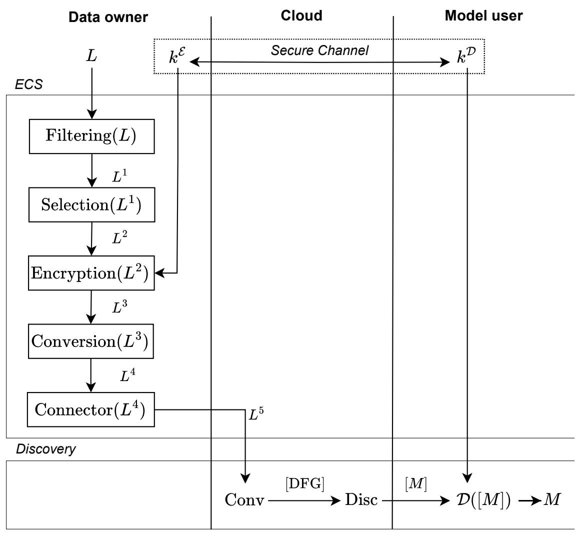
5.1 Methodology

The methodology used to carry out the study proposed in this article consisted of:

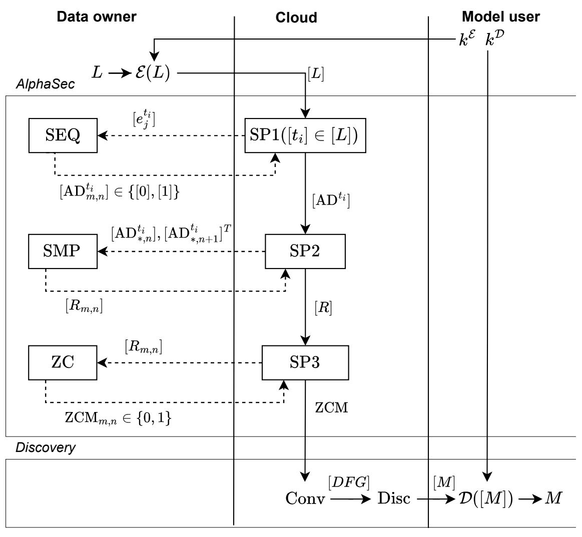
1. Study and selection of business process model discovery methods based on cryptographic algorithms.
2. Selection of evaluation parameters and experimental design.
3. Implementation of the selected methods under the same conditions (programming language and validation and evaluation tools).
4. Collection, analysis, and reporting of results.

5.2 Workflow

Each strategy devotes different numbers of operations to each of the three environments. As shown in Figure 4, the ECS strategy performs all its work from the data owner's environment, while the AlphaSec strategy performs most of its work in the cloud environment but keeps interactions with the data owner. In both cases the cloud is responsible for converting (Conv) the output of both strategies into an encrypted DFG structure ([DFG]). This is easily achieved since both outputs ($[L]^4$ and ZCM) provide the same information in different format. In both strategies the cloud handles the discovery (Disc) of the encrypted model ([M]) using [DFG], as described in Figure 4 in the *Discovery* section box.



(a) ECS



(b) AlphaSec

Figure 4. Workflow performed by each strategy.

5.3 Performance

The performance of the ECS and AlphaSec strategies was done under the same conditions: same implementation programming language and same computing platform, Java and MacOS (M1 Pro, 16 GB), respectively. The three PMaaS scenario environments (data owner, cloud and model user) were deployed from the same computer, omitting data transfer times for this evaluation. The strategies were configured with the same equivalent security level: 128-bit AES encryption for ECS and 3072-bit Paillier encryption for AlphaSec. The actual medical data event logs described in Table 4 were used as input.
The ECS strategy uses deterministic encryption, so there is no need for the owner to encrypt each of the activities, since the ciphertext will always be the same. On the other hand, AlphaSec must encrypt each of the activities, even if they are repeated, because its semantic security relies on the ciphertexts being different. Figure 5 shows the performance of each strategy to protect each of the logs. The ECS strategy can be affected by increasing the number of traces, while AlphaSec is notably affected (exponential increase) by increasing the number of activities.

Table 4. Event logs of medical data.

Event log	Events	Activities	Traces	Size (MB)
Sepsis Cases [5]	15,214	16	1,050	5.4
Hospital Billing [6]	451,359	18	100,000	174.3
BPIC11 [16]	150,291	624	1,143	85.4

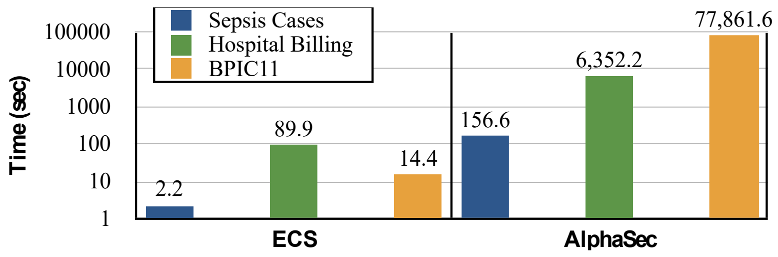


Figure 5. Performance of each strategy.

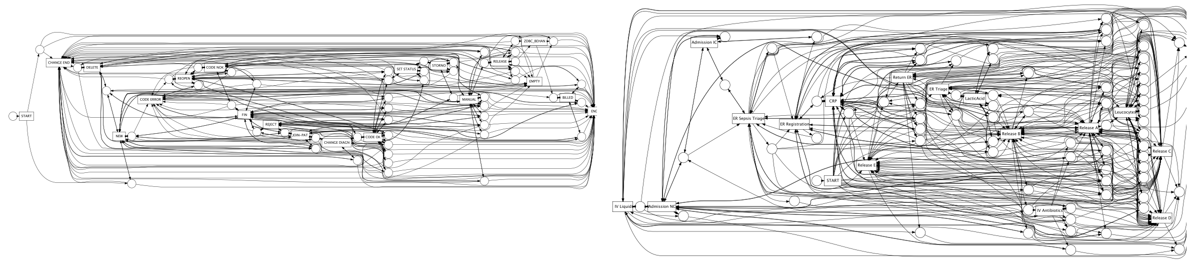
5.4 Utility Loss

The utility loss was assessed using the real medical data event logs described in Table 4 and their versions protected by the ECS and AlphaSec strategies in two process model discovery algorithms: (1) Alpha [15] and (2) Inductive Miner [17]. To perform the evaluation, the conformance of each discovered model was tested by calculating its *fitness* and *precision* measures to compare the corresponding models of the same log (protected and unprotected).

Figure 6 presents the decrypted models discovered with the *Alpha* algorithm [15] using the Sepsis Cases and Hospital Billing protected logs. Similarly, Figure 7 presents the decrypted models discovered with the *Inductive Miner* algorithm [17]. Table 5 shows a cumulative evaluation of the resulting models, where the results between models of the same log are the same, i.e., they do not change in terms of utility loss with respect to what would be obtained with the unprotected log.

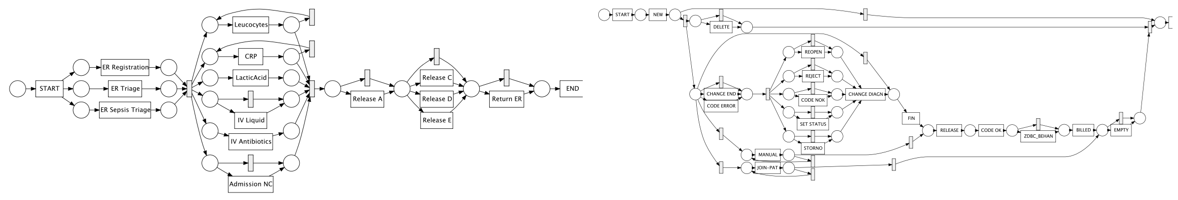
Table 5. Evaluation of the discovered models ('-': unprotected).

Discovery	Event Log	Protection	Arcs	Places	Transitions	Density	Fitness	Precision
Alpha Miner	Sepsis Cases	-	451	70	18	0.178	1.0	#
		ECS	451	70	18	0.178	1.0	#
		AlphaSec	451	70	18	0.178	1.0	#
	Hospital Billing	-	365	53	20	0.167	#	#
		ECS	365	53	20	0.167	#	#
		AlphaSec	365	53	20	0.167	#	#
Inductive Miner	Sepsis Cases	-	64	24	25	0.053	1.0	0.437
		ECS	64	24	25	0.053	1.0	0.437
		AlphaSec	64	24	25	0.053	1.0	0.437
	Hospital Billing	-	90	30	40	0.037	0.825	0.677
		ECS	90	30	40	0.037	0.825	0.677
		AlphaSec	90	30	40	0.037	0.825	0.677



(a) *Sepsis Cases.* (b) *Hospital Billing*.

Figure 6. PN models discovered with the (original) *Alpha* algorithm from encrypted event logs.



(a) *Sepsis Cases.* (b) *Hospital Billing*.

Figure 7. PN model discovered with the *Inductive Miner* algorithm from encrypted event logs.

6 Conclusion

Privacy preserving in process mining is becoming a crucial requirement in the context of data science as a service. This approach is more common in the presence of a Big Data era and the continuing adoption of cloud computing. A data owner in possession of large event log could find huge benefits to outsource the process mining tasks, such as process model discovery. In this work we provided an assessment of suitable privacy preserving process discovery methods based on cryptographic principles. A such approach resulted to be attractive because no loss of utility is exhibited at the time that privacy over sensitive data in the event log is preserved. The methods were identified from the state of the art, studied and assessed under the same conditions. The first method (ECS) uses deterministic encryption and the second one (AlphaSec) relies on homomorphic encryption. Experimental results revealed that both strategies can come to the same results in the form of a DFG model usable in conventional process model discovery algorithms. While ECS takes advantage of the cloud to perform noticeably fewer computations, AlphaSec provides semantic security to the log, which prevents the cloud from being able to learn information through frequency analysis, thus providing a higher security. It is planned to extend the study presented in this work consider quantum safe encryption techniques to face the currently quantum threat to encryption algorithms to consider.

References

1. Batista, E., Martínez, A., & Solanas, A. (2022). Privacy-preserving process mining: A microaggregation-based approach. *Journal of Information Security and Applications, 68*, 103235. https://doi.org/10.1016/j.jisa.2022.103235.
2. Burattin, A., Conti, M., & Turato, D. (2015). Toward an Anonymous Process Mining. En *Proceedings of 2015 International Conference on Future Internet of Things and Cloud* (pp. 58–63). FiCloud. https://doi.org/10.1109/FiCloud.2015.58
3. Daemen, J., & Rijmen, V. (2002). *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Berlin, Heidelberg.
4. Elkoumy, G., Fahrenkrog-Petersen, S., Fani Sani, M., Koschmider, A., Mannhardt, F., Nuñez von Voigt, S., Rafiei, M., & von Waldthausen, L. (2021). Privacy and confidentiality in process mining: Threats and research challenges. *ACM Transactions on Management Information Systems*. https://doi.org/10.1145/3464757
5. Mannhardt, F. (2016). Sepsis Cases - Event Log. Retrieved from https://data.4tu.nl/articles/dataset/Sepsis_Cases_-_Event_Log/12707639/1 (Consulted on 20-06-2023).
6. Mannhardt, F. (2017). Hospital Billing - Event Log. Retrieved from https://data.4tu.nl/articles/dataset/Hospital_Billing_-_Event_Log/12705113/1 (Consulted on 20-06-2023).
7. Mannhardt, F., Koschmider, A., Baracaldo, N., Weidlich, M., & Michael, J. (2019). Privacy-preserving process mining. *Business and Information Systems Engineering, 61*, 595–614. https://doi.org/10.1007/s12599-019-00599-0.
8. Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. En J. Stern (Ed.), *Advances in Cryptology - EUROCRYPT '99* (pp. 223–238). Springer.
9. Pika, A., Wynn, M., Budiono, S., Hofstede, A., van der Aalst, W., & Reijers, H. (2020). Privacy-preserving process mining in healthcare. *Journal of Environmental Research and Public Health, 17*. https://doi.org/10.3390/ijerph17010123.
10. Rafiei, M., & van der Aalst, W. M. (2021). Group-based privacy preservation techniques for process mining. *Data Knowledge Engineering*. https://doi.org/10.48550/arXiv.2105.11983
11. Rafiei, M., Waldthausen, L. v., & van der Aalst, W. (2020). Supporting confidentiality in process mining using abstraction and encryption. *8th International Symposium on Data-Driven Process Discovery and Analysis* (pp. 101–123). Springer. https://doi.org/10.1007/978-3-030-61703-7_7
12. Tillem, G., Zekeriya, E., & Lagendijk, E. (2017). Mining encrypted software logs using alpha algorithm. *SECRYPT. SciTePress*, 267–274.
13. Van der Aalst, W., Weijters, T., & Maruster, L. (2004). Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering, 16*, 1128–1142. https://doi.org/10.1109/TKDE.2004.47.
14. Van der Aalst, W. M. (2019). A practitioner's guide to process mining: Limitations of the directly-follows graph. *Procedia Computer Science, 164*, 321–328. https://doi.org/10.1016/j.procs.2019.12.209
15. Van der Aalst, W. M. P. (2016). *Process Mining: Data Science in Action*. Springer. https://doi.org/10.1007/978-3-662-49851-4
16. Van Dongen, B. (2011). Real-life event logs - Hospital log. Retrieved from https://data.4tu.nl/articles/dataset/Real-life_event_logs-Hospital_log/12716513/1 (Consulted on 20-03-2022).
17. Leemans, S. J. J., Fahland, D., & van der Aalst, W. M. P. (2013). Discovering block-structured process models from event logs. En J. M. Colom & J. Desel (Eds.), *Application and Theory of Petri Nets and Concurrency* (pp. 311–329). Springer. https://doi.org/10.1007/978-3-642-38697-8_17.
18. Nateghizad, M., Erkin, Z., & Lagendijk, R. L. (2016). Efficient and secure equality tests. *IEEE International Workshop on Information Forensics and Security (WIFS)*. Abu Dhabi, United Arab Emirates. https://doi.org/10.1109/WIFS.2016.7823921.
19. Erkin, Z., Veugen, T., Toft, T., & Lagendijk, R. L. (2012). Generating private recommendations efficiently using homomorphic encryption and data packing. *IEEE Transactions on Information Forensics and Security, 7*, 1053–1066. https://doi.org/10.1109/TIFS.2012.2199612.
20. Elkoumy, G., Pankova, A., & Dumas, M. (2022). Differentially private release of event logs for process mining. *arXiv*. https://doi.org/10.48550/arXiv.2201.03010.
21. De la Fuente-Anaya, H. A., Marin-Castro, H. M., Morales-Sandoval, M., & Garcia-Hernandez, J. J. (2023). Descubrimiento de modelos de procesos de negocios a partir de bitácoras de eventos cifradas. *Abstraction and Application, 43*, 12–21.